University of Trento

Department of Computer Science

PhD Degree in Computer Science and Engineering

~ · ~

Academic Year 2025–2026

Active Evaluation for Generative AI: Towards an Adaptive Lifecycle

Supervisor
Prof. Nicu Sebe

PhD Student
Federico Betti
241293

Final examination date: February Xth, 2026

to a life of constant learning and growth

Acknowledgments

My first and most heartfelt thanks go to Professor Nicu Sebe for his constant support and incredible availability. He has been a mentor who allowed me the freedom to explore my ideas while always providing precise, insightful feedback and the experience necessary to bring this work to completion and achieve the best possible results.

A huge thank you goes to all the people who have been close and helpful throughout this journey, especially to Lorenzo and Davide, with whom we carried out very interesting work. And a really special thanks to Professor Baraldi, Professor Cornia and Professor Staiano for their huge help and valuable advice.

I am also incredibly grateful to David, George, Martina, Luca, and everyone at Typewise, as they allowed me to complete this adventure and bring this new knowledge into the company. A special thank you goes to Janis; I feel I have learned so much from you. Your openness to change and new opinions, your passion, curiosity, energy, and warmth toward others will always be a source of inspiration for me.

I am also deeply grateful to my friends across Europe, who were fundamental to finding a balance between work and everything else: Lorenzo, Donatella, Valerio, Marco, and Gabriele from Zurich. To all the guys in Modena, Andrea, Andrea, Edoardo, Lorenzo, Matteo, Michele, Tommaso, and everyone else, thank you for your friendship, our trips, and the many fun moments we shared.

My deepest and warmest thanks go to my family. You have been a vital support, helping me persevere through the most difficult moments. Thank you to Vittoria and Stefano. However, in this case, a very special mention must go to my mother, Rita. Now, we can finally discuss the boat... You are all a constant inspiration to me, and your presence has profoundly shaped my life.

Noemi, you have been by my side not only throughout this path but also in everything that came before, and hopefully in all those to come. You have been present in all kinds of moments, from the most difficult to the most fun. You have helped me push forward, making me believe in myself and encouraging me to always aim for greater goals and achievements. I am truly grateful to have you by my side.

Abstract

Generative Artificial Intelligence, particularly text-to-image models, has achieved remarkable progress, yet this rapid scaling has introduced critical bottlenecks in reliability, controllability, and computational sustainability. Current evaluation protocols, typically relying on post-hoc global metrics such as LLM-based scores, are insufficient for diagnosing fine-grained failures or guiding resource-efficient generation. This thesis argues for an *evaluation-centric lifecycle*, in which evaluation mechanisms are embedded directly into the generative pipeline to actively monitor, verify, and refine model behaviour.

Concretely, we instantiate this lifecycle with three methodological contributions that act at distinct stages of the image generation and editing process. Hallucination Early Detection (HEaD) monitors internal consistency during the diffusion process to anticipate semantic failures and enable early stopping or resampling, reducing unnecessary compute. Visual Concept Evaluation (ViCE) performs post-hoc, concept-level verification by decomposing prompts into atomic visual concepts and using visual question answering to diagnose specific failures, producing interpretable, concept-level explanations of where and why a generation fails, rather than a single aggregate score. These explanations enable practitioners to identify systematic weaknesses and to target improvements at specific visual concepts. Finally, addressing the editing domain, Differential Evaluation of Localised Image Edits (DICE) jointly detects what has changed under instruction-guided editing and assesses whether those changes align with the user's intent.

Unified by three cross-cutting pillars of explainability, granularity, and sustainability, these contributions demonstrate that treating evaluation as a first-class component enables the development of generative systems that are more robust and aligned with human intent. By moving from opaque global scores to structured, actionable feedback at the level of individual generations and edits, we establish a closed loop in which evaluation directly informs and improves the generative process.

Contents

Publications

This page lists the peer-reviewed publications on which this thesis is based and introduces the acronyms used to refer to them throughout the manuscript. Publications are ordered chronologically by year.

ViCE F. Betti, J. Staiano, L. Baraldi, L. Baraldi, R. Cucchiara, N. Sebe. "Let's ViCE! Mimicking Human Cognitive Behavior in Image Generation Evaluation." In: *Proceedings of the 31st ACM International Conference on Multimedia (ACM MM)*, 2023. Related thesis chapter: Chapter 4.

HEaD F. Betti*, L. Baraldi*, L. Baraldi, R. Cucchiara, N. Sebe. "Optimizing Resource Consumption in Diffusion Models through Hallucination Early Detection." In: *Proceedings of the European Conference on Computer Vision Workshops (ECCV Workshops)*, 2024. Related thesis chapter: Chapter 3.

DICE L. Baraldi*, D. Bucciarelli*, F. Betti*, M. Cornia, L. Baraldi, N. Sebe, R. Cucchiara. "*What Changed?* Detecting and Evaluating Instruction-Guided Image Edits with Multimodal Large Language Models." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. Related thesis chapter: Chapter 5.

HEAD+ F. Betti*, L. Baraldi*, L. Baraldi, R. Cucchiara, N. Sebe. "Hallucination Early Detection in Diffusion Models." *International Journal of Computer Vision (IJCV)*, 2025. Related thesis chapter: Chapter 3.

*Equal contribution.

Nomenclature

DICE	Differential Evaluation of Localised Image Edits
FID	Fréchet Inception Distance
GAN	Generative Adversarial Network
HEaD	Hallucination Early Detection
HP	Hallucination Prediction
InsideGen	Dataset for Hallucination Early Detection
IS	Inception Score
LLM	Large Language Model
LLM	Large Language Model
MLLM	Multimodal Large Language Model
PFI	Predicted Final Image
RoIs	Regions of Interest
VAE	Variational Autoencoder
ViCE	Visual Concept Evaluation
ViT	Vision Transformer
VQA	Visual Question Answering
HEaD+	Hallucination Early Detection+

Chapter 1

Introduction

Context and Motivation

Generative Artificial Intelligence (AI) has rapidly become a visible, contested, and increasingly indispensable part of contemporary digital life. Text and image generators are now embedded in search engines, productivity suites, creative software, and mobile devices; conversational agents mediate access to information; and new tools for code, design, and content creation are being deployed at scale across industries. Entire ecosystems of start-ups and established companies are building products, services, and decision pipelines on top of model outputs, and few sectors remain untouched by this transformation. What was recently a specialised research area has thus become a global, everyday technology that shapes how individuals work, create, and interact with information.

This broad adoption is enabled by a period of exceptional technical progress in generative modelling. Over roughly the past decade, models have evolved from the early success of Generative Adversarial Networks (GANs) to diffusion-based architectures and large-scale transformers that power state-of-the-art systems across modalities [38, 119, 45, 98, 100]. In just a few years, systems that were once fragile prototypes have become robust, general-purpose generators of text, images, audio, and video. In parallel, multimodal large language models (MLLMs) have begun to unify perception and reasoning, broadening the scope of tasks for which generation is both practical and useful [84].

Yet scale and capability bring new burdens. As model sizes and training corpora have grown, so too have the computational and energy costs of both generation and evaluation [115, 29]. Contemporary uses of generation are often open-ended, instruction-driven, or highly personalised, which increases the variability of outputs and the difficulty of assessing their quality, safety, and faithfulness in a systematic way. At the same time, the societal role of these systems makes it essential to monitor their progress and limitations: practitioners, regulators, and end users alike require independent, reliable, and comparable evaluations in order to choose between models, understand failure modes, and decide both when and for what purposes these systems can be trusted.

This raises a broader set of questions around the *sustainability*, *reproducibility*, and *explainability* of generative AI. On the one hand, the continuous evaluation that accompanies model development and deployment must itself be resource-aware, so that monitoring does not become prohibitively expensive or environmentally costly. On the other hand, evaluation protocols need to produce signals that are transparent, reproducible across datasets and model families, and sufficiently interpretable to feed back into model design, training, and governance. Without such signals, it is difficult to close the loop between observed behaviour "in the wild" and principled improvements to the underlying systems.

The central, cross-cutting question of this thesis is therefore how to evaluate generative systems in a way that is rigorous, scalable, and sustainable, while remaining practically useful for improving both models and their deployment. More concretely, we seek evaluation frameworks that can be embedded into the full lifecycle of generative models, from sampling to editing and post-hoc analysis, and that help build calibrated trust in their behaviour for end users and stakeholders who increasingly rely on these systems as part of their everyday tools.

From Evaluation to an End-to-End Lifecycle: The Thesis Trajectory

The research arc of this thesis is organised around the *journey of an image* through a modern text-to-image system, tracing the path from an initial, provisional sample towards a final result that matches what the user ultimately expects and wants. As shown in Figure 1.1,

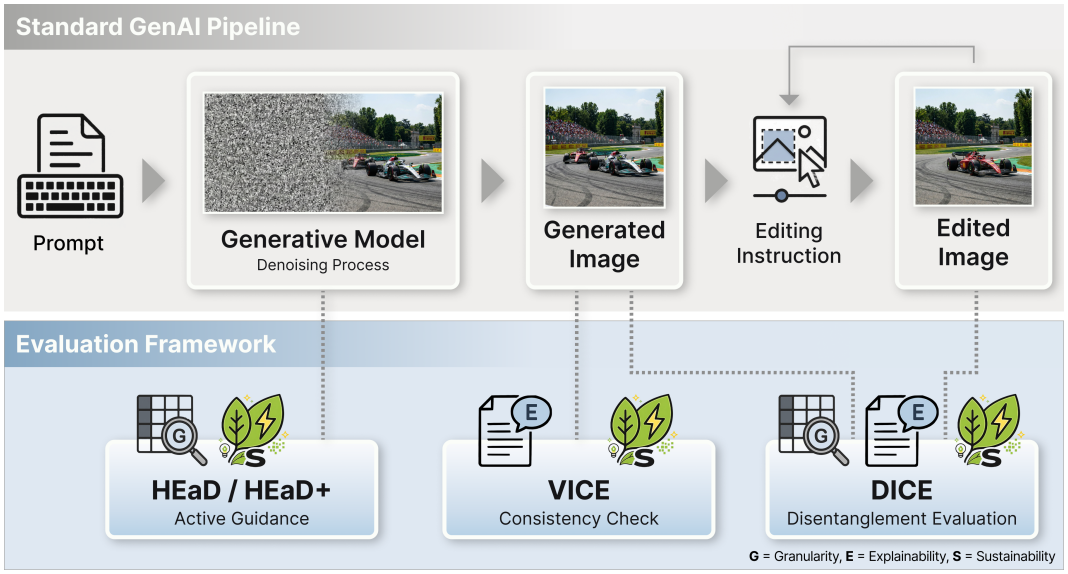


Figure 1.1: Proposed evaluation-centric lifecycle for generative AI. The top layer depicts the standard text-to-image pipeline from prompt to edited image. The bottom layer shows the evaluation framework introduced in this thesis: HEaD provides *active guidance* inside the denoising process, ViCE performs a *consistency check* on the generated image, and DICE delivers *disentanglement evaluation* by jointly analysing the generated and edited images.

starting from a natural language *prompt*, the model performs a *generative process* to produce a *generated image*; users may then issue one or more *editing instructions* to obtain a sequence of *edited images*, iteratively refining the output until it aligns with their target intent. Designing systems that robustly support this potentially long editing loop is a major open challenge in the community, and a focus of intense effort from both academia and large industrial players. Rather than treating evaluation as a single number computed at the end of this pipeline, we view it as an *evaluation-centric lifecycle* that accompanies the image at multiple stages: guiding the generative process, auditing the generated image, and verifying that edits are both correct and well-localised.

Within this lifecycle, the contributions of the thesis occupy three complementary roles. First, we move from traditional, offline evaluation to *active guidance* during generation. In Hallucination-Aware Control of Text-to-Image Generation, we bring together two consecutive works on Hallucination Early Detection. These methods monitor consistency signals during the denoising process itself, anticipating semantic failures so that trajectories likely to hallucinate can be truncated or resampled.

Second, we study how to *diagnose* whether a generative model is structurally reliable on its own outputs. In Human-Aligned Evaluation for Text-to-Image Models, we introduce a visual concept evaluation framework that mimics human questioning and verification to probe whether generated content satisfies natural-language instructions. The approach emphasizes faithfulness and controllability rather than superficial similarity, drawing on insights from VQA-style checks and instruction-following benchmarks [47, 43], and yields interpretable signals that can be reproduced and scaled across datasets and models [8].

Finally, we address the more complex setting of instruction-guided image editing, where models must apply precise, localized changes while preserving irrelevant content. In Differential Evaluation of Localised Image Edits, we develop an integrated framework that couples instruction-following evaluation with change detection, jointly assessing what changed and whether the change was correct. As illustrated in Figure 1.1, DICE sits at the final stage of the lifecycle, jointly analysing the generated and edited images to assess disentanglement. The framework interfaces naturally with emerging editing models [11, 54, 83] and with new datasets and evaluation protocols for editing [133, 49, 51].

Across these works, three methodological choices recur and lay the groundwork for the paradigm shift we develop throughout the thesis: *explainability*, *granularity*, and *sustainability*. We move beyond opaque scalar scores to structured rationales and natural language feedback, enabling evaluation to act as an interpretable signal for model improvement rather than a black box verdict. We decompose evaluation into specific components, including temporal dynamics in diffusion, visual concepts in prompts, and localized regions in editing, to achieve a level of precision that global metrics lack. Finally, we treat compute as a first-class constraint, designing methods that enable early stopping or reuse lightweight adapters on top of a shared backbone so that evaluation remains scalable and memory-efficient. We return to these three pillars in the conclusion, where we argue that they are essential ingredients for an evaluation-in-the-loop paradigm for generative AI.

Beyond the academic setting, part of this thesis was also influenced by an industrial perspective. During my PhD, I completed an internship at Typewise, a Swiss AI start-up backed by Y Combinator and collaborating with ETH Zurich, developing AI agents to support human operators in customer care. The internship specifically focused on evaluation of MLLM-based systems in a customer-care setting, where the model output is textual

rather than visual. However, the underlying evaluation principles are the same: the need for reproducible and cost-effective experimental pipelines, and for model-agnostic evaluation signals that enable rapid comparisons and safe model updates. It also reinforced the value of granular checks that go beyond overall output quality by verifying specific requirements (e.g., whether key questions were answered, language constraints were met, and response structure followed domain conventions), while accounting for the non-determinism of large generative models.

Chapter 2

Preliminaries & Background

This chapter introduces the technical background underpinning the rest of the thesis. It provides a unified set of definitions, notation and conceptual tools that will be reused throughout the following chapters, so that later discussions can focus on the specific evaluation, detection and editing contributions without repeatedly rederiving the same preliminaries.

2.1 Diffusion Models

Diffusion models are a class of generative models that learn to synthesise data by reversing a gradual noising process [112, 45, 26]. In this section, we summarise the forward and reverse processes, the standard training objective, latent diffusion and text conditioning, guidance mechanisms, and computational aspects that are particularly relevant for large-scale image generation.

2.1.1 Forward and Reverse Diffusion Processes

Let $x_0 \sim q(x_0)$ denote a real data point, for instance an image. The forward (or *diffusion*) process gradually perturbs x_0 into a sequence $\{x_t\}_{t=1}^{T}$ by adding Gaussian noise with a pre-defined variance schedule $\{\beta_t\}_{t=1}^{T}$:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\big(x_t; \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t \mathbf{I}\big), \tag{2.1}$$

so that after sufficiently many steps x_T is close to an isotropic Gaussian. Because (2.1) defines a Markov chain with simple Gaussian transitions, one can derive a closed-form expression for x_t given x_0:

$$q(x_t \mid x_0) = \mathcal{N}\big(x_t; \sqrt{\bar{\alpha}_t}\, x_0,\, (1-\bar{\alpha}_t)\mathbf{I}\big), \qquad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s). \tag{2.2}$$

Equivalently, one can sample

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \mathbf{I}), \tag{2.3}$$

which is the form used in practice during training. These equations describe a forward corruption process in which a clean image is gradually mixed with Gaussian noise: for small t the sample still resembles the data, while for large t it becomes almost indistinguishable from pure noise.

The generative model is defined as a reverse-time Markov chain $p_\theta(x_{t-1} \mid x_t)$ that aims to invert the forward process:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t), \qquad p(x_T) = \mathcal{N}(0, \mathbf{I}). \tag{2.4}$$

In the original DDPM formulation [45], each transition is parameterised as a Gaussian whose mean depends on a neural network prediction of the noise added in the forward process. This yields a stable likelihood-based generative model that can produce high-fidelity samples without adversarial training.

At a high level, the reverse chain (2.4) aims to undo the corruption of the forward chain: starting from random noise, the model applies a sequence of denoising steps that gradually transform noise into a plausible image sample.

2.1.2 Training Objective and Noise Prediction

In practice, diffusion models are commonly trained as *noise predictors*. Given a time index t, a clean data point x_0 and Gaussian noise ϵ, the noised sample x_t is computed as in (2.3). The model $\epsilon_\theta(x_t, t)$ is then trained to predict ϵ by minimising a weighted mean-squared

error:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{x_0,t,\epsilon}\big[\lambda_t \, \|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\big], \tag{2.5}$$

where λ_t is a positive weighting term that can be derived from a variational lower bound on the data likelihood [45]. This loss can be interpreted as a form of denoising score matching, since ϵ_θ learns to predict the score $\nabla_{x_t} \log q(x_t \mid x_0)$ up to a scale factor [114].

In simpler terms, the model is shown examples of noisy inputs and is trained to guess the noise that was added; learning to remove this noise is what ultimately allows it to generate new samples by denoising pure noise into realistic images.

At sampling time, one starts from $x_T \sim \mathcal{N}(0, \mathbf{I})$ and iteratively denoises:

$$x_{t-1} = \mu_\theta(x_t, t) + \sigma_t z, \qquad z \sim \mathcal{N}(0, \mathbf{I}), \tag{2.6}$$

where μ_θ is a function of x_t, t and $\epsilon_\theta(x_t, t)$ determined by the forward variance schedule, and σ_t controls the amount of stochasticity. Deterministic samplers (such as DDIM-style updates) can be obtained by setting $\sigma_t = 0$ and choosing an alternative discretisation of the underlying stochastic differential equation [68, 113].

Operationally, the recursion in (2.6) expresses the sampling procedure: at each step the model predicts which part of the current sample looks like noise, subtracts that component, and (optionally) adds a small amount of fresh noise before moving to the next, slightly cleaner sample.

The backbone ϵ_θ is typically a UNet-like convolutional network with residual blocks, self-attention layers and time-embedding mechanisms [99, 45]. For image generation, this architecture provides a good balance between local detail modelling and global context aggregation.

2.1.3 Latent Diffusion Models

While the original DDPM formulation operates in pixel space, high-resolution images make this approach computationally expensive. Latent diffusion models mitigate this issue by running the diffusion process in the latent space of a variational autoencoder (VAE) [55, 98].

Let E and D denote the encoder and decoder of a VAE trained to reconstruct images:

$$z = E(x), \qquad \hat{x} = D(z) \approx x. \tag{2.7}$$

Diffusion is then applied to the latent variable z instead of x. Denoting by z_t the noisy latent at step t, the noise-prediction network operates as

$$\epsilon_\theta(z_t, t, c), \tag{2.8}$$

where c is a conditioning vector (e.g. a text embedding). The training objective becomes

$$\mathcal{L}_{\text{latent}} = \mathbb{E}_{z\sim E(x),c,\epsilon,t}\big[\|\epsilon - \epsilon_\theta(z_t, t, c)\|_2^2\big]. \tag{2.9}$$

After denoising, the final latent z_0 is decoded into pixel space:

$$x_0 = D(z_0). \tag{2.10}$$

Latent diffusion dramatically reduces the spatial dimensionality of the generative process and therefore its memory and compute footprint, while maintaining high perceptual quality [98, 91]. Intuitively, instead of denoising millions of pixels directly, the model learns to denoise a compact representation z; once the latent has been cleaned, the decoder D turns it back into a high-resolution image.

2.1.4 Schedulers and Sampling Efficiency

The choice of sampler (or *scheduler*) governs the sequence of noise levels and update rules used during generation. Many schedulers can be written in terms of an update function Δ that maps a latent z_t at time t to a new latent $z_{t'}$ at time $t' < t$, given the predicted noise:

$$\begin{aligned} \epsilon_t &= \epsilon_\theta(z_t, t, c), \\ z_{t'} &= \Delta(z_t, \epsilon_t, t, t'). \end{aligned} \tag{2.11}$$

Different choices of Δ recover Euler, Heun, or higher-order solvers for the underlying diffusion or rectified-flow dynamics, and enable high-quality sampling in tens of steps rather than

hundreds [28, 90].

Sampling efficiency is critical for large-scale applications. A single denoising trajectory may require dozens of forward passes through a large UNet, each of which is memory intensive. Acceleration techniques include: 1. designing variance schedules that concentrate denoising where it matters most [82], 2. using deterministic samplers with larger step sizes [113, 72], 3. distilling the diffusion process into a smaller number of steps via teacher–student training [101, 138], 4. early stopping strategies that terminate sampling when an image is already sufficiently refined [75, 18].

2.1.5 Text Conditioning and Guidance

Modern text-to-image diffusion models condition the generative process on textual prompts. Let y denote a text description (prompt), and $c(y)$ its embedding obtained through a text encoder, such as CLIP's Transformer-based language encoder [92] or a dedicated T5- or BERT-style model. The conditional noise predictor becomes

$$\epsilon_\theta(z_t, t, c(y)), \tag{2.12}$$

and the reverse updates in (2.6) and (2.11) depend on $c(y)$.

In latent diffusion models such as Stable Diffusion [98, 91], conditioning is injected through *cross-attention* layers. Given a spatial feature map $F_t \in \mathbb{R}^{P\times d}$ from the UNet at time t, and a sequence of text token embeddings $T \in \mathbb{R}^{N\times d}$, cross-attention computes, for each spatial position, a weighted combination of text features:

$$\mathrm{Attn}(F_t, T) = \mathrm{softmax}\left(\frac{Q_F K_T^\top}{\sqrt{d}}\right) V_T, \tag{2.13}$$

where $Q_F = F_t W_Q$, $K_T = TW_K$, $V_T = TW_V$ for learned weight matrices W_Q, W_K, W_V. This mechanism allows the model to align spatial regions with textual concepts and is central to the controllability of text-to-image diffusion.

An important practical technique is *classifier-free guidance* [98, 83], which trades off sample diversity for improved text alignment. During training, the model is exposed to conditional and unconditional inputs, enabling it to predict both $\epsilon_\theta(z_t, t, c(y))$ and $\epsilon_\theta(z_t, t, \varnothing)$.

At sampling time, one forms a guided prediction

$$\tilde{\epsilon}_\theta(z_t, t, y) = \epsilon_\theta(z_t, t, \varnothing) + s\big(\epsilon_\theta(z_t, t, c(y)) - \epsilon_\theta(z_t, t, \varnothing)\big), \tag{2.14}$$

where $s \geq 1$ is the guidance scale. Larger values of s typically improve prompt adherence, at the risk of artefacts such as over-saturated colours or reduced diversity.

2.1.6 Instruction-guided Image Editing with Diffusion Models

Instruction-guided image editing aims to transform an input image x into an edited image x' according to a natural-language instruction y, while preserving the content that is irrelevant to the requested change. This can be formalised as learning or defining a conditional distribution $p_\theta(x' \mid x, y)$ whose samples satisfy two coupled requirements: edit adherence (apply the requested semantic modification) and content preservation (avoid unintended changes and keep identity/background consistent). Compared to text-to-image generation, editing is therefore constrained not only by the prompt, but also by the need to remain faithful to the input image.

Diffusion-based editing methods span several families [49].

Training-based instruction editors. Models such as InstructPix2Pix learn a direct instruction-conditioned editor by training on large collections of (image, instruction, edited-image) triplets, enabling fast edits without per-image optimisation at inference time [11].

Test-time optimisation and personalisation. Given a real image, methods such as Imagic adapt the model (or conditioning) at inference time so as to reconstruct the input and then steer it towards a new textual target [54]; related personalisation approaches learn new token embeddings from a few examples to represent a specific subject or style, which can then be reused for subject-consistent generation and editing [33].

Tuning-free prompt and attention control. Another line of work edits by manipulating the generation trajectory of a pretrained text-to-image model, for example via cross-attention/self-attention control, enabling localized prompt edits without retraining [42, 14].

Noise-based and image-to-image formulations. Finally, image-to-image diffusion editing can be performed by injecting noise into the input and denoising under a new condition; the noise level controls the strength of the edit [79, 89]. Mask-guided variants (inpainting) provide explicit spatial control when only a region of the image should change [3, 74].

While these methods can produce high-quality edits, their evaluation remains challenging because ground-truth edited targets are rarely available and subtle failure modes such as over-editing, under-editing, and unintended side-effects are common, motivating the editing-focused evaluation approach developed in Chapter 5.

2.1.7 Representative Diffusion-based Image Generators

The diffusion framework described above underlies most recent large-scale text-to-image systems. Early models based on pixel-space diffusion demonstrated that these models can match or surpass GANs in image synthesis quality [26, 44]. Latent diffusion brought high-resolution synthesis within reach of commodity hardware [98], and its successors (e.g. SDXL) scaled up UNet capacity, text encoders and training data to further improve photorealism and prompt fidelity [91]. Alternative architectures such as diffusion transformers and rectified-flow models extend these ideas with Transformer backbones and continuous-time flows [28, 90].

Despite their impressive performance, diffusion models remain computationally demanding and inherently stochastic: different random seeds can yield qualitatively different yet plausible images. This stochasticity, together with the complex interaction between text conditioning and image content, motivates the evaluation protocols discussed in Section 2.3.

2.2 Multimodal Large Language Models

Large Language Models (LLMs) trained on text have recently been extended to handle visual inputs, giving rise to MLLMs. These models are capable of processing images and text jointly and generating natural-language outputs that depend on both modalities [84, 13]. We briefly review the main architectural paradigms, representative models, and key challenges, with an emphasis on vision–language understanding and instruction following.

2.2.1 Architectural Paradigms

Current MLLMs can be broadly organised into two families. The first is the *dual-model* paradigm, in which a pretrained vision encoder is connected to a pretrained language model via a lightweight bridging module. The second is the *unified* paradigm, where a single Transformer processes both visual and textual tokens in a joint sequence.

In the dual-model setting, a vision encoder f_{vis} (typically a CNN or Vision Transformer) produces image features $v = f_{\text{vis}}(I)$. These features are transformed into a sequence of *visual tokens* that are compatible with the language model. For instance, a simple linear projection can map v into the embedding space of a decoder-only LLM, where it is concatenated with text tokens and processed autoregressively [67]. More advanced variants introduce an intermediate module (e.g. a cross-attention or query-transformer block) that distils the visual representation into a fixed number of informative tokens [63, 62].

In unified architectures, both images and text are represented as tokens and passed through the same Transformer. One approach embeds image patches as tokens, optionally with modality-specific positional encodings, and uses a causal or bidirectional attention mask depending on the task. Another approach discretises visual features into a learned codebook, allowing image patches to be represented as "visual words" [16, 2, 48].

Formally, given an image I and a text prompt $X = (x_1, \ldots, x_M)$, an MLLM models the conditional probability of an output sequence $Y = (y_1, \ldots, y_N)$ as

$$p_\theta(Y \mid I, X) = \prod_{n=1}^{N} p_\theta\big(y_n \mid I, X, y_{<n}\big), \tag{2.15}$$

where $y_{<n} = (y_1, \ldots, y_{n-1})$ denotes previously generated tokens. The presence of I in the conditioning ensures that text generation is grounded in the visual input.

2.2.2 Representative Multimodal Systems

Many concrete MLLM instances follow the general templates above. We briefly highlight a few families that are representative of the current landscape.

Vision-Encoder + LLM with Lightweight Projection. In models such as LLaVA, a CLIP-style vision encoder produces an image embedding that is projected into the token

space of a decoder-only LLM [67]. The system is then instruction-tuned on image–text conversations, enabling open-ended visual dialogue, description and question answering.

Vision-Encoder + Query Transformer + LLM. The BLIP-2 framework [62] introduces a query-based transformer (Q-Former) that interacts with frozen image features and outputs a small set of visual tokens, which are in turn mapped to the embedding space of a frozen LLM. This design minimises the number of trainable parameters while achieving strong performance on captioning and VQA benchmarks.

Unified Transformer Models. Some systems train a single Transformer from scratch on interleaved image–text sequences, so that visual and textual information is modelled within a unified architecture. These models can handle complex inputs, such as long documents with embedded figures, and exhibit emergent capabilities in visual reasoning and instruction following [13].

In parallel, proprietary models such as GPT-4V showcase the potential of scaling MLLMs to billions of parameters and web-scale multimodal data, achieving strong performance across captioning, VQA, chart understanding and OCR-free document analysis [84, 85].

High-capacity open MLLMs (Qwen2-VL, IDEFICS). Recent open-source systems such as Qwen2-VL and its successors extend the dual-model paradigm by combining strong vision encoders with large decoder-only LLMs that can process high-resolution images and long image sequences, and by explicitly targeting instruction-following and tool-use scenarios [120, 4]. Similarly, the IDEFICS family was designed as an open reproduction and extension of Flamingo-style architectures, scaling up both vision and language components while remaining accessible to the research community [60, 2]. In the context of this thesis, such models are representative of high-capacity multimodal backbones that can serve as judges or assistants in evaluation-centric pipelines, particularly for instruction-guided image editing.

2.2.3 Grounding, Reasoning and Tool Use

Effective multimodal behaviour requires more than recognising objects in images: models must *ground* linguistic expressions in visual content, reason over this content, and follow

instructions. Grounding can be viewed as learning a mapping between visual features and textual concepts such that conditional text generation reflects what is present in the image. This is often encouraged through contrastive pretraining objectives, captioning losses, or instruction tuning with human- or LLM-generated examples [63, 67, 92].

Reasoning involves operations such as counting, relational understanding and temporal ordering. Many MLLMs benefit from prompting strategies originally developed for text-only LLMs, such as chain-of-thought (CoT) prompting, where the model is encouraged to articulate intermediate reasoning steps before producing a final answer [124, 137]. CoT-style prompts can be adapted to multimodal settings by explicitly asking the model to describe relevant parts of the image before answering a question.

Recent works also explore *tool use*, where an MLLM delegates sub-tasks to external modules: for example, calling an object detector to localise entities, or invoking an OCR engine for fine-grained text reading in images. The model learns to decide when and how to use such tools based on text instructions and visual context [130, 84].

2.2.4 Limitations and Challenges

Despite rapid progress, current MLLMs exhibit important limitations.

- **Visual mistakes (hallucination).** Models may describe objects, attributes or relationships that are not present in the image, or may ignore salient content. This phenomenon mirrors analogous failure modes in text-only LLMs and can be exacerbated by biases in training data. Quantifying and mitigating multimodal hallucinations is an active research topic [13, 128].

- **Temporal and contextual drift.** When applied to sequences of images (e.g. videos), many models struggle to maintain temporal coherence and may forget earlier frames. Similarly, their world knowledge reflects the training data snapshot and may be outdated for rapidly evolving domains.

