NePTune: A Neuro-Pythonic Framework for Tunable Compositional Reasoning on Vision-Language

Danial Kamali
Michigan State University
kamalida@msu.edu

Parisa Kordjamshidi
Michigan State University
kordjams@msu.edu

Abstract

Modern Vision-Language Models (VLMs) have achieved impressive performance in various tasks, yet they often struggle with compositional reasoning, the ability to decompose and recombine concepts to solve novel problems. While neuro-symbolic approaches offer a promising direction, they are typically constrained by crisp logical execution or predefined predicates, which limit flexibility. In this work, we introduce NePTune, a neuro-symbolic framework that overcomes these limitations through a hybrid execution model that integrates the perception capabilities of foundation vision models with the compositional expressiveness of symbolic reasoning. NePTune dynamically translates natural language queries into executable Python programs that blend imperative control flow with soft logic operators capable of reasoning over VLM-generated uncertainty. Operating in a training-free manner, NePTune, with a modular design, decouples perception from reasoning, yet its differentiable operations support fine-tuning. We evaluate NePTune on multiple visual reasoning benchmarks and various domains, utilizing adversarial tests, and demonstrate a significant improvement over strong base models, as well as its effective compositional generalization and adaptation capabilities in novel environments.

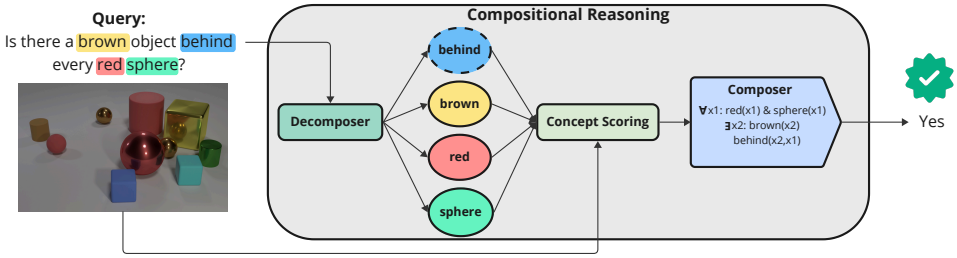

Figure 1: A natural language query ("*Is there a brown object behind every red sphere?*") is *decomposed* into symbolic concepts, such as red, and $sphere$. These concepts are then *composed* to enable explicit reasoning over objects and their relations. This illustrates how complex queries can be mapped into structured logical forms.

1 Introduction

One key aspect of intelligence is the ability to generalize and compose known basic components to solve novel, complex problems. In vision-language reasoning, this capacity for compositional generalization is crucial. Humans easily decompose a complex query like *"Is there a brown object*

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: SPACE in Vision, Language, and Embodied AI.

behind every red sphere?" into its constituent concepts and reason about their relationships [23], as illustrated in Figure 1. However, modern vision-language models with transformer-based architectures often fail at dealing with novel compositions, revealing a gap between their pattern-matching skill and robust, human-like understanding [35].

Recent studies have demonstrated the superiority of neuro-symbolic (NeSy) methods over end-to-end neural architectures for reasoning beyond their observed data [13, 33]. In recent years, methods such as VisProg [6] and ViperGPT [26] have leveraged Large Language Models (LLMs) to produce programs from language queries, breaking down complex questions into a sequence of executable steps. These programs are then run using pre-trained vision models for perceptual grounding. Although powerful, this paradigm has key shortcomings. First, these systems often rely on a sequence of crisp, intermediate decisions, creating brittle pipelines that are sensitive to perceptual errors. A single mistake in an early step (e.g., misidentifying an object) can cause the entire reasoning chain to fail. Second, these pipelines are inference-only and non-differentiable, which prevents them from adapting to new domains where pre-trained models may not perform well. On the other hand, methods such as LEFT [8] and NeSyCoCo [13] employ differentiable declarative symbolic reasoning, but their predicates are limited to the set of learned predicates from training on the target domain, which limits their zero-shot applicability. Table 1 summarizes this landscape and highlights where NePTune differs.

Table 1: Comparison of recent neuro-symbolic frameworks. We analyze the reasoning scope (local vs global), types of supported predicates, and predicate source. **VFM:** vision foundation models, e.g. XVLM; **VLM:** vision-language models, e.g. Qwen2VL.

			Predicate Types				**Predicates**	
Model	**Reasoning**	**Supported Predicates**	**Class**	**Attr.**	**Rel.**	**Spatial**	**Pretrained**	**Trainable**
VisProg	Local	Predefined Modules	✓	✓	✗	✓	Modules	✗
ViperGPT	Local	Dynamic	✓	✓	✗	✓	VFMs	✗
LEFT	Global	Limited to Training Data	✓	✓	✓	✓	✗	✓
NeSyCoCo	Global	Limited to Similar Data as Training	✓	✓	✓	✓	✗	✓
NAVER	Global	Dynamic	✓	✓	✓	✓	VFMs	✗
NePTune	Hybrid	Dynamic	✓	✓	✓	✓	VLMs	✓

To address the challenges of expressive compositional inference and domain adaptation, we propose NePTune, a neuro-symbolic visual reasoning framework. NePTune leverages an LLM to generate expressive Python programs where its execution does not solely rely on crisp decisions. Instead, it performs both soft compositional reasoning on the continuous scores provided by a VLM under uncertainty and imperative sequential reasoning. The result is a framework that decouples reasoning from the perception of atomic concepts, leading to remarkable generalization.

Contributions.

1. A Hybrid Neuro-Symbolic Execution Model: We propose a novel framework that seamlessly combines the imperative control flow of Python with soft compositional logic, enabling complex reasoning under uncertainty.
2. Domain Adaptable Framework: We present a modular system that uses an LLM to generate programs on the fly, eliminating the need for predefined predicates and enabling zero-shot generalization as well as neuro-symbolic fine-tuning for domain adaptation.
3. Strong Compositional Generalization: We demonstrate through extensive experiments that our approach significantly outperforms existing methods, particularly on challenging domain-shift benchmarks, showing highly robust results for vision-language reasoning.

2 Related Work

NePTune is a framework that integrates multi-modal foundation models and logical reasoning to achieve compositional vision-language reasoning. Therefore, we focus on the three topics below.

2.1 Compositional Vision-Language Reasoning

Compositional reasoning is central to building reliable vision–language systems [25, 10], enabling models to represent and execute multi-step, relational structures [21]. While VLMs achieve strong results across broad tasks [31, 28, 17], diagnostic evaluations like CLEVR [11] and ReaSCAN [30] expose persistent gaps in compositional generalization [35] and brittleness under distribution shift [33, 16] in end-to-end neural methods. Although improved architectures show promise [24], explicit and structured neuro-symbolic methods have been particularly effective on these challenges [13]. Motivated by these findings, we adopt a neuro-symbolic approach that performs explicit compositional reasoning to achieve more robust and generalizable visual reasoning.

2.2 Neuro-Symbolic Approaches

Neuro-symbolic methods integrate neural perception with symbolic reasoning. Recent approaches utilize LLMs as semantic parsers to interpret linguistic queries. There are two common strategies for such methods. One line of work, including VisProg [6] and ViperGPT [26], generates imperative code for a sequence of local reasoning steps that rely on crisp decisions. While powerful, the sequential nature of these methods limits global reasoning under the uncertainty of these local decisions. The second strategy, employed by models such as NSCL [20], LEFT [8], and NeSyCoCo [13], focuses on generating a declarative, symbolic representation of the query to enable global reasoning. However, these frameworks rely on training data for concept learning and grounding in each domain, and are therefore constrained by the observed predicates in the training data. While NeSyCoCo improves concept generalization by utilizing a predicate embedding model and covers the lexical variety of natural language, it remains limited to concepts similar to those in its training data. NAVER [3], on the other hand, emphasizes orchestration through a finite-state controller that manages neural perception and utilizes ProbLog for global reasoning in referring expression grounding. However, our findings indicate that generating ProbLog queries is more error-prone for LLMs compared to Python code. Our aim in this work is to synthesize the advantages of both approaches. NePTune is a neuro-symbolic framework that performs both imperative and soft compositional reasoning over atomic predicates. Unlike works like NeSyCoCo and LEFT, the training is not necessary since the concept scores can be obtained zero-shot by harnessing the power of VLMs. However, with differentiable composition functions, it offers optional fine-tuning for domain adaptability by providing the capacity for both compositional inference as well as neuro-symbolic training.

