# Beyond Tokens: A Survey on Decoding Methods for Large Language and Vision-Language Models

Haoran Wang[1], Xiongxiao Xu[2], Philip S. Yu[3], Kai Shu[1]
[1] Emory University, [2] Illinois Institute of Technology, [3] University of Illinois Chicago
haoran.wang@emory.edu, xxu85@hawk.illinoistech.edu, psyu@uic.edu, kai.shu@emory.edu

## ABSTRACT

Large language models (LLMs) and large vision-language models (LVLMs) have demonstrated impressive generative capabilities, yet ensuring their outputs align with user intent is still challenging. While most existing approaches address this issue at the training stage, inference-time approaches like decoding methods offer a more efficient and scalable solution. Decoding methods control model generation by guiding token-level selection, performing sequence-level generation, or generating tokens in parallel to accelerate the process. In this survey, we identify three emerging paradigms from recent works on decoding methods for LLMs and LVLMs, provide a systematic review of these methods, highlight ongoing challenges, and discuss potential future research directions. Our goal is to underscore the efficiency and effectiveness of decoding methods and offer a practical view of their applications. Paper lists and more resources on decoding methods for LLMs and LVLMs can be found at https://github.com/wang2226/Awesome-LLM-Decoding.

## 1. INTRODUCTION

Large language models (LLMs) and large language-vision models (LVLMs) have demonstrated that scaling up both model size and training datasets can greatly improve the model's generative capabilities. However, with the rise of massive foundational models such as Llama3 405B [87] and Megatron [118], which boasts 530 billion parameters, the focus has increasingly shifted toward developing *more efficient and scalable approaches* to control generation during inference time (§ 2). One popular approach is prompt engineering, due to its simplicity and effectiveness. However, it is highly task-specific, requiring human expertise to craft optimal prompts, and is susceptible to prompt sensitivity issues [109; 83], where minor variations in prompt wording can lead to significant performance differences. Other techniques, such as ROME [85] and ITI [64], modify model internals at inference time but suffer from limited generalizability and scalability. Recently, there has been increasing interest in decoding methods, which play a critical role in controlling next-token prediction and text generation.

Decoding methods have a rich history in language modeling, ranging from greedy decoding and beam search to sampling-based approaches like top-$p$ sampling [43]. These methods transform the vector representations produced by the model into coherent text while controlling the quality and attributes of the generated output. Recent works have shown that adopting more advanced decoding strategies can effectively *mitigate hallucination* [20; 148; 185; 144; 33], *improve safety* [73; 182; 161], *enhance visual grounding* [25; 59], *improve reasoning* [159; 186], and *increase robustness against noisy context* [116; 55; 104]. Beyond improving text generation, decoding methods can also *provide interpretability of the models*. For instance, [149] showed that modifying the decoding process can elicit chain-of-thought reasoning paths from LLMs. Finally, a recent line of research [119; 155; 150; 61; 15] has focused on *improving the efficiency of LLM and LVLM generation* by decoding multiple tokens simultaneously to accelerate generation.

In this survey, we use decoding to denote inference-time procedures that transform model output distributions into output sequences, including token selection, search strategies, and parallel generation. While decoding is operationally part of generation in autoregressive models, we distinguish it from training-time alignment and prompt engineering, and focus specifically on inference-time control mechanisms.

This survey provides a comprehensive overview of advanced decoding methods for LLMs and LVLMs, highlighting their capabilities, applications, and potential. We first review inference-time generation control methods (§ 2) and classical decoding strategies (§ 3). We then introduce three modern decoding paradigms (§ 4), followed by their applications (§ 5). Finally, we discuss open challenges and future directions (§ 6). Figure 1 presents a typology of key concepts related to decoding methods in LLMs and LVLMs. Although we cover decoding methods for both LLMs and LVLMs, the current literature is substantially richer for text-based LLM decoding. Accordingly, this survey primarily emphasizes text decoding, while visual, video, and code decoding are discussed as emerging extensions.

**Major Paradigm Shift in Language Modeling** As highlighted by [75], the field of language modeling and its related tasks has undergone several paradigm shifts. Initially, there was a shift from fully supervised learning to the *pre-train and fine-tune* approach [105; 99; 30], largely driven by the success of pre-trained LMs like BERT [27]. More recently, there has been another major shift toward the *pre-train, prompt, and predict* paradigm, where prompt engineering [106; 7; 107] has become a popular approach for adapting LMs to downstream tasks through carefully designed prompts.

However, both fine-tuning and prompt engineering face challenges when applied to LLMs and LVLMs. Fine-tuning models with tens of billions of parameters requires substan-

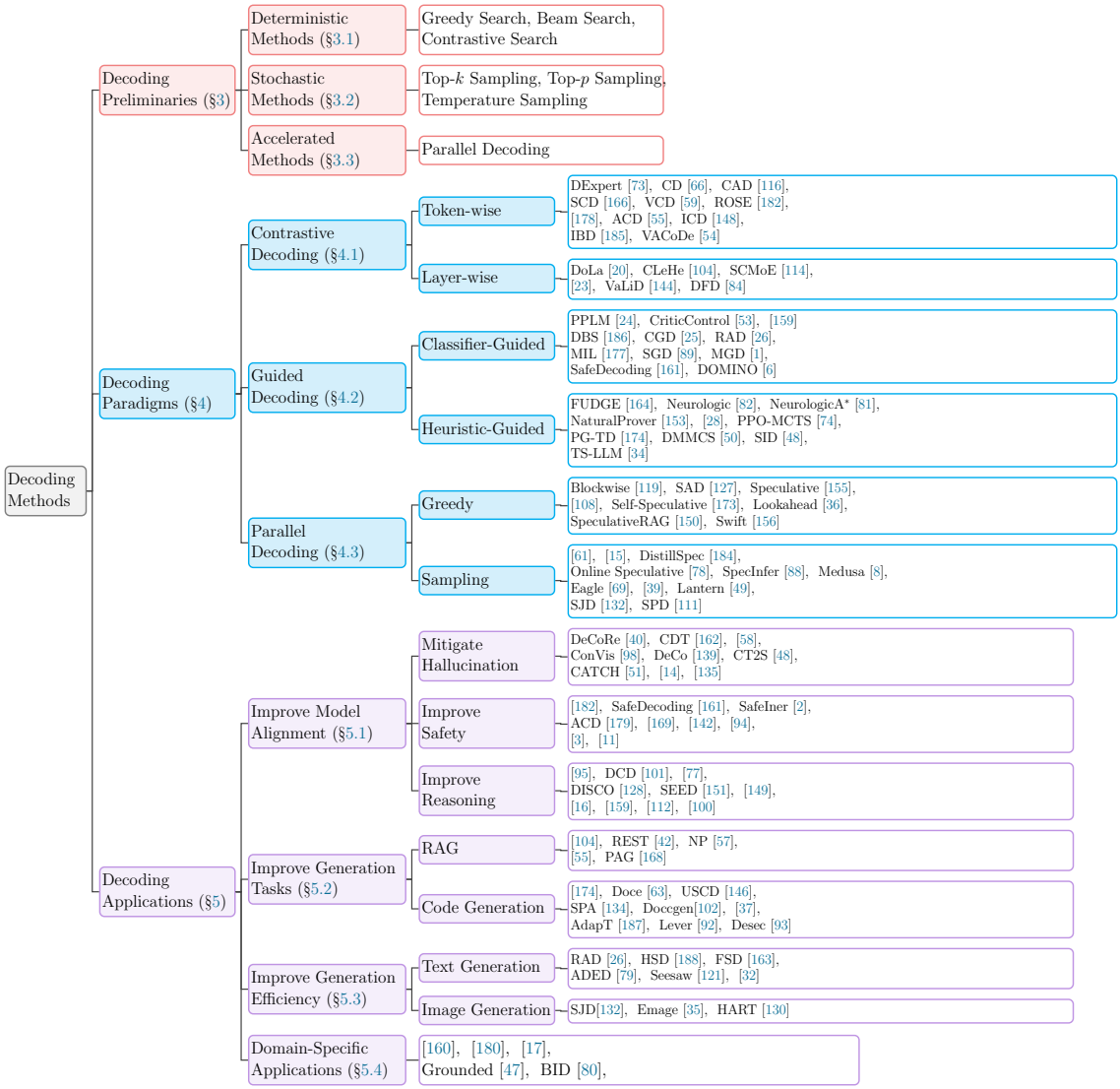


Figure 1: Typology of decoding methods for LLMs and LVLMs.

tial computational resources, raising concerns about energy consumption, scalability, and accessibility. While prompt engineering is computationally efficient, it remains highly task-specific and requires expert knowledge to craft effective prompts. Moreover, LLMs have been shown to have prompt sensitivity issues [109; 83], where even slight changes in prompt format can result in considerable variations in performance. Recently, a growing body of work has focused on advanced decoding methods [29; 154]. Figure 2 illustrates the evolution of decoding methods, highlighting key milestones from classical search to modern contrastive and speculative paradigms. We argue that the field is undergoing another significant paradigm shift toward universal, effective, and scalable methods for controlling LLM generation [70].

## 2. INFERENCE METHODS TO IMPROVE GENERATION

LLMs and LVLMs have demonstrated incredible performance across a wide range of NLP tasks. However, bridging the gap between their training objectives and user expectations remains challenging. While LLMs are trained to minimize contextual word prediction errors using large datasets, users expect the models to follow instructions in a helpful and safe manner. This highlights the need for alignment [113; 125] to ensure that models behave in ways that align with human values. To achieve controllable text generation and generate aligned outputs, researchers have proposed various methods, which can be broadly categorized into training-stage and inference-stage approaches, as outlined by [70]. Training-stage methods include fine-tuning [167; 172; 183; 171; 181; 152] and reinforcement learning [120; 96; 22; 52; 137; 170], which leverage reward signals to guide model outputs toward specific control objectives.

In this section, we focus on methods that improve generation during the inference stage for three key reasons. First, inference-time methods enable real-time improvements without the need for re-training or altering the underlying model, making them an *efficient approach* for controlling genera-

tion across models of any size. Second, these methods are typically *model-agnostic*, meaning they can be applied to most decoder-only transformers, thus enhancing their versatility. Finally, some inference-time techniques like decoding methods offer better *interpretability*. We categorize these inference-time approaches into three categories: *(1) prompting*, *(2) latent space manipulation*, and *(3) decoding algorithms*.

## 2.1 Prompt Engineering

Prompt engineering directly controls text generation by crafting specific prompts for the task. The primary goal of this approach is to guide model outputs by providing clear instructions or examples. [117] introduced AutoPrompt, an automated method for creating prompts across a wide range of tasks using a gradient-guided search approach. To provide a lightweight alternative to fine-tuning, [67] proposed prefix-tuning, which optimizes a sequence of continuous, task-specific vectors known as the "prefix" for natural language generation tasks. Additionally, [60] introduced prompt tuning, a straightforward yet effective technique for learning soft prompts that condition frozen language models to perform specific downstream tasks. More recently, [72] proposed Direct Large Model Alignment (DLMA), an automatic alignment method that generates preference data using contrastive prompt pairs, calculates a self-rewarding score, and applies the DPO algorithm to align LLMs. Black-Box Prompt Optimization (BPO) [18] optimizes user prompts to align with LLMs' input understanding, achieving the user's intent without modifying model parameters. By leveraging human preferences, BPO outperforms traditional prompt engineering in aligning LLMs with user goals. Moving beyond text, [62] proposed ALPRO, a video-text pre-training framework that aligns features without explicit object detectors.

## 2.2 Latent Space Manipulation

Latent space manipulation modifies the model's internal structure for controlled generation, such as attention heads or adding steering vectors to the activation layers. The core idea is that the necessary information to generate the target output is already encoded within the model's structure, eliminating the need for re-training or fine-tuning. By operating directly on the latent space, these techniques enhance output accuracy, diversity, and coherence, while remaining computationally efficient.

[12] introduced GENhance, a generative framework that enhances attributes through a learned latent space. Additionally, [124] extracts latent vectors, called steering vectors, directly from PLM decoders without fine-tuning. When added to the model's hidden states, these vectors allow for controlled generation. Extending steering vectors to LLMs, [136] introduced activation engineering, a method for modifying activations during inference to steer model outputs. This approach can also be utilized to control alignment [10; 145; 9; 141; 103].