- **Alignment and safety.** Ensuring that outputs follow user instructions, respect safety constraints, and avoid harmful or biased content remains challenging. Existing alignment techniques (e.g. reinforcement learning from human feedback) must be adapted to multimodal settings with appropriate evaluation data [88, 84].

- **Computational cost.** Large models require substantial compute for training and inference, especially when processing high-resolution images or long sequences. This motivates research into more efficient architectures, quantisation and low-rank adaptation methods [46, 25].

2.2.5 Visual Question Answering Models

Visual Question Answering (VQA) has long served as a benchmark for multimodal understanding: given an image and a question, the goal is to produce a short textual answer. Early VQA systems were typically task-specific architectures combining CNNs for images and RNNs or Transformers for questions, trained directly on VQA datasets. Recent MLLMs can treat VQA simply as another prompting pattern within the general sequence modelling framework (2.15).

Given an image I and a question Q, one constructs a prompt of the form

$$X = \text{“}\texttt{USER: }\text{‘}Q\text{”}\ \texttt{ASSISTANT:}\text{”}, \tag{2.16}$$

optionally preceded by system-level instructions or few-shot examples. The model then generates an answer sequence A by autoregressive decoding. In this view, the same backbone can support both general dialogue and specialised VQA tasks, with performance controlled by the quality of instruction tuning and the diversity of training data [67, 13].

2.3 Evaluation Protocols for Generative Models

Evaluating generative models that operate over images and text is intrinsically challenging. There is usually no single “correct” output, and quality depends on many factors, including realism, diversity, semantic alignment and faithfulness. In this section, we review commonly used automatic metrics and human evaluation protocols, together with practical considerations for reproducibility.

2.3.1 Distribution-level Image Quality Metrics

Classical metrics such as Fréchet Inception Distance (FID) and Inception Score (IS) evaluate the overall quality of a set of generated images, largely ignoring the conditioning text.

FID [44] compares the distribution of real images and generated images in the feature space of a pretrained classifier (typically Inception-v3). Let $\phi(\cdot)$ denote the feature extractor, and let (μ_r, Σ_r) and (μ_g, Σ_g) be the empirical mean and covariance of features for real and generated images, respectively. Assuming Gaussian feature distributions, FID is defined as

$$\mathrm{FID} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\big(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}\big). \tag{2.17}$$

Lower values indicate that the generated distribution is closer to the real one. FID is widely used for unconditional or class-conditional image generation, but does not directly measure the faithfulness of an image to a text prompt.

The Inception Score [102] instead considers the classifier's predictive distribution $p(y \mid x)$ over labels y, and encourages images that are both individually classifiable and diverse:

$$\mathrm{IS} = \exp\Big(\mathbb{E}_x\big[\mathrm{KL}\big(p(y \mid x) \,\|\, p(y)\big)\big]\Big), \tag{2.18}$$

where $p(y)$ is the marginal label distribution over generated images. Although historically influential, IS is less commonly used in contemporary work, as it is sensitive to the choice of classifier and ignores conditioning information.

2.3.2 Image–Text Alignment Metrics

For conditional image generation and image captioning, it is crucial to assess how well images and text align. Embedding-based metrics such as CLIPScore and BLIPScore have become popular due to their simplicity and empirical correlation with human judgments [43, 63].

CLIPScore [43] uses a CLIP model [92] to obtain an image embedding $f_I(I)$ and a text embedding $f_T(T)$. The similarity between an image I and a text T is given by the cosine similarity:

$$s_{\mathrm{CLIP}}(I, T) = \frac{f_I(I)^\top f_T(T)}{\|f_I(I)\|_2 \, \|f_T(T)\|_2}. \tag{2.19}$$

In practice, CLIPScore may include a scaling and truncation to emphasise positive similarity [43]. When used as an evaluation metric, T is often the input prompt (for text-to-image) or a generated caption (for image-to-text), and $s_{\mathrm{CLIP}}(I, T)$ serves as a proxy for alignment.

BLIPScore is an analogous metric based on BLIP-style vision–language encoders [63]. Such metrics are attractive because they are reference-free and relatively cheap to compute. However, they can be insensitive to fine-grained errors (e.g. incorrect object counts or attributes) and may be biased by the properties of the underlying model [43, 128].

When reference captions are available, classical text-based metrics like BLEU, METEOR, CIDEr and SPICE [7, 65] can be applied to the generated captions. These metrics measure lexical or semantic overlap between generated and reference texts, but they are less suited to evaluating the alignment between images and prompts in the absence of reference captions.

2.3.3 VQA-based Faithfulness Metrics

To capture fine-grained faithfulness between a text description and a generated image, recent work proposes metrics based on visual question answering (VQA). The core idea is to break down a textual description into a set of factual statements, convert them into questions, and test whether a VQA model can recover the expected answers from the image.

TIFA [47] is a prominent example. Given a prompt P and an image I, a language model is used to generate a collection of question–answer pairs $\{(q_k, a_k)\}_{k=1}^{K}$ that probe different aspects of the prompt (objects, attributes, relations, counts, and so on). For each question, a VQA model g predicts an answer $\hat{a}_k = g(I, q_k)$, and TIFA is computed as

$$\mathrm{TIFA}(I, P) = \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}\big[\hat{a}_k = a_k\big], \tag{2.20}$$

where $\mathbf{1}[\cdot]$ is the indicator function. Higher scores indicate that more prompt-derived facts are supported by the generated image.

VQA-based metrics provide interpretable, attribute-level feedback (e.g. which objects or relations are incorrect), but their reliability is limited by the accuracy and biases of the underlying VQA and language models [47, 128]. For example, VQA models can exhibit yes-bias, answering "yes" to questions even when the image does not contain the queried object. This motivates careful design of evaluation pipelines and, when possible, complementary

human studies.

2.3.4 Human Evaluation and LLM-as-a-Judge

Despite considerable progress in automatic metrics, human evaluation remains essential, especially for nuanced aspects such as aesthetic quality, creativity and overall preference. Typical protocols include:

- **Rating tasks**, where annotators assign numerical scores to images based on criteria such as realism, prompt relevance or style.
- **Pairwise comparisons**, where annotators choose which of two (or more) images better satisfies a prompt, enabling the estimation of win rates between models.
- **Verification tasks**, where annotators confirm whether specific textual statements about an image are true or false.

These studies can be conducted with expert annotators or crowdworkers, but are expensive and require careful experimental design to ensure reliability and reproducibility [87]. Inter-rater agreement measures (e.g. Cohen's κ or Bland–Altman plots) can be used to quantify consistency between annotators [10].

Recent work explores using powerful LLMs themselves as automatic judges: given a prompt, an image and possibly a reference output, the LLM is asked to rate the quality or choose between alternatives [73]. This *LLM-as-a-judge* paradigm can approximate expert judgments at scale, but must be calibrated and validated carefully, as the evaluator inherits the biases and limitations of the underlying model.

2.3.5 Reproducibility and Reporting Practices

Finally, robust evaluation requires attention to reproducibility. Key aspects include:

- **Randomness control.** Diffusion models rely on stochastic sampling; it is therefore important to fix random seeds where possible, report the number of samples per prompt, and, for metrics such as FID or CLIPScore, aggregate over sufficiently large sample sizes to obtain stable estimates.

- **Dataset splits.** Training, validation and evaluation datasets should be clearly described, with any prompt filtering or deduplication procedures documented. When evaluating on public benchmarks (e.g. COCO for image captioning [65]), the specific splits and preprocessing should be stated explicitly.

- **Prompt design.** Evaluation prompts should be representative of intended use cases and publicly released where possible. Prompt leakage between training and test sets should be avoided, particularly when using web-scale training data [87].

- **Metric configuration.** Implementation details for metrics (e.g. which CLIP backbone and pre-processing pipeline are used) can substantially affect results and should therefore be reported.

- **Human-study protocols.** For human evaluation, experiment design (instructions to participants, sampling strategy, number of raters per item) and statistical analysis procedures should be documented to enable replication.

Taken together, distribution-level metrics, alignment scores, VQA-based faithfulness measures and human evaluations provide a multi-faceted view of generative model performance. No single metric captures all aspects of quality and alignment, so it is good practice to report a complementary set of measures and to analyse failure cases qualitatively [87, 128].

Chapter 3

Hallucination-Aware Control of Text-to-Image Generation

3.1 Chapter Overview

Text-to-image diffusion models have rapidly become the default backbone for modern generative systems, but their impressive expressive power comes with substantial computational cost and a non-negligible risk of hallucinations. Generating a single high-resolution image typically requires dozens of forward passes through a large UNet or a Vision Transformer, and small changes in random seed or scheduler configuration can lead to qualitatively different outputs [72, 101, 138]. At the same time, downstream applications demand images that are both faithful to the requested content and efficient to obtain, especially when generation is deployed at scale.

This chapter investigates how *evaluation* signals, extracted during the generation process itself, can be used to improve both quality and efficiency, operationalizing two of the thesis's core pillars: **Sustainability** and **Granularity**. Rather than treating evaluation as a separate, offline step that comes after image synthesis, we use it in HEaD to decide *how long* generation should run and *when* it is safe to stop early. In this way, hallucination early detection becomes an active resource-saving mechanism that cuts off trajectories predicted to end in meaningless or hallucinated generations. To achieve this, we move beyond global image

metrics and exploit the fine-grained temporal dynamics of the diffusion process, monitoring internal consistency signals as they evolve timestep by timestep. In the evaluation-centric lifecycle of Figure 1.1, this corresponds to the hallucination early detection block, instantiated first by the original HEaD conference model and then by its extended HEaD+ journal version.

Concretely, we study *Hallucination Early Detection* (HEaD) for diffusion models through two closely related works:

- **ECCV paper** The first work, originally published at the *European Conference on Computer Vision (ECCV)*, introduces the core HEaD framework on a controlled, two-object benchmark and defines hallucination early detection as the task of predicting, partway through the denoising trajectory, whether the final image will omit any requested objects. In this setting, it analyses how these predictions can be used to stop and restart generations, estimating the gains in time needed to reach *perfect* generations.

- **IJCV paper** The second work, published in the *International Journal of Computer Vision (IJCV)*, significantly extends this formulation, scaling it to more complex prompts and multiple generators. It revisits the HEaD idea with refined hallucination prediction architectures that combine Predicted Final Images, cross-attention maps and textual embeddings, and with an extensive experimental study on the InsideGen benchmark, including ablations and robustness checks across models and prompt types.

Across both studies, hallucination predictions are made at the level of specific objects extracted from the text prompt and the corresponding regions they should occupy in the (predicted) image. This object- and region-level reasoning provides the granularity needed to explain exactly *which* part of the image is expected to fail, and thus why an early stop or restart is warranted.

From the perspective of this thesis, the two papers should be read together as a single contribution to *evaluation-guided generation*. The ECCV version provides the initial formulation and evidence that hallucination-aware evaluation can directly improve generation efficiency, while the IJCV version consolidates and generalises these findings, broadening

both methodology and empirical support. To avoid unnecessary repetition, general preliminaries on diffusion models, schedulers, MLLMs and evaluation protocols are presented once in Chapter 2, and the reader is referred there for background definitions and notation.

From an empirical perspective, both papers demonstrate that hallucination early detection can substantially improve the efficiency of text-to-image diffusion models while preserving, or even improving, the rate of *perfect* generations. In the controlled two-object setting of the ECCV study, HEaD recovers most hallucinated cases and yields up to about 12% savings in average generation time, illustrating the potential of early stopping even when the base model is already accurate. The IJCV extension then scales this idea to the more challenging InsideGen benchmark and to multiple generators, where a refined hallucination prediction network and extensive ablations identify the best-performing configurations and show that equipping existing models with HEaD+ can save up to around 38% of the time needed to reach a *perfect* generation on SD2 while consistently increasing the probability of perfect four-object generations across models. Crucially, the hallucination prediction module is architected as a model-agnostic *add-on*: it can be attached on top of diffusion models without retraining them, operating on intermediate representations to decide whether to continue, restart or stop early. While the dominant sustainability gains come from stopping doomed trajectories before they complete, this reusable, model-agnostic design further reinforces the thesis's sustainability pillar, since the same hallucination predictor can be reused across heterogeneous generators rather than retrained from scratch for each backbone.

3.2 Optimizing Resource Consumption in Diffusion Models through Hallucination Early Detection

3.2.1 Introduction

In the rapidly evolving domain of AI, generative models have emerged as a notable subfield, demonstrating an exceptional ability to generate complex visual and textual content [26, 91]. The advent of Text-to-Image (T2I) generation marked a significant leap in this domain through the introduction of GAN-based approaches [38, 97, 132] and further advancements through large-scale pre-trained Diffusion Models (DM) such as Stable Diffusion (SD) [98] and others [93, 17]. These approaches have been instrumental in shaping the generative AI landscape, delivering images that are increasingly indistinguishable from real ones.

Generative models, while progressing, often hallucinate "long-tail" objects, which are underrepresented elements in training datasets, and have significant shortcomings when generating multiple objects [136, 103, 17]. Furthermore, they frequently hallucinate attributes, counts, and semantic object relations, which is especially problematic when tasked with rendering scenes involving multiple objects [69, 31].

The challenge is further intensified when generating combinations of specific objects, where diffusion patterns often produce inconsistencies, significantly impacting the quality of the output [41]. The choice of the initial seed, which dictates the initial latent noise, is fundamental in navigating the latent space for correct image generation. The dependency on seed selection highlights the unpredictability and variability of these models [53, 104, 69, 126, 17]. Although automatic evaluation mechanisms could offer a potential solution to these challenges, their adoption is not straightforward. While some attempts in this direction have been made [8, 73], indeed, they still fail at ensuring a sufficiently fast and reliable evaluation. However, employing these automatic evaluations tends to be slow and resource-intensive. This is largely due to the numerous incorrect results, which require images to be regenerated, thus escalating both time and resource costs.

Addressing these challenges, we introduce HEaD, the first approach designed to enhance both the efficiency and accuracy of generative DMs. HEaD incorporates the use of cross-attention maps to examine the relationship between the prompt and the internal attention

Figure 3.1: Overview of the HEaD pipeline: during the generation process, HEaD assesses whether all designated objects will be accurately represented in the final image, determining if the generation process should continue or be restarted with a different seed.

layers of the model, along with the *Predicted Final Image* (PFI) - a prediction of the expected outcome at intermediate stages of the generation process. The combination of PFIs and cross-attention maps allows for the early identification of potential errors by predicting the presence of objects requested by the initial prompt. By preemptively detecting these anomalies, HEaD hints at stopping the generation diffusion process, thereby conserving resources and reducing the time spent on generating images that would not eventually meet quality standards. Aiming for a *complete* generation – where all requested objects are accurately depicted – halting the generation process based on a prediction of the final outcome proves to be far more efficient than completing an image generation and subsequently evaluating it. This approach not only streamlines the generation process but also enhances resource utilization by avoiding the production and evaluation of substandard images.

We trained two types of networks, each with a different backbone for handling PFI data, followed by CNN-based processing of cross-attention maps. This training occurred at different points in the generation pipeline to assess their impact on prediction quality and potential time savings. Results indicate that using a Visual Transformer as a backbone yields superior outcomes. Moreover, while networks trained towards the later stages of the generation pipeline benefit from higher-quality input and thus demonstrate better performance, those trained earlier exhibit greater potential for time and resource savings. It is also important to note that the methodologies and models described in this study are model-agnostic, i.e. they can be seamlessly adapted to any diffusion-based generative model.

In this work, we focus on specific hallucinations: the omission in the generated image of one or more target objects indicated in the textual prompt. We propose both a detector (trained on a dataset of corrected and hallucinated generated images) and a general approach for time-saving prediction that accounts for both the hallucination probability of the specific

generative model and the accuracy of the detector. For instance, when generating images with prompts involving two objects in non-trivial combinations, SD2 produces hallucinations or missing object errors in 41% of cases, according to our dataset. Our HEaD approach can detect the majority of these errors with minimal time overhead, thus saving up to 12% of the average generation time in this simple scenario.

To sum up, our main contributions are as follows:

- We introduce a new element, PFI, and demonstrate that its integration with cross-attention maps effectively facilitates the early detection of objects within generated images.
- We propose a comprehensive framework for time saving evaluation. We demonstrate the potential time and resource saving for the generation of *complete* images from multi-object prompts, without compromising the output integrity and generation quality.
- A novel classifier for Hallucination Early Detection has been developed that, when integrated into the diffusion process, combines information at each diffusion level and acts as an early evaluator. This classifier stops the process if a hallucination is detected, thereby enhancing the efficiency and accuracy of the generation.

3.2.2 Related Works

Text-to-Image Evaluation. Quantifying the alignment between the generated image and the initial prompt is a challenging task, and as of now, no effective solutions have been identified. Among the assessment metrics, CLIPScore [43] evaluates the cosine similarity between the prompt and the image, both having undergone processing through their respective visual and textual CLIP backbones. Recent studies [8, 73] have proposed innovative scoring mechanisms that leverage the capabilities of Large Language Models (LLMs) and Visual Question Answering. In alignment with this research trajectory, various investigations [47, 128] have introduced diverse methodologies, positioning their work within the reasoning paradigm facilitated by LLMs.

Despite their success in identifying hallucinatory elements in generative models, these works still require the generated image as input, which is produced only in the final step of the

diffusion process. Additionally, they incorporate evaluation steps beyond image generation, resulting in delays due to the reliance on foundational models within the evaluation pipeline. Conversely, HEaD enables the detection of hallucinations during the generative process itself, preventing the creation of images that do not align with their prompts.

Attention Maps in Image Generation. The integration of attention mechanisms has been a cornerstone in improving image synthesis within generative models. Notably, Chefer et al. refines these processes to enhance image detail [17]. Cross-attention layers [98] have significantly boosted visual fidelity, a concept further explored by Hertz et al. to maintain coherence between text prompts and visual outputs [42]. The importance of semantic layouts in improving image quality and interpretability has also been highlighted by Wand et al. [122]. Building on these ideas, SynGen was introduced [95], which aligns attention maps with prompt syntax to improve attribute correspondence, optimizing the generation process without the need for model retraining. Furthermore, Ma et al. developed a novel method for controlling image synthesis by editing initial noise images, demonstrating that manipulating pixel blocks in initial latent images can influence specific content generation [76]. Additionally, Balaji et al. proposed eDiff-I, an ensemble of expert denoisers that enhances text alignment and visual quality by specializing models for different stages of synthesis [6].

Following the consensus on the effectiveness of cross-attention as a telltale sign of the fidelity of the generation, our work exploits this information as a discriminating factor for the accurate generation of the final image.

Seed Importance. In Text-to-Image generation, images are significantly impacted by the initial state or starting seed of the diffusion process. Indeed, different seeds produce completely different image results as highlighted by Karthik et al., which claims to generate better-aligned images by evaluating multiple seeds [53]. Furthermore, image editing by directly manipulating the initial noise instead of steering the generation process with additional mechanisms has also been proposed [76, 104].

Seed selection has gained relevance in the generation of long-tail concepts [136]. Samuel et al. propose that, in the generation of rare subjects, training predominantly involves exposure to a limited segment of the initial noisy latent space [103]. This selective exposure during training contributes to the generation of unsatisfactory outcomes across a majority of generative seeds at inference time. Hence, the exploration of diverse generative seeds

remains a critical aspect in enhancing generative outcomes. To mitigate the occurrence of hallucinations, HEaD suggests altering the seed in the event of detecting hallucinations in the generative process.

3.2.3 Hallucination Early Detection

HEaD primary goal is to detect and preemptively interrupt faulty generative processes. Its novelty lies in its ability to perform this detection at intermediate stages of the image generation, leveraging one or more time steps of the diffusion pipeline. Consequently, if the Hallucination Prediction (HP) network predicts that the image will not be *complete* – indicating the absence of at least one target object – the generative process can be halted and restarted with an alternative seed. This preemptive detection conserves computational resources by preventing the completion of flawed images, eliminating the need to sample a new seed, and avoiding a complete restart from scratch.

In this section, we illustrate the proposed HEaD approach at inference time to streamline the generation process and, as a result, enable automatic quality assessment of the final output. The pipeline initial step involves extracting the target objects from the prompt and providing hallucination indicators for the HP network to evaluate.

3.2.3.1 Cross-Attention Maps and PFIs Extraction

Given a prompt y containing a set of target objects O to be generated, the extraction process of these target objects can be formalized as follows:

$$O = \text{TOE}(y) \tag{3.1}$$

where TOE$(\cdot)$ represents the Target Object Extraction function. Here, the term "objects" refers to words in the prompt directly associated with discernible elements in the image, for which we will extract the corresponding cross-attention maps. While our current methodology primarily focuses on objects, it holds the capability for future expansion to include a wider spectrum of visual concepts, thereby transcending the confines of object-based extraction.

We define a sequence of *critical timesteps*, denoted as $\mathcal{T} = \{t_{c_1}, \ldots, t_{c_k}\}$, as specific

steps in the diffusion process where cross-attention maps and PFI are extracted. These components will serve as inputs for the HP network.

In diffusion models, the UNet employs cross-attention layers at resolutions from 64 to 8, producing a combined attention map $A_t \in \mathbb{R}^{64\times64\times N}$, where N is the number of tokens from the prompt y. For each object o in the target set O and each critical timestep $t_c \in \mathcal{T}$, the specific cross-attention map A_{o,t_c} is derived by filtering A_t for object o.

For each critical timestep $t_c \in \mathcal{T}$, a Predicted Final Image (PFI_{t_c}) is extracted. PFI_{t_c} represents the prediction of the expected outcome at the end of the generation process, using only information available at timestep t_c. In particular, the scheduler projects the latents at t_c to the final step, and the decoder translates these predicted latents into the image space. The process is defined as follows:

$$\begin{aligned} \epsilon_{t_c} &= \epsilon_\theta(z_{t_c}, t_c) \\ z_0^{t_c} &= \Delta(z_{t_c}, \epsilon_{t_c}, t_c, 0) \\ \text{PFI}_{t_c} &= D(z_0^{t_c}) \end{aligned} \tag{3.2}$$

where ϵ_{t_c} represents the predictive noise obtained from the UNet model at critical timestep t_c. The function Δ updates the latents z_{t_c} to the predicted latents at the final timestep, denoted as $z_0^{t_c}$. Finally, the VAE decoder D translates these predicted final latents into PFI_{t_c}.

Examples of PFIs extracted at different timesteps are shown in Fig. 3.2. The collection of PFIs, namely $\text{PFI}_{\mathcal{T}}$, and attention maps, $A_{O,\mathcal{T}}$, across all critical timesteps provides a comprehensive dataset for the HP network to analyze and predict the presence of objects in the final image.

3.2.3.2 Hallucination Prediction Network

During the evaluation phase, the Hallucination Prediction network takes as input the cross-attention maps A_o for a specific object and the $\text{PFI}_{\mathcal{T}}$ and outputs a binary prediction indicating the presence or absence of that specific target object in the final image:

$$H_o = \text{HP}(A_{o,\mathcal{T}}, \text{PFI}_{\mathcal{T}}) \tag{3.3}$$

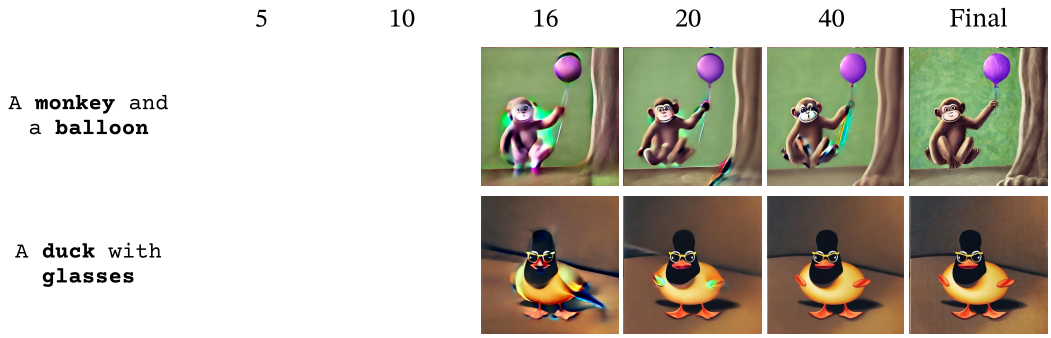


Figure 3.2: Qualitative examples of the Predicted Final Image for each prompt at different critical timesteps. Already from the 16^{th} step the final image is fully represented and the presence of objects can be predicted.

where H_o is the binary prediction for object o. The training methodology for the HP network is detailed in Section 3.2.4.

The reliability of the HP network is critical to prevent unnecessary terminations of the generation and ensure that objects that would have been present are not prematurely discarded.

In all network configurations, as feature extractor from cross-attention maps, we utilize a series of four convolutional layers that bring the input from a $1 \times 64 \times 64$ to a final $128 \times 7 \times 7$ shape.

HP-R – HP-V. For scenarios where the number of critical timesteps selected is 1, i.e. $|\mathcal{T}| = 1$, we have explored two distinct backbone options: Resnet50 and Vision Transformer (ViT), which correspond to the HP-R and HP-V architectures respectively. These architectures are tailored to process the PFI to extract features that are then combined with those extracted from the cross-attention maps to get the final prediction. HP-R concatenates the output of the Resnet backbone on the channel level with the processed cross-attention maps and a further two-layer CNN is used to merge channels down to 512. HP-V processing outputs a vector that is concatenated to the flattened version of the cross-attention maps output. Both networks employ a final linear layer for the binary prediction.

HP-MultiR. For scenarios where $|\mathcal{T}| > 1$, we have created the HP-MultiR network in which Resnet50 was used to extract features from different timesteps in parallel. Features and the processed cross-attention maps are concatenated on the sequence dimension before applying two Conv3d layers with a final 3D pooling to reduce dimensions. A final classifier

is eventually used for the final prediction.

HP-A. Additionally, we developed the HP-A network to specifically investigate the influence of attention maps on hallucination prediction. This network configuration excludes the PFI from its input, focusing solely on the features extracted from attention maps. A final prediction layer is attached directly to the common cross-attention maps feature extractor. By employing a similar architecture with convolutional layers as the other HP variants, the HP-A network focuses on evaluating how well attention maps alone can predict hallucinations. The results from this model provide critical insights into how effectively attention maps alone can inform the hallucination prediction process in diffusion models.

3.2.4 Hallucination Network Training

Dataset Creation. To train the Hallucination Prediction network we collected 900 prompts obtained by combining 75 distinct animal subjects with 12 objects with the prompt "`A {animal} and a {object}`". To augment the dimensionality of the dataset and thoroughly investigate output variations influenced by different seeds, we generated 12 images for each prompt using distinct seeds. Following this protocol we generated nearly 10.000 images by making use of Stable Diffusion v2.0 generator [98]. During generation, we fixed 50 steps of the diffusion process and we collected the PFI and cross-attention maps A at multiple time steps[1].

Target Objects Extraction. While our dataset comprises prompts with objects in predetermined positions for simplicity, we integrated object extraction to simulate real-world scenarios. For this purpose, we employed gpt-3.5-turbo-1106 [84], selected for its robust zero-shot generalization abilities. This method stands in contrast to conventional text tagging techniques that generally necessitate specific training for each domain.

The extraction procedure is time-efficient and can be executed concurrently with the initial diffusion steps. Details on the specific prompts used in this study can be found in the supplementary materials.

Label Creation. An essential feature is the development of an automatic labeling system to confirm the presence of particular objects in the generated images. This system must

[1] In particular, the critical steps $\mathcal{T}$ are chosen as follows: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 25, 40]

function without a fixed set of object labels, requiring the adoption of an open vocabulary approach. To achieve this, we adopted OWLv2 [80], an open vocabulary detector renowned for its robust detection capabilities and for providing confidence scores for each identified object.

3.2.5 Time Saving Analysis

Our study primarily explores the time-saving benefits of the HEaD approach in DMs when trying to generate a *complete* image. In our analysis, we found that accurate generation of both objects in complex scenarios was achieved in only 59% of cases by SD2 without HEaD. This statistic underscores the challenges models encounter when generating multiple objects accurately, particularly as the complexity of the prompt and object combinations increase. Certainly, with more objects and increasingly complex prompts, the probability of correct generation diminishes, which in turn heightens the impact of HEaD on time-saving.

In the dataset, which is made with 12 seeds per prompt, we found that every prompt successfully led to at least one correctly generated image, and 98.4% resulted in at least three accurate images. This confirms the feasibility of the prompts for some random seeds. HEaD serves here as an implicit evaluator, swiftly identifying instances where the generated image is likely to be inaccurate. By promptly halting these less promising generative paths, HEaD allows for more efficient use of resources, enabling quicker initiation of new generation attempts with different seeds.

3.2.5.1 HEaD impact on Time Saving

HP Performance. Labels in our dataset are created using an open vocabulary detector, which assesses whether each object is present (1) or absent (0) in the images. The HP network, based on these labels, decides whether to continue or halt the image generation process. When a True Positive (TP) occurs, the correct generation proceeds uninterrupted, having no effect on computation time. Conversely, a False Positive (FP) allows an incorrect generation to continue without interruption, thus missing an opportunity for time savings, but still not impacting computation time. A True Negative (TN) indicates an incorrect generation has been correctly halted, leading to time savings. Finally, a False Negative (FN) means a correct generation is mistakenly stopped, resulting in a loss of time.

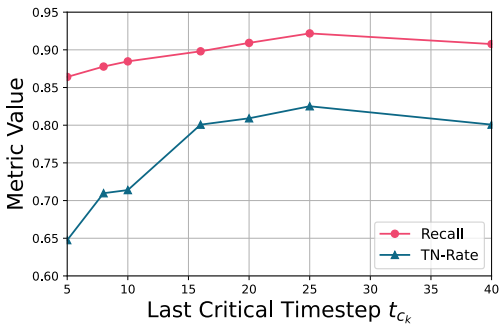


Figure 3.3: Recall and TN-rate values for HP-R across various t_{c_k}. Lower t_{c_k} values, associated with lower quality input, significantly impact the TN-Rate but minimally affect Recall. Consequently, the overall time saved tends to be greater for smaller t_{c_k} values.

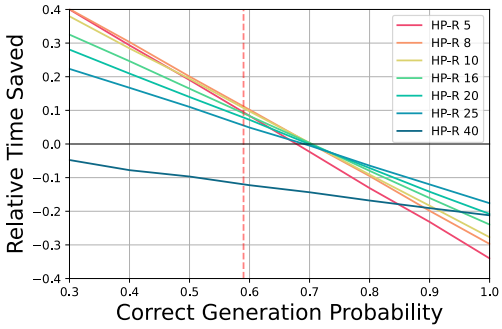


Figure 3.4: Relative time saving between adopting or not the HEaD approach to reach a *complete* generation, using HP-R with different t_{c_k}, depending on the probability of a correct image generation. The vertical red line marks the probability of correct generation in a two-objects scenario, i.e. 59%.