2.3 Predicate Level Concept Understanding

A critical distinction between reasoning frameworks lies in how they derive and utilize concept-level understanding. General-purpose VLMs [28, 17] learn concepts implicitly through end-to-end training, embedding this knowledge directly into the model's weights. In contrast, neuro-symbolic methods externalize this process. A program generating framework, such as VisProg [6], utilizes the concept grounding through a library of specialized, trained, hard-coded vision APIs, which return discrete labels or values. Similarly, ViperGPT [26] utilizes foundation vision-language models, such as XVLM [34], to obtain labels and discrete decisions. Another class of methods, including LEFT and NeSyCoCo, trains an MLP to generate continuous concept scores, representing the relations or attributes of an object in the visual regions. Our approach utilizes visual prompting, highlighting parts of the image via bounding boxes, to elicit referential concept scores from VLMs, which then serve as scores within our framework for reasoning under uncertainty.

3 Methodology

The NePTune framework performs visual reasoning via three core components, as shown in Figure 2. The process begins with the **LLM-based Program Generator**, which translates a natural language query into both a Python program and a set of relevant object names. These names are passed to the **Perceptual Grounding** to detect all relevant candidate objects in the scene. Finally, the **Symbolic Executor** runs the generated program. During execution, it interacts with the Grounding Interface to obtain atomic concept scores and reason over them to get the final answer.

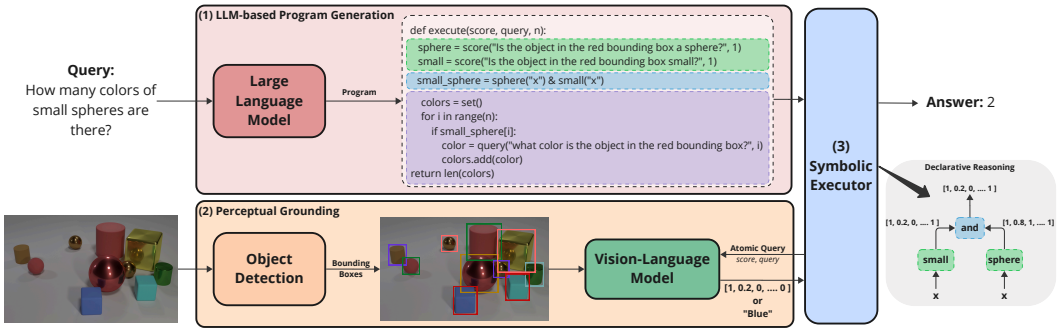

Figure 2: NePTune overview. Given an image and a query, the **(1) LLM-based Program Generation** converts the natural language query to a Pythonic program. Then **(2) Perceptual Grounding** extracts the object bounding boxes. The **(3) Symbolic Executor** then runs the Python code to reason over concepts extracted from the VLM using both soft composition and imperative logic to derive the final answer.

3.1 Component 1: LLM-based Program Generation

The first component of NePTune addresses semantic parsing. We leverage an LLM as a powerful few-shot parser to convert a natural language query into a formal, executable Python program. Given a query like "*Is there a big brown dog?*", the LLM is prompted to decompose it into a multi-step program that first identifies a set of candidate "dogs", and then reasons over the composition of atomic concepts such as "big", "dog", and "brown" to get the final answer. In the same LLM call, it also extracts object names for the region proposals. We chose Python as the target symbolic language for two key reasons. First, its Turing-complete nature, including rich control flow structures like loops and conditionals, is essential for expressing complex procedural reasoning. Second, the prevalence of open-source Python code makes LLMs particularly adept at generating it.

3.2 Component 2: Perceptual Grounding

This component connects the symbolic program to the visual world. It consists of two main parts: an object proposal module and a concept grounding module.

3.2.1 Object Proposal Generation

To ground objects in the scene to the concepts such as `blue` in the program, we must first identify all potentially relevant objects in the scene. We use the object names extracted by the LLM. For example, from "*Are there more cats to the left of the vase or dogs?*", the LLM extracts "dog", "vase", and "cat". We then feed these object names into Grounding DINO [18], a zero-shot object detection model that takes the image and the list of object names as input and outputs a set of bounding boxes for all matching objects. We note that while this textual query-based proposal is chosen for computational efficiency in real-world applications, our framework can operate with any reasonable region proposal method.

3.2.2 Concept Grounding Interface

Once a query is translated into a program, we must ground its atomic predicates to the visual content of the image, such as the objects in the scene. We handle this *perceptual grounding* using a VLM through two primary simplified interfaces `score` and `query`, where the image and bounding boxes are abstracted away with global variables for simplicity. The formal definitions of these grounding functions are as follows:

`score(query:str, num_objects:int)`: This function is the core of our reasoner. It takes a natural language question, an image, and related bounding boxes, and returns the concept probabilities for related object bounding boxes. This score is the VLM's normalized confidence for a "Yes" answer, computed from the logits of the "Yes" and "No" tokens. Formally, given an image I, set of N detected

objects, a visual prompt v, and atomic query q_a, the score s is:

$$s = p(\text{"Yes"}|I, v, q_a) = \frac{e^{\text{logit("Yes")}}}{e^{\text{logit("Yes")}} + e^{\text{logit("No")}}}$$

We treat these outputs as probability scores. To generate an object-centric concept score or answer via a VLM, we represent a symbolic predicate with a natural language question. For example, for the symbolic predicate `blue`, we use "*Is the object in the red bounding box blue?*". The bounding box b provided to the functions will be drawn on target objects as a *visual prompt*, and it is controlled by the `num_objects` parameter, which specifies the predicate type:

- `num_objects=0`: For queries about the entire image (e.g., "*Is the photo taken indoors?*"). No visual prompt is used, and the output is a scalar probability.
- `num_objects=1`: For single-object queries, we mark the target object with a red bounding box around it as a visual prompt (e.g., "*Is the object in the red bounding box brown?*"). The output is a vector of size N.
- `num_objects=2`: For multi-object (relational) queries, we mark the main object with a red bounding box, and the secondary object with a green bounding box, as the visual prompt (e.g., "*Is the person in the red box talking to the person in the green box?*"). The output is an $N \times N$ matrix.

`query(query:str, object_id:int)`: This query function is used for tasks requiring open-ended answers. It takes a natural language question, an image, and an optional visual prompt, and returns a natural language string. Formally, given an image I, visual prompt v, and atomic query q_a, the generated answer A:

$$A = \underset{\text{response}}{\text{argmax}}\, p(\text{response}|I, v, q_a)$$

The answer A serves two purposes. First, it can provide the final output for questions that ask "*what*" or "*which*" (e.g., 'return query("*What color is the object in the red bounding box?*")'). Second, it can be used as an intermediate variable to enable complex conditional constructs within the generated Python program, allowing the reasoning path to change based on the VLM's perception (e.g., 'if query("*What shape is the object in the red bounding box?*") == "*cube*": ...').

3.3 Component 3: Symbolic Executor

The final component of NePTune is the **Symbolic Executor**, which runs the LLM-generated program. A key innovation of our framework is its hybrid execution model, which integrates two distinct reasoning modes.

1) Soft Compositional Reasoning: To reason about visual concepts themselves, we employ a set of soft logical operations based on fuzzy logic principles. Instead of operating on binary true/false values, these operations work directly on the uncertainty scores obtained from the VLM. This is implemented through our custom data structures, which encapsulate the scores for a given predicate and overload standard Python operators (`&` for AND, `|` for OR) to perform the corresponding logical operations. Details of these operations are shown in Table 2. For example, when the program executes `brown & dog`, our framework takes the element-wise minimum of the scores in the two corresponding tensors, implementing the fuzzy t-norm for conjunction.

