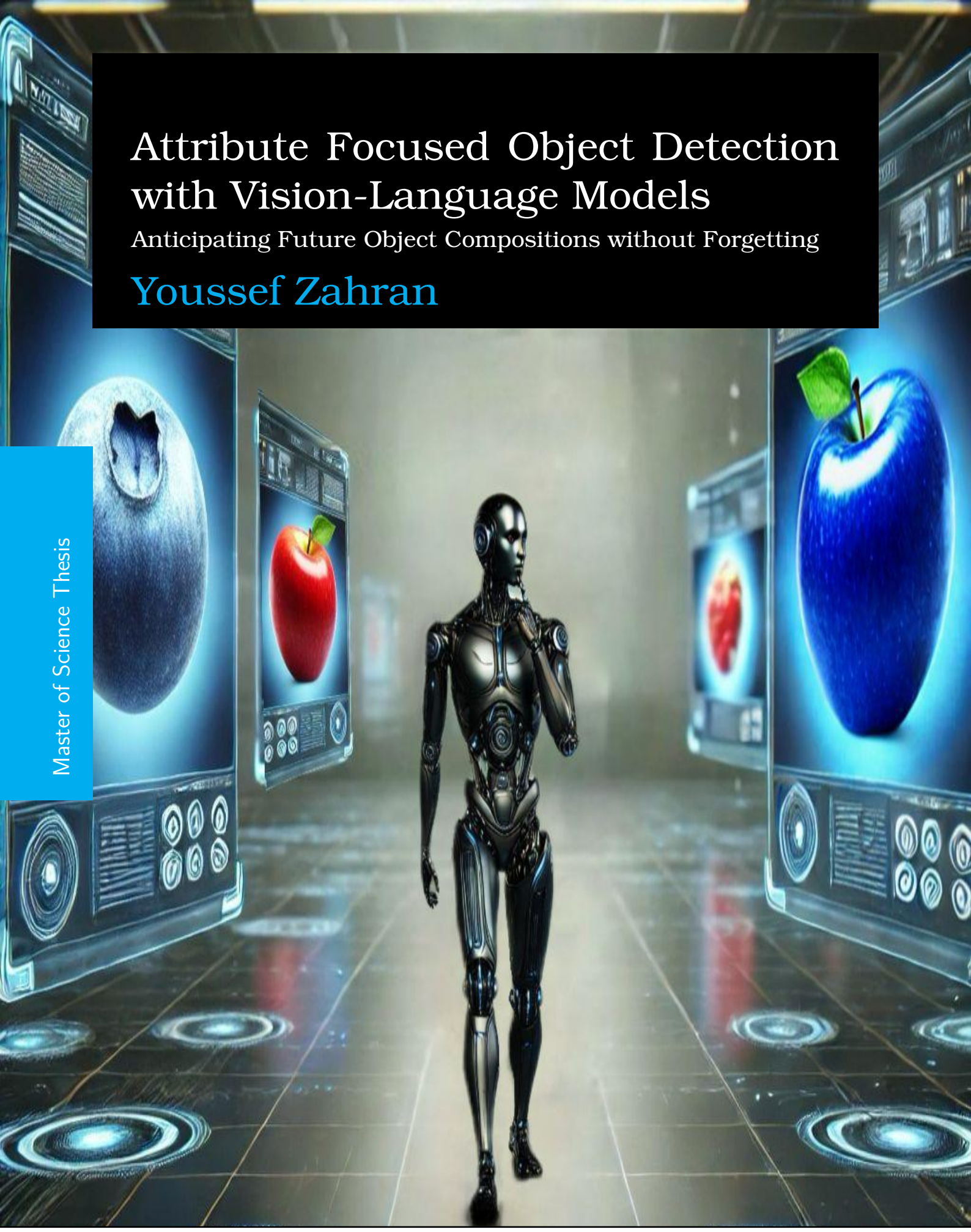
Attribute Focused Object Detection with Vision-Language Models

Anticipating Future Object Compositions without Forgetting

Youssef Zahran



Master of Science Thesis

TU Delft Delft University of Technology

TNO

Attribute Focused Object Detection with Vision-Language Models

Anticipating Future Object Compositions without Forgetting

Master of Science Thesis

For the degree of Master of Science in Robotics at Delft University of Technology

Youssef Zahran

Faculty of Mechanical Engineering (ME) · Delft University of Technology


TU Delft
Delft
University of
Technology


TNO

Delft University of Technology
Department of
Cognitive Robotics (CoR)

Attribute Focused Object Detection with Vision-Language Models

by

Youssef Zahran

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday July 19, 2024 at 9:00 AM.

Thesis committee:	Chair & daily supervisor	Dr.ir. Y.B. Eisma	TU Delft
	Daily Supervisor	Dr. G. Burghouts	TNO
	Committee member	Dr. Ir. J.C.F. de Winter	TU Delft
	Committee member	MSc. R. Zhang	TU Delft

Preface & Acknowledgements

I would like to thank my supervisors Dr. G. Burghouts and Dr.ir. Y.B. Eisma, for their support and valuable insights during the course of my thesis. Their constructive feedback and insightful ideas helped shape this thesis into what it is. Special thanks to Dr. G. Burghouts for making time in his busy schedule to hold weekly meetings with me. These meetings were helpful in keeping me on track, sometimes providing perspective to avoid overcomplicating issues. Next, I would like to thank my friends and everyone else who has helped me during this thesis and throughout my studies. I owe a special thanks to my mother, brother and sister for their unconditional support and love.

Delft, University of Technology — Youssef Zahran

Thesis

Anticipating Future Object Compositions without Forgetting

Youssef Zahran[1,2][0009−0008−5386−5530], Gertjan Burghouts[2][0000−0001−6265−7276], and Yke B. Eisma[1][0000−0003−3437−2761]

[1] Department of Cognitive Robotics, Delft University of Technology
[2] Intelligent Imaging Department, TNO

Abstract. Despite the significant advancements in computer vision models, their ability to generalize to novel object-attribute compositions remains limited. Existing methods for Compositional Zero-Shot Learning (CZSL) mainly focus on image classification. This paper aims to enhance CZSL in object detection without forgetting prior learned knowledge. We use Grounding DINO and incorporate Compositional Soft Prompting (CSP) into it and extend it with Compositional Anticipation. We achieve a 70.5% improvement over CSP on the harmonic mean (HM) between seen and unseen compositions on the CLEVR dataset. Furthermore, we introduce Contrastive Prompt Tuning to incrementally address model confusion between similar compositions. We demonstrate the effectiveness of this method and achieve an increase of 14.5% in HM across the pretrain, increment, and unseen sets. Collectively, these methods provide a framework for learning various compositions with limited data, as well as improving the performance of underperforming compositions when additional data becomes available.

Keywords: compositional zero-shot learning · prompt tuning · incremental learning.

1 Introduction

Although humans have never seen a blue apple, they can easily picture it. This is due to the inherent human ability to generalize to novel concepts by combining the known entity "apple" with the color "blue." However, do computer vision models possess this capability? This question has motivated the development of Compositional Zero-Shot Learning (CZSL) [5, 21, 22, 26]. In CZSL, the goal is to recognize unseen object-attribute combinations, referred to as compositions, based on the compositions seen during training. For this, models should understand the attributes and objects that compose these compositions to generalize to all possible compositions.

Vision Language Models (VLMs), pretrained on large-scale image-text pairs, are promising for CZSL due to their ability to understand the relationship between the visual content and the textual description [16, 22, 26]. For object detection, VLMs such as Grounding DINO [10] and GLIP [7] learn to associate

regions of text with regions of images by pulling the embeddings of paired image regions and text descriptions close while pushing others away [25]. These models perform cross-modality fusion throughout the whole architecture, which makes the textual features image-aware and the visual features text-aware. Liu et al. [10] argue that VLMs benefit from frequent cross-modality fusion, making Grounding DINO superior to GLIP. Therefore, throughout this paper, we will solely focus on Grounding DINO.

Unfortunately, these models tend to be biased towards object categories rather than attributes, which makes it suffer from feature misalignment when used directly for attribute recognition [3]. Fine-tuning VLMs can solve this but often leads to catastrophic forgetting of prior knowledge [28], thereby compromising their generalization ability. To address this, we explore how to fine-tune VLMs to perform well in CZSL without forgetting any prior knowledge. Nayak et al. [15] introduced Compositional Soft Prompting (CSP), which combats catastrophic forgetting by adding auxiliary tokens for all words in a given dataset and training only these tokens. CSP improves model performance in CZSL for image classification. We incorporate CSP in Grounding DINO, to leverage it for object detection.

We consider CSP as a baseline and improve it for CZSL by introducing Compositional Anticipation (CA). CA consists of two components: Compositional Smoothing and Compositional Independence. Compositional Smoothing anticipates novel object-attribute compositions by assigning soft labels when predictions are partially correct, e.g. the object is correct but the attribute is different. This approach deviates from conventional Label Smoothing [20], which assigns soft labels to all classes. Compositional Independence disentangles objects from attributes through Separation and Decorrelation. Separation introduces a separation loss to maximize the distinction between object and attribute classes by applying intra-class separation within objects and attributes and an inter-class separation between objects and attributes. Decorrelation minimizes the correlation between objects and attributes to reduce dependency between the two.

For incremental learning on newly added compositions, we use prior knowledge to address specific mistakes related to confusion between similar compositions. Inspired by recent developments in prompt tuning [17, 29, 30], we introduce a novel method called Contrastive Prompt Tuning, specifically tailored for object attributes. Contrastive Prompt Tuning addresses cases where the model confuses similar compositions, such as mistaking a blue apple for a red apple, by adding a trainable prompt in front of the confused class: "is not red apple but is blue apple". This approach utilizes our prior knowledge to harness the ability of a VLM to exploit language.

In summary, our main contributions are:

1. We incorporate CSP [15] into Grounding DINO [10] and extend it with Compositional Anticipation. Compositional Anticipation consists of:
   - Compositional Smoothing, which assigns soft labels when predictions are partially correct.
   - Compositional Independence, which disentangles objects from attributes.

2. We develop Contrastive Prompt Tuning, a method that adds a learnable prompt for compositions that are confused with each other during training. This technique harnesses the power of language and our understanding of the model to improve performance beyond simply training with additional data.

2 Related Work

In this section, we review literature related to our work. We cover Compositional Zero-Shot Learning (CZSL), Prompt Tuning in Vision-Language Models (VLMs), and Class Incremental Learning (CIL). Our research focuses on improving the CZSL capabilities of Grounding DINO [10], a VLM designed for object detection, by utilizing prompt tuning. Additionally, we address underperforming compositions in a class-incremental manner to further improve model performance.

Compositional Zero-Shot Learning The main objective of CZSL is to recognize unseen compositions from the compositions encountered during training. In CZSL, individual objects and attributes are referred to as primitives. Misra et al. [13] use a limited set of compositions to learn linear classifiers for each primitive. Then, they learn a transformation network that takes these classifiers as input and composes them to produce a classifier for their combination. Since then, multiple works [8, 12, 14, 19] have been proposed to tackle the CZSL task.

Recent works focus on adapting pretrained VLMs for CZSL by fine-tuning primitive tokens. While CSP [15] only trains these tokens, others [11, 21] also introduce prompt disentangled tuning. This technique addresses entanglement, where optimizing one primitive's embedding affects another. Prompt disentangled tuning divides the process into three phases with different prompts: one for the entire composition, one for the attribute, and one for the object. This ensures attributes and objects learn their optimal parameters independently.

While [11, 21] improve upon [15] with an average performance increasement of 1.7% and 2.3%, respectively, the gains are marginal relative to the increased complexity. Our work is closely related to [11, 15] as we adapt CSP for object detection and address entanglement through Compositional Independence.

Prompt Tuning in VLMs Ever since CLIP [16] demonstrated that prompt templates such as "a photo of a [CLASS]" improve the results of VLMs compared to using only the classname, several other works [17, 24, 29, 30] have been introduced to replace the hand-crafted prompt with learnable soft prompts. CoOP [30] introduces soft prompts that are shared across all classes, resulting in prompts like $[v_1], [v_2], \dots [v_M]$ for all images. CoCoOp [29] improves upon this by proposing soft prompts that are image-conditioned, generating prompts such as $[v_1(x)], [v_2(x)], \dots [v_M(x)]$ for each image x. Building upon these advancements, Rao et al. [17] use contextual information from the image to prompt the language model.

Our work is closely related to these works but is unique in its focus on improving the performance of confused compositions using learnable prompts that are initialized based on our knowledge of the model's errors.

Class Incremental Learning Class-Incremental Learning (CIL) refers to learning new classes while retaining previously learned classes [27, 28]. In typical CIL scenarios, learning occurs through a sequence of training tasks, each of which introduce new classes without any overlap of the classes from previous tasks. The main challenge is avoiding catastrophic forgetting, where learning new classes leads to a loss of knowledge from previous tasks. Our approach bears resemblance to Blurry CIL [1, 2], where former classes can be revisited during training. Similarly, we train incrementally with underperforming compositions while allowing former compositions to be revisited.

3 Method

3.1 Problem Definition

Compositional Zero-Shot Learning We follow [21, 22, 26] and formalize the CZSL task as follows. Let $\mathcal{A}$ denote the set of attributes, and $\mathcal{O}$ the set of objects, and $\mathcal{C} = \mathcal{A} \times \mathcal{O}$ the set of all compositions. $\mathcal{T} = \{(x_j, c_j)\}_{j=1}^{N}$ denotes the train set where $x_j \in \mathcal{X}$ is a sample in the input (image) space $\mathcal{X}$ and $c_j \in \mathcal{C}_s$ is a composition in the subset $\mathcal{C}_s \subseteq \mathcal{C}$. The seen set $\mathcal{C}_s \subseteq \mathcal{C}$ consists of all compositions encountered during training, whereas the unseen set $\mathcal{C}_u \subseteq \mathcal{C}$ consists of compositions not seen during training. Let $\mathcal{C}_s$ and $\mathcal{C}_u$ be two sets such that $\mathcal{C}_s \cap \mathcal{C}_u = \emptyset$. While $\mathcal{C}_s$ and $\mathcal{C}_u$ are disjoint, the objects $\mathcal{O}_u$ and attributes $\mathcal{A}_u$ are defined such that $\mathcal{O}_u \subseteq \mathcal{O}_s$ and $\mathcal{A}_u \subseteq \mathcal{A}_s$

Catastrophic Forgetting VLMs, such as Grounding DINO [10], are known for their ability to generalize well across diverse tasks due to the extensive and varied data used during pre-training. However, fine-tuning these models on a new dataset often compromises their generalization capability, as the rich features learned during pre-training are replaced by features specific to the new dataset. This can lead to catastrophic forgetting, where the model's performance on previously learned tasks significantly deteriorates. In the context of CZSL with VLMs, catastrophic forgetting is particularly problematic. While the model may perform well on the specific compositions present in the new dataset it was fine-tuned on, it risks becoming overly specialized. This specialization may result in a model that loses its ability to generalize to other compositions, objects, or concepts, and instead becomes exceptionally good at predicting the compositions seen during training. Such a limitation is especially undesirable for open-set object detectors like Grounding DINO, which are meant to recognize a wide range of concepts.