Thus, in order to save computational time, the network should be trained to balance both high recall and a high TN-rate. High recall ensures the HP network effectively identifies all instances of correct generation, minimizing FNs and avoiding unnecessary termination of accurate processes. Simultaneously, a high TN-rate boosts the HP network's capability to maximize true negative outcomes, allowing for early termination of incorrect generations by accurately identifying cases where not all requested objects in the prompt are included. This dual focus on both recall and TN-rate optimizes the generation process by reducing time loss, yet still maintaining the quality of the output.

Critical Timesteps selection. The ratio $\frac{t_{c_k}}{T}$, where t_{c_k} denotes the latest t in the set of critical timesteps $\mathcal{T}$ and T is the total number of steps in the generation process (with 50 being the standard for SD), plays a crucial role in determining the percentage of time saved. An earlier detection, indicated by a smaller t_{c_k}, can potentially lead to greater time savings in case of a correct hallucination identification. However, this scenario presents a significant challenge: in the initial stages the quality of attention maps and PFIs is lower. This lower quality affects the performance of the HP network resulting in reduced recall and TN-rate, as shown in the plot in Fig. 3.3. Therefore, this tradeoff between early detection and maintaining the quality of attention maps and PFIs is essential for maximizing the efficiency of the HEaD approach.

Finally, to quantify the time saved or lost using the model, we conducted Monte Carlo simulations based on the models presented in the next section. The algorithm calculates a savings of $\frac{t_{c_k}}{T}$ of the generation time when a true negative (TN) is detected. Conversely, it accounts for a time loss when a new restart is necessary due to a false negative (FN). The detailed algorithm and simulation results are provided in the supplementary materials.

3.2.6 Experimental Results

The evaluation of various HP Network variants underscores their influence on the image generation process. The computed Recall and TN-rate metrics, which are influenced by the t_{c_k} value, serve as key indicators of model performance. As depicted in Fig. 3.3, the TN Rate typically increases with a higher t_{c_k}, whereas Recall tends to remain stable across different stages of detection.

To provide a final efficiency assessment, the percentage of time saved during generation has been adopted as the primary metric for final comparison. This metric integrates the effects of varying t_{c_k}, Recall, and TN-Rate values, offering a quantifiable measure of each model's effectiveness in reducing generation times. For these experiments, a correct generation probability of 59%, as derived from the dataset, has been employed to ensure accuracy in the evaluations. Table 3.1 provides a comparative analysis of HP-R and HP-V, illustrating the time saved when these networks operate at different t_{c_k} intervals. Both networks have the highest impact when $t_{c_k} = 8$, where HP-V saves up to 12.66% of generation time. Higher t_{c_k} values can enhance input quality and metric results, but they may limit time-saving opportunities. No models bring any benefit when using $t_{c_k} \geq 25$, as the time saved in case of a correct prediction is insufficient.

In Fig. 3.4, an analysis is presented to illustrate the relationship between the relative time saved and the generation probability across different t_{c_k} values. The vertical line indicates a 59% correct generation probability, typical for scenarios involving two objects, as observed in our dataset. More complex prompts, which often require synthesizing additional objects, tend to have lower probabilities of achieving a *complete* generation, thus enhancing potential time savings. Notably, $t_{c_k} = 8$ offers the optimal balance, providing significant time savings, especially when the probability of *complete* generation is as low as 40%, where time savings can reach up to 30%. Conversely, when the probability of a *complete* generation is high, using

$\mathcal{T}$	**HP-V**	**HP-R**
5	9.7	9.11
8	**12.66**	**10.56**
10	9.68	10.34
16	6.72	8.93
18	5.77	5.78
20	5.75	7.25
25	-0.35	5.32
40	-14.11	-11.67

Table 3.1: Percentage of time saved for all models. t_{c_k} is the last diffusion timestamp considered over the 50 of SD2.

Model	$\mathcal{T}$	**% Time Saved**
HP-A	10	6.65%
	16	3.04%
	20	-0.73%
HP-Multi	6-8-10	-3.72%
	10-12-14	8.99%
	16-18-20	6.88%

Table 3.2: Percentage of time saved for HP-A and HP-Multi in different $\mathcal{T}$ scenarios.

$t_{c_k} = 5$ results in considerable time loss due to imperfect Recall, which can prematurely halt a correct generation. Additionally, employing HEaD at $t_{c_k} = 40$ provides no benefits in any scenario, as the time saved in the rare event of a true negative is merely 20%, considering the 50 steps generation pipeline of SD2.

In Table 3.2, HP-A testing serves as an ablation study to underscore the significance of Predicted Final Images. In the absence of PFIs, which are unique per image and not per object, the HP-A model shows a marked decrease in its ability to detect early hallucinations and thus in time saved. With $t_{c_k} = 10$ only 6.65% of generation time is saved.

The HP-Multi model takes an advanced approach by focusing on multiple $\mathcal{T}$. A noteworthy aspect of HP-MultiR performance is its effectiveness in later timesteps ($t_{c_k} = 14$), compared to a less marked performance in early timesteps. This discrepancy can be attributed to the inhomogeneity of the data in the early stages, where the characteristics of the data change considerably from one step to the next. This variability makes the mixing of the features in these early stages less effective. In contrast, data in later stages tend to be more uniform and stable, allowing for more effective learning and integration of features from multiple time steps, thus improving model performance.

3.2.7 Conclusions

This paper introduces HEaD, an innovative approach that not only enhances the efficiency and accuracy of image generation with Diffusion Models but also significantly reduces computational resources. A key innovation is the Predicted Final Image, an effective early error prediction indicator when used in conjunction with cross-attention maps. The effectiveness of our framework in saving time is closely tied to the recall and TN-rate of the Hallucination Prediction network, highlighting HEaD's capacity to improve image generation in a variety of complex scenarios.

HEaD represents a preliminary step in exploring the sustainability and effectiveness of diffusion models, especially for large, complex datasets. Looking ahead, we are committed to further advancing this field of study also by collecting larger datasets with more target objects and more complex visual prompts and proposing challenges for the scientific community to test better early detectors.

3.3 Hallucination Early Detection in Diffusion Models

3.3.1 Introduction

The advent of diffusion models [112, 45] (DMs) and their latent variants [98] has significantly enhanced the fidelity and alignment of generated content with user-provided context [26]. Nonetheless, these models continue to face challenges in faithfully representing the requested prompts. For instance, when a prompt specifies multiple objects, there is a considerable probability that not all objects will be accurately depicted, resulting in an *incomplete* generation [17, 1, 123].

These inaccuracies are problematic for both users, who may not receive the intended content, and for companies, as the production of erroneous images results in wasted computational resources and energy.

Furthermore, diffusion models have exhibited subpar performance in generating "long-tail" objects [136], often misrepresenting rare concepts in the generated images [103]. This challenge has led researchers to explore the influence of initial seeds on the generation process, investigating their effects and developing criteria for their optimal selection [103]. Since

different seeds can produce varying results, selecting the appropriate seed could significantly mitigate both the long-tail issue and the problem of hallucinations.

In our study, we assess the performance of Stable Diffusion v1.4 (SD1.4) and Stable Diffusion v2 (SD2) [98] when tasked with generating images from prompts that request the presence of a different number of objects. Our analysis reveals that the probability of a *complete* generation – i.e. an image showing all requested objects – with four objects is only 26.96% for SD1.4 and improves to 30.61% for SD2, as illustrated in Fig. 3.5(a). The effectiveness of an image generation diffusion model depends strongly on the choice of the initial seed, which establishes the starting latent noise and guides the model through the latent space [53, 104, 69, 126, 17]. This aspect is underlined by our results: in scenarios with four objects, at least one among the 11 seeds we tested leads to a *complete* generation in 79.87% of the prompts. This significant increase highlights the central role of seed selection in overcoming the unpredictability and variability associated with these models, indicating substantial opportunities for improving generative accuracy.

While some attempts have been made to develop automatic evaluation metrics for image generation [8, 73], these still fail at ensuring a sufficiently fast and reliable evaluation. Further, even in the presence of a reliable assessment of the generated image, the generation process should be repeated multiple times until reaching a *complete* generation. In this paper, we instead take a different path and explore solutions for an early (i.e., at early stages of the diffusion process) detection of hallucination. We use this in combination with a procedure to abort or restart generation with another seed to save time and improve the final quality. To this aim, we focus on specific hallucinations: the omission in the generated image of one or more target objects requested in the textual prompt.

Our approach, termed **HEaD+**, or Hallucination Early Detection+, is the first approach designed to enhance both the efficiency and accuracy of generative DMs. HEaD+ leverages intermediate cross-attention maps and textual information to assess the connection between the input prompt and the internal attention operations of the model. Further, HEaD+ operates on the *Predicted Final Image* (PFI), which denotes a forecast of the final generated image at a specific time step of the generation process. The combination of PFIs and cross-attention maps allows for the early identification of potential errors by predicting the inclusion or exclusion of objects requested by the initial prompt. This proactive error identi-

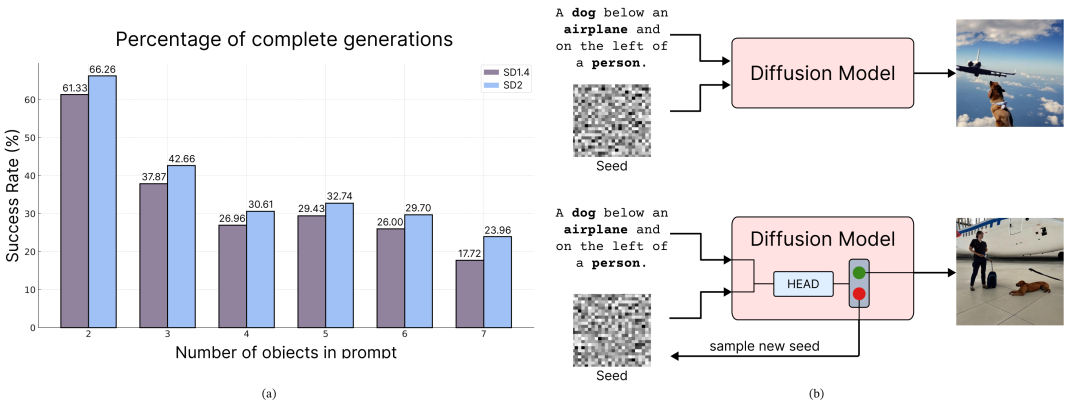


Figure 3.5: a) Proportion of *complete* generations, i.e. images featuring all requested objects, as a function of the number of objects requested in the prompt (InsideGen dataset). b) Summary of the HEaD+ pipeline: throughout the generation process, HEaD+ evaluates whether all specified objects will be correctly depicted in the final image, determining whether to proceed with the current generation or restart with a different seed.

fication allows HEaD+ to suggest terminating the diffusion process early, thereby optimizing resource usage and reducing the time spent on generating images that are unlikely to meet quality standards. In addition to object presence, we equip HEaD+ with a localization module that, when enabled, predicts object centroid positions to verify simple pairwise relations (e.g. left/right, above/below) when specified in the prompt. Ultimately, this decision is combined with presence predictions to determine whether to continue or restart generation.

HEaD+ is developed through the training of an Hallucination Prediction (HP) network on a dataset featuring both accurate and hallucinated images. Specifically, we generate **InsideGen**, a dataset that encompasses 45,000 images generated with SD1.4 [98] and SD2, saving cross-attention maps and PFIs at intermediate steps. Leveraging this dataset, HEaD+ detector is designed to integrate seamlessly with all existing DMs, enhancing their ability to reliably produce images that depict all requested objects. This improvement is evident when HEaD+ is applied to SD1.4 and TokenCompose [123], even with datasets and prompt categories that are entirely different from those in InsideGen.

In the ECCV paper (Section 3.2), we introduced hallucination early detection in a controlled setting with two requested objects. However, when scaled to a more complex experimental scenario, that initial model achieved limited efficiency, with only 11.55% time savings. This journal extension instead introduces a more sophisticated dataset, InsideGen, featuring complex and realistic prompts, and improves the overall HP architecture by in-

corporating the textual information associated with the hallucinated objects and by better combining intermediate signals.

Contributions. To sum up, our main contributions are as follows:

- We propose HEaD+, a novel framework designed to assess the presence of objects in images during the generation process. This opens the way to predicting aspects of the final output in mid-generation, with potential benefits on the entire image generation pipeline using diffusion models.
- Our approach is based on a new indicator for the early detection of objects in the generated images, the Predicted Final Image. Moreover, it employs an Hallucination Prediction network that combines PFI, cross-attention maps, and textual embeddings via a Transformer Decoder approach. This network is model-agnostic, i.e. it can be added to any diffusion models without the need for re-training.
- Our experimental findings show that using HEaD+ in combination with SD1.4 or TokenCompose improves the probability of accurately generating an image from a four-object prompt by 7.85% and 6.52%, respectively.
- As a complimentary contribution, we build and release the InsideGen dataset, which annotates hallucinations and records cross-attention maps and PFIs at various stages of the generation. InsideGen dataset is available for download on the project page[2].

3.3.2 Related Works

Text-to-Image Generation Evaluation. Evaluating the alignment between a generated image and its initial prompt remains a challenging task, with no universally accepted solutions currently available. Among the assessment metrics, CLIPScore [43] evaluates the cosine similarity between the prompt and the image, both having undergone processing through their respective visual and textual CLIP backbones [92]. On the other hand, [8] have introduced a novel scoring mechanism that harnesses the capabilities of Large Language Models (LLMs) and Visual Question Answering. Consistent with this research direction, several other investigations [73, 47, 58, 111] propose diverse methodologies, framing their research within the common reasoning paradigm advanced by LLMs. Additionally, the use of Visual

[2]Project page available at: aimagelab.github.io/HEaD

Question Answering in the context of text-to-image evaluation has been separately explored by [66].

While these methods have demonstrated their proficiency in highlighting the hallucinatory aspects of generative models, they still require the generated image as input, which is produced only in the final step of the diffusion process. Moreover, they include additional processing beyond the generation phase, thus introducing delays in the overall evaluation due to the utilization of foundational models in the evaluation pipeline. In contrast, HEaD+ enables the detection of hallucinations during the generative process, thereby preventing the creation of images that are inconsistent with their respective prompts.

Attention Maps for Image Generation. The integration of attention mechanisms has been a cornerstone in improving image synthesis. Cross-attention layers [98] have significantly improved visual fidelity, a concept further explored by [42] to maintain coherence between text prompts and visual outputs. The role of semantic layouts in image synthesis has also been emphasized for quality and interpretability [122]. Extending these concepts, [95] tackled linguistic binding in diffusion models with their SynGen approach, which aligns attention maps with prompt syntax to improve attribute correspondence, optimizing the generation process without retraining the model. [78] introduced a novel method of controlling image synthesis by editing initial noise images, revealing that pixel blocks in initial latent images can be manipulated to influence a specific content generation. [6] proposed eDiff-I, an ensemble of expert denoisers for text-to-image diffusion models, which enhances text alignment and visual quality by specializing models for different synthesis stages. Concurrently, PixArt-α [19] introduces a Transformer-based diffusion model that integrates text via cross-attention into a Diffusion Transformer (DiT) [90], employs a decomposed training strategy, and leverages dense pseudo-captioning, improving image quality up to 1024px with markedly reduced training costs.

Cross-attention maps have been also employed for correcting missing-object problems in image generation. For instance, TokenCompose [123] proposes to fine-tune the Diffusion U-Net by enforcing consistencies between cross-attention maps and relative object segmentation maps. Attend-and-Excite [17] refines the latent embedding on the fly, by maximizing the cross-attention activation for the most neglected subject. Further, cross-attention maps have been found relevant for enhancing binding between represented subjects and their

properties [64]. Similarly, [1] proposes an attention segregation loss to reduce the overlap of different concepts. In contrast, Structured Diffusion [31] adjusts cross-attention representations by exploiting linguistic structures, thus exhibiting enhanced proficiency in the generation of images with complex compositional semantics. Layout Guidance [21] introduces a methodology to direct the image generation process by optimizing cross-attention maps in alignment with a predefined layout. Furthermore, Composable Diffusion [69] adopts a novel approach by segmenting the prompt into distinct conditions and applying a score-based mechanism to refine the image quality.

Following the consensus on the effectiveness of cross-attention as a telltale sign of the fidelity of the generation, our work exploits this information as a factor for predicting the likelihood of accurate generation from early diffusion steps.

Seed Importance in Image Generation. In text-to-image generation, images are significantly impacted by the starting seed of the diffusion process. Indeed, different seeds can produce completely different images, as highlighted by [53], which claims to generate better-aligned images by evaluating multiple seeds. Furthermore, [78, 104] propose to edit the image by manipulating the initial noise instead of steering the generation process with additional mechanisms.

Seed selection has gained relevance in the generation of long-tail concepts [136]. As underlined by [103], in the generation of rare subjects, training predominantly involves exposure to a limited segment of the initial noisy latent space. This selective exposure during training contributes to the generation of unsatisfactory outcomes across a majority of generative seeds at inference time. Hence, the exploration of diverse generative seeds remains a critical aspect in enhancing generative outcomes. To mitigate the occurrence of hallucinations, HEaD+ suggests altering the seed in the event of detecting hallucinations in the generative process.

3.3.3 Preliminaries

In this section, we provide a detailed explanation of latent diffusion models, noise scheduling, and text conditioning, providing the background on relevant definitions and notations. For a broader and more general introduction to diffusion models and latent diffusion, including the

notation shared across the thesis, we refer the reader to Chapter 2, in Sections 2.1 and 2.1.3; here we focus on the specific instance used by our HEaD+ methodology.

Latent Diffusion Models. In this study, we examine the Stable Diffusion (SD) architecture [98], which performs the diffusion process over a latent space of a VAE decoder [55] instead of the traditional pixel image space. Initially, an encoder E converts an image x into a latent code $z = E(x)$. The decoder D seeks to achieve precise reconstruction, ensuring that $D(E(x)) \approx x$. Within this framework, a denoising diffusion probabilistic model (DDPM) [45, 112] operates. This model works on the latent space, creating a denoised version of the input latent z_t at each timestep t. Significantly, the procedure is improved by the inclusion of a conditioning vector $c(y)$, usually obtained from a textual prompt y using a CLIP text encoder [92].

The final training objective consists of minimizing the following loss function:

$$L = \mathbb{E}_{z\sim E(x),y,\epsilon\sim\mathcal{N}(0,1),t} \left\|\epsilon - \epsilon_\theta(z_t, t, c(y))\right\|^2, \tag{3.4}$$

where ϵ_θ is a UNet network [99], with attention layers, that aims to predict the added noise ϵ.

To obtain the final image from the denoised latent representation, the last step involves passing the final latent representation through a the decoder D. This decoder translates the latent space back into the pixel space, thus completing the image generation process. The transition from the final latent state z_0 to the generated image x_0 can thus be described by

$$x_0 = D(z_0). \tag{3.5}$$

For further details, we refer the reader to [98].

Schedulers in Diffusion Models. In diffusion models, schedulers are used to manage the denoising process, guiding image generation by adjusting noise levels throughout the steps. These algorithms facilitate the transition from a noisy latent representation to a refined image without adversarial training. In our HEaD+ approach, we have adapted the scheduler's function to extract the PFI at intermediate diffusion steps. This adjustment is designed to obtain the most precise representation of the final image during the generation

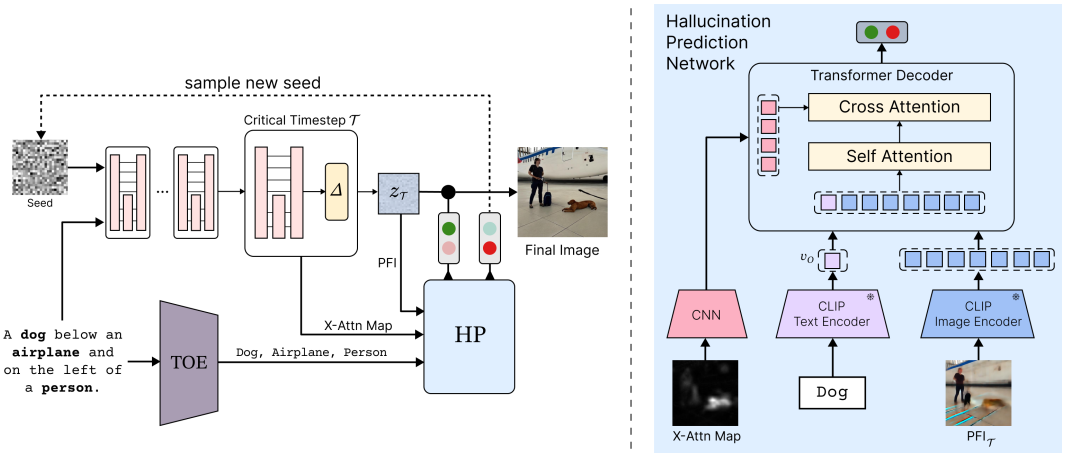


Figure 3.6: Overview of the HEaD+ process and detail of the Hallucination Prediction network. The process starts with extracting subjects from the prompt using the TOE model. At timestep $\mathcal{T}$ of the diffusion process, cross-attention maps and PFI are produced. HP network processes all inputs using specific feature extractors, and combines them using a Transformer-based decoder to predict whether generation should continue or not.

process. We refer to Chapter 2, Section 2.1.4 for a more general discussion of schedulers and sampling efficiency in diffusion models.

The transition of latents z_t at time step t to a subsequent state $z_{t'}$ is governed by the following procedure. The predicted noise ϵ_t is firstly estimated from the output of the UNet model, and the updated latents $z_{t'}$ are computed through the update function of the scheduler Δ, as

$$\begin{aligned} \epsilon_t &= \epsilon_\theta(z_t, t) \\ z_{t'} &= \Delta(z_t, \epsilon_t, t, t'). \end{aligned} \tag{3.6}$$

In this formulation, ϵ_t is derived by the current latents and time step, while Δ is the scheduler update function computing the new latents $z_{t'}$ based on the predicted noise ϵ_t. The specific characteristics of Δ depend on the chosen scheduler, which ultimately determines the intricate dynamics of the denoising process.

Text-Conditioning via Cross-Attention. Stable Diffusion [98] employs cross-attention for text guidance. Indeed, the UNet incorporates self-attention followed by cross-attention layers, functioning across different resolutions (ranging from 64 to 8). For each spatial dimension P of the feature map and text tokens N extracted from y, an attention map

$A_t \in \mathbb{R}^{P \times P \times N}$ is formed, influencing the interaction between text tokens and spatial regions of the image. The attention map extraction for a specific object o, given its index in the prompt, i_o, and the latent representation z_t at time t, can be expressed as

$$A_{o,t} = \mathrm{a}(i_o, z_t), \tag{3.7}$$

where $A_{o,t}$ denotes the attention map associated with the object o at timestep t, extracted from the latent representation z_t. An overview of text conditioning and cross-attention mechanisms in modern diffusion models is provided in Chapter 2, Section 2.1.5.

3.3.4 Hallucination Early Detection+

The primary goal of HEaD+ is to detect and preemptively interrupt faulty generative processes. Our approach is designed to verify the consistency between the input textual guidance and the expected output *during* the diffusion image generation process to save computational time and ensure correct generation. A distinctive feature of our approach is its capability to conduct this verification at a specific intermediate timestep of the diffusion pipeline. To this aim, we designed an Hallucination Prediction (HP) network to detect the risk of hallucination. If an hallucination risk is predicted, our approach forces the generation process to restart using a different initial seed that might lead to better results. In the following, we illustrate the proposed HEaD+ approach at inference time to streamline the generation process and, as a result, enable automatic quality assessment of the final output.

3.3.4.1 Multi-modal Inputs Extraction

Consider a prompt y that includes a set of target objects O to be represented in the image. The procedure for identifying these target objects from the prompt is defined as follows:

$$O = \mathrm{TOE}(y), \tag{3.8}$$

where $\mathrm{TOE}(\cdot)$ denotes the Target Object Extraction function. In this context, the term "objects" pertains to words in the prompt that correspond directly to visible elements in the image. For these elements, the associated cross-attention maps will be derived. Although the

current method is primarily focused on objects, it is designed to be extensible to encompass a broader range of visual concepts, such as attributes or colors, going beyond the limits of object-based extraction.

We then define the *critical timestep*, denoted as $\mathcal{T}$, as a specific step in the diffusion process where cross-attention maps for each object ($A_{O,\mathcal{T}}$) and the Predicted Final Image ($\text{PFI}_{\mathcal{T}}$) are extracted.

In particular, for each object $o \in O$, the cross-attention map $A_{o,\mathcal{T}}$ is obtained by applying the function $a(\cdot)$, as described in Eq. 3.7. $\text{PFI}_{\mathcal{T}}$, instead, represents the prediction of the expected outcome at the end of the generation process, using only information available at timestep $\mathcal{T}$. In particular, the scheduler projects the latents at $\mathcal{T}$ to the final step, and the decoder translates these predicted latents into the image space. Formally, this process can be defined as follows:

$$\begin{aligned} \epsilon_{\mathcal{T}} &= \epsilon_\theta(z_{\mathcal{T}}, \mathcal{T}) \\ z_0^{\mathcal{T}} &= \Delta(z_{\mathcal{T}}, \epsilon_{\mathcal{T}}, \mathcal{T}, 0) \\ \text{PFI}_{\mathcal{T}} &= D(z_0^{\mathcal{T}}), \end{aligned} \tag{3.9}$$

where $\epsilon_{\mathcal{T}}$ represents the predictive noise obtained from the UNet model at critical timestep $\mathcal{T}$. The function Δ updates the latents $z_{\mathcal{T}}$ to the predicted latents at the final timestep, denoted as $z_0^{\mathcal{T}}$. Finally, the decoder D translates these predicted final latents into the Predicted Final Image, $\text{PFI}_{\mathcal{T}}$, which represents a global snapshot at the given timestep and is utilized for predictions across all specified objects.

As an additional input for the model, we extract an embedding vector for each object. This vector is produced using CLIP [92] from the text of the requested objects extracted from the prompt y. Specifically, for each object, we have $v_o = \text{CLIP}_{\text{Text}}(y_o)$ obtained by applying the CLIP model to the textual representation y_o of each object.

Examples of PFIs extracted at different timesteps are shown in Fig. 3.7. $\text{PFI}_{\mathcal{T}}$, the attention maps $A_{O,\mathcal{T}}$ and the textual embeddings v_O, enable the HP network to meticulously assess and predict the presence of specified objects in the final generated image, ensuring a coherent and accurate output aligned with the initial textual guidance.

3.3.4.2 Hallucination Prediction Network

At a specific timestep $\mathcal{T}$ of the generation, the Hallucination Prediction network evaluates each object $o \in O$ individually, utilizing the corresponding cross-attention map $A_{o,\mathcal{T}}$, the textual embedding vector v_o, and the Predicted Final Image, $\text{PFI}_{\mathcal{T}}$, as inputs. Subsequently, the network generates a binary output that indicates whether the target object is present or absent in the final image. Formally, the process can be written as

$$H_o = \text{HP}(A_o, \text{PFI}_{\mathcal{T}}, v_o), \tag{3.10}$$

where H_o is the binary prediction for object o. An image is considered complete if $\forall o \in O, H_o = 1$, otherwise, at inference time, the process must restart with a different seed.

The HP architecture consists of a Transformer model with cross-attention layers that integrate the three input streams ($\text{PFI}_{\mathcal{T}}$, $A_{o,\mathcal{T}}$, v_o) through self-attention and cross-attention mechanisms. Initially, all streams are elaborated separately. Indeed, the PFI is subjected to processing via the visual CLIP backbone, with the extraction of features from the last attention layer. This is followed by the concatenation of the textual token v_o to this sequence. Simultaneously, a three-layer convolutional network processes the cross-attention map.

Following these operations, both outputs are adjusted to the embedding dimension of the transformer model, predetermined at 192, by employing 1D convolution. This process sets the stage for the application of a self-attention mechanism on the stream derived from the CLIP processing. This result (acting as the query) is integrated with a cross-attention layer with the features from the cross-attention maps (serving as keys and values). Skip connections are added following the architecture of the Transformer decoder [119]. This attention block is consistently applied 12 times, concluding with a classification head that takes as input the activation corresponding to the first element of the input sequence, i.e. v_o. The model architecture is shown in Fig. 3.6.

Throughout the training phase, the visual and textual backbones of CLIP are kept frozen. This approach guarantees substantial generalization capabilities towards objects not present in the training dataset, thereby accommodating applications on different datasets and scenarios.

3.3.4.3 Localization Module for Spatial Relations

In addition to object-presence prediction, we introduce an integrated localization module that verifies pairwise spatial relations required by the prompt. This module shares the same backbone as the HP network and differs only in the final prediction head. It operates at the same critical timestep $\mathcal{T}$ using the identical multi-modal inputs, namely $\text{PFI}_{\mathcal{T}}$, cross-attention maps, and textual tokens.

Prompt relation extraction. Let O be the set of objects extracted as in Eq. 3.8. If the prompt specifies positional constraints between two objects, we define the set of relations using a compact extractor. Let $\Pi = \{\text{top}, \text{bottom}, \text{left}, \text{right}\}$. We compute

$$\mathcal{R} = \text{REL}(y, O) \subseteq O \times O \times \Pi, \tag{3.11}$$

where $\text{REL}(\cdot)$ is a lightweight LLM-based relation extractor, applied in parallel with the early diffusion steps, similarly to object extraction described in Sec. 3.3.5.2.

Centroid estimation. For each object $o \in O$, the Localization Prediction (LP) network predicts the centroid $c_o = (x_o, y_o)$ of its final placement in image coordinates using inputs at $\mathcal{T}$. The centroid is obtained by a regression head over the shared features computed from $\text{PFI}_{\mathcal{T}}$, the cross-attention map $A_{o,\mathcal{T}}$, and the text token v_o:

$$c_o = \text{LP}(A_{o,\mathcal{T}}, \text{PFI}_{\mathcal{T}}, v_o), \tag{3.12}$$

where $\text{LP}(\cdot)$ shares the backbone of HP and replaces the final classification layer with a 2D regression layer for (x_o, y_o).

Relation checking. Given two objects i and j, their predicted centroids $c_i = (x_i, y_i)$ and $c_j = (x_j, y_j)$ are compared to verify whether each relation in Π holds (e.g., above/below or left/right) using straightforward coordinate comparisons with a small tolerance (5% of image size). For a relation $r = (i, j, \rho) \in \mathcal{R}$, the localization module produces a binary decision

$$L_{i,j,\rho} = \mathbb{1}[\,\rho(i, j) \text{ holds under } c_i, c_j\,], \tag{3.13}$$

which evaluates whether the positional constraint is satisfied at $\mathcal{T}$.