To effectively control in-context learning, [76] proposed a method that first creates the in-context vector from the latent embedding of the LLM, and then shifts the latent states of the LLM using these vectors to more effectively follow the demonstration examples. Furthermore, [56] explores strategies to steer LLM outputs toward specific styles, such as sentiment, emotion, or writing style, by incorporating style vectors into the activations of hidden layers during text generation.

On a separate line of work, [85] analyzed the storage and recall of factual associations in autoregressive transformer language models, finding that these associations correspond to localized, directly editable computations. They modify feed-forward weights using Rank-One Model Editing (ROME) to alter factual associations within LLMs. Subsequently, [86] developed MEMIT, a method for directly updating a language model with many memories, demonstrating its ability to scale to thousands of associations for LLMs. More recently, [64] introduced Inference-Time Intervention (ITI) to enhance the truthfulness of LLMs. Specifically, it shifts model activations during inference, guided by a set of directions across a limited number of attention heads.

## 2.3 Decoding Algorithm

Decoding algorithms are applied during the decoding phase of transformer-based generative models to modify the logits or probability distribution of the model's output. It guides the generated text toward desired attributes by adjusting these probabilities, offering dynamic control over the text generation process, and ensuring the output aligns with specific requirements. Methods such as temperature scaling, top-$k$ sampling, and nucleus sampling can be used to influence the diversity, creativity, or coherence of the generated text by altering the probability distribution.

# 3. PRELIMINARIES OF DECODING STRATEGIES

This section reviews classical token-level decoding strategies such as greedy search, beam search, and sampling, which form the foundational building blocks for modern decoding approaches. We define key concepts and outline the general objectives of text generation. **Auto-regressive Language Generation** operates by predicting the next token in the sequence based on the preceding tokens. Formally, considering a sequence of tokens $w = (w_1, w_2, ..., w_t)$, the probability distribution of a word sequence can be decomposed into the product of conditional next word distributions:

$$P(w_{1:T}|W_0) = \prod_{t=1}^{T} P(w_t|w_{1:t-1}, W_0)$$

with $W_0$ being the initial context word sequence. The length $T$ of the word sequence is usually determined on-the-fly and corresponds to the timestamp $t = T$ the EOS token is generated from $P(w_t|w_{1:t-1}, W_0)$. The decoding problem is equivalent to selecting the most probable sequence given the probability distribution.

## 3.1 Deterministic Methods

Deterministic methods generate text by selecting the continuation with the highest probability determined by the LM. However, these methods often lead to model degeneration, where the output becomes unnatural, marked by repetitive and overly predictable language. As a result, the text lacks variety and fails to reflect natural human expression.

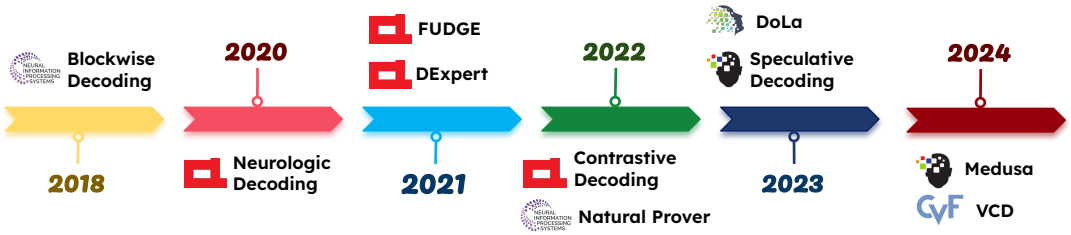


Figure 2: Timeline illustrating the evolution of decoding methods for LLMs and LVLMs.

#### 3.1.1 *Greedy Search*

Greedy search selects the word with the highest probability as its next word $\text{argmax}_w P(w|w_{1:t-1})$ at each timestep $t$ until reaching either an end-of-sequence (EOS) token or a maximum timestep $T$. The primary drawback of greedy search is that it may overlook high-probability words that are accessible only after a lower-probability word. As a result, greedy decoding does not formally guarantee a global optimum of the decoding objective, as its choices are only locally optimal. Despite its simplistic approach, greedy decoding remains a widely used generation algorithm. For example, it is employed in Google's Gemini report [131] and is commonly available in standard language model APIs.

#### 3.1.2 *Beam Search*

Beam search reduces the risk of overlooking high-probability word sequences by maintaining a fixed number of the most likely hypotheses (or "beams") at each timestep, ultimately selecting the hypothesis with the highest overall probability. While beam search consistently finds output sequences with higher probabilities than greedy search, it does not guarantee the most likely output. Beam search performs well in tasks where the desired generation length is relatively predictable, such as machine translation or summarization [90; 165]. However, this approach is less suited for open-ended generation tasks, like dialogue or story generation, where the desired output length can vary significantly.

Beam search is prone to generating repetitive outputs. To address the lack of diversity in the generated sequences, [138] proposed Diverse Beam Search (DBS). This algorithm divides the beam into multiple sub-groups and introduces an inner iteration at each timestep to maximize diversity between these groups.

#### 3.1.3 *Contrastive Search*

Contrastive search [123; 122] mitigates string repetition by penalizing the selection of previously generated token sequences. This method can suppress repetitions more effectively than beam search while utilizing a comparable amount of computational resources.

### 3.2 Stochastic Methods

Human-generated text exhibits greater variance in token probabilities, reflecting a diverse range of word choices, often unexpected. In contrast, the output from deterministic methods shows minimal variance, resulting in more predictable and potentially repetitive text. To address these limitations, stochastic approaches introduce randomness during the decoding process, leading to more diverse and natural text generation. Commonly used stochastic techniques include top-$k$ sampling, top-$p$ sampling, and temperature scaling.

#### 3.2.1 *Top-k Sampling*

In Top-$k$ sampling [31; 44], the $k$ most probable next words are selected, and their probability mass is redistributed to form a new distribution. Specifically, the method first identifies the top $k$ tokens with the highest probabilities for the current sequence. The probabilities of these $k$ tokens are then normalized to sum to 1, resulting in a truncated distribution. A token is then randomly sampled from this distribution and appended to the current sequence. This process is repeated iteratively until a termination condition is satisfied. GPT-2 employed this sampling strategy, which significantly contributed to its effectiveness in story generation.

While Top-$k$ sampling is both effective and significantly more efficient than beam search, it has limitations in specific scenarios, particularly in two edge cases. First, when the next-token distribution is widely spread and approaches a uniform distribution, Top-$k$ sampling may arbitrarily exclude numerous potentially interesting tokens, thereby reducing the diversity of the generated text. Conversely, when the distribution is highly concentrated, Top-$k$ sampling might either include unnecessary tokens if $k$ is too large or exclude equally probable ones if $k$ is too small.

The primary challenge with Top-$k$ sampling is how to determine an optimal $k$ value. The ideal choice of $k$ can vary based on the context and the shape of the probability distribution at each step. Using a fixed $k$ value may prove too restrictive in some scenarios while being overly permissive in others.

#### 3.2.2 *Top-p (Nucleus) Sampling*

To address the limitations of Top-$K$ sampling, where restricting the sample pool to a fixed size $K$ can lead to gibberish outputs for sharp distributions and stifle creativity for flat distributions, [43] proposed Top-$P$ sampling. This method selects the smallest set of words whose cumulative probability meets or exceeds a predefined threshold $p$. Rather than sampling exclusively from the top $k$ most probable words, Top-$p$ redistributes the probability mass across this dynamic set. This adaptive approach allows the size of the word set (i.e., the number of included words) to increase or decrease based on the shape of the next-token probability distribution.

#### 3.2.3 *Temperature Sampling*

Temperature sampling is among the most commonly used decoding strategies. Determining the optimal temperature

value typically involves ad-hoc experimentation tailored to the specific application. The core idea of temperature sampling is to control the "sharpness" of the probability distribution by introducing a temperature parameter $t$. This parameter is applied in the softmax function after the transformer's final layer to compute token probabilities. The temperature $t$ directly influences the level of randomness in the sampling process, with higher values increasing randomness and lower values reducing it.

## 3.3 Speculative Decoding

In addition to output quality, decoding strategies must also consider inference efficiency due to the ever-growing size of models. One of the main drawbacks of autoregressive decoding is that its token-by-token generation can lead to increased inference latency, scaling with both the length of the generated sequence and the model's size. To accelerate inference for LLMs, speculative decoding [119; 61; 15] has been introduced, allowing for the simultaneous decoding of multiple tokens per step. Specifically, in each decoding step, speculative decoding first efficiently drafts multiple tokens as speculation for future decoding steps of the target LLM, and then utilizes the LLM to verify all drafted tokens in parallel. Only those tokens that meet the LLM's verification criteria are accepted as final outputs, ensuring generation quality [157].

# 4. DECODING PARADIGMS

Earlier decoding methods (§ 3) primarily focused on token-level diversity and fluency. More recent work extends decoding toward sequence-level control, structured guidance, and generation efficiency. In this survey, we identify three paradigms in recent decoding methods for LLMs and LVLMs: **contrastive decoding**, **guided decoding**, and **parallel decoding**. Table 1 lists recent works categorized by their decoding paradigms. Unlike earlier token-centric decoding methods, these paradigms emerge from recent efforts that explicitly optimize sequence-level quality, controllability, or efficiency.

## 4.1 Contrastive Decoding

The primary goal of contrastive decoding is to enhance the quality of generated outputs by contrasting positive and negative examples during the decoding process. Unlike greedy or sampling-based decoding methods, which offer only basic control at the token level, contrastive decoding operates at both the token and layer levels, allowing for greater control over the quality of the entire generated sequence.

**Definition 1 (Contrastive Decoding).** Contrastive decoding (CD) searches for the next token that maximizes a weighted difference in likelihood between two logits $z^+$ and $z^-$.

$$P(W_t|w_{<t}) = softmax(z^+ + \alpha(z^+ - z^-))$$

where $\alpha$ controls the strength of the modification. Contrastive decoding methods can be broadly classified into two categories based on the level at which contrastive examples are utilized: *token-wise CD* and *layer-wise CD*.

### 4.1.1 Token-wise Contrastive Decoding

Token-wise CD contrasts the token-level probability distributions produced by pairs of contrasting examples. These contrasting examples can be model-generated. Specifically, an expert model generates logits $z^+$, representing the user-desired direction while a weak or base model generates logits $z^-$, providing a baseline probability distribution. The contrastive decoding mechanism then adjusts token probabilities by weighting the difference between $z^+$ and $z^-$, regulated by the parameter $\alpha$. For example, DExpert [73] utilizes both expert and anti-expert models to guide output for language detoxification and sentiment-controlled generation. During the decoding process, each token is assigned a high probability if it is deemed likely by the expert LM and unlikely by the anti-expert LM. Similarly, [66] proposed a contrastive method that directly compares off-the-shelf LMs by computing the difference between their log probabilities. With the increasing size of LLMs, the efficiency benefits of advanced decoding methods have become more apparent. ROSE [182] applies contrastive decoding to pairs of carefully crafted reverse prompts to enhance LLM safety. Addressing the role of contextual information in text generation, [116] proposed context-aware decoding (CAD), which uses a contrastive output distribution to highlight differences in token probabilities when the model generates text with and without contextual input. Building on this, [178] developed a method that combines contrastive decoding with adversarial irrelevant passages as negative samples, improving robust contextual grounding by contrasting relevant and irrelevant contexts. Extending these approaches to noisy contexts, adaptive contrastive decoding (ACD) [55] effectively leverages contextual influences to address challenges posed by noisy or imperfect inputs. To further enhance the efficiency of contrastive decoding, [166] introduced Speculative Contrastive Decoding (SCD), a simple yet powerful approach that utilizes predictions from smaller LMs to achieve faster decoding while maintaining generation quality.

In the context of LVLMs, various CD methods have been proposed to enhance accuracy and mitigate hallucination. Visual Contrastive Decoding (VCD) [59] ensures visual grounding by comparing outputs from original and distorted visual inputs. Image-biased decoding (IBD) [185] contrasts predictions from conventional and image-biased LVLMs to improve image-text alignment and reduce hallucinations. Instruction Contrastive Decoding (ICD) [148] refines outputs by comparing distributions from standard and perturbed instructions, filtering hallucinated concepts. More recently, Visual Augmented Contrastive Decoding (VACoDe) [54] introduced an adaptive approach, selecting optimal augmentations per task via a novel softmax distance metric, surpassing single-augmentation methods.