2) Imperative Reasoning: Our symbolic executor leverages a standard Python interpreter to handle the program's overall structure and control flow. This is possible since we have defined iteration and Boolean operations on the concept objects, allowing for complex, procedural logic, including conditionals (*if/else*), loops (*for*), and variable assignments, giving our framework the full expressive power of a general-purpose programming language. While some of these operations such as counting over string sets (as shown in Figure 2) can break the computation graph. This hybrid design enables the system to reason fluidly under uncertainty while fully leveraging the expressive power of a general-purpose programming language.

4 Experiments

We structure our experiments around four research questions. **RQ1:** How does NePTune perform on zero-shot compositional reasoning on synthetic data? (Experiment 1), **RQ2:** How does NePTune

Table 2: Mathematical Expressions: Logical Forms, Descriptions, and Differentiable Implementations. Here α represents an object-centric or scalar probabilistic score, β represents a relation probability score, $\tau = 0.25$ is a temperature parameter, and $\gamma = 0.25$ is a margin.

Syntax	**Logical Form**	**Description**	**Differentiable Implementation**
α_x.exists()	$\exists x\, \alpha_x$	Existential quantification	$\max(\alpha_x)$
α_x.forall()	$\forall x\, \alpha_x$	Universal quantification	$\min(\alpha_x)$
$\alpha_x \,\&\, \alpha_y$	$\alpha_x \wedge \alpha_y$	Logical conjunction	$\min(\alpha_x, \alpha_y)$
$\alpha_x \,\&\, \beta_{xy}$	$\alpha_x \wedge \beta_{xy}$	Relational conjunction	$\sum_y \alpha_x \cdot \beta_{xy}$
$\alpha_x \mid \alpha_y$	$\alpha_x \vee \alpha_y$	Logical disjunction	$\max(\alpha_x, \alpha_y)$
α_x.implies(α_y)	$\alpha_x \rightarrow \alpha_y$	Logical implication	$\max(1 - \alpha_x, \alpha_y)$
not α_x	$\neg \alpha_x$	Logical negation	$1 - \alpha_x$
α_x.iota(var)	$\iota(\text{var}, \alpha_x)$	Best match	$\text{softmax}(\alpha_x)$
α_x.count()	$\text{count}(\alpha_x)$	Counting elements	$\sum \alpha_x$
$s_1 == s_2$	$s_1 = s_2$	Scalar equality	$\sigma\left(\frac{\tau(\gamma - \lvert s_1 - s_2 \rvert)}{\gamma}\right)$
$s_1 > s_2$	$s_1 > s_2$	Scalar inequality	$\sigma\left(\tau(s_1 - s_2 - 1 + \gamma)\right)$

perform on complex human-generated questions? (Experiment 2), **RQ3:** How does NePTune perform in grounding referring expressions in natural images? (Experiment 3), and **RQ4:** How does NePTune perform under domain shift and adapt to an unseen environment? (Experiment 4). To address these questions, we evaluate NePTune across a diverse set of datasets covering synthetic, human-annotated, and realistic environments, spanning both question answering and referring expression grounding tasks. Details of the datasets and tasks are provided in Appendix D.

4.1 Experiment 1: Core Compositional Reasoning

Table 3: Accuracy comparison across CLEVR question categories. Results are shown by reasoning type and model paradigm, including zero-shot, end-to-end, and neuro-symbolic approaches.

	InternVL2.5	**NePTune**	**ViperGPT**	**NeSyCoCo**	**LEFT**
Training	Zero-Shot	Zero-Shot	Zero-Shot	Trained	Trained
Category	End-to-End	NeSy	NeSy	NeSy	NeSy
Final Accuracy	90.25	**92.65** (↑ 2.40)	36.05	99.68	99.50
Exist	87.10	**93.19** (↑ 6.09)	48.75	99.28	98.92
Query Attribute	**98.26**	96.81 (↓ 1.45)	29.42	100.00	99.86
Compare Attribute	**98.61**	91.94 (↓ 6.67)	53.06	99.44	99.72
Count	74.60	**87.10** (↑ 12.50)	21.37	99.79	98.99
Compare Number	90.86	**92.57** (↑ 1.71)	48.57	100.00	100.00

To evaluate NePTune's reasoning capabilities, we evaluate its performance on the **CLEVR** benchmark. CLEVR is a standard benchmark for compositional reasoning that features synthetic 3D-rendered images and questions testing compositional visual reasoning. Details of the question categories are provided in Appendix D. The results in Table 3 show that, within the family of neuro-symbolic systems, NePTune establishes itself as the strongest zero-shot method, achieving 92.65% accuracy compared to only 36.05% for ViperGPT. While trained approaches such as NeSyCoCo and LEFT nearly saturate CLEVR (99%), NePTune demonstrates that competitive compositional reasoning can be achieved without dataset-specific supervision. Compared to its backbone VLM, InternVL2.5, NePTune still yields improvements raising overall accuracy from 90.25% to 92.65% (↑ 2.40%). The largest gains appear in quantitative categories where explicit compositional structure is most useful: *Count* rises from 74.60% to 87.10% (↑ 12.50%), and *Compare Number* increases from 90.86% to 92.57% (↑ 1.71%). We also see a notable improvement on *Exist* from 87.10% to 93.19% (↑ 6.09%), consistent with the executor reducing spurious correlation when filtering by attributes and relations. In contrast, attribute-heavy categories regress: *Query Attribute* drops by ↓ 1.45% points and *Compare Attribute* drops by ↓ 6.67% points. A closer look at concept-level accuracy revealed that analogical concepts such as *same color* or *same shape* remain among the most challenging to capture in this benchmark. More discussion of atomic concept evaluation is provided in Section 5.
In addition to the original CLEVR benchmark, we evaluate our method on various compositional challenges based on the CLEVR environment introduced in LEFT [8], including referring expressions

Table 4: Accuracy on CLEVR extension tasks. Methods marked with † use ground-truth programs. Improvements from the backbone VLM (InternVL2.5) are marked with the arrow sign (↑).

	Method	Ref(%)	Puzzles(%)	RPM(%)
Trained	NeSyCoCo†	**100.00**	**95.00**	**100.00**
	NeSyCoCo	94.00	94.00	74.00
	LEFT	94.00	85.00	87.00
Zero-shot	Qwen2VL.5-8B	21.00	43.00	**53.00**
	InternVL2.5-8B	**27.00**	**52.00**	47.00
	Ovis1.6-9B	4.00	47.00	49.00
	ViperGPT	8.00	34.00	4.00
	VisProg	35.00	27.00	51.00
	NePTune†	**99.00** (↑ 72)	**65.00** (↑ 13)	**99.00** (↑ 52)
	NePTune	91.00 (↑ 64)	60.00 (↑ 8)	80.00 (↑ 33)

CLEVR-Ref, visual puzzles **CLEVR-Puzzles**, and Raven's Progressive Matrices **CLEVR-RPM**. As shown in Table 4, NePTune shows the highest performance among the zero-shot methods and outperforms the end-to-end VLMs. In addition, compared to NeSyCoCo and LEFT with trained concepts, we show competitive performance.

4.2 Experiment 2: Complex Human Queries

Here, we evaluate NePTune in a more complex and diverse human-generated language. We utilize the **CLEVR-Humans** [12] benchmark to evaluate and compare NePTune against other neuro-symbolic and end-to-end models. As shown in Table 5, NePTune significantly outperforms prior declarative neuro-symbolic methods such as LEFT and NeSyCoCo by a large margin of ↑ 30.98% even compared to end-to-end methods such as MDETR [14]. It also surpasses its imperative symbolic backbone, ViperGPT. Furthermore, it improves upon its powerful end-to-end backbone (InternVL2.5-8B) by ↑ 1.72%, demonstrating its effectiveness on complex, human-generated questions.

Table 5: Accuracy on the CLEVR-Humans (CH).