3.2 Incremental CZSL

In practical settings, models often encounter new data or need to improve performance on underperforming compositions after the initial training phase. To address this, we introduce an increment set $\mathcal{C}_i \subseteq \mathcal{C}$ to CZSL. Let $\mathcal{C}_p$ be the set used for the initial fine-tuning of the model, with $\mathcal{C}_p = \mathcal{C}_s$. After introducing $\mathcal{C}_i$, the set of seen compositions becomes $\mathcal{C}_s = \mathcal{C}_p \cup \mathcal{C}_i$. The increment set $\mathcal{C}_i$ consists of compositions introduced after the initial fine-tuning to improve performance on underperforming compositions. Improving these underperforming compositions with additional compositions is challenging because $\mathcal{C}_p$ is designed to cover $\mathcal{A}$ and $\mathcal{O}$ with the minimum number of compositions. Extending $\mathcal{C}_s$ with $\mathcal{C}_i$ makes the attributes and objects in $\mathcal{C}_i$ overrepresented in $\mathcal{C}_s$, which can bias the model towards these attributes and objects. In this paper, we focus solely on improving performance on compositions $c_j \in \mathcal{C}$ without extending the attribute set $\mathcal{A}$ or the object set $\mathcal{O}$.

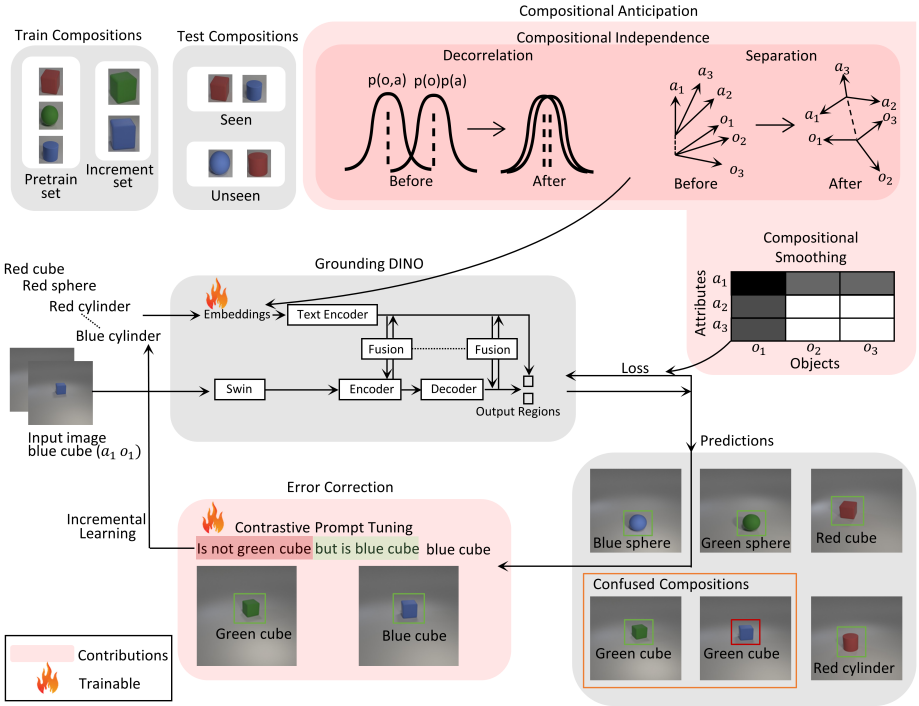


Fig. 1: Our method anticipates unseen, future object-attribute compositions through Compositional Independence and Compositional Smoothing. Forgetting is mitigated by creating auxiliary tokens for the language embeddings and refining only these tokens. Errors in compositions are incrementally corrected using Contrastive Prompt Tuning, which contrasts confused compositions.

3.3 Compositional Soft Prompting

To prevent catastrophic forgetting in Grounding DINO [10], we follow CSP [15] and modify it for object detection. Objects and attributes that form compositions are treated as learnable tokens within the VLMs vocabulary. Each attribute $a_j \in \mathcal{A}$ and each object $o_j \in \mathcal{O}$ is represented as an auxiliary token t_{a_j} and t_{o_j} respectively, where $t_{a_j}, t_{o_j} \in \mathbb{R}^d$, with d being the dimension of the vocabulary embedding. During training, only these auxiliary tokens are tuned, resulting in $(|\mathcal{A}| + |\mathcal{O}|) \times d$ learnable parameters.

To illustrate, CSP creates auxiliary tokens for each attribute and object such as t_{blue} for the attribute "blue" and t_{apple} for the object "apple". These tokens are adjusted during training while the rest of the weights, such as those of the encoder and decoder in Grounding DINO, remain unchanged. By doing this, CSP prevents catastrophic forgetting and preserves the pretrained weights of the model.

3.4 Compositional Smoothing

To combat bias during training for $\mathcal{C}_s$, where the model becomes overly confident with the seen classes, we assign soft labels rather than hard labels (0 and 1) in the classification loss. This is referred to as Label Smoothing [20], and it prevents the model from becoming overly confident in its predictions, thereby improving its generalization capability. Conventional Label Smoothing adjusts the target labels by distributing a small portion of the probability mass to all other labels. For a given true label y in a classification problem with k classes, the smoothed label y_{smooth} is defined as:

$$y_{\text{smooth}} = (1 - \epsilon)y + \frac{\epsilon}{k}, \tag{1}$$

where ϵ is the smoothing parameter, and the term $\frac{\epsilon}{k}$ distributes the smoothing equally among all classes.

We deviate from conventional Label Smoothing [20] and assign soft labels based on the correctness of the object, attribute, or the entire composition. We refer to this as Compositional Smoothing. Let $p_{\mathcal{O}}$, $p_{\mathcal{A}}$, and $p_{\mathcal{C}}$ represent the probabilities for object, attribute, and overall composition predictions, respectively. For a given true composition c_t composed of object o_t and attribute a_t, and predicted composition c_p composed of object o_p and attribute a_p, the smoothed label $y_{\text{o,a}}$ is defined as:

$$y_{\text{o,a}} = \begin{cases} p_{\mathcal{O}} & \text{if } o_p = o_t \wedge a_p \neq a_t, \\ p_{\mathcal{A}} & \text{if } a_p = a_t \wedge o_p \neq o_t, \\ p_{\mathcal{C}} & \text{if } a_p = a_t \wedge o_p = o_t, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Compositional Smoothing ensures that there is a difference between having partial correctness and no correctness in the prediction, guiding the model to learn

what the compositions are composed of rather than learning the compositions themselves. This, in turn, should lead to better performance on $\mathcal{C}_u$.

Figure 1 (top right) illustrates Compositional Smoothing. For a given ground truth label (a_1, o_1), predictions where both the attribute and object are correct are shown in black, and the smoothed label becomes $p_{\mathcal{C}}$. Partial correctness is depicted in gray, with the smoothed label being either $p_{\mathcal{O}}$ or $p_{\mathcal{A}}$. When both the object and attribute are completely wrong, no smoothing is applied.

3.5 Compositional Independence

In CZSL, it is important to disentangle objects from attributes and have clear distinctions withing each category. For example, a cube and a cylinder should be easily distinguishable to prevent confusion. Additionally, colors should be distinguished from specific objects, such as cubes, to ensure their independence. This prevents similar attributes or objects to be confused with each other and helps the model treat attributes and objects as distinct concepts.

We achieve this independence through two components: Separation and Decorrelation. Separation enforces orthogonality within the embeddings of objects and attributes and maximizes the distance between their mean embeddings. Decorrelation minimizes the correlation between the embeddings of objects and attributes. This is achieved using the Hilbert-Schmidt Independence Criterion (HSIC) [4], a kernel statistical test commonly used to measure independence between two random variables, which proved to be effective for CZSL image classification [18] and is leveraged here for object detection.

Separation To help the model differentiate between similar attributes or objects, we introduce an orthogonality loss. We achieve orthogonality within the groups of attributes and objects by minimizing the average absolute similarity between the normalized embeddings within each group:

$$\mathcal{L}_{\text{orth}}(\mathbf{E}) = \frac{1}{|\mathbf{E}|^2 - |\mathbf{E}|} \sum_{i=1}^{|\mathbf{E}|} \sum_{\substack{j=1 \\ j \neq i}}^{|\mathbf{E}|} |\mathbf{e}_i \cdot \mathbf{e}_j| \tag{3}$$

where $\mathbf{E}$ is the set of normalized embeddings, and $\mathbf{e}_i$ and $\mathbf{e}_j$ are embeddings within this set. The summation $\sum_{\substack{j=1 \\ j \neq i}}$ ignores self-similarity, and $\frac{1}{|\mathbf{E}|^2 - |\mathbf{E}|}$ ensures that self-similar terms are excluded during normalization. This orthogonality loss is applied to both the attributes and objects:

$$\mathcal{L}_{\mathcal{A}} = \mathcal{L}_{\text{orth}}(\mathbf{E}_{\mathcal{A}}) \tag{4}$$

$$\mathcal{L}_{\mathcal{O}} = \mathcal{L}_{\text{orth}}(\mathbf{E}_{\mathcal{O}}) \tag{5}$$

where $\mathbf{E}_{\mathcal{A}}$ and $\mathbf{E}_{\mathcal{O}}$ represent the sets of normalized embeddings for the attributes and objects, respectively.

Additionally, to enforce a clear distinction between attributes and objects, we ensure that the mean embeddings of attributes and objects are significantly separated:

$$\mathcal{L}_{\text{distance}} = -\log(\|\mu_{\mathcal{A}} - \mu_{\mathcal{O}}\|_2) \tag{6}$$

where $\mu_{\mathcal{A}} = \frac{1}{|\mathcal{A}|}\sum_{j=1}^{|\mathcal{A}|}\mathbf{e}_{\mathcal{A}_j}$ and $\mu_{\mathcal{O}} = \frac{1}{|\mathcal{O}|}\sum_{j=1}^{|\mathcal{O}|}\mathbf{e}_{\mathcal{O}_j}$ represent the mean embeddings of attributes and objects, respectively. The distance is computed using the L_2 norm between the mean embeddings of the two groups.

The total Separation loss is a weighted combination of the orthogonality and mean separation components:

$$\mathcal{L}_{\text{separation}} = \lambda_1\mathcal{L}_{\text{distance}} + \lambda_2\mathcal{L}_{\mathcal{A}} + \lambda_3\mathcal{L}_{\mathcal{O}} \tag{7}$$

where λ_1,λ_2 and λ_3 are hyperparameters controlling the contribution of $\mathcal{L}_{\text{distance}}$, $\mathcal{L}_{\mathcal{A}}$ and $\mathcal{L}_{\mathcal{O}}$ to the final loss, respectively.

Decorrelation To further ensure the independence between object and attribute embeddings, we introduce Decorrelation by using HSIC [4]. For an object o_j with attribute a_j, we formulate the HSIC loss as follows:

$$\mathcal{L}_{\text{hsic}} = \lambda_h\text{HSIC}(o_j, a_j) \tag{8}$$

Here, λ_h is a hyperparameter that controls the contribution of the HSIC term to the total loss.

3.6 Compositional Anticipation

Our method, which we refer to as Compositional Anticipation (CA), consists of both Compositional Smoothing and Compositional Independence. Figure 1 shows how we implement CA in Grounding DINO [10].

3.7 Contrastive Prompt Tuning

To improve the performance of some underperforming composition after training with $\mathcal{C}_p$, we extend $\mathcal{C}_s$ with an additional set $\mathcal{C}_i$ to improve performance. Our approach begins with analyzing the predictions to identify compositions that are frequently confused with each other. For instance, if c_j and c_k are often mixed-up, both compositions are included in $\mathcal{C}_i$, and a trainable prompt is added in front of the underperforming class(es). For example, if c_j performs poorly, we add the following prompt in front of the class: *"is not c_k but is c_j"*. This prompt contains both a negative and an affirmative component.

We refer to this method as Contrastive Prompt Tuning and it does not modify any of the tokens present in the sets $\mathcal{A}$ and $\mathcal{O}$. Instead, it focuses solely on the learnable prompt, which leads to fewer changes in the performance of other compositions and mitigates catastrophic forgetting. By doing this, we exploit the ability of a VLM to understand language and use a semantically meaningful initial prompt to learn to distinguish between similar compositions. This step is depicted as Incremental Learning in Figure 1.

4 Experiments

4.1 Evaluation

Dataset We evaluate our approach using a synthetic dataset generated following the CLEVR framework [6]. This dataset consists of three types of objects: cube, cylinder, and sphere. Each object is associated with six attributes: blue, red, green, purple, brown, and yellow.

The dataset intentionally excludes non-visual attributes (e.g., heavy) and attributes that exhibit significant variation across different objects (e.g., wet in wet dog versus wet car). This yields a dataset that is reliable for assessing a model's performance in the CZSL task. Given that there are no ambiguous attributes present in this dataset, a poorly performing model would indicate that the model is bad in the CZSL task.

