CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

Context Aware Attribute-Object Guided Compositional Zero-Shot Learning

by

Syed Baqir Ali Naqvi

A thesis submitted in partial fulfillment for the degree of Master of Science

in the

Faculty of Computing
Department of Software Engineering

2025

CERTIFICATE OF APPROVAL

Context Aware Attribute-Object Guided Compositional Zero-Shot Learning

by

Syed Baqir Ali Naqvi

(MAI241001)

THESIS EXAMINING COMMITTEE

S. No.	Examiner	Name	Organization
(a)	External Examiner	Dr. Hassan Mujtaba Kayani	FAST, Islamabad
(b)	Internal Examiner	Dr. Farah Haneef	CUST, Islamabad

Dr. Syed Saqib Raza Rizvi
Thesis Supervisor
December, 2025

Dr. Nadeem Anjum
Head
Dept. of Software Engineering
December, 2025

Dr. Muhammad Abdul Qadir
Dean
Faculty of Computing
December, 2025

Author's Declaration

I, **Syed Baqir Ali Naqvi** hereby state that my MS thesis titled "**Context Aware Attribute-Object Guided Compositional Zero-Shot Learning**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad.

At any time if my statement is found to be incorrect even after my graduation, the university has the right to withdraw my MS Degree.

(Syed Baqir Ali Naqvi)

Registration No: MAI241001

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "**Context Aware Attribute-Object Guided Compositional Zero-Shot Learning**" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MS Degree, the University reserves the right to withdraw/revoke my MS degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Syed Baqir Ali Naqvi)

Registration No: MAI241001

Acknowledgement

First and foremost, I dedicate this thesis to my mother, Syeda Noor Bano Zaidi. Though she is no longer with us, her love continues to guide me. She taught me the true meaning of life, compassion and resilience. May her soul find eternal peace and hapiness in paradise — You are deeply missed every single day **!**

I also dedicate this work to my father, who, as a single parent, has been my unwavering source of strength. His endless trust, sacrifices, and constant support carried me through every trial and triumph. Thank you for believing in me when I sometimes doubted myself.

This research work is tribute to my esteemed supervisor, **Syed Saqib Raza Rizvi**, who stood by me in moments of joy and struggle alike.

(Syed Baqir Ali Naqvi)

Abstract

Understanding and recognizing unseen Attribute-Object (AO) compositions is a major challenge of machine learning models. Most existing approaches rely heavily on labeled data and often fail when faced with unseen combinations, showing strong seen–unseen bias. These limitations restrict generalization, reduce interpret-ability, and make it harder to use such models in real-world applications that catered through Compositional Zero-Shot Learning (CZSL) based framework.

This research introduces a framework under the domain of CZSL, designed to improve recognition of unseen AO compositions while also enhancing semantic grounding and interpret-ability.

The framework integrates a CLIP ViT-B/16 image encoder and CLIP text encoder with two enhancement modules: a Visual Feature Enhancer (VFE) to refine visual embeddings and a Visual/Text Feature Enhancer (VTE) to strengthen text–visual alignment. A key innovation is the use of Vision-Language Primitive Decomposition (VLPD), which breaks down both images and text into primitive components. These are recombined using Composition Fusion to form more robust and generalizable AO representations. To provide human-understandable outputs, the predicted compositions are further passed into a Retrieval-Augmented Generation (RAG) module, which leverages OPT-1.3 to generate sentence-level descriptions of predictions.

Experiments were conducted on two benchmark datasets: MIT-States (a complex dataset with 115 attributes, 245 objects, and thousands of AO pairs) and UT-Zappos50K (a simpler dataset with fine-grained shoe categories). Both datasets were evaluated under Closed World (CW) and Open World (OW) settings using standard CZSL metrics: Seen Accuracy (S), Unseen Accuracy (U), Harmonic Mean (H), and Area Under the Curve (AUC). Results demonstrate that the proposed framework consistently outperforms state-of-the-art baselines (CLIP, CoOp, ProDA, CSP, PCVL, HPL, DFSP). Notably, it achieves higher unseen accuracy and better harmonic mean, showing reduced seen–unseen bias and more balanced generalization.

In closed-world setting, proposed czsl model achieved 49.1% seen accuracy, 52.1% unseen accuracy against MIT-States dataset and 66.9% seen accuracy, 68.8% unseen accuracy against UT-Zappos. In Open-World setting, proposed czsl model achieved 48% seen accuracy, 18.5% unseen accuracy against MIT-States dataset and 66.8% seen accuracy, 55.5% unseen accuracy against UT-Zappos dataset.

The framework also provides qualitative improvements by generating sentence-level descriptions of predictions through the RAG + OPT-1.3 module. This not only makes the model more interpretable but also assists in error analysis and enhances user trust in predictions.

In conclusion, this work contributes a novel end-to-end CZSL framework that improves both performance and explainability. By combining feature enhancement, primitive decomposition, and natural language grounding, CAAO-CZSL moves closer to human-like reasoning in visual recognition tasks. Future research can extend this approach by adopting larger language models, scaling to more diverse multi-modal datasets, and applying the framework to real-world applications such as e-commerce recommendation, assistive vision, and robotics.

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ANN	Approximate Nearest Neighbor
AO	Attribute-Object
AUC	Area Under the Curve
BERT	Bidirectional Encoder Representations from Transformers
BM25	Best Matching 25
BPE	Byte Pair Encoding
CLIP	Contrastive Language-Image Pre-training
CNN	Convolutional Neural Network
CoCoOp	Conditional Context Optimization
CoOp	Context Optimization
CoT	Chain-of-Thought
CPU	Central Processing Unit
CGE	Compositional Graph Embedding
CSP	Compositional Soft Prompting
CV	Computer Vision
CW	Closed-World
CZSL	Compositional Zero-Shot Learning
CSV	Comma-Separated Values
DFSP	Dense Feature Synthesis for Prompting
DL	Deep Learning
DPR	Dense Passage Retrieval
FFN	Feed-Forward Network
FiD	Fusion-in-Decoder

GAN	Generative Adversarial Network
GCN	Graph Convolutional Network
GPU	Graphics Processing Unit
H or **HM**	Harmonic Mean
HPL	Hyperbolic Prompt Learning
LLM	Large Language Model
LoRA	Low-Rank Adaptation
LSTM	Long Short-Term Memory
MaPLe	Multi-modal Prompt Learning
MHSA	Multi-Head Self-Attention
ML	Machine Learning
MLP	Multi-Layer Perceptron
MIT-States	MIT-States Dataset
NLP	Natural Language Processing
OBJ	Object Accuracy
OPT-1.3B	Open Pre-trained Transformer (1.3 Billion parameters)
OW	Open-World
OW-CZSL	Open-World Compositional Zero-Shot Learning
PCVL	Prompt-based Compositional Vision-Language Learning
PIL	Python Imaging Library
PPL	Probabilistic Prompt Learning
ProDA	Probabilistic Prompt Distribution for Prompt Learning
RaR	Rephrase and Respond
RAG	Retrieval-Augmented Generation
RAM	Random Access Memory
RE2	Re-reading
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
S	Seen Accuracy
S2A	System 2 Attention
SSD	Solid-State Drive
SVM	Support Vector Machine

TFE	Text Feature Enhancer
U	Unseen Accuracy
UT-Zappos	UT-Zappos Dataset
VFE	Visual Feature Enhancer
ViT	Vision Transformer
ViT-B/16	Vision Transformer Base model with 16x16 patch size
VLM	Vision-Language Model
VLMs	Vision-Language Models
VLPD	Vision-Language Primitive Decomposition
VRAM	Video Random Access Memory
ZSL	Zero-Shot Learning

Chapter 1

Introduction

1.1 Background and Motivation

Humans are naturally very good at imagining and understanding things they have never seen before. This is possible because we have already experienced so many different objects, colors, shapes, and textures. By using this knowledge, we can easily combine familiar ideas to recognize or visualize something new. For example, even if someone has never seen a "blue apple," they can still imagine it by mixing what they know about the color blue and the shape of a apple [1].

Machines Learning models, however, do not naturally have this ability. They usually rely on being trained with large amounts of labeled data, and they often struggle when faced with situations or combinations that were not part of their training. For Machine Learning (ML) to become more powerful and useful, models must also learn to generalize from what they already know and adapt to new or unseen situations without requiring explicit training for every possibility [2].

This motivation drives much of today's Artificial Intelligence (AI) researches. The ultimate aim is to make machines more flexible and closer to human reasoning—able to reuse past knowledge, handle unexpected cases, and imagine new possibilities. Achieving this would not only improve how machines learn but also open doors to more intelligent and adaptable systems in the real world.

1.2 The Role of ML and Computer Vision

ML has become one of the most important drivers of AI. It allows machines to discover patterns in data and improve performance with experience, rather than relying only on hard-coded rules [2]. In simple terms, ML helps machines to "learn" in a way similar to humans, such as recognizing patterns, making predictions, or solving problems [3]. Over time, ML has expanded its role to include tasks such as reasoning, language understanding, and even creativity [4, 5].

Historically, humans created tools and machines to solve practical problems. As these tools became more advanced, many processes were automated, reducing the need for direct human involvement [6]. In the same way, ML represents a leap forward, because it enables machines not just to follow instructions but to adapt and make decisions on their own [7]. One of the most significant applications of ML is in visual understanding, which naturally extends into the domain of Computer Vision, where learning algorithms enable machines to perceive and reason about the world visually[8].

Computer Vision (CV), a close partner of ML, focuses on enabling machines to interpret and understand visual data. Just like humans can recognize unfamiliar objects by relating them to things they already know, CV systems aim to do the same. This field has grown rapidly over the last decade, particularly with breakthroughs in image classification and object detection [9]. The goal of CV is to give machines "eyes" that allow them to perceive the world visually and reason about it, similar to human cognitive perception.

CV is also highly interdisciplinary—it connects with optics, signal processing, mathematics, ML, robotics, and graphics. Applications range from image and video processing to object detection, segmentation, tracking, and even visual question answering [10–12]. Together, ML and CV form the foundation for developing systems that can both "think" and "see", which is essential for advancing toward human-like reasoning in AI. These advancements highlight how visual and learning systems increasingly complement each other in modern AI applications.

1.3 Deep Learning and the Challenge of Generalization

Over time, Deep Learning (DL) has become the backbone of most Computer Vision tasks. DL models have achieved state-of-the-art results in almost every vision task, from object recognition to scene understanding [13]. These models are typically trained using supervised learning, where a network learns from labeled data and improves by adjusting its parameters through backpropagation [14].

While supervised learning is powerful, it also comes with important limitations that make it less flexible than human learning:-

Dependence on Labeled Data: DL models often require millions of labeled examples to train effectively [15]. However, collecting such large datasets is costly and time-consuming. Moreover, models trained on huge datasets often overfit and perform poorly when applied to small datasets or unseen situations [16].

Adding New Classes Requires Re-training from Scratch: Humans can recognize tens of thousands of visual categories throughout their lives [17], but DL models cannot easily do this. If a new object category appears, the model must often be retrained from scratch, making it inefficient and inflexible [18].

Lack of Semantics: Traditional supervised learning models mainly focus on visual features and ignore the role of language or text. Humans, however, combine both visual and linguistic information when reasoning about the world. Without integrating textual or semantic information, DL models remain limited in their reasoning abilities [19].

Because of these limitations, DL models are described as "data-hungry": no matter how large a dataset is, it will always miss many possible real-world variations [20]. This highlights the urgent need for methods that can handle new and unseen challenges without relying heavily on labeled data. This gap in traditional supervised learning has led to the rise of Zero-Shot Learning (ZSL). These challenges the traditional deep learning systems struggle to match the adaptability and efficiency

of human perception. As real-world environments continue to grow more diverse and unpredictable, the need for learning models that can generalize beyond their training data becomes increasingly critical.

1.4 Zero-Shot Learning

Zero-Shot Learning (ZSL) is a technique that categories new or unseen instances considering its semantics and prior knowledge with the help of high-level inference and reasoning capabilities [9, 20].

ZSL overcomes the limitations of supervised learning and deep neural network models that require large dataset to train on. The common short-comings of supervised learning that countered by the ZSL are discussed [9].

First; supervised learning models significantly rely on high volume annotated data. Second; supervised learning models require retraining to add a new class. Lastly, supervised learning models lack the notion of human way of understanding [9].

ZSL process knowledge of the seen or known traits to correctly classify unseen to address uncertainty. Though ZSL focuses on recognizing entirely unseen classes using semantic descriptions, it falls short when faced with novel combinations of known attributes and objects, to fill this gap Compositional Zero-Shot Learning (CZSL) is designed.

1.4.1 Compositional Zero-shot Learning

Compositional Zero-Shot Learning (CZSL) enhance the abilities of ZSL by enabling its models to recognize unseen combinations of attributes and objects. It investigates visuals with seen combinations of attributes (such as “white” and “brown”) and objects (such as “cats” and “bears”) to infer unseen compositions (such as “white bears” or “brown cats”) depicted in Figure 1.1 [21]. CZSL and standard ZSL differ in their methods and objectives [22–28] as ZSL recognize unseen attributes by mapping visual features using inference abilities from prior

knowledge [29]. However, CZSL method consider both attributes and objects of seen compositions instead of categorization of attributes only [30–33].

CZSL understands that how attributes affect objects, likewise extending this knowledge from seen compositions to different attributes of the similar objects means that CZSL explore the interactions between objects and attributes to apply the facts inferred from prior knowledge and apply to unseen compositions respectively. As shown in Figure 1.1, knowledge from seen compositions, e.g., "white dog" and "white cat", build the capacity of model to learn the "white" attribute, and learn the object concept "bear" through "black bear" and "brown bear" [21].

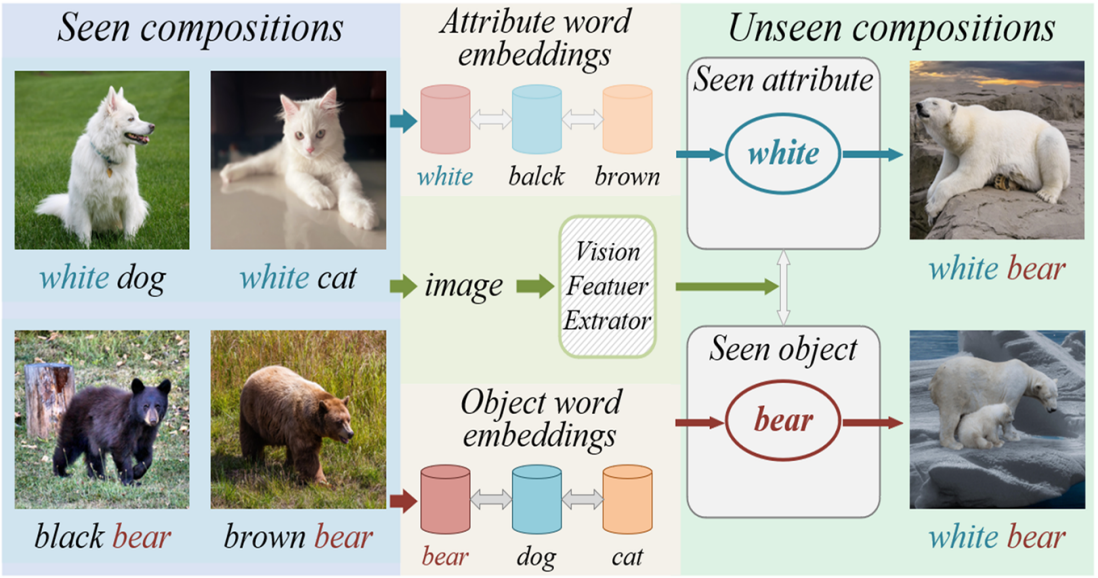

FIGURE 1.1: Overview of an Attribute and Objects Pairs for CZSL [21]

Recognizing compositional visual concepts is challenging for even advanced deep learning systems but also foundational characteristic of the human intellect [34]. Like, human being can recognize unseen sliced tomatoes having seen sliced potatoes and red tomatoes with ease. CZSL technique is capable to recognize novel concepts by optimally utilizing a huge combinatorial semantic space. If a dataset contains 400 attributes and 500 objects, this implies at least 200,000 classes in an open-world combinatorial ZSL scenario. However, accessible classes were only 5% during training; therefore, the CZSL technique considerably reduces the requirement of large-scale data for training.