	Method	CH (%)
Trained	LEFT	56.69
	NeSyCoCo	56.12
	MDETR	**81.73**
Zero-shot	Qwen2VL-7B	84.12
	InternVL2.5-8B	**85.95**
	Ovis1.6-9B	79.96
	ViperGPT	31.05
	NePTune	**87.67**

4.3 Experiment 3: Real-world Images

While various versions of CLEVR used in the aforementioned experiments are strong testbeds for compositional reasoning, after all, these are toy environments with limited diversity. To demonstrate NePTune's ability to operate in realistic environments, we evaluate it on Referring Expression Grounding (REG) of real-world images using the **RefCOCO-Adversarial** [1] (Ref-Adv) benchmark. On Ref-Adv, NePTune is the strongest zero-shot reasoner among symbolic baselines such as ViperGPT and NAVER. To ensure a fair comparison with NAVER, we follow its setup and use the same backbones (Grounding DINO + XVLM + InternVL2-8B), denoted as NePTune‡. Under this setting, NePTune‡ reaches 63.71 vs. 36.45 for NAVER† (execution only), and with a simple verification (similar to NAVER) attains 71.57 vs. 65.13 for NAVER. Compared to end-to-end methods, NePTune surpasses all the specialized grounding methods, such as Grounding DINO and Florence2 [31]. Additionally, compared to their VLM backbones, both NePTune and NePTune‡ with verification surpass their respective backbones. Details on verification are available in Appendix E.

Table 6: Accuracy on Ref-Adv dataset.

Method	Ref-Adv (%)
Grounding DINO-B	60.85
Florence2-L	71.73
Ovis1.6-9B	30.70
InternVL2-8B	72.92
InternVL2.5-8B	76.13
ViperGPT	60.66
NAVER†	36.45
NAVER	65.13
NePTune‡	63.71
+ Verification	75.54
NePTune	71.57
+ Verification	**78.08**
NePTune (1B)	60.69
+ Fine-tuning	68.06 ± 0.56
+ Verification	74.59 ± 0.12

4.4 Experiment 4: Generalization and Adaptation

When using popular benchmarks, as we did in Experiments 1-3, the possible data contamination makes zero-shot performance less reliable. In this experiment, we test NePTune's ability to generalize and adapt to novel environments using a less popular photo-realistic gaming environment that is **Ref-GTA** [27] benchmark. Ref-GTA is a challenging REG benchmark with images from a game simulation, which creates a significant domain shift from the natural images on which most VLMs are pre-trained. Based on the reports of Ovis1.6 [19] and InternVL2.5 [4], this source is not included in their training data. As shown in Table 7, this domain shift causes a significant performance drop for most methods. The powerful end-to-end InternVL2.5-8B model, for example, fails catastrophically, with its accuracy dropping to 6.95%. In contrast, NePTune, when paired with the same VLM, achieves a remarkable 69.69% accuracy. This demonstrates that by utilizing our global symbolic reasoner and atomic-level concept understanding, our framework achieves robustness and compositional generalization that monolithic models lack. Furthermore, while NePTune demonstrates strong zero-shot robustness, its soft operations also enable fine-tuning. As shown in Table 7, by fine-tuning a smaller VLM (InternVL2.5-1B) using our neuro-symbolic differentiable computations with only 1000 samples, we can further improve its performance and obtain 69% accuracy. While fine-tuning VLM using original neural training, it only reaches 32% accuracy. These findings are promising for future research and using neuro-symbolic reasoning as a source of supervision for larger-scale VLM training. More details on fine-tuning are presented in Appendix F.

Table 7: Accuracy on Ref-GTA benchmark.

Method	**Ref-GTA(%)**
GroundingDINO-B	27.90
Florence2-L	58.65
Ovis1.6-9B	2.78
InternVL2.5-8B	6.95
InternVL2.5-1B	1.64
+ Fine-tuning	32.61 ± 0.35
ViperGPT	1.40
NAVER†	54.84
NAVER	58.73
NePTune‡	62.73
NePTune	**69.69**
NePTune (1B)	34.92
+ Fine-tuning	69.90 ± 1.16

5 Discussion

VLM for Concept Understanding. A central idea of NePTune is to employ VLMs as underlying perception modules and concept grounders. The basic assumption here is that while VLMs are prone to fail at reasoning over complex compositions, they should perform better in perceiving basic concepts. However, basic perception questions require isolating parts of the visual input, a process known as visual prompting, which has been shown to be challenging in complex questions [2]. In this section, we aim to evaluate the hypothesis that modern VLMs guided by visual prompts are effective as concept grounders for our purpose. We utilize ground-truth scene graphs from the **CLEVR** [11] and **Visual Genome** [15] datasets to automatically generate a set of atomic templated questions. For CLEVR, we create a benchmark using scene graphs of 200 sampled scenes, and for Visual Genome, we randomly sample 1000 questions from 200 scene graphs. We generate simple Yes or No questions about their class, attributes, and relations, such as, "*Is the object in the bounding box a bird?*". We then use our `score` function that employs the underlying VLM to answer these atomic questions.

Table 8: Backbone VLMs performance on CLEVR and VG scene graphs (Micro F1-Score x 100)

Category	**InternVL2.5-8B**		**Ovis1.6-9B**		**Qwen2VL-7B**	
	CLEVR	**VG**	**CLEVR**	**VG**	**CLEVR**	**VG**
Attribute	**97.25**	**89.37**	95.87	86.75	89.27	83.45
Class/Object	90.00	81.76	**91.98**	80.42	82.21	**83.28**
Relation	**90.94**	82.35	87.91	80.00	80.04	**83.02**
Spatial	**89.82**	93.42	86.83	93.39	89.53	**94.41**
Overall	**94.54**	90.41	90.35	88.19	86.59	**90.77**

The results are shown in Table 8. Our analysis reveals that VLMs are particularly strong at identifying object, classes, properties, and spatial relations. However, the performance is weaker on more complex relational concepts, especially for analogical comparisons such as *same size*. This problem is more severe in the Qwen2VL model. Although visual prompting can be problematic for general

VQA [2], our results show this is not as challenging in our setting for grounding atomic queries. These results demonstrate that VLMs perform significantly better in answering atomic queries compared to complex multi-step queries, as reported in Tables 4 and 5. For example, there is a performance gap of up to 67% when comparing CLEVR-Ref (27%) compared to CLEVR average atomic evaluation (94%) using InternVL2.5. Detailed concept-level results of the experiment are presented in the Appendix B. Furthermore, the results in Table 9 demonstrate the end-to-end performance of different VLMs in NePTune, where backbones show an average improvement of up to ↑ 29%.

Table 9: Performance of different VLM backbones (end-to-end) and with NePTune across benchmarks. Δ values indicate gains or losses relative to the raw VLMs.

Model	CLEVR (%)	Ref-Adv (%)	Ref-GTA (%)	Ref (%)	Puzzles (%)	RPM (%)	Avg. (%)
InternVL2.5-8B	90.25	76.13	6.95	27.00	52.00	47.00	49.89
+ NePTune	92.65 ↑ 2.40	78.08 ↑ 1.95	69.69 ↑ 62.74	91.00 ↑ 64.00	60.00 ↑ 8.00	80.00 ↑ 33.00	78.57 (↑ 28.68)
Ovis1.6-9B	85.00	58.47	2.78	4.00	47.00	49.00	41.04
+ NePTune	88.80 ↑ 3.80	63.25 ↑ 4.78	56.32 ↑ 53.54	75.00 ↑ 71.00	57.00 ↑ 10.00	80.00 ↑ 31.00	70.06 (↑ 29.02)
Qwen2VL-7B	93.55	81.07	1.75	21.00	43.00	53.00	48.90
+ NePTune	82.55 ↓ 11.00	80.93 ↓ 0.14	68.87 ↑ 67.12	71.00 ↑ 50.00	53.00 ↑ 10.00	71.00 ↑ 18.00	71.23 (↑ 22.33)

Ablation Study. Table 10 presents an ablation study of NePTune on CLEVR-Humans, starting from a declarative reasoning backbone. Incorporating InternVL2.5 as the concept scoring module yields a ↑ 12.36 improvement, showing that strong VLMs provide more robust and generalizable concept grounders than trained concept grounders on CLEVR. Building on this, adding NePTune's imperative reasoning further boosts accuracy by ↑ 19.19, closing the gap left by purely declarative systems.

Table 10: Ablation study on CLEVR-Humans (CH).