Joint gating with HP. The overall decision to continue generation at timestep $\mathcal{T}$ is the con-

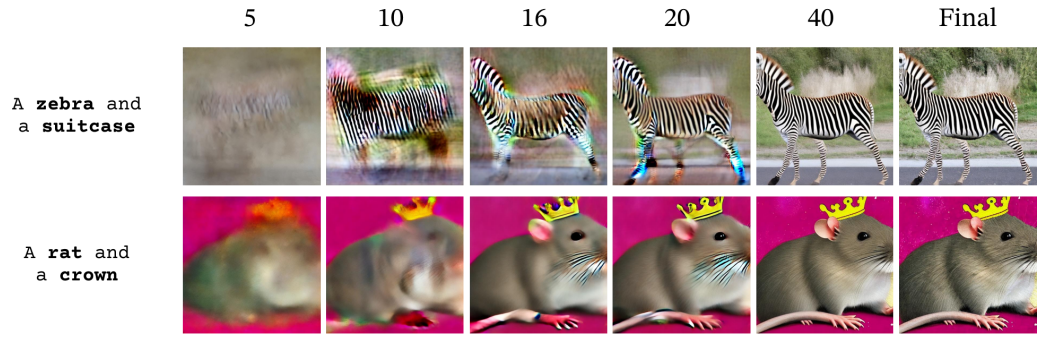


Figure 3.7: Qualitative examples of the Predicted Final Image at different critical timesteps on simple prompts. Already from the 16^{th} step the final image is fully represented and the presence of objects can be predicted.

junction of module decisions. Let H_o be the per-object presence predictions from Eq. 3.10. If no spatial relations are specified ($\mathcal{R} = \emptyset$), we recover the original criterion $\forall o \in O,\ H_o = 1$. Otherwise, we require

$$G = \Big(\forall o \in O,\ H_o = 1\Big) \wedge \Big(\forall (i, j, \rho) \in \mathcal{R},\ L_{i,j,\rho} = 1\Big). \tag{3.14}$$

Generation proceeds if and only if $G = 1$; otherwise, the current run is aborted and restarted with a new seed, consistent with the HEaD+ policy. This module is model-agnostic and can be enabled alongside other modules within the same gating scheme.

3.3.5 InsideGen Dataset

3.3.5.1 Prompts Selection

To train the HP network, we require a dataset containing intermediate outputs from the diffusion process. Since our goal is to train HEaD+ on realistic scenarios, we curated 4,100 prompts from the dataset introduced by [5], ensuring coverage of diverse objects and situations. These prompts are built specifically to stress different generation capabilities, namely spatial composition, size composition, action composition, fairness, and bias. To augment the dimensionality of the dataset and thoroughly investigate output variations influenced by different seeds, we generated 11 images for each prompt using distinct seeds. In total, we generated nearly 45,000 images, both with Stable Diffusion v1.4 and Stable Diffusion v2.0 generators [98]. During generation, we fixed 50 steps of the diffusion process and we collected

the PFI and cross-attention maps A at multiple time steps for every requested objects to give the possibility to make different tests for different critical timesteps $\mathcal{T}$[3]. Dataset was divided in train, val and test (80-10-10), splitting over the prompts, keeping all seeds in the same set.

3.3.5.2 Target Objects Extraction

During the dataset creation process and during real-time inference, objects, denoted as O, are extracted from the textual prompt y, as described in Eq. 3.8. For this purpose, we employed GPT-3.5 [84], selected for its robust zero-shot generalization abilities. This method stands in contrast to conventional text tagging techniques that generally necessitate specific training for each domain.

The extraction procedure is time-efficient and can be executed concurrently with the initial diffusion steps. This parallel processing is made possible by selecting a non-initial critical timestep $\mathcal{T}$, which enables the object extraction to proceed in tandem with the diffusion process.

We instructed the system to use a specific prompt to guide its entity recognition process. The prompt used was as follows:

```
You are a system that is able to recognize entities in a text.
Entities are objects, people, animals, etc.  that have a physical
representation.  Avoid to include abstract subjects.  Do not consider
adjectives in the entities.
```

To enhance the model accuracy, we also provided a few-shot learning approach with relevant examples. This method was crucial in ensuring that the model was focused on extracting only concrete entities while excluding abstract concepts and adjectives, aligning with the objectives of our research and the operational requirements of the HEaD+ pipeline.

3.3.5.3 Automatic Label Creation

A key aspect of our dataset collection approach is an automatic labeling pipeline to verify the presence of specific objects in the generated images. A strong requirement for this label-

[3]In particular, the critical steps $\mathcal{T}$ are chosen as follows: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 25, 40]

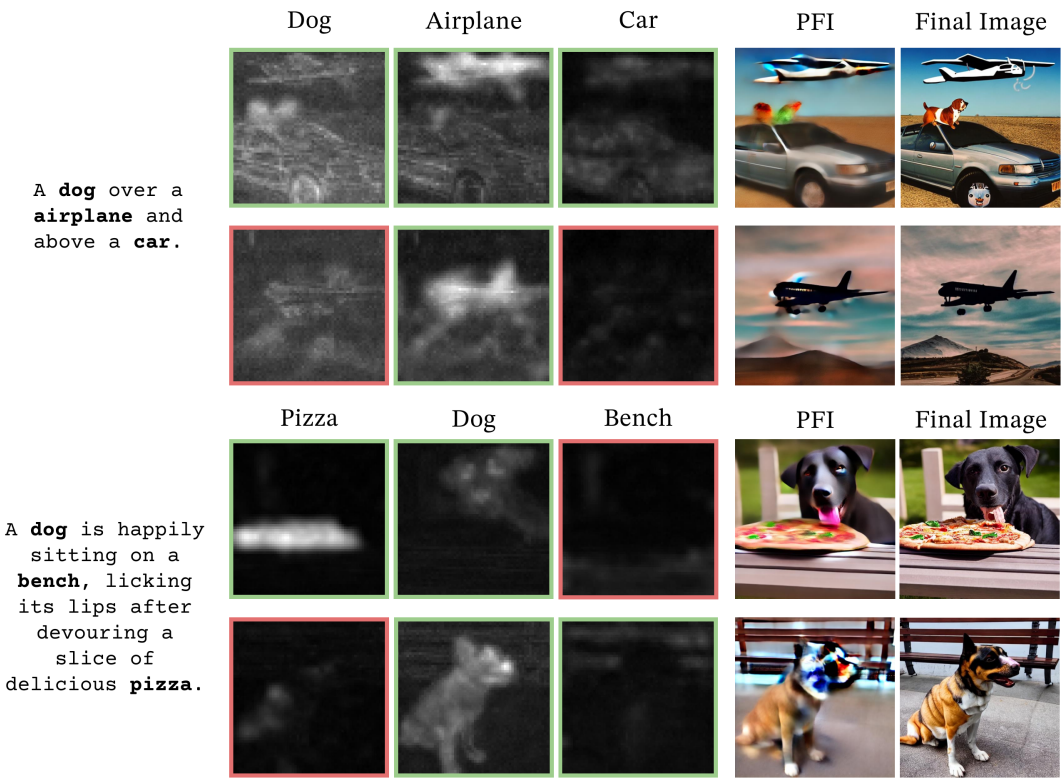


Figure 3.8: Examples of Target Objects Extraction, cross-attention maps, and the Predicted Final Image at timestep $\mathcal{T} = 16$. The cross-attention maps are highlighted with a green border when they correspond with the object in the image; otherwise, they are highlighted in red.

ing system is the ability to operate without a predefined set of object labels, necessitating the use of an open vocabulary approach. This approach provides flexibility and adaptability in identifying a wide range of objects, regardless of their characteristics. According to this objective, we adopted OWLv2 [80], an open vocabulary detector renowned for its robust detection capabilities and for providing confidence scores for each identified object. Furthermore, we leverage OWLv2 to extract object centroids, which are crucial for training the LP network. The ability of this model to accurately detect and quantify the presence of various objects significantly improved the reliability and integrity of our dataset. The likelihood of achieving a *complete* generation, meaning all requested objects are present in the final image, for each object count and model type is depicted in Fig. 3.5(a).

3.3.5.4 InsideGen qualitatives

We present different qualitative samples of the InsideGen dataset in Figure 3.8. In these examples, we performed the object extraction pipeline following the procedure detailed in Section 3.3.5.2, and generated the images using Stable Diffusion 1.4 [98]. Notably, first insights on subject hallucinations are still detectable at timestep 16 of the generation process. For instance, considering the prompt `A dog over a airplane and above a car`, the second row does not represent either the `dog` or the `car` in its PFI. Moreover, the cross-attention maps of these missing subjects are less emphasized compared to the upper row, where all the objects are well represented. Similar outcomes are observed in the prompt `A dog is happily sitting on a bench, licking its lips after devouring a slice of delicious pizza`. Indeed, `pizza` is missing from both the final image and the PFI in the example in the fourth row. Compared to the third instance, where all the subjects are well-represented in the PFI, the cross-attention map is more activated in the case of `pizza` subject.

3.3.6 Experiments

Training and implementation details. The training for the Hallucination Prediction network is conducted using a learning rate of 5×10^{-4}, with a reduction on plateau strategy. Specifically, the learning rate is reduced by a factor of 0.5 if there is no improvement observed for 10 epochs, demonstrating a patient approach to convergence. HP is trained on SD2 with a fixed batch size of 128. From a technical perspective, the Transformer Decoder is equipped with 12 self-attention and cross-attention blocks, each of which has 3 attention heads. The convolutional layers that process the cross-attention maps instead have a 3×3 kernel with stride 1 and padding 1 with an output channel of 32, 64, and 128 for respectively the first, second, and third layers. Following each convolutional layer, batch normalization, ReLU activation, and MaxPooling operations are sequentially applied. An exception resides in the terminal convolutional layer, which incorporates adaptive average pooling to refine the dimensionality to a 14×14 feature map. This resultant map is then flattened and projected to the 192 embedding dimension of the Transformer decoder.

Model	**TN-Rate**	**Recall**	**Time Saved**
HEaD+ 5	50.16	90.22	**37.73**
HEaD+ 8	56.93	91.06	36.37
HEaD+ 10	58.37	90.09	30.03
HEaD+ 12	63.36	91.15	30.11
HEaD+ 14	61.65	93.42	32.91
HEaD+ 16	64.42	93.60	30.03
HEaD+ 18	66.86	93.13	25.64
HEaD+ 20	65.43	**94.81**	26.85
HEaD+ 25	**76.95**	93.40	14.18
HEaD 8	43.73	85.67	11.55
HEaD 25	52.16	88.02	-11.70

Table 3.3: TN-Rate, Recall, and percentage of time saved for the HEaD+ model at different timesteps. In addition to the model name, the timestep $\mathcal{T}$ used for training the model is specified. HEaD refers to our previous method [9].

3.3.6.1 Time Saving Experiment

This experiment is designed to evaluate a specific aspect of the performance of our method, namely its ability to achieve a *complete* generation. Traditional models typically require a full run followed by a post-generation evaluation, restarting the process if the output is not *complete*. The unique advantage of employing HEaD+ lies in its capability to perform evaluations during the generation process itself, potentially allowing for much earlier restarts when non-ideal outputs are detected. This can significantly reduce the overall time spent on generating and evaluating images. It is important to note that, for the purposes of this experiment, we do not account for the time that would normally be consumed by a model without HEaD+ in conducting post-generation evaluations.

HP Performance Impact. The HP network predicts whether to continue or halt the image generation process. When a True Positive (TP) occurs, the correct generation proceeds uninterrupted, having no effect on computation time. Conversely, a False Positive (FP) allows an incorrect generation to continue without interruption, thus missing an opportunity for time savings, but still not impacting computation time. A True Negative (TN) indicates an incorrect generation has been correctly halted, leading to time savings. Finally, a False Negative (FN) means a correct generation is mistakenly stopped, resulting in a loss of time.

The HP network is expected to demonstrate proficient recall performance, characterized

by minimal FNs, thereby preventing the unwarranted cessation of correct runs. Concurrently, the network should achieve high true-negative rate, characterized by a minimal number of FPs relative to TNs. This is critical to ensure the timely interruption of incorrect generations, thereby enhancing time-saving. Consequently, in the training phases of the HP network, emphasis was placed on achieving both recall and true-negative rate metrics.

Best $\mathcal{T}$ selection. The most impactful hyperparameter in HEaD+ is $\mathcal{T}$, the step in which the HP network is applied. An earlier detection, with a small $\mathcal{T}$, can potentially lead to greater time savings in the case of TNs (correct hallucination identification). Nevertheless, this approach is met with a notable challenge: during the preliminary phases, the quality of cross-attention maps and PFIs is suboptimal, thereby affecting the performance efficacy of the HP network.

To quantify the time saved or lost using the model, we conducted Monte Carlo simulations until a *complete* generation is found. The detailed algorithm and pseudo-code are provided in the Appendix. We compared the same model trained on different $\mathcal{T}$ and results are shown in Table 3.3. Recall values are maintained using smaller $\mathcal{T}$ values while TN-Rate changes significantly. The overall time saved for a *complete* generation in an ideal scenario with HEaD+ used with $\mathcal{T} = 5$ is the highest, with a potential 37.73% of time saved over a normal version of SD2. Early detection has the biggest impact considering recall values are similar.

We compared the current models with HEaD [9] models which exhibit significantly poorer performance on the InsideGen dataset. For instance, when comparing at the same timestep $\mathcal{T} = 25$, HEaD+ increases time-saving by 25.88%. Further, HEaD obtains no benefit (-11.70%) when introduced in this complex scenario. Similarly when comparing the methodologies at lower timestep $\mathcal{T} = 8$, HEaD+ obtains a gain of 26.18% in time-saving.

3.3.6.2 Generation Quality Comparison

To compare the final image quality after a full generation, we followed the evaluation protocol introduced in [123]. Specifically, in this benchmark, the text-to-image model is prompted to generate five subjects (i.e. *A, B, C, D, E*) with the caption *A photo of A, B, C, D, and E*. Subsequently, an open-vocabulary object detector [80] is asked to detect the presence of the subjects in the generated images. In Table 3.4 the performance of different models, in

Method	COCO			
	MG2	MG3	MG4	MG5
SD 1.4 [98]	$90.72_{1.33}$	$50.74_{0.89}$	$11.68_{0.45}$	$0.88_{0.21}$
SD 1.4 w/ HEaD+	$\mathbf{95.44}_{0.67}$	$\mathbf{65.03}_{1.76}$	$\mathbf{19.53}_{1.73}$	$\mathbf{1.78}_{0.24}$
Composable Diffusion [69]	$63.33_{0.59}$	$21.87_{1.01}$	$3.25_{0.45}$	$0.23_{0.18}$
Layout Guidance [21]	$93.22_{0.69}$	$60.15_{1.58}$	$19.49_{0.88}$	$2.27_{0.44}$
Structured Diffusion [31]	$90.40_{1.06}$	$48.64_{1.32}$	$10.71_{0.92}$	$0.68_{0.25}$
Attend-and-Excite [17]	$93.64_{0.76}$	$65.10_{1.24}$	$28.01_{0.90}$	$6.01_{0.61}$
Token Compose [123]	$98.08_{0.40}$	$76.16_{1.04}$	$28.81_{0.95}$	$3.28_{0.48}$
Token Compose w/ HEaD+	$97.61_{0.40}$	$81.27_{1.40}$	$35.33_{1.97}$	$4.93_{0.57}$
PixArt-α [19]	$98.59_{0.49}$	$83.19_{2.18}$	$42.80_{4.09}$	$8.80_{2.10}$
PixArt-α w/ HEaD+	$\mathbf{99.25}_{0.28}$	$\mathbf{89.19}_{0.77}$	$\mathbf{51.65}_{1.89}$	$\mathbf{12.72}_{1.46}$

Table 3.4: Comparison with state-of-the-art methodologies, on the evaluation protocol of [123]. Results without HEaD+ are taken from [123].

their base form or with HEaD+, are presented. This split includes 80 objects extracted from COCO [65] combined as previously defined, building 1000 different prompts. Considering the capability of each model to potentially introduce hallucinations, the generated images may feature between one to five subjects. The metric MG-N is utilized to quantitatively ascertain the count of generated images that accurately portray a minimum of N subjects as requested in the initial prompt. Results are reported with mean and standard deviation over 10 seeds following the original implementation.

To promote a fair comparison, we impose a limitation on the maximum number of restarts for HEaD+, specifically setting this limit to five iterations. This constraint is designed to standardize the inference time, facilitating a direct comparison with alternative methodologies. In instances where a *complete* image fails to materialize within these iterations, the seed with the highest number of objects predicted by HEaD+ is chosen for the generation. Further, the HEaD+ model is trained on SD2 on InsideGen, which differs significantly from the datasets used in these experiments. The number of inference steps is set to 50 for SD models and 20 for PixArt, maintaining the default settings; similarly, the critical timestep $\mathcal{T}$ is set to 25 and 10 for SD models and PixArt, respectively. The defined HEaD+ methodology proves beneficial in enhancing the overall quality of the output.

As illustrated in Table 3.4, the adoption of HEaD+ enhances the detection of object presence in comparison to SD 1.4 [98]. Specifically, an average increase of 14.29% in the detection of three objects is observed when HEaD+ is integrated with SD1.4 compared

Name	TN-Rate	Recall
Text + XA	62.35	93.82
Text + PFI	74.84	91.21
PFI + XA	33.72	**95.22**
HEaD+	**76.95**	93.40

Table 3.5: Ablation of different input types: TN-Rate and Recall for different model settings. Experiments done at $\mathcal{T} = 25$. In the Table, XA and Text represent cross-attention maps and v_o, respectively. HEaD+ is the model with all input types that we have used throughout our experiments.

to the raw SD. Additionally, improvements of 7.85% and 0.9% are noted for MG4 and MG5, respectively. When compared to Composable Diffusion [69], Layout Guidance [21], and Structured Diffusion [31], HEaD+ with SD1.4 surpasses them in nearly all the MG categories. Specifically, for MG3 metric, SD1.4 equipped with HEaD+ obtains a gain of 43.16%, 4.8%, and 16.39% over Composable, Layout Guidance, and Structured Diffusion respectively.

Significantly, HEaD+ was initially trained using the model SD2, which demonstrates the capability of the model to adapt to various diffusion models beyond the one it was originally trained on. This adaptability is further corroborated by performance gains when HEaD+ is used with TokenCompose [123]: HEaD+ yields improvements of 5.11%, 6.52%, and 1.65% on MG3–MG5 over TokenCompose. Moreover, with Transformer-based PixArt-α [19], HEaD+ achieves the best MG scores across MG2–MG5 (Table 3.4). Overall, these results confirm that HEaD+ is model-agnostic, effectively complementing both UNet- and Transformer-based diffusion generators without retraining the target model.

3.3.6.3 Ablation on Input Types

To understand the individual impact of cross-attention maps, PFI, and the textual feature vector on the final assessment, we conducted an ablation study. This involved creating variations of the HP network, each excluding one of the inputs, and comparing their performance. The analysis reveals that the most accurate results are achieved when all inputs are incorporated. Specifically, when the textual token v_o is dropped, the performance decreases by 24% in TN-Rate. The strong interaction between CLIP features extracted from the PFI and v_o is relevant for HEaD+, as it precisely specifies the target elements that the model needs to identify within the visual inputs. Indeed, when only textual information and cross-attention

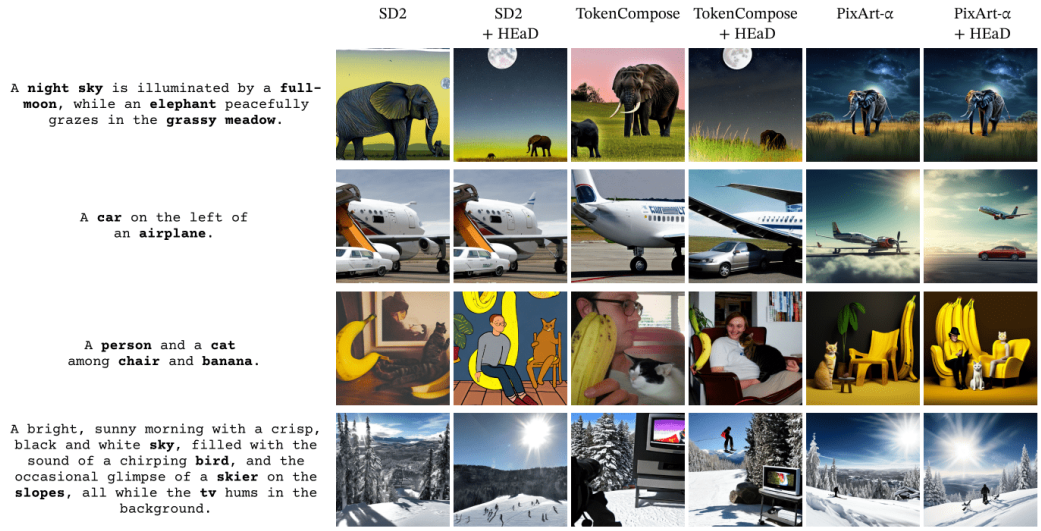


Figure 3.9: Qualitative analysis on the comparison between a normal Diffusion Model and the same version using HEaD+. The same starting seed was used for all the experiments on the same prompt.

maps are combined performance still takes a decrease of 12.41% on TN-Rate, proving the coupling between the textual input and PFI. Although the cross-attention map has the least impact compared to the other inputs, it increases Recall by 1.19% and TN-Rate by 2.11% compared to the model without it. This is attributable to the ability of the cross-attention map to guide the focus of the model toward certain regions within the image, facilitating the verification of object presence.

3.3.6.4 Qualitative Comparison

In Figure 3.9, we provide some qualitative comparisons of generated images by SD2 and TokenCompose with and without HEaD+, evaluated on prompts from InsideGen test set. The four rows represent different contributes of HEaD+. In the top row, both SD2 and TokenCompose fail to represent correctly the subject `moon`. Triggered by this hallucination, HEaD+ reiterates with another seed until a *complete* generation is reached. Conversely, in the second row, SD2 accurately renders both `car` and `aircraft`, inducing HEaD+ to allow the generative process to continue, thereby yielding an identical result. In contrast, TokenCompose exhibits a failure by negating the presence of the `car` during its generative process, a deviation promptly addressed by HEaD+ through the termination of generation

and the introduction of a new seed. Similarly, in the third row, SD2 hallucinates `person` and TokenCompose `chair` while HEaD+ corrects them. The concluding row presents a very complex scenario with five objects in the prompt; SD2 solely depicts `slopes`, and HEaD+, despite its efforts, fails to generate all intended objects, reaching the maximum number of iterations. Nonetheless, in its interaction with TokenCompose, HEaD+ successfully intervenes by incorporating the omitted `skier`, showcasing its ability to correct hallucinations effectively.

3.3.6.5 Localization Module Results

Training. We train the Localization Prediction (LP) network with the same optimization setup described in Section 3.3.6, using object centroids from InsideGen for non-hallucinated objects.

Evaluation protocol. To validate the LP network, we sample two objects from COCO and compose a two-object prompt with a single positional relation $\rho \in \Pi = \{\text{top}, \text{bottom}, \text{left}, \text{right}\}$. For each prompt, we generate 5 images with different seeds and collect inputs at the critical timestep $\mathcal{T}$, exactly as for HP. Following the procedure in Section 3.3.6, we evaluate three model configurations: the base generator, the base generator with HEaD+, and the base generator augmented with both HEaD+ and the LP network. We verify the positional relation on generated images by applying OWLv2 to detect object centroids and checking whether ρ holds within tolerance of a 5% of the image size. Subsequently, we report the following metrics: (i) MG2: the percentage of images in which both objects are present; (ii) MG_{loc}: the percentage of images in which both objects are present and the specified positional relation is satisfied; and (iii) Relation Consistency: the proportion of correct relations among all images in which both objects are present.

Discussion. Table 3.6 presents the evaluation of the localization module. Consistent with prior findings, incorporating HEaD+ yields systematic improvements in MG2 over the base models. Furthermore, enabling the localization module brings additional gains in both MG_{loc} and Relation Consistency, as generations predicted to violate the specified positional relation are terminated early and resampled. For example, Relation Consistency increases by 22.0%, 13.9%, and 17.5% relative to SD 1.4, SD 2, and TokenCompose with HEaD+ but without the localization module, respectively. Relative to the base models, the most

Method	MG2	MG_{loc}	Relation Consistency (%)
SD 1.4	39.6	19.2	48.5
SD 1.4 w/ HEaD+ wo/ localization	60.2	20.0	33.2
SD 1.4 w/ HEaD+ w/ localization	**61.2**	**33.8**	**55.2**
SD 2	51.6	22.2	43.0
SD 2 w/ HEaD+ wo/ localization	**69.2**	29.0	41.9
SD 2 w/ HEaD+ w/ localization	68.8	**38.4**	**55.8**
Token Compose	67.6	32.8	48.5
Token Compose w/ HEaD+ wo/ localization	**71.3**	35.7	50.0
Token Compose w/ HEaD+ w/ localization	70.8	**47.8**	**67.5**

Table 3.6: Impact of the localization module. MG2: both objects present. MG_{loc}: both objects present and the required relation satisfied. Relation Consistency (%) is the percentage of correct relations among images where both objects are present.

substantial improvement is observed with TokenCompose: equipping it with HEaD+ and the localization module raises Relation Consistency from 48.5% to 67.5%, indicating that relation-consistent generations are encouraged more effectively than with either the base model alone or the base model plus HEaD+.

3.3.7 Conclusions

This paper presents HEaD+, a state-of-the-art methodology designed to significantly improve the efficiency and accuracy of image generation processes using diffusion models. The key to our innovation lies in the integrated use of cross-attention maps, Predicted Final Image, and textual data to predict the outcome of the image generation process. HEaD+ has been shown to improve the generation fidelity of the requested objects inside the final image. A crucial takeaway from our research is that HEaD+, in its current form without retraining, can be seamlessly integrated with various diffusion models, reliably ensuring that all specified objects are accurately represented in the final image. The internal operations of the diffusion model are not affected by the functionality of HEaD+. Furthermore, we created InsideGen, which is a dataset that consists of intermediate diffusion output (attention maps and PFIs) with annotated hallucinations. This resource will facilitate further investigation into how the internal generation data can be leveraged to refine the image generation process as a whole.

3.3.8 Monte Carlo HEaD+ simulation

```
# complete_generation_probability: probability of complete image
# recall: recall of the HP network
# TN_Rate: TN_Rate of the HP network
# time_per_model_iteration: time for completing a generation
# max_step_used: last step used for HEaD evaluation
# num_objects: number of objects to evaluate
# total_steps: number of generation step, 50 for SD2
# num_simulations: number of Monte Carlo simulations

# Computing time when HEaD model detects failure
time_used_per_TN = (max_step_used / total_steps) * time_per_model_iteration
# Time with HEaD approach
time_with_head = 0

for _ in range(num_simulations):
    success = False
    while not success:
        # Generate an image that will be complete
        # with a probability of complete_generation_probability
        actual_success = random() < complete_generation_probability
        if actual_success:
            # HP network must predict all 1s to halt the generation
            model_predicts_success = all(
                random.random() < recall for _ in range(num_objects)
            )
            if model_predicts_success: # TP
                time_with_head += time_per_model_iteration
                success = True
            else: # FN
                time_with_head += time_per_model_iteration
        else:
            # The generation has at least one object hallucinated.
            # The HP must find at least one hallucinated object to
            # restart the generative process
            model_predicts_failure = any(
                random() < TN_Rate
                for _ in range(num_objects)
```

```
            )
            if model_predicts_failure: # TN
                time_with_head += time_used_per_TN
            else: # FP
                time_with_head += time_per_model_iteration

# Time with HEaD approach
avg_time_with_HEaD = time_with_head / num_simulations

# Time without HEaD approach
avg_time_no_HEaD = time_per_model_iteration / complete_generation_probability

return 1 - avg_time_with_HEaD / avg_time_no_HEaD
```

Listing 3.1: Python pseudo code for HEaD+ Monte Carlo simulation.

The Python pseudocode detailed in Algorithm 3.1 simulates the time savings achieved by implementing the HEaD+ approach within the image generation process. Its effectiveness depends on the performance of the model, particularly in terms of recall, true-negative rate, and the number of subjects present $|O|$. HEaD+ analyzes each subject independently, and it only requires one of the objects to be predicted as absent to halt the generation and restart with a new seed. The time saving occurs when the model incorrectly generates an image, i.e. a subject is not present, and HEaD+ is able to predict this and immediately restart the generation with a different seed. The time saved in each of these instances is dependent on $\mathcal{T}$, which represents the timestep in which the prediction is done.

3.3.9 Additional qualitative comparisons

In addition to the main qualitative analysis in Fig. 3.9, we provide a larger panel of comparisons between base generators and their counterparts equipped with HEaD+. As illustrated in Fig. 3.10, HEaD+ systematically recovers missing subjects and improves compositional correctness across SD 1.4, TokenCompose [123], and PixArt-α [19]. These examples complement the quantitative gains discussed in Section 3.3.6 and in Table 3.4.

Figure 3.10: Extended qualitative comparison across SD 1.4, TokenCompose, and PixArt-α with and without HEaD+ on multi-object prompts. HEaD+ helps recover missing subjects and yields images that better satisfy the requested compositions.

3.4 Discussion and Limitations

From a broader perspective, the two HEaD+ papers illustrate a general strategy for *evaluation-centric control* of generative models: using predictive signals to adapt how much compute to spend, and which generative path to follow, rather than treating all prompts and trajectories as equally difficult. In text generation, similar ideas already underpin routing schemes where an initial estimate of query complexity is used to select between smaller and larger language models [107, 30]; analogously, our work suggests that image generators could first estimate the "complexity" of satisfying a given prompt, or run a cheap preliminary generation, and then choose between lighter and heavier diffusion backbones, or between different guidance policies, based on the predicted risk of hallucination. Pushing this idea further, one could imagine pre-generation predictors that operate before any full-resolution sampling takes place, using only the prompt and possibly low-cost proxies (such as low-resolution or few-step generations) to decide which model family to use, whether to enable hallucination early detection, or when to fall back to more powerful but expensive architectures. These directions point towards a future in which evaluation signals inform not only when to stop a given trajectory, but also how to allocate generative resources across a heterogeneous pool of models and tasks.

The HEaD+ line of work provides a concrete demonstration that evaluation signals can be used *inside* the denoising loop, but it also comes with important limitations. First, the focus is deliberately narrow: both the ECCV and IJCV studies target a specific class of hallucinations, namely missing objects in multi-object prompts (and, in the journal version, simple positional relations), without addressing other failure modes such as incorrect attributes, style mismatches, global semantic inconsistencies or text rendering errors. Even within this scope, hallucination prediction is not perfect: the detector can fail to flag faulty generations or prematurely interrupt runs that would have converged to acceptable images, and its effectiveness is tightly coupled to the recall and true-negative rates achieved on the training distribution.

Second, the proposed framework is intrinsically tied to diffusion-based image generators with sufficiently many denoising steps. The approach assumes access to intermediate latents, cross-attention maps and Predicted Final Images at a chosen critical timestep, and its time-

saving benefits rely on having a substantial remaining portion of the trajectory that can be skipped when a hallucination is detected. This makes HEaD+ a natural add-on for multi-step latent diffusion models, both UNet- and Transformer-based, but also means that its advantages diminish as the field moves towards few-step diffusion and distillation-based samplers [72, 101, 138] and alternative generative architectures such as flow-based models [56], where early stopping offers less headroom for savings. Designing hallucination predictors that operate primarily on model-agnostic signals (prompts, generic embeddings, coarse outputs) would further improve portability and ease of deployment. As such, the degree to which the learned hallucination predictors and time-saving estimates transfer to other domains, modalities (e.g., video, 3D) or proprietary generators remains an open question, and points to promising directions for future work on more general, robust and lightweight in-loop evaluators and on evaluation-centric control mechanisms that extend beyond diffusion architectures altogether.