### 4.1.2 Layer-wise Contrastive Decoding

Recent studies have shown that transformer models tend to encode lower-level information, such as part-of-speech tags, in the earlier layers, whereas more semantic information is encoded in the later layers [133]. For instance, [21] found that knowledge neurons are concentrated in the topmost layers of the BERT model. Moreover, [85] demonstrated that factual knowledge can be edited by manipulating a specific set of feedforward layers within an autoregressive model. Building on these observations, contrastive decoding has been extended to operate on layers within the models, besides just token-level adjustments.

Decoding by Contrasting Layers (DoLa) [20] enhances fac-

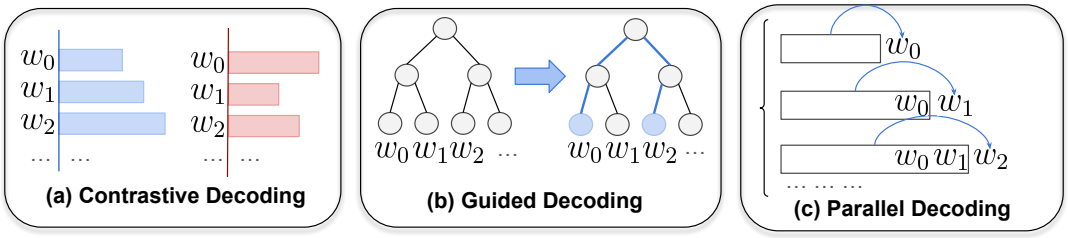


Figure 3: Illustration of different decoding paradigms. **Contrastive decoding** selects the next token by maximizing the contrast between two underlying probability distributions. **Guided decoding** determines the next token based on the highest score from a guidance function. **Parallel decoding** generates multiple token candidates simultaneously and selects the most probable one.

tual knowledge in LMs by leveraging modular encoding and contrastive decoding. It derives the next-token distribution by contrasting logits from different transformer layers projected onto the vocabulary space, based on the observation that factual knowledge is localized in specific layers. In a similar fashion, [23] proposed an entropy-guided method to extrapolate token probabilities beyond the last layer for more accurate contrastive decoding, instead of relying solely on the final layer. To reduce noise from retrieved context, [104] employed an entropy-based document-parallel ensemble decoding, which prioritizes low-entropy distributions from retrieved documents. Specifically, it compares this low-entropy ensemble with the model's high-entropy internal distribution, emphasizing reliable external information. By leveraging the modular and hierarchical nature of factual knowledge within LLMs, Dynamic Focus Decoding (DFD) [84] adaptively adjusts the decoding focus based on distributional differences across layers.

More recently, [114] addressed the underutilization of Mixture-of-Experts (MoE) models, where unchosen experts do not contribute to the output. They proposed Self-Contrast Mixture-of-Experts (SCMoE), an inference strategy that improves performance by contrasting strong and weak expert activations within the model. For LVLMs, [144] found that key visual features in early layers often distort as they propagate to the output. Building on this finding, they introduced Visual Layer Fusion Contrastive Decoding (VaLiD), which uses uncertainty to guide visual layer selection, reducing distortions and mitigating hallucinations.

## 4.2 Guided Decoding

Selecting the right decoding "path" can greatly enhance the generation quality of LLMs and LVLMs. For example, chain-of-thought reasoning can emerge simply by modifying the decoding process [149]. To determine the optimal decoding path," various strategies have been proposed to guide the decoding process. We refer to these strategies collectively as guided decoding, which is defined as follows:

**Definition 2 (Guided Decoding).** Guided decoding (GD) searches for the next token that maximizes the score from a guidance function $\mathcal{G}$.

$$P'(w_t|w_{<t}) \propto \mathcal{G}(P(w_t|w_{<t}), C)$$

where $\mathcal{G}$ is a guidance function that adjusts the model's distribution based on certain criteria, and $C$ denotes the control condition. We classify guided decoding into two main categories: *classifier-guided* and *heuristic-guided* decoding.

### 4.2.1 *Classifier-Guided Decoding*

Classifier-guided decoding utilizes an external classifier to influence the decoding process, allowing for control over specific attributes during text generation. The classifier can take various forms, such as a reward model, a pre-trained model, or an API, and adjust the model's output accordingly.

By using an attribute model to guide the decoding process, Plug and Play Language Model (PPLM) [24] controls text generation by combining a pre-trained LM with one or more simple attribute classifiers. To detoxify PLMs at the token level, [177] proposed Multiple Instance Learning Decoding (MIL-Decoding), which computes token-level toxicity scores and adjusts probabilities dynamically based on context. CriticControl [53] combines reinforcement learning and weighted decoding, using an LM-steering critic for controlled text generation. Similarly, Reward-Augmented Decoding (RAD) [26] modifies token probabilities using a unidirectional reward model, optimizing sampling for better attribute control.

To address uncertainty in multi-step reasoning, [159] proposed a decoding algorithm using self-evaluation guidance through stochastic beam search, improving prediction quality by enhancing search efficiency. To reduce reasoning errors in intermediate steps, Deductive Beam Search (DBS) [186] integrates chain-of-thought and deductive reasoning with a step-wise beam search and a verifier to check the validity of each step, reducing error accumulation. SafeDecoding [161] mitigates jailbreak risks by guiding the decoding process with a trained expert model to generate helpful and safe responses. For improved code generation in LLMs, [1] introduced monitor-guided decoding (MGD), where a monitor uses static analysis to guide decoding. Lastly, [6] proposed DOMINO, a decoding algorithm that enforces subword-aligned constraints with minimal overhead, addressing challenges in aligning sub-word tokens and grammar terminals.

For LVLMs, CLIP-Guided Decoding (CGD) [25] enhances visual grounding by using CLIP to guide the decoding process, with CLIP similarity to the image serving as a stronger indicator of hallucinations than token likelihoods. [33] introduced dropout decoding, inspired by dropout regularization, to reduce hallucinations by quantifying and masking uncertain visual tokens during inference. Summary-Guided Decoding (SGD) [89] addresses hallucinations by shortening token length through summarization, promoting greater visual detail while controlling image-related part-of-speech tokens to maintain text quality.

### 4.2.2 Heuristic-Guided Decoding

To provide more effective guidance within the token search space, heuristics are used to steer the decoding process with specific rules or search strategies. These heuristics could include constraints, results from lookahead search, or value functions.

Future Discriminators for Generation (FUDGE) [164] adjusts the probability distribution during generation by predicting the attribute probability of the evolving sequence and modifying the logits to align with desired attributes. To address the challenges PLMs face in adhering to constraints during generation, [82] proposed Neurologic Decoding, which enforces lexical constraints during decoding. [81] later improved it with A*-inspired lookahead heuristics for better performance. For generating mathematical proofs, [153] developed NaturalProver, a knowledge-grounded model that generates proofs by conditioning on background theorems and definitions, optionally enforcing them through constrained decoding. Lastly, [28] introduced Adaptive Decoding, a neural module that predicts the optimal sampling temperature for specific tasks.

Using heuristics from lookahead search results, Planning-Guided Transformer Decoding (PG-TD) [174] leverages a planning algorithm to conduct lookahead searches and guide the model in generating more effective programs for code generation with LLMs. To leverage value models trained as byproducts when aligning LMs with human preferences, [74] proposed an effective method for applying Monte-Carlo tree search decoding on top of PPO-trained policy and value models. This approach integrates the value network from PPO, enabling it to collaborate closely with the policy network during inference-time generation. More recently, TS-LLM [34] leverages tree search with a learned value function to guide LLM decoding, enhancing reasoning, planning, and decision-making capabilities. Integrative Decoding [19] improves actuality by implicitly incorporating self-consistency within its decoding objective.

To mitigate hallucinations in LVLMs, [48] introduced Self-Introspective Decoding (SID) with the Context and Text-aware Token Selection (CT2S) strategy. SID retains only the least important vision tokens after the early decoder layers, adaptively addressing hallucinations in vision-and-text associations during autoregressive decoding. This approach leverages the ability of pre-trained LVLMs to introspectively evaluate the significance of vision tokens based on prior vision, text, and generated content. For diagnostic captioning, [50] proposed Distance from Median Maximum Concept Similarity (DMMCS), a data-driven guided decoding method that incorporates medical image tags to generate more accurate diagnostic text.

## 4.3 Parallel Decoding

Unlike standard sequential decoding, parallel decoding executes multiple decoding processes concurrently where it first generates multiple candidate sequences simultaneously and then selects the most likely one based on a set of predefined criteria.

**Definition 3 (Parallel Decoding).** Parallel decoding first decodes multiple future tokens $y_1, y_2, ..., y_m$ simultaneously, a process often referred to as *drafting*.

$$\begin{cases} y_1 = \arg\max P(w_t|w_0) \\ y_2 = \arg\max P(y_2|y_1, w_0) \\ \vdots \\ y_m = \arg\max P(y_m|y_{1:m-1}, w_0) \end{cases}$$

Then, it aggregates these tokens $y_1, y_2, ..., y_m$ in parallel using the target LLM to speed up inference – a process known as *verification*.

$$P(w_t|w_{<t}) = \mathcal{M}(w|w_{\leq t}, \hat{w}_{\leq i}), i = 1, ..., K+1$$

where $\mathcal{M}$ is the draft model, and $\hat{w}_1$, $\hat{w}_2$, ..., $\hat{w}_K$ are the draft tokens outputted by $\mathcal{M}$. This *draft-then-verify* paradigm can be further classified into two categories: *greedy* and *sampling*.

### 4.3.1 Greedy

To enhance the decoding process in deep autoregressive models, [119] introduced blockwise decoding, a parallel approach in which predictions are made for multiple time steps simultaneously, followed by a rollback to the longest valid prefix as determined by a scoring model. To improve online inference efficiency in transformer-based models for instantaneous grammatical error correction, Shallow Aggressive Decoding (SAD) [127] uses a shallow decoder to aggressively decode as many tokens as possible in parallel. To overcome the efficiency limitations of autoregressive decoding in transformers, [108] redefine standard greedy autoregressive decoding for machine translation by adopting a parallel formulation that uses Jacobi and Gauss-Seidel fixed-point iteration methods to achieve faster inference.

Notably, [155] introduced Speculative Decoding (SpecDec), a method to accelerate autoregressive decoding through speculative execution, or draft-then-verify. It consists of two components: Spec-Drafter, an independent model for efficient token drafting, and Spec-Verification, a mechanism for validating the drafted tokens. To enhance inference efficiency in RAG, [150] proposed SpeculativeRAG, a framework in which a larger generalist LM verifies multiple RAG drafts generated in parallel by a smaller, distilled specialist LM. Each draft, based on distinct subsets of retrieved documents, reduces token counts and offers diverse perspectives, improving comprehension, mitigating position bias, and speeding up the RAG process by limiting the generalist LM to a single verification pass.

To eliminate the need for separate draft and verifier models, self-speculative decoding [173] uses a single LLM for both drafting and verification, avoiding additional training and memory overhead. More recently, [36] introduced Lookahead Decoding, a parallel algorithm that accelerates LLM decoding without relying on auxiliary models or data stores. This approach balances per-step log (FLOPs) with the total decoding steps, making it highly parallelizable on modern accelerators and compatible with memory-efficient attention mechanisms like FlashAttention. Lastly, [121] proposed a technique to facilitate dynamic reconfiguration of parallelization strategies across prefilling and decoding stages to improve the efficiency of distributed LLM inference.