Vision-based traditional methods focus on learning features extracted from images to decomposition and then re-composition of attributes and objects primitives [35, 36]. Due to CLIP (a large pre-trained vision-language models (VLMs)), state-of-the-art CZSL methods have came into hands [37].

Focus to CZSL after ZSL is shifted in prospect of adding more meanings amid classification of unseen or adding certainty to uncertain. Further to this spree CSP [38] added the hard prompt template of CLIP, i.e., an image of [attribute][object], where a static statement being added with composed attributes and objects.

Besides, CoOp [39] introduced soft prompts, leading to better CZSL performance, by adding meaningful / relevant prompt with attributes and objects. Notwithstanding the effectiveness of current CLIP-based techniques, this study identifies a number of crucial factors to achieve context-aware compositions for improved CZSL.

These contributions in the field of CZSL diverted it towards making it context aware by adding more semantics in a way that attributes and objects appear with complete sentence level description.

1.4.2 Context Aware Attribute-Object Guided CZSL

This thesis introduces Context Aware Attribute-Object Guided Compositional Zero-Shot Learning framework that focuses on two things to make CZSL more semantically correct. First, inter-class and intra-class entanglement among the concepts, and secondly, making CZSL composition more contextual in term of having compositions along with complete sentence level description.

Intra-class entanglement refers to the dependencies that exist among attributes within the same object category. Attributes are often not independent but rather influence one another, and combining them without considering their relationships can lead to semantically invalid descriptions [40]. For instance, consider the object "apple." An apple can be green, ripe, or sliced, but it cannot logically be described as both whole and sliced at the same time.

Similarly, if an apple is sliced, it cannot simultaneously be described as whole, and if it is green, it is unlikely to also be ripe. These attribute relationships highlight how some attributes are mutually exclusive while others may co-occur, meaning that intra-class entanglement must be taken into account to prevent contradictions [41]. Ignoring these dependencies results in invalid predictions such as "whole sliced apple," which do not align with real-world semantics. (Figure 1.2).

Both intra-class and inter-class entanglement present significant challenges for CZSL [40]. Addressing these entanglements is therefore essential to ensure that generated attribute-object compositions are realistic and contextually meaningful. By modeling intra-class entanglements, the system avoids logically impossible combinations, while accounting for inter-class entanglements ensures that the context-dependent meaning of attributes is preserved. This allows CZSL models to produce predictions that are not only accurate but also more consistent with human understanding of language and perception, making them suitable for real-world applications such as image search, recommendation, and assistive AI systems.

FIGURE 1.2: Challenges of Compositional Recognition

To reduce entanglement in the visual domain, the Visual Feature Enhancer (VFE) is introduced. While CLIP's image encoder provides strong baseline features, these embeddings often contain background noise and overlapping cues that can mislead attribute–object recognition. VFE refines these raw features by emphasizing discriminative patterns such as color, texture, and shape that directly correspond to attributes, while suppressing irrelevant visual signals. This focused enhancement reduces intra-class confusion (e.g., distinguishing between "red apple" and "green apple") and ensures that visual features align more consistently with their textual counterparts [41].

On the language side, the Text Feature Enhancer (TFE) addresses entanglement by refining the embeddings produced by CLIP's text encoder. Although the encoder captures semantic meaning of attributes and objects, it may overlook subtle distinctions that are critical for compositional reasoning. TFE strengthens these embeddings by highlighting fine-grained semantic cues and filtering redundant or overlapping signals. For example, it helps the model separate closely related terms like "broken" and "cracked" or "striped" and "patterned." This prevents inter-class confusion and ensures that textual features carry sharper, context-aware meanings for accurate composition alignment [41, 42].

Generalization of concepts learned by decomposition of visual concept is made more difficult because of entanglement between visual concepts as depicted in Figure 1.2 [42]. However, [43] learns to decompose and then re-compose the primitives after enhancement of features to get more clarity among concepts.

This is CLIP-based approach. It uses large language models (LLMs) to improve class embeddings and construct class-specific distributions in order to have diversity and information amid textual class representations. Similarly a technique, Visual-Language-Primitive-Decomposition (VLPD) module is also integrated to decompose image data into simplest primitives to recognize attributes and objects.

The Vision-Language Primitive Decomposition (VLPD) tackles entanglement at a deeper structural level by explicitly breaking down both visual and textual representations into their primitives—attributes and objects. Rather than learning compositions as indivisible labels (e.g., "wet dog"), VLPD decomposes them into separate but interacting primitives ("wet" + "dog"). This decomposition reduces redundancy, enables systematic recombination, and preserves context-dependent meanings of attributes. By aligning decomposed primitives across vision and language, VLPD ensures that the framework generates attribute–object pairs that are not only accurate but also semantically consistent with human perception, thus mitigating both intra- and inter-class entanglements [21]. In recent years, the rise of large-scale vision–language models such as CLIP [37] and generative language models such as OPT has introduced a new paradigm in guiding model

behavior through prompts and contextual knowledge. While both prompt and context engineering aim to enhance model output, their underlying mechanisms and goals differ. Understanding these distinctions is crucial for extending CZSL toward more explainable and human-centered applications[38].

Prompting [44] refers to the manual or automated design of natural language prompts that guide a model's generation or classification. For example, when using OPT to describe an attribute–object composition such as "broken phone", a prompt like "Describe a broken phone in detail" can influence the model's output toward producing detailed, coherent descriptions. Prompt engineering therefore controls how a model is asked a question or given an instruction. However, it is limited by the quality of the crafted prompts and may lead to shallow or repetitive outputs if not augmented with external knowledge.

Context Engineering [45], on the other hand, focuses on enriching the input with relevant background information or external knowledge before passing it to the model. In the case of CZSL, context engineering can be implemented using a retrieval-augmented generation (RAG) pipeline, where the predicted composition (e.g., "broken phone") is linked to retrieved knowledge such as "A broken phone usually has a cracked screen and may stop functioning". This context is then injected into the prompt, enabling the model to generate more humanized and factually grounded descriptions. Unlike prompt engineering, which optimizes the phrasing of instructions, context engineering ensures that the semantic richness and factual grounding of the input are enhanced.

The significance of these two approaches in a CLIP-based CZSL pipeline is twofold. First, prompt engineering ensures that language models produce coherent and structured responses tailored to the attribute–object composition. Second, context engineering ensures that the responses are not only coherent but also semantically enriched and factually aligned with real-world knowledge. When combined, they transform CZSL outputs from simple label predictions (e.g., "broken phone") into human-readable, contextualized explanations (e.g., "The phone appears broken with a cracked display, which usually requires repair"). Thus, in the proposed framework, prompting plays the role of shaping linguistic structure, while context

engineering provides the knowledge grounding necessary to ensure semantic validity. Together, they bridge the gap between recognition and explanation, enabling a Context-Aware CZSL system capable of generating human-friendly outputs.

1.5 Problem Statement

Compositional Zero-Shot Learning (CZSL) has two major issues: **First**, current CZSL models struggle to distinguish and combine fundamental components of attribute-object pairs due to inter-class and intra-class entanglement between concepts that lacks the semantics of predicted pairs. **Second**, CZSL models find it challenging to contextualize predicted attributes and objects compositions through sentence level description.

1.6 Research Objectives

- To develop a framework that resolves inter-class and intra-class entanglement in attribute and object concepts using feature decomposition techniques. [**RQ1**]
- To extend the proposed framework for generating meaningful sentence-level explanations of predicted attribute-object pairs.[**RQ2**]

1.7 Research Questions

- How inter-class and intra-class entanglement can be reduced in CZSL so that predicted attribute-object pairs are semantically valid ?
- How CZSL predictions can be contextualized vide sentence level description?

Chapter 2

Literature Review

2.1 Chapter Overview

This chapter reviews the progress and challenges in CZSL, a field that focuses on teaching machines to recognize and reason about novel attribute–object combinations that were not seen during training. The review begins by examining the foundational studies that introduced the problem of compositional generalization and established benchmark datasets such as MIT-States and UT-Zappos. It then explores key methodological directions, including attribute–object embedding techniques, graph-based reasoning, and recent advances in vision–language pretraining. Special attention is given to the problem of intra-class and inter-class entanglement, where attributes may conflict within the same object or carry different meanings across objects, making prediction more complex. By analyzing prior approaches and their limitations, this chapter highlights why context-aware methods are needed to move beyond simple pair prediction toward richer and more human-like understanding. Ultimately, this review provides the background necessary to situate our proposed framework within existing research and to show how it addresses critical gaps in current CZSL methods. This chapter therefore serves as a bridge between existing literature and the methodological novelty.

2.2 Compositional Zero-Shot Learning

CZSL is about utilization of prior knowledge to generalize it on unknown attributes and objects [22], unlike ZSL.

To assist computers with recognizing objects they have never seen before, researchers have created two primary methods: one focuses on integrating known components, such as attributes and objects, while the other focuses on comprehending the connections between them [31].

The first approach, referred to as composition-based [31], attempts to create new combinations (such as "furry chair") from existing ones, but it frequently overlooks the more profound relationships between the image and its components, which makes it less accurate in situations that are unknown.

Although the second, relationship-based [32] approach is more effective in recognizing the relationships between attributes and objects as well as the image, it may not capture the ways in which various components affect one another.

In order to overcome these problems, researchers are currently trying to integrate the two approaches to capitalize on their advantages and create a system that is more precise and adaptable.

2.2.1 Composition Based Methods

The focus of this approach is to learn from seen to unseen compositions. The most widely used approach teaches attributes and objects independently, then combines them during reasoning.

In [46], Misra et al. presented CZSL by engaging the weight vector of SVM for embedding of visual primitives and generalizing to unseen compositions by altering the embeddings of visual concepts. This work proposes a context-aware method to compose visual classifiers for complex concepts (e.g., "red tomato") from primitive classifiers (e.g., "red" and "tomato") without external knowledge.

First, linear SVMs are trained on base visual primitives (attributes, objects, or verbs) using CNN features (VGG-M fc7). A transformation network T—a 3-layer MLP with LeakyReLU activations—is then trained to map pairs or triples of primitive classifier weights (e.g., w_{red}, w_{tomato}) to a new classifier for their composition (e.g., $w_{\text{red tomato}}$). Crucially, T is optimized using a binary cross-entropy loss that directly evaluates the composed classifier's performance on images, ensuring the output discriminates the complex concept (e.g., "red tomato" vs. unrelated images). The network is initialized with identity blocks to enable meaningful initial transformations. At test time, T synthesizes classifiers for unseen combinations of known primitives (e.g., "small elephant") or even unseen primitives (e.g., a new "raspberry" classifier composed with "red") through zero-shot inference[46].

However, this approach [46] has several limitations: (1) It relies on the eminence of the base primitive classifiers, as poor base classifiers propagate errors to composed concepts; (2) training requires a subset of composed concepts (e.g., some [attribute, object] pairs), which may be scarce for rare combinations; (3) the identity initialization is heuristic and may not generalize to all architectures; (4) the computational cost increases with primitive types as the input size to T grows linearly; (5) the method only uses visual features and overlooks the potential advantages of multimodal (e.g., textual) context for disambiguation.

In [47], Nagarajan et al. converted object features into number grids that act on object word lists, and represent the process of combining them changes these grids to affect the word lists. In this work, attributes (such as "red" or "sliced") are treated as mathematical operators that change objects. The vector of an object is altered by each attribute, which is a learned matrix (for example, "apple" $\rightarrow$ "sliced apple"). A neural network is used to map images and compositions (attribute-object pairings) to a common space. The model is trained with (1) a triplet loss to align images with their correct compositions; (2) linguistic rules (e.g., "blunt" undoes "sharp"; attributes commute like "red sliced = sliced red"). At test time, unseen compositions (e.g., "ripe persimmon") are created by applying attribute matrices to object vectors. The limitations in this work [47] were: (1) word reliance: uses predefined word meanings (GloVe), not visual details; (2)

rare object: fails on objects with few examples, for instance, "hat"; (3) simple math: assumes attributes change objects with basic math, but real changes can be complex, for instance, "ripe" vs. "unripe"; (4) single attribute: cannot combine multiple attributes, for instance, "small red apple".

In [48], Chen et al. By applying attribute classifiers to a variety of objects based on the idea of shared attributes, object-specific attribute models can be created without the need for object-specific training data. The attribute classifiers developed in this research (such as "striped" or "fluffy") are tailored to particular objects (e.g., "striped dog" vs. "striped shirt"). Using some labeled photos, it first trains simple classifiers with an emphasis on same-object instances (e.g., dog images for "fluffy dog"). It thus saves time and money by predicting new classifiers for unknown combinations (like "striped dog") using mathematical patterns (tensor factorization) without the need for additional annotated photos.

The limitations of this research work [48] were: (1) needs some starting labeled data to work; (2) struggles if objects/attributes are too different, e.g., "metallic cat" might fail if only "metallic car" is known; (3) depends on initial classifiers' quality—errors can spread; (4) works best with related objects, e.g., different dog breeds.

In [49], Li et al. the guidance of group axioms, suggested learning symmetry principles for attribute-object transitions. They proposed the SymNet paradigm, which handles attributes (like "peeled") as if they were actions (like "adding" or "removing") on objects (like "apple"). SymNet guarantees logical transformations by applying group theory concepts (such as symmetry and invertibility); for instance, adding "peeled" to an apple that has previously been peeled does not alter it. A coupling network is used to add attributes, and a decoupling network is used to remove them. In order to classify attributes robustly for recognition, it calculates Relative Moving Distance (RMD), which is the amount that an object's representation changes when an attribute is added or removed.

Limitations of this research [49] work were: (1) oversimplified transformations: only handles binary attributes (e.g., "peeled" or "not peeled"), ignoring degrees

(e.g., "half-peeled"); (2) synonym confusion: struggles with similar attributes (e.g., "old" vs. "ancient"), hurting accuracy; (3) theoretical compromises: skips strict adherence to some group theory rules (e.g., associativity) for practicality; (4) data dependency: performance drops on noisy or fine-grained datasets (e.g., MIT-States vs. UT-Zappos).

In [50], Wei et al. used a Generative Adversarial Network (GAN) to generate attribute-object compositions. Compared to direct sample synthesis, adversarial learning captures more unique attribute and object properties. By using created adversaries to test the model, it improves feature discriminability. In order to achieve fine-grained zero-shot learning, this approach trains neural language models from scratch using raw text descriptions (such as "red wings with a black beak") rather than human attributes. They gather a fresh collection of human-written descriptions of flowers and birds (CUB). By optimizing compatibility between matching pairs (such as an image and its description) and eliminating mismatches, a symmetric joint embedding model (DS-SJE) aligns text and images. They evaluate text encoders at the word and character levels, including CNNs, LSTMs, and hybrid CNN-RNNs (convolutional layers + recurrent layers).

The limitations of this research [50] were: (1) data dependency: requires costly human annotations (10 descriptions per image); (2) character-level under performance: character-based models (e.g., Char-LSTM) lag behind word-level ones; (3) embedding compactness: text embeddings are less efficient than manual attributes (e.g., retrieval AP@50: 48.7% vs. 50.0% on CUB); (4) vocabulary constraints: struggles with entirely new concepts outside training vocabulary.

In [51], Saini et al. visual similarities using spatial features to identify attributes and objects, while [52] used cross-attention between paired images with the same concept (attribute or object) to detect their features. In order to distinguish between object (like "flower") and attribute (like "yellow") features in photos, this work developed ADE (Attention as Disentangler). In Vision Transformers (ViT), they employ cross-attention between image pairs that have the same item or attribute (for example, two "yellow" objects). Regularization based on an EMD compels attention to concentrate solely on the target notion (for example,

"yellow" rather than "flower"). They combine composition, object, and attribute probabilities to make predictions during inference.

The limitations of the work [51] were: (1) scalability: slow for large attribute/object sets (e.g., C-GQA has 413 attributes × 674 objects); (2) semantic ambiguity: fails when the same word means different things (e.g., "open curtain" vs. "open computer"); (3) backbone dependency: relies on ViT; weaker with CNNs like ResNet; (4) computation cost: training requires triplets (target + attribute-sharing + object-sharing images), increasing complexity; (5) lack of context awareness.