Train-Test Split Throughout this section, all experiments for the CZSL task are trained using the set: {red cube, blue cube, green sphere, purple sphere, brown cylinder, yellow cylinder} as $\mathcal{C}_p$ with 10 shots per composition. This split ensures that $\mathcal{C}_s$ covers the entire set of objects $\mathcal{O}$ and attributes $\mathcal{A}$. Testing is performed with the whole set of composition $\mathcal{C}$ with 60 samples per composition.

Evaluation Metric We adopt the NMS mAP evaluation metric introduced by Yoa et al. [23]. In this work they argued that the traditional COCO mAP [9] is deceiving for open vocabulary detection models, such as Grounding DINO [10]. Consider an image annotated with two ground-truth instances: a purple cylinder and a green cylinder, assuming these are the only cylinder categories in the model. These models tend to be able to detect and locate the presence of all cylinders in the image, but they struggle with the contextual description. They would predict two overlapping bounding boxes for each object, mistakenly assigning both 'green' and 'purple cylinder' labels to each object. All four of these boxes would be predicted with a high confidence score. Additionally, the highest scoring label is not necessarily the correct one. Consequently, the AP for each category would misleadingly be 0.50, despite the model failing to correctly comprehend the target objects. Yao et al. [23] refer to this as the 'inflated AP problem'.

To address this issue, Yao et al. [23] propose applying class-agnostic Non-Maximum Suppression (NMS) before calculating the mAP. This method suppresses redundant bounding boxes, ensuring that only the prediction with the highest confidence score is used in the calculation of the mAP. We adopt this NMS mAP metric to provide a more realistic measure of our model's performance.

4.2 CSP base

We adapt CSP [15] and modify it for Grounding DINO [10] and this integration serves as our baseline method. To assess its performance, we begin by training

Table 1: Compositional Anticipation improves both object detection performance and generalization to unseen compositions. Compositional Smoothing contributes the most to these improvements, followed by Separation and Decorrelation.

Compositional Anticipation (CA)					
Compositional Smoothing	Separation	Decorrelation	Seen	Unseen	HM
✗	✗	✗	81.4 ± 7.6	4.5 ± 4.6	8.0 ± 8.1
✗	✗	✓	81.3 ± 7.7	10.8 ± 6.7	18.2 ± 10.5
✗	✓	✗	82.5 ± 6.8	15.1 ± 4.2	25.4 ± 6.1
✗	✓	✓	84.4 ± 7.1	20.8 ± 4.7	33.1 ± 6.4
✓	✗	✗	86.2 ± 6.8	64.3 ± 5.9	73.5 ± 5.3
✓	✗	✓	92.4 ± 3.0	61.6 ± 5.7	73.8 ± 4.5
✓	✓	✗	86.0 ± 6.1	67.7 ± 4.7	75.7 ± 5.0
✓	✓	✓	88.7 ± 4.9	70.6 ± 7.4	**78.5 ± 6.0**

Table 2: Our model does not forget. It achieves good performance on the fine-tuned CLEVR [6] dataset while preserving performance on MS-COCO [9], whereas conventional fine-tuning of Grounding DINO [10] leads to forgetting on MS-COCO.

	CLEVR [6]		MS-COCO [9]	
Model	Before	After	Before	After
Grounding DINO [10]	23.4	91.0 ↑67.6	41.1	11.8 ↓29.3
+ CSP [15] + CA (ours)		76.6 ↑53.2		41.1 =0.0

it with $\mathcal{C}_p = \mathcal{C}$. This yields an NMS mAP of 87.2 ± 6.8, demonstrating that good performance can be achieved by only training the embeddings of $\mathcal{O}$ and $\mathcal{A}$.

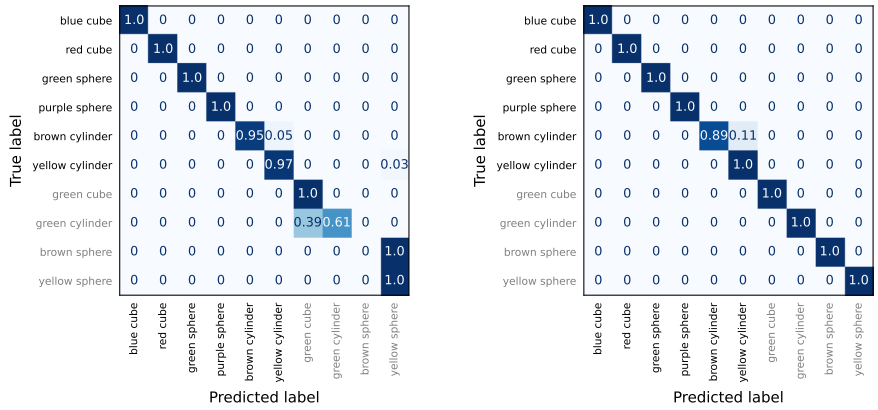
4.3 CZSL Comparison

We compare the CSP [15] baseline with our proposed method, Compositional Anticipation (CA) which extends CSP with Compositional Independence and Compositional Smoothing. The results, averaged over 13 experimental runs, are shown in Table 1 and are denoted using the NMS mAP metric [23]. Our results show that our method substantially improves upon the CSP baseline, with the harmonic mean (HM) between seen and unseen compositions improving by 70.5%. This improvement is predominately achieved on the unseen compositions, which improved by 66.1%.

Additionally, we showcase that our method does not suffer from catastrophic forgetting by evaluating its generalization ability compared to the conventional fine-tuning of Grounding DINO [10]. We compare the results on the MS-COCO [9] dataset before and after fine-tuning on the CLEVR [6] dataset for the CZSL task. Table 2 shows that conventional fine-tuning of Grounding DINO [10] achieves

Table 3: Our Contrastive Prompt Tuning is effective for incremental learning. It improves performance across all classes including the unseen ones.

Method	Tunable Tokens	Pretrained	Increment	Unseen	HM
Class-Specific Tuning					
CSP	$\mathcal{O}+\mathcal{A}$	88.1 ± 4.5 $^{\downarrow 2.9}$	72.9 ± 20.3 $^{\uparrow 13.6}$	0.0 ± 0.0 $^{\downarrow 74.3}$	0.0 ± 0.0 $^{\downarrow 71.7}$
+ CA	$\mathcal{O}+\mathcal{A}$	80.9 ± 5.0 $^{\downarrow 10.1}$	60.5 ± 18.2 $^{\uparrow 1.2}$	76.5 ± 8.4 $^{\uparrow 2.1}$	69.6 ± 7.0 $^{\downarrow 2.1}$
CSP	$\mathcal{O}_i+\mathcal{A}_i$	89.0 ± 2.3 $^{\downarrow 2.0}$	64.1 ± 20.9 $^{\uparrow 4.8}$	69.4 ± 8.3 $^{\downarrow 4.9}$	70.6 ± 8.2 $^{\downarrow 1.1}$
+ CA	$\mathcal{O}_i+\mathcal{A}_i$	89.2 ± 3.9 $^{\downarrow 1.8}$	62.4 ± 19.6 $^{\uparrow 3.1}$	78.6 ± 4.9 $^{\uparrow 4.3}$	73.3 ± 9.1 $^{\uparrow 1.5}$
Compositional Prompt Tuning (ours)					
Affirmation	Prompt	88.8 ± 5.5 $^{\downarrow 2.2}$	82.8 ± 17.8 $^{\uparrow 23.5}$	74.8 ± 6.6 $^{\uparrow 0.5}$	80.2 ± 7.9 $^{\uparrow 8.5}$
Negation	Prompt	91.2 ± 2.0 $^{\uparrow 0.2}$	81.9 ± 13.8 $^{\uparrow 22.6}$	76.6 ± 5.8 $^{\uparrow 2.3}$	82.1 ± 6.0 $^{\uparrow 10.4}$
Both	Prompt	92.6 ± 1.5 $^{\uparrow 1.6}$	93.7 ± 2.1 $^{\uparrow 34.4}$	75.2 ± 5.0 $^{\uparrow 0.9}$	**86.2 ± 2.1** $^{\uparrow 14.5}$



(a) Before incremental learning (b) After incremental learning

Fig. 2: Our Contrastive Prompt Tuning method is effective in incremental learning. It improves performance on the increment compositions (in gray) while preserving performance of the pretrained compositions (in black).

a 67.6% improvement on CLEVR, whereas our method achieves a 53.2% improvement. This suggests that conventional fine-tuning of Grounding DINO is superior in CZSL. However, conventional fine-tuning of Grounding DINO leads to a 29.3% performance drop on MS-COCO, whereas our method maintains stable performance with no drop at all. This demonstrates that conventional fine-tuning suffers from catastrophic forgetting, while our method does not.

4.4 Improving Incrementally

In this experiment, we incrementally learn new classes using the model initially trained with CSP [15] extended with Compositional Anticipation. We continue training the model using a dataset that includes both $\mathcal{C}_p$ and $\mathcal{C}_i$.

We explore two different fine-tuning methods: fine-tuning class-specific tokens and our proposed method, Contrastive Prompt Tuning. For fine-tuning class-specific tokens, we compare CSP [15] with CSP extended with Compositional Anticipation. Additionally, we conduct this fine-tuning in two ways: (1) allowing the fine-tuning of all tokens in the sets $\mathcal{O}$ and $\mathcal{A}$, and (2) fine-tuning only objects and attributes present in $\mathcal{C}_i$, specifically $\mathcal{O}_i$ and $\mathcal{A}_i$. For Contrastive Prompt Tuning, all tokens are frozen and only the prompt is fine-tuned. The prompt is initialized with semantically meaningful information, including both an affirmative and a negative component. For instance, if "green cylinder" is often confused with "green cube", the prompt is initialized as "is not green cube but is green cylinder". We also analyze the individual contributions of each component to the overall performance enhancements.

To evaluate performance, we compare the model's results before and after introducing $\mathcal{C}_i$. Specifically, we determine performance across the sets $\mathcal{C}_p$, $\mathcal{C}_i$, and $\mathcal{C}_u$. Initially, $\mathcal{C}_u$ is defined as $\mathcal{C} - \mathcal{C}_p$. After introducing $\mathcal{C}_i$, $\mathcal{C}_u$ becomes $\mathcal{C} - \mathcal{C}_p - \mathcal{C}_i$. The results on these sets after introducing $\mathcal{C}_i$ are shown in Table 3, with the absolute changes compared to the initial values indicated with arrows.

Our results show that our method, Contrastive Prompt Tuning, which fine-tunes a prompt initialized with prior knowledge to address specific mistakes related to confusion between similar compositions, is superior to the class-specific tuning strategy. With Contrastive Prompt Tuning, we achieve a 12.9% enhancement in the HM across the pretrain, increment, and unseen sets compared to the best class-specific tuning method. This improvement is predominately achieved across the increment set, which improved by 31.3% compared to the best class-specific tuning method. Furthermore, Contrastive Prompt Tuning benefits from both the affirmative and negative components of the prompt.

Figure 2 shows the effects of Compositional Prompt Tuning on the model's predictions. Figure 2a shows that before incremental learning, 39% of all instances of "green cylinder" are misclassified as "green cube", and all instances of "brown sphere" are misclassified as "yellow sphere". Figure 2b demonstrates that after applying Compositional Prompt Tuning, "green cube," "green cylinder," "yellow sphere," and "brown sphere" are classified correctly on all instances.

5 Conclusion

In this paper, we demonstrated that conventional fine-tuning of Grounding DINO achieves an NMS mAP of 91.0 when fine-tuned on the CLEVR dataset for CZSL. However, this approach suffers from catastrophic forgetting, as confirmed by a 29.3% decrease in performance on the MS-COCO dataset post fine-tuning. To address this, we proposed incorporating CSP into Grounding DINO to mitigate forgetting by only fine-tuning auxiliary tokens. However, we observed that using CSP alone resulted in an NMS mAP of only 8.0 for the HM between seen and unseen compositions. Therefore, we extended CSP with Compositional Anticipation, which improved the HM by 70.5%. While our method improves upon the CSP baseline, it does not surpass conventional fine-tuning of Grounding

DINO. Additionally, we introduced Contrastive Prompt Tuning to incrementally improve compositions that are confused with each other during training. With Contrastive Prompt Tuning, we improve performance on the HM across the pre-train, increment, and unseen sets by 12.9% compared to the best class-specific tuning method.

Given these findings, we recommend conventional fine-tuning of Grounding DINO for applications where performance on a specific dataset is prioritized, and our method for scenarios emphasizing overall performance across datasets.

Considering that Grounding DINO excels in CZSL, likely due to the cross-modality fusion between image and text embeddings and our proposed methods involve anticipating how the embeddings should be positioned in the embedding space. Having demonstrated the benefits of our approach for CZSL, further investigation into the positioning of embeddings in the fused embedding space could potentially yield results approaching those achieved by conventional fine-tuning of Grounding DINO, but without encountering catastrophic forgetting.