Ablation Setting	CH (%)
Declarative + Trained Concepts	56.12
+ VLM Concepts	68.48 (↑ 12.36)
+ Imperative Reasoning	**87.67** (↑ 19.19)

Choice of Symbolic Program. We analyze program execution success rates (no syntax or runtime errors) on 500 Ref-Adv and CLEVR-Humans using the GPT-4o [22] LLM. As shown in Table 11, frameworks that rely on specialized, non-Pythonic syntax, such as LEFT, NeSyCoCo, and NAVER (which uses ProbLog), frequently suffer from generation failures. We found that the LLM often struggles to correctly translate natural language into their specific, strict formalisms. In contrast, methods that utilize Python, such as ViperGPT and NePTune, demonstrate significantly higher rates of successful program execution. However, our analysis of ViperGPT's failures reveals that its purely imperative approach, which often relies on selecting objects by a fixed index, is a primary source of error. The NePTune hybrid reasoner proves to be the most robust. Our analysis shows that the main source of error for NePTune shifts from low-level syntax errors to higher-level logical errors, where the LLM fails to compose the correct sequence of predicates. Qualitative examples are shown in Appendix A.

Table 11: Execution success rate (%) of different neuro-symbolic methods on Ref-Adv and CLEVR-Humans. Incompatible models are excluded.

Method	Ref-Adv	CLEVR
NeSyCoCo	N/A	70.33±3.47
LEFT	N/A	64.33±3.94
ViperGPT	48.67±1.70	95.42±0.30
NAVER	23.02±4.71	N/A
NePTune	**98.66**±0.82	**97.24**±0.85

6 Conclusion

In this work, we introduce NePTune, a novel neuro-symbolic framework for compositional vision-language reasoning. Our approach leverages a novel hybrid reasoner that combines the imperative control flow of Python with a declarative, probabilistic soft logic that operates directly on the uncertain outputs of a VLM. This design enables our framework to be both highly expressive and robust in the face of perceptual uncertainty. Our extensive experiments demonstrate the effectiveness of this approach. NePTune demonstrates clear improvement over strong baselines, as well as the underlying VLMs, on complex compositional reasoning benchmarks, in visual question answering, and referring expression, exhibiting remarkable generalization capabilities. While the performance of NePTune is dependent on the quality of its underlying components, such as the LLM for program generation and the VLM for concept grounding, our work demonstrates a flexible and powerful

paradigm for building more robust and generalizable AI systems, leveraging these models to elevate their capabilities to a comparatively higher level of performance. Future work could explore methods for more efficient grounding and the integration of our work into visual reasoning orchestration.

Acknowledgments

This project is supported by the Office of Naval Research (ONR) grant N00014-23-1-2417. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Office of Naval Research.

References

[1] Arjun Akula, Spandana Gella, Yaser Al-Onaizan, Song-Chun Zhu, and Siva Reddy. Words aren't enough, their order matters: On the robustness of grounding visual referring expressions. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6555–6565, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.586. URL https://aclanthology.org/2020.acl-main.586/.

[2] Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P. Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. Making large multimodal models understand arbitrary visual prompts. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2024.

[3] Zhixi Cai, Fucai Ke, Simindokht Jahangard, Maria Garcia de la Banda, Reza Haffari, Peter J. Stuckey, and Hamid Rezatofighi. Naver: A neuro-symbolic compositional automaton for visual grounding with explicit logic reasoning. *arXiv preprint arXiv:2502.00372*, 2025.

[4] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025. URL https://arxiv.org/abs/2412.05271.

[5] DeepSeek-AI-Team. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

[6] Tanmay Gupta and Aniruddha Kembhavi. Visual Programming: Compositional visual reasoning without training . pages 14953–14962, June 2023. doi: 10.1109/CVPR52729.2023.01436. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.01436.

[7] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing, 2017. To appear.

[8] Joy Hsu, Jiayuan Mao, Josh Tenenbaum, and Jiajun Wu. What's left? concept grounding with logic-enhanced foundation models. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Drew A. Hudson and Christopher D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering, 2019. URL https://arxiv.org/abs/1902.09506.

[10] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? (extended abstract). In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/708. URL https://doi.org/10.24963/ijcai.2020/708. Journal track.

[11] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

[12] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning, 2017.

[13] Danial Kamali, J. Barezi, Elham, and Parisa Kordjamshidi. Nesycoco: A neuro-symbolic concept composer for compositional generalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(4):4184–4193, Apr. 2025. doi: 10.1609/aaai.v39i4.32439. URL https://ojs.aaai.org/index.php/AAAI/article/view/32439.

[14] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. pages 1760–1770, 2021. URL http://dblp.uni-trier.de/db/conf/iccv/iccv2021.html#KamathSLSMC21.

[15] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations, 2016. URL https://arxiv.org/abs/1602.07332.

[16] Chuanhao Li, Wenbo Ye, Zhen Li, Yuwei Wu, and Yunde Jia. Multi-sourced compositional generalization in visual question answering, 2025. URL https://arxiv.org/abs/2505.23045.

[17] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

[18] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[19] Shiyin Lu, Yang Li, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Han-Jia Ye. Ovis: Structural embedding alignment for multimodal large language model. *arXiv:2405.20797*, 2024.

[20] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJgMlhRctm.

[21] Santiago Ontañón, Joshua Ainslie, Vaclav Cvicek, and Zachary Fisher. Making transformers solve compositional tasks. 1:3591–3607, 8 2021. URL http://arxiv.org/abs/2108.04378.

[22] OpenAI-Team. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

[23] Barbara Partee et al. Compositionality. *Varieties of formal semantics*, 3:281–311, 1984.

[24] Linlu Qiu, Hexiang Hu, Bowen Zhang, Peter Shaw, and Fei Sha. Systematic generalization on gSCAN: What is nearly solved and what is next? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2180–2188, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.166. URL https://aclanthology.org/2021.emnlp-main.166.

[25] Sania Sinha, Tanawan Premsri, and Parisa Kordjamshidi. A survey on compositional learning of ai models: Theoretical and experimental practices, 2024. URL https://arxiv.org/abs/2406.08787.

[26] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning, 2023.

[27] Mikihiro Tanaka, Takayuki Itamochi, Kenichi Narioka, Ikuro Sato, Yoshitaka Ushiku, and Tatsuya Harada. Generating easy-to-understand referring expressions for target identifications, 2019. URL https://arxiv.org/abs/1811.12104.

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

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[30] Zhengxuan Wu, Elisa Kreiss, Desmond C. Ong, and Christopher Potts. ReaSCAN: Compositional reasoning in language grounding. *NeurIPS 2021 Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=Rtquf4Jk0jN.

[31] Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. *arXiv preprint arXiv:2311.06242*, 2023.

[32] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024.

[33] Tian Yun, Usha Bhalla, Ellie Pavlick, and Chen Sun. Do vision-language pretrained models learn composable primitive concepts? *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=YwNrPLjHSL.

[34] Yan Zeng, Xinsong Zhang, and Hang Li. Multi-grained vision language pre-training: Aligning texts with visual concepts. *arXiv preprint arXiv:2111.08276*, 2021.

[35] Wang Zhu, Jesse Thomason, and Robin Jia. Generalization differences between end-to-end and neuro-symbolic vision-language reasoning systems. pages 4697–4711, December 2022. doi: 10.18653/v1/2022.findings-emnlp.345. URL https://aclanthology.org/2022.findings-emnlp.345.

A Qualitative Analysis

A.1 Referring Expressions Grounding

Examples of NePTune on Ref-GTA are shown in Figure 3. **Expressions 1 and 2** show the superiority of compositional reasoning compared to the VLM and object detection backbones. They demonstrate how the VLM fails to correctly locate the object in the new environments, while atomic concepts are correctly scored by the VLM.