### 4.3.2 Sampling

Similar to how stochastic decoding methods often surpass

Table 1: List of works categorized by the three paradigms identified in this survey. Based on chronological order.

| Paradigm | Work | Description | Model | Year (↓) |
|---|---|---|---|---|
| Contrastive Decoding | DExpert [73] | Detoxification with contrastive decoding | PLM | 2021 |
| | CD [66] | Formulate Contrastive Decoding | PLM | 2022 |
| | CAD [116] | Context-aware Decoding | LLM | 2023 |
| | SCD [166] | Speculative contrastive decoding | LLM | 2023 |
| | DoLa [20] | Decoding by contrasting layers | LLM | 2023 |
| | VCD [59] | Visual contrastive decoding | LVLM | 2024 |
| | ROSE [182] | Reverse prompt contrastive decoding | LLM | 2024 |
| | Zhao et al. [178] | Multi-input contrastive decoding | LLM | 2024 |
| | CLeHe [104] | Entropy-based contrastive decoding | LLM | 2024 |
| | ACD [55] | Adaptive contrastive decoding | LLM | 2024 |
| | SCMoE [114] | Self-contrast Mixture-of-Experts | LLM | 2024 |
| | Das et al. [23] | Entropy guided extrapolative decoding | LLM | 2024 |
| | ICD [148] | Instruction contrastive decoding | LVLM | 2024 |
| | IBD [185] | Image-biased decoding | LVLM | 2024 |
| | VACoDe [54] | Visual augmented contrastive decoding | LVLM | 2024 |
| | VaLiD [144] | Visual Layer Fusion contrastive decoding | LVLM | 2024 |
| Guided Decoding | PPLM [24] | Attribute model guidance | PLM | 2020 |
| | Neurologic [82] | Constrained decoding | PLM | 2020 |
| | FUDGE [164] | Future discriminator guidance | PLM | 2021 |
| | Neurologic$A^\star$ [81] | Lookahead heuristics guidance | PLM | 2021 |
| | CriticControl [53] | Critic-guided decoding | PLM | 2022 |
| | NaturalProver [153] | Stepwise constrained decoding | PLM | 2022 |
| | MIL-Decoding [177] | Multiple instance learning guidance | PLM | 2023 |
| | RAD [26] | Reward augmented decoding | LLM | 2023 |
| | PPO-MCTS [74] | Value-guided Monte-Carlo tree search | LLM | 2023 |
| | PG-TD [174] | Planning-guided decoding | LLM | 2023 |
| | Xie et al. [159] | Self-evaluation guided beam search | LLM | 2023 |
| | DBS [186] | Decoding deducible rationale for CoT | LLM | 2024 |
| | DMMCS [50] | Data-driven guided decoding | LVLM | 2024 |
| | SGD [89] | Summary-Guided decoding | LVLM | 2024 |
| | SID [48] | Self-Introspective decoding | LVLM | 2024 |
| | TS-LLM [34] | AlphaZero-like tree-search | LLM | 2024 |
| | Dropout [33] | Dropout decoding | LVLM | 2024 |
| | MGD [1] | Monitor-guided decoding | LLM | 2024 |
| | SafeDecoding [161] | Expert model guidance | LLM | 2024 |
| | DOMINO [6] | Minimally-Invasive constrained decoding | LLM | 2024 |
| | CGD [25] | Clip-guided decoding | LVLM | 2024 |
| | DFD [84] | Dynamic Focus Decoding | LLM | 2025 |
| | AttnReal [135] | Attention reallocation | LVLM | 2025 |
| Parallel Decoding | Blockwise [119] | Blockwise parallel decoding | PLM | 2018 |
| | SAD [127] | Shallow aggressive decoding | PLM | 2021 |
| | SpecDec [155] | Speculative decoding for seq2seq generation | PLM | 2023 |
| | Santilli et al. [108] | Hybrid GS-Jacobi decoding | PLM | 2023 |
| | Self-speculative [173] | Self-speculative decoding | LLM | 2023 |
| | Speculative [61] | Speculative sampling | LLM | 2023 |
| | Speculative [15] | Speculative sampling with distributed serving | LLM | 2023 |
| | DistillSpec [184] | Speculative via knowledge distillation | LLM | 2023 |
| | SpecInfer [88] | Tree-based speculative verification | LLM | 2023 |
| | Online Speculative [78] | Online Speculative | LLM | 2023 |
| | Speculative RAG [150] | Speculative RAG | LLM | 2024 |
| | Lookahead [36] | Lookahead decoding | LLM | 2024 |
| | Medusa [8] | Multiple decoding heads | LLM | 2024 |
| | Eagle [69] | Extrapolative speculative sampling | LLM | 2024 |
| | Gagrani et al. [39] | Multimodal speculative | LVLM | 2024 |
| | Lantern [49] | Latent neighbor token acceptance relaxation | LVLM | 2024 |
| | SJD [132] | Speculative Jacobi decoding | LVLM | 2024 |
| | SPD [111] | Superposed decoding | LLM | 2024 |
| | Swift [156] | On-the-fly Self-speculative decoding | LLM | 2024 |
| | Seesaw [121] | Dynamic Model Resharding | LLM | 2025 |

deterministic approaches, integrating speculative sampling can significantly boost the performance of parallel decoding. Speculative sampling [61] speeds up sampling from autoregressive models while preserving accuracy by extending speculative execution to the stochastic setting. It enhances exact decoding from large models by running them in parallel with approximation models, generating multiple tokens concurrently without altering the underlying distribution. Similarly, [15] used speculative sampling to generate multiple tokens per transformer call, optimizing distributed model serving. To align compact draft models with target models, DistillSpec [184] applies knowledge distillation before speculative decoding, customizing the divergence function. More recently, [111] introduced Superposed Decoding, which generates $k$ drafts with a single autoregressive inference pass and predicts $k^2$ drafts per step using n-gram interpolation to filter out incoherent outputs.

To address low predictive accuracy in draft models, especially with diverse text inputs and significant gaps between draft and target models, [78] proposed Online Speculative Decoding. This method updates draft models based on user query data, improving predictions by adapting to query distributions. To minimize additional parameters or training for effective draft models, [156] introduced Swift, a plug-and-play speculative decoding solution that uses layer-skipping, leveraging the compact draft model by skipping intermediate layers of the target LLM.

Token tree verification combines multiple candidate draft sequences into a token tree, sharing prefixes, and applies a tree attention mask for efficient verification. [88] introduced SpecInfer, a tree-based parallel decoding algorithm that uses small speculative models to predict LLM outputs and organizes predictions into a token tree. This approach reduces latency and computational costs while maintaining model quality. [8] proposed Medusa, an efficient method that enhances LLM inference by adding extra decoding heads to predict multiple tokens in parallel, reducing decoding steps through parallel processing. [69] introduced EAGLE, a speculative sampling framework that addresses feature-level autoregression uncertainty by incorporating a token sequence advanced by one time step, enabling efficient second-to-top-layer feature prediction. Subsequently, [68] improved this with EAGLE-2, introducing a context-aware dynamic draft tree for more accurate draft modeling, leveraging well-calibrated draft models with closely approximated confidence scores.

To explore speculative decoding for LVLMs, [39] enhanced inference efficiency in LVLMs, focusing on the LLaVA 7B model. [49] address token selection ambiguity, where visual autoregressive models assign uniformly low probabilities to tokens, impairing speculative decoding. They propose LANTERN, a relaxed acceptance condition utilizing token interchangeability in latent space, restoring speculative decoding effectiveness while maintaining image quality and semantic coherence through a total variation distance bound. [132] introduced Speculative Jacobi Decoding (SJD) for autoregressive text-to-image generation, enabling the model to predict and accept multiple tokens per step, generating images more efficiently than traditional methods.

# 5. DECODING APPLICATIONS

In this section, we shift our focus to organizing these methods according to the applications in which they have been used. Decoding algorithms play a crucial role across a variety of applications, from enhancing model alignment to optimizing specific generation tasks. Understanding how these algorithms are adapted to different applications provides valuable insights into their versatility and effectiveness. Figure 4 illustrates the diverse applications of decoding methods.

## 5.1 Improve Model Alignment

Decoding strategies play a vital role in improving model alignment by mitigating hallucinations, enhancing safety, and strengthening reasoning capabilities. These strategies involve tailoring the output generation process during inference to achieve the desired outcomes. Unlike other methods of improving model alignment, decoding strategies provide a dynamic and adaptive approach, ensuring that the model consistently meets user expectations while adhering to ethical standards.

### *5.1.1 Mitigate Hallucination*

Decoding methods have emerged as an effective inference-time tool for mitigating hallucinations [46], which refers to the generation of plausible but factually incorrect content by the model. Compared to prompt-based [41; 140; 175] and knowledge-editing [176; 64] approaches, decoding methods are model-agnostic and offer better interpretability. [40] introduced Decoding by Contrasting Retrieval Heads (DeCoRe), a decoding strategy that mitigates hallucinations by dynamically contrasting the outputs of a base and masked LLM, guided by conditional entropy. Similarly, [162] proposed a Comparator-driven Decoding-Time (CDT) framework, which generates hallucinatory and truthful comparators using multi-task fine-tuning and refines next-token predictions by contrasting logit differences between the target LLMs and these comparators.

In addition to mitigating hallucinations in LLMs, recent studies have demonstrated the effectiveness of contrastive decoding in addressing object hallucinations in LVLMs. [58] explores visual contrastive decoding techniques, such as image downsampling and editing, to reduce hallucinations. [98] proposed ConVis, which reconstructs images from hallucinated captions using a text-to-image model and compares probability distributions to capture visual contrastive signals that penalize hallucination generation. To counter hallucinations caused by strong language model priors suppressing visual input, [139] introduced DeCo, a dynamic correction decoding method that selectively integrates knowledge into the final layer to adjust output logits. [48] developed $CT^2S$, which preserves only the least important vision tokens after early decoder layers, enhancing vision-text associations during autoregressive decoding. Inspired by the Information Bottleneck theory, [51] proposed CATCH, which integrates complementary visual decoupling for information separation, non-visual screening for hallucination detection, and adaptive token-level contrastive decoding for mitigation. More recently, [14] has discovered an "attention hijacking" phenomenon, where interference from instruction tokens distorts visual perception, diverting attention to less discriminative regions and leading to hallucinations. Additionally, [135] proposed an attention reallocation mechanism that redistributes excess attention from output tokens to vi-

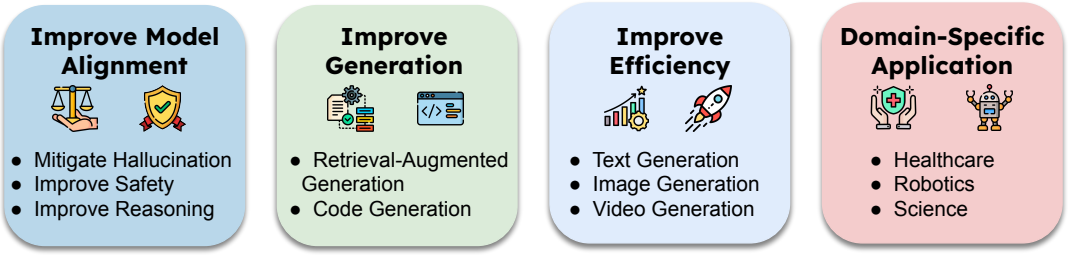


Figure 4: Decoding applications.

sual tokens, reducing LVLMs' reliance on language priors and strengthening visual input dependence to mitigate hallucinations.

### 5.1.2 *Improve Safety*

Besides mitigating hallucination, decoding methods are also used to enhance model safety. [2] introduced Safeinfer, a context-adaptive safety alignment strategy for generating safe responses using safety-guided decoding, which selects tokens based on the safety-optimized distributions to ensure ethical content. Similarly, [169] proposed the Root Defense Strategy, a decoder-oriented defense that corrects harmful queries directly, rather than rejecting them outright. Recognizing that LLMs may appear to block harmful queries but still harbor latent risks, [142] proposed Jailbreak Value Decoding (JVD). JVD evaluates the probability of generating harmful content in subsequent decoding steps from any given point.

Instead of manually selecting contrastive models or instruction templates, [179] proposed Adversarial Contrastive Decoding (ACD), an optimization-based framework that generates two opposing system prompts for prompt-based contrastive decoding. Using contrastive decoding, [94] successfully mitigates toxicity in a parameter-efficient manner. To improve alignment through decoding, [11] uses Q-learning to adjust the response distribution directly, maximizing a target reward without requiring model updates. Moreover, [3] identifies amplification bias and homogeneity issues in existing LLM decoding methods for recommendations. They propose Debiasing-Diversifying Decoding ($D^3$), which disables length normalization for ghost tokens to reduce amplification bias and uses a text-free assistant model to promote less frequent token generation, addressing recommendation homogeneity. To ensure the safety of conversational LLMs, [38] examines the effect of various decoding methods on the alignment between LLM-generated and human conversations. They show that fewer beams in beam search and lower $P$ values in Nucleus sampling could lead to better alignment. PAD [143] adaptively injects calibrated Gaussian noise into token logits to mitigate privacy leakage of retrieved context in RAG.