References

1. Bang, J., Kim, H., Yoo, Y., Ha, J.W., Choi, J.: Rainbow memory: Continual learning with a memory of diverse samples. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8218–8227 (2021)
2. Bang, J., Koh, H., Park, S., Song, H., Ha, J.W., Choi, J.: Online continual learning on a contaminated data stream with blurry task boundaries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9275–9284 (2022)
3. Chen, K., Jiang, X., Hu, Y., Tang, X., Gao, Y., Chen, J., Xie, W.: Ovarnet: Towards open-vocabulary object attribute recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 23518–23527 (June 2023)
4. Gretton, A., Fukumizu, K., Teo, C., Song, L., Schölkopf, B., Smola, A.: A kernel statistical test of independence. Advances in neural information processing systems **20** (2007)
5. Huang, S., Wei, Q., Wang, D.: Reference-limited compositional zero-shot learning. In: Proceedings of the 2023 ACM International Conference on Multimedia Retrieval. pp. 443–451 (2023)
6. Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., Girshick, R.: Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2901–2910 (2017)
7. Li, L.H., Zhang, P., Zhang, H., Yang, J., Li, C., Zhong, Y., Wang, L., Yuan, L., Zhang, L., Hwang, J.N., et al.: Grounded language-image pre-training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10965–10975 (2022)
8. Li, X., Yang, X., Wei, K., Deng, C., Yang, M.: Siamese contrastive embedding network for compositional zero-shot learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9326–9335 (2022)

9. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
10. Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J., et al.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection. arXiv preprint arXiv:2303.05499 (2023)
11. Lu, X., Liu, Z., Guo, S., Guo, J., Huo, F., Bai, S., Han, T.: Drpt: Disentangled and recurrent prompt tuning for compositional zero-shot learning. arXiv preprint arXiv:2305.01239 (2023)
12. Mancini, M., Naeem, M.F., Xian, Y., Akata, Z.: Open world compositional zero-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5222–5230 (June 2021)
13. Misra, I., Gupta, A., Hebert, M.: From red wine to red tomato: Composition with context. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1792–1801 (2017)
14. Nagarajan, T., Grauman, K.: Attributes as operators: factorizing unseen attribute-object compositions. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 169–185 (2018)
15. Nayak, N.V., Yu, P., Bach, S.H.: Learning to compose soft prompts for compositional zero-shot learning. arXiv preprint arXiv:2204.03574 (2022)
16. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
17. Rao, Y., Zhao, W., Chen, G., Tang, Y., Zhu, Z., Huang, G., Zhou, J., Lu, J.: Denseclip: Language-guided dense prediction with context-aware prompting. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 18082–18091 (2022)
18. Ruis, F., Burghouts, G., Bucur, D.: Independent prototype propagation for zero-shot compositionality. Advances in Neural Information Processing Systems **34**, 10641–10653 (2021)
19. Saini, N., Pham, K., Shrivastava, A.: Disentangling visual embeddings for attributes and objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13658–13667 (2022)
20. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
21. Wang, H., Yang, M., Wei, K., Deng, C.: Hierarchical prompt learning for compositional zero-shot recognition. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. pp. 1470–1478 (2023)
22. Wang, Q., Liu, L., Jing, C., Chen, H., Liang, G., Wang, P., Shen, C.: Learning conditional attributes for compositional zero-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11197–11206 (2023)
23. Yao, Y., Liu, P., Zhao, T., Zhang, Q., Liao, J., Fang, C., Lee, K., Wang, Q.: How to evaluate the generalization of detection? a benchmark for comprehensive open-vocabulary detection (2024)
24. Yao, Y., Zhang, A., Zhang, Z., Liu, Z., Chua, T.S., Sun, M.: Cpt: Colorful prompt tuning for pre-trained vision-language models. AI Open **5**, 30–38 (2024)

25. Zhang, J., Huang, J., Jin, S., Lu, S.: Vision-language models for vision tasks: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
26. Zheng, Z., Zhu, H., Nevatia, R.: Caila: Concept-aware intra-layer adapters for compositional zero-shot learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1721–1731 (2024)
27. Zhou, D.W., Wang, Q.W., Qi, Z.H., Ye, H.J., Zhan, D.C., Liu, Z.: Deep class-incremental learning: A survey. arXiv preprint arXiv:2302.03648 (2023)
28. Zhou, D.W., Zhang, Y., Ning, J., Ye, H.J., Zhan, D.C., Liu, Z.: Learning without forgetting for vision-language models. arXiv preprint arXiv:2305.19270 (2023)
29. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Conditional prompt learning for vision-language models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16816–16825 (June 2022)
30. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. International Journal of Computer Vision **130**(9), 2337–2348 (2022)

Literature Review

Generalization in Robotic Object Detection: GLIP vs Grounding DINO

Youssef Zahran

***Abstract*— In this paper, we explore the capabilities of two SOTA VLMs, specifically Grounding DINO and GLIP, within the context of object detection for robotic applications. We focus on evaluating their proficiency in the CZSL task. In CZSL, models are trained on a subset of existing compositions (attribute-object combinations) and are expected to recognize all existing compositions. We introduce two new datasets for this task, Single-CLEVr and Multiple-CLEVr. These datasets contain images of basic geometric shapes and colors, which provide a controlled environment for evaluating the models in CZSL. A crucial aspect of robotic applications is the range of objects a model needs to recognize, which changes over time. A model would benefit from learning classes incrementally while retaining previously learned classes, referred to as CIL. Therefore, we also investigate the effect of CIL on CZSL. Our experiments reveal that Grounding DINO excels in the CZSL task and significantly outperforms GLIP in this task. We also show that Grounding DINO struggles with CIL, which in turn negatively impacts its CZSL performance.**

I. Introduction

Recent advancements in Artificial Intelligence (AI) have facilitated the development of AI-embodied robots. These robots are becoming more and more intelligent making them capable for tasks beyond the traditional repetitive tasks, such as welding and painting [1]. They can be deployed for complex and dangerous tasks, such as disabling explosives or dealing with radioactive materials [2]. But, their usage is not limited to high-risk fields; they can be deployed in sectors such as healthcare [3], agriculture [4], et cetera.

A fundamental aspect of enhancing the capabilities of these robots involves their interaction with the environment and the ability to adapt their behavior to different situations. An important component in this adaptive mechanism is the understanding of attributes. Following [5], *attributes* are defined as high-level, semantically meaningful properties that characterize objects. Understanding and recognizing these attributes allow robots to engage in more sophisticated tasks, such as robotic grasping, a crucial task in robotic manipulation.

Yang et al. [6] successfully trained a model to recognizes objects based on attributes such as color and basic geometric shapes (like spheres or cubes). During deployment, the robot is able to grasp novel, unseen objects by describing them by their visual characteristic; for example an apple is described as a 'red sphere'. Furthermore, the recognition of attributes impacts the behavior of robots, for instance a robot has to exert different amount of force grasping a metal object compared to a glass object. This demonstrates the practical benefit of attribute recognition in robotic interactions, emphasizing the critical role of attributes in robotic perception.

The concept of attribute learning has been an area of interest. Generally speaking, one model is trained to identify and distinguish various attributes, while a separate model is trained to recognize objects. However, the true challenge is achieving this with a single, unified model. Such a model, capable of detecting both objects and their attribute simultaneously rather than separately, is more likely able to generalize to unseen situations.

Yet, models are often trained with datasets that do not cover all possible attribute-object combinations. For example, training a model with the classes 'red mug' and 'blue cup' does not imply that the model is able to recognize new combinations such as 'blue mug' or 'red cup'.

This introduces the necessity for Compositional Zero-Shot Learning (CZSL) [7, 8, 9, 10]. In CZSL, certain compositions (attribute-object combinations) are not encountered during training. However, each attribute and object is exposed at least once in a composition during training. To illustrate, consider a dataset with 10 objects and 20 attributes, thus resulting in a total of 200 possible compositions. However, in CZSL, the training dataset is selected such that each of the 10 objects and 20 attributes is represented at least once. Consequently, rather than needing all 200 compositions, a minimum of 30 compositions is sufficient. During inference, the model is expected to recognize all compositions, with as goal to imitate the generalization ability of human beings [9].

The top performing models in the CZSL task on well-established benchmark datasets are vision-language models (VLMs). CANet [11] is the best performing model on both MIT-States [12] and UT-Zappos 50K [13], while CAILA [14] performs best on MIT-States Generalized Split [12]. This highlights the potential of VLMs in the context of CZSL. VLMs are models that are pre-trained on a large-scale dataset consisting of rich image-text pairs enabling them to understand the relationship between the visual content and the textual information [15]. In doing so, they effectively bridge the gap between the realms of visual and textual information.

However, these models are designed for image classifications. No prior studies have investigated the performance of VLMs in the CZSL task in the context of object detection. Traditional object detection models [16, 17, 18] have two important modules: a backbone for feature extraction and a head for box prediction. Unlike image classification, which assigns one or more labels to the entire image, object detection involves localizing and classifying specific regions within the image. This paper will explore the performance

of two state-of-the-art (SOTA) VLMs in CZSL for object detection: Grounding DINO [19] and Grounded Language-Image Pre-training (GLIP) [20]. These models tend to be biased towards object categories rather than attributes [21]. This raises questions about their effectiveness in CZSL scenarios where such bias may pose a significant issue.

An existing challenge in object detection for robotic application is that the range of objects a model needs to recognize can change over time. Consider, for example, a robot operating in a supermarket. Here, the items available for sale change frequently. Sometimes, an item is introduced for a short period of time, removed, and later reintroduced again, while at other times, entirely new items are added. In both cases, it is desirable to have a robot that can quickly be adopted to recognize these new items. Traditionally, this scenario would require retraining the model using the entire original dataset, now including the new item(s). However, this approach becomes impractical as the dataset size increases. Moreover, the original data may not be accessible anymore due to storage constraints [22, 23] or privacy issues [24, 25].

To quickly add new objects or attributes, these models should be capable of learning new classes incrementally while retaining previously learned classes, referred to as Class-Incremental Learning (CIL) [26]. A critical challenge of CIL is catastrophic forgetting, i.e. learning new classes leads to loss of previously learned classes [27].

This paper aims to investigate the effectiveness of Grounding DINO and GLIP in CZSL within the context of object detection for robotic applications. The goal of the CZSL task is to recognize unseen compositions based on the learning from seen compositions. In this context, the effectiveness of a model is determined by its ability to generalize to unseen compositions [7, 8, 9, 10]. Therefore, to maintain performance consistency across the 'seen' and 'unseen' compositions, we propose that a model performing well in CZSL should demonstrate comparable performance between these two. To explore this further, we will evaluate the following sub-questions:

1) Do Grounding DINO and GLIP perform well in the CZSL task when trained with the entire dataset directly?

2) Does CIL affect the performance in the CZSL task?

Following that, section II explains concepts that are important for this paper, such as GLIP [20] and Grounding DINO [19]. Section III discusses existing literature and research that are related to this study. Section IV describes the experiments conducted and the dataset used to investigate the research question. Section V presents the findings of the experiments. In Section VI, the implications of the findings are discussed and future research work is proposed.

II. Background

A. VLM Foundations

Joulin et al. [28] noticed that there is a correlation between success of large networks and the size of the fully supervised datasets on which they are trained. This begs the question: Are fully supervised methods the most effective approach for developing better models? Fully supervised training is time-intensive, introduces strong biases towards specific tasks [29] and is inefficient compared to how humans learns [30].

The core of VLMs is learning perception from supervision contained in natural language [15]. This approach, motivated by the availability of large quantities of data that is available, allows models to learn passively from the rich information present in natural language. This approach is more scalable compared to standard crowd-sourced labeling [15]. The introduction of CLIP [15] has caused significant attraction towards VLMs pre-training [19, 20, 31, 32, 33].

A VLM is a deep neural network that extract image and text features from image-text pairs. It has an image encoder and a text encoder, which encode the image and text pairs into image and text embeddings. The VLM pre-training objectives can be grouped into three categories: contrastive objectives, generative objects and alignment objectives [34]

Contrastive objectives [15, 35] learn discriminative representations by pulling paired samples close and pushing others far away in the feature space. Generative objectives learn semantic features by training networks in image, text, or cross-modal generation tasks. Alignment objectives align the image-text pair on the embedding space. This either done via global image-text matching or local region-word matching [34].

B. CLIP

Like other VLMs, the core of CLIP [15] is to learn perception from supervision contained in natural language. CLIP is trained using a batch of N image-text pairs with the objective to match each image to its corresponding text. Figure 1 (left) shows the steps involved in pre-training CLIP [15]. First, the batch of N image-text pairs is separated. Next, the images are fed into an image encoder and the texts are fed into a text encoder. This yields N image features and N textual features. These features are in a multi-modal embedding space, i.e the features are from different modalities (images and text) but share a common vector space. CLIP [15] maximizes the cosine-similarity between the image and text embeddings of the N correct pairs, while simultaneously minimizing the cosine-similarity for the $N^2 - N$ incorrect pairs, all within the multi-modal embedding space. This is known as image-text contrastive learning.

Figure 1 (right) shows that CLIP [15] can be used as a zero-shot classifier. For example, when evaluated on the COCO [36] dataset, all 80 classes are presented as textual prompts within a template structure: "A photo of a {object}", where "{object}" is replaced with the classname. CLIP [15] determines the class by calculating the cosine-similarity between the image embedding and each of the 80 text embeddings. The image is then assigned the class associated with the highest cosine similarity score.

CLIP [15] can also be fine-tuned on a specific dataset. Here, typically initial layers are frozen to preserve the model's pre-trained understanding of concepts, while weights

of the latter layers are updated to adapt to the new dataset. During fine-tuning, the categories of the dataset are transformed into textual prompts, similar to the zero-shot evaluation. Now the model updates its weights using image-text contrastive learning.

C. Open-Vocabulary Detection

Object detection is a critical component in robotic applications. These models are often trained with well-labeled large-scale datasets such as COCO [36] and ImageNet [37]. Training with these datasets requires extensive resources, yet the model remains constrained to a limited set of object categories. Open-Vocabulary Detection (OVD) methods [38, 39, 40, 41] address this constraint by using VLMs to learn about novel categories using the weak supervision of the largely available image-text pairs.