A.2 Visual Questions Answering

Examples of NePTune on CLEVR-Humans are shown in Figure 4. **Example 1** illustrates a case where reasoning over meta-concepts such as distinct shapes is required. NeSyCoCo is unable to resolve the query due to this limitation to declarative, while ViperGPT struggles to identify the correct anchor object because it does not perform global reasoning across the object set. NePTune, by contrast, is able to provide the correct interpretation. **Example 2** presents a more complex scenario

Figure 3: Qualitative examples of NePTune on the RefGTA dataset. Green boxes indicate objects detected by Grounding DINO, blue boxes show objects selected by the VLM (InternVL2.5-8B), and red boxes highlight the final selections made by NePTune.

where reflection reasoning and calibration are critical. Both the sphere and the cylinder are reflected in the cyan cube, with the cylinder's reflection being more prominent and thus the more likely answer. NePTune does not fully capture this subtle distinction and produces an incorrect response. The competing baselines, however, fail for different and more fundamental reasons: VLM score calibration issues lead to incorrect selection, and ViperGPT relies on bounding box size comparisons that ignore perspective, which causes it to treat equally sized cubes as different. In this case, the code selects the shiny brown cube in the foreground and concludes that there is no reflection. Although NePTune also errs, this example highlights the inherent difficulty of fine-grained reflection reasoning and the challenges posed by subtle visual cues.

A.3 Failure Example.

Example shown in Figure 5 illustrates an error caused by an incorrect LLM-generated program in NePTune. Two issues occur simultaneously:

1. **Incorrect predicate arity:** The program invokes the function `is_baby_giraffe` with `num_object=2` instead of `1`. This leads to a dimensional mismatch when the resulting score tensor is composed with variable `x2`, causing execution failure.

2. **Incorrect bounding box reference:** The program refers to the bounding box with the color *green* instead of *red*, which is the intended single-object reference.

While errors such as incorrect bounding box colors can be alleviated to a certain extent with regex-based matching, the dimensional inconsistency in predicate scoring is unrecoverable without regeneration.

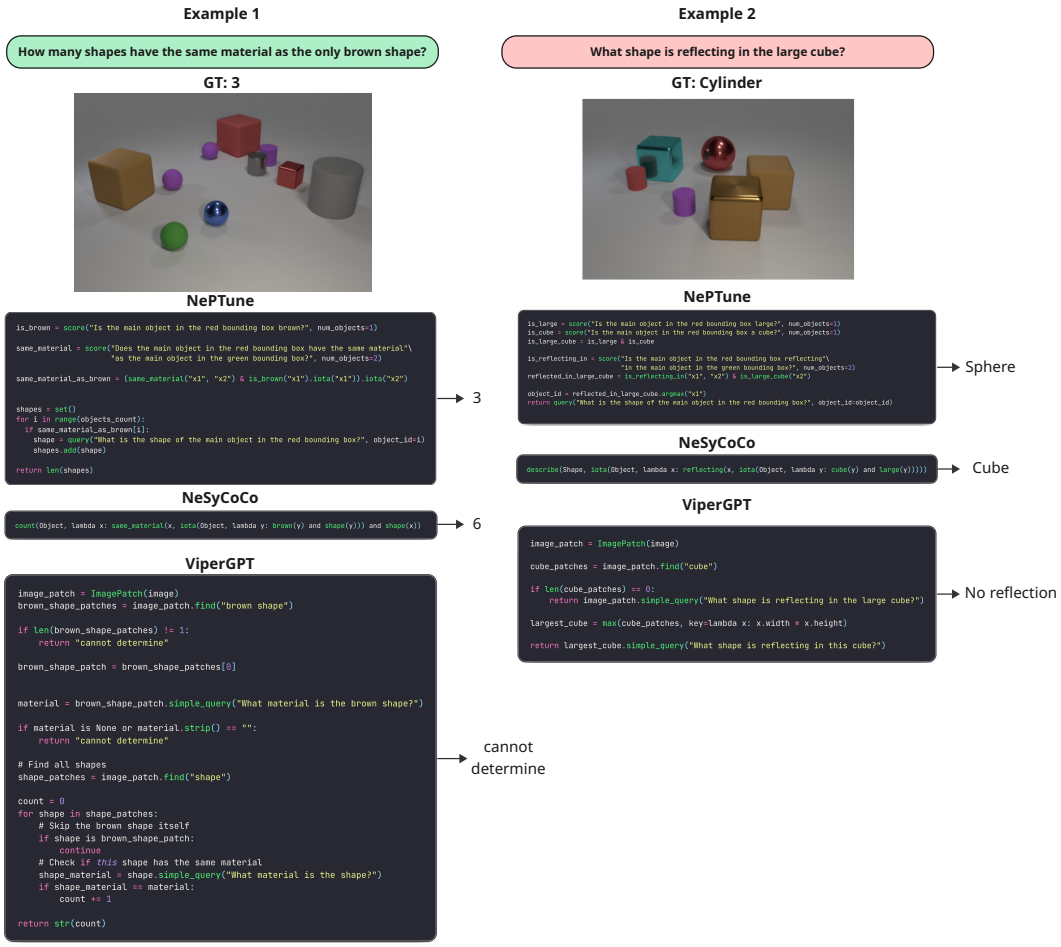

Figure 4: Qualitative examples of NePTune compared to ViperGPT and NeSyCoCo on CLEVR-Humans.

```
is_adult_giraffe = score("Is the main object inside of the red bounding box an adult giraffe?", num_objects=1, type="class")
is_baby_giraffe = score("Is the main object inside of the green bounding box a baby giraffe?", num_objects=2, type="class")
licking = score("Is the adult giraffe inside of the red bounding box licking the baby giraffe inside of the green bounding box?", num_objects=2, type="relation")
final_predicate = is_adult_giraffe('x1') & is_baby_giraffe('x2') & licking('x1', 'x2')
return final_predicate.iota('x1')
```

Figure 5: Qualitative example of wrong NePTune program generation. Mistakes are highlighted with red boxes.

B VLM Concept Analysis

B.1 Concept-Level Performance

An analysis of the atomic concept grounding performance on the CLEVR dataset shown in Table 12 reveals clear patterns of weakness, primarily centered around analogical and complex relational concepts.

B.1.1 Analogical Relations

The most significant challenge for the VLMs concept is abstract, analogical comparisons. The concept *same size* is a clear example, yielding the lowest F1 score in the entire table for Qwen2VL-7B at 0.604. Ovis1.6-9B also struggles significantly with it, scoring just 0.794. While InternVL2.5-8B performs better, *same color* is a notable weak point for it at 0.840. This indicates a systemic difficulty in performing comparative judgments with visual prompting compared to simple identification.

Table 12: F1 Scores for atomic concept grounding on CLEVR using visual prompting.

Concept	Ovis1.6	Qwen2VL	InternVL2.5
blue	**0.972**	0.806	0.986
brown	**0.989**	0.921	**0.989**
cube	0.915	0.952	**0.991**
cyan	0.961	0.805	**1.000**
cylinder	0.964	0.960	**1.000**
gray	0.958	0.911	**0.980**
green	**0.989**	0.838	0.988
large	0.956	0.975	**1.000**
metal	0.961	0.965	**0.997**
purple	0.988	0.927	**1.000**
red	0.973	0.829	**1.000**
rubber	0.980	0.924	**1.000**
small	0.976	0.868	**1.000**
sphere	0.980	0.980	**0.996**
yellow	0.974	0.871	**0.982**
behind	0.836	0.877	**0.868**
front	**0.890**	0.850	0.850
left	**0.919**	0.885	0.943
right	**0.934**	0.863	0.930
same_color	**0.982**	0.957	0.840
same_shape	0.862	0.855	**0.936**
same_size	0.794	0.604	**0.951**
Macro F1	0.943	0.874	**0.964**
Micro F1	0.903	0.866	**0.945**

B.1.2 3D Spatial Relations are a Close Second

The next most difficult category is 3D spatial relationships. Concepts like *behind* and *front* consistently score lower than basic attributes across all models. For instance, the F1 scores for *behind* are 0.836 (Ovis1.6), 0.877 (Qwen2VL), and 0.868 (InternVL2.5). This demonstrates a weaker understanding of object relations in three-dimensional space compared to intrinsic properties like shape or material.

B.1.3 Color Confusion

Certain colors also pose challenges, particularly for the Qwen2VL model. The color *cyan* shows a notable performance dip for Qwen2VL (0.805). Further analysis suggests this is due to confusion with visually similar colors like *blue* (0.806) and *green* (0.838). This pattern highlights that while models can identify common colors well, they are less robust with more specific shades and can struggle to differentiate them.