2.2.2 Relationship Based Methods

These techniques often project text-image compositions onto a common area, learning the underlying similarity metrics as well as their relationships, in order to guarantee generality.

In [53], Purushwalkam et al. Created a flexible, task-focused system to model images, objects, and attributes together. The authors propose a "Task-Driven Modular Network" (TMN) to handle both the variety of sub-concepts and the relationships between combined concepts at the same time. It makes use of tiny, reusable neural modules, or mini-networks, that are managed by a gating mechanism. The gating system activates different combinations of modules based on the task (the specific object-attribute pair you want to check). In order to assess how well an image matches a new object-attribute pair, the model can "compose" knowledge (for example, by reusing "wrinkled" and "envelope" modules). It has been trained to assign high scores to triplets that are correct (picture, object, attribute).

Even though it [53] performed better than previous approaches, the overall accuracy (as determined by AUC) was still quite low (e.g., 3.5% on MIT-States), indicating that the issue is still very difficult to solve. The model may have trouble if entirely new primitive items or attributes (not in the training vocabulary)

are introduced because it depends on predefined sets of objects and attributes encountered during training. Their recommended "generalized" evaluation technique, which tested on both visible and unseen pairs, also showed that earlier approaches were not as reliable as previously believed.

In [54], Yang and Cheng et al. discovered particular sub-concepts in each category after breaking down image data into attribute and object categories using labels. Three different kinds of mixed samples were then merged from these categories to comprehend integrated concepts in a single, cohesive space. HiDC (Hierarchical Decomposition-and-Composition) is the solution suggested in this research. Using distinct neural networks, it first breaks down a picture (such as "young tiger") into distinct attribute ("young") and object ("tiger") features. These features are assigned to different "subspaces." It then creates new ideas in three different ways: (1) visual: word embeddings are combined with visual features; (2) hybrid: visual and word features are combined; (3) word: visual and word features are combined. It employs "adaptively semi-positive" ideas to increase accuracy, modifying loss margins according to concept similarity (for example, treating "young cat" as more similar to "young tiger" than "young horse").

However, this work [54] outperformed previous methods but: (1) only handles two-part compositions (attribute + object), not more complex concepts; (2) performance drops on datasets with few training examples per concept (e.g., MIT-States); (3) while better than past methods, accuracy remains low (e.g., ∼15% on MIT-States), showing the task is still challenging.

In [55, 56], Naeem and Mancini et al. suggested the use of a graph convolutional network (GCN) in the Compositional Graph Embedding (CGE) technique. [57, 58] to investigate a new topic called Open-World CZSL (OW-CZSL) and discover the relationships between characteristics and objects in the composition space. [32], something the majority of earlier works disregarded. Compared to CZSL, OW-CZSL performs larger and more adaptably in both visible and invisible compositions. In order to decrease the output space's size for OW-CZSL attribute and object prediction, Karthik et al. [59] improved performance by removing infeasible compositions during testing and estimating the feasibility scores of compositions

using an external knowledge base. As vision-language models (VLMs) have become more popular, numerous studies [60–62] recently implemented the CLIP [63] model to the CZSL domain as a feature extractor or prompt module, with comparatively good results.

The CZSL challenge has long been examined in the literature from a vision-based standpoint. They can break down the visual qualities into more basic primitives like attributes and objects, or they can learn the compositional visual aspects directly. In [64, 65], For direct classification, for example, the compositional visual characteristics are projected into a common feature space, and in [35, 42, 66], to achieve compositional recognition, the visual features are decomposed into basic primitives, from which re-composition can be learned. The paper suggests SymNet to manage attribute-object combinations (such as "peeled apple"), drawing inspiration from group theory. Two neural networks are used: (1) a coupling network (CoN) "adds" an attribute to an object (e.g., adds "peeled" to "apple"); (2) a decoupling network (DecoN) "removes" an attribute (e.g., removes "peeled" from "peeled apple"). These networks adhere to the rules of symmetry: an object shouldn't be altered by adding an attribute it already has or taking away one it doesn't. Group-theory rules such as convertibility—which states that adding and then deleting "peeled" yields the original—are enforced by training. It employs Relative Moving Distance (RMD) for recognition: to determine whether an object possesses an attribute, it analyzes the distance it "moves" in feature space while adding versus removing that attribute.

While this method [64, 65] : (1) struggles with synonyms (e.g., "ancient" vs. "old" are confused visually and linguistically); (2) cannot handle graded attributes (e.g., "half-peeled" or "very old")—only binary (has/doesn't have); (3) performance drops on noisy datasets like MIT-States due to data scarcity per concept; (4) relies on predefined attributes/objects and can't generalize to unseen primitives.

CLIP-based methods have taken over the CZSL task as a result of recent prompt learning [67–69]. Strong zero-shot compositional generalization has been demonstrated empirically through the use of the frozen CLIP model to build textual

embeddings of basic primitives. This study uses learnable prompts and LLM-generated text for distributional alignment to address the importance of diversity and informativeness for zero-shot generalization, in contrast to [70, 71] that create crude adapters and [37, 72, 73] that use deterministic vision-language alignment with learnable cues.

2.3 Vision-Language Models

Vision Language Model is pre-trained on web based datasets, Vision-Language Models (VLMs) like CLIP [35] have recently drawn a lot of attention due to their potent zero-shot identification performance on a variety of downstream tasks. Usually, prompt engineering is used to modify previously trained VLMs in order to accomplish this capacity. The template "a photo of [CLS]" is used as the input in early prompting techniques like the hard prompt in CLIP. In order to improve model adaptation efficiency, soft prompt tuning procedures now leverage learnable embeddings as the textual context of class names.

In [74], Zhou and K. Yang et al. introduce a technique for automatically learning more effective cues for vision-language models like CLIP, called Context Optimization (CoOp). CoOp [39] uses learnable vectors to substitute descriptive words like "a photo of a" rather than handwriting prompts like "a photo of a [CLASS]". While maintaining the pre-trained model frozen, these vectors are improved using a small number of labeled samples (as low as 1–16 images per class). One common prompt for all classes ("unified context") and distinct prompts for each class ("class-specific context") are the two variations that are tried. The limitations of the work [74] were: (1) hard to interpret: the learned prompt vectors do not correspond to clear English words, making it difficult to understand what the model learned; (2) sensitive to noise: performance drops on datasets with messy labels (e.g., Food101), suggesting CoOp [39] struggles with label errors. CoOp automates prompt-tuning with data-driven vectors but sacrifices interpretability and robustness to noisy labels. In [75], Zhou and J. Yang et al. proposed Conditional

Context Optimization (CoCoOp), which enhances CoOp [39] by making prompts dynamic and image-specific. CoCoOp adds a lightweight "Meta-Net" that creates a distinct prompt token for every input image in place of fixed learnable vectors (like CoOp). By combining this token with the base context vectors, prompts can now adjust to specific images instead of being the same for all classes. However, generating per-image prompts makes CoCoOp much slower and more memory-intensive than CoOp [39] limiting batch size during training, and on 7 out of 11 datasets, CoCoOp's accuracy on unseen classes still trails zero-shot CLIP (using hand-crafted prompts), showing room for improvement.

In [76], Razdaibiedina, A., Mao et al. provide Residual Prompt Tuning, which adds a residual link to the learnable prompt embeddings to improve on traditional prompt tuning. It adds the outcome back to the original embedding after passing each prompt token through a small MLP network (with down/up projections). This process of "re-parameterization" increases the flexibility and stability of training prompts. The main model remains frozen; only the small MLP and prompt tokens are adjusted. However, this work requires longer input sequences due to prepended prompts, the MLP adds slight overhead during training (though discarded afterward), and it still trails full fine-tuning (e.g., 7.8 points lower on SuperGLUE).

In [77], M. Khan, S. Khan et al. proposed MaPLe, which improves CLIP (a vision-language AI) by adding learnable "prompts" (small adjustable inputs) to both its image and text branches simultaneously. Unlike prior methods that only tweak text prompts, MaPLe links image and text prompts via a coupling function, forcing them to work together. These prompts are added to multiple early layers of CLIP, allowing gradual refinement of features. Only the prompts are trained (not the whole model), making it efficient. This method behaves sensitively in the initial stages and excels on niche datasets (e.g., satellite/texture images) but offers smaller improvements on common categories (e.g., everyday objects) where CLIP already performs well.

Annotated description learning for multi-models like both text and image made

possible by MaPLe. However, because to its deterministic cues and lack of diversity to capture the appearance variance in fine-grained visual data, these methods are prone to overfitting the training data. ProDA [78] addresses this problem by explicitly introducing soft prompting in complete sets that triggered Gaussian distribution related to each class. This improves zero-shot performance and motivates PPL [79] to achieve recent success in the dense prediction challenge.

In [78], Lu and Y. Liu et al. proposed ProDA, a method that uses text clues (also known as "prompts") to enhance AI's image comprehension. ProDA automatically learns a large number of different clues from a small number of example photos, as opposed to relying on a single fixed text hint that might not cover all permutations of an object. It accomplishes this by modeling these clues as a Gaussian distribution (similar to a cluster of points), concentrating on the meaning (output embeddings) of these hints rather than the words themselves, and employing a unique training technique to eschew intricate computations. By positioning the object name in various places inside the clue and giving each suggestion a unique meaning, it also promotes the diversity of the hints. The primary drawback is that ProDA is only designed for image identification tasks, which include identifying the objects in a picture. Other crucial computer vision tasks, such as identifying the locations of objects (detection), defining objects (segmentation), or altering image styles, cannot be done with it directly.

In [79], Kwon and H. Song et al. proposed PPL, a technique to improve AI's comprehension of images for intricate tasks like object detection and segmentation. PPL generates several probabilistic prompts that describe universal qualities (such as color and position) that are shared by all objects, as opposed to using a single fixed text prompt for each item. It models a distribution of potential text descriptions (as a Mixture of Gaussians) by combining these characteristics with class names and visual context from the image. A customized "probabilistic pixel-text matching loss," which minimizes inaccurate predictions brought on by uncertainty in complicated scenes, is calculated using many prompts taken from this distribution. The method may not be immediately applicable to other vision tasks. Image synthesis, because it is specifically made for dense prediction tasks

(segmentation/detection). Additionally, compared to single-prompt approaches, sampling several prompts increases computational overhead, which may restrict real-time use.

2.4 Vision and Language Multimodal Learning

Learning to correlate the picture's content with the written description's content in image-text learning. (for example, "cat" in the picture and "cat" in the text). This model that mine the links between different modalities must be properly constructed. Many studies on attention mechanisms [80] and Transformer [81–83] have sparked a lot of interest in transformer-based multimodal techniques. An overview of the current status of research on cross-modal embedding strategies in multimodal learning is mostly given in this section.

2.4.1 Embedding of Visual and Text Features

Combining different modal information, such as text and visual, into a shared embedding space allows for the learning of richer and more comprehensive knowledge. In [84], Akata et al. identified some drawbacks of direct image-based prediction methods. These include the need to train attribute classifiers and class classifiers in two stages, the difficulty of incorporating extra side information, and the inability to fine-tune using label data.

The technology known as Attribute Label Embedding (ALE) was introduced by [84]. The attribute vector space represents each class, and the alignment of picture and label embeddings is assessed using a compatibility function. This method presents the task of label embedding for attribute-based image categorization. In order to classify photos, the researchers suggested a technique called "Attribute Label Embedding" (ALE), particularly for object classes that lack training instances (zero-shot learning). For example, ALE interprets each class as a vector

of its attributes (e.g., "has stripes," "has paws") rather than predicting attributes separately and then combining them (like prior approaches).

It [84] learns a compatibility function that assigns a score based on how well an image matches the attribute vector of a class. It modifies this function during training to guarantee that accurate image-class combinations outperform wrong ones. In addition, the approach can substitute text descriptions or class hierarchies for attributes.

However, this method struggled with: (1) interpretability: ALE doesn't optimize for accurately predicting individual attributes (unlike older methods), making it harder to understand why the model assigns certain attributes to an image; (2) dependence on side information: performance relies heavily on the quality of the provided attributes, hierarchies, or text data—if this information is noisy or incomplete, accuracy drops; (3) performance variability: attributes worked best for zero-shot learning, while class hierarchies and text descriptions (like Word2Vec) gave weaker results; (4) limited gain with more data: when plenty of training images are available, simpler methods (like one-vs-rest classifiers) performed as well as or better than ALE, reducing its advantage.

In [85], Frome et al. proposed a visual semantic embedding model, DeViSE, which utilizes image data and large-scale text data for joint training. A linear transformation layer is used to project visual features into the textual feature space in order to train the model.

This research [85] work created a model called DeViSE (Deep Visual-Semantic Embedding Model) that blends language comprehension and image recognition. In order to learn word meanings as vectors, they first trained a language model on Wikipedia text (for example, "shark" and "tiger shark" have similar vectors). They separately used ImageNet photos to build a deep neural network (similar to AlexNet).

Lastly, they linked these models: a ranking loss that pushes correct labels closer to the image than random words was used to retrain the image network to predict the word vector of an object's name (for example, mapping a shark photo to the

"shark" word vector). By comparing image vectors to word vectors, the model was able to detect new objects (also known as "zero-shot"). Large amounts of unlabeled text data are used to extract semantic information, and the SKIP-GRAM technique is used to learn word vector representation [86, 87] to improve textual and visual information's capacity for semantic relationship mining. To learn word meanings as vectors, the researchers developed two effective models (CBOW and Skip-gram) (e.g., "king" → [0.2, -0.5, 0.7]). Whereas Skip-gram predicts "sat" from "cat," CBOW predicts a word from its surrounding context (e.g., "cat" from "the _ sat"). Both link similar words (such as "dog" and "puppy") to nearby vectors using shallow neural networks that have been trained on billions of pieces of text. Using parallel computing and hierarchical softmax to speed up training is one of the main advancements.

CNN is used to retrieve image features, and visual information is extracted from annotated picture classification data. For example, Reed et al. [88] proposed an end-to-end deep symmetric structured joint embedding model that uses convolutional neural networks (CNNs) to map text descriptions and images to a common multimodal embedding space [89], Long-term memory (LSTM) and recurrent neural networks (RNNs) [90] to become familiar with picture and text description embedding representations.

2.4.2 Vision and Text Cross Modal Reasoning

Measurement of the relevance between a language query and images is crucial in image retrieval tasks, and multimodal image techniques are more popular than other approaches for this reason. Recent notable works in image feature association and inference include VirTex, a pretraining technique that employs semantically dense captions to learn visual representations, proposed by Karan et al. [54], and CLIP, an effective pretraining method for image-text comparisons, proposed by Radford et al. [16]. Without anticipating the precise description text of the images, By evaluating and contrasting how well photos match associated written descriptions, CLIP directly learns image representations. After a user inputs a

search term, CLIP computes picture features, extracts text features, and assesses how relevant the candidate images are to the search query. In order to reduce labeling requirements and make inference predictions for subsequent visual tasks, CLIP uses ViT, an image encoder that is more efficient than ResNet [55]. VirTex relies on easily obtained image-text pairs for pretraining rather than requiring humans to densely label images. The majority of visual-verbal multimodal tasks now in use rely on pretrained visual and verbal models, according to Lu et al. [19].

CLIP Under natural language direction, it picks up visual notions. By using images and their written descriptions to jointly train a CLIP model. CLIP is a groundbreaking endeavor that jointly learns text and image embeddings through contrastive pretraining between language and vision [91]. AI models may learn to recognize the similarities and differences among several data points with the use of a technique known as contrastive learning. CLIP handles image-text data as positive pairings when they occur together and as negative pairs when they exist separately. The goal is to determine which embedding similarity is better for positive pairings and which is worst for negative pairings. Assume that a human has three items: a separate item ('negative sample"), a comparable item ("positive"), and a major item ("anchor sample") [92]. As seen in, the objective is to help the model recognize that the anchor and the positive item are similar, which causes it to mentally pull them closer together while acknowledging that the negative item is different and pushing it away. Figure 2.1.