Zareian et al. [41] were the first to propose a framework for OVD. First, they pre-train a model using low-cost image caption pairs, thereby constructing a visual-semantic space. Next, this pre-trained model is fine-tuned with specific cate-

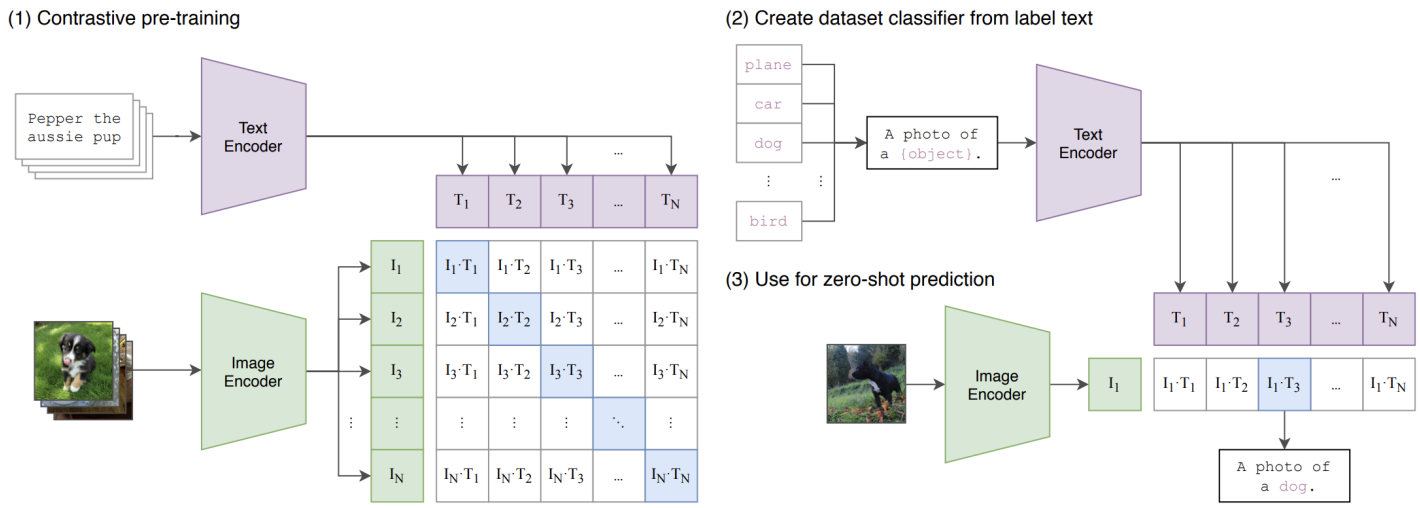


Fig. 1: Representation of the architecture of CLIP [15]. In (1), the model is trained with image-text pairs. The cosine-similarity is maximized for correct pairs and minimized for all others. Additionally, CLIP [15] can be used for zero-shot prediction on a dataset by generating text prompts for each class in a dataset (2), where the classification is determined by the highest cosine-similarity between the image embedding and the prompts (3).

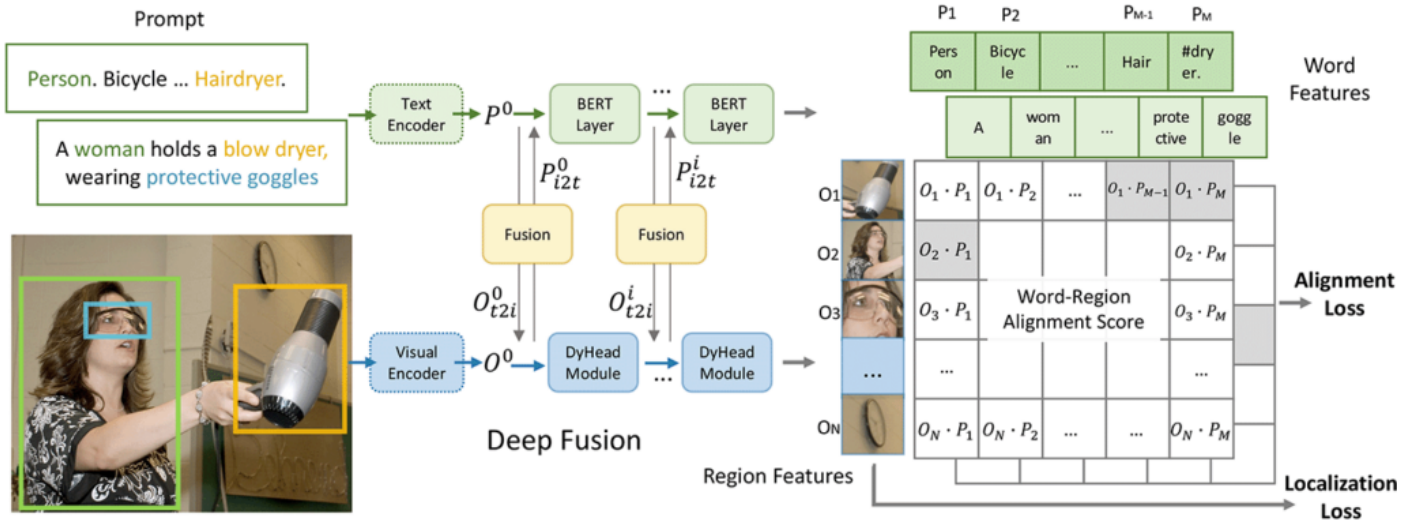


Fig. 2: Representation of the unified detection and grounding framework as implemented in GLIP [20]. Traditional object detection models assign categorical classes to detected objects, whereas GLIP [20] reinterprets detection as a grounding task, aligning image regions/boxes with corresponding noun phrases in a text prompt. This model jointly trains an image and language encoder to forecast accurate region-word pairings. GLIP [20] adds the cross-modality deep fusion to early fuse information from two modalities and to learn a language-aware visual representation.

gories on one of the costly annotated detection datasets such as COCO [36] and ImageNet [37]. ViLD [39] uses CLIP [15] as a teacher model to supervise student object detectors. They were the first to evaluate on more than 1000 categories.

Bravo et al. [42] extended this OVD task to the open-vocabulary Attribute Detection (OVAD) task. In OVAD, the goal is to detect objects and classify their attributes within an open vocabulary framework. This is challenging, since in the existing image-text teachers models, e.g. CLIP [15], the representations learned from image-text pairs tend to be biased towards object categories rather than attributes [21].

D. GLIP

GLIP [20] unifies object detection and phrase grounding by reformulating object detection as phrase grounding. Given an image and a corresponding caption, the phrase grounding task aims to ground each entity mentioned by a noun phrase in the caption to a region in the image. Due to this reformulation, instead of aligning each region to one of the predefined 'c' classes in a traditional classification head, GLIP [20] aligns each region with one of the 'c' phrases present in a text prompt.

Figure 2 shows the steps involved in GLIP [20]. It starts by inputting an image and its corresponding textual prompt into their respective visual and text encoders. The visual encoder is a Swin Transformer [43] and BERT [44] is used as text encoder. The features outputted by the Swin Transformer [43] and Bert [44] are denoted as O^0 and P^0 respectively. These outputs are then fed into a cross-modality multi-head attention module (X-MHA), depicted as 'fusion' in Figure 2. Here the modalities are fused using the two-cross attention processes. The first transforms textual feature into image-aware features (P_{t2i}): the model learns which image features are important for a specific textual feature. Simultaneously, the second transforms image features into text-aware features (O_{i2t}): the model learns which textual features are important for a specific image feature.

After 'fusion', the cross-informed features are added with the features that were given as input to 'fusion' block. These are then given as input to a BERT [44] Layer and a DyHead [45] module for feature enhancement, where the textual and image features are enhanced while their dimensions are maintained. Then, the outputs are fused again. This repeats itself multiple times, each time making the visual features language aware and the textual features image aware.

The final step involves generating a pre-defined number of regions for the visual features using a Feed-Forward Network (FFN). Finally, the feature corresponding to each region is compared to each textual feature and a similarity score is computed. During training, contrastive learning is used to maximize the similarity score between the regions and text that correspond to each other while minimizing the scores for non-matching pairs.

E. DETR

The goal of object detection is to predict bounding boxes and category labels for objects of interest. DEtection Transformer (DETR) [46] proposes a one-stage detector. Other

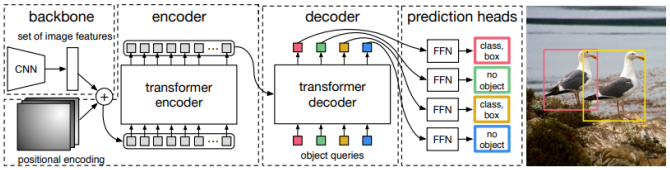


Fig. 3: Illustration of the DETR [46] model, which integrates a CNN with a transformer architecture for object detection. The CNN backbone creates a 2D feature representation of the input image. These features are then flattened and combined with positional encoding before being fed into the transformer encoder. The transformer decoder then takes as input a small fixed number of 'object queries' which are learned during training, and attends to the encoder's output. Each output of the decoder is fed into a FFN that predict the class and bounding box.

object detectors use intermediate steps, such as the Feature Proposal Network [47] (FPN) or defining anchors, to obtain detections. DETR [46] simplifies this by viewing object detection as a direct prediction problem, bypassing all of the intermediate steps. Figure 3 shows the steps involved in the DETR architecture. It contains three main components: a Convolutional Neural Network (CNN) backbone, an encoder-decoder Transformer and a FFN for the final detections.

The CNN backbone generates feature maps, which are then flattened and fed into the transformer encoder as input tokens. Each encoder layer follows a standard architecture [48], and consists of a multi-head self-attention module and a FFN. Given that self-attention is permutation-invariant, positional encoding is added to the input of each attention layer. These positional encodings are fixed for each input element and enables the model to understand the relative positions of tokens within the sequence.

Each encoder layer has a standard architecture [48]. It consists of a multi-head self-attention module and a FFN. Since self-attention is permutation-invariant, positional encoding is added to the input of each attention layer. The encoder outputs embeddings. Next, the decoder receives N input embeddings, referred to as 'object queries'. These 'object queries' are initially randomly initialized but are learned during training. To ensure that the decoder becomes permutation variant, the N 'object queries' are constrained to be distinct from each other. Also, embeddings generated by the encoder are also fed into the decoder. The decoder, in turn, outputs N embeddings, which are then input into an FFN. This FFN predicts N bounding boxes, which, along with the class labels, includes $\emptyset$ that denotes the absence of an object.

F. Grounding DINO

Grounding DINO [19] is built upon DINO [49]. DINO [49] is a DETR-like model [46]. It replaces the CNN backbone of traditional DETR [46] with the Swin Transformer [43]. DINO [49] also introduces a mixed query selection method. Where traditional DETR [46] only has learnable

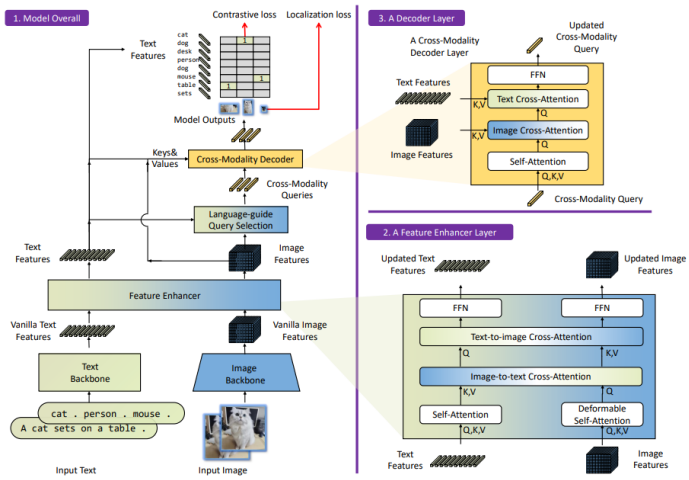


Fig. 4: The framework of Grounding DINO [19]. Similar to GLIP [20], Grounding DINO [19] phrases object detection as a grounding task.

'object queries', the queries in DINO [49] contain two parts: a content part and a positional part. The positional part is formulated as dynamic anchors [49] and these are initialized with a subset of the encoder outputs. The content queries are randomly initialized and are set to be learnable during training.

Grounding DINO [19] follows GLIP [20] and reformulates object detection as a phrase grounding task. In GLIP [20], the text and vision modalities are fused in the neck of the network. Liu et al. [19] argue that more feature fusion in the pipeline enables the model to perform better. The transformer based architecture of Grounding DINO [19] is similar to language models, making it easier to fuse cross-modality features in its whole pipeline.

Figure 4 shows the Grounding DINO framework. It consists of image and text backbones, a Feature Enhancer, a Language-guided Query Selection and a Cross-Modality Decoder. The image backbone is a Swin Transformer [43] and BERT [44] is used as the text backbone.

The Feature Enhancer block is the encoder of the architecture and is responsible for cross-modality feature fusion. It consists of multiple feature enhancer layers. Block 2 of Figure 4 illustrates what happens inside a feature enhancer layer. It shows that fusion between the modalities is performed twice within each layer. In the Image-to-text Cross-Attention block, the model learns which part of the textual input is relevant for a specific visual feature. In the Text-to-image Cross-Attention block, the model does the opposite: it learns what parts of the image are important for a specific textual feature. This is inspired from the X-MHA module of GLIP [20].

As mentioned in section II-E, a DETR [46] decoder takes 'object queries' as input. Following DINO [49], the 'object queries' in Grounding DINO [19] contain two parts: a content part and a positional part. Grounding DINO [19] uses the Language-guide Query selection module to initialize the positional part of the 'object queries' and they refer to them as cross-modality queries. Here, a similarity score for each image and text feature is computed. Then, the maximum score for each image feature across all text features is found. Following this, the top K indices of the highest scoring image features are returned. These indices are used to initialize the positional queries with the correct image features.

The Cross-Modality Decoder takes as input the output embeddings of the encoder (Feature Enhancer) and the cross-modality queries. Block 3 illustrates what happens inside a decoder layer. As in a feature enhancer layer, there is an Image Cross-Attention block and a Text Cross-Attention block. These are responsible for fusing the queries with the modalities. Each decoder layer outputs updated cross-modality queries that are used as input for the next layer.

Grounding DINO then compares the queries outputted by the decoder with the text features outputted by the encoder (Feature Enhancer). The dot product is computed between each text feature and query. Contrastive loss is used to match the query with its text feature. In order to obtain the predictions corresponding to each query, the queries are also fed into a FFN that predicts bounding boxes. This step is not shown in Figure 4.

III. Related Work

A. Compositional Zero-shot Learning

CZSL aims to recognize unseen object-attribute pairs (compositions) with prior knowledge of known compositions. In CZSL, individual attributes and objects are referred to as primitives. Misra et al. [50] composes classifiers from different types of primitive visual concepts. During training, they use a limited set of compositions to learn linear classifiers for each primitive. Then, they learn a transformation network that takes these classifiers as input and outputs a classifier for their composition. At test time, this transformation is used to generate classifiers for unseen compositions.

The method introduced by Misra et al. [50] highlights an important step in the development of CZSL models. However, this work and others such as [51] approach CZSL within a closed compositional space, i.e. they predefined all possible compositions. For example, the widely used MIT states [12] covers less than 6% of the whole compositional space. Mancini et al. [52] proposed the more realistic Open World CZSL (OW-CZSL), where no constraints were imposed on the search space. They also proposed CompCos [52] for this task. CompCos [52] embeds both images and compositional representations into a shared semantic space, where scores are computed between images and compositions with the cosine-similarity. However, in OW-CZSL, the model has to figure out implausible compositions (e.g. ripe dog) and discard them. This caused existing models to suffer severely in performance on the OW-CZSL.