Chapter 4

Human-Aligned Evaluation for Text-to-Image Models

4.1 Chapter Overview

In the evaluation-centric lifecycle of Figure 1.1, this chapter focuses on the first verification stage that takes place *after* sampling has completed but *before* any editing is performed, asking whether the image produced by a generative model is structurally reliable with respect to the user's request.

ViCE operates at the level of the final image, performing a global consistency check that is agnostic to the internal architecture of the generator, without access to its latent states or training process. The central challenge it addresses is that existing automatic metrics, from distributional scores such as FID and IS to alignment-oriented measures like CLIPScore, BLIP-ITM or captioning-based proxies, fail to robustly capture instruction-level faithfulness, compositional structure and logical coherence. In practice, human judgment remains the gold standard for assessing whether a generated image truly satisfies a prompt, but systematic human evaluation is costly and difficult to scale. This creates a tension between the need for reliable, human-aligned benchmarks and the practical constraints of evaluating ever-larger generative models.

The key idea of this chapter is to narrow that gap by explicitly modelling the way humans

reason about an image, grounding the method in the thesis's pillars of **Explainability** and **Granularity**. Rather than compressing an entire prompt-image pair into a single embedding similarity or classifier score, ViCE introduces *Visual Concept Evaluation*: an evaluation framework that mimics human cognitive behaviour. To achieve explainability, it avoids black-box scoring in favour of a structured question–answering process that yields interpretable rationales. To support this, it relies on granularity: it extracts the atomic visual concepts implied by a prompt (and, for editing tasks, by the input image) and verifies each one individually. At a high level, the framework decomposes evaluation into three intertwined stages: concept and prompt analysis, question generation, and image-question answering, whose outputs are aggregated into interpretable scores that reflect how well the generated image realises the requested concepts. By design, these scores are not only scalar numbers but also come with a trail of questions and answers that can be inspected to understand *why* a generation is deemed successful or deficient.

Concretely, the ViCE pipeline, detailed in the remainder of this chapter, instantiates this human-aligned reasoning loop using a Large Language Model as a "reasoning agent" and a Visual Question Answering backbone as an image interpreter. These components interact in an iterative question–answering process over the visual concepts extracted from the prompt and, when relevant, from the input image, gradually refining their understanding of the generated scene. The final evaluation score is then derived from this reasoning trace, so that ViCE turns evaluation into an explicit, stepwise process that more closely mirrors how humans naturally interrogate and judge visual content.

From the perspective of this thesis, an important strength of ViCE is its generality across tasks. Although the core experiments focus on text-to-image generation without ground-truth references, the same visual concept and questioning machinery can be extended to Image Targeted Editing, where the goal is to apply precise semantic changes to an input image while preserving irrelevant content. This makes the framework a natural bridge towards the later disentanglement evaluation developed in DICE, while still remaining architecturally decoupled from any specific editing model.

Empirically, the chapter shows that this evaluation-by-questioning approach aligns more closely with human judgments than widely used automatic metrics. Across a range of prompts and generated images, ViCE achieves substantially higher agreement with human

scores than traditional distributional and alignment-based metrics, while remaining competitive with (and often more balanced than) recent LLM-based evaluation schemes. The work has been accepted to the *Brave New Ideas* track at ACM Multimedia, which further highlights its role as a speculative, forward-looking proposal for evaluation-centric generative AI.

4.2 Let's ViCE! Mimicking Human Cognitive Behavior in Image Generation Evaluation

4.2.1 Brave Idea Introduction

Quantitatively assessing the results of image generation models is a complex task. The challenge is clear when we consider simpler generative models such as VAEs, and it escalates when we delve into unsupervised or self-supervised models like GANs or Diffusion Models [112, 26]. Often, the evaluation of proposed models is based on some measurements in the latent space or on specific features extracted from the generated images [43, 7]; sometimes it is associated with checking the sharpness and diversity of results in comparison to a reference test dataset [44]. Thus, whether the model generates images from a single text prompt or modifies an existing image based on a textual input (a process commonly referred to as Image Target Editing), accurately gauging their effectiveness is an ongoing challenge.

In fact, thus far, the only universally agreed upon evaluation methodology is the ultimate human judgment.

In recent years, new models and commercial products have been introduced that offer unprecedented perceptual quality and realistic representation. Systems for prompt-based generation, such as those based on Diffusion Models [112, 45, 6], and multimodal models that allows for partial modification of the input image [11, 134], at first glance, seem to deliver satisfactory results. However, appearances can be misleading.

Given that the research community largely agrees that metrics are necessary for benchmarking these generative process, the key question here is: how can we evaluate the effectiveness of a generative process triggered by text or multimodal input without ground truth? This includes other underlying inquiries like: does the output image meet perceptual and

Figure 4.1: Different types of Image Generation tasks. *Top*: in a multimodal Image Targeted Editing setup, given an input image paired with a textual instruction, the generative system is called to modify the former according to the latter. *Bottom*: in a cross-modal image generation setup, the generative system is called to produce an image based on the textual description provide as the sole input.

semantic expectations? Does the output image meet the constraints of the textual prompt? Does the image accurately reflect changes requested during a generative editing process in a multimodal setting? Until now, no reference-less quantitative scoring framework has been proposed.

Let's consider a situation where we ask a generative system to partially modify the content of a given image: we could request to only change the color of the motorbike to green (see Figure 4.1). Here, a concrete challenge lies in determining whether the system can fulfill these kinds of requests without introducing unintended alterations, and still effectively implement the desired modifications and maintain a high level of perceptual quality. In other words, *how can we evaluate whether the system has precisely executed the requested changes, neither exceeding nor falling short, and has produced an aesthetically pleasing output?*

Right now, human input is crucial for this process. This involves either having humans annotate data beforehand or getting humans to assess the result after the generative process. Such human feedback can also be harnessed to continuously improve the models, e.g. through reinforcement learning methods [88].

Despite the complex task of perfectly imitating human judgment, we can aim to emulate the strategy employed by humans to make judgments, mainly by asking and answering questions. This provides a viable approach to modelling human judgement within an AI

system. This is the essence of what we term as human-aligned ViCE.

To sum up, our contributions include:

- a novel interpretation method for images based on question answering, that reflects the human cognitive process;
- a universal evaluation protocol applicable to all image generation tasks, including Image Targeted Editing (ITE);
- an AI system, which leverages Large Language Models (LLMs) for dynamic question generation, which circumvents reliance on a static question pool;
- a semantic complement to perceptual quality metrics, contributing additional depth to evaluations, rather than attempting to replace existing metrics;
- enhanced alignment with human evaluations, bolstering the trustworthiness and authenticity of AI-generated assessments.

Embarking in this direction is indeed bold, as it seeks to bridge the cognitive gap between artificial intelligence systems and humans.

4.2.2 Related Works

Text-to-Image generation and editing. Over the past few years, many approaches have emerged in the realm of Generative AI, aiming to enhance the efficacy of image-generation tasks. Notably, advancements in the application of GANs [38, 132, 105, 117] and Diffusion Model [112, 57, 45, 26, 93, 6] have significantly elevated the current state-of-the-art with regard to the Text-to-Image paradigm, which involves the generation of an image from a given textual description (or prompt). For example, in [98] each step of the diffusion process is conditioned on the textual prompt input by the user, resulting in an output aimed at representing the starting textual concept.

Given the significant advances in this field, more recent efforts have enlarged the scope of the Text-to-Image paradigm to encompass human-written instructions for image editing [11, 54, 134]. In this particular task, the objective is to manipulate the semantics of an image using a textual prompt, while simultaneously avoiding any undesired alterations to the image

itself. A concise introduction to diffusion-based text-to-image generators and MLLMs that underpin these systems is provided in Chapter 2, in Sections 2.1 and 2.2.

Metrics for Automatic evaluation of image generation and editing. Despite the significant efforts by the research community to enhance the qualitative outcomes of image editing and generation, only a limited number of techniques have been proposed to effectively evaluate the produced results of both methods.

As emphasized in [87], current automatic metrics exhibit limited performance in evaluating Text-to-Image generation when compared to human evaluations. Metrics such as Fréchet Inception Distance (FID) [44] and Inception Score (IS) [102] primarily focus on assessing image fidelity, disregarding the alignment between the generated image and the associated text. Conversely, CLIPScore [43] aims to measure the cosine similarity between the image and text tokens that are tokenized using CLIP image and text encoders. However, there are instances [34, 83] where generative models employ this metric to optimize image generation during training, leading to potential biases and unfair measurements at evaluation time.

To address this challenge, [87] propose a solution involving human evaluation as the primary method of evaluating Text-to-Image models. Further, a recently proposed automatic metric, LLMScore [73], despite combining global and local descriptions using a Large Language Model (LLM) into an object-centric visual description, presents some limitations. A broader overview of distribution-level, alignment and VQA-based metrics, together with human evaluation protocols and their limitations, is provided in Chapter 2, in Section 2.3.

A significant drawback is that the generated captions often contain additional details that are not sourced from the image captioners, but instead fabricated by the LLM. Moreover, the final caption does not sufficiently incorporate the requirements and inputs from the original prompt, differing significantly from a human-like evaluation process. Eventually, LLMScore compares this description with the textual prompt used during the generation process and utilizes an LLM to compute the final score. Our proposed metric strives to overcome these issues and to more effectively replicate human visual reasoning by incorporating a pipeline that specifically evaluates the extent to which the generated image fits the textual requests.

Our work shares some commonalities with QuestEval [106], which implemented a similar strategy for text summarization tasks. In QuestEval, concepts crucial to the content were identified by means of question generation and question answering, and the summarized

output was then evaluated based on the presence/absence of the same question/answer pairs.

4.2.3 Visual Concept Evaluation

The process of Visual Concept Evaluation, as we define it, aims to replicate human behavior during the assessment of a generated image. When a human is asked to rate, on a scale of 1 to 10, how well an image generation task has been executed, his/her brain unconsciously starts considering the "visual concepts" they expect to see within the generated image. Visual concepts go beyond basic elements, such as shapes and colors, and include complex aspects such as specific objects and their contextual interaction within a scene.

These concepts are dictated by the initial text that forms the basis for the image generation in the case of traditional image generation tasks. However, for multimodal inputs, these concepts hinge on both the text and the input image. Additionally, evaluators utilize their implicit knowledge to infer other intuitive aspects. For instance, if the prompt is "a cat on the stairs", the evaluator expects to see a cat, of which 1 to 4 legs might be visible, with the paws placed on the steps and a tail. All this information is easily deduced by a human brain and corresponds to the thought process a person goes through when assessing an image.

Hence, it is crucial to encapsulate not only the explicit instructions derived from the prompts, but also the implicit assumptions and expectations that humans naturally make. This brings forth the intricacy of the challenge - it's about recognizing and integrating these nuanced aspects of human cognition into the evaluation framework. Such broader understanding forms the foundation of the Visual Concept Evaluation process and is the key to aligning AI systems closer to human-like image assessment capabilities. To represent the creation of the visual concepts, which we denote as v_i, we can use the following formulas.

Visual concepts are generated from the text T, and, in case of ITE task, the input image I_{input}.

The visual concepts can be represented as:

$$\begin{aligned} V_T &= f(T) = v_1, v_2, ..., v_n \\ V_{TI_{input}} &= g(T, I_{input}) = v_1, v_2, ..., v_m \end{aligned} \tag{4.1}$$

where f translates the text into visual concepts and g translates the text and image into visual concepts, and v_i are the individual visual concepts. For the sake of simplicity and clarity in the following discussion, we will refer to them as V. Humans, likewise, as soon as they receive a prompt to inspect are able to directly generate the visual concepts.

Once visual concepts are formulated, the human being immediately proceeds to examine whether these visual concepts are manifested in the image and how they interact with each other. This exploration is not a simple casual observation, but involves an unconscious questioning process in which the mind raises a multitude of implicit questions and then attempts to answer them using its inherent ability to understand visual content. The same idea drives the ViCE process.

In ViCE, the genesis of the process is marked by the generation of a group of "blind questions." These questions are derived from the use of previously formulated visual concepts. They are called blind because they are not based, unlike the refinement questions, on information that has been seen and processed from the generated image. This can be expressed as:

$$Q_0 = q(V) \tag{4.2}$$

Here, Q_0 denotes the initial set of blind questions, which are generated by applying function q over the visual concepts, V.

Next, the image to be evaluated, I, is examined and, through reasoning, an effort is made to answer the questions and determine the presence of expected elements. This key step requires a comprehensive understanding and interpretation of the image.

$$A_0 = a(I, Q_0) \tag{4.3}$$

In the equation above, A_0 are the initial answers. The function a encapsulates the human-like capacity of the model to interpret and reason about the image to furnish responses to the blind questions. Hence, ViCE reflects the way a human mind functions when comprehending and evaluating visual content.

After obtaining the initial set of answers from the blind questions and having a clear understanding of the presence of the required visual concepts, the model (or human evaluator) has to make a decision D if to request additional information to make some aspects clearer,

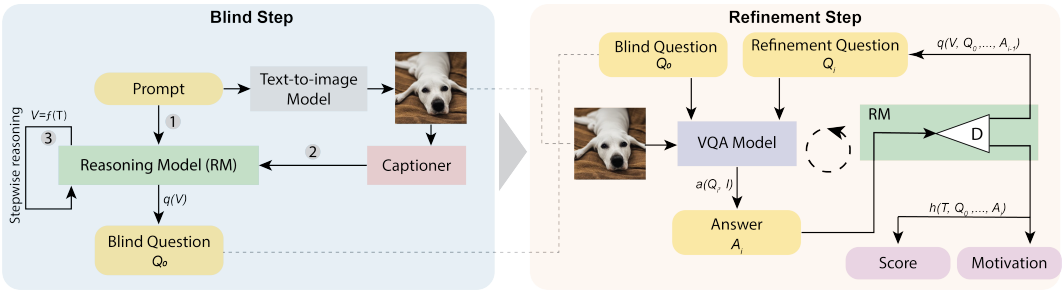


Figure 4.2: Visual Concept Evaluation Pipeline

or to close the process and make the final evaluation. If more information is needed, the model creates a new set of questions, known as "refinement questions".

Such iterative process can be conducted indefinitely:

$$\begin{aligned} Q_i &= q'(V, Q_0, A_0, ..., Q_{i-1}, A_{i-1}) \\ A_i &= a(I, Q_i) \end{aligned} \tag{4.4}$$

Finally, the evaluation score is computed, using all the questions, answers, and initial visual concepts:

$$E = h(T, Q_0, A_0, Q_1, A_1, ..., Q_i, A_i) \tag{4.5}$$

This recurrent process mirrors the human strategy for assessing a generated image, with each phase performing a crucial function in the overall evaluation.

4.2.4 Implementation

In our implementation, we establish a pipeline that integrates various models, each having a specific function corresponding to the steps delineated in the previously mentioned equations 4.1, 4.2, 4.3, 4.5. The objective is to construct an autonomous system capable of evaluating synthetic generated images.

We have integrated a Large Language Model, specifically the GPT-3.5-turbo [84], for the reasoning process. We refer to this agent as the "Reasoning Model". This choice was driven by our intent to simulate human-like reasoning, which is inherently stepwise, a characteristic LLMs readily adapt to [127]. Stepwise reasoning consists in first asking

the model what it expects to find in an image generated with that prompt and on what criteria it should evaluate the effectiveness of the generation. Only thereafter the actual questions are generated. As part of our future work, we plan to conduct a comparative study using different-sized LLMs and different stepwise reasoning approaches to verify and measure any impact on the results. To aid the model's reasoning process, we supplemented it with an image caption. Our analysis indicated that particularly for images that deviated from the expected generation, providing an image description significantly improved the model's capability to pose pertinent questions in the subsequent stages.

The question-generation phase unfolds in several steps. Initially, we set the model to generate a fixed number of questions (N=15 in our experiments) based on the image prompt and the expected visual concepts. Emulating the human evaluation process, the Reasoning Model may seek additional information to refine its understanding of the image. Therefore, as illustrated in Fig 4.2, the model is queried after the initial response phase about whether it requires further information. This triggers a refinement cycle featuring an iterative exchange of questions and answers until the model is satisfied with its comprehension of the image.

The questions span across semantic and qualitative aspects of the image, examining the presence or absence of objects described in the prompt, their interrelations, and qualitative characteristics. It is noteworthy how our Reasoning Model, in its pursuit to mirror human cognition, transcends the mere 'words' in the prompt. It comprehends the necessity to validate whether the objects are in the correct semantic relationship. For instance, in response to the prompt 'a vase of flowers', the model not only confirms the presence of the vase and the flowers but also verifies that the flowers are indeed in the vase, the vase is positioned on a surface, and the setting is congruent.

The responsibility of visual image analysis and generation of answers is vested in the Visual Question Answering (VQA) model. For our implementation, we utilized the BLIP2 model [62], built from the `salesforce-lavis` library [61]. As with the LLM, we intend to investigate the influence of the VQA model on the final output in our future endeavors. The capabilities of the VQA model are crucial as they enable the Reasoning Model to construct a detailed image schema that informs the subsequent question cycles and, ultimately, the final evaluation.

4.2.5 Experiments

Our experimental setup focused on evaluating images that were generated from textual prompts. The key objective was to determine the extent of alignment between the evaluation scores procured from the Visual Concept Evaluation (ViCE) model and those rendered by human evaluators.

In the beginning, we used the Stable Diffusion 2 model to generate images, utilizing prompts extracted equally from a variety of datasets [31, 23, 100, 65] for a total number of 1000 images. The task given to the external evaluators was to assess the level of consistency between the prompt and the generated image by scoring on a scale from 0 to 10.

We compared the evaluation scores from our ViCE model with automated metrics such as CLIPScore [43] and BLIP-ITC/ITM [62], along with other model-based evaluation techniques like LLMScore. CLIPScore and BLIP-ITC measure the distance between the embedding of the generated image and the embedding of the prompt. BLIP ITM has an additional network submodule that outputs a probability of matching.

4.2.5.1 Comparison with Human Evaluation

We conducted a comparative study between the scores derived from human evaluations and the calculated metrics. This comparison was accomplished using two correlation coefficients: Spearman's rank correlation coefficient and Pearson's correlation coefficient; additionally, we also used the Bland-Altman plot to illustrate the agreement between human and model-derived scores. More in detail, we employed:

- Spearman's Rank Correlation Coefficient: This non-parametric measure assesses the strength and direction of the relationship between two ranked variables. As it is less sensitive to outliers and does not assume a linear relationship, it is ideal for comparing ordinal variables.

- Pearson's Correlation Coefficient: This measure evaluates the linear correlation between two continuous variables.

- Bland-Altman Plot: This graphical method measures the agreement between two different ways of measuring a variable (in our case, human and model-derived evalua-

Model	Pearson	Spearman
CLIPscore	0.19467	0.17452
BLIP ITM	0.19404	0.18752
BLIP ITC	0.26943	0.25421
LLMScore	0.29264	**0.34065**
ViCE_5	0.25221	0.24981
ViCE_blind	0.27547	0.28325
ViCE	**0.33249**	0.32762

Table 4.1: Comparison of Evaluation Models. All metrics report p-value lower than 0.05, indicating statistically significant correlations. ViCE_5 applies the same pipeline with 5 questions and without refinements questions; ViCE_blind only uses the blind questions, without refinement.

tions). The plot showcases the difference between the two measurements against their average.

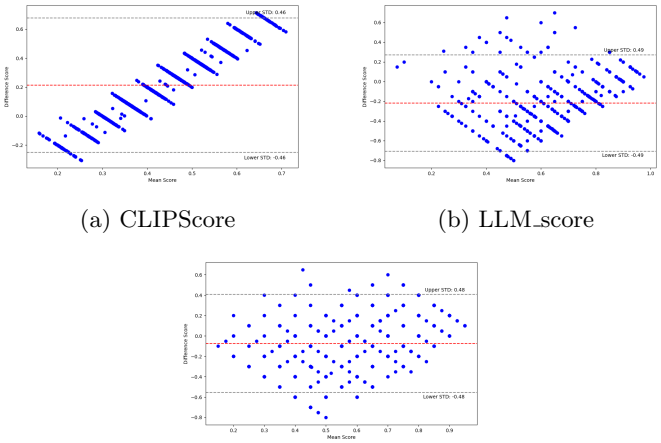
4.2.5.2 Results

The results presented in Table 4.1 reflect the evaluation carried out across several datasets, thus providing an overall score that accounts for different domains across these datasets. Notably, both LLMScore and ViCE significantly surpass all other automated metrics. An interesting observation is that while LLMScore performs better in terms of Spearman correlation, ViCE excels in Pearson correlation.

This outcome warrants a brief exploration. Spearman correlation evaluates the monotonic relationship between the two datasets, while Pearson correlation assesses the linear relationship. Therefore, ViCE's superiority in Pearson correlation might suggest a better linear relationship with the human scores.

Moving forward, our goal is to further refine ViCE by introducing an initial caption similar to the strategy employed by LLMScore. We envision that incorporating local and global descriptors, drawing from the methodology of GRIT [125], could improve the effectiveness of ViCE.

Additionally, in Table 4.1, we include the results from the ViCE model with only 5 initial questions ('ViCE_5') and without the refinement questions ('ViCE_blind'). Our hypothesis, which is supported by these results, suggests that reducing the number of questions or completely removing the refinement process prevents the model from effectively reason and



(a) CLIPScore (b) LLM_score

(c) ViCE

Figure 4.3: Bland-Altman plots for different automated metrics.

use the visual feedback, two elements that even humans leverage during evaluation.

In Figure 4.3 we report the Bland-Altman graphs [10], an established method to visualize the differences between two measurement techniques. In our setup, it provides a visual representation of the agreement between a standard reference measure (i.e. the human evaluation) and an automated metric of interest (i.e. one amongst CLIPScore, LLM_score, and our proposed ViCE), while simultaneously exposing any potential biases in the assessment.

It can be observed how CLIPScore's data fluctuates in a narrow range. Conversely, LLM_score tends to assign higher scores than human assessments, a fact that indicates a potential overestimation of image quality. On the other hand, ViCE shows a balanced distribution, indicating a closer alignment with human evaluations and suggesting it can indeed offer a more reliable method for automatic evaluation.

4.2.6 Extension to ITE

Expanding on our prior discussions, we suggest that the Visual Concept Evaluation (ViCE) approach is not confined to image generation but can be extended to Image Targeted Editing (ITE). In the ITE task, the input comprises both an image and a descriptive prompt, with the latter containing instructions for the desired semantic changes to be applied to the image. Such modifications, rather than being stylistic adjustments, involve content alterations that

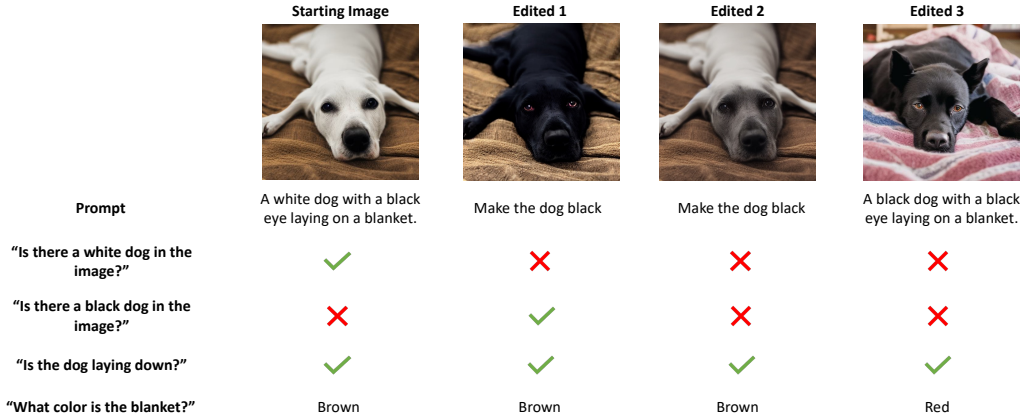


Figure 4.4: ViCE applied to the ITE task, whereby an LLM generates context-specific queries to assess the quality of the edit. The variation in the generated responses offers insights about the effectiveness of the edit operations.

touch upon only a section of the original image.

In the (recent) past, these models required an explicit mask to pinpoint the image section to be modified [79, 3]. However, the currently available large Vision-Language models are now capable of autonomously identifying the region for modification [54, 134, 11].

Still, the evaluation of such a task requires human evaluators to identify which parts of the image should remain untouched and which should be altered, and then to evaluate the precision of the implemented changes.

In this context, visual concepts can be divided into three distinct sets:

1. V_{remain}: Visual concepts that should be kept from the original image;
2. V_{remove}: Visual concepts that should no longer be present;
3. V_{add}: Visual concepts that should be added to the output image.

Thus, the set of visual concepts V to be checked for in the edited image compounds to:

$$V = V_{\text{remain}} - V_{\text{remove}} + V_{\text{add}} \tag{4.6}$$

Through the reasoning process, questions related to the visual concepts that will be modified can be formulated, and the responses can subsequently be used to evaluate the effectiveness of the modification.

Visual concepts belonging to V_{remain} are expected to stay constant, and any change in the responses associated with these concepts would suggest that the portion of the image meant to be preserved has been altered. An illustrative example of this scenario can be found in Figure 4.4.

4.2.7 Conclusions

This work marks an initial step towards mirroring human reasoning when it comes to synthetic image evaluation. We have devised an approach that acknowledges both explicit and implicit facets of human cognition, creating a close alignment with human judgment.

This bold venture aims to narrow the cognitive gap between AI and humans, thereby advancing towards a more nuanced and reliable image evaluation methodology.

4.3 Discussion and Limitations

ViCE was developed at a time when both language models and vision–language systems were substantially less capable than current MLLMs. The pipeline therefore enforced a form of stepwise reasoning "from the outside": instead of relying on built-in chain-of-thought behaviour, it orchestrated separate question generation and visual question answering modules to approximate how a human evaluator would decompose a prompt into visual concepts and interrogate a generated image. In perspective, this makes ViCE an early instance of the broader paradigm of reasoning-based LLMs that has since become common across generative models and evaluators.

From a thesis-level perspective, the main strength of ViCE is to show that human-aligned evaluation can be achieved by explicitly modelling the process that leads to a judgment, rather than only regressing to final scores. The framework demonstrates that prompting a language model to articulate what should be present in an image, and then to ask and answer targeted questions about those expectations, yields metrics that better track human assessments of instruction-level faithfulness than traditional distributional or alignment-based scores. At the same time, its design exposes a number of weaknesses related to data, computation and model behaviour.

A first limitation lies in the supervision available for the reasoning process itself. The

paper evaluates ViCE primarily through the correlation between its final scores and human ratings, without a dedicated dataset to assess the quality and coverage of the generated questions. As a result, it is difficult to disentangle whether failures are due to missing or poorly phrased questions, to imperfect visual reasoning from the VQA backbone, or to aggregation choices in the final scoring step. Subsequent work and future extensions could benefit from explicit benchmarks for question quality and completeness, for instance by collecting human-written question sets or expert annotations against which LLM-generated questions can be compared.

Secondly, ViCE is constrained by the capabilities and design choices of the underlying language and vision models. At the time of its development, LLMs had more limited reasoning abilities and MLLMs were not yet widely available, so the pipeline combines a text-only LLM with a separate VQA model and explores only a small number of ablations over model families and question counts. Modern multimodal architectures, which jointly process text and images and support richer reasoning patterns, could change both the optimal design of the pipeline and its absolute performance, suggesting that the results in this chapter should be read as a lower bound on what the Visual Concept Evaluation idea can achieve with contemporary tools.

The framework also comes with computational and practical constraints. Because ViCE relies on iterative question generation and answering, evaluation is significantly more expensive than computing a single embedding similarity or classifier score, and the reasoning loop is only partially parallelisable: later questions depend on earlier answers, which limits throughput when scoring large numbers of images.

Finally, the current formulation focuses on semantic faithfulness and does not fully capture more subjective aspects of human preference. ViCE is designed to answer whether an image satisfies the explicit and implicit constraints of a prompt, rather than how pleasing or aesthetically valuable that image is.

Extending the framework towards preference-aware evaluation, for instance by conditioning on user profiles or by studying how aesthetic judgments change over time, across geographic regions or application domains, would open up richer, but also more challenging, forms of human alignment. In parallel, integrating ViCE with complementary signals from other models or detectors, and exploring variants that operate earlier in the generative

pipeline (for example on low-resolution drafts before final upsampling, or in conjunction with hallucination early detection as in Chapter 3), would help turn question-based evaluation into a more efficient and flexible component of the broader evaluation-centric lifecycle developed in this thesis.

Chapter 5

Differential Evaluation of Localised Image Edits

5.1 Chapter Overview

In the evaluation-centric lifecycle of Figure 1.1, this chapter occupies the final stage of the pipeline, where an already generated image is refined through one or more instruction-guided edits. At this point, the question is no longer whether the original generation is globally faithful to the prompt, but whether the editing model has applied the *right* change in the *right* place while preserving everything that should remain untouched. Standard metrics for image editing tend to collapse this problem into a single similarity score between the edited image and either the prompt or a reference, which makes it difficult to disentangle correct edits from unwanted alterations and to explain why a particular model fails.

The DICE framework addresses this gap by unifying the thesis's three pillars: it achieves **Granularity** by isolating object-level edits, ensures **Explainability** by producing textual descriptions of changes, and maintains **Sustainability** through efficient adapter-based training. Conceptually, DICE plays the role of a *disentanglement evaluator* at the end of the lifecycle: it receives the pre- and post-edit images and returns a decomposition of the outcome into coherent object-level modifications. Rather than producing a single scalar metric by design, it outputs localised judgments that can be aggregated into benchmark statistics,

enabling both fine-grained error analysis and model-level comparisons.

At the core of the chapter is a two-stage pipeline that deliberately separates *what changed* from *whether the change was appropriate*. In the first stage, a difference detector identifies object-level discrepancies between the original and edited images, predicting the type of modification (addition, removal, or edit), its location, and a short textual description. Crucially, this detector is trained to operate independently of the editing instruction, so that it remains agnostic to what *should* have happened and can faithfully surface both requested changes and unintended side effects. In the second stage, a coherence estimator revisits each detected difference in the context of the prompt and decides whether that modification is consistent with the user's request, providing an explicit rationale alongside a binary coherence label.

Both stages are implemented on top of the same autoregressive MLLM, re-used with two lightweight fine-tuning adapters. This design makes the framework practical to deploy: the visual encoder and backbone are loaded once, and only small, task-specific adapters are swapped between difference detection and coherence estimation. Training follows a progressive strategy that first teaches the model to compare real image pairs drawn from object detection datasets, then bridges the gap to edited images using synthetic inpainting, and finally distils supervision and rationales from more powerful models and human annotations. On top of this, the chapter introduces the DICE-D dataset for object-level difference and coherence annotation in instruction-guided editing, which serves as the testbed for both detection quality and downstream evaluation.