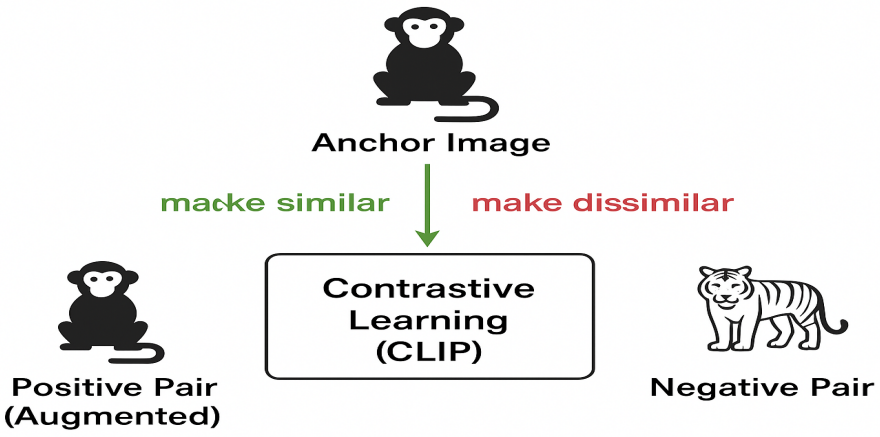

FIGURE 2.1: Contrastive Learning Methods

Training a model, like a convolutional neural network, to distinguish disparate visual representations and bring similar ones closer together is the aim of contrastive learning in computer vision. While a "negative" image would be completely different, usually from another category (e.g., cats), a similar or "positive" image might be from the same category (e.g., dogs) or a modified version of the primary image [92].

CLIP maps text and images into a common latent space using a dual-encoder architecture, as shown in Figure 2.2. Two encoders are concurrently trained to make it function: one for text (a Transformer-based language model) and one for images (a Vision Transformer).

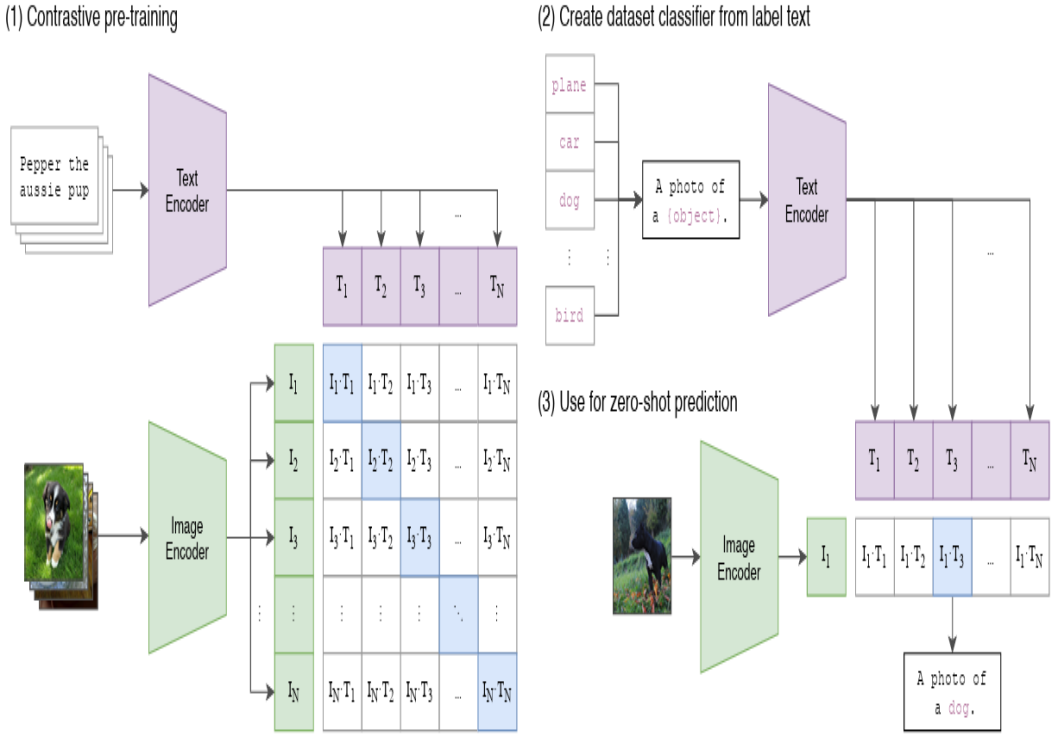

FIGURE 2.2: Overview of CLIP Architecture

Important features are extracted from the visual input by the image encoder. Using an image as input, this encoder generates a high-dimensional vector representation. Usually, a convolutional neural network (CNN) [89] architecture, such as ResNet [93], is used to extract information from images. The semantic meaning of the associated textual description is encoded by the text encoder. It creates

another high-dimensional vector representation using a text caption or label as input. It frequently processes text sequences using a transformer-based design, such as a Transformer or BERT. A common vector space is used by the two encoders to generate embeddings. Underlying relationship of text and image can be compared by CLIP, thanks to these shared embedding spaces [91].

The model is shown image-text compositions during pre-training. While some of these pairs are mismatched, others are true matches. Shared latent space embeddings are produced. Multiple text descriptions—including the correct and incorrect ones—are generated for every image. As a result, a variety of positive and negative sample pairs are produced. Class-specific embeddings are produced by the text encoder using these descriptions as input. One important function also comes into play at this point: the Contrastive Loss Function. When the model matches pairs (image-text) poorly, this function penalizes it. However, it is rewarded when it matches pairs correctly [94].

Now, a zero-shot classification is applied to the learned text encoder. CLIP is capable of making zero-shot predictions with a new image. This is achieved by passing it through the image encoder and dataset classifier without making any changes. The cosine similarity between each pair of image and text description embeddings is calculated using CLIP. It improves the parameters of the encoders to increase the similarity of the correct pairs. As a result, erroneous couples have less resemblance. In this way, Semantically related words and images are mapped close to one another in the multimodal embedding space that CLIP learns. The highest logit value is found in the predicted class. [91].

The zero-shot image categorization capability of CLIP is among its most remarkable capabilities. AI models are trained on specially labeled datasets for conventional image classification tasks, which restricts their capacity to identify objects or situations that are outside of their training domain. CLIP allows the model to be given descriptions in natural language. As a result, it doesn't need particular training in those categories to classify and generalize photos based on textual input. CLIP uses contrastive learning to learn by aligning images and their text descriptions (for example, "zebra" with a zebra image) into a shared semantic space

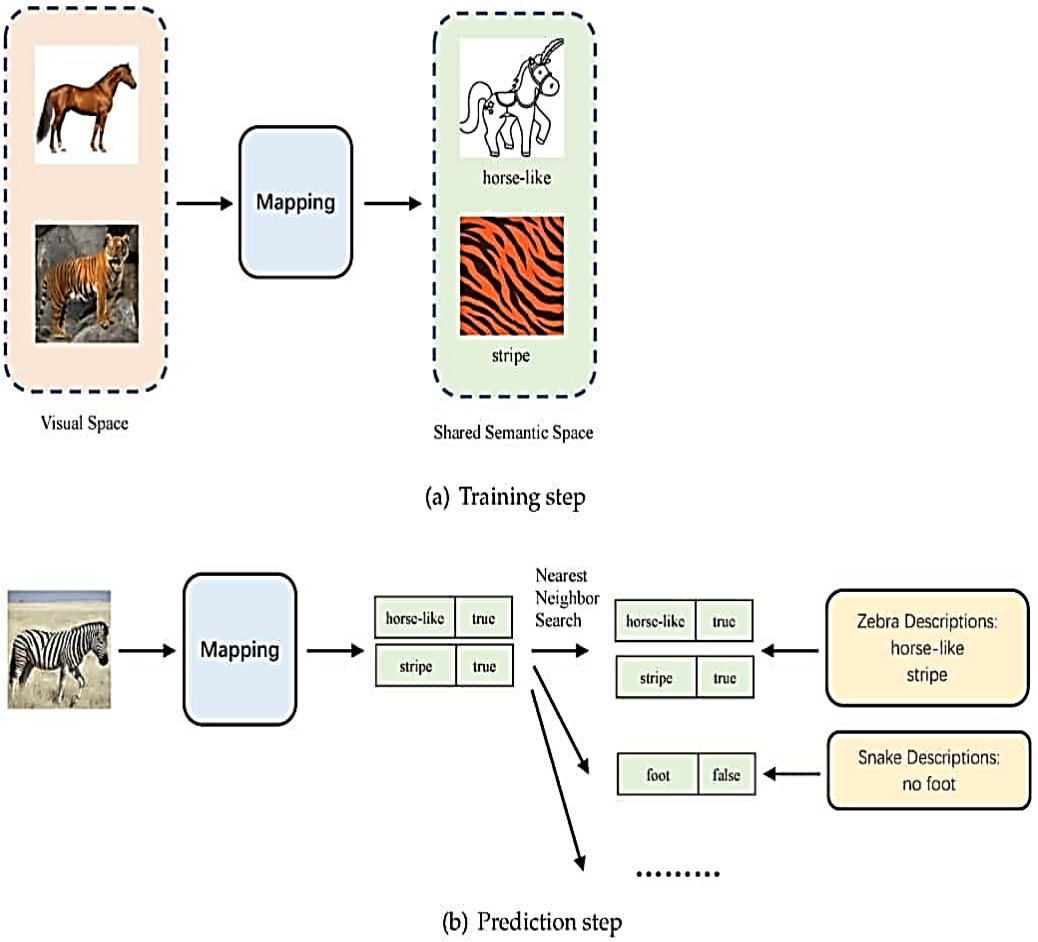

FIGURE 2.3: CLIP Training and Prediction Steps

[91, 92]. It implicitly learns visual concepts such as "horse-like" and "stripe" by pushing matching pairs (e.g., zebra image + "zebra" word) close together while pulling non-matches apart, as shown in Figure 2.3. It also encodes a new image into the visual space. The vectors of possible text prompts (such as "zebra: horse-like stripe" and "snake: no foot") in the same space are then compared to this image vector (using nearest neighbor search). The prediction is "true" for a correct match and "false" for an incorrect one; this is determined by the closest text description (greatest similarity), as shown in Figure 2.3. This ability to connect visual content with flexible textual prompts allows CLIP to perform recognition far beyond closed-set conditions. Instead of depending on fixed class labels, the model can dynamically interpret new categories using semantic cues embedded in language. This property makes CLIP especially powerful for Zero-Shot Learning tasks, where the goal is to recognize unseen concepts [91].

2.5 Summary of Key Literature Review

Key literature review is summarized in Table 2.1.

TABLE 2.1: Summary of Key Literature Review

Ref	**Method**	**Key Idea**	**Limitations**
Misra et al. [46]	SVM / MLP	Composes classifiers for unseen AO pairs using primitive embeddings.	Relies on strong base classifiers; requires some AO pairs; ignores multiple cues due to entanglement of concepts.
Nagarajan et al. [47]	Attribute as Operator	Attributes act as matrices transforming object embeddings ("red apple" = red × apple).	GloVe reliance; fails for rare objects; simple linear transforms; limited multi-attribute capability.
Chen et al. [48]	Attribute Transfer via Tensor Factorization	Learns shared attributes transferable to unseen objects.	Needs initial labeled data; struggles with dissimilar AO pairs; error propagation.
Li et al. (SymNet) [49]	Group Theory Inspired Symmetry Network	Attributes as reversible transformations on objects.	Only binary attributes; synonym confusion; ignores graded attributes.
Wei et al. [50]	GAN-based CZSL + Text Embeddings	Generates AO pairs via GAN + text-image alignment.	Heavy annotation (10 desc/img); poor text embeddings; vocabulary limitations reduce model effectiveness.

Table 2.1 continued from previous page

Ref	Method	Key Idea	Limitations
Saini et al. [51]	Attention as Disentangler (ADE)	Uses ViT + cross-attention to disentangle AO features.	Scalability issues; semantic ambiguity; backbone dependency; expensive training.
Purushwalkam et al. (TMN) [52]	Task-Driven Modular Networks	Reusable mini-networks + gating for AO compositions.	Low accuracy (3.5% MIT-States); fails on unseen primitives.
Yang et al. (HiDC) [53]	Hierarchical Decomposition-Composition	Decomposes images into AO subspaces, recombines with hybrid embeddings.	Only AO pairs; low accuracy (∼15%).
Naeem / Mancini et al. [55]	GCN-based Compositional Graph Embedding (OW-CZSL)	Models AO relations in graph space and extends to OW-CZSL.	Data dependency; difficulty with unseen primitives; scalability issues.
Karthik et al. [59]	Feasibility Filtering in OW-CZSL	Filters infeasible AO pairs via external KB.	Depends on KB quality; may remove rare valid pairs.
Manicini et al.[60]	CLIP for CZSL	Uses frozen CLIP for AO embeddings + prompt tuning.	Sensitive to prompt design; lacks context-awareness.
Zhou et al. (CoOp) [74]	Context Optimization (Prompt Learning)	Learns prompts instead of hand-crafted templates.	Non-interpretable; noisy labels hurt performance.

Table 2.1 continued from previous page

Ref	Method	Key Idea	Limitations
Zhou et al. (CoCoOp) [75]	Conditional CoOp	Image-specific dynamic prompts via Meta-Net.	Slower; memory-heavy; weaker on unseen cases.
Razdaibiedina et al. [76]	Residual Prompt Tuning	MLP-based residual reparameterization of prompts.	Longer input sequence; training overhead; worse than full fine-tuning.
Khan et al. (MaPLe) [77]	Joint Prompting (Image + Text)	Couples image and text prompts in CLIP.	Unstable early training; smaller gains on common datasets.
Lu et al. (ProDA) [78]	Probabilistic Prompt Distribution	Learns Gaussian-distributed diverse prompts.	Limited to classification; not generalizable to detection/segmentation.
Kwon et al. (PPL) [79]	Probabilistic Pixel-Text Matching	Gaussian prompt mixtures for segmentation/detection.	High computation; unsuitable for generative tasks.
Akata et al. (ALE) [84]	Attribute Label Embedding	Embeds class labels as attribute vectors; learns compatibility score.	Relies on side information; not interpretable; drops with noisy attributes.
Frome et al. (DeViSE) [85]	Deep Visual-Semantic Embedding	Joint CNN + Skip-gram semantic embedding.	Needs large-scale image–text data; dependent on semantic richness for better Image-Text combinations.

2.6 Contextual Compositional Zero-Shot Learning

A prompt is an input to a Generative AI model, that is used to guide its output. Prompts may consist of text, image, sound, or other media [92]. The most straightforward version of CoT contains zero exemplars. It involves appending a thought inducing phrase like "Let's think step by step" [94] uses LLMs to generate "Let's work this out in a step by step way to be sure we have the right answer". [95], searches for an optimal thought inducer. Zero-Shot CoT approaches are attractive as they don't require exemplars and are generally task agnostic.

Step Back Prompting is a modification of CoT where the LLM is first asked a generic, high-level question about relevant concepts or facts before delving into reasoning. This approach has improved performance significantly on multiple reasoning benchmarks for both PaLM-2L and GPT-4 [96].

Analogical Prompting is similar to SG-ICL, and automatically generates exemplars that include CoTs. It has demonstrated improvements in mathematical reasoning and code generation tasks [97]. Tabular Chain-of-Thought consists of a Zero-Shot CoT prompt that makes the LLM output reasoning as a markdown table. This tabular design enables the LLM to improve the structure and thus the reasoning of its output [98].

Zero-shot prompting uses zero exemplars. There are a number of well known standalone zero-shot techniques. In [99], Role prompting also known as persona prompting assigns a specific role to the Gen AI in the prompt. For example, the user might prompt it to act like "Madonna" or a "travel writer". This can create more desirable outputs for open-ended tasks and in some cases may improve accuracy on benchmarks.

Style prompting [**?**] involves specifying the desired style, tone, or genre in the prompt to shape the output of a GenAI. A similar effect can be achieved using role prompting. Emotion Prompting [100] incorporates phrases of psychological

relevance to humans (e.g., "This is important to my career") into the prompt, which may lead to improved LLM performance on benchmarks and open-ended text generation.

System 2 Attention (S2A) [101] first asks an LLM to rewrite the prompt and remove any information unrelated to the question therein. Then, it passes this new prompt into an LLM to retrieve a final response.

SimToM [102] deals with complicated questions which involve multiple people or objects. Given the question, it attempts to establish the set of facts one person knows, then answer the question based only on those facts. This is a two prompt process and can help eliminate the effect of irrelevant information in the prompt.