### 5.1.3 *Improve Reasoning*

Reasoning [45] is a key aspect of human intelligence and an important trait for LLMs. Recent works have shown that reasoning chains are embedded in token selection, and decoding methods can enhance reasoning ability. [95] demonstrated that contrastive decoding improves reasoning in LLMs, outperforming existing methods by preventing abstract reasoning errors and avoiding simpler strategies like copying input sections during chain-of-thought reasoning. [101] proposed Distillation Contrastive Decoding (DCD), which enhances LLM reasoning at inference time by employing Contrastive Chain-of-Thought Prompting and advanced distillation techniques, including Dropout and Quantization, without requiring expert and amateur models. By investigating top-k alternative tokens, [149] found that CoT paths often emerge within these sequences and can be elicited from LLMs by modifying the decoding process.

Recently, to improve the performance of models that have been knowledge-edited for reasoning questions, [128] proposed Outdated Issue-Aware Decoding (DISCO), which captures the difference in the probability distribution between the original and edited models. To improve LLM reasoning in cross-domain settings, [112] proposed a method to teach multiple LLMs to collaborate by interleaving their generation at the token level. In parallel to syntactic decoding, such as auto-regressive decoding, [100] introduced semantic decoding, a perspective that views collaborative processes as optimization procedures within a semantic space, offering an abstraction for search and optimization directly in the space of semantic tokens.

In addition to improving reasoning performance, decoding methods can enhance inference efficiency for reasoning tasks. To reduce the cost of generating a full CoT reasoning chain, [77] introduced an auxiliary CoT model that generates and compresses the entire thought process into a compact token representation, semantically aligned with the original CoT output. To reduce the inference latency in tree-search-based reasoning methods, [151] introduced SEED, an efficient inference framework that accelerates reasoning tree construction using speculative scheduled execution, parallel drafting with speculative decoding, and a rounds-scheduled strategy to manage parallel drafts without verification conflicts. To reduce the cost of self-consistency decoding from the sampling process, [16] proposed self-para-consistency, where multiple paraphrases are generated for each test question.

## 5.2 Improve Generation Tasks

Decoding strategies enhance large generative models in tasks such as retrieval-augmented generation and code generation, improving both output quality and efficiency. These advancements focus on refining decoding methods, optimizing model architectures, and incorporating external resources to enhance performance.

### 5.2.1 *Retrieval Augmented Generation*

Decoding methods can effectively improve RAG at inference

time. [104] proposed entropy-based decoding to enhance truthfulness in retrieval-augmented LLMs, addressing challenges in ensuring faithful information retrieval. Similarly, adaptive contrastive decoding (ACD) [55] effectively incorporates contextual influence, improving the model's ability to generate contextually relevant outputs. To overcome the limitations of the generative retrieval model's fixed parametric capacity, [57] introduced Nonparametric Decoding (Np Decoding), which replaces standard embeddings with nonparametric contextualized vocab embeddings. Additionally, [168] proposed PAG, an optimization and decoding approach that guides the autoregressive generation of document identifiers in generative retrieval models through simultaneous decoding, streamlining the document retrieval process. More recently, to speed up language model generation with a retrieval-based approach, [42] proposed Retrieval-Based Speculative Decoding (REST). Unlike previous methods that rely on a draft language model, REST uses retrieval to generate draft tokens, leveraging the observation that text generation often follows common phrases and patterns.

### 5.2.2 Code Generation

Besides RAG, various decoding methods have been used for code generation with LLMs. [187] introduced Adaptive Temperature (AdapT) sampling, which adjusts the temperature during token decoding: higher for challenging tokens to explore diverse options, and lower for confident tokens to reduce tail randomness noise. [63] proposed Decoding Objectives for Code Execution (DOCE), a framework for execution-based evaluation. They focus on the effects of high-temperature sampling, execution-based reranking with high-quality unit tests, and self-debugging with multiple candidates. [102] introduced DocCGen, a two-step NL-to-code generation framework for structured domain-specific languages like YAML and JSON. It first detects relevant libraries using documentation to match the query, then constrains decoding with schema rules from these libraries.

To address security and correctness in code generation with Code LLMs, [37] investigates a defense approach using constrained decoding to generate secure code, proving more effective than prefix tuning without requiring a specialized training dataset. [146] presents Uncertainty-Aware Selective Contrastive Decoding (USCD), which improves one-pass code generation performance. It pre-judges noise in output distributions using standard deviation, then applies a "lame" prompt to reduce noise and enhance code quality. [92] proposed LEVER, which trains verifiers to assess program correctness using natural language input, the program, and its execution results. Programs are reranked by combining verification scores with LLM probabilities, marginalizing over identical execution results.

Additionally, decoding methods can help mitigate hallucination during code generation. [93] proposed DESEC, a two-stage method that uses token-level features to guide decoding. It builds an offline token scoring model with a proxy Code LLM and adjusts token likelihoods during decoding based on these scores. [134] introduced Selective Prompt Anchoring (SPA), which addresses self-attention dilution in LLMs for code generation. SPA strengthens the influence of selected parts of the initial prompt by adjusting the logit distribution based on the difference between anchored and non-anchored text.

## 5.3 Improve Generation Efficiency

Decoding methods are used to boost generation efficiency across multiple domains, such as text, image, and video.

### 5.3.1 Text Generation

Beyond speculative decoding, several works focus on enhancing text generation efficiency through innovative decoding strategies. Reward-Augmented Decoding (RAD) [26] used a lightweight unidirectional reward model to guide a language model in producing text with desired properties. [188] introduced Hierarchical Skip Decoding (HSD), an efficient autoregressive method that adaptively skips decoding layers based on sequence length, reducing computational overhead without requiring additional trainable components. [163] proposed Frustratingly Simple Decoding (FSD), which uses an anti-language model to penalize repetitive content. The anti-LM can be implemented as an n-gram model or a vectorized variant, adding no extra parameters and incurring minimal cost, making it as fast as greedy search.

More recently, [79] proposed ADED, a decoding methodology that enhances LLM efficiency without fine-tuning, utilizing an adaptive draft-verification process. [121] proposed Seesaw, which uses dynamic model resharding to enable reconfiguration of parallelization strategies during decoding. [32] introduced Position-Aware Depth Decay Decoding, which employs a power-law decay function to optimize the number of layers retained during token generation for more efficient performance.

### 5.3.2 Image Generation

To accelerate autoregressive text-to-image generation, [132] introduced Speculative Jacobi Decoding (SJD), a training-free probabilistic parallel decoding algorithm. [35] explores non-autoregressive text-to-image models that generate image tokens in parallel. They introduce an iterative mask-predict approach, enabling the model to refine its predictions using partially observed tokens, which enhances convergence speed and output quality. [130] introduced the Hybrid Autoregressive Transformer (HART), an autoregressive visual generation model that employs hybrid tokenization. This approach enables continuous feature decoding during generation, overcoming the limitations of finite VQ codebooks and improving overall generation quality.

### 5.3.3 Video Generation

By reformulating dense caption generation as a set of prediction tasks, [147] proposed PDVC, an end-to-end dense video captioning framework with parallel decoding. To ensure global coherence and local realism in video generation, GLOBER [126] uses a video decoder that processes global features and synthesizes video frames in a non-autoregressive manner. For enhanced flexibility, the video decoder incorporates normalized frame indexes to perceive temporal information, enabling the generation of arbitrary sub-video clips with predefined starting and ending frame indexes. Similarly, VideoGen [65] employs an advanced video decoder trained on unlabeled data to produce high-definition videos with strong temporal consistency and high frame fidelity.

## 5.4 Domain-Specific Applications

Decoding strategies can significantly enhance domain-specific

applications, such as healthcare and robotics. In the *healthcare* domain, [160] addresses hallucination in medical information extraction tasks with Alternate Contrastive Decoding (ALCD). They redefine the task as an identification-and-classification process, separating these steps by masking token optimization during fine-tuning. During inference, ALCD improves both identification and classification by contrasting output distributions from sub-task models, thereby minimizing interference from other LLM capabilities. Additionally, an adaptive constraint strategy refines the contrastive token scope, further enhancing performance. In the scientific domain, [180] implements cross-subject semantic decoding for video-stimulated fMRI, while [17] explores open-vocabulary auditory neural decoding using fMRI-prompted LLMs.

In *robotics*, [47] formulates the construction of action sequences as a probabilistic filtering problem, ensuring that the sequences are both probable according to the LM and feasible within grounded environmental models. Their approach demonstrates how grounded models can be derived from both simulation and real-world domains. By integrating knowledge from language models and grounded models, their decoding strategy effectively addresses complex, long-horizon embodiment tasks, enabling the generation of accurate and feasible action sequences. Similarly, [80] introduced Bidirectional Decoding (BID), an inference algorithm that combines action chunking with closed-loop operations. BID samples multiple predictions at each time step, optimizing for backward coherence (alignment with prior decisions) and forward contrast (high likelihood of future plans). This dual optimization approach ensures consistent decision-making while allowing the system to adapt to unexpected environmental changes.

# 6. DISCUSSIONS

Despite the significant potential of decoding methods across various tasks and applications, several challenges remain. In this section, we highlight the limitations of current decoding methods and discuss possible future research directions.

## 6.1 Exploring Dynamic and Universal Decoding Methods

Although decoding methods are effective for specific tasks, they often rely on manually crafted examples. As discussed earlier, token-wise and layer-wise contrastive decoding (§ 4.1) enhance control over text generation in LLMs and LVLMs. However, their effectiveness heavily depends on the selection of contrastive examples or layers. For instance, [66] compares outputs from smaller language models to those from larger ones, assuming that bigger models produce higher-quality text. However, this assumption does not always hold, as there are cases where the generation is worse with CD. Moreover, [110] even points out that smaller LLMs tend to be less sycophantic, meaning they are less likely to prioritize aligning with user beliefs over providing truthful responses. This highlights the complexity of crafting contrastive examples, as it requires balancing multiple qualities in the generated text. Similarly, in selecting contrasting layers, the optimal layers in DoLa [20] are sensitive across datasets, making it less versatile since it requires a task-specific validation set. This limitation presents an opportunity for future research to develop methods for constructing dynamic, universally applicable contrastive examples.

For guided decoding (§ 4.2), search methods like the one used in [174], rely on test cases and are constrained by a small search space. While [34] demonstrated that TS-LLMs perform well across reasoning, planning, alignment, and decision-making tasks on trees with a depth of 64, they still struggle to scale to larger scenarios due to the computational overhead introduced by node expansion and value evaluation. This highlights the need for more efficient strategies that can scale to larger problem spaces without introducing excessive computational costs.

## 6.2 Interpreting Decoding Methods

While some decoding methods, such as those explored by [149], offer insights into the intrinsic reasoning abilities of LLMs through the lens of decoding, more theoretical foundations are needed to understand why models behave in certain ways during the decoding process. For instance, [115] underscores the critical role of hyperparameter tuning in optimizing decoding methods. Their findings highlight that while some approaches can achieve impressive performance, they often require substantial effort to dial in the hyperparameters. In an initial effort, [13] theoretically demonstrates that contrastive decoding can be viewed as linearly extrapolating the next-token logits from a large, hypothetical language model. They find that this linear extrapolation may prevent contrastive decoding from producing the most obvious answers, as these are already assigned high probabilities by the base LM.

Moving forward, future research should focus on exploring the interpretability of these methods, which could provide valuable insights into how different decoding strategies align with human reasoning and decision-making processes. One potential approach is *mechanistic interpretability* [5; 71; 4], which seeks interpretability by reverse-engineering black-box models. For instance, [97] investigates the possibility of decoding multiple future tokens from a single token's hidden representation. Their causal intervention study reveals that certain layers can approximate the model's output with up to 48% accuracy from a single hidden state, providing significant insights into the model's prediction chain at each hidden state.

## 6.3 Combining Different Decoding Paradigms

As discussed in Section 4, contrastive decoding, guided decoding, and parallel decoding are versatile paradigms that have proven effective for a variety of tasks, such as controlling generation, improving quality, and enhancing efficiency. Given their individual successes, it is natural to explore the potential of combining these paradigms. For instance, [166] combined Speculative Decoding with contrastive decoding, achieving both improved generation quality and inference speedup. Additionally, these decoding paradigms can be integrated with other advanced generation techniques, such as RAG. [150] enhanced RAG by incorporating drafting with a set of specialized drafters, which provide diverse perspectives on the evidence while reducing the token count per draft. Future research could investigate further combinations of decoding paradigms to address the unique challenges posed by LLMs.