Empirically, the results show that DICE can reliably localise object-level edits, achieve competitive detection performance compared to open-vocabulary detectors and alternative MLLMs, and distinguish coherent from non-coherent modifications in a way that aligns well with human judgments. Aggregating its local decisions over large collections of edited images yields an editing benchmark that separates correct edits, unwanted changes, and cases where models fail to modify the image at all, enabling a nuanced comparison of state-of-the-art editing methods. Moreover, by using DICE to selectively mask coherent or non-coherent regions, the chapter demonstrates that existing CLIP-based metrics for prompt adherence and background preservation become more correlated with human ratings, illustrating how disentanglement signals can strengthen other evaluation tools.

5.2 *What Changed?* Detecting and Evaluating Instruction-Guided Image Edits with Multimodal Large Language Models

5.2.1 Introduction

The field of Generative AI has recently gained significant attention in both industry and research, driven by the development of models capable of generating content that closely follows the distribution of natural language [27, 39, 84] and real images [98, 91, 28]. Thanks to the availability of cross-modal operators and architectures, multimodal generative approaches, which can seamlessly integrate both the vision and language modalities, are reaching increasing levels of accuracy [13, 67, 129, 120]. These advancements are driving the development of new applications and tasks while enhancing user control over the generation process.

Among these proposals, instruction-based image editing models [11, 133, 36, 49] enable the modification of an input image based on a free-form textual instruction from the user. This extends traditional text-to-image models, shifting the focus from generating an image from scratch to modifying an existing one while adhering to a given prompt. While such models promise to offer increased personalization levels, the understanding of the results they generate (and thus their proper evaluation) clearly becomes more challenging than in traditional text-to-image contexts.

Prior work has approached the evaluation of image editing models by developing annotated benchmarks with ground-truth edited images [133, 109, 110] or by proposing metrics based on pre-trained backbones such as CLIP [92] and DINO [16]. Other methods leverage pre-trained large language models to assess the quality of image edits [77, 51, 8]. While these proposals might offer a first viable solution to assess the performance of image editing models, they still do not reach a sufficient alignment with human evaluation. Further, being based on either global embedding vectors or replies from pre-trained (and often private) models, explaining their evaluations becomes cumbersome.

Drawing inspiration from these considerations, we take a novel approach and propose a learnable model for understanding localized image edits, which can provide increased align-

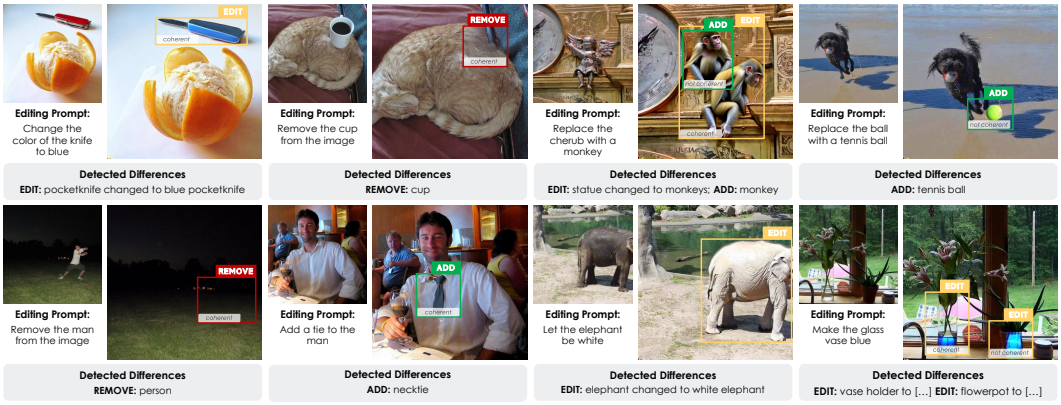


Figure 5.1: Qualitative examples from DICE. Our approach detects differences between an original image and an edited one, identifying the involved objects and the type of edit. Further, DICE evaluates each difference to determine its coherence with the editing prompt.

ment with human preferences and more easily explainable results. Although instruction-guided image editing encompasses a broad range of possible transformations, this work focuses on object-level edits, motivated by the observation that precise, localized modifications remain a significant challenge for generative models and thus require specialized evaluation protocols. Accordingly, our model first identifies object-level differences between the original and the edited images and subsequently evaluates the coherence of each detected modification relative to the user's input instruction. We term our pipeline as **DICE**, short for **DI**fference **C**oherence **E**stimator. Sample results from our model are shown in Fig. 5.1.

Maintaining the goal of creating a fully explainable pipeline, we develop two distinct models, one for each stage, both built on an autoregressive MLLM [13]. While both models leverage the understanding of regions of interest (RoIs), the first primarily focuses on their prediction, whereas the second emphasizes their interpretation to assess coherence with the given prompt. In particular, we first propose a difference detector that, given both images, estimates a set of localized differences that describe the change in content from one image to the other. The model also predicts the category of detected modifications, as either addition, removal, or editing of an object. A second model, based on the same architecture, is instead trained to predict the coherence of an object-level modification with respect to the modification request made by the user and provide a textual rationale for each decision made.

We experimentally assess the appropriateness of our strategy by carefully investigating

the performance of both our difference detector and our coherence estimator, in comparison with existing solutions, and when employing different MLLMs. Further, we conduct a dedicated user study to measure the alignment of the evaluations predicted by our proposal in terms of model ranking and evaluation. Experimental results demonstrate that our proposal achieves increased ranking capabilities while benefiting from an explainable-by-default approach. Further, when integrated into existing metrics, it acts as the basis for building model evaluation metrics with increased human alignment.

5.2.2 Related Work

Image Editing Models. Recent developments in GANs [38, 52] and diffusion models [98, 91, 28, 26] have driven the rise of AI-generated content. However, users increasingly demand not only the creation of new visual data but also the modification of existing images, altering properties such as style [52, 35] and content [11, 32]. Within this domain, tasks vary based on the input used for conditioning. Image inpainting [116, 22], for instance, uses user-designed masks to guide edits in specific regions. In contrast, our work focuses on instruction-based image editing, where the model relies solely on textual instructions and the original image.

Within instruction-based editing, traditional approaches rely on diffusion models fine-tuned for instruction following. A fundamental contribution in this domain is InstructPix2Pix [11], which is trained on fully synthetic triplets composed of an input image, an editing instruction, and the edited image. However, since reliance on synthetic data may introduce noise, MagicBrush [133] refines the InstructPix2Pix framework using a high-quality, curated dataset. Differently, InstructDiffusion [36] is trained to perform a wide variety of tasks, including image editing and grounding. Further, HIVE [134] proposes a reward-based training that incorporates human feedback to enhance its instruction following capabilities. Differently, MGIE [32] employs a frozen MLLM to generate concise edit instructions that guide the diffusion model during image editing, whereas SmartEdit [50] captures the interactions between image and text via MLLM, improving prompt understanding.

Image Editing Evaluation and Benchmarks. Instruction-based image editing models have traditionally been evaluated along two dimensions: prompt adherence and background preservation. The former assesses the extent to which the output aligns with the given

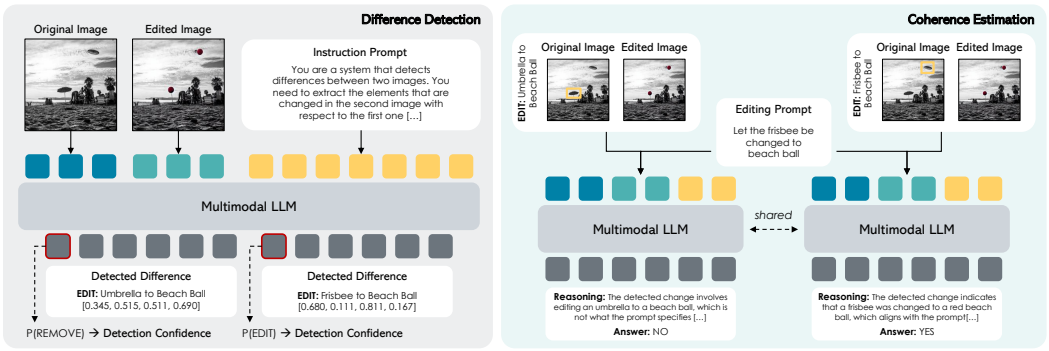


Figure 5.2: Illustration of DICE. We employ an MLLM and fine-tune it for two different tasks. In the first stage (difference detection), the MLLM is trained to detect semantic differences between the original image and the edited one. In the second stage (coherence estimation), the MLLM is instructed to analyze and assess the coherence of each detected difference with respect to the given user prompt.

prompt instruction, whereas the latter evaluates the preservation of the unmodified contextual information. While benchmarks featuring annotated ground-truth edited images are available [133, 109, 110], our work prioritizes a reference-free evaluation that eliminates the need for human-defined labels. In this setting, background preservation is typically evaluated by using either CLIP [92] or DINO [16] to compute cosine similarities between the visual tokens extracted from the source and edited images [11, 134, 32]. Likewise, prompt adherence is usually quantified using CLIP-based similarities between the target caption and the edited image [108, 133].

With the recent proposal of MLLMs, these methods have been employed for prompt adherence evaluation. For instance, in HQ-Edit [51], GPT-4V [84] is used to assess coherence and alignment metrics. Additionally, I^2E Bench [77] incorporates human annotation for certain queries, subsequently leveraging GPT-4V for question answering. However, reliance on GPT-based evaluations introduces potential variability and reproducibility issues, as model updates may alter the final scores. In contrast, we propose an interpretable solution based on an open-source MLLM, thereby increasing both the rationale behind our scoring mechanism and the reproducibility of the results.

Multimodal Large Language Models. With the introduction of LLM, the research community has put great effort into connecting visual modalities with textual reasoning [67, 24, 12]. Building on these efforts, MLLMs have also been used to address visual grounding

tasks. Among these, Shikra [20] leverages a novel training paradigm to give the MLLM the capacity to reason with bounding boxes, making it able to precisely locate objects and relations in the image. Similarly, LISA [59] and GROUNDHOG [135] perform segmentation incorporating pixel-level representations for fine-grained visual understanding.

Compared to existing MLLMs for visual grounding, our model leverages multiple images to detect semantic differences, introducing a novel setting that requires an MLLM capable of multi-image understanding. In this context, Idefics3 [60] is able to comprehend multiple visual inputs interleaved with text, using a novel image encoding pipeline. Differently, Qwen [120] extends classical multimodal capabilities to video and multi-image understanding thanks to a novel encoding strategy and positional embedding technique for the visual inputs. Finally, mPLUG-Owl3 [129] introduces a novel attention-based strategy to efficiently encode long image sequences. Building on these multi-image models, we extend the capabilities of MLLMs by introducing a novel framework for multi-image detection, enabling the identification of semantic differences across multiple images within a unified system.

5.2.3 Proposed Method

5.2.3.1 Preliminaries

The objective of our approach is to assess the quality of outputs generated by instruction-based image editing models [11, 134, 32]. Formally, given an original image $\boldsymbol{x}$ and a textual instruction $\boldsymbol{t}$, an image editing method f generates a new image $\boldsymbol{e} = f(\boldsymbol{x}, \boldsymbol{t})$, which incorporates the modifications specified by the textual prompt $\boldsymbol{t}$, while preserving the contextual integrity of $\boldsymbol{x}$.

While the modifications introduced by image editing models can be diverse – owing to the expressiveness of textual instructions and the flexibility of generative models – we focus specifically on object-level modifications. By means of this choice, we consider transformations that are clearly localizable inside the input scene. In particular, we assume that the original image contains a finite set of objects, denoted as $\{o_i\}_{i=1}^{n}$, and that the textual instruction $\boldsymbol{t}$ specifies an operation to be applied to it. Given a correct application of the specified operation, the expected edited image, denoted as $\tilde{\boldsymbol{e}}$, serves as the ground-truth for evaluating the intended modification.

Object-level Modifications. Without loss of generality, we assume that the alterations required in the textual prompt can be classified as either the addition, removal, or modification of objects inside the input image. Specifically, we consider three categories of alterations, namely:

1. **Addition of an object.** A new object instance o_{n+1} is introduced into the scene, yielding a new image representation. Treating images as a set of objects, and with a slight abuse of notation, this operation results in a new image that can be expressed as $\tilde{e} = (\bigcup_{i=1}^{n} o_i) \cup \{o_{n+1}\}$.

2. **Removal of an object.** The operation removes an object o_j which was present in the original image, which results in $\tilde{e} = \left(\bigcup_{i=1}^{j-1} o_i \cup \bigcup_{i=j+1}^{n} o_i\right)$.

3. **Edit of an object.** An existing object o_j is altered in at least one visual aspect. Let $\tilde{o}_j$ denote the modified object; the updated image is then represented as $\tilde{e} = (\bigcup_{i=1}^{n} o_i \setminus \{o_j\}) \cup \{\tilde{o}_j\}$.

5.2.3.2 Overview of the Approach

An image editing network can produce an image $\boldsymbol{e}$ that does not accurately respect all the object-level operations required to produce $\tilde{e}$. To quantify this discrepancy, we propose **DI**fference **C**oherence **E**stimator (**DICE**): a two-step approach in which (i) we detect the object-level differences between the edited and the original image and then (ii) we evaluate the coherence of each difference with the textual prompt to determine whether a discrepancy reflects the intended modifications or constitutes an unwanted alteration. An overview of the two stages of DICE is depicted in Fig. 5.2.

Difference Detection. In this stage, we extract the set of objects that are present in the original image $\boldsymbol{x}$ but absent or changed in the predicted image $\boldsymbol{e}$ or vice versa. We denote this collection of differences as *semantic difference*, which is formally defined as

$$\Delta(\boldsymbol{x}, \boldsymbol{e}) = \{o \mid o \in \{\boldsymbol{x} \setminus \boldsymbol{e}\} \cup \{\boldsymbol{e} \setminus \boldsymbol{x}\}\}. \tag{5.1}$$

The aforementioned set of differences is estimated through a learnable difference detector $D(\boldsymbol{x}, \boldsymbol{e})$ that, given the original and modified image, predicts a set of object-level differences

which serves as an approximation of $\Delta(\boldsymbol{x}, \boldsymbol{e})$, expressed in terms of RoIs. Notably, this detection process is executed independently of the textual prompt $\boldsymbol{t}$ to ensure that the outcome is not biased by the intended edit.

Coherence Estimation. In the second step, we evaluate each detected difference individually to assess its coherence with the textual prompt. Specifically, a coherence estimator operates on each detected object-level change $o_i \in D(\boldsymbol{x}, \boldsymbol{e})$, determining whether the visual modifications align with the intended edit $\tilde{\boldsymbol{e}}$. Formally,

$$C(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{t}, o_i) = \begin{cases} 1 & \text{if } o_i \in \tilde{\boldsymbol{e}}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

This process enables the differentiation between intended and unintended edits, thereby facilitating a comprehensive and interpretable evaluation of the task.

5.2.3.3 Detecting Differences

To build the difference detector and the coherence estimator, we design two novel architectures building upon the general structure of autoregressive MLLMs, which offer a sufficiently general framework for encoding images, prompts, and RoIs.

For difference detection, we train an MLLM $D(\mathbf{x}, \mathbf{e})$ to predict object-level differences between two images in terms of (i) the type of modification (i.e., addition, removal, edit), (ii) the RoI of the modified region, and (iii) the subjects involved in the modification. Formally, the difference detector operates as:

$$D(\boldsymbol{x}, \boldsymbol{e}) = [(c_i, S_i, b_i)]_{i=1}^{k}, \tag{5.3}$$

where c_i denotes a command from $\{\texttt{ADD}, \texttt{EDIT}, \texttt{REMOVE}\}$, S_i describes the modified object in free text, and $b_i \in \mathbb{R}^4$ defines the bounding box coordinates in the format $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$.

Each tuple (c_i, S_i, b_i) is fully specified in the textual domain, and the model is optimized to output structured text representations of modifications, formatted as: `"COMMAND: object, [bb1, bb2, bb3, bb4]"`. This structured output ensures precise localization and identification of changes.

Custom System Prompt:

```
You are a system that detects differences between two images.

- Extract the elements that are changed in the second image with respect to the
first one.
- Create a new entry for each distinct change.
- For each entry, use the following format:
"<CHANGE_COMMAND>:  <CHANGED_ELEMENT>, (<BOUNDING_BOX>)"

CHANGE_COMMAND:
- ADD: If a new element appears in the second image that was not present in the
first.
- REMOVE: If an element from the first image is missing in the second.
- EDIT: If an element in the second image is different but in the same location as
an element in the first image.

CHANGED_ELEMENT: Describe the element that has changed.

BOUNDING_BOX: Use normalized coordinates [x0, y0, x1, y1] for the changed element
position in the second image, where (x0, y0) is the top-left corner, and (x1, y1)
is the bottom-right corner.  The coordinates should be scaled between 0 and 1,
with 0 representing one edge of the image and 1 representing the opposite edge.
```

Table 5.1: Prompting template for the difference detection stage of the DICE pipeline.

Prompt Format. We prompt the difference detector with a fixed system instruction that enforces structured, per-difference outputs and normalized bounding boxes. The full prompt template used in our experiments is reported in Tab. 5.1.

To estimate the confidence of a predicted RoI, we compute the probability of the predicted command c_i relative to the total probability mass of all possible commands. This confidence measure aligns with conventional object detection confidence estimation techniques [15, 37].

Training Procedure Overview. Our training pipeline consists of two phases: (i) a pre-training stage leveraging real image pairs and (ii) a fine-tuning stage using images modified with inpainting models. Notably, the model is intentionally not trained on instruction-based paired edited images due to two key considerations. First, such training may lead to overfitting to specific editing architectures while underperforming on unseen methods. Second, instruction-edited image pairs would necessitate extensive human annotations for commands and RoIs, introducing significant cost and time constraints.

Stage 1: Learning to Compare Similar Images. In this stage, we instruct the difference detector to identify object-level differences when considering pairs of similar real images. Specifically, the MLLM is trained to predict objects that are not mutually present in both images, leveraging an existing object detection dataset.

In detail, we employ images and object-level annotations from the LVIS dataset [40]. We identify pairs of similar images by computing the cosine similarity in the DINOv2 [86] embedding space, and retaining only those with a cosine similarity greater than 0.6, ensuring high visual similarity. In addition, we make sure that the aforementioned pairs contain at least one common object class and differ by fewer than 15 object classes. This filtering results in a total of 118k image pairs. When encoding image pairs in this stage, objects present in the first image but absent in the second are labeled as `REMOVE`, while objects appearing in the second image but not in the first are labeled as `ADD`. Objects exhibiting an intersection-over-union above a predefined threshold across both images are classified as `EDIT`. Since LVIS spans 1,723 categories, annotation inconsistencies may cause spurious differences (e.g., missed objects, or the same object labeled under a different category). To reduce noisy supervision, we verify candidate missing objects with the open-vocabulary detector OWL-ViT [81], which checks whether objects annotated in one image (but not in its paired image) are genuinely absent. Further, we remove overly small annotations (at least one side < 16 pixels) to avoid unreliable RoIs. The resulting stage-1 dataset contains 118k image pairs with 795k, 725k, and 94k `ADD`, `REMOVE`, and `EDIT` operations, respectively.

Stage 2: Learning to Detect Inpainted Areas. In the second stage, we refine the model by bridging the gap between pre-training on real image pairs and application to edited images. To accomplish this, we select images from the LVIS dataset and sample non-overlapping annotated objects. Starting from LVIS images, we employ inpainting to generate original and edited image pairs. For each object, an operation is randomly selected among $\{\texttt{ADD}, \texttt{EDIT}, \texttt{REMOVE}\}$. Inpainting is then performed using LaMa [116]: for the added objects, the corresponding region is inpainted on the original image, while for removed objects, the inpaint is performed on the edited one. For the edit operation, 30% of the edited objects undergo a color change, while the remaining 70% are substituted with a different object. We rely on GPT-4o to generate target objects suitable for substitution by prompting it with a caption of the original image as well as the original object. To perform the edits, the Kandinsky 2.2 inpaint model [96] is employed to inpaint selected regions. This process leads to a training set of 97k elements and a test set of 19k.

To ensure data quality, we sample at most 4 objects per image, filter out masks covering less than 3% of the image area, and constrain the maximum overlap between selected masks

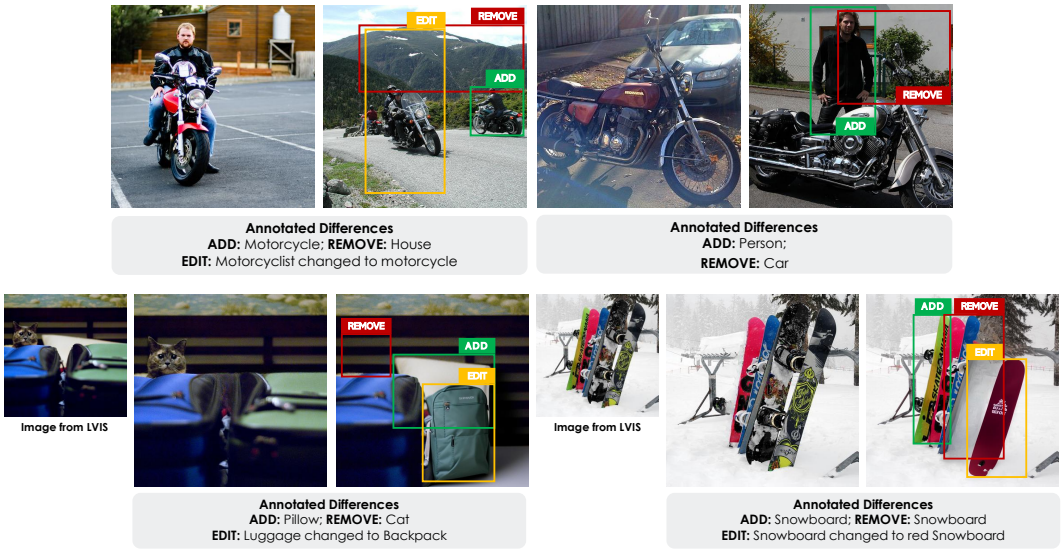


Figure 5.3: Qualitative examples from the difference detection training datasets: the first row shows samples from the dataset used in stage 1, and the second row from the dataset employed in stage 2.

to 5%. Kandinsky 2.2 is run with 100 diffusion steps and a guidance scale of 4. To avoid unbalanced predictions, we sample the three operations uniformly and include a subset of unchanged pairs in both splits. Qualitative examples from stage 1 and stage 2 are shown in Fig. 5.3.

5.2.3.4 Estimating the Coherence

Given the original and edited image, along with a detected difference, the coherence estimator produces a binary decision (i.e., Yes/No) and a textual rationale indicating whether the modification is coherent with the intended edit. To induce the model to focus on a specific difference, the bounding box of the detected difference is visually depicted on the original and edited image before they are fed to the coherence estimator. Further, we also include the type of modification c_i and its textual description S_i as input to the model.

Prompt Format. Similarly to the detector, the coherence estimator is driven by a fixed template that specifies a strict matching criterion and requires a short rationale alongside the binary decision. The prompt template is reported in Tab. 5.2.

To train the model, we annotated 116 samples from the EmuEdit [108] test set, with

Custom System Prompt:

```
You are evaluating if a specific change detected by an AI vision model
matches the request in the original edit prompt.

## Task
Determine if the detected change, as described and bounded by the provided
colored bbox, matches the request in the original edit prompt.
A match is valid only if the localized detected change is 100% compatible
with the requested prompt.
Any unwanted modification of the original image (even small) should avoid
a match.

## Context
- The original image and the edited image are provided, in this order.  The
edited image is
the original with some changes applied.  Focus only on the area specified
by the bbox in the detected change.
- Another AI model has detected a change in the image, including its bbox.
- ADD: An object is only added in the edited image (on the background).
- EDIT: An object is substituted with another one in the edited image.
- REMOVE: An object is removed in the edited image.
- Be strict:  An EDIT means that an object has been removed and substituted
with another one,
ensure nothing was removed unless explicitly stated in the prompt.  If an
object has been removed unexpectedly, then you should say NO.

## Example Response
- Reasoning:  <REASONING>
- Decision:  "YES" or "NO"
```

User Prompt:

```
## Instructions
1.  The original edit prompt is:  {SUBSTITUTE_PROMPT}
2.  The detected change to evaluate is:  {SUBSTITUTE_CHANGE}
3.  Use only the text and the observations from the specified bbox area
(colored) in both the
original and edited images to decide if the specific detected change
aligns with the original edit prompt.
```

Table 5.2: Prompting template for the coherence estimation stage of the DICE pipeline.

the corresponding edited images generated using InstructDiffusion [36]. For each sample, we manually annotated both the observed differences and a binary coherence indicator. In addition, we employed GPT-4o to generate a rationale for each annotation, which was subsequently refined through human review to eliminate any inaccuracies. While the command c and the subjects S were provided in textual format, the bounding boxes were directly depicted on the images using a color-coded scheme: red for additions, green for edits, and blue for removals. Specifically, additions are superimposed on the edited image, whereas the bounding boxes corresponding to removals and edits are applied solely to the original image.

	Confidence	Training		Class-agnostic Detection					Class-aware Detection							
		Stage 1	Stage 2	AP	AP_{50}	AP_{75}	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_M	AP_L	AP_{ADD}	AP_{REM}	AP_{EDIT}
Grounding Models																
OWLv2 [80]	✓	-	-	16.4	25.2	17.3	12.4	21.4	9.8	14.0	10.7	9.1	10.6	12.3	7.1	9.8
Grounding DINO [70]	✓	-	-	12.8	17.6	13.4	9.3	15.2	6.8	8.6	7.4	6.1	7.8	4.8	8.6	6.9
Alternative MLLMs																
GlaMM [94]	-	-	-	5.0	8.3	4.8	2.0	6.3	2.7	4.1	2.8	1.1	3.5	3.9	1.3	2.9
GPT-4 [84]	-	-	-	0.5	1.9	0.2	0.0	0.8	0.3	1.1	0.1	0	0.5	1.7	1.9	0.3
Qwen2-VL-7B [120]	-	✓	✓	2.0	5.3	1.3	0.3	3.4	1.6	3.3	1.2	0.1	2.4	0.1	0.1	3.2
Qwen2-VL-7B [120]	✓	✓	✓	2.3	5.6	1.8	0.1	3.5	1.6	4.1	1.1	0	1.6	0.1	0.1	3.2
mPLUG-Owl3-7B [129]	-	✓	✓	3.8	10.3	2.4	0.5	5.1	2.5	6.7	1.7	0.3	3.3	1.1	1.5	5.1
mPLUG-Owl3-7B [129]	✓	✓	✓	5.2	13.7	3.2	0.5	7.3	4.4	10.6	3.1	0.3	5.5	0.1	7.3	4.8
Ablation Studies																
Idefics3-8B [60]	-	-	✓	15.1	30.1	13.4	8.5	18.6	9.4	18.1	9.3	5.5	11.0	7.3	7.5	13.4
Idefics3-8B [60]	-	✓	✓	16.7	30.7	16.0	10.4	20.3	10.6	17.9	10.9	6.6	12.5	7.4	7.7	16.5
Idefics3-8B [60]	$✓_{rand}$	✓	✓	16.8	31.2	16.2	10.3	20.5	9.7	16.5	10.0	6.7	11.2	7.0	7.9	14.2
Idefics3-8B [60]	✓	✓	-	2.9	5.1	2.8	1.9	4.2	1.4	2.2	1.4	0.9	1.8	2.0	0.4	1.8
Idefics3-8B [60]	✓	-	✓	18.2	36.8	16.0	9.8	22.7	12.2	22.2	12.1	7.3	15.3	7.2	13.4	15.8
DICE (Ours)																
Idefics3-8B [60]	✓	✓	✓	**22.3**	**40.1**	**21.9**	**13.5**	**36.8**	**15.5**	**24.8**	**16.5**	**11.4**	**15.4**	**19.6**	**14.8**	**16.4**

Table 5.3: Performance comparison of various MLLMs in the difference detection stage of our pipeline, evaluated under both class-agnostic and class-aware settings. Results are presented in terms of AP metrics across various training configurations.

5.2.4 Experimental Results

5.2.4.1 Implementation and Training Details

In our experiments, we consider three recent MLLMs designed to support multi-image input tasks: Idefics3-8B [60], Qwen2-VL-7B [120], and mPLUG-Owl3-7B [129]. Following the training steps described in Sec. 5.2.3.3 and Sec. 5.2.3.4, we fine-tune each model using QLoRA [25]. Specifically, the vision encoder and projector of each model remain frozen, while the language model is quantized and augmented with LoRA [46] adapters. LoRA is applied to the self-attention and MLP layers across all 32 Transformer blocks. In all our experiments, we apply QLoRA with rank 64, scaling factor 16, and dropout 0.1, using 4-bit quantization and the paged AdamW 8-bit optimizer [71].

For image encoding, images are always center-cropped based on the smaller dimension to maintain a square aspect ratio. When employing Idefics3, images are internally resized to 1,456 pixels and encoded in a list of 364-pixel crops, generating a grid of 16 crops separately encoded. For Qwen2-VL, the image is encoded using a bidimensional positional embedding and image tokens are compressed by an MLP layer that encodes 2×2 adjacent tokens into a single one, keeping the input image resolution at 1,456. For mPLUG-Owl3, the images are directly encoded through a SigLIP400m-384 [131].

Difference Detection. All stages of difference detection training use 8 NVIDIA A100

	Confidence	Coherence over GT Areas	Coherence over Detected Areas		
		Accuracy	AP	AP_{50}	AP_{75}
Alternative MLLMs					
Qwen2-VL-7B [120]	-	76.6	1.4	3.8	0.6
Qwen2-VL-7B [120]	✓	76.6	1.8	4.9	0.9
mPLUG-Owl3-7B [129]	-	85.9	3.8	10.0	2.4
mPLUG-Owl3-7B [129]	✓	85.9	4.5	11.3	2.9
DICE (Ours)					
Idefics3-8B [60]	-	85.4	12.0	20.3	12.0
Idefics3-8B [60]	✓	85.4	**15.5**	**26.0**	**16.1**

Table 5.4: Performance comparison of various MLLMs in the coherence estimation stage of our pipeline. Coherence accuracy is measured using ground-truth differences, while coherence over detected areas employs the output of the first stage of our pipeline and is evaluated using AP metrics.

64GB GPUs, with learning rate 1×10^{-4} and total batch size 8. In stage 1, Idefics converges after 9k steps with evaluations every 1k steps; in stage 2, convergence is reached after 30k steps with evaluations every 5k steps. During stage 2, to mitigate low-level artifacts introduced by inpainting models, we apply random JPEG compression between 15% and 50% magnitude during training, following established practice in deepfake detection [121]. At inference time, JPEG compression is fixed at 33%.

Coherence Estimation. Training is performed on a single NVIDIA A100 64GB GPU with batch size one and learning rate 1×10^{-5}. In this case, Idefics reaches convergence at 550 steps, with evaluations every 50 steps.

5.2.4.2 Evaluating Detected Differences

To evaluate the performance of our detection and coherence pipeline, we construct a new dataset, termed as DICE-D, specifically designed for the task. This dataset comprises 800 edited images extracted from the I^2EBench [77] benchmark. We select images and prompts from a subset of the edit categories that involve object-based modifications, in particular: color alteration, counting (only if involving a single object change), object removal, and object replacement. Each pair of images is manually annotated with detected differences represented as bounding boxes, editing commands (i.e., `ADD`, `REMOVE`, `EDIT`), and textual descriptions. Also, we assign a coherence label for each bounding box to indicate its

alignment with the prompt.