Rephrase and Respond(RaR) [103] instructs the LLM to rephrase and expand the question before generating the final answer. For example, it might add the following phrase to the question: "Rephrase and expand the question, and respond". This could all be done in a single pass or the new question could be passed to the LLM separately. RaR has demonstrated improvements on multiple benchmarks.

Re-reading (RE2) [104] adds the phrase "Read the question again:" to the prompt in addition to repeating the question. Although this is such a simple technique, it has shown improvement in reasoning benchmarks, especially with complex questions.

Self-Ask [105] prompts LLMs to first decide if they need to ask follow up questions for a given prompt. If so, the LLM generates these questions, then answers them and finally answers the original question.

In order to direct reasoning, lessen hallucinations, and enhance performance across tasks, the field of context engineering focuses on organizing, selecting, and dynamically managing the input context for language models. This includes prompts, retrieved knowledge, dialogue history, metadata, and external signals.

LLMs retain information in parameters, despite the fact that many operations call for fresh, validated, domain-specific evidence. Early RAG systems combined neural retrievers and generators to ground outputs in recovered passages. The

retrieve-then-generate loop was formalized by Retrieval-Augmented Generation (RAG) [106]; Fusion-in-Decoder (FiD) [107] improved the use of evidence by enabling the decoder to combine several documents; REALM [108] jointly trained retrieval with the LM, while Dense Passage Retrieval (DPR) [109] showed how dual-encoder retrieval might replace brittle keyword search. Together, this produced the traditional pipeline and current objectives, which include enhanced retrieval recall/precision, scalable serving, and faithful grounded generation.

The subsequent wave enhanced the formulation and selection of questions and supporting data. Retrieval quality was improved via hybrid dense+sparse retrievers (ColBERTv2 [110]), query rewriting/expansion, and iterative/multi-step retrieval (IRCoT; Self-Ask [111]). The most beneficial passages were consistently pushed into the context by re-ranking with cross-encoders (BERT re-ranker [8]), frequently without adjusting the base LLM.

Pre-retrieval (query rewriting/expansion), retrieval, post-retrieval (rerank/compress), and generation are interchangeable processes that make up modern RAG. Plug-and-play modules and subcomponents for A/B composition are exposed by toolkits like FlashRAG [112]. In an effort to reduce hallucinations, domain-specific variations increasingly combine vector search (KARE [113]) with knowledge-graph signals. Systems like as ComposeRAG [114] incorporate self-reflection over query rewriting and question decomposition. Instead of following a set recipe, the focus now is on creating the ideal assembly for every task and corpus.

Agentic approaches interleave planning, tool calls, retrieval, and self-criticism rather than "retrieve once, then generate." ReAct [115] combines actions and reasoning traces; Self-RAG [116] teaches models to determine when to retrieve and to evaluate their own outputs; PlanRAG [117] anticipates information needs prior to retrieval; Reflexion [118] incorporates self-reflection to rectify errors; and CDF-RAG [119] completes the loop with RL-driven query refining and hallucination correction. As a result, retrieval time and content are dynamic and driven by task progress and uncertainty. Knowledge was re–framed as graphs of entities and relations, which was a significant step. HippoRAG [120] traverses multi-hop evidence using Personalized PageRank; RAPTOR [121] constructs hierarchical

summary trees for retrieval at many granularities; GraphRAG [122] popularized hierarchical/community-aware retrieval; and LightRAG [123] combines graph and vector retrieval. Clearer reasoning pathways, enhanced multi-hop synthesis, and reduced context drift on challenging tasks are among the advantages.

Deployed RAG must keep up with changing corpora, low latency, and tight context budgets. Advances include dynamic retrieval during generation [124], context compression (LLMLingua [125], LongLLMLingua [23]) to fit strict windows, efficient retrieval (ColBERTv2) and high-performance vector indexing with FAISS [126]. These ensure accuracy gains survive real workloads.

Chapter 3

Research Methodology

3.1 Chapter Overview

This chapter will discuss the proposed framework for Compositional Zero-Shot Learning (CZSL) task. Our approach is blend of Contrastive Language-Image Pre-training (CLIP) [79], Visual Language Primitive Decomposition (VLPD) modules, Retrieval Augmented Generation (RAG) [116] and integration of Large Language Model (LLM) i.e OPT 1.3 [94] to create a compact framework. This framework aims to identify new attribute and object composition (such as "wet dog" or "sliced tomato") that have not appeared during training of model and adding context by sentence level explanation of predicted composition. Traditional models encounter difficulties in generalizing to such unseen compositions due to entanglement of concepts / inter and intra classes variance. Our approach aims to bridge these gaps by improving compositional generalization.

The methodology combines visual and textual representations with retrieval-based mechanisms to capture semantic relationships and provide contextual explanations for predicted compositions. Its modular design allows easy extension with language or vision models that capable handling of unseen AO combinations.

3.2 Proposed Research Methodology

The overall research methodology adopted for this study is illustrated in Figure 3.1, which highlights the systematic flow of activities undertaken to achieve the research objectives. Each stage is designed to build logically on the previous one, ensuring a structured and valid approach towards solving the problem.

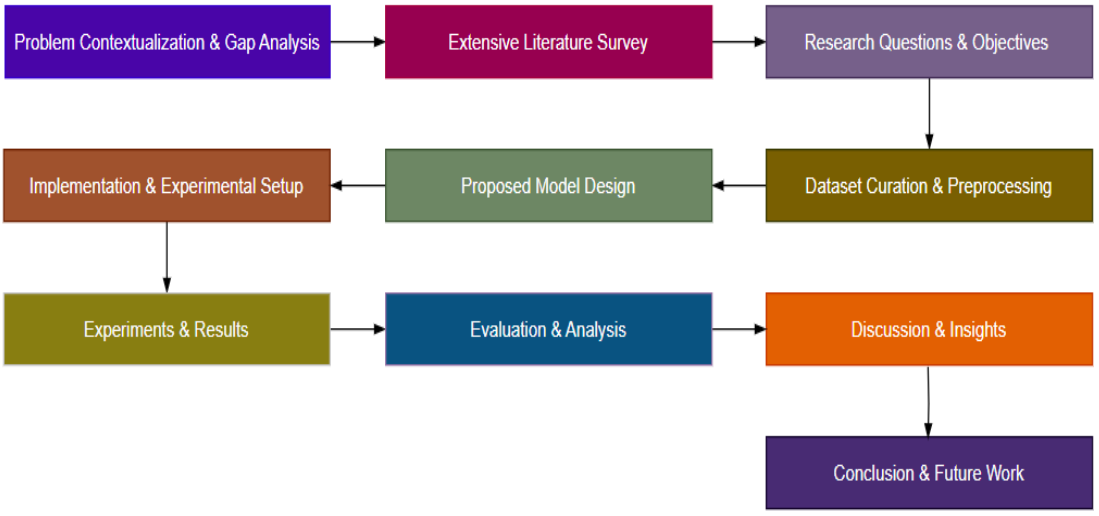

FIGURE 3.1: Propose Research Methodology

The research begins with establishing the context, identifying the importance of Zero-Shot and Compositional Zero-Shot Learning (CZSL), and exploring their limitations in handling human attribute–object recognition. This provides the foundation for the research motivation.

A detailed review of existing work is conducted to understand current advancements, challenges, and gaps in CZSL and related fields. This step ensures that the proposed work builds on prior research while addressing unexplored areas. From the literature analysis, specific limitations in handling unseen attribute–object compositions, contextual awareness, and real-world adaptability are identified. This step defines the research gap that the proposed study aims to fill.

Based on the problem statement, precise research questions and objectives are formulated. These guide the research direction, keeping the study focused and measurable. Appropriate datasets are selected (e.g., MIT-States, UT-Zappos).Preprocessing involves cleaning corrupted images, normalizing naming conventions,

resizing to standard dimensions, and filtering valid attribute–object pairs. Structured metadata is then prepared to enable efficient training and evaluation.

The central contribution of the research is the proposed model. This integrates visual feature extraction (e.g., Vision Transformers or CLIP), textual embedding (attribute–object descriptions), and contextual modules for better compositional understanding. The architecture is designed to handle both seen and unseen pairs effectively.

The proposed model is trained and evaluated under both closed world and open world CZSL settings. Performance metrics such as Seen Accuracy (S), Unseen Accuracy (U), Harmonic Mean (H), and AUC are computed. Comparisons with baseline models (e.g., CLIP, CoOp, ProDA) demonstrate the effectiveness of the proposed approach.

Quantitative results are supported by ablation studies, analyzing the effect of different components (e.g., contextual awareness module, data augmentations). Qualitative evaluation is also performed by visualizing predictions for unseen compositions.

The findings are interpreted in light of the research objectives. The impact of contextual awareness, dataset characteristics, and compositional generalization is discussed. This section also highlights limitations and challenges encountered.

A summary of the contributions, findings, and insights gained is provided. The conclusion reinforces how the proposed methodology addresses the identified research gap. Finally, directions for future research are outlined, including extensions to multi-modal data, real-world deployment, and further enhancements in context engineering for CZSL. This structured workflow ensures that each stage of the research is logically connected and contributes meaningfully toward addressing the identified gap. Moreover, the systematic progression from literature exploration to model evaluation enhances the clarity and rigor of the overall study. In doing so, the methodology establishes a strong foundation for validating the proposed framework's effectiveness in real-world CZSL scenarios which ensures reliability of Attribute-Object composition.

3.3 Proposed Framework

The Figure 3.2, illustrates the overall workflow of the proposed framework. The process begins with raw datasets, which first undergo preprocessing to prepare the visual and textual data. On the visual side, images are converted into visual embeddings, which are then refined through a Visual Feature Extractor (VFE) and further processed by a visual decomposition module to separate attribute and object information. Similarly, on the textual side, labels or descriptions are transformed into text embeddings, which are enhanced using a Text Feature Extractor (TFE) and decomposed into meaningful attribute and object components.

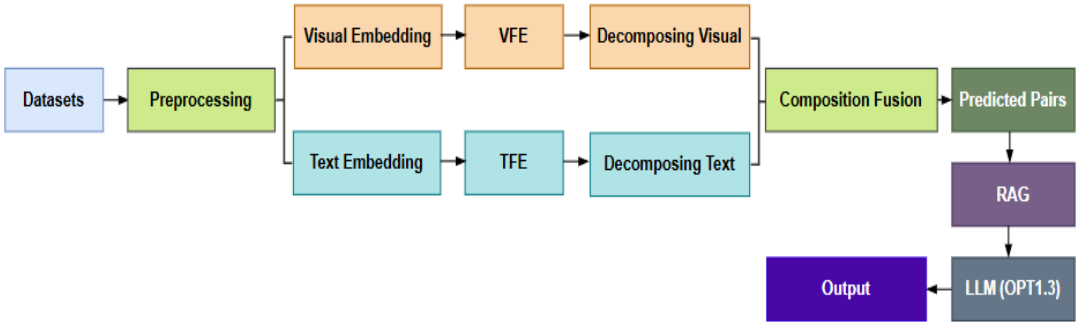

FIGURE 3.2: Proposed Framework

Both streams are then combined in a composition fusion module, producing candidate predicted attribute–object pairs. To enrich these predictions with better context and reasoning, the system integrates a Retrieval-Augmented Generation (RAG) step, which passes information to a large language model (LLM – OPT1.3). Finally, the framework generates a more context-aware output that combines visual recognition with language-driven explanation, making the results more human-like and interpretable.

This approach ensures that the predicted compositions are not only visually accurate but also semantically meaningful. Moreover, it provides a foundation for future improvements by allowing easy incorporation of additional knowledge sources or more advanced model architectures. This combination of visual and linguistic reasoning enables the system to produce outputs that align more closely with human interpretive patterns. As a result, the framework not only enhances prediction quality but also strengthens the overall robustness of compositional understanding.

3.4 Datasets

In order to evaluate the proposed framework, this research used two benchmark datasets. MIT-States and UT-Zappos. Each dataset possesses unique challenges that make them suitable for assessing the ability of the framework to handle attribute-object compositions, as well as generalization to unseen pairs.

3.4.1 MIT-States Dataset

MIT-States dataset contains 53,753 images, annotated with 245 unique object categories and 115 unique attribute categories, resulting in approximately 24,725 attribute-object compositions. The dataset primarily consists of natural objects (e.g., "car," "dog," "apple") annotated with attributes such as broken, red, sliced, metallic, etc as shown in Figuare 3.3.

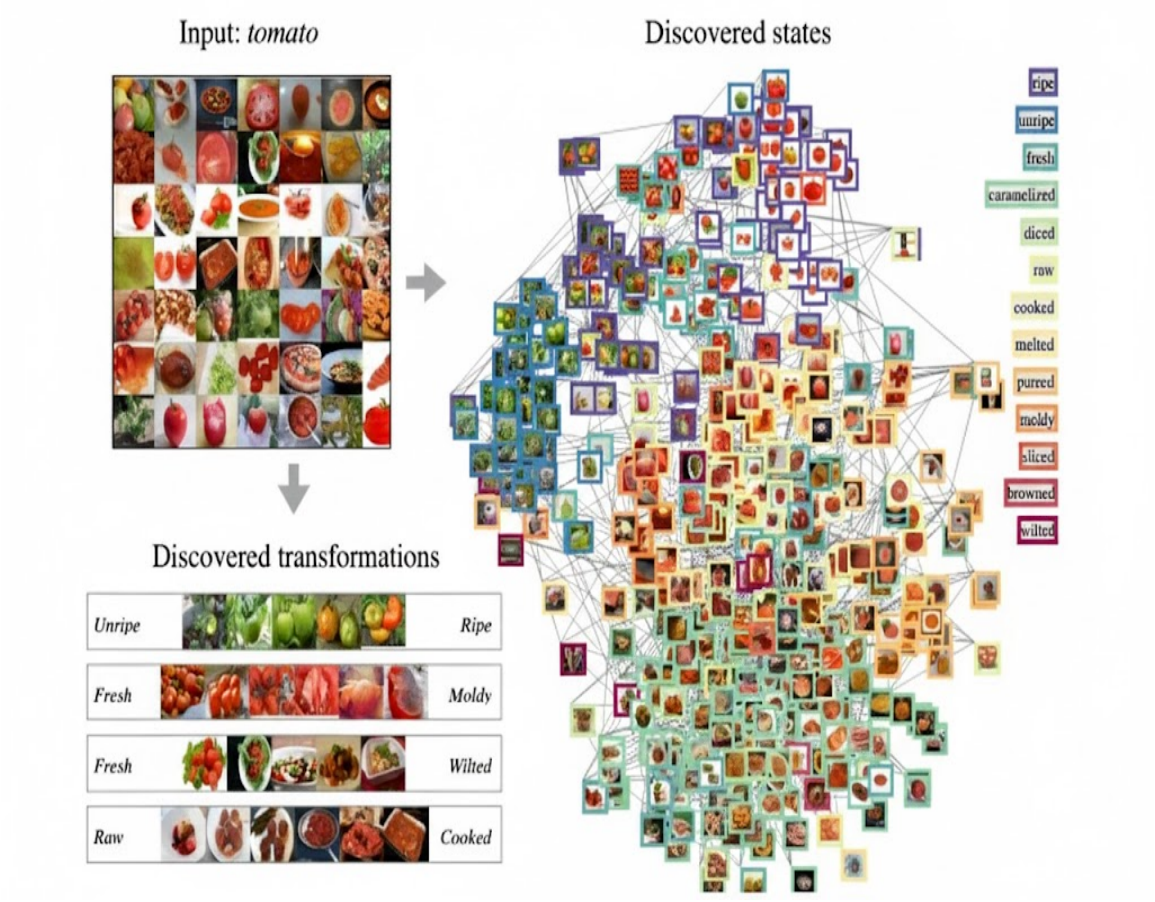

FIGURE 3.3: MIT-States Dataset Sample

3.4.2 UT-Zappos Dataset

The UT-Zappos dataset is a fine-grained shoe dataset as shown in Figure 3.4, containing 50,025 images of shoes annotated with four meta-categories (shoes, sandals, slippers, boots). The dataset focuses on fine-grained attribute recognition, such as pointy, sporty, black, leather, and combinations like black boots, leather sandals.