## 6.4 Expanding the Diversity of Decoding Objectives

Traditionally, decoding methods for LMs have primarily focused on text generation quality. However, recent works on LLMs and LVLMs have shifted toward improving the alignment of generated outputs. While several aspects of alignment have been explored, including toxicity [73], truthfulness [116; 20], and safety [161; 182], significant gaps remain in addressing other critical issues, such as *privacy, bias, and copyright concerns*. Additionally, while decoding methods have been applied across various domains, there are limited applications in high-stakes sectors like healthcare, law, and finance. Finally, for works focusing on improving the decoding efficiency through parallel decoding, there is a need to ensure the balance of decoding accuracy and efficiency. As noted by [158], there is still room for improvement to align the drafter with the target LLM for speculative decoding to scale up the drafter to improve decoding accuracy while maintaining its efficiency advantages. Future research could explore leveraging decoding methods to tackle these issues, offering a plug-and-play solution to enhance the trustworthiness of LLMs for important applications.

## 6.5 Improving Adversarial Robustness

Decoding methods are often overlooked from the security perspective. While much of the research focuses on enhancing the security of LLMs through techniques such as preventing data leakage and defending against jailbreaking attacks, the decoding mechanism itself is often neglected as a potential security risk. Decoding methods like SafeDecoding [161] and ROSE [182] are designed to generate safe responses from models, but they can also be exploited by attackers to generate malicious outputs. More concerningly, [91] demonstrated that adversaries with typical API access can steal the type and hyperparameters of a model's decoding algorithm at a low cost. This highlights a growing security concern regarding potential attacks that manipulate the underlying decoding method of a model. Moving forward, it would be valuable to explore strategies for defending against decoding-based attacks. As suggested by [91], watermarking [129] could serve as a potential countermeasure, adding noise to the final probability distribution and making it more difficult for attackers to extract hyperparameters from the target model.

# 7. CONCLUSION

This survey provides a comprehensive review of three primary decoding paradigms and their diverse applications in LLMs and LVLMs, showcasing their effectiveness and efficiency in tackling complex generation tasks. We believe that decoding methods offer a cost-effective way to enhance and extend LLM capabilities and hope our work sparks further discussion and research in this area.

# Acknowledgments

This material is based upon work supported by NSF awards (SaTC-2241068, IIS-2506643, and POSE-2346158), a Cisco Research Award, and NSF NAIRR Pilot Award #240469. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the National Science Foundation. This work is supported in part by NSF under grants III-2106758.

# 8. REFERENCES

[1] L. A. Agrawal, A. Kanade, N. Goyal, S. Lahiri, and S. Rajamani. Monitor-guided decoding of code lms with static analysis of repository context. *Advances in Neural Information Processing Systems*, 36, 2024.

[2] S. Banerjee, S. Tripathy, S. Layek, S. Kumar, A. Mukherjee, and R. Hazra. Safeinfer: Context adaptive decoding time safety alignment for large language models. *arXiv preprint arXiv:2406.12274*, 2024.

[3] K. Bao, J. Zhang, Y. Zhang, X. Huo, C. Chen, and F. Feng. Decoding matters: Addressing amplification bias and homogeneity issue for llm-based recommendation. *arXiv preprint arXiv:2406.14900*, 2024.

[4] N. Belrose, Z. Furman, L. Smith, D. Halawi, I. Ostrovsky, L. McKinney, S. Biderman, and J. Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

[5] L. Bereska and E. Gavves. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*, 2024.

[6] L. Beurer-Kellner, M. Fischer, and M. Vechev. Guiding llms the right way: Fast, non-invasive constrained generation. *arXiv preprint arXiv:2403.06988*, 2024.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] T. Cai, Y. Li, Z. Geng, H. Peng, J. D. Lee, D. Chen, and T. Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*, 2024.

[9] Z. Cao, Y. Yang, and H. Zhao. Nothing in excess: Mitigating the exaggerated safety for llms via safety-conscious activation steering. *arXiv preprint arXiv:2408.11491*, 2024.

[10] S. Casper, L. Schulze, O. Patel, and D. Hadfield-Menell. Defending against unforeseen failure modes with latent adversarial training. *arXiv preprint arXiv:2403.05030*, 2024.

[11] S. Chakraborty, S. S. Ghosal, M. Yin, D. Manocha, M. Wang, A. S. Bedi, and F. Huang. Transfer q star: Principled decoding for llm alignment. *arXiv preprint arXiv:2405.20495*, 2024.

[12] A. Chan, A. Madani, B. Krause, and N. Naik. Deep extrapolation for attribute-enhanced generation. *Advances in Neural Information Processing Systems*, 34:14084–14096, 2021.

[13] H.-S. Chang, N. Peng, M. Bansal, A. Ramakrishna, and T. Chung. Explaining and improving contrastive decoding by extrapolating the probabilities of a huge and hypothetical lm. *arXiv preprint arXiv:2411.01610*, 2024.

[14] B. Chen, X. Lyu, L. Gao, J. Song, and H. T. Shen. Attention hijackers: Detect and disentangle attention hijacking in lvlms for hallucination mitigation. *arXiv preprint arXiv:2503.08216*, 2025.

[15] C. Chen, S. Borgeaud, G. Irving, J.-B. Lespiau, L. Sifre, and J. Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

[16] W. Chen, W. Wang, Z. Chu, K. Ren, Z. Zheng, and Z. Lu. Self-para-consistency: Improving reasoning tasks at low cost for large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14162–14167, 2024.

[17] X. Chen, C. Du, C. Liu, Y. Wang, and H. He. Open-vocabulary auditory neural decoding using fmri-prompted llm. *arXiv preprint arXiv:2405.07840*, 2024.

[18] J. Cheng, X. Liu, K. Zheng, P. Ke, H. Wang, Y. Dong, J. Tang, and M. Huang. Black-box prompt optimization: Aligning large language models without model training. *arXiv preprint arXiv:2311.04155*, 2023.

[19] Y. Cheng, X. Liang, Y. Gong, W. Xiao, S. Wang, Y. Zhang, W. Hou, K. Xu, W. Liu, W. Li, et al. Integrative decoding: Improve factuality via implicit self-consistency. *arXiv preprint arXiv:2410.01556*, 2024.

[20] Y.-S. Chuang, Y. Xie, H. Luo, Y. Kim, J. Glass, and P. He. Dola: Decoding by contrasting layers improves factuality in large language models. *arXiv preprint arXiv:2309.03883*, 2023.

[21] D. Dai, L. Dong, Y. Hao, Z. Sui, B. Chang, and F. Wei. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*, 2021.

[22] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[23] S. Das, L. Jin, L. Song, H. Mi, B. Peng, and D. Yu. Entropy guided extrapolative decoding to improve factuality in large language models. *arXiv preprint arXiv:2404.09338*, 2024.

[24] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

[25] A. Deng, Z. Chen, and B. Hooi. Seeing is believing: Mitigating hallucination in large vision-language models via clip-guided decoding. *arXiv preprint arXiv:2402.15300*, 2024.

[26] H. Deng and C. Raffel. Reward-augmented decoding: Efficient controlled text generation with a unidirectional reward model. *arXiv preprint arXiv:2310.09520*, 2023.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[28] S. Dhuliawala, I. Kulikov, P. Yu, A. Celikyilmaz, J. Weston, S. Sukhbaatar, and J. Lanchantin. Adaptive decoding via latent preference optimization. *arXiv preprint arXiv:2411.09661*, 2024.

[29] C. Dong, Y. Li, H. Gong, M. Chen, J. Li, Y. Shen, and M. Yang. A survey of natural language generation. *ACM Computing Surveys*, 55(8):1–38, 2022.

[30] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon. Unified language model pre-training for natural language understanding and generation. *Advances in neural information processing systems*, 32, 2019.

[31] A. Fan, M. Lewis, and Y. Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

[32] S. Fan, X. Fang, X. Xing, P. Han, S. Shang, and Y. Wang. Position-aware depth decay decoding: Boosting large language model inference efficiency. *arXiv preprint arXiv:2503.08524*, 2025.

[33] Y. Fang, Z. Yang, Z. Chen, Z. Zhao, and J. Zhou. From uncertainty to trust: Enhancing reliability in vision-language models with uncertainty-guided dropout decoding. *arXiv preprint arXiv:2412.06474*, 2024.

[34] X. Feng, Z. Wan, M. Wen, S. M. McAleer, Y. Wen, W. Zhang, and J. Wang. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023.

[35] Z. Feng, R. Hu, L. Liu, F. Zhang, D. Tang, Y. Dai, X. Feng, J. Li, B. Qin, and S. Shi. Emage: Non-autoregressive text-to-image generation. *arXiv preprint arXiv:2312.14988*, 2023.

[36] Y. Fu, P. Bailis, I. Stoica, and H. Zhang. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*, 2024.

[37] Y. Fu, E. Baker, Y. Ding, and Y. Chen. Constrained decoding for secure code generation. *arXiv preprint arXiv:2405.00218*, 2024.

[38] S. Furniturewala, K. Jaidka, and Y. Sharma. Impact of decoding methods on human alignment of conversational llms. *arXiv preprint arXiv:2407.19526*, 2024.

[39] M. Gagrani, R. Goel, W. Jeon, J. Park, M. Lee, and C. Lott. On speculative decoding for multimodal large language models. *arXiv preprint arXiv:2404.08856*, 2024.

[40] A. P. Gema, C. Jin, A. Abdulaal, T. Diethe, P. Teare, B. Alex, P. Minervini, and A. Saseendran. Decore: Decoding by contrasting retrieval heads to mitigate hallucinations. *arXiv preprint arXiv:2410.18860*, 2024.

[41] Z. Gou, Z. Shao, Y. Gong, Y. Shen, Y. Yang, N. Duan, and W. Chen. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

[42] Z. He, Z. Zhong, T. Cai, J. D. Lee, and D. He. Rest: Retrieval-based speculative decoding. *arXiv preprint arXiv:2311.08252*, 2023.

[43] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[44] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi. Learning to write with cooperative discriminators. *arXiv preprint arXiv:1805.06087*, 2018.

[45] J. Huang and K. C.-C. Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[46] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2023.

[47] W. Huang, F. Xia, D. Shah, D. Driess, A. Zeng, Y. Lu, P. Florence, I. Mordatch, S. Levine, K. Hausman, et al. Grounded decoding: Guiding text generation with grounded models for embodied agents. *Advances in Neural Information Processing Systems*, 36, 2024.

[48] F. Huo, W. Xu, Z. Zhang, H. Wang, Z. Chen, and P. Zhao. Self-introspective decoding: Alleviating hallucinations for large vision-language models. *arXiv preprint arXiv:2408.02032*, 2024.

[49] D. Jang, S. Park, J. Y. Yang, Y. Jung, J. Yun, S. Kundu, S.-Y. Kim, and E. Yang. Lantern: Accelerating visual autoregressive models with relaxed speculative decoding. *arXiv preprint arXiv:2410.03355*, 2024.

[50] P. Kaliosis, J. Pavlopoulos, F. Charalampakos, G. Moschovis, and I. Androutsopoulos. A data-driven guided decoding mechanism for diagnostic captioning. *arXiv preprint arXiv:2406.14164*, 2024.

[51] Z. Kan, C. Zhang, Z. Liao, Y. Tian, W. Yang, J. Xiao, X. Li, D. Jiang, Y. Wang, and Q. Liao. Catch: Complementary adaptive token-level contrastive decoding to mitigate hallucinations in lvlms. *arXiv preprint arXiv:2411.12713*, 2024.

[52] M. Khalifa, H. Elsahar, and M. Dymetman. A distributional approach to controlled text generation. *arXiv preprint arXiv:2012.11635*, 2020.

[53] M. Kim, H. Lee, K. M. Yoo, J. Park, H. Lee, and K. Jung. Critic-guided decoding for controlled text generation. *arXiv preprint arXiv:2212.10938*, 2022.

[54] S. Kim, B. Cho, S. Bae, S. Ahn, and S.-Y. Yun. Vacode: Visual augmented contrastive decoding. *arXiv preprint arXiv:2408.05337*, 2024.