Moreover, existing methods suffered from grasping the contextually between attributes and objects; visual appearance of attributes can change significantly with different

objects. For instance, the attribute 'old' has a different visual appearance in 'old car' compared to in 'old cat'. This has resulted in studies that emphasize the importance of context-aware approaches in CZSL [53, 54, 9]. The most popular solution is disentanglement, in which independent layers are allocated to extract intermediate visual representations of attribute and object.

B. VLMs for CZSL

Several studies [9, 55, 56, 57] have developed methods to enhance VLMs for the CZSL task. For instance, Nayak et al. [57] introduce compositional soft prompting (CSP), which forces CLIP [15] to learn the individual primitives of a composition. Typically, CLIP [15] is fine-tuned using prompt-templates like "A photo of a {object}", where {object} represents the object of interest. However, this method does not necessarily encourage the model to discriminate between the individual components of a composition. For instance, if trained with the object "red cup", it would be placed as a whole in the object placeholder, creating a single token for "red cup". In contrast, CSP [57] proposes to divide the prompt template into two trainable parts: one for the attribute and another for the object. This modification results in a new prompt template: "A photo of a Attribute Object". Consequently, when trained with "red cup," the words "red" and "cup" receive individual tokens. This method forces the model to learn about each component individually, which will aid for the CZSL task.

Want et al. [9] introduced Hierarchical Prompt Learning (HPL), a concept similar to CSP, but with an emphasis on learning compositional concepts in a hierarchical manner. They use three hierarchical embedding spaces designed to model attributes, objects, and their compositions. They refer to them as three distinct 'experts'. For instance, when analyzing an image of a 'purple apple', the composition expert may prefer to predict it as a 'red apple' or a 'purple eggplant', since these samples are more common during training. On the other hand, the object and attribute experts are more likely to correctly identify it as an 'apple' with the attribute 'purple'. Each expert uses a different prompt template: for attributes, it's "a photo of a {attribute} thing"; for objects, "A photo of the {object}"; and for compositions, "A photo of a {attribute} {object}". This method achieves better results than CSP.

C. Class-Incremental Learning

CIL refers to learning new classes incrementally while retaining previously learned classes [26, 27]. In typical CIL scenarios, learning occurs through a sequence of training tasks, each of which introduce new classes without any overlap of the classes from previous tasks. The main challenge is avoiding catastrophic forgetting, where learning new classes leads to a loss of knowledge from previous tasks [27]. Several CIL methods have been proposed to reduce catastrophic forgetting and they are classified into data-centric, model-centric and algortihm-centric methods [27]. The intuitive idea behind these methods is to make sure that optimizing the model for new classes will not hurt former ones.

Data-centric methods [58, 59, 60] seek help from former data to reduce catastrophic forgetting. An intuitive way to resist forgetting is by reviewing former classes [58, 59]. This approach, often referred to as Blurry CIL [61, 58, 62], reduces the learning difficulty since former classes are allowed to be revisited [27]. Other works [60, 63] build regularization terms with the former classes.

Model-centric methods [64, 65, 66, 67] tackle catastrophic forgetting by focusing on the evolvement of the model during training tasks. Some work [64, 65, 66] focus on expanding the network with each new training task. The idea behind this is that deep neural networks produce task-specific features [27]. For instance, when a model is trained on vehicles, it learns to represent features like wheels and windows. However, when the model is updated with new classes containing cats, it starts to represent features like whiskers and fur patterns. Model-centric methods assume that the capacity of a network is finite and therefore adapting to these new features would override the old ones and result in forgetting [27].

Early works [64, 65] added neurons when the representation ability is not enough to capture new classes. DER [66] duplicates the backbone network per task for stronger representation ability. However, having a backbone per task increases memory requirements significantly. MEMO [67] introduced a more memory-efficient strategy. They observed that the shallow layers of the backbones tend to be more generalizable, whereas deeper layers are more task-specific. Therefore, expanding shallow layers is not worth the memory increase. MEMO [67] proposes to address this by decoupling the backbone at the middle layers, dividing it into two types of blocks: specialized and generalized. Unlike DER [66], MEMO [67] only expands the specialized blocks for each new training task. This leads to a more memory-efficient approach.

Algorithm-centric methods [68, 69, 70] focus on algorithms to maintain the model's knowledge in former tasks. An intuitive way is to use Knowledge Distillation (KD). This is a method where a model (the student) is trained to mimic the output of a previously trained model (the teacher) [68]. The assumption here is that the teacher model already performs well on the seen classes and the student model would preserve this performance as it learns new classes.

Li et al. [69] are the first to successfully apply KD into CIL. The goal is to balance between learning new classes and remembering old classes. They achieved this by introducing a regularization term that prompts the student model to produce outputs similar to the teacher model for the classes known to the teacher, while also minimizing the loss on new classes. Following this, several other works [71, 72, 70] have built upon this. These works are all based on the adaptation or introduction of regularization terms into the loss function during training.

D. CIL with VLMs

CIL with VLMs remains a relatively unexplored field. PROOF [26] was the first to address this. In PROOF [26], new concepts are learned through the creation of projection mappings. Training is splitted into sequential tasks, each compromising a unique set of classes. For each new class, new projections are initialized while the projections from previous classes are frozen. The model learns projections for both the text and image embeddings.

Liu et al. [73] modify CLIP [15] by adding a Linear Adapter Layer after the CLIP [15] image encoder. This Linear Adapter Layer projects the features. During training all layers are frozen except the Adapter. They introduce a parameter retention strategy for updating the adapter, which is designed to mitigate forgetting. This updating strategy relies on selectively preserving a proportion of the original parameters based on their deviation from a dynamic threshold, thus maintaining previously learned knowledge while assimilating new tasks. This method allows for the balance of retaining learned information and adapting to new data.

E. Evaluation Metric

Object detection studies often use the mean Average Precision (mAP) as evaluation metric. It is defined as the average of the Average Precision (AP) across all classes.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

With N representing the number of classes. AP quantifies the model's precision and recall across different confidence thresholds, calculated as the area under the precision-recall curve:

$$AP \approx \sum_{k=1}^{K}(r(k) - r(k-1)) \cdot p(k),$$

Here, $p(k)$ and $r(k)$ denote the precision and recall at the kth threshold, respectively, defined as follows:

$$precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$

Yao et al. [74] argue that these metrics are deceiving for open vocabulary detection models, such as GLIP [20] or Grounding DINO [19]. Consider an image annotated with two ground-truth instances: a purple cylinder and a green cylinder, assuming these are the only cylinder categories in the model. These models tend to be able to detect and locate the presence of all cylinders in the image, but they struggle with the contextual description. They would predict two overlapping bounding boxes for each object, mistakenly assigning both 'green' and 'purple cylinder' labels to each object. All four of these boxes would be predicted with a high confidence score. Additionally, the highest scoring label is not necessarily the correct one. Consequently, the AP for each category would misleadingly be 0.50, despite the model failing to correctly comprehend the target objects. Yao et al. [74] refer to this as the 'inflated AP problem'.

The solution to this problem is incorporating Non-Maximum Suppression (NMS). In NMS, redundant bounding boxes are eliminated by selecting the most relevant ones based on their confidence scores and suppressing overlapping bounding boxes based on IoU. Before calculating the AP, a class-ignored NMS (C-NMS) is applied to remove the redundant predictions. This ensures that multiple bounding box predictions for the same object are handled appropriately and only the prediction with the highest confidence score is used in the calculation of the AP.

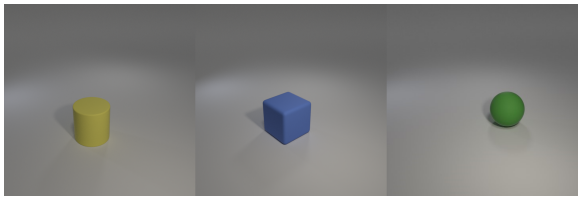
Algorithm 1 NMS-AP Metrics [74]

Require: $preds$: predictions
Require: GT: ground-truth
$pickedPreds = keepPreds = []$
for k *in* GT **do**
  for p *in* $preds$ **do**
    if $IoU(p, k) > 0.5$ **then**
      $pickedPreds = pickedPreds \cup p$
    else
      $keepPreds = keepPreds \cup p$
  $keepPreds = keepPreds \cup$ C-NMS$(pickedPreds)$
$mAP = AP(keepPreds, GT)$
return mAP

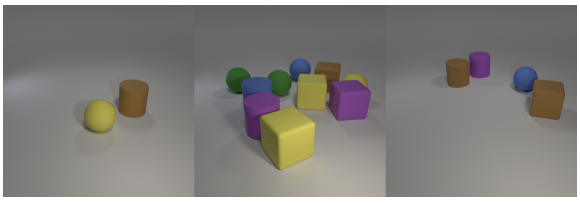
IV. Method

A. Dataset

Despite several studies on CZSL, interpreting the results obtained from current benchmark datasets, such as MIT-States[12], can be difficult or even misleading. For instance, the visual representation of the attribute 'wet' varies in 'wet dog' compared to in 'wet car'. Additionally, some attributes, such as 'heavy', are non-visual. This introduces a degree of ambiguity in the results. Does the model's performance



(a)



(b)

Fig. 5: Examples of images present in the Single-CLEVr (a) and Multiple-CLEVr (b) dataset.

reflects its capabilities in CZSL or is it a byproduct of dataset inconsistencies?

In response to this, we have created two new datasets: Single-CLEVr and Multiple-CLEVr. Both of which consist of synthetic-images that are generated following the CLEVr framework [75]. Single-CLEVr contains images with only one instance per image. Multiple-CLEVr contains 2 to 10 instances per image. In order to ensure the uniqueness of each instance, the positioning of the object and the direction of the light are slightly adjusted for every image in both datasets.

Both datasets consist of three objects: a cylinder, a sphere and a cube. These are simple, geometric shapes that should be easily distinguishable by a model. Additionally, there are six attributes: blue, red, green, purple, brown, and yellow. These attributes behave the same across all objects. There are a total of 18 unique compositions in the dataset, each representing a different object-attribute combination. We have ensured that all objects are of the same size and are made of a non-reflective, matte material. This, along the consistent behavior of attributes across different objects yields in dataset that is reliable for assessing a models performance on CZSL.

Both Single-CLEVr and Multiple-CLEVr serve as test datasets. In Single-CLEVr, each composition is presented 60 times, while Multiple-CLEVr has 300 instances of each composition. However, only Single-CLEVr is utilized for training, simulating a scenario where an object is placed in front of a camera. The test set of Single-CLEVr is employed to assess the models' performance in a controlled environment. On the other hand, Multiple-CLEVr tests how well the model adapts to complex, real-world conditions.

B. Label Correction

The bounding box labels generated by the CLEVr framework [75] were not tight around the object. The bounding boxes either were too large around the object (figure 6 (a)) or too small (figure 6 (b)). We fixed this by using Grounding DINO [19]. Both Single-CLEVr and Multiple-CLEVr were given as input to Grounding DINO [19] with as target class 'object'. Grounding DINO [19] successfully produced bounding boxes that are tightly fitted around the object. The result is shown in Figure 6 (c and d).

To assess the effectiveness of CZSL on VLMs and to investigate the impact of CIL on these models, we have designed two experiments. Experiment 1 focuses on evaluating the influence of sample size on the effectiveness of VLMs in CZSL, while also examining the trade-off between overfitting on the 'seen' split and achieving generalization on the 'unseen' split. Whereas experiment 2 is designed to evaluate the impact of CIL on CZSL. Experiment 1 helps to answer the question 'Do Grounding DINO and GLIP perform well in the CZSL task when trained with the entire dataset directly?' Experiment 2 helps in answering the question 'Does CIL affect the performance in the CZSL task?'. Both experiments are evaluated on the test datasets explained in section IV-A.

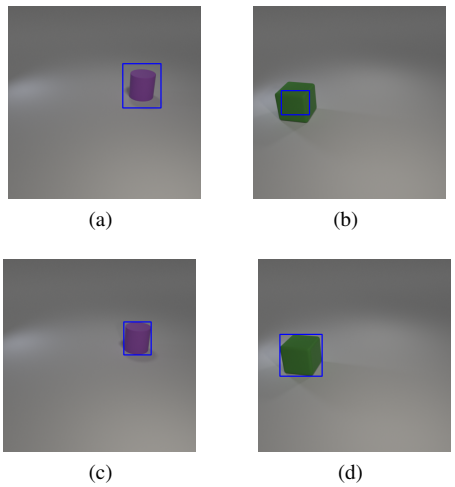


Fig. 6: Labels before (a and b) and after (c and d) correction.

1) Experiment 1: Sample Size Impact on CZSL: The goal of this experiment is to assess the proficiency of GLIP [20] and Grounding DINO [19] in CZSL. Following the protocols described in prior work [19, 20], these models are fine-tuned using a dataset consisting of six compositions: 'blue cube', 'red cube', 'green sphere', 'purple sphere', 'brown cylinder' and 'yellow cylinder', referred to as the 'seen' compositions. The remaining 12 compositions are considered the 'unseen' compositions. This split ensures an unbiased distribution of colors and objects: each color is seen once and associated with a single object, while each object appears twice. With this controlled dataset, a well performing model in CZSL would demonstrate comparable performance between 'seen' and 'unseen' compositions.

In order to explore the impact of sample size on CZSL, these models are trained with varying samples sizes per composition. Additionally, by adjusting these sizes, we aim to identify the sample size at which the performance on the 'unseen' split begins to deteriorate.

2) Experiment 2:CIL in Grounding DINO for CZSL: The goal of this experiment is to evaluate the impact of CIL on the models' ability to perform in CZSL.

This experiment consists of three distinct training procedures, each starting with the same initial weights. All procedures use exactly the same training set as experiment 1. Given that experiment 1 was trained with six compositions, each procedure includes six training steps. Note that "from scratch" in this context refers to starting from these initial weights.