B.1.4 Real-World Image Challenges in Visual Genome

Transitioning from the synthetic CLEVR environment to the complex, real-world images of the Visual Genome [15] dataset reveals a notable shift in performance dynamics. We evaluate our models on atomic queries generated from the cleaned VG scene graphs in GQA [9] using these templates:

- **Relation Prompt:**
  Is the {class_1/object_1} in the red bounding box {class_2/object_2} the {relation} in the green bounding box?
- **Class Prompt:**
  Is the object {inside of/in} the red bounding box {article} {class}?
- **Attribute Prompt:**
  Is the {class} in the red bounding box {attribute}?

As shown in Table 13, a key challenge emerges in fundamental object identification. While the models remain highly proficient at identifying spatial relations and attributes, their performance in the "Class/Object" category is consistently lower than CLEVR. Ovis1.6-9B drops to an F1-score of 0.804, with InternVL2.5-8B and Qwen2VL-7B at 0.818 and 0.833, respectively. This suggests that the visual complexity, clutter, and vast number of fine-grained categories in real-world images make the foundational task of object recognition a more significant hurdle than in the controlled CLEVR environment.

Table 13: Top 20 common concepts F1-Score comparison on real-world scene graphs.

Concept	Ovis1.6-9B	Qwen2VL-7B	InternVL2.5
right of	0.940	**0.971**	0.962
left of	0.951	0.970	**0.980**
man	0.906	**0.914**	0.892
shirt	0.840	0.893	**0.963**
person	0.821	0.792	**0.824**
window	0.857	0.894	**0.957**
pole	0.842	0.900	**0.927**
building	0.621	0.839	**0.889**
tree	0.750	0.833	**0.839**
wall	0.815	**0.857**	0.800
on	0.636	**0.727**	0.696
sign	0.800	0.815	**0.833**
car	0.870	0.960	**1.000**
hand	**0.929**	0.929	0.923
in	0.444	0.421	**0.500**
white	0.750	**0.889**	0.846
near	**0.923**	0.880	**0.923**
sky	0.500	0.667	**0.800**
ear	0.818	0.800	**0.909**
woman	**0.952**	**0.952**	**0.952**

B.2 End-to-End Performance

These findings align with the paper's broader conclusion that errors in the final reasoning pipeline often originate from the VLM's poor performance on these specific types of relational and analogical concepts. Based on the results of this analysis, in addition to the *score* perceptual interface, we generate a set of the most common spatial predicates, such as *left*, *right*, *behind*, *front*, etc., using the position and depth estimation of bounding boxes for the sake of efficiency and accuracy.

Table 14: Comparison of model accuracies across various categories. Maximum values for each NePTune and end-to-end are bolded. Changes compared to normal models are marked with colors and arrows.

Metric	Qwen2VL	Ovis1.6	InternVL2.5	Qwen-NeSy	Ovis-Nesy	Intern-Nesy
Final Accuracy (%)	**93.55**	85.00	90.25	82.55 (↓ 11.00)	88.80 (↑ 3.80)	**92.65** (↑ 2.40)
Exist (%)	**98.21**	86.74	87.10	84.59 (↓ 13.62)	88.53 (↑ 1.79)	**93.19** (↑ 6.09)
Query Attribute (%)	93.77	90.14	**98.26**	91.30 (↓ 2.47)	96.23 (↑ 6.09)	**96.81** (↓ 1.45)
Compare Attribute (%)	**99.44**	90.28	98.61	78.33 (↓ 21.11)	88.33 (↓ 1.95)	**91.94** (↓ 6.67)
Count (%)	**87.50**	75.40	74.60	70.16 (↓ 17.34)	78.83 (↑ 3.43)	**87.10** (↑ 12.50)
Compare Number (%)	90.29	78.29	**90.86**	88.57 (↓ 1.72)	89.14 (↑ 10.85)	**92.57** (↑ 1.71)

The results in Table 14 clearly show that NePTune enhances the compositional reasoning abilities of strong base VLMs such as Ovis1.6 and InternVL2.5. The most significant gains are on quantitative compositional tasks, with accuracy on counting questions improving by over 12% for InternVL2.5 and counting comparison questions by nearly 11% for Ovis1.6. This highlights the value of our structured, programmatic approach for tasks that end-to-end models find challenging. Conversely, our framework degrades the performance of the already accurate model Qwen2VL. A deeper look reveals that the primary source of this degradation is the analogical questions, which are the most

apparent in the "Compare Attribute" task, where performance drops by 21%. This aligns perfectly with our findings in Experiment 1, which showed that this specific VLM struggles with analogical reasoning (e.g., *same color*, *same shape*). This demonstrates both the power of our framework when paired with a reliable grounding VLM and its sensitivity to the underlying VLM's weaknesses, as errors in perception are propagated by the logical executor.

C Experimental Setup

All experiments were conducted on a server running Ubuntu OS, equipped with an AMD EPYC 7413 24-core CPU, 700GB of system RAM, and an NVIDIA A6000 GPU (48GB). Unless specified otherwise, our core models included DeepSeekV3 [5] as the backbone LLM, GroundingDINO-B for object detection, and DepthAnythingV2 [32] for depth estimation. We utilized the Hugging Face Transformers library [29] for VLM interaction and the InternVL-2.5 codebase for finetuning[1]. To supplement the LLM's output, we also used spaCy [7] Named Entity Recognition to extract keywords from the query.

D Datasets

Here, we provide a detailed breakdown of the datasets used to evaluate the NePTune framework across a range of visual reasoning challenges.

D.1 CLEVR

We evaluate core compositional reasoning using the **CLEVR (Compositional Language and Elementary Visual Reasoning)** dataset [11]. It contains 3D-rendered synthetic images of simple objects and is used for the task of **Visual Question Answering (VQA)**, where the model must answer complex, programmatically generated questions about object attributes, counts, and relations. We measure performance using **Accuracy (%)** and, for answers requiring semantic equivalence matching (e.g., matching "2" with "two"). For CLEVR dataset benchmarks where a deterministic evaluation did not capture the correctness of the answer, such as matching "2" with "two", we use GPT-4o as an AI judge for evaluation. The prompt used for this evaluation is shown in Figure 7. Since the CLVER evaluation set was large (700K), we used the first 2000 samples for evaluating our method.

D.1.1 CLEVR Query Categories

Our fine-grained analysis on CLEVR, presented in Table 3, breaks down performance across the following five distinct types of reasoning skills using the programs in the dataset:

Exist These questions test for the presence of an object with specific properties, requiring a "Yes/No" answer. For example, *"Is there a large green cube behind the small red sphere?"*

Query Attribute These questions ask for a specific property (e.g., color, material) of a uniquely identified object, such as, *"What color is the small shiny cylinder?"*

Compare Attribute This category involves "Yes/No" questions that compare a single property between two objects. For example, *"Does the large sphere have the same material as the small cube?"*

Count These questions require the model to return the total number of objects matching a description, such as, *"How many red metallic objects are there?"*

Compare Number These questions involve comparing the quantities of two different sets of objects, also resulting in a "Yes/No" answer. For example, *"Are there more spheres than cubes?"*

D.2 CLEVR-Humans

The **CLEVR-Humans** dataset [12] allows us to evaluate performance on more natural language. It uses the same synthetic images as CLEVR but features more complex and linguistically diverse questions written by humans. Similar to CLVER, performance is measured by Accuracy (%).

[1] https://internvl.readthedocs.io/en/latest/internvl2.5/finetune.html

D.3 CLEVR Extensions

To probe a wider range of reasoning skills, we use a collection of compositional challenges from Hsu et al., 2024, built on the CLEVR environment [8]. Each task tests a unique aspect of complex reasoning:

CLEVR-Ref This task tests referring expression grounding, where the model must identify a target object based on its relationship to other objects. The evaluation is conducted by measuring the IOU value greater than 0.5.