[55] Y. Kim, H. J. Kim, C. Park, C. Park, H. Cho, J. Kim, K. M. Yoo, S.-g. Lee, and T. Kim. Adaptive contrastive decoding in retrieval-augmented generation for handling noisy contexts. *arXiv preprint arXiv:2408.01084*, 2024.

[56] K. Konen, S. Jentzsch, D. Diallo, P. Schütt, O. Bensch, R. E. Baff, D. Opitz, and T. Hecking. Style vectors for steering generative large language model. *arXiv preprint arXiv:2402.01618*, 2024.

[57] H. Lee, J. Kim, H. Chang, H. Oh, S. Yang, V. Karpukhin, Y. Lu, and M. Seo. Nonparametric decoding for generative retrieval. *arXiv preprint arXiv:2210.02068*, 2022.

[58] Y.-L. Lee, Y.-H. Tsai, and W.-C. Chiu. Delve into visual contrastive decoding for hallucination mitigation of large vision-language models. *arXiv preprint arXiv:2412.06775*, 2024.

[59] S. Leng, H. Zhang, G. Chen, X. Li, S. Lu, C. Miao, and L. Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13872–13882, 2024.

[60] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[61] Y. Leviathan, M. Kalman, and Y. Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023.

[62] D. Li, J. Li, H. Li, J. C. Niebles, and S. C. Hoi. Align and prompt: Video-and-language pre-training with entity prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4953–4963, 2022.

[63] H.-S. Li, P. Fernandes, I. Gurevych, and A. F. Martins. Doce: Finding the sweet spot for execution-based code generation. *arXiv preprint arXiv:2408.13745*, 2024.

[64] K. Li, O. Patel, F. Viégas, H. Pfister, and M. Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024.

[65] X. Li, W. Chu, Y. Wu, W. Yuan, F. Liu, Q. Zhang, F. Li, H. Feng, E. Ding, and J. Wang. Videogen: A reference-guided latent diffusion approach for high definition text-to-video generation. *arXiv preprint arXiv:2309.00398*, 2023.

[66] X. L. Li, A. Holtzman, D. Fried, P. Liang, J. Eisner, T. Hashimoto, L. Zettlemoyer, and M. Lewis. Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*, 2022.

[67] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[68] Y. Li, F. Wei, C. Zhang, and H. Zhang. Eagle-2: Faster inference of language models with dynamic draft trees. *arXiv preprint arXiv:2406.16858*, 2024.

[69] Y. Li, F. Wei, C. Zhang, and H. Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024.

[70] X. Liang, H. Wang, Y. Wang, S. Song, J. Yang, S. Niu, J. Hu, D. Liu, S. Yao, F. Xiong, and Z. Li. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*, 2024.

[71] Z. Lin, S. Basu, M. Beigi, V. Manjunatha, R. A. Rossi, Z. Wang, Y. Zhou, S. Balasubramanian, A. Zarei, K. Rezaei, et al. A survey on mechanistic interpretability for multi-modal foundation models. *arXiv preprint arXiv:2502.17516*, 2025.

[72] A. Liu, H. Bai, Z. Lu, X. Kong, S. Wang, J. Shan, M. Cao, and L. Wen. Direct large language model alignment through self-rewarding contrastive prompt distillation. *arXiv preprint arXiv:2402.11907*, 2024.

[73] A. Liu, M. Sap, X. Lu, S. Swayamdipta, C. Bhagavatula, N. A. Smith, and Y. Choi. Dexperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*, 2021.

[74] J. Liu, A. Cohen, R. Pasunuru, Y. Choi, H. Hajishirzi, and A. Celikyilmaz. Making ppo even better: Value-guided monte-carlo tree search decoding. *arXiv preprint arXiv:2309.15028*, 2023.

[75] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[76] S. Liu, H. Ye, L. Xing, and J. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*, 2023.

[77] T. Liu, Z. Chen, Z. Liu, M. Tian, and W. Luo. Expediting and elevating large language model reasoning via hidden chain-of-thought decoding. *arXiv preprint arXiv:2409.08561*, 2024.

[78] X. Liu, L. Hu, P. Bailis, A. Cheung, Z. Deng, I. Stoica, and H. Zhang. Online speculative decoding. *arXiv preprint arXiv:2310.07177*, 2023.

[79] X. Liu, B. Lei, R. Zhang, and D. Xu. Adaptive draft-verification for efficient large language model decoding. *arXiv preprint arXiv:2407.12021*, 2024.

[80] Y. Liu, J. I. Hamid, A. Xie, Y. Lee, M. Du, and C. Finn. Bidirectional decoding: Improving action chunking via closed-loop resampling. *arXiv preprint arXiv:2408.17355*, 2024.

[81] X. Lu, S. Welleck, P. West, L. Jiang, J. Kasai, D. Khashabi, R. L. Bras, L. Qin, Y. Yu, R. Zellers, et al. Neurologic a* esque decoding: Constrained text generation with lookahead heuristics. *arXiv preprint arXiv:2112.08726*, 2021.

[82] X. Lu, P. West, R. Zellers, R. L. Bras, C. Bhagavatula, and Y. Choi. Neurologic decoding:(un) supervised neural text generation with predicate logic constraints. *arXiv preprint arXiv:2010.12884*, 2020.

[83] Y. Lu, M. Bartolo, A. Moore, S. Riedel, and P. Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*, 2021.

[84] W. Luo, F. Song, W. Li, G. Peng, S. Wei, and H. Wang. Odysseus navigates the sirens' song: Dynamic focus decoding for factual and diverse open-ended text generation. *arXiv preprint arXiv:2503.08057*, 2025.

[85] K. Meng, D. Bau, A. Andonian, and Y. Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[86] K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, and D. Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

[87] A. Meta. Introducing llama 3.1: Our most capable models to date. *Meta AI Blog*, 12, 2024.

[88] X. Miao, G. Oliaro, Z. Zhang, X. Cheng, Z. Wang, Z. Zhang, R. Y. Y. Wong, A. Zhu, L. Yang, X. Shi, et al. Specinfer: Accelerating generative large language model serving with tree-based speculative inference and verification. *arXiv preprint arXiv:2305.09781*, 2023.

[89] K. Min, M. Kim, K.-i. Lee, D. Lee, and K. Jung. Mitigating hallucinations in large vision-language models via summary-guided decoding. *arXiv preprint arXiv:2410.13321*, 2024.

[90] K. Murray and D. Chiang. Correcting length bias in neural machine translation. *arXiv preprint arXiv:1808.10006*, 2018.

[91] A. Naseh, K. Krishna, M. Iyyer, and A. Houmansadr. Stealing the decoding algorithms of language models. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1835–1849, 2023.

[92] A. Ni, S. Iyer, D. Radev, V. Stoyanov, W.-t. Yih, S. Wang, and X. V. Lin. Lever: Learning to verify language-to-code generation with execution. In *International Conference on Machine Learning*, pages 26106–26128. PMLR, 2023.

[93] Y. Nie, C. Wang, K. Wang, G. Xu, G. Xu, and H. Wang. Decoding secret memorization in code llms through token-level characterization. *arXiv preprint arXiv:2410.08858*, 2024.

[94] T. Niu, C. Xiong, S. Yavuz, and Y. Zhou. Parameter-efficient detoxification with contrastive decoding. *arXiv preprint arXiv:2401.06947*, 2024.

[95] S. O'Brien and M. Lewis. Contrastive decoding improves reasoning in large language models. *arXiv preprint arXiv:2309.09117*, 2023.

[96] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[97] K. Pal, J. Sun, A. Yuan, B. C. Wallace, and D. Bau. Future lens: Anticipating subsequent tokens from a single hidden state. *arXiv preprint arXiv:2311.04897*, 2023.

[98] Y. Park, D. Lee, J. Choe, and B. Chang. Convis: Contrastive decoding with hallucination visualization for mitigating hallucinations in multimodal large language models. *arXiv preprint arXiv:2408.13906*, 2024.

[99] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In M. Walker, H. Ji, and A. Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[100] M. Peyrard, M. Josifoski, and R. West. The era of semantic decoding. *arXiv preprint arXiv:2403.14562*, 2024.

[101] P. Phan, H. Tran, and L. Phan. Distillation contrastive decoding: Improving llms reasoning with contrastive decoding and distillation. *arXiv preprint arXiv:2402.14874*, 2024.

[102] S. Pimparkhede, M. Kammakomati, S. Tamilselvam, P. Kumar, A. P. Kumar, and P. Bhattacharyya. Doccgen: Document-based controlled code generation. *arXiv preprint arXiv:2406.11925*, 2024.

[103] C. Qian, J. Zhang, W. Yao, D. Liu, Z. Yin, Y. Qiao, Y. Liu, and J. Shao. Towards tracing trustworthiness dynamics: Revisiting pre-training period of large language models. *arXiv preprint arXiv:2402.19465*, 2024.

[104] Z. Qiu, Z. Ou, B. Wu, J. Li, A. Liu, and I. King. Entropy-based decoding for retrieval-augmented large language models. *arXiv preprint arXiv:2406.17519*, 2024.

[105] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. *Improving language understanding with unsupervised learning*, 2018.

[106] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[107] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[108] A. Santilli, S. Severino, E. Postolache, V. Maiorca, M. Mancusi, R. Marin, and E. Rodolà. Accelerating transformer inference for translation via parallel decoding. *arXiv preprint arXiv:2305.10427*, 2023.

[109] M. Sclar, Y. Choi, Y. Tsvetkov, and A. Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*, 2023.

[110] M. Sharma, M. Tong, T. Korbak, D. Duvenaud, A. Askell, S. R. Bowman, N. Cheng, E. Durmus, Z. Hatfield-Dodds, S. R. Johnston, et al. Towards understanding sycophancy in language models. *arXiv preprint arXiv:2310.13548*, 2023.

[111] E. Shen, A. Fan, S. M. Pratt, J. S. Park, M. Wallingford, S. M. Kakade, A. Holtzman, R. Krishna, A. Farhadi, and A. Kusupati. Superposed decoding: Multiple generations from a single autoregressive inference pass. *arXiv preprint arXiv:2405.18400*, 2024.

[112] S. Z. Shen, H. Lang, B. Wang, Y. Kim, and D. Sontag. Learning to decode collaboratively with multiple language models. *arXiv preprint arXiv:2403.03870*, 2024.

[113] T. Shen, R. Jin, Y. Huang, C. Liu, W. Dong, Z. Guo, X. Wu, Y. Liu, and D. Xiong. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025*, 2023.

[114] C. Shi, C. Yang, X. Zhu, J. Wang, T. Wu, S. Li, D. Cai, Y. Yang, and Y. Meng. Unchosen experts can contribute too: Unleashing moe models' power by self-contrast. *arXiv preprint arXiv:2405.14507*, 2024.

[115] C. Shi, H. Yang, D. Cai, Z. Zhang, Y. Wang, Y. Yang, and W. Lam. A thorough examination of decoding methods in the era of llms. *arXiv preprint arXiv:2402.06925*, 2024.

[116] W. Shi, X. Han, M. Lewis, Y. Tsvetkov, L. Zettlemoyer, and S. W.-t. Yih. Trusting your evidence: Hallucinate less with context-aware decoding. *arXiv preprint arXiv:2305.14739*, 2023.

[117] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[118] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

[119] M. Stern, N. Shazeer, and J. Uszkoreit. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31, 2018.

[120] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

[121] Q. Su, W. Zhao, X. Li, M. Andoorveedu, C. Jiang, Z. Zhu, K. Song, C. Giannoula, and G. Pekhimenko. Seesaw: High-throughput llm inference via model resharding. *arXiv preprint arXiv:2503.06433*, 2025.

[122] Y. Su and N. Collier. Contrastive search is what you need for neural text generation. *arXiv preprint arXiv:2210.14140*, 2022.

[123] Y. Su, T. Lan, Y. Wang, D. Yogatama, L. Kong, and N. Collier. A contrastive framework for neural text generation. *Advances in Neural Information Processing Systems*, 35:21548–21561, 2022.

[124] N. Subramani, N. Suresh, and M. E. Peters. Extracting latent steering vectors from pretrained language models. *arXiv preprint arXiv:2205.05124*, 2022.

[125] L. Sun, Y. Huang, H. Wang, S. Wu, Q. Zhang, C. Gao, Y. Huang, W. Lyu, Y. Zhang, X. Li, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 2024.