1) In the first procedure, referred to as CIL, the model is trained sequentially with each individual composition. Each training step starts with the weights obtained at the end of the previous step. It starts with fine-tuning the model on a 'red cube'. Then, the model is fine-tuned again, this time on a 'blue cube'. This continues until the model has been fine-tuned with all composition mentioned in the 'seen' split.

2) The second procedure, referred to as Blurry CIL, is similar to the first, but with a significant modification. In each training step, the composition of the previous step is included in the current training set. As in CIL, each training step continues from the weights obtained at the end of previous step. It starts with fine-tuning the model on a 'red cube'. Then, in the next step, the training set comprises both the 'red cube' and a 'blue cube'. This continues, cumulatively adding each new composition to the training set until all compositions in the 'seen' split have been covered.

3) The third procedure is similar to the second procedure, but now each training step is trained from scratch. This procedure is included to ensure that at each training step all procedures can be compared against the traditional way of training and therefore serves as baseline. This procedure is referred to as traditional learning.

C. Evaluation Metric

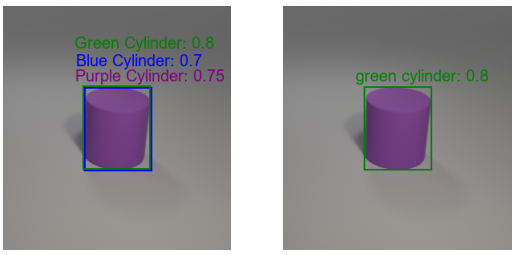
In this paper, we adopt the NMS-AP evaluation metric introduced by Yao et al. [74]. In this work, they introduced an issue known as the 'inflated AP problem' as explained in section III-E. The 'inflated AP problem' is crucial for models that are intended for applications where a robot acts based on defections. For example, a robot might respond differently to a detected 'green cylinder' than to a 'blue cylinder'. Therefore, it is important that the evaluation metric should reflect the robot's behavior in response to the detected objects. For instance, if a model detects both a 'blue cylinder' and a 'green cylinder' for the same object, the robot will most likely act on the detection with the highest confidence score. The NMS-AP metric, introduced by Yao et al. [74], aligns with this decision making since it only uses the label with the highest confidence score. This makes NMS-AP a suitable metric for robotic applications.

V. Results

A. Experiment 1: Sample Size Impact on CZSL

Experiment 1 aims to evaluate the proficiency of GLIP [20] and Grounding DINO [19] in CZSL and to determine the influence of the sample size on their performance. This experiment was performed using two datasets: Single-CLEVr, which served both as a training and testing dataset and Multiple-CLEVr, which is used solely for testing.

Single-ClEVr, provides a controlled setting to assess the CZSL proficiency. Figure 8 shows the superiority of Grounding DINO [19] over GLIP [20]. Figure 8 (a) shows that training with 15 examples per 'seen' composition results in the best NMS-mAP score for both models. The same conclusion is drawn from Figure 8 (b). It also demonstrates that Grounding DINO [19] performs almost equally well on both the 'seen' and 'unseen' sets when trained with this number of examples, showcasing its effectiveness in CZSL. On the other hand, GLIP [20] exhibits a large performance gap between the 'seen' and 'unseen' sets, showcasing its limitation in CZSL.



(a) (b)

Fig. 7: An illustration of the 'inflated AP problem' as highlighted in [74]. On the left (a), the model detects and locates the cylinder in the image but assigns labels for a purple, green, and blue cylinder to the same object. The green cylinder label is given the highest confidence score. This reflects the model's struggle with contextual descriptions, such as color, despite correctly identifying the object. On the right (b), NMS is applied and only the label with the highest confidence score remains, which in this case is the incorrect green cylinder.

In addition, Figure 8 (b) shows that for sample sizes smaller than 15 Grounding DINO [19] still achieves reasonable performance. This suggests that the model can be effectively trained in a few-shot setting and is expected to perform well in CZSL when limited data is available.

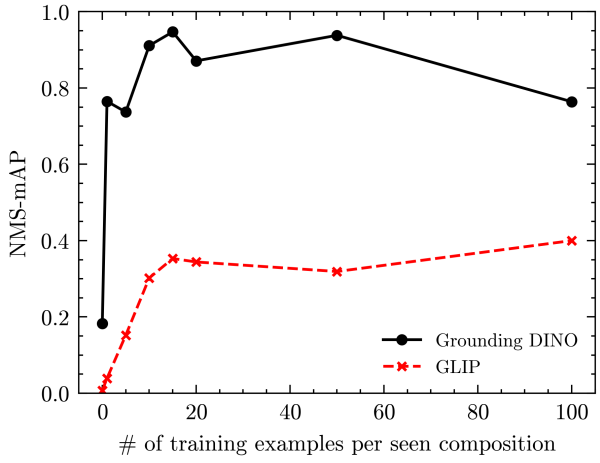
Figure 9 presents the performance of the models when tested on Multiple-CLEVr. Here, Grounding DINO's [19] peak performance significantly decreases compared to its near-perfect results on Single-CLEVr. However, the trend remains consistent: its performance on 'unseen' sets is nearly equivalent to that on 'seen' sets. GLIP's [20] performance does not appear to be affected much by the increased complexity of the test dataset. However, it is still outperformed by Grounding DINO [19] on the 'seen' compositions and has poor performance on the 'unseen' compositions.

Figure 8 (b) shows that training with more than 15 examples per 'seen' composition leads to a large loss in generalization ability on the 'unseen' split, and it also leads to a small decrease in performance on the 'seen' split. Further, Figure 9 reveals that this loss of generalization is even more notable on the Multiple-CLEVr dataset.

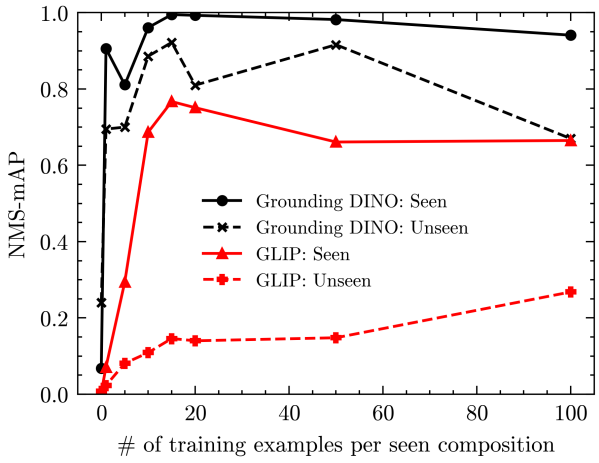
This experiment shows that Grounding DINO [19] performs well in CZSL when trained with the entire dataset while GLIP [20] does not.

B. Experiment 2:CIL in Grounding DINO for CZSL

Given the poor performance of GLIP [20] in experiment 1, experiment 2 is only conducted with Grounding DINO. The poor performance of GLIP [20] in CZSL under a traditional training setup implies that its performance in an incremental setup would, at best, match it current performance. Therefore, the focus of experiment 2 is to explore the impact of CIL on the performance of Grounding DINO [19] in the



(a)



(b)

Fig. 8: Results from experiment 1 on the Single-CLEVr dataset. (a) illustrates the performance differences between Grounding DINO and GLIP using the NMS-AP metric. (b) depicts the performance of the models on 'seen' versus 'unseen' compositions, demonstrating the superior capability of Grounding DINO in CZSL over GLIP.

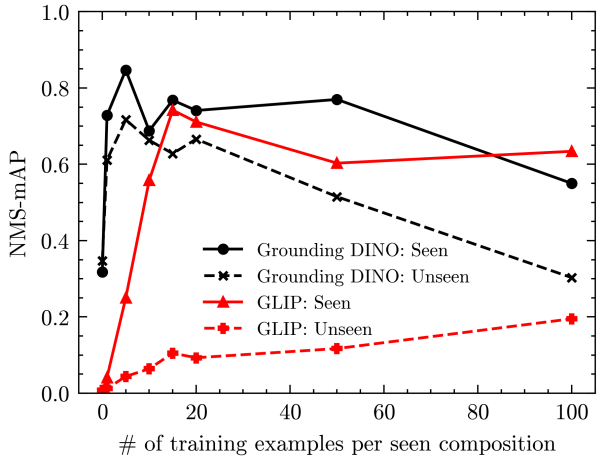


Fig. 9: Results from experiment 1 on the Multiple-CLEVr dataset. The performance on on Multiple-CLEVR is lower compared to those on Single-CLEVr (fig. 8).

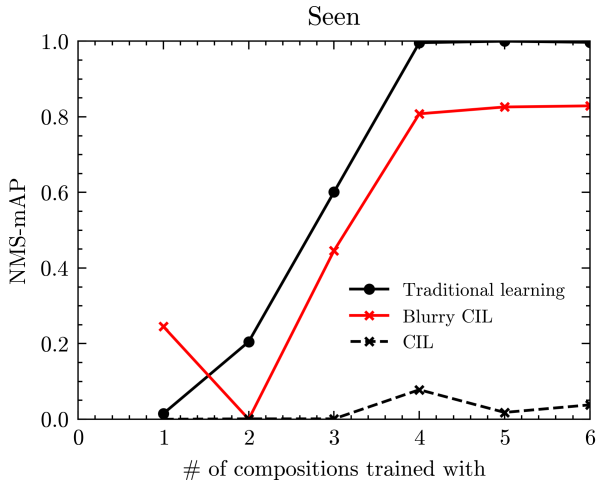
context of CZSL.

We implemented three distinct training procedures, as detailed in section IV-B.2. All training procedures in this experiment are performed using 15 samples per composition. This sample size was selected based on the results obtained from experiment 1. Figure 10 shows the effects of each training procedures on the performance. The order of training is 'blue cube', 'red cube', 'green sphere', 'purple sphere', 'brown cylinder', and 'yellow cylinder'. When the number of compositions trained with is one, the model has only been exposed to 'blue cube'. This increases incrementally and when this number is six, it has encountered all six compositions. At this point, a model proficient in CZSL should be able to recognize all 18 compositions in the Single-CLEVr dataset.

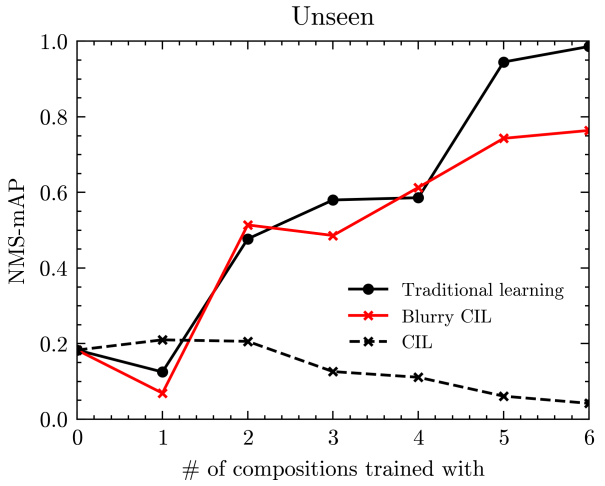
It is important to acknowledge that due to the inherent stochasticity in training a model, such as the randomness in the selection of mini-batches during gradient descent, trained models can yield different outcomes with each iteration. To achieve reproducible results, one can fix the random seed, ensuring that every training session with this specific seed replicates the exact same conditions, thus producing the identical model. However, our experiments were conducted without fixing the random seed, leading to variability in the outcomes. This is particularly notable when the number of compositions trained with is one, where each training procedure has only been exposed to a 'blue cube'. With a fixed seed, the results would have been identical, but in our case, they exhibit variation. While performing multiple runs is advisable to mitigate this variability, time limitation and GPU restriction prevented us from doing so. However, the differences between the procedures are substantial enough to conclude that even with multiple runs, the same findings would likely have been obtained.

In Figure 10 (a), we can see that the performance trends of traditional learning and Blurry CIL are similar on the 'seen' split, though Blurry CIL exhibits lower performance, and CIL does not seem to improve at all. At the last training step, Blurry CIL shows a performance decline of 16.9% compared to traditional learning, highlighting Grounding DINO's limitations under CIL. CIL, which involved training with one composition at a time, results in the worst performance. This can be attributed to the models' tendency to learn shortcuts when available. For instance, training only with a 'blue cube' leads the model to a suboptimal solution that does not truly learn the attribute 'blue' (or the object 'cube'). This suboptimal start makes it challenging to reach the ideal solution in the subsequent training steps.

Figure 10 (b) shows that for CIL, performance decreases with each new training step. This aligns with its lack of improvement during training, but in reverse: as training progresses, the model's ability to identify 'unseen' compositions deteriorates. Additionally, the difference in performance



(a)



(b)

Fig. 10: Results of Grounding DINO fine-tuned on Single-CLEVr with a sample size of 15 from Experiment 2 on the three procedures of incremental training incremental training performance. (a) shows Grounding DINO's performance on the 'seen' compositions and (b) shows on the 'unseen' compositions. Throughout the training stages, the compositions of the 'seen' set increase; when the number of compositions trained with is one, it includes only the 'blue cube', when this is 6, it contains the 'seen' set as defined in experiment 1. On the other hand, the 'unseen' set undergoes a reverse progression, initiating at 0. This point symbolizes the model's initial zero-shot performance, where it has not yet been exposed to any compositions.

between traditional learning and Blurry CIL is also visible on the 'unseen' compositions, with Blurry CIL performing 22.2% worse than traditional learning at the final training step. This slightly larger drop suggests that while Blurry CIL significantly affects Grounding DINO's overall performance, its impact on CZSL is less significant but still existing.