FIGURE 3.4: UT-Zappos50K Dataset Sample

3.5 Data Preprocessing

Before model training and evaluation, dataset preprocessing is a crucial step to ensure consistency, quality, and compatibility with the proposed architecture. Raw data usually contains variations in size, format, and noise, which can hinder the learning process. Therefore, preprocessing prepares both the visual and textual components of the dataset. On the visual side, images are resized, normalized, and converted into visual embeddings, which are further refined through the VFE and decomposed into separate attribute and object representations. On the textual side, labels and descriptions are embedded into vector form, processed by the TFE, and decomposed into meaningful components.

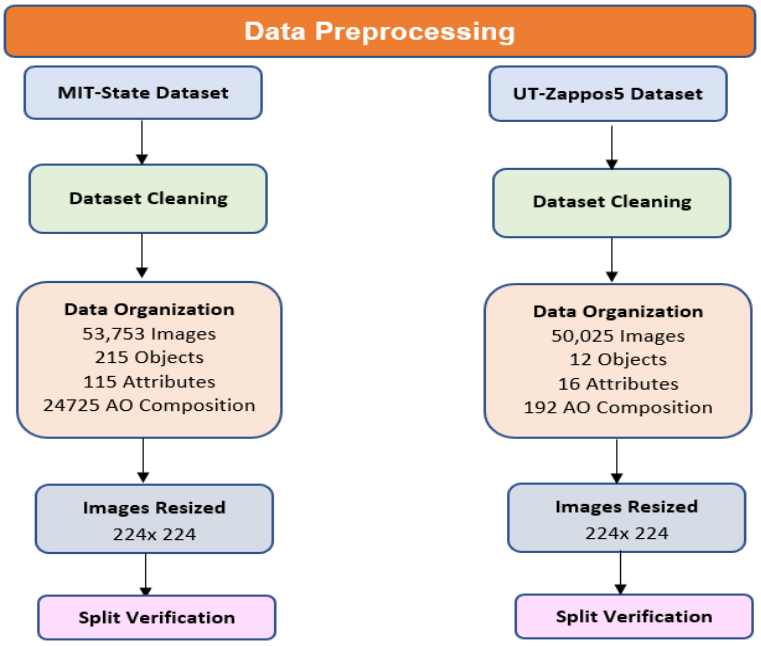

FIGURE 3.5: Data Preprocessing

As shown in Figure 3.5, these parallel streams of preprocessed visual and textual data are then fused in a composition module, which creates candidate attribute–object pairs. This pre-processing stage ensures that the model not only receives uniform and high-quality inputs but also that both modalities (image and text) are aligned for meaningful composition. This alignment between modalities strengthens the model's ability to recognize subtle attribute–object relationships during training. Ultimately, such careful preprocessing enhances both the reliability and overall performance of the downstream CZSL framework.

3.5.1 Preprocessing of MIT-State

3.5.1.1 Dataset Cleaning

To ensure consistency in dataset, following necessary to populate before implementation of proposed methodology;

- Removed corrupted and unreadable images using PIL checks.
- Normalized naming / labels conventions (replacing spaces with underscores).
- Verified Attribute–Object (AO) Pairs: only attribute–object combinations listed in official splits were retained.
- Attribute–Object Pair Filtering: Since not all attribute–object combinations are valid (e.g., *sliced sky*), pairs are filtered according to the provided train, validation, and test splits.

3.5.1.2 Dataset Organization

- The raw dataset comes with image folders and three split files: `train_pairs.txt`, `val_pairs.txt`, and `test_pairs.txt`. It defines the paradigm of training, validation and testing of the model. Each `.txt` file contains Attribute–Object (AO) compositions in the format, for instance: *red apple*, *sliced bread*.
- This organized structure ensures efficient data loading during training and evaluation, enabling the model to consistently access attribute–object pairs along with their corresponding visual samples.
- Preprocessing script parsed this dataset splits and mapped each AO pair to its corresponding image path, creating a structured `CSV/JSON` file with fields:
  - `is_seen` $\rightarrow$ 1 for training pairs
  - `is_seen` $\rightarrow$ 0 for unseen evaluation pairs

3.5.1.3 Image Preprocessing

- All images were resized to 224×224 pixels using bilinear interpolation to ensure compatibility with the CLIP and ViT backbones.

3.5.1.4 Text Preprocessing

- Attributes and objects were lowercased and stripped of special characters.
- To embed the text into vector space, CLIP's text encoder was primarily used. For baselines, GloVe (300-d) embeddings were also generated using `gensim`.
- Attribute–object pairs were wrapped into natural language prompts using sentence-level descriptions, e.g., "a photo of a {attribute} {object}", "an image showing a {attribute} {object}".
- This helped align the textual descriptions with the visual domain.

3.5.1.5 Split Verification

- A Python script ensured that no attribute–object (AO) pair from test pairs appeared in the training data.
- This maintained the strict CZSL protocol, where models must generalize to unseen attribute–object combinations.

3.5.2 Preprocessing of UT-Zappos

3.5.2.1 Dataset Cleaning

To ensure consistency in dataset, following necessary to populate before implementation of proposed methodology that preparatory step guarantees that all

data components are uniformly structured, allowing the methodology to operate smoothly and without ambiguity;

- Removed corrupted and unreadable images using `PIL` checks.
- Normalized file naming conventions (replacing spaces with underscores, e.g., `blue_sandal.jpg`).
- Verified attribute–object (AO) pairs: only attribute–object combinations listed in official splits were retained.
- Attribute–Object Pair Filtering: Since not all attribute–object combinations are valid (e.g., *sliced shoe*), pairs were filtered according to the provided train, validation, and test splits.

3.5.2.2 Dataset Organization

- Since Zappos is primarily object-centric (footwear), attributes such as *color*, *material*, and *style* are paired with shoe categories to form compositional labels.
- Similar to MIT-States, pairs are split into train, validation, and test sets with a strict division between seen and unseen pairs.
- Each attribute–object (AO) composition was parsed and matched with image paths. A structured metadata CSV was generated in the following format: `image_path, attribute, object, pair_label, is_seen`.
- This ensured efficient data loading during training.

3.5.2.3 Image Preprocessing

- Images were resized to 224×224 pixels and normalized using the ImageNet mean and standard deviation, consistent with MIT-States. This standardization ensures uniform visual input quality across all samples and stabilizes feature extraction. This strict separation preserves the integrity of the

zero-shot evaluation setting and prevents any inadvertent data leakage. As a result, the model's performance truly reflects its ability to generalize to novel attribute–object combinations.

3.5.2.4 Text Preprocessing

- Attributes and objects were tokenized and embedded using CLIP's text encoder.
- To make textual prompts more contextual, multiple templates were employed, such as: "a close-up of a {attribute} {object}", "a {attribute} {object} for fashion".
- This ensured better alignment between text and subtle visual cues.
- Handling Class Imbalance: UT-Zappos contains certain over-represented attributes (e.g., black shoes) and under-represented ones (e.g., red slippers). To address this, sampling strategies are applied to balance training.

3.5.2.5 Split Verification

UT-Zappos preprocessing ensured that unseen AO pairs (from `test_pairs.txt`) had no overlap with training pairs. This was implemented with Python assertions in the preprocessing script, ensuring consistency across experiments.

3.5.3 Common Preprocessing Across Datasets

- Dataset's Split: The dataset is divided into training, validation, and testing subsets as depicted in Table 3.1 to ensure proper evaluation and generalization of the model. This division allows the model to learn from seen compositions while being evaluated on both familiar and novel attribute–object pairs. Proper splitting is essential to accurately measure performance metrics and compositional generalization.

TABLE 3.1: Datasets Split for MIT-States UT-Zappos

Dataset	**Att**	**Obj**	**Training**		**Validation**			**Testing**		
			Cs	I	Cs	Cu	I	Cs	Cu	I
UT-Zappos	16	12	83	23k	15	15	3k	18	18	3k
MIT-States	115	245	1262	30k	300	300	10k	400	400	13k

3.5.4 Final Data Loader Implementation

After pre-processing both datasets, custom PyTorch Dataset and Data Loader classes were implemented to efficiently feed data to the model:

- Each sample returned a tuple (image tensor, attribute embedding, object embedding, pair label, is seen), providing both visual and textual inputs along with metadata.
- Data loading was optimized using GPU memory pinning and pre-fetching to minimize bottlenecks during training.
- On-the-fly transformations (`torchvision.transforms.Compose`) ensured normalization were applied dynamically, supporting robust learning.
- Batch sampling was carefully controlled to maintain a balance between seen and unseen attribute–object pairs, preventing skewed training.
- Shuffling of training data was implemented at each epoch to improve generalization and reduce model overfitting.
- Data Loader was configured to handle variable batch sizes for experimentation and hyperparameter tuning.
- Collate functions were customized to correctly stack multi-modal embeddings and handle missing or corrupted samples gracefully.
- Logging and monitoring hooks were integrated to track the number of samples processed, batch composition, and data augmentation effects in real time.
- Additional prefetch queues and worker threads were employed to maximize throughput and minimize idle GPU time during training.

3.6 Proposed Model: CAAO-CZSL

To address the challenges of Compositional Zero-Shot Learning (CZSL) tasks, this research introduces a context-aware attribute–object guided model (Figure 3.6) that combines state-of-the-art vision–language models with feature decomposition and knowledge augmentation strategies. Traditional CZSL approaches often suffer from two critical shortcomings: first, they treat attribute–object compositions as independent entities, ignoring intra-class and inter-class entanglements; second, they fail to capture the broader semantic and contextual meaning of these compositions in real-world scenarios. As a result, predictions are often semantically valid in isolation but contextually misaligned, leading to errors such as predicting "sliced phone" when a "broken phone" is observed.

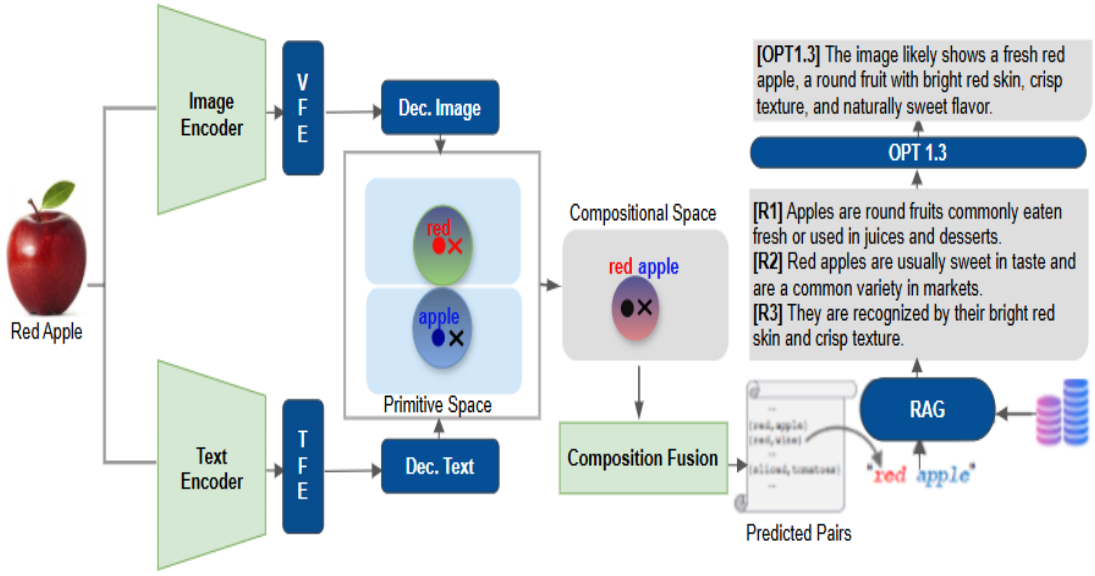

FIGURE 3.6: CAAO-CZSL Model

The proposed model is designed to mitigate these limitations by leveraging both vision–language pretraining and contextual reasoning. At its core, the model utilizes CLIP [35] as a dual encoder for extracting image and text embeddings, ensuring that both modalities are aligned within a shared representational space. These embeddings are further refined through the Vision–Language Primitive Decomposition (VLPD) [32] module, which explicitly disentangles and recomposes primitive (attributes and objects) to reduce entanglement errors. This decomposition–fusion model to reason about unseen attribute–object combinations more effectively.

To move beyond surface-level composition, the framework incorporates Retrieval-Augmented Generation (RAG) [45] for external knowledge grounding. Once a candidate composition is predicted, RAG retrieves relevant contextual information from an external knowledge base (e.g., textual descriptions of how certain attributes manifest in different objects). This external context is then provided to a large language model (OPT 1.3) [44], which produces a humanized sentence-level description of the composition. In this way, the framework not only predicts unseen attribute–object pairs but also explains them in a natural, interpretable manner, bridging the gap between machine predictions and human understanding. Together, these components form a context-aware CZSL pipeline that is robust, semantically consistent, and capable of producing interpretable outputs. The following subsections provide a detailed discussion of each component in the proposed framework, starting from CLIP-based feature extraction and decomposition to contextual fusion and knowledge-enhanced description generation.

3.6.1 CLIP Based Image Encoder

In this thesis, CLIP Image Encoder (ViT-B/16) is employed as the backbone for visual feature embedding. Each input image from the dataset resized and normalized according to CLIP's preprocessing pipeline before being passed through the encoder. The image was divided into fixed-size patches, each of which was linearly projected into a higher-dimensional feature space. These projected patches were enriched with positional embeddings and then processed through a multi-layer Transformer encoder. The final [CLS] token representation was extracted as the global image embedding, which served as the visual feature representation in our compositional zero-shot learning framework as depicted in Figure 3.7. This approach allows the model to capture both local and global visual patterns effectively, enhancing attribute–object recognition. Moreover, leveraging a pretrained CLIP encoder provides rich semantic features, improving generalization to unseen compositions. This approach allows the model to capture both local and global visual patterns effectively, enhancing attribute–object recognition.

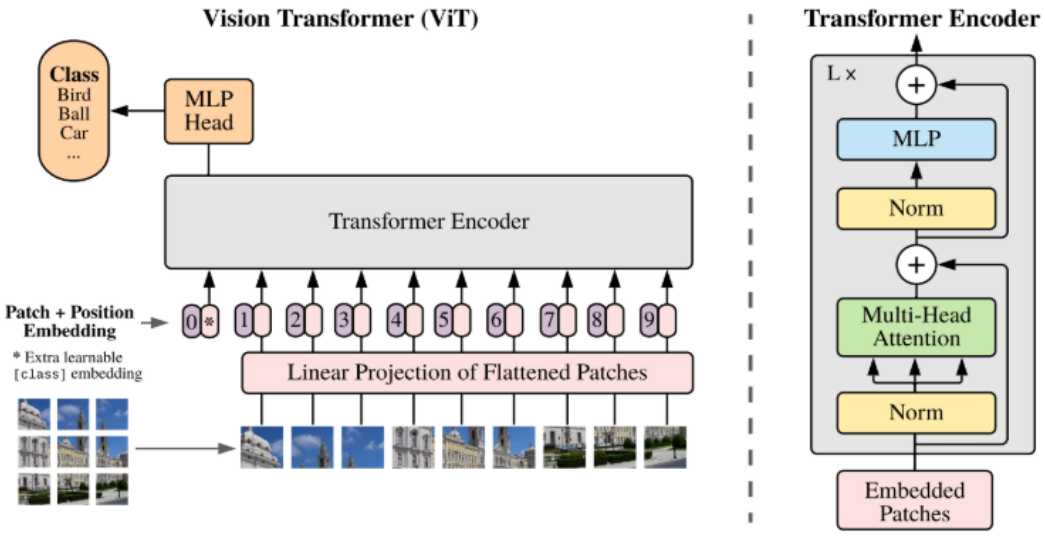

FIGURE 3.7: Image Encoder

3.6.1.1 Components of CLIP Image Encoder

The CLIP Image Encoder used in this work is based on a Vision Transformer (ViT) backbone. Its processing pipeline can be explained through the following components:

- Input Layer: The input image $I \in \mathbb{R}^{H \times W \times 3}$ is resized to 224×224 pixels and divided into non-overlapping patches of size 16×16. This results in a sequence of patches.

$$N = \frac{H \times W}{16^2} \tag{1}$$

- Patch Embedding Layer: Each image patch is flattened into a vector and linearly projected into a d-dimensional embedding space using a learnable projection matrix $E \in \mathbb{R}^{(16^2 \cdot 3) \times d}$. Formally:

$$z_0^i = x^i E, \quad i = 1, \ldots, N \tag{2}$$

where x^i is the i-th patch. Where each image patch is flattened into a vector and linearly projected into a d-dimensional embedding space using a learnable projection matrix $E \in \mathbb{R}^{(16^2 \cdot 3) \times d}$.