Difference Detection Results. We first evaluate the ability of the difference detection model to detect modifications between the original and edited images. Following object detection literature [65, 15], we compute the mean average precision (AP) between predicted and labeled bounding boxes from DICE-D. Specifically, we average results at 10 IoU thresholds from 0.5 to 0.95. Further, we include AP_{50} (average precision at 50% IoU), AP_{75} (average precision at 75% IoU), AP_{M} (for medium-sized objects), and AP_{L} (for large-sized objects), ensuring a comprehensive evaluation across different object scales and IoU thresholds.

Results are reported in Table 5.3, where we consider both class-agnostic detection and class-aware detection and compare DICE against other MLLMs and open-vocabulary detectors, OWLv2 [80] and Grounding DINO [70]. In class-agnostic detection, all predictions are treated as belonging to the same category, to evaluate only the bounding box location. Vice versa, in class-aware detection, both the bounding box location and the editing command are considered, evaluating the accuracy of both the detection and the associated command.

Difference Detection with Open-Vocabulary Detectors. Standard object detection models like OWLv2 and Grounding DINO operate on single images independently and therefore must be adapted to perform difference detection. To enable this, we apply object detection separately to the original and edited images (using all LVIS classes as labels), and then compare detections across images. Given a detection in the original image: if a detection in the edited image has an IoU greater than 0.5 and a different label, it is classified as an `EDIT` difference; if no detection in the edited image exceeds the IoU threshold, it is considered a `REMOVE` difference. Conversely, if a detection in the edited image has no corresponding detection in the original image above the threshold, it is labeled as an `ADD` difference. Finally, if two detections with the same label have an IoU over the threshold, then no difference is found. The same conversion procedure can be applied to any single-image detector when used as a difference baseline. We also consider GLaMM [94] as a zero-shot single-image detector baseline, and apply the same conversion procedure above to extract `ADD`/`REMOVE`/`EDIT` differences from its detections.

As shown, using Idefics as the base model consistently achieves the best performance across all metrics, with an increase of 17.1 and 20.0 in AP in a class-agnostic setting compared to Qwen and mPLUG. Similarly, when considering the command labels, Idefics outperforms

Qwen and mPLUG by 11.8 and 13.9. This superior performance may be attributed to the distinctive image encoding pipeline employed by Idefics, in which each image is resized to a larger dimension and segmented into a grid of crops, each encoded independently. Although this approach increases the amount of visual information processed, the grid-based encoding simultaneously introduces a localization factor to the captured visual details that can be leveraged in a task of detection. Further, the inferior performance of OWLv2 [80] and Grounding DINO [70] underscores that difference detection is fundamentally distinct from standard object detection.

When validating the contribution of the key components of our pipeline, it is worth noting that the initial pre-training stage significantly contributes to the final performance. Specifically, training Idefics in both stages results in improvements of 1.6 and 1.2 in class-agnostic and class-aware AP, respectively, compared to training solely on the second stage dataset. Notably, even training exclusively on the first stage enhances class-agnostic detection compared to Qwen (i.e., +0.4 AP). This observation implies that predicting missing objects in pairs of similar real images not only facilitates the learning of prediction syntax but also enables the transfer of relevant knowledge to the difference detection task in edited images.

Moreover, incorporating the extracted confidence score enhances difference localization results, with improvements of 5.5 and 3.1 AP observed for Idefics-based models trained on both stages and solely on the second stage respectively, compared to using a fixed confidence value of 1. Additionally, using random confidence values results in worse performance, further indicating that the uncertainty in the command correlates with the actual prediction confidence.

Coherence Estimation Results. The coherence estimation model assesses whether each detected difference is correctly aligned with the prompt using a binary prediction (i.e., Yes/No). Moreover, the coherence step incorporates explicit model reasoning [124], improving the interpretability of the results. Under this setting, models are evaluated on DICE-D, in terms of coherence over ground-truth areas and coherence over detected areas. To measure the coherence accuracy, ground-truth differences are fed into the coherence estimator, and results are evaluated against the manually annotated coherence. Instead, for the coherence over detected areas, the performance of DICE is measured through AP, considering the

Editing Models	Correct Edits		Unwanted Edits		No Visual Change	
	DICE (%) ↑	Humans (1-5) ↑	DICE (%) ↓	Humans (1-5) ↑	DICE (%) ↓	Humans (%) ↓
HIVE [134]	11.0	2.0	40.5	3.1	54.5	40.5
InstructPix2Pix [11]	15.5	2.3	32.1	3.3	44.0	22.0
MGIE [32]	23.0	2.7	21.6	3.8	11.5	10.0
MagicBrush [133]	24.5	2.9	23.1	3.9	8.5	5.5
InstructDiffusion [36]	30.0	3.0	19.1	4.0	13.5	10.5

Table 5.5: Benchmark comparison of model rankings generated by DICE and those derived from the user study. The first two columns contrast average human ratings with scores obtained using DICE. The final column compares the percentage of unchanged images in the user study – cases with maximal background preservation and minimal prompt adherence – to the corresponding one identified by DICE.

coherence label of the difference as the ground-truth category of the bounding box. This is done by extracting differences with the detector and subsequently predicting the coherence of each difference.

Results are summarized in Table 5.4, evaluating various MLLMs as coherence estimators. Idefics consistently achieves the highest overall performance on AP, surpassing mPLUG and Qwen by 11.0 and 12.7 points, respectively, while maintaining comparable coherence accuracy. Additionally, incorporating confidence from the difference detection stage into the AP computation leads to performance improvements across all evaluated MLLMs. This highlights the importance of confidence extraction in the difference detection phase and motivates further research on aligning token probability with confidence estimation in object detection. Based on these results, Idefics3-8B is selected as the base MLLM for DICE in all subsequent experiments.

5.2.4.3 Evaluation of Image Editing Models

User Study Details. We conduct a user study to evaluate our proposed framework within the context of instruction-based image editing. In detail, we collect 200 prompts along with their corresponding original images, 50% from MagicBrush [133] and 50% from I^2EBench [77]. For MagicBrush, we use GPT-4o to filter out prompts that do not involve object modification. Differently, for the I^2EBench subset, we employ the same selection criteria as in the construction of DICE-D (cf. Sec. 5.2.4.2). We select five state-of-the-art generators and generate 200 edited images per generator. Specifically, our model selection includes HIVE [134], InstructPix2Pix [11], MGIE [32], MagicBrush [133], and InstructDif-

fusion [36]. To assess the quality of the generated edits, we asked 10 human annotators to evaluate each sample on two dimensions: *prompt adherence* and *background preservation*. Prompt adherence measures how well the modifications in the edited image align with the instructions given in the prompt. Differently, background preservation assesses how well the elements that were not intended to be modified remained unchanged.

Data Generation. The editing models used for user study data generation are set according to their default configuration. In particular, for InstructDiffusion [36] the image guidance scale is set to 1.25 and the text guidance scale is set to 5, while for all the other models we use 1.5 and 7 respectively. For HIVE [134] and InstructDiffusion the number of inference steps is 100, while for all the other models we set the inference steps to 20.

Rating Protocol. Participants rated prompt adherence on a 1–5 Likert scale: *not applied at all*, *slightly applied*, *moderately applied*, *well applied*, and *perfectly applied*. Likewise, they evaluated background preservation on a 1–5 scale: *changed completely*, *moderately altered*, *mostly preserved*, *nearly fully preserved*, and *preserved completely*. The study was conducted through an interactive interface that allowed participants to compare the original and the edited image while reading the prompt (Fig. 5.4).

Editing Model Ranking. To evaluate the performance of editing models, we establish three axes to rank the considered generators using DICE. *Correct edits* measure the percentage of images in which at least one detected difference is classified as coherent. Conversely, *unwanted edits* quantify the percentage of the total image area containing differences marked as non-coherent. Lastly, *no visual change* represents the percentage of images where DICE does not detect any visual alteration, potentially due to insufficient prompt clarity, lack of understanding, or inability of the model to execute the requested modification. Notably, *correct* and *unwanted edits* can be measured with the average prompt adherence and background preservation ratings from the user study. Similarly, from the user study, *no visual change* is assessed by identifying images rated 5 in background preservation and 1 in prompt following.

In Table 5.5, we compare the rankings generated by our evaluation with those obtained from the user study. As it can be seen, model rankings from DICE align with human evaluations. In particular, InstructDiffusion and MagicBrush emerge as the top-performing models

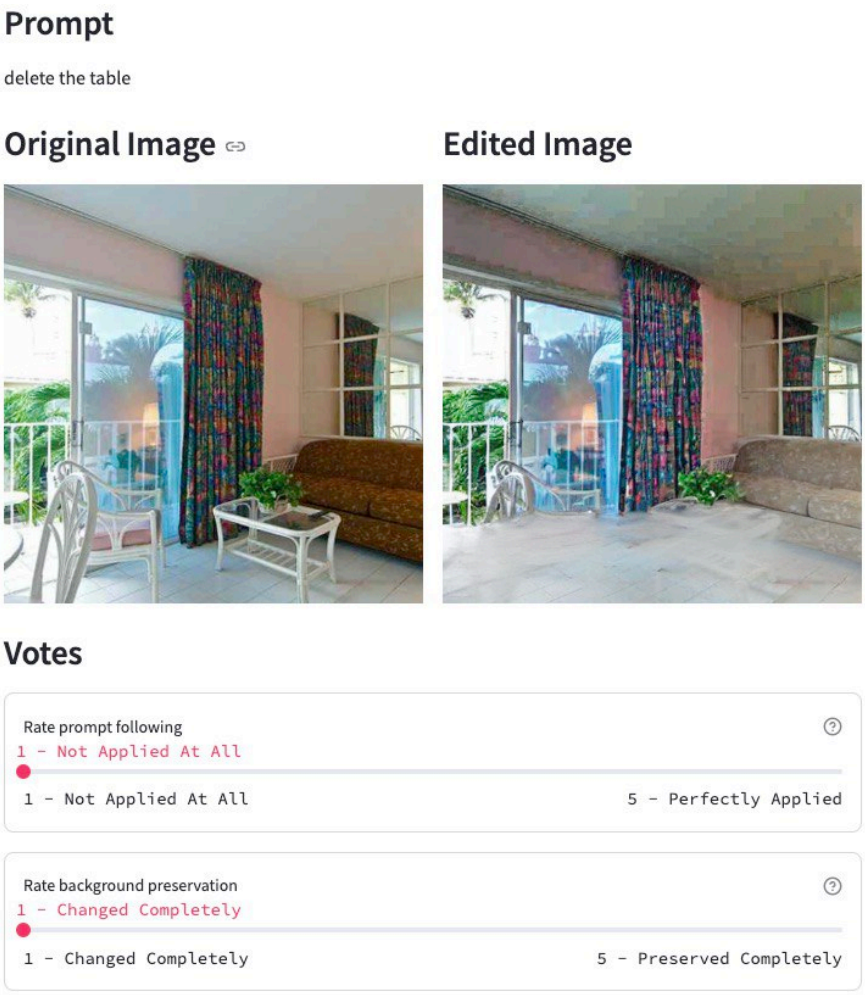


Figure 5.4: User study interface displaying the original and the edited image alongside the editing prompt.

both according to DICE and human ratings, excelling in performing correct modifications while effectively preserving the background. Likewise, the ranking order for *no visual change* remains consistent across DICE and the user study, further validating the reliability of our approach.

To qualitatively validate the predictions of DICE, Fig. 5.5 presents editing samples showcasing difference detection and coherence predictions across various prompts and models. Notably, DICE effectively identifies image differences and evaluates their coherence with the editing prompt.

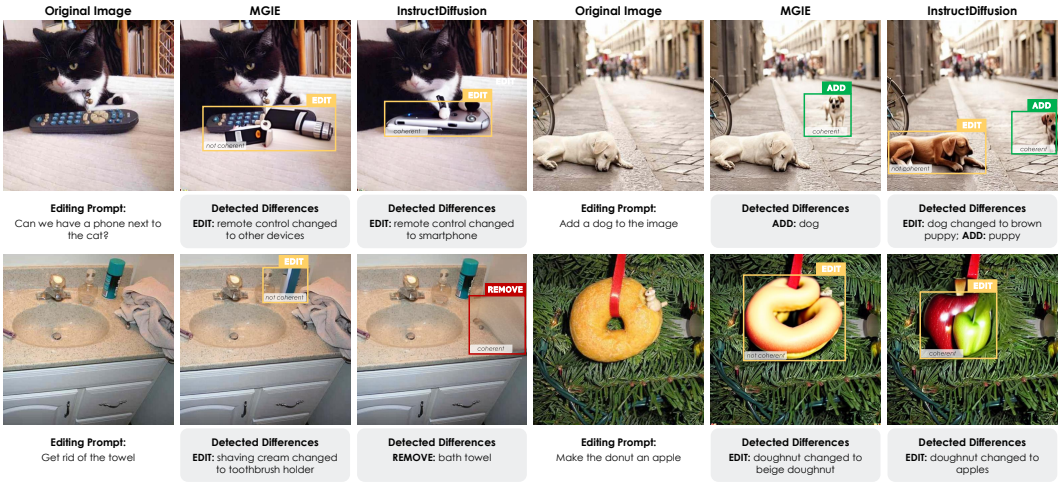


Figure 5.5: Qualitative samples of DICE applied on images edited by MGIE [32] and InstructDiffusion [36] models.

5.2.4.4 Additional Qualitative Results

Fig. 5.6 illustrates a wide range of successful edits handled by DICE across diverse scenarios, featuring highly precise bounding boxes around the modified elements. In several examples, the system accurately detects color changes (e.g., altering the color of a bird or an umbrella) and clearly distinguishes added objects, such as a watermelon or a necktie. The pipeline also robustly identifies removed elements, whether it is a coffee table or a giraffe, and captures more subtle edit operations, for instance replacing a bird with a boy or a cat with a dog. These qualitative results highlight how DICE maintains spatial accuracy and class awareness when performing additions, removals, and edits, thereby demonstrating its adaptability to different editing requests.

Additionally, the coherence evaluator provides clear, step-by-step reasoning for each detected change, explaining why it is coherent or not. For example, as shown in Fig. 5.7, in the "change the color of the rose to blue" prompt, the system points out that "cyan" is still a valid shade of blue and correctly labels the edit as coherent. Likewise, when asked to "add a basketball to the top of the car", it confirms not only that the basketball has been introduced but also that it is located precisely where the prompt requires – on top of the car. These reasoned explanations highlight the transparency of our pipeline, helping users understand which aspects of the prompt were fulfilled and why certain edits might fall short

	ρ	ρ_s	τ
Background Preservation			
CLIP-I	51.1	42.9	33.3
w/ patch on random areas	34.2	25.6	19.4
w/ patch on all detected differences	32.7	15.6	11.6
w/ **DICE** (patch on coherent differences)	**54.5**	**45.4**	**35.1**
Prompt Adherence			
CLIP-T	21.5	23.1	17.1
w/ patch on random areas	17.8	19.5	14.3
w/ patch on all detected differences	13.3	14.6	10.7
w/ **DICE** (patch on non-coherent differences)	**24.6**	**26.6**	**19.6**

Table 5.6: Correlation analysis showing improved alignment with human judgment when integrating DICE in CLIP-I and CLIP-T.

of the requested modifications.

Correlation with Human Judgment. To further validate the alignment between the detected differences and human judgment, we integrate DICE predictions within CLIP-based evaluation metrics measuring image similarity (CLIP-I) for background preservation and image-text similarity (CLIP-T) for prompt adherence [43]. In particular, CLIP-I measures the similarity between the original and the edited image. To compute CLIP-T, instead, we caption the original image using an MLLM (i.e., Idefics3-8B in our experiments) and feed the extracted caption along with the editing prompt to GPT-4o to produce a caption of the ideal edited image (i.e., target caption). Through CLIP-T we then measure the CLIP similarity between the target caption and the actual edited image. To integrate DICE in CLIP-I and CLIP-T, we modify both the original and edited images by selectively masking specific regions, taking into account DICE predictions. Specifically, for background preservation, we mask coherent differences, whereas, for prompt adherence assessment, non-coherent differences are patched.

For this experiment, we compare the correlation scores of standard CLIP-I and CLIP-T metrics with those obtained by masking original and edited images using DICE. Results are shown in Table 5.6, where we measure Spearman's ρ, Pearson's ρ_s, and Kendall's τ correlation coefficients between automated evaluation scores and human judgment. We also compare when randomly masking areas from the differences detected by DICE, or when masking all detected differences, including both coherent and non-coherent changes.

As it can be seen, DICE effectively enhances the correlation of both CLIP-I and CLIP-T metrics with human ratings. Specifically, for background preservation, CLIP-I exhibits the strongest correlation with human judgments when masking only coherent differences, achieving a Pearson's score of 54.5. This is a significant improvement over the baseline score of 51.1 (no masking), and substantially higher than the scores obtained with random masking or masking all detected differences. This result suggests that excluding correctly modified regions allows CLIP-I to better focus on the actual background, producing a metric that more accurately aligns with human perception.

For prompt adherence, CLIP-T shows better correlation when we patch non-coherent differences, achieving a Pearson's score of 24.6, compared to 21.5 without masking. Instead, masking all differences leads to a drop to 13.3, likely due to indiscriminate removal of both erroneous and correct edits. This outlines that selectively removing only incorrect modifications helps CLIP-T focus on prompt-relevant changes, thus improving alignment with human evaluations.

Overall, these results demonstrate that DICE enhances the reliability of CLIP-based metrics, making them more aligned with human perception of editing quality. In particular, distinguishing between coherent and non-coherent changes is crucial for obtaining meaningful evaluations.

5.2.4.5 Ablation Study on Localization

In this section, we build on the experiments in Tab 5.6 and further motivate the relevance of the localized output of DICE. Indeed, localization allows the evaluator model to focus on relevant regions and enhances interpretability. To prove this, we employ Gemma3-27B [118] and Qwen2.5-VL-32B [4] in a fully zero-shot setting, measuring correlation with human ratings with and without DICE detections in the prompt. For Prompt Adherence, we provide bounding boxes of differences labeled as "coherent"; for Background Preservation, we use boxes predicted as "non-coherent". These boxes are overlaid on the image, and their corresponding textual descriptions are appended to the prompt, following the strategy of our coherence evaluator. As shown in Tab. 5.7, adding localization significantly boosts correlation with human evaluations. This confirms that grounding detected objects enables MLLMs to better interpret and reason about visual edits. Notably, the substantial gains in

	Gemma3-27B			Qwen2.5-VL-32B		
	ρ	ρ_s	τ	ρ	ρ_s	τ
Background Preservation						
w/o DICE detections	13.9	12.2	10.5	20.1	20.3	17.5
w/ DICE detections	**36.2**	**36.3**	**32.8**	**37.3**	**35.9**	**31.5**
Prompt Adherence						
w/o DICE detections	31.0	27.0	23.0	67.7	70.2	**59.8**
w/ DICE detections	**32.8**	**34.3**	**30.3**	**67.9**	**70.6**	59.5

Table 5.7: Impact of localization on human correlations.

Background Preservation suggest that MLLMs evaluators may be biased toward verifying whether the instructed change occurred, while neglecting unintended modifications. Our method explicitly identifies such unintended changes during the detection phase, leading to more robust evaluations.

5.2.4.6 Failure Cases

We report representative failure cases in Fig. 5.8, where detection noise can propagate to coherence judgments and strict matching criteria may occasionally reject otherwise acceptable edits.

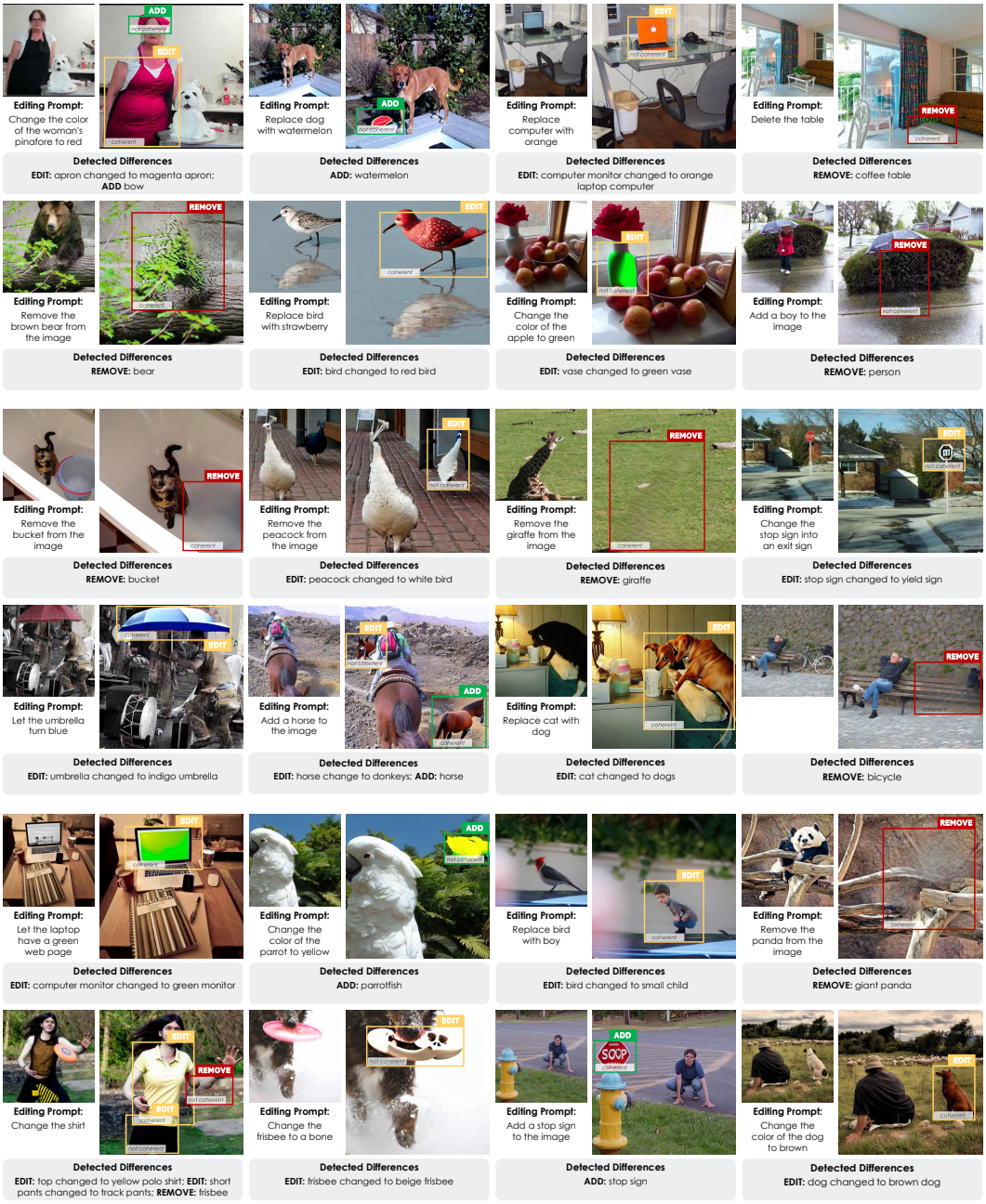


Figure 5.6: Additional qualitative results. Each instruction-based edit shows the original image (left) and the edited version (right), alongside the given prompt.

Figure 5.7: Examples illustrating the reasoning of the coherence evaluator that justifies its 'YES' or 'NO' decisions. Each box pairs a detected difference with an explanation of why the edit either fulfills or fails the user's request, highlighting the ability of the pipeline to handle both spatial and semantic context.

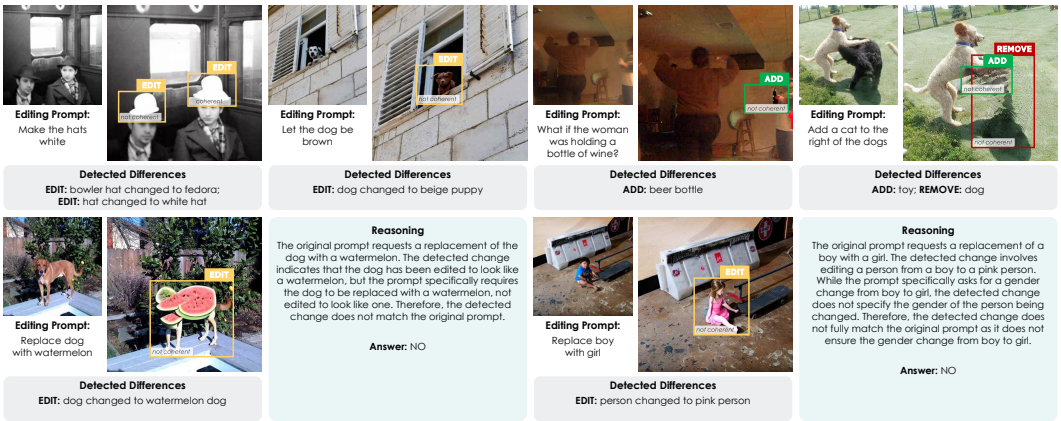


Figure 5.8: Failure cases where detection errors may impact coherence evaluation, potentially leading to misclassifications. Inaccurate identification of edits can introduce ambiguity for the coherence evaluator, while strict coherence criteria might occasionally reject valid changes, highlighting the interdependence of both stages.

5.3 Discussion and Limitations

From the perspective of this thesis, DICE exemplifies how evaluation can be made both local and structurally informative in the most complex part of the image lifecycle: instruction-guided editing. By explicitly decomposing editing outcomes into object-level differences and coherence judgments, it moves beyond global similarity scores and makes it possible to ask *what* was changed, *where* it happened, and *whether* those changes were actually requested.

Conceptually, DICE frames edit-level evaluation as a disentanglement problem, separating intended object-level operations from spurious or excessive changes across pairs of images. The proposed pipeline shows that a single multimodal backbone, combined with two fine-tuned heads for difference detection and coherence estimation, can produce these structured signals at scale and with strong alignment to human annotations. In doing so, it supports multiple roles: providing direct, interpretable judgments on individual edits, defining aggregate statistics that rank editing models along axes such as "correct edits", "unwanted edits", and "no visual change", and acting as a plug-in module that increases the reliability of other metrics when used to mask coherent or incoherent regions. Taken together, these elements advance the thesis goal of transforming evaluation from a passive scoring procedure into an active, lifecycle-wide analysis of model behaviour.

Beyond its role within DICE, the difference detection head is also a generally useful component. A prompt-agnostic change detector between original and edited images can, for example, support automatic quality control in editing pipelines, assist in analysing sequences of similar images captured at different time frames (e.g., for monitoring progressive edits or environmental changes), and help curate before/after datasets for training or evaluation.

At the same time, the current instantiation of DICE comes with some limitations. A first limitation concerns the *scope of edits* it can reliably handle. The framework deliberately focuses on object-level modifications that can be localised in space and categorised as additions, removals, or edits of discrete entities. This makes the problem well-defined and enables clear supervision, but leaves out richer classes of edits such as global style changes, fine-grained colour adjustments, geometric transformations, or subtle attribute shifts that do not correspond to the appearance or disappearance of a single object. While some of these phenomena are touched upon in the training procedure (for example, through colour

changes in synthetic inpainting), the main evaluation protocol and the DICE-D dataset are centred on object-level operations, so generalisation to more diffuse or stylistic edits remains an open question.

A second limitation is that DICE analyses only the *differences that actually materialise* between the original and edited images. The pipeline is sensitive to spurious edits, namely unwanted objects or changes that were not requested by the prompt, and is able to flag them as non-coherent, but it does not explicitly reason about *missing changes*. If an editing model fails to perform a requested modification and leaves the image unchanged, DICE will detect no differences and therefore cannot distinguish between a correct "no-op" instruction and an unfulfilled request. Extending the framework to extract candidate operations directly from the textual instruction and to cross-check whether they are realised in the visual domain would allow it to capture this complementary failure mode, and would bring it closer to a full audit of the editing process.

The multi-stage nature of the pipeline also introduces opportunities for error accumulation. Both stages are built on top of a single MLLM backbone: if the difference detector fails to localise a modified region, the coherence estimator never receives it as input, and its judgments can only be as reliable as the underlying detections. This dependence is particularly salient when editing models produce low-quality or highly distorted content, where object boundaries become ambiguous and both detection and coherence estimation are more difficult. In practice, this issue is somewhat mitigated by the increasing visual quality of modern editing models, and by the choice to train the detector without conditioning on the prompt, which reduces bias towards only the expected changes and encourages the discovery of genuinely unexpected differences. Nonetheless, the evaluation remains most trustworthy in regimes where objects are sufficiently well-formed to be detected, and future work will need to explore more robust detectors and uncertainty-aware aggregation schemes.

From a computational perspective, DICE sits between lightweight embedding-based metrics and heavyweight LLM judges. By reusing a single MLLM architecture with two small adapters, it avoids loading distinct large models for detection and coherence estimation, but each image pair still requires one forward pass through the detector and at least one pass per detected difference for coherence, including the generation of textual rationales. This makes inference more expensive than computing CLIP similarities or simple classifier scores, even

though it remains significantly cheaper than querying proprietary, large-scale multimodal models for every sample. Exploring smaller or distilled backbones, as well as strategies for pruning low-confidence differences early in the pipeline, represents a natural direction for making such disentanglement-based evaluation more accessible in large-scale or real-time settings.

Pushing disentanglement-based evaluation beyond object-level edits, improving its robustness and efficiency, and integrating it with complementary supervision from users and other models are natural next steps towards an evaluation-centric lifecycle that is both practically deployable and theoretically grounded.

Chapter 6

Discussion and Conclusions

Revisiting the Thesis Journey

The central question driving this thesis has been how to evaluate generative systems in a way that is rigorous, scalable, and practically useful for the entire model lifecycle. We started from the observation that as generative models grow in capability and scale, traditional evaluation metrics, often reduced to single scalar values calculated post-hoc, are no longer sufficient. They fail to capture the nuance of user intent, they are computationally expensive to compute blindly on every sample, and they offer little actionable guidance on how to improve the model.

In response, we proposed an *evaluation-centric lifecycle* that integrates evaluation at three critical stages of the generative process. In Hallucination-Aware Control of Text-to-Image Generation, we introduced *in-loop guidance*, demonstrating that evaluation signals can be monitored during the denoising process to anticipate failures and save computation. In Human-Aligned Evaluation for Text-to-Image Models, we moved to *post-hoc verification*, developing a human-aligned framework that checks consistency through visual concept reasoning. Finally, in Differential Evaluation of Localised Image Edits, we addressed *edit-level disentanglement*, providing a mechanism to distinguish between intended edits and unwanted side effects.

Taken individually, these contributions offer specific solutions to the problems of hallu-

cination, faithfulness, and editing quality. However, taken together, they point towards a more fundamental shift in how we conceive the role of evaluation in generative systems.