> *Example: "There is a sphere that is front the gray cylinder, find the small cylinder that is left of it."*

CLEVR-Puzzles This benchmark evaluates the model's ability to solve visual constraint satisfaction problems by finding a set of objects that simultaneously satisfy multiple attribute and relational constraints.

> *Example: "Can you find four objects from the image such that: object 1 is a large metal object; object 2 is a metal object; object 3 is a small rubber cylinder; object 4 is a small yellow metal cylinder; object 1 is front object 2; object 1 is behind object 4; object 1 is behind object 3."*

CLEVR-RPM This task tests abstract relational reasoning by mimicking Raven's Progressive Matrices. The model is presented with objects in a grid that follow a pattern and must identify a candidate object from the scene that correctly completes that pattern.

> *Example: "There are 9 objects, ordered in a 3x3 grid: row 1 col 1 is a small rubber object; row 1 col 2 is a small rubber object; row 1 col 3 is a large rubber object; row 2 col 1 is a small metal object; row 2 col 2 is a small metal object; row 2 col 3 is a large metal object; row 3 col 1 is a small metal object; row 3 col 2 is a small metal object; I am missing one object at row 2 col 2. Can you find an object in the scene that can fit there?"*

D.4 RefCOCO-Adversarial (Ref-Adv)

To test performance in real-world scenarios, we use the **RefCOCO-Adversarial** benchmark [1]. This dataset consists of real-world images with visually similar distractor objects, making the task of Referring Expression Grounding particularly challenging. REG requires the model to locate the specific object corresponding to a text description, and we evaluate the model using Grounding Accuracy (%), where a prediction is considered correct if the Intersection over Union (IoU) with the ground-truth bounding box is 0.5 or greater.

D.5 Ref-GTA

To measure generalization under a significant domain shift, we use the **Ref-GTA** benchmark [27]. This is a REG benchmark where images are sourced from a photo-realistic game simulation, creating an out-of-distribution challenge for models pre-trained on natural images. Performance is measured by Grounding Accuracy (%), defined by an Intersection over Union (IoU) threshold of 0.5 between the predicted and ground-truth bounding boxes.

E Verification

To recover from occasional symbolic execution failures, we add a lightweight *verification* stage that operates after reasoning has produced a candidate answer. We explore two simple strategies:

1. **Pairwise Arbiter:** The model is presented with two candidates, the backbone prediction and the symbolic reasoning output, along with the query. It acts as an arbiter, directly judging which option better satisfies the description. Our prompt, similar to the one used by NAVER for this process, is shown in Figure 6.
2. **Confidence Gating:** We compute a softmax distribution over the symbolic executor's scores with temperature T, and use a threshold τ to decide whether to trust the executor.

You're an image analyst designed to check if the highlighted objects in the image meets the query description, and which one is more likely to meet the query description.
The query is: "{query}"
Please check the highlighted object "0" [in the red bounding box] and "1" [in the green bounding box] in the image and answer the question: Which object is more likely to meet the query description? Your answer should be "0", "1". Answer with one word or phrase.

Figure 6: Prompt used for the Pairwise Arbiter verification strategy.

If $\max(\text{softmax}) < \tau$, we fallback to the backbone prediction; otherwise, we keep the executor's output. Both τ and T are tuned on a 500-example held-out set for each backbone.

Table 15 summarizes the performance of both light-weight verification strategies across different backbones. Overall, we observe that the *Pairwise Arbiter* tends to outperform the *Confidence Gating* method, highlighting the benefit of leveraging a VLM to arbitrate between backbone and symbolic predictions.

Table 15: Post-verification accuracy on Ref-Adv.

Backbone VLM	Verification	Accuracy (%)	τ	T	Symbolic Share(%)
Qwen2VL-7B	Confidence Gating	80.91	0.70	0.40	73.59
	Pairwise Arbiter	80.93	N/A	N/A	80.23
Ovis1.6-9B	Confidence Gating	60.49	0.30	0.10	86.79
	Pairwise Arbiter	63.25	N/A	N/A	29.73
InternVL2.5-8B	Confidence Gating	76.65	0.60	0.50	71.06
	Pairwise Arbiter	78.08	N/A	N/A	77.86

You are an automatic answer checker! I will give you a question and answer, and a generated response, and you will tell me if the response is correct given the ground truth answer. If the response is correct, you will say <Yes>, otherwise you will say <No>. Rubber and Matte are the same. Shiny and Metal or Metallic are the same. Square and Cube are the same. Yellow and Golden colors are similar in the images, too. Balls and spheres are similar. Check if the response is correct given the questions and the ground truth answer, deeply thinking about the context. Answer with <Yes> or <No> and don't mention them other than for the final answer. Important! Ignore the red bounding box reference in the generated response.
Question: "{Question}"
Ground Truth Answer: "{Answer}"
Generated Response: "{Prediction}"

Figure 7: Visual Question Answering judge LLM prompt.

F Hyperparameters

The fine-tuning hyperparameters for our experiments are detailed in Table 16. These settings are based on the configurations defined in our training script.

During our experiments, we found that applying LoRA fine-tuning to the vision backbone, in addition to the language model, was highly effective for domain adaptation. To illustrate, without any neuro-symbolic fine-tuning, the base NePTune framework achieved **56.49%** accuracy on Ref-GTA. In contrast, a standard fine-tuning approach on the base VLM only reached **4.19%** accuracy. This performance gap underscores the effectiveness of our neuro-symbolic method and justifies fine-tuning the vision components to achieve optimal results in novel environments.

Table 16: Hyperparameters for Fine-Tuning Experiments

Hyperparameter	Standard Fine-Tuning	NePTune Neuro-Symbolic Fine-Tuning on RefCOCO-Adv	NePTune Neuro-Symbolic Fine-Tuning on Ref-GTA
General Training			
Learning Rate (lr)	4×10^{-5}	4×10^{-5}	4×10^{-5}
Batch Size	1	1	1
Gradient Accumulation	4	4	4
Weight Decay	0.01	0.01	0.01
LR Scheduler	Cosine	Cosine	Cosine
Warmup Ratio	0.03	0.03	0.03
Loss Function	Cross-Entropy	Binary Cross-Entropy	Binary Cross-Entropy
LoRA Configuration			
Rank (r)	16 (LM), 8 (Vision)	16 (LM)	16 (LM), 8 (Vision)
Alpha (α)	32 (LM), 16 (Vision)	32 (LM)	32 (LM), 16 (Vision)
Dropout	0.05	0.05	0.05

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims are supported by extensive experiments presented in the paper. For instance, the results in Table 8 show strong generalization on a challenging domain-shift benchmark, directly validating the contributions outlined in our abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Section 5 ("Discussion"). We address the framework's sensitivity to the choice of VLM, its computational cost.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.

- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: There is no theory assumptions in our work.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided sufficient detail for our results to be reproduced. Our methodology is described in Section 3, and full details regarding the experimental setup and all hyperparameters can be found in Appendices B and C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case

of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Inlcuded in the submission

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: specify all necessary training and test details for our experiments. A complete account of the hardware, software, data splits, and hyperparameters is provided in Appendices B and C.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report error bars for key results (e.g., Tables 8 and 9) to account for statistical variance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide sufficient information on the compute resources used. The hardware is detailed in Appendix B, and execution time metrics are provided in the "Computational Cost" discussion in Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: To the best of our knowledge, our research conforms to the NeurIPS Code of Ethics. We use standard public datasets and our work does not involve sensitive data collection or human subject research.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: Our paper does not contain a dedicated discussion of broader societal impacts. We have focused on the technical contributions of our work rather than potential negative applications.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This question is not applicable to our work. We introduce a framework rather than a new high-risk model or dataset that would require specific release safeguards.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have credited the creators of all existing assets via citation.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This question is not applicable, as we do not introduce or release any new assets such as datasets or models in this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This is not applicable to our research. We did not conduct any new crowdsourcing or experiments involving human subjects, relying instead on existing public datasets.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This is not applicable. Our research did not involve human subjects and therefore did not require IRB approval.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our paper describes using an LLM as a functional component within our methodology (Section 3.1) and as a tool for evaluation (Section 4.3). We do not, however, declare any use of LLMs for the ideation or generation of the core research concepts themselves.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.