Figure 11 shows that Grounding DINO's best zero-shot

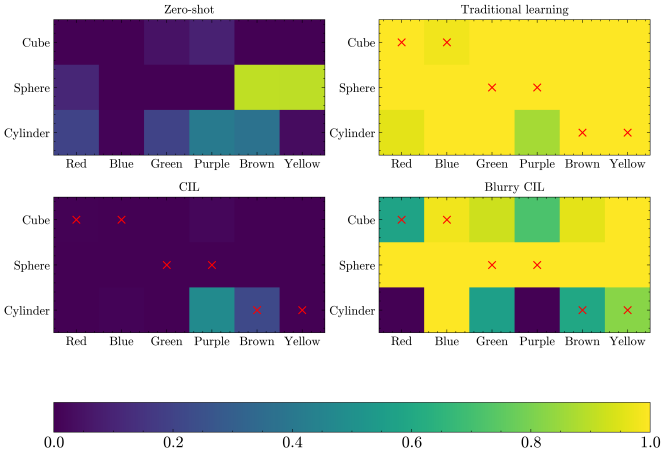


Fig. 11: Heatmaps depicting the NMS-mAP for Grounding DINO across all compositions in zero-shot, traditional learning, CIL and Blurry CIL. Each heatmap, except for the zero-shot, shows the performance of the final training step by which the model has been exposed to all six compositions. Compositions included in the training set are denoted by a red 'X'.

performance is for 'brown sphere' and 'yellow sphere'. After traditional learning, the model reaches top performance across all compositions, with a slight dip for 'purple cylinder'. The heatmap of CIL shows that 'yellow cylinder,' despite being the last trained composition in CIL, registers a score of zero. This is consistent for all compositions with the exception of 'brown cylinder' and 'purple cylinder,' which were trained in the second last and third last training steps, respectively. In the Blurry CIL heatmap, despite each shape being trained with twice, top performance is achieved only for compositions involving a sphere. This indicates that the model has difficulties in distinguishing 'cube' and 'cylinder' rather than color differences. For instance, 'purple cylinder' receives a performance of zero, whereas 'purple sphere' achieves top performance and 'purple cube' scores decent. This indicates that when presented with a 'purple cylinder', the model always misidentifies it as a 'purple cube,' affirming that the model recognizes the color 'purple' even though it had only encountered 'purple spheres' during training. The same can be derived from other compositions such as 'green cylinder' and 'green cube'.

This experiment shows that Grounding DINO performs poorly in CZSL when trained in a CIL fashion and suffers from catastrophic forgetting. However, training in a Blurry CIL reduces this catastrophic forgetting and enables the model to generalize to 'unseen' compositions.

VI. Conclusion

This paper investigates the effectiveness of Grounding DINO [19] and GLIP [20] in CZSL learning within the context of object detection for robotic applications. For robotic applications it is also desirable to be able to perform well in CIL. Our findings show that Grounding DINO [19] is

able to perform well in CZSL whereas GLIP [20] does not. Grounding DINO [19] fuses the text and vision modalities more often then GLIP [20], suggesting that frequent fusion aids in the model's generalization ability. However, Grounding DINO [19] shows limitations in standard CIL, though it shows reasonable performance in the Blurry CIL task. Our findings indicate that Blurry CIL deteriorates the model's effectiveness in CZSL, indicating that improvements in CIL could similarly affect CZSL performance.

Our study shows that there exists a large gap in the performance of Grounding DINO [19] in CIL compared to traditional learning. Traditional learning, while effective within static environments, fall short in dynamic environments encountered in robotics where rapid adaptability is desirable. CIL, on the other hand, facilitates towards models capable of online learning.

The large performance gap identified in our study not only marks an interesting topic for future research, but also offers an opportunity to advance in the field of robotics. Nevertheless, it is important for future research to consider the improvements in CIL could negatively affect the model's CZSL abilities. Therefore, future research should aim at enhancing the CIL performance of Grounding DINO [19] while preserving its CZSL capabilities.

REFERENCES

[1] Robots Done Right. (2023) History of robot applications. Accessed: October 18, 2023. [Online]. Available: https://robotsdoneright.com/Articles/history-of-robot-applications.html

[2] A. K. R. Nadikattu, "Influence of artificial intelligence on robotics industry," *International Journal of Creative Research Thoughts (IJCRT), ISSN*, pp. 2320–2882, 2021.

[3] M. Kyrarini, F. Lygerakis, A. Rajavenkatanarayanan, C. Sevastopoulos, H. R. Nambiappan, K. K. Chaitanya, A. R. Babu, J. Mathew, and F. Makedon, "A survey of robots in healthcare," *Technologies*, vol. 9, no. 1, 2021. [Online]. Available: https://www.mdpi.com/2227-7080/9/1/8

[4] P. Venkata Reddy, K. Nandini Prasad, and C. Puttamadappa, "Farmer's friend: Conversational ai bot for smart agriculture," *Journal of Positive School Psychology*, vol. 6, no. 2, pp. 2541–2549, 2022.

[5] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 3, pp. 453–465, 2013.

[6] Y. Yang, Y. Liu, H. Liang, X. Lou, and C. Choi, "Attribute-based robotic grasping with one-grasp adaptation," *CoRR*, vol. abs/2104.02271, 2021. [Online]. Available: https://arxiv.org/abs/2104.02271

[7] S. Huang, Q. Wei, and D. Wang, "Reference-limited compositional zero-shot learning," 2023.

[8] Q. Wang, L. Liu, C. Jing, H. Chen, G. Liang, P. Wang, and C. Shen, "Learning conditional attributes for compositional zero-shot learning," 2023.

[9] X. Zhang, S. Gui, Z. Zhu, Y. Zhao, and J. Liu, "Hierarchical prototype learning for zero-shot recognition," *CoRR*, vol. abs/1910.11671, 2019. [Online]. Available: http://arxiv.org/abs/1910.11671

[10] Z. Zheng, H. Zhu, and R. Nevatia, "Caila: Concept-aware intra-layer adapters for compositional zero-shot learning," 2023.

[11] Q. Wang, L. Liu, C. Jing, H. Chen, G. Liang, P. Wang, and C. Shen, "Learning conditional attributes for compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 197–11 206.

[12] P. Isola, J. J. Lim, and E. H. Adelson, "Discovering states and transformations in image collections," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1383–1391.

[13] A. Yu and K. Grauman, "Fine-grained visual comparisons with local learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 192–199.

[14] Z. Zheng, H. Zhu, and R. Nevatia, "Caila: Concept-aware intra-layer adapters for compositional zero-shot learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1721–1731.

[15] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[17] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[19] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," 2023.

[20] L. H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J. Hwang, K. Chang, and J. Gao, "Grounded language-image pre-training," *CoRR*, vol. abs/2112.03857, 2021. [Online]. Available: https://arxiv.org/abs/2112.03857

[21] K. Chen, X. Jiang, Y. Hu, X. Tang, Y. Gao, J. Chen, and W. Xie, "Ovarnet: Towards open-vocabulary object attribute recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 23 518–23 527.

[22] G. Krempl, I. Žliobaite, D. Brzeziński, E. Hüllermeier, M. Last, V. Lemaire, T. Noack, A. Shaker, S. Sievi, M. Spiliopoulou *et al.*, "Open challenges for data stream mining research," *ACM SIGKDD explorations newsletter*, vol. 16, no. 1, pp. 1–10, 2014.

[23] M. M. Gaber, "Advances in data stream mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 79–85, 2012.

[24] M. A. P. Chamikara, P. Bertók, D. Liu, S. Camtepe, and I. Khalil, "Efficient data perturbation for privacy preserving and accurate data stream mining," *Pervasive and Mobile Computing*, vol. 48, pp. 1–19, 2018.

[25] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3366–3385, 2021.

[26] D.-W. Zhou, Y. Zhang, J. Ning, H.-J. Ye, D.-C. Zhan, and Z. Liu, "Learning without forgetting for vision-language models," *arXiv preprint arXiv:2305.19270*, 2023.

[27] D.-W. Zhou, Q.-W. Wang, Z.-H. Qi, H.-J. Ye, D.-C. Zhan, and Z. Liu, "Deep class-incremental learning: A survey," *arXiv preprint arXiv:2302.03648*, 2023.

[28] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache, "Learning visual features from large weakly supervised data," *CoRR*, vol. abs/1511.02251, 2015. [Online]. Available: http://arxiv.org/abs/1511.02251

[29] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *CVPR 2011*. IEEE, 2011, pp. 1521–1528.

[30] J. J. DiCarlo, D. Zoccolan, and N. C. Rust, "How does the brain solve visual object recognition?" *Neuron*, vol. 73, no. 3, pp. 415–434, 2012.

[31] Z.-Y. Dou, A. Kamath, Z. Gan, P. Zhang, J. Wang, L. Li, Z. Liu, C. Liu, Y. LeCun, N. Peng, J. Gao, and L. Wang, "Coarse-to-fine vision-language pre-training with fusion in the backbone," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 32 942–32 956. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/d4b6ccf3acd6ccbc1093e093df345ba2-Paper-Conference.pdf

[32] L. Yao, J. Han, Y. Wen, X. Liang, D. Xu, W. Zhang, Z. Li, C. Xu, and H. Xu, "Detclip: Dictionary-enriched visual-concept paralleled pre-training for open-world detection," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9125–9138, 2022.

[33] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language

representation learning with noisy text supervision," in *International conference on machine learning*. PMLR, 2021, pp. 4904–4916.

[34] J. Zhang, J. Huang, S. Jin, and S. Lu, "Vision-language models for vision tasks: A survey," *arXiv preprint arXiv:2304.00685*, 2023.

[35] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 4904–4916. [Online]. Available: https://proceedings.mlr.press/v139/jia21b.html

[36] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[38] C. Feng, Y. Zhong, Z. Jie, X. Chu, H. Ren, X. Wei, W. Xie, and L. Ma, "Promptdet: Towards open-vocabulary detection using uncurated images," 2022.

[39] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," *arXiv preprint arXiv:2104.13921*, 2021.

[40] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting twenty-thousand classes using image-level supervision," 2022.

[41] A. Zareian, K. D. Rosa, D. H. Hu, and S.-F. Chang, "Open-vocabulary object detection using captions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 393–14 402.

[42] M. A. Bravo, S. Mittal, S. Ging, and T. Brox, "Open-vocabulary attribute detection," 2023.

[43] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *CoRR*, vol. abs/2103.14030, 2021. [Online]. Available: https://arxiv.org/abs/2103.14030

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

[45] X. Dai, Y. Chen, B. Xiao, D. Chen, M. Liu, L. Yuan, and L. Zhang, "Dynamic head: Unifying object detection heads with attentions," *CoRR*, vol. abs/2106.08322, 2021. [Online]. Available: https://arxiv.org/abs/2106.08322

[46] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," *CoRR*, vol. abs/2005.12872, 2020. [Online]. Available: https://arxiv.org/abs/2005.12872

[47] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," *CoRR*, vol. abs/1612.03144, 2016. [Online]. Available: http://arxiv.org/abs/1612.03144

[48] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[49] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," 2022.

[50] I. Misra, A. Gupta, and M. Hebert, "From red wine to red tomato: Composition with context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1792–1801.

[51] T. Nagarajan and K. Grauman, "Attributes as operators: factorizing unseen attribute-object compositions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 169–185.

[52] M. Mancini, M. F. Naeem, Y. Xian, and Z. Akata, "Open world compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 5222–5230.

[53] N. Saini, K. Pham, and A. Shrivastava, "Disentangling visual embeddings for attributes and objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 658–13 667.

[54] X. Li, X. Yang, K. Wei, C. Deng, and M. Yang, "Siamese contrastive embedding network for compositional zero-shot learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 9326–9335.

[55] X. Lu, Z. Liu, S. Guo, J. Guo, F. Huo, S. Bai, and T. Han, "Drpt: Disentangled and recurrent prompt tuning for compositional zero-shot learning," 2023.

[56] S. Huang, B. Gong, Y. Feng, Y. Lv, and D. Wang, "Troika: Multi-path cross-modal traction for compositional zero-shot learning," 2023.

[57] N. V. Nayak, P. Yu, and S. H. Bach, "Learning to compose soft prompts for compositional zero-shot learning," *arXiv preprint arXiv:2204.03574*, 2022.

[58] J. Bang, H. Kim, Y. J. Yoo, J. Ha, and J. Choi, "Rainbow memory: Continual learning with a memory of diverse samples," *CoRR*, vol. abs/2103.17230, 2021. [Online]. Available: https://arxiv.org/abs/2103.17230

[59] H. Zhao, H. Wang, Y. Fu, F. Wu, and X. Li, "Memory-efficient class-incremental learning for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp. 5966–5977, 2021.

[60] J. Jiang, E. Cetin, and O. Celiktutan, "Ib-drr-incremental learning with information-back discrete representation replay," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3533–3542.

[61] H. Koh, D. Kim, J.-W. Ha, and J. Choi, "Online continual learning on class incremental blurry task configuration with anytime inference," *arXiv preprint arXiv:2110.10031*, 2021.

[62] J. Bang, H. Koh, S. Park, H. Song, J.-W. Ha, and J. Choi, "Online continual learning on a contaminated data stream with blurry task boundaries," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9275–9284.

[63] S. Tang, D. Chen, J. Zhu, S. Yu, and W. Ouyang, "Layerwise optimization by gradient decomposition for continual learning," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2021, pp. 9634–9643.

[64] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong learning with dynamically expandable networks," *arXiv preprint arXiv:1708.01547*, 2017.

[65] X. Li, Y. Zhou, T. Wu, R. Socher, and C. Xiong, "Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3925–3934.

[66] S. Yan, J. Xie, and X. He, "DER: dynamically expandable representation for class incremental learning," *CoRR*, vol. abs/2103.16788, 2021. [Online]. Available: https://arxiv.org/abs/2103.16788

[67] D.-W. Zhou, Q.-W. Wang, H.-J. Ye, and D.-C. Zhan, "A model or 603 exemplars: Towards memory-efficient class-incremental learning," *arXiv preprint arXiv:2205.13218*, 2022.

[68] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[69] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[70] J. Smith, Y.-C. Hsu, J. Balloch, Y. Shen, H. Jin, and Z. Kira, "Always be dreaming: A new approach for data-free class-incremental learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 9374–9384.

[71] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[72] K. Lee, K. Lee, J. Shin, and H. Lee, "Overcoming catastrophic forgetting with unlabeled data in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 312–321.

[73] X. Liu, X. Cao, H. Lu, J. wen Xiao, A. D. Bagdanov, and M.-M. Cheng, "Class incremental learning with pre-trained vision-language models," 2023.

[74] Y. Yao, P. Liu, T. Zhao, Q. Zhang, J. Liao, C. Fang, K. Lee, and Q. Wang, "How to evaluate the generalization of detection? a benchmark for comprehensive open-vocabulary detection," 2023.

[75] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2901–2910.