6.1 From Linear Pipelines to Adaptive Cycles

The current landscape of Generative AI, and specifically of image and text generation products, is defined by a predominantly linear lifecycle. Foundation models are trained on massive, static datasets until convergence, frozen, and then deployed to users. In this paradigm, the flow of information is unidirectional: from the pre-training data to the final pixel output. Consequently, the resulting systems behave as **static artifacts**: powerful, general-purpose engines that possess immense knowledge but lack the ability to adapt to the specific nuances of an individual user's workflow. This linearity dictates the nature of current products. Because the model cannot adapt to the user, the user is forced to adapt to the model, iterating on wording, constraints, and examples to steer the output. When a generation fails to meet expectations, the feedback loop is broken; the user can only retry with a different query, but the model itself learns nothing from the interaction. Evaluation, in this context, plays a limited role as a final gatekeeper, computed once before deployment to benchmark performance against generic standards.

We argue that realizing the full potential of Generative AI requires breaking this linearity: rather than a one-shot pipeline, the system must operate as a **continuous, adaptive cycle**. The motivation for this shift is not only technical efficiency, but also a different relationship between people and models. Instead of a single, homogeneous system serving everyone in the same way, we envision models and agents that are genuinely tailored to the individuals who use them, adapting to their values, prior knowledge, working style, language, and cultural context. In this perspective, the aim of generative AI is not to flatten users into an average, but to empower their individual differences and support them in their own goals. Achieving this kind of subjective, personalised behaviour is challenging because the amount of explicit feedback any one person is willing to provide is very small: a handful of accepted generations, a few edits, or occasional corrections. Each of these signals must therefore be used as efficiently as possible, extracting from it every bit of information that can guide future behaviour. This in turn requires not only precise evaluation, but also

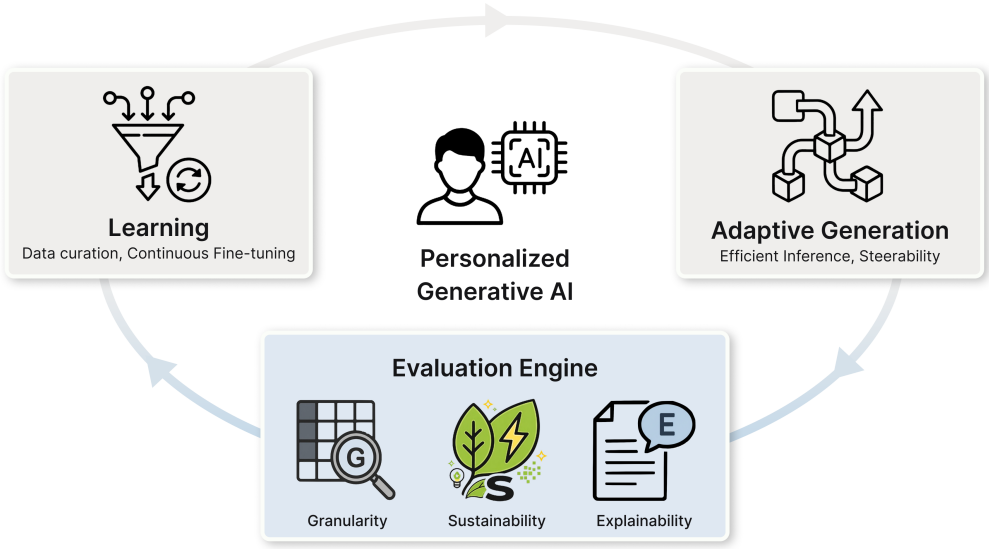


Figure 6.1: From a linear pipeline to a continuous, adaptive cycle for Personalized Generative AI. *Adaptive Generation*, *Evaluation*, and *Learning* are linked in a closed-loop refinement process, with the evaluation engine providing actionable signals for adaptation.

training and adaptation mechanisms that can reliably update large models from very small, high-value datasets, and do so frequently without incurring prohibitive computational or environmental costs. Taken together, these constraints motivate a continuous loop in which generation, evaluation, and learning constantly interact to turn sparse feedback into targeted model updates. At a high level, as shown in Figure 6.1, this cycle consists of three tightly coupled stages that follow one another in sequence: *Adaptive Generation*, *Evaluation*, and *Learning*. In the adaptive generation stage, the system decides how to respond to a request, for example by choosing whether to reuse an existing draft or regenerate from scratch, or by selectively activating specialized modules (such as safety filters or domain-specific adapters) on top of a shared backbone only when the task demands it. In the learning stage, the system assimilates information from past interactions and updates its internal state or parameters so that future generations better reflect the user's preferences and constraints. The evaluation stage mediates between these two, measuring, for each interaction, how well the system has aligned with the user's intent and which aspects of the behaviour should be reinforced or corrected, making each sparse feedback signal as informative as possible.

Transitioning to this cyclic paradigm requires concerted research efforts across the entire

generative stack. It demands adaptive generation strategies that can dynamically allocate compute and modelling capacity. Decisions such as when to engage richer customization modules on top of a shared base model, or when a full regeneration with more powerful reasoning or editing capabilities is needed, are only initial examples in a much broader design space that will require substantial exploration. It also requires advances in continuous learning to update models on the fly without catastrophic forgetting, extracting as much information as possible from relatively few, high-value examples. But crucially, this entire cycle relies on a robust evaluation engine to close the loop. Without reliable evaluation signals at the level of individual generations and edits, the feedback arrow from Evaluation to Learning remains too noisy to drive safe and efficient adaptation.

6.2 The Central Role of Evaluation

If the continuous cycle is the engine of the future agent, evaluation is its steering wheel. In the transition from linear pipelines to adaptive cycles, evaluation ceases to be a passive metric and becomes the active control mechanism of the system. It acts as the critical bridge that connects the *Generation* stage (inference) with the *Learning* stage (feedback), enabling automated data curation.

To make this loop work in practice, models must become far more efficient in how they extract information from data. Continuously training ever larger models on ever larger datasets is increasingly costly in terms of computation and energy, as discussed in the broader literature on the environmental and resource impacts of large-scale model training. In a cyclic setting, we therefore want each interaction to carry as much learning signal as possible. Evaluation is the component that turns raw interaction logs into structured feedback: it decides which examples are trustworthy enough to learn from, which ones reveal systematic failures, and how to translate successes and mistakes into positive and negative signals for the agent. Crucially, this feedback must also respect the diversity of users. The goal of personalization is not to homogenise behaviour around a single average preference, but to support each individual in their specific goals and constraints.

Based on the methodologies developed in this thesis, we identify three cross-cutting pillars, namely **Explainability**, **Granularity**, and **Sustainability**, that define the require-

ments for the evaluation engine of the future.

Explainability. In a self-correcting cycle, knowing *that* an error occurred is insufficient; the system must understand *why*. As demonstrated in ViCE, evaluation must provide a rationale, a chain of reasoning, rather than a simple score. This explainability is what transforms evaluation into a semantic gradient. Instead of just updating weights to minimize a generic loss, the model can leverage these rationales to adjust the prompt, steer the sampling trajectory, or fine-tune specifically on identified failure modes. Explainable evaluation becomes the language through which the user and the agent align their understanding.

Granularity. Personalization often lies in the details. Global metrics mask the specific nuances required for high-quality adaptation. Our work on DICE and HEaD+ highlights the need for evaluation to operate at high resolution, both spatially (on specific objects) and temporally (during specific generation steps). This granularity is essential for efficiency in the learning loop. If the system can identify that a generated image is largely correct but flawed in a specific region, it can perform targeted corrections ("surgery") on the content rather than discarding the entire sample. This allows for learning from partial successes, accelerating the personalization process.

Sustainability. Finally, in a continuous loop, evaluation runs as frequently as generation. It is unsustainable for the critic to be heavier than the creator. As shown in HEaD+, evaluation must be resource-aware, capable of acting as an agile supervisor. Only sustainable evaluation allows the loop to run in real time, potentially even on-device, preserving privacy and reducing latency.

This focus on sustainability and reproducibility is also aligned with practical constraints observed in an industrial setting. During my PhD internship at Typewise (a Swiss AI start-up backed by Y Combinator), evaluation protocols for text-based customer-care assistants had to be repeatable, efficient, and robust to model non-determinism in order to support reliable feature validation and safe model updates. These constraints mirror the core thesis message that evaluation, whether for text or images, must be designed as an actionable, resource-aware component of the deployment loop rather than an afterthought.

6.3 Future Research Directions

The transition from static evaluation to the adaptive cycle illustrated in Figure 6.1 opens a vast design space. While this thesis has established the core pillars of Explainability, Granularity, and Sustainability, several concrete research directions remain to fully realize the vision of a personalized, self-improving agent.

Semantic and contextual consistency beyond surface alignment. Current evaluation metrics check whether an image matches the prompt, but rarely investigate its semantic and contextual correctness. However, even when an image is well aligned with the literal text, it may still violate common-sense constraints (e.g. implausible combinations of objects and environments), domain conventions (e.g. incorrect anatomy or physics), or context-specific priors (e.g. historical inconsistencies between clothing and setting). A principled future direction is therefore to move from generic alignment metrics to *context-aware* evaluators that explicitly model which semantic violations matter in a given use case, and under what conditions they are acceptable (e.g. intentionally surreal prompts). This direction naturally extends the premise of ViCE: rather than producing a single score, the evaluator should articulate *which* constraint was broken and *why* it matters for the requested intent.

Hierarchical completeness: from explicit requirements to implicit expectations. A second direction concerns *completeness*, i.e. whether a generation contains all the information required to satisfy a user. We distinguish two complementary levels. First, the evaluator should verify the *explicit* requirements of the prompt by decomposing it into atomic claims (objects, attributes, relations, counts) and checking each claim individually, in the spirit of the granular reasoning used in ViCE and the edit-focused analysis in DICE. Second, evaluation should account for *implicit expectations*: users often assume unstated elements that are typical of the requested scene (e.g. a "train station" implicitly suggests platforms, tracks, signage, and possibly a train). Capturing these expectations requires modelling the user's prior knowledge and the domain context, and deciding which implicit elements are essential versus optional. This shifts evaluation from purely instruction-following verification to a broader notion of *task satisfaction* that better matches human judgement.

Personalised evaluation agents as user-specific priors. These implicit expectations are inherently user dependent. Two users may request the same concept while implicitly valuing different properties (e.g. a commuter focusing on functional elements versus an architect focusing on structural details). Future work should therefore explore *personalised evaluators* that learn user-specific priors over time from sparse interaction signals (accepted generations, edits, and corrections), and that can adapt their notion of "good" to an individual's workflow. In the cycle of Figure 6.1, this corresponds to making the evaluation engine itself adaptive: its checks, thresholds, and error taxonomies become part of the personalisation loop, enabling feedback that is not merely correct on average, but correct *for that user*.

Tool-augmented evaluation for grounded verification. Many consistency and completeness checks require knowledge that is not reliably stored in a static model, or that changes over time. This motivates *tool-augmented* evaluators that can selectively call external capabilities (e.g. retrieval modules, structured knowledge bases, domain-specific detectors, OCR, face recognition under appropriate governance) to validate claims against real-world information. The key research challenge is to decide *when* to invoke such tools and *how* to fuse their outputs into an explainable rationale.

From evaluation signals to self-correction and targeted refinement. If evaluation is the steering wheel, a natural next step is to connect it directly to the system's actuators. Rather than treating evaluation just as a final analysis, future systems should use structured feedback to perform targeted continuous corrections: revise prompts, trigger local re-generation of a specific region, or request a constrained edit that addresses a particular failure mode. This closed-loop refinement is not only a conceptual ideal: during my PhD internship at Typewise, in a text-only setting, feeding structured feedback back to the AI assistant to immediately revise its output proved highly promising, and the resulting improvement in reliability was noticeable to end users. In this setting, evaluation becomes an interface between perception and optimisation, enabling faster convergence towards user intent with fewer full re-generations.

Safety, guardrails, and governance as first-class evaluation objectives. Finally, evaluation-centric cycles must explicitly incorporate safety constraints. Beyond detecting generic harmful content, future evaluators should address privacy risks, identity-sensitive content, and the creation of persuasive synthetic media that could be used for manipulation. This requires robust auditing signals, calibrated uncertainty, and clear escalation protocols for human oversight. Importantly, as evaluation becomes a control mechanism, it also becomes a governance mechanism: it can enforce policies, document why content was blocked or modified, and provide transparent evidence for compliance in regulated settings.

6.4 Closing Remarks

This thesis began by tracing the journey of an image through a standard, linear generation pipeline. It concludes by suggesting that the journey does not end with the pixel output. By embedding rigorous, explainable, granular, and sustainable evaluation into the heart of the process, we enable the transition from static generators to self-improving, personalized agents. The methods presented here are steps towards a future where AI systems are not just powerful tools, but reliable and personalised partners in our creative and professional workflows.

List of Figures

List of Tables

Bibliography

[1] Aishwarya Agarwal et al. “A-STAR: Test-time Attention Segregation and Retention for Text-to-image Synthesis”. In: *ICCV*. 2023.

[2] Jean-Baptiste Alayrac et al. “Flamingo: a visual language model for few-shot learning”. In: *NeurIPS* 35 (2022), pp. 23716–23736.

[3] Omri Avrahami, Dani Lischinski, and Ohad Fried. “Blended diffusion for text-driven editing of natural images”. In: *CVPR*. 2022.

[4] Shuai Bai et al. “Qwen2.5-VL Technical Report”. In: *arXiv preprint arXiv:2502.13923* (2025).

[5] Eslam Mohamed Bakr et al. “HRS-Bench: Holistic, Reliable and Scalable Benchmark for Text-to-Image Models”. In: *ICCV*. 2023.

[6] Yogesh Balaji et al. “eDiff-I: Text-to-Image Diffusion Models with an Ensemble of Expert Denoisers”. In: *arXiv preprint arXiv:2211.01324* (2022).

[7] Satanjeev Banerjee and Alon Lavie. “METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments”. In: *ACL Workshops*. Ann Arbor, Michigan, June 2005.

[8] Federico Betti et al. “Let’s ViCE! Mimicking Human Cognitive Behavior in Image Generation Evaluation”. In: *ACM Multimedia*. 2023.

[9] Federico Betti et al. “Optimizing Resource Consumption in Diffusion Models through Hallucination Early Detection”. In: *ECCV Workshops*. 2024.

[10] J Martin Bland and DouglasG Altman. “Statistical methods for assessing agreement between two methods of clinical measurement”. In: *The lancet* 327.8476 (1986).

[11] Tim Brooks, Aleksander Holynski, and Alexei A Efros. "Instructpix2pix: Learning to follow image editing instructions". In: *CVPR*. 2023.

[12] Davide Bucciarelli et al. "Personalizing Multimodal Large Language Models for Image Captioning: An Experimental Analysis". In: *ECCV*. 2024.

[13] Davide Caffagni et al. "The Revolution of Multimodal Large Language Models: A Survey". In: 2024.

[14] Mingdeng Cao et al. "MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing". In: *ICCV*. 2023.

[15] Nicolas Carion et al. "End-to-End Object Detection with Transformers". In: *ECCV*. 2020.

[16] Mathilde Caron et al. "Emerging Properties in Self-Supervised Vision Transformers". In: *ICCV*. 2021.

[17] Hila Chefer et al. "Attend-and-Excite: Attention-Based Semantic Guidance for Text-to-Image Diffusion Models". In: *ACM Transactions on Graphics (TOG)* (2023).

[18] Haoxuan Chen et al. "Accelerating diffusion models with parallel sampling: Inference at sub-linear time complexity". In: *NeurIPS* 37 (2024).

[19] Junsong Chen et al. "PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis". In: *ICLR*. 2024.

[20] Keqin Chen et al. "Shikra: Unleashing Multimodal LLM's Referential Dialogue Magic". In: *arXiv preprint arXiv:2306.15195* (2023).

[21] Minghao Chen, Iro Laina, and Andrea Vedaldi. "Training-Free Layout Control with Cross-Attention Guidance". In: *WACV*. 2024.

[22] Mang Tik Chiu et al. "Brush2Prompt: Contextual Prompt Generator for Object Inpainting". In: *CVPR*. 2024.

[23] Jaemin Cho, Abhay Zala, and Mohit Bansal. "Dall-eval: Probing the reasoning skills and social biases of text-to-image generative transformers". In: *arXiv preprint arXiv:2202.04053* (2022).

[24] Federico Cocchi et al. “LLaVA-MORE: A Comparative Study of LLMs and Visual Backbones for Enhanced Visual Instruction Tuning”. In: *arXiv preprint arXiv:2503.15621* (2025).

[25] Tim Dettmers et al. “QLoRA: Efficient Finetuning of Quantized LLMs”. In: *NeurIPS*. 2023.

[26] Prafulla Dhariwal and Alexander Nichol. “Diffusion Models Beat GANs on Image Synthesis”. In: *NeurIPS*. 2021.

[27] Abhimanyu Dubey et al. “The Llama 3 Herd of Models”. In: *arXiv preprint arXiv:2407.21783* (2024).

[28] Patrick Esser et al. “Scaling Rectified Flow Transformers for High-Resolution Image Synthesis”. In: *ICML*. 2024.

[29] Bryan Faiz et al. “LLMCarbon: Modeling the End-to-End Carbon Footprint of Large Language Models”. In: *ICLR*. 2024.

[30] William Fedus, Barret Zoph, and Noam Shazeer. “Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity”. In: *Journal of Machine Learning Research* (2022).

[31] Weixi Feng et al. “Training-Free Structured Diffusion Guidance for Compositional Text-to-Image Synthesis”. In: *ICLR*. 2023.

[32] Tsu-Jui Fu et al. “Guiding Instruction-based Image Editing via Multimodal Large Language Models”. In: *ICLR*. 2024.

[33] Rinon Gal et al. *An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion*. 2022.

[34] Rinon Gal et al. “StyleGAN-NADA: CLIP-guided domain adaptation of image generators”. In: *ACM Transactions on Graphics (TOG)* (2022).

[35] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. “Image style transfer using convolutional neural networks”. In: *CVPR*. 2016.

[36] Zigang Geng et al. “InstructDiffusion: A Generalist Modeling Interface for Vision Tasks”. In: *CVPR*. 2024.

[37] Ross Girshick. “Fast R-CNN”. In: *ICCV*. 2015.

[38] Ian Goodfellow et al. “Generative Adversarial Networks”. In: *NeurIPS* (2014).

[39] Daya Guo et al. “DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning”. In: *arXiv preprint arXiv:2501.12948* (2025).

[40] Agrim Gupta, Piotr Dollar, and Ross Girshick. “LVIS: A Dataset for Large Vocabulary Instance Segmentation”. In: *CVPR*. 2019.

[41] Alec Helbling, Evan Montoya, and Duen Horng Chau. “ObjectComposer: Consistent Generation of Multiple Objects Without Fine-tuning”. In: *arXiv preprint arXiv:2310.06968* (2023).

[42] Amir Hertz et al. “Prompt-to-Prompt Image Editing with Cross Attention Control”. In: *arXiv preprint arXiv:2208.01626* (2022).

[43] Jack Hessel et al. “CLIPScore: A Reference-free Evaluation Metric for Image Captioning”. In: *EMNLP*. 2021.

[44] Martin Heusel et al. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *NeurIPS* (2017).

[45] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *NeurIPS* (2020).

[46] Edward J Hu et al. “LoRA: Low-Rank Adaptation of Large Language Models”. In: *ICLR*. 2022.

[47] Yushi Hu et al. “TIFA: Accurate and Interpretable Text-to-Image Faithfulness Evaluation with Question Answering”. In: *arXiv preprint arXiv:2303.11897* (2023).

[48] Shaohan Huang et al. “Language is not all you need: Aligning perception with language models”. In: *NeurIPS* (2023).

[49] Yi Huang et al. “Diffusion Model-Based Image Editing: A Survey”. In: *arXiv preprint arXiv:2402.17525* (2024).

[50] Yuzhou Huang et al. “SmartEdit: Exploring Complex Instruction-based Image Editing with Multimodal Large Language Models”. In: *CVPR*. 2024.

[51] Mude Hui et al. “HQ-Edit: A High-Quality Dataset for Instruction-based Image Editing”. In: *arXiv preprint arXiv:2404.09990* (2024).

[52] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *CVPR*. 2019.

[53] Shyamgopal Karthik et al. “If at First You Don’t Succeed, Try, Try Again: Faithful Diffusion-based Text-to-Image Generation by Selection”. In: *arXiv preprint arXiv:2305.13308* (2023).

[54] Bahjat Kawar et al. “Imagic: Text-based real image editing with diffusion models”. In: *CVPR*. 2023.

[55] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *ICLR*. 2014.

[56] Durk P Kingma and Prafulla Dhariwal. “Glow: Generative flow with invertible 1x1 convolutions”. In: *Advances in neural information processing systems* 31 (2018).

[57] Durk P Kingma et al. “Improved variational inference with inverse autoregressive flow”. In: (2016).

[58] Max Ku et al. “VIEScore: Towards Explainable Metrics for Conditional Image Synthesis Evaluation”. In: *arXiv preprint arXiv:2312.14867*. 2023.

[59] Xin Lai et al. “LISA: Reasoning Segmentation via Large Language Model”. In: *CVPR*. 2024.

[60] Hugo Laurençon et al. “Building and better understanding vision-language models: insights and future directions”. In: *NeurIPS Workshops*. 2024.

[61] Dongxu Li et al. *LAVIS: A Library for Language-Vision Intelligence*. 2022.

[62] Junnan Li et al. “BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models”. In: *ICML*. 2023.

[63] Junnan Li et al. “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation”. In: *ICML*. 2022.

[64] Yumeng Li et al. “Divide & Bind Your Attention for Improved Generative Semantic Nursing”. In: *BMVC*. 2023.

[65] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *ECCV*. 2014.

[66] Zhiqiu Lin et al. “Evaluating Text-to-Visual Generation with Image-to-Text Generation”. In: *arXiv preprint arXiv:2404.01291*. 2024.

[67] Haotian Liu et al. “Visual Instruction Tuning”. In: *NeurIPS*. 2023.

[68] Luping Liu et al. “Pseudo Numerical Methods for Diffusion Models on Manifolds”. In: *ICLR*. 2022.

[69] Nan Liu et al. “Compositional Visual Generation with Composable Diffusion Models”. In: *ECCV*. 2022.

[70] Shilong Liu et al. “Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection”. In: *ECCV*. 2024.

[71] Ilya Loshchilov and Frank Hutter. “Decoupled Weight Decay Regularization”. In: *ICLR*. 2019.

[72] Cheng Lu et al. “Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps”. In: *NeurIPS* (2022).

[73] Yujie Lu et al. “LLMScore: Unveiling the Power of Large Language Models in Text-to-Image Synthesis Evaluation”. In: *arXiv preprint arXiv:2305.11116* (2023).

[74] Andreas Lugmayr et al. “RePaint: Inpainting using Denoising Diffusion Probabilistic Models”. In: *CVPR*. 2022.

[75] Zhaoyang Lyu et al. “Accelerating diffusion models via early stop of the diffusion process”. In: *arXiv* (2022).

[76] Chuofan Ma et al. “Groma: Localized Visual Tokenization for Grounding Multimodal Large Language Models”. In: *ECCV*. 2024.

[77] Yiwei Ma et al. “I2EBench: A Comprehensive Benchmark for Instruction-based Image Editing”. In: *NeurIPS*. 2024.

[78] Jiafeng Mao, Xueting Wang, and Kiyoharu Aizawa. “Guided Image Synthesis via Initial Image Editing in Diffusion Model”. In: *ACM Multimedia*. 2023.

[79] Chenlin Meng et al. *SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations*. 2022.

[80] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. “Scaling Open-Vocabulary Object Detection”. In: *NeurIPS*. 2024.

[81] Matthias Minderer et al. “Simple Open-Vocabulary Object Detection with Vision Transformers”. In: *ECCV*. 2022.

[82] Alexander Quinn Nichol and Prafulla Dhariwal. “Improved denoising diffusion probabilistic models”. In: *ICLR*. 2021.

[83] Alexander Quinn Nichol et al. “GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models”. In: *ICML*. 2022.

[84] OpenAI. *GPT-4 Technical Report*. 2023. DOI: 10.48550/arXiv.2303.08774.

[85] OpenAI. *GPTV System Card*. 2023. URL: https://cdn.openai.com/papers/GPTV_System_Card.pdf.

[86] Maxime Oquab et al. “DINOv2: Learning Robust Visual Features without Supervision”. In: *arXiv preprint arXiv:2304.07193* (2023).

[87] Mayu Otani et al. “Toward verifiable and reproducible human evaluation for text-to-image generation”. In: *CVPR*. 2023.

[88] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[89] Gaurav Parmar et al. “Zero-shot Image-to-Image Translation”. In: *arXiv preprint arXiv:2302.03027* (2023).

[90] William Peebles and Saining Xie. “Scalable Diffusion Models with Transformers”. In: *ICCV*. 2023.

[91] Dustin Podell et al. “SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis”. In: *arXiv preprint arXiv:2307.01952* (2023).

[92] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *ICML*. 2021.

[93] Aditya Ramesh et al. “Hierarchical Text-Conditional Image Generation with CLIP Latents”. In: *arXiv preprint arXiv:2204.06125* (2022).

[94] Hanoona Rasheed et al. “GLaMM: Pixel Grounding Large Multimodal Model”. In: *CVPR*. 2024.

[95] Royi Rassin et al. “Linguistic Binding in Diffusion Models: Enhancing Attribute Correspondence through Attention Map Alignment”. In: *NeurIPS*. 2024.

[96] Anton Razzhigaev et al. “Kandinsky: an Improved Text-to-Image Synthesis with Image Prior and Latent Diffusion”. In: *arXiv preprint arXiv:2310.03502* (2023).

[97] Scott E. Reed et al. “Generative Adversarial Text to Image Synthesis”. In: *ICML*. 2016.

[98] Robin Rombach et al. “High-resolution image synthesis with latent diffusion models”. In: *CVPR*. 2022.

[99] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *MICCAI*. Springer. 2015, pp. 234–241.

[100] Chitwan Saharia et al. “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”. In: *NeurIPS* (2022).

[101] Tim Salimans and Jonathan Ho. “Progressive distillation for fast sampling of diffusion models”. In: *ICLR* (2022).

[102] Tim Salimans et al. “Improved techniques for training gans”. In: *NeurIPS* (2016).

[103] Dvir Samuel et al. “Generating images of rare concepts using pre-trained diffusion models”. In: *AAAI*. 2023.

[104] Dvir Samuel et al. “Norm-guided latent space exploration for text-to-image generation”. In: *NeurIPS*. 2024.

[105] Axel Sauer et al. “Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis”. In: *ICML*. PMLR. 2023, pp. 30105–30118.

[106] Thomas Scialom et al. “QuestEval: Summarization Asks for Fact-based Evaluation”. In: *EMNLP*. Jan. 2021, pp. 6594–6604.

[107] Noam Shazeer et al. “Outrageously large neural networks: The sparsely-gated mixture-of-experts layer”. In: *arXiv preprint arXiv:1701.06538* (2017).

[108] Shelly Sheynin et al. “Emu Edit: Precise Image Editing via Recognition and Generation Tasks”. In: *CVPR*. 2024.

[109] Jing Shi et al. “A benchmark and baseline for language-driven image editing”. In: *ACCV*. 2020.

[110] Jing Shi et al. “Learning by Planning: Language-Guided Global Image Editing”. In: *CVPR*. 2021.

[111] Jaskirat Singh and Liang Zheng. “Divide, Evaluate, and Refine: Evaluating and Improving Text-to-Image Alignment with Iterative VQA Feedback”. In: *NeurIPS*. 2023.

[112] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *ICML*. 2015.

[113] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising diffusion implicit models”. In: *ICLR* (2020).

[114] Yang Song et al. “Score-Based Generative Modeling through Stochastic Differential Equations”. In: *ICLR*. 2021.

[115] Emma Strubell, Ananya Ganesh, and Andrew McCallum. “Energy and Policy Considerations for Deep Learning in NLP”. In: *ACL*. 2020.

[116] Roman Suvorov et al. “Resolution-robust Large Mask Inpainting with Fourier Convolutions”. In: *WACV*. 2022.

[117] Ming Tao et al. “Galip: Generative adversarial clips for text-to-image synthesis”. In: *CVPR*. 2023, pp. 14214–14223.

[118] Gemma Team et al. “Gemma 3 Technical Report”. In: *arXiv preprint arXiv:2503.19786* (2025).

[119] Ashish Vaswani et al. “Attention Is All You Need”. In: *NeurIPS*. 2017.

[120] Peng Wang et al. “Qwen2-VL: Enhancing Vision-Language Model’s Perception of the World at Any Resolution”. In: *arXiv preprint arXiv:2409.12191* (2024).

[121] Sheng-Yu Wang et al. “CNN-generated images are surprisingly easy to spot... for now”. In: *CVPR*. 2020.

[122] Weilun Wang et al. “Semantic Image Synthesis via Diffusion Models”. In: *arXiv preprint arXiv:2207.00050* (2022).

[123] Zirui Wang et al. “TokenCompose: Grounding Diffusion with Token-level Supervision”. In: *arXiv preprint arXiv:2312.03626* (2023).

[124] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. In: *NeurIPS*. 2022.

[125] Jialian Wu et al. “Grit: A generative region-to-text transformer for object understanding”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 207–224.

[126] Qiucheng Wu et al. “Harnessing the Spatial-Temporal Attention of Diffusion Models for High-Fidelity Text-to-Image Synthesis”. In: *ICCV*. 2023.

[127] Shunyu Yao et al. “Tree of thoughts: Deliberate problem solving with large language models”. In: *NeurIPS* (2023).

[128] Michal Yarom et al. “What you see is what you read? improving text-image alignment evaluation”. In: *NeurIPS* (2024).

[129] Jiabo Ye et al. “mPLUG-Owl3: Towards Long Image-Sequence Understanding in Multi-Modal Large Language Models”. In: *arXiv preprint arXiv:2408.04840* (2024).

[130] Shukang Yin et al. “A survey on multimodal large language models”. In: *National Science Review* (2024).

[131] Xiaohua Zhai et al. “Sigmoid Loss for Language Image Pre-Training”. In: *ICCV*. 2023.

[132] Han Zhang et al. “StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks”. In: *ICCV*. 2017.

[133] Kai Zhang et al. “MagicBrush: A Manually Annotated Dataset for Instruction-Guided Image Editing”. In: *NeurIPS*. 2023.

[134] Shu Zhang et al. “Hive: Harnessing human feedback for instructional visual editing”. In: *CVPR*. 2024.

[135] Yichi Zhang et al. “GROUNDHOG: Grounding Large Language Models to Holistic Segmentation”. In: *CVPR*. 2024.

[136] Yifan Zhang et al. “Deep Long-Tailed Learning: A Survey”. In: *IEEE Trans. PAMI* (2023).

[137] Zhuosheng Zhang et al. “Multimodal Chain-of-Thought Reasoning in Language Models”. In: *Transactions on Machine Learning Research* (2023).

[138] Zhenyu Zhou et al. “Simple and fast distillation of diffusion models”. In: *NeurIPS* (2024).