[126] M. Sun, W. Wang, Z. Qin, J. Sun, S. Chen, and J. Liu. Glober: coherent non-autoregressive video generation via global guided video decoder. *Advances in Neural Information Processing Systems*, 36, 2024.

[127] X. Sun, T. Ge, F. Wei, and H. Wang. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*, 2021.

[128] Z. Sun, Y. Liu, J. Wang, F. Meng, J. Xu, Y. Chen, and J. Zhou. Outdated issue aware decoding for factual knowledge editing. *arXiv preprint arXiv:2406.02882*, 2024.

[129] S. Szyller, B. G. Atli, S. Marchal, and N. Asokan. Dawn: Dynamic adversarial watermarking of neural networks. In *Proceedings of the 29th ACM international conference on multimedia*, pages 4417–4425, 2021.

[130] H. Tang, Y. Wu, S. Yang, E. Xie, J. Chen, J. Chen, Z. Zhang, H. Cai, Y. Lu, and S. Han. Hart: Efficient visual generation with hybrid autoregressive transformer. *arXiv preprint arXiv:2410.10812*, 2024.

[131] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[132] Y. Teng, H. Shi, X. Liu, X. Ning, G. Dai, Y. Wang, Z. Li, and X. Liu. Accelerating auto-regressive text-to-image generation with training-free speculative jacobi decoding. *arXiv preprint arXiv:2410.01699*, 2024.

[133] I. Tenney. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

[134] Y. Tian and T. Zhang. Selective prompt anchoring for code generation. *arXiv preprint arXiv:2408.09121*, 2024.

[135] C. Tu, P. Ye, D. Zhou, L. Bai, G. Yu, T. Chen, and W. Ouyang. Attention reallocation: Towards zero-cost and controllable hallucination mitigation of mllms. *arXiv preprint arXiv:2503.08342*, 2025.

[136] A. M. Turner, L. Thiergart, G. Leech, D. Udell, J. J. Vazquez, U. Mini, and M. MacDiarmid. Activation addition: Steering language models without optimization. *arXiv e-prints*, pages arXiv–2308, 2023.

[137] B. Upadhyay, A. Sudhakar, and A. Maheswaran. Efficient reinforcement learning for unsupervised controlled text generation. *arXiv preprint arXiv:2204.07696*, 2022.

[138] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and D. Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

[139] C. Wang, X. Chen, N. Zhang, B. Tian, H. Xu, S. Deng, and H. Chen. Mllm can see? dynamic correction decoding for hallucination mitigation. *arXiv preprint arXiv:2410.11779*, 2024.

[140] H. Wang and K. Shu. Explainable claim verification via knowledge-grounded reasoning with large language models. *arXiv preprint arXiv:2310.05253*, 2023.

[141] H. Wang and K. Shu. Trojan activation attack: Red-teaming large language models using steering vectors for safety-alignment. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2347–2357, 2024.

[142] H. Wang, B. Wu, Y. Bian, Y. Chang, X. Wang, and P. Zhao. Probing the safety response boundary of large language models via unsafe decoding path generation. *arXiv preprint arXiv:2408.10668*, 2024.

[143] H. Wang, X. Xu, B. Huang, and K. Shu. Privacy-aware decoding: Mitigating privacy leakage of large language models in retrieval-augmented generation. *arXiv preprint arXiv:2508.03098*, 2025.

[144] J. Wang, Y. Gao, and J. Sang. Valid: Mitigating the hallucination of large vision language models by visual layer fusion contrastive decoding. *arXiv preprint arXiv:2411.15839*, 2024.

[145] P. Wang, D. Zhang, L. Li, C. Tan, X. Wang, K. Ren, B. Jiang, and X. Qiu. Inferaligner: Inference-time alignment for harmlessness through cross-model guidance. *arXiv preprint arXiv:2401.11206*, 2024.

[146] S. Wang, L. Ding, L. Shen, Y. Luo, Z. He, W. Yu, and D. Tao. Uscd: Improving code generation of llms by uncertainty-aware selective contrastive decoding. *arXiv preprint arXiv:2409.05923*, 2024.

[147] T. Wang, R. Zhang, Z. Lu, F. Zheng, R. Cheng, and P. Luo. End-to-end dense video captioning with parallel decoding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6847–6857, 2021.

[148] X. Wang, J. Pan, L. Ding, and C. Biemann. Mitigating hallucinations in large vision-language models with instruction contrastive decoding. *arXiv preprint arXiv:2403.18715*, 2024.

[149] X. Wang and D. Zhou. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*, 2024.

[150] Z. Wang, Z. Wang, L. Le, H. S. Zheng, S. Mishra, V. Perot, Y. Zhang, A. Mattapalli, A. Taly, J. Shang, et al. Speculative rag: Enhancing retrieval augmented generation through drafting. *arXiv preprint arXiv:2407.08223*, 2024.

[151] Z. Wang, J. Wu, Y. Lai, C. Zhang, and D. Zhou. Seed: Accelerating reasoning tree construction via scheduled speculative decoding. *arXiv preprint arXiv:2406.18200*, 2024.

[152] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[153] S. Welleck, J. Liu, X. Lu, H. Hajishirzi, and Y. Choi. Naturalprover: Grounded mathematical proof generation with language models. *Advances in Neural Information Processing Systems*, 35:4913–4927, 2022.

[154] G. Wiher, C. Meister, and R. Cotterell. On decoding strategies for neural text generators. *Transactions of the Association for Computational Linguistics*, 10:997–1012, 2022.

[155] H. Xia, T. Ge, P. Wang, S.-Q. Chen, F. Wei, and Z. Sui. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3909–3925, 2023.

[156] H. Xia, Y. Li, J. Zhang, C. Du, and W. Li. Swift: On-the-fly self-speculative decoding for llm inference acceleration. *arXiv preprint arXiv:2410.06916*, 2024.

[157] H. Xia, Z. Yang, Q. Dong, P. Wang, Y. Li, T. Ge, T. Liu, W. Li, and Z. Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*, 2024.

[158] H. Xia, Z. Yang, Q. Dong, P. Wang, Y. Li, T. Ge, T. Liu, W. Li, and Z. Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7655–7671, Bangkok, Thailand, Aug. 2024. Association for Computational Linguistics.

[159] Y. Xie, K. Kawaguchi, Y. Zhao, X. Zhao, M.-Y. Kan, J. He, and Q. Xie. Self-evaluation guided beam search for reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[160] D. Xu, Z. Zhang, Z. Zhu, Z. Lin, Q. Liu, X. Wu, T. Xu, X. Zhao, Y. Zheng, and E. Chen. Mitigating hallucinations of large language models in medical information extraction via contrastive decoding. *arXiv preprint arXiv:2410.15702*, 2024.

[161] Z. Xu, F. Jiang, L. Niu, J. Jia, B. Y. Lin, and R. Poovendran. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *arXiv preprint arXiv:2402.08983*, 2024.

[162] D. Yang, D. Xiao, J. Wei, M. Li, Z. Chen, K. Li, and L. Zhang. Improving factuality in large language models via decoding-time hallucinatory and truthful comparators. *arXiv preprint arXiv:2408.12325*, 2024.

[163] H. Yang, D. Cai, H. Li, W. Bi, W. Lam, and S. Shi. A frustratingly simple decoding method for neural text generation. *arXiv preprint arXiv:2305.12675*, 2023.

[164] K. Yang and D. Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

[165] Y. Yang, L. Huang, and M. Ma. Breaking the beam search curse: A study of (re-) scoring methods and stopping criteria for neural machine translation. *arXiv preprint arXiv:1808.09582*, 2018.

[166] H. Yuan, K. Lu, F. Huang, Z. Yuan, and C. Zhou. Speculative contrastive decoding. *arXiv preprint arXiv:2311.08981*, 2023.

[167] Y. Zeldes, D. Padnos, O. Sharir, and B. Peleg. Technical report: Auxiliary tuning and its application to conditional text generation. *arXiv preprint arXiv:2006.16823*, 2020.

[168] H. Zeng, C. Luo, and H. Zamani. Planning ahead in generative retrieval: Guiding autoregressive generation through simultaneous decoding. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 469–480, 2024.

[169] X. Zeng, Y. Shang, Y. Zhu, J. Chen, and Y. Tian. Root defence strategies: Ensuring safety of llm at the decoding level. *arXiv preprint arXiv:2410.06809*, 2024.

[170] Y. Zeng, G. Liu, W. Ma, N. Yang, H. Zhang, and J. Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.

[171] H. Zhang, S. Si, H. Wu, and D. Song. Controllable text generation with residual memory transformer. *arXiv preprint arXiv:2309.16231*, 2023.

[172] H. Zhang and D. Song. Discup: Discriminator cooperative unlikelihood prompt-tuning for controllable text generation. *arXiv preprint arXiv:2210.09551*, 2022.

[173] J. Zhang, J. Wang, H. Li, L. Shou, K. Chen, G. Chen, and S. Mehrotra. Draft & verify: Lossless large language model acceleration via self-speculative decoding. *arXiv preprint arXiv:2309.08168*, 2023.

[174] S. Zhang, Z. Chen, Y. Shen, M. Ding, J. B. Tenenbaum, and C. Gan. Planning with large language models for code generation. *arXiv preprint arXiv:2303.05510*, 2023.

[175] S. Zhang, L. Pan, J. Zhao, and W. Y. Wang. The knowledge alignment problem: Bridging human and external knowledge for large language models. *arXiv preprint arXiv:2305.13669*, 2023.

[176] S. Zhang, T. Yu, and Y. Feng. Truthx: Alleviating hallucinations by editing large language models in truthful space. *arXiv preprint arXiv:2402.17811*, 2024.

[177] X. Zhang and X. Wan. Mil-decoding: Detoxifying language models at token-level via multiple instance learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 190–202, 2023.

[178] Z. Zhao, E. Monti, J. Lehmann, and H. Assem. Enhancing contextual understanding in large language models through contrastive decoding. *arXiv preprint arXiv:2405.02750*, 2024.

[179] Z. Zhao, X. Zhang, K. Xu, X. Hu, R. Zhang, Z. Du, Q. Guo, and Y. Chen. Adversarial contrastive decoding: Boosting safety alignment of large language models via opposite prompt optimization. *arXiv preprint arXiv:2406.16743*, 2024.

[180] R. Zheng and L. Sun. Llm4brain: Training a large language model for brain video understanding. *arXiv preprint arXiv:2409.17987*, 2024.

[181] X. Zheng, H. Lin, X. Han, and L. Sun. Toward unified controllable text generation via regular expression instruction. *arXiv preprint arXiv:2309.10447*, 2023.

[182] Q. Zhong, L. Ding, J. Liu, B. Du, and D. Tao. Rose doesn't do that: Boosting the safety of instruction-tuned large language models with reverse prompt contrastive decoding. *arXiv preprint arXiv:2402.11889*, 2024.

[183] W. Zhou, Y. E. Jiang, E. Wilcox, R. Cotterell, and M. Sachan. Controlled text generation with natural language instructions. In *International Conference on Machine Learning*, pages 42602–42613. PMLR, 2023.

[184] Y. Zhou, K. Lyu, A. S. Rawat, A. K. Menon, A. Rostamizadeh, S. Kumar, J.-F. Kagy, and R. Agarwal. Distillspec: Improving speculative decoding via knowledge distillation. *arXiv preprint arXiv:2310.08461*, 2023.

[185] L. Zhu, D. Ji, T. Chen, P. Xu, J. Ye, and J. Liu. Ibd: Alleviating hallucinations in large vision-language models via image-biased decoding. *arXiv preprint arXiv:2402.18476*, 2024.

[186] T. Zhu, K. Zhang, J. Xie, and Y. Su. Deductive beam search: Decoding deducible rationale for chain-of-thought reasoning. *arXiv preprint arXiv:2401.17686*, 2024.

[187] Y. Zhu, J. A. Li, G. Li, Y. Zhao, J. Li, Z. Jin, and H. Mei. Improving code generation by dynamic temperature sampling. *arXiv preprint arXiv:2309.02772*, 2023.

[188] Y. Zhu, X. Yang, Y. Wu, and W. Zhang. Hierarchical skip decoding for efficient autoregressive text generation. *arXiv preprint arXiv:2403.14919*, 2024.