- Positional Encoding: To retain spatial information, a learnable positional embedding $p^i \in \mathbb{R}^d$ is added to each patch embedding:

$$z_0^i = z_0^i + p^i \tag{3}$$

- Transformer Encoder Layers: The sequence of patch embeddings passes through $L = 12$ transformer layers. Each layer consists of Multi-Head Self Attention (MHSA) and a Feed-Forward Network (FFN) with residual connections and layer normalization. For each layer l:

$$z_l = \text{FFN}(\text{MHSA}(z_{l-1})) \tag{4}$$

- Final Embedding: The output of the transformer is aggregated (via the [CLS] token) to form the final visual representation:

$$v = z_L^{[CLS]} \in \mathbb{R}^d \tag{5}$$

  This d-dimensional vector v represents the global image embedding used in the compositional zero-shot learning task.

3.6.2 CLIP Based Text Encoder

The CLIP Text Encoder in our proposed framework is utilized to generate semantic embeddings of attribute-object compositions (e.g., "broken phone", "red shoe"). Unlike traditional word embedding models such as Word2Vec or GloVe, which produce static word vectors, the CLIP Text Encoder leverages a Transformer-based architecture that encodes contextual meaning at the sentence level. This is particularly important for CZSL tasks, where the semantic nuance of an attribute can vary significantly depending on the object it modifies (e.g., "broken glass" vs. "broken phone"). During preprocessing, each attribute-object label from the dataset

is first tokenized using Byte Pair Encoding (BPE). The sequence of tokens is then passed through a multi-layer Transformer encoder (12 layers with 8 self-attention heads in the base version), where both syntactic and semantic dependencies are captured. The final hidden state corresponding to the special end-of-sequence token is extracted as the sentence-level embedding. This embedding resides in the same joint multimodal space as the CLIP Image Encoder output, enabling direct comparison and alignment between text and visual features.

By using CLIP's text encoder, we ensure that attribute-object compositions are represented in a contextualized, multimodal-aware embedding space, which is crucial for predicting unseen combinations in CZSL tasks.

3.6.2.1 Components of CLIP Text Encoder

The CLIP Text Encoder in our framework is based on a Transformer architecture (12 layers, 8 attention heads). It converts textual attribute-object descriptions into dense embeddings aligned with the image space. The main components are as follows. This alignment ensures that visual and textual representations occupy a shared semantic space, facilitating accurate composition matching. Additionally, the text encoder benefits from pretrained language knowledge, improving understanding of unseen or rare attribute–object pairs.

- Input Representation: Each textual input (e.g., "broken phone") is tokenized into a sequence of subword tokens using Byte-Pair Encoding (BPE). If the sequence length is M, the tokens are denoted as:

$$T = [t_1, t_2, \ldots, t_M] \tag{6}$$

- Token Embedding Layer: Each token t_i is mapped into a d-dimensional vector space using an embedding matrix $E_t \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the vocabulary size:

$$e_i = E_t[t_i], \quad i = 1, \ldots, M \tag{7}$$

- Positional Encoding: To capture word order, a positional embedding $p_i \in \mathbb{R}^d$ is added to each token embedding:

$$z_0^i = e_i + p_i \tag{8}$$

- Transformer Encoder Layers: The embedded sequence is passed through $L = 12$ transformer layers, each consisting of Multi-Head Self Attention (MHSA) and a Feed-Forward Network (FFN):

$$z_l = \text{FFN}(\text{MHSA}(z_{l-1})), \quad l = 1, \ldots, L \tag{9}$$

- Final Sentence Embedding: The hidden state corresponding to the special end-of-text token (`[EOS]`) is extracted as the global textual embedding:

$$t = z_L^{[EOS]} \in \mathbb{R}^d \tag{10}$$

This d-dimensional vector t represents the attribute-object composition in text space, aligned with the visual embedding v.

3.6.3 Visual Feature Enhancer

Visual Feature Enhancer (VFE) is the extra layer on top of CLIP Image Encoder to refine embeddings before they are recomposed. VFE is introduced in our framework to improve the quality and discriminative power of the visual embeddings produced by the CLIP Image Encoder. While CLIP provides strong general-purpose image features, they are optimized for broad visual-text alignment and may not fully capture fine-grained attribute-object variations required in Compositional Zero-Shot Learning (CZSL). For example, distinguishing between ripe apple" and sliced apple" requires subtle feature sensitivity. The VFE addresses these limitations by projecting and refining the visual features in a task-specific embedding

space. Technically, this module consists of a feed-forward projection layer followed by non-linear transformations and normalization. It enhances feature separability by emphasizing discriminative attribute-relevant cues while reducing redundancy.

- Input Features: The input to VFE is the visual embedding from the CLIP Image Encoder:

$$v \in \mathbb{R}^d \tag{11}$$

  where $d = 512$ for CLIP ViT-B/16.

- Projection Layer: Features are projected into a task-specific space using a linear transformation:

$$v' = W_v v + b_v \tag{12}$$

  where $W_v \in \mathbb{R}^{d' \times d}$ and $b_v \in \mathbb{R}^{d'}$. Typically, $d' < d$.

- Non-linearity and Normalization: To improve discriminative capacity, a non-linear activation is applied:

$$h = \sigma(v') \tag{13}$$

  followed by normalization:

$$\hat{h} = \text{Norm}(h) \tag{14}$$

- Residual Refinement (optional): Residual connections preserve general visual knowledge while refining task-specific cues:

$$z = \hat{h} + v \tag{15}$$

- Output Features: The refined feature embedding z is obtained, which is then fed into the Visual-Language Primitive Decomposition (VLPD) module for attribute-object decomposition. This step ensures that both attribute and object information are effectively separated and represented for accurate compositional prediction. This refined representation further strengthens the model's ability to generalize across unseen attribute–object combinations.

3.6.4 Text Feature Enhancer

CLIP text encoder provides powerful semantic embeddings for attribute–object pairs, these embeddings may still lack contextual richness when representing complex compositions. For example, the textual embedding of "broken phone" might not fully capture the nuanced difference between "broken phone" and "broken glass", since CLIP primarily encodes co-occurrence statistics learned during pre-training. To address this limitation, the Text Feature Enhancer (TFE) is introduced to refine and adapt the text embeddings for the specific compositional zero-shot learning (CZSL) task.

The TFE applies a lightweight transformation network on top of the CLIP text embeddings to enhance their discriminative power. Formally, if $t \in \mathbb{R}^{512}$ denotes the raw text embedding obtained from the CLIP transformer, the TFE transforms it as:

$$t' = \sigma(W_t t + b_t) \tag{16}$$

where:

- $W_t \in \mathbb{R}^{512\times512}$ is the learnable weight matrix.
- $b_t \in \mathbb{R}^{512}$ is the bias term.
- $\sigma(\cdot)$ is a non-linear activation function (ReLU in this case).

To further stabilize training, normalization is applied as:

$$\hat{t} = \frac{t'}{\|t'\|} \tag{17}$$

This normalized embedding $\hat{t}$ is then used in downstream composition and fusion modules. Normalization ensures that all embeddings lie on a consistent scale, preventing dominance of any single feature during training. This promotes stable

convergence and improves the reliability of similarity computations in the fusion modules.

- The TFE reduces semantic overlap between similar attributes across objects (e.g., "old chair" vs. "old man").
- It emphasizes context-dependent distinctions, allowing the model to generalize better to unseen attribute–object compositions.
- By refining text embeddings, the TFE ensures tighter alignment with visual features enhanced by the VFE.

3.6.5 Decomposition of Images and Text Through Vision Language Primitive Decomposition

After all, the main task of Compositional Zero-Shot Learning (CZSL) is to generalize seen knowledge over unseen and amidst this the fundamental challenge lies in disentanglement of concepts (attributes and objects).

The fundamental challenge in CZSL lies in the disentanglement of attributes and objects from raw image–text representations. Conventional methods often treat attribute–object pairs as holistic entities, without explicitly decomposing them into their primitive components. This results in poor generalization. To address this limitation, the proposed framework employs a Vision–Language Primitive Decomposition (VLPD) module, which systematically breaks down image and text embeddings into their respective attribute and object subspaces.

VLPD acts as a shared decomposition space between modalities. Both visual embeddings (from the CLIP Image Encoder + VFE) and textual embeddings (from the CLIP Text Encoder + VTE) are projected into this space and then disentangled into two branches as depicted in Figure 3.8.

In particular, $\mathbf{v}$ is broken down into the attribute visual feature $f_s(\mathbf{v})$ and the object visual feature $f_o(\mathbf{v})$ using two parallel neural networks, f_s and f_o. Given

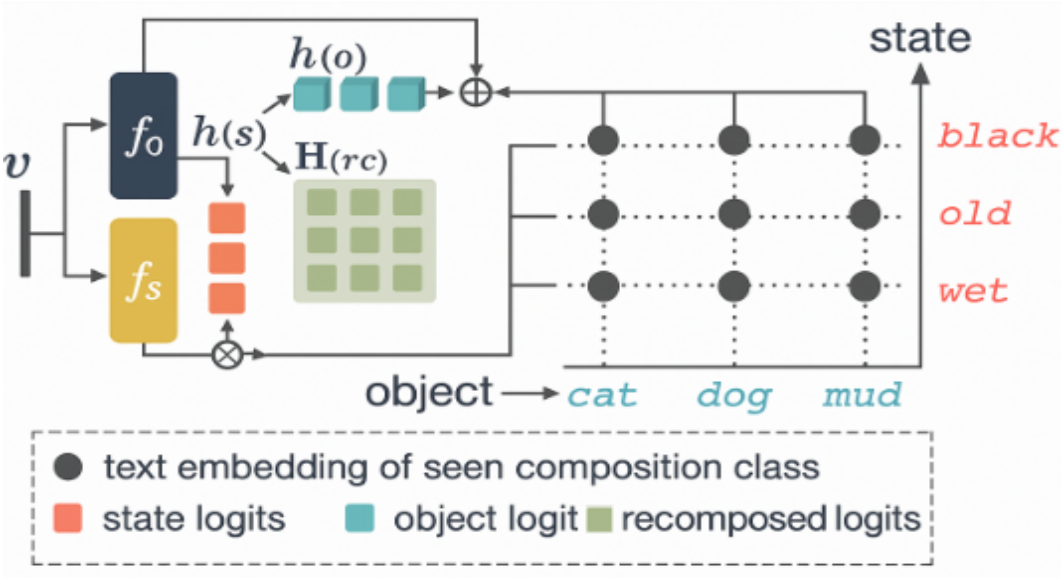

FIGURE 3.8: Vision-Language Primitive Decomposition

the training compositions $C(s)$ (as shown by the circle dots in Figure 3.8), we group their enhanced embeddings $\{\mathbf{t}_y\}$ over the subset y_o, where all compositions share the same given object o (as shown by the vertical ellipses in Figure 3.8), and group $\{\mathbf{t}_y\}$ over the subset y_s, where all compositions share the same given attributes (as shown by the horizontal ellipses in Figure 3.8). This allows us to obtain the primitive-level supervisions.

Consequently, the projected object logit h_s and attribute logit h_o given an attribute s and an object o are calculated by

$$h_s = \cos\left(f_s(\mathbf{v}), \frac{1}{|\mathcal{V}_s|}\sum_{y\in\mathcal{V}_s}\mathbf{t}_y\right) \tag{18}$$

$$h_o = \cos\left(f_o(\mathbf{v}), \frac{1}{|\mathcal{V}_o|}\sum_{y\in\mathcal{V}_o}\mathbf{t}_y\right) \tag{19}$$

We also employ f_s and f_o to breakdown visual characteristics $\mathbf{v}$, in contrast to DFSP [73], which only decomposes text features. We empirically demonstrate the superiority of conducting both visual and linguistic decomposition. The distributions over attribute and object categories are also introduced in keeping with the spirit of distribution modeling. The corresponding DSP, represented as $f(s)$

and $f(o)$, are produced by grouping $D(y)$ over $Y(s)$ and $Y(o)$, respectively. The upper-bounded cross-entropy losses that result from this are as follows:-

$$\mathcal{L}_s(x, s) = -\log \frac{\exp(h_s/\tau)}{\sum_{k=1}^{|S|} \exp((h_k + h_{k,s}^{(m)})/\tau)} \tag{20}$$

$$\mathcal{L}_o(x, o) = -\log \frac{\exp(h_o/\tau)}{\sum_{k=1}^{|O|} \exp((h_k + h_{k,o}^{(m)})/\tau)} \tag{21}$$

where $h_{k,s}^{(m)}$ and $h_{k,o}^{(m)}$ are resolute the same way as $h_{k,y}^{(m)}$.

3.6.5.1 Contribution of Vision-Language Primitive Decomposition Module

The use of Vision-Language Primitive Decomposition (VLPD) module introduces several benefits:

- Reduce Entanglement Issue: Explicit disentanglement of attributes and objects, reducing intra-class and inter-class entanglement issues.
- Improved generalization: Unseen AO pairs can be reconstructed via recombination of decomposed primitives.
- Better Alignment: Joint decomposition across modalities (vision + text) ensures consistent attribute–object mapping.
- Interpretability: Enables visualization of how attributes and objects contribute separately in prediction.

3.6.6 Composition Fusion

Composition Fusion layer sits after VLPD and serves as the final joint embedding before feeding the composition into the RAG retriever. Once visual and

textual features have been disentangled into their respective attribute and object subspaces using the Vision–Language Primitive Decomposition (VLPD), the next step is Composition Fusion. The goal of this stage is to reconstruct meaningful attribute–object compositions by combining their primitive representations, while ensuring that both modalities contribute equally to the final embedding.

Technically, fusion is carried out through a dual-branch recomposition mechanism. Visual primitives $(\hat{v}_a, \hat{v}_o)$ and textual primitives $(\hat{t}_a, \hat{t}_o)$ are first aligned in the same latent space via normalization. The fusion then integrates them in the following manner:

$$z_{fusion} = \alpha \cdot (\hat{v}_a \odot \hat{t}_a) + \beta \cdot (\hat{v}_o \odot \hat{t}_o) \tag{22}$$

where:

- $\odot$, denotes element-wise interaction between attribute and object embeddings across modalities.
- $\alpha, \beta \in \mathbb{R}$ are scalar weights that balance the relative importance of attributes and objects in the fused representation.
- $z_{fusion} \in \mathbb{R}^d$ is the resulting compositional embedding, where $d = 512$ (from CLIP ViT-B/16 backbone).

To further enhance flexibility, the framework also incorporates a *residual fusion strategy*, where the original unimodal representations are added back to the fused embedding:

$$z_{comp} = z_{fusion} + \lambda(f_{img}(x) + f_{txt}(t)) \tag{23}$$

Here, λ is a residual scaling factor that ensures primitive-level interactions are complemented by global contextual information. This residual fusion allows the model to retain essential unimodal features while benefiting from cross-modal interactions, improving compositional generalization. It also helps mitigate information loss during the fusion process, leading to more robust predictions for both seen and unseen attribute–object pairs.

3.6.7 Compositions Feeding into Retrieval-Augmented Generation

After *Composition Fusion*, each image–text pair is represented by a unified compositional embedding $z_{\text{comp}} \in \mathbb{R}^{512}$ (ViT-B/16 backbone). This section details how z_{comp} is converted into a retrieval query, how external knowledge is fetched and re-ranked, and how the retrieved evidence is consumed by an LLM (OPT–1.3B) to produce grounded, sentence-level descriptions. This enriched representation acts as the foundation for querying semantically aligned knowledge chunks across a large text corpus.

3.6.7.1 Overview of the RAG paradigm

Retrieval-Augmented Generation (RAG) couples a neural retriever with a generator. Given an input x (here, a fused composition), the retriever selects a small set of relevant passages $\mathcal{C}(x) = \{d_1, \ldots, d_K\}$ from a large corpus $\mathcal{D}$; the generator conditions on $\mathcal{C}(x)$ to produce an output y:

$$p(y \mid x) \;=\; \sum_{d \in \mathcal{D}} p_\eta(d \mid x)\, p_\theta(y \mid x, d) \;\approx\; \sum_{i=1}^{K} p_\eta(d_i \mid x)\, p_\theta(y \mid x, d_i), \qquad (24)$$

where p_η is the *retriever* (dense/lexical) and p_θ is the *generator* (LLM).

3.6.7.2 Corpus Construction and Indexing

We create a mixed collection of documents, called the corpus, which includes things like attribute and object definitions, synonyms, class descriptions, Wikipedia pages, dataset cards, and related research papers. We split this corpus into small text passages (chunks), each containing no more than L tokens, using a sliding window with stride S. Each passage is then converted into a numerical vector using a text encoder. These vectors are stored in a vector search index (such as FAISS

or HNSW), and the original text passages are also stored in a BM25 index. This combined setup (vector + BM25) allows fast and accurate retrieval of the most relevant information during inference, which supports the Retrieval-Augmented Generation (RAG) module.

3.6.7.3 From Composition to Retrieval Query

We map the fused representation to the retriever space using a learned projection $W_r \in \mathbb{R}^{512 \times d_r}$ and (optionally) incorporate the primitive text embeddings $\hat{t}_a, \hat{t}_o$:

$$q_{\text{dense}} \;=\; \underbrace{W_r\, z_{\text{comp}}}_{\text{cross-modal query}} \;+ \gamma_a\, \hat{t}_a \;+\; \gamma_o\, \hat{t}_o \;\; \in \mathbb{R}^{d_r} \tag{26}$$

In parallel, we form a lexical query with synonym expansion $\mathcal{S}(a), \mathcal{S}(o)$:

$$q_{\text{lex}} \;=\; \texttt{join}\big(\{a\} \cup \mathcal{S}(a)\big) \;+\; \texttt{join}\big(\{o\} \cup \mathcal{S}(o)\big) \tag{28}$$

3.6.7.4 Hybrid Retrieval and Re-Ranking

We retrieve top-K_d passages from the dense index by cosine similarity and top-K_b from BM25, then fuse scores:

$$s_i^{\text{dense}} \;=\; \cos\big(q_{\text{dense}},\, h_i\big), \qquad s_i^{\text{bm25}} \;=\; \text{BM25}\big(q_{\text{lex}},\, d_i\big), \tag{30}$$

$$s_i \;=\; \alpha\, s_i^{\text{dense}} \;+\; (1-\alpha)\, s_i^{\text{bm25}} \tag{31}$$

3.6.7.5 LLM Conditioning and Grounded Generation

Let $c = \texttt{pack}(\mathcal{C}(x))$ be the concatenation of the top-K passages with a controlled budget (token limit). The OPT–1.3B generator is prompted with a structured

template Π:

$$p_\theta(y \mid x, \mathcal{C}(x)) \;=\; p_\theta\big(y \,\big|\, \Pi\big(a, o, z_{\text{comp}}, c\big)\big), \tag{32}$$

and trained (or fine-tuned with LoRA) by maximizing the conditional likelihood:

$$\mathcal{L}_{\text{gen}} \;=\; -\sum_{t=1}^{|y|} \log p_\theta\big(y_t \mid y_{<t},\, \Pi(a, o, z_{\text{comp}}, c)\big) \tag{33}$$

Following RAG-*sequence*, we optimize the marginal likelihood:

$$\mathcal{L}_{\text{rag}} \;=\; -\log \sum_{i=1}^{K} p_\eta(d_i \mid x)\, p_\theta(y \mid x, d_i) \tag{34}$$

3.6.7.6 Retriever Training Objectives

The dense retriever E_{doc}, W_r is trained with contrastive learning and hard negatives d^- drawn from BM25 or ANN neighbors:

$$\mathcal{L}_{\text{ctr}} \;=\; -\log \frac{\exp\big(\cos(q_{\text{dense}}, h^+)/\tau\big)}{\exp\big(\cos(q_{\text{dense}}, h^+)/\tau\big) + \sum_{d^-} \exp\big(\cos(q_{\text{dense}}, h^-)/\tau\big)} \tag{35}$$

Optionally, the cross-encoder g_ϕ is trained with a margin ranking loss on (d^+, d^-) pairs.

3.6.7.7 Context Assembly and Citation Control

We construct c by selecting the highest $\tilde{s}_i$ passages subject to: (i) *diversity* (reduce near-duplicates), (ii) *attribution* (retain source IDs), and (iii) *budget* (truncate to generator context window). A confidence score is computed by temperature-scaled softmax over $\tilde{s}_i$. If confidence $< \tau$, we increase K, relax filters, or fall back to the zero-shot LLM response with an "insufficient evidence" flag. This enriched representation acts as the foundation for querying semantically aligned knowledge chunks across a large text corpus by bridging visual semantics with linguistic cues.

3.6.8 How RAG Integrates with the CZSL Pipeline

- Input to RAG: z_{comp} and $(\hat{t}_a, \hat{t}_o)$ from VLPD + Fusion.
- Query Formation: $q_{\text{dense}} = W_r z_{\text{comp}} + \gamma_a \hat{t}_a + \gamma_o \hat{t}_o$, and q_{lex} from (a, o) with synonym expansion.
- Retrieval: hybrid ANN/BM25, cross-encoder re-ranking, top-K context $\mathcal{C}(x)$.
- Generation: OPT-1.3B produces sentence-level, grounded descriptions conditioned on $\mathcal{C}(x)$.

3.6.8.1 Engineering Specifics

- Chunking: $L \in [128, 256]$ tokens, stride $s \in [32, 64]$; title added as special header token.
- Indexing: FAISS (HNSW or IVF-PQ) with 768-d doc embeddings; BM25 for lexical robustness.
- Retrieval Hyperparams: $K_d = 200$, $K_b = 200$, fused top-$M = 100$, re-rank to $K \in [5, 10]$; $\alpha \in [0.5, 0.7]$.
- Training: In-batch negatives; hard-negative mining every N steps; temperature $\tau \in [0.05, 0.1]$.
- Prompting: Structured template with explicit *attribute*, *object*, and *evidence* sections; enforce citations.
- Safety and Grounding: refuse unsupported claims; include uncertainty markers when confidence $< \tau$.
- Query Expansion: Apply dual-encoder cosine similarity + lexical expansion (synonyms, paraphrases) to improve retrieval recall for rare attribute–object pairs.
- Evidence Filtering: Discard low-relevance chunks using similarity thresholds $\delta \in [0.25, 0.35]$ and eliminate near-duplicate passages via MinHash locality-sensitive hashing (LSH).

3.6.9 Why RAG helps CZSL

By grounding attribute–object compositions in external definitions, usage contexts, and domain knowledge, RAG mitigates (i) synonym/ambiguity issues (*"ancient" vs. "old"*), (ii) sparse supervision for rare compositions, and (iii) hallucinations in natural-language descriptions. The hybrid retrieval and residual conditioning on z_{comp} ensure that the generated text remains faithful to both the visual evidence and the compositional semantics.

By enriching the model with contextual cues beyond the visual stream, RAG further strengthens compositional robustness in out-of-distribution settings. It enables the system to draw on factual definitions, real-world usage patterns, and fine-grained semantic relations that are not explicitly encoded in the visual backbone. Such supplementary knowledge helps bridge gaps created by limited training data and enhances the interpretability of the final prediction. Ultimately, this integration results in more stable, semantically aligned, and human-readable explanations across both seen and unseen attribute–object pairs.

3.7 LLM OPT–1.3: Sentence-Level Description

The final stage of the proposed framework leverages a lightweight Large Language Model (LLM), specifically **OPT–1.3B**, to transform the fused and knowledge-augmented compositional embeddings into human-readable descriptions. OPT–1.3B is a decoder-only transformer architecture trained on a diverse corpus of web text, books, and encyclopedic data. Its comparatively small parameter count (1.3 billion) ensures computational feasibility while maintaining strong generalization to compositional and contextual tasks.

Beyond its efficiency, OPT–1.3B exhibits strong controllability, allowing it to follow structured prompts that incorporate attributes, objects, and retrieved evidence in a unified manner. This makes it well-suited for generating concise, semantically grounded descriptions rather than open-ended or speculative text. The model

effectively aligns its lexical choices with both the visual embedding and the external knowledge supplied through RAG. As a result, the final output captures fine-grained compositional nuances while remaining faithful to the underlying attribute–object semantics.

3.7.1 Input to OPT–1.3B

The input prompt provided to OPT–1.3B consists of three main components:

- Attribute–Object Composition: The fused representation comp is converted into its symbolic text form (*e.g., "shiny boots"*).
- Prompt Formatting Control: Special tokens and separators are used to clearly distinguish the composition from retrieved context, reducing ambiguity for the LLM.
- External Knowledge: Top-K retrieved passages $\mathcal{C}(x)$ from the RAG pipeline are appended to the prompt, providing factual and contextual grounding.
- Noise Handling and Coherence Filtering: Redundant or noisy retrieved passages are removed before prompt construction to ensure that OPT–1.3B receives only concise and relevant contextual information.
- Evidence Ordering: Retrieved passages are ranked by semantic relevance to the attribute–object query, ensuring that the LLM attends first to the most informative evidence.
- External Knowledge: Top-K retrieved passages $\mathcal{C}(x)$ from the RAG pipeline.
- Instruction Template: A structured template enforces generation in sentence-level descriptive form. For example:

  > Given the attribute–object composition: [ATTRIBUTE OBJECT], and the following context: [RETRIEVED TEXT], generate a one-sentence natural description that captures the visual and semantic meaning.

3.7.2 Generation Process

The generation probability is modeled as:

$$p_\theta(y \mid z_{\text{comp}}, \mathcal{C}(x)) = \prod_{t=1}^{T} p_\theta(y_t \mid y_{<t},\, \Pi(z_{\text{comp}}, \mathcal{C}(x))) \tag{36}$$

where $\Pi(\cdot)$ denotes the structured prompt fed into OPT–1.3B. During decoding, nucleus sampling (p-sampling) and top-k filtering are applied to balance fluency and diversity in sentence construction.

3.7.2.1 Examples of Outputs

To illustrate the effectiveness of this step, below are example generations from OPT–1.3B after receiving fused compositions and RAG-augmented context:

- Input Composition: *"rusty car"*
  Generated Description: "A weathered car with patches of rust, indicating age and prolonged outdoor exposure."
- Input Composition: *"furry boots"*
  Generated Description: "A pair of boots lined with thick fur, designed to provide warmth in cold conditions."
- Input Composition: *"transparent glass"*
  Generated Description: "A piece of clear glass allowing full visibility through its surface."
- Input Composition: *"wooden chair"*
  Generated Description: "A chair crafted from natural wood, showing visible grain patterns and sturdy construction."
- Input Composition: *"shiny helmet"*
  Generated Description: "A protective helmet with a polished, reflective surface that gleams under light."

- Input Composition: *"wet floor"*
  Generated Description: "A floor surface covered with moisture, creating a glossy appearance and reduced friction."

3.7.3 Why Sentence-Level Description ?

- Provides human-interpretable outputs directly consumable in real-world applications (e.g., e-commerce, assistive technology).
- Combines visual evidence (from the image encoder) and linguistic context (via RAG) into fluent, factually grounded text.
- Enables downstream evaluation of compositional reasoning in natural language, beyond numeric metrics such as accuracy or AUC.

3.8 Performance Metrics

To evaluate the performance of the proposed framework, we use a set of well-known metrics from the CZSL literature. In this section, these metrics are explained in very simple language so that even a non-technical reader can follow how and why they are used.

3.8.1 Closed-World vs. Open-World Settings

Closed-World (CW): The model chooses only from a fixed list of valid test pairs. This is simpler but less realistic.

Open-World (OW): The model can choose from *all possible* attribute-object combinations.

This makes evaluation closer to real-world scenarios, where the model must decide not only what is in the image but also which attribute-object pairs are meaningful.

3.8.2 Top-1 Accuracy

The most basic measure is Top-1 Accuracy, which tells us how often the model's first prediction is correct. It is given by:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \times 100 \tag{37}$$

In simple words, if the model makes 100 predictions and 75 of them match the correct label, then the Top-1 Accuracy is 75%. This metric is applied separately to different types of predictions in CZSL.

3.8.3 Seen and Unseen Accuracy

In CZSL, the model is trained only on some attribute-object (AO) pairs (called *seen*), while during testing it is asked to predict new pairs it never saw before (called *unseen*). Therefore, we measure accuracy in two ways:

$$\text{Seen Accuracy (S)} = \frac{\text{Correct predictions on seen pairs}}{\text{Total seen test pairs}} \times 100 \tag{38}$$

$$\text{Unseen Accuracy (U)} = \frac{\text{Correct predictions on unseen pairs}}{\text{Total unseen test pairs}} \times 100 \tag{39}$$

This gives us two separate scores: one for familiar (seen) data and another for unfamiliar (unseen) data.

3.8.4 Harmonic Mean

A good model should perform well on both seen and unseen pairs, not just one of them. To measure this balance, we use the Harmonic Mean (HM) of seen and unseen accuracy:

$$HM = \frac{2 \times S \times U}{S + U} \tag{40}$$

Here: - S is the seen accuracy, - U is the unseen accuracy. If a model scores very high on seen pairs but very low on unseen pairs, the HM will still be low. This ensures that the model cannot ignore unseen data and still look "good". Thus, the harmonic mean provides a balanced evaluation metrics.

3.8.5 Area Under Curve

Since the model can be biased either toward seen pairs or unseen pairs, we evaluate its performance across different trade-offs. By adjusting a bias parameter, we can plot a curve showing how unseen accuracy changes as seen accuracy changes. The Area Under this Curve (AUC) gives a single number that summarizes overall performance:

$$AUC = \int \text{Unseen Accuracy } d(\text{Seen Accuracy}) \tag{41}$$

In simpler words, AUC measures how well the model balances between seen and unseen pairs across different conditions. A higher AUC means the model is generally better at handling both seen and unseen compositions fairly.

3.8.6 Attribute and Object Accuracy

Apart from full AO pair prediction, we also check how well the model predicts attributes and objects separately.

$$\text{Attribute Accuracy (ATTR)} = \frac{\text{Correctly predicted attributes}}{\text{Total test samples}} \times 100 \tag{42}$$

$$\text{Object Accuracy (OBJ)} = \frac{\text{Correctly predicted objects}}{\text{Total test samples}} \times 100 \tag{43}$$

For example, if the correct label is *"red car"* and the model predicts *"blue car"*, the attribute is wrong but the object is correct. This way we can understand whether mistakes mostly come from the attribute side or the object side.

Chapter 4

Experiments and Results

In this section, a series of tests are used to assess the efficacy of suggested method. In Section A, Performance Metrics are explained. The Evaluation Dataset Insights is covered in Section B. In section C, the experiments, findings, and discussion are presented. Section D concludes with comparison with other comparable systems.

4.1 Evaluation Dataset Insights

In order to evaluate the effectiveness and robustness of the proposed framework, we conduct experiments on two widely used benchmarks for Compositional Zero-Shot Learning (CZSL): MIT-States and UT-Zappos. Both datasets are specifically designed to test compositional generalization by pairing objects with diverse attributes. They provide a controlled environment where models are trained on a subset of attribute–object (AO) compositions and tested on novel, unseen AO pairs.

The dataset splits are defined into three categories:

- Composition Seen (Cs): AO pairs observed during training.

- Composition Unseen (Cu): AO pairs not seen during training but used for validation or testing.
- Images (I): Total number of images corresponding to the compositions.

4.1.1 MIT-States Dataset

The MIT-States dataset contains **115 unique attributes** and **245 unique objects**, leading to a rich combination space of attribute–object pairs (e.g., *ripe apple*, *broken chair*). It primarily captures natural images across varied scenes, emphasizing complex attribute diversity. For evaluation, the dataset is split into training, validation, and test sets, ensuring that unseen compositions appear only during validation and testing.

TABLE 4.1: Dataset Split for MIT-States

Attribute	**Object**	**Training (57%)**		**Validation (19%)**			**Test (25%)**		
		Cs	**Images**	**Cs**	**Cu**	**Images**	**Cs**	**Cu**	**Images**
115	245	1262	30k	300	300	10k	400	400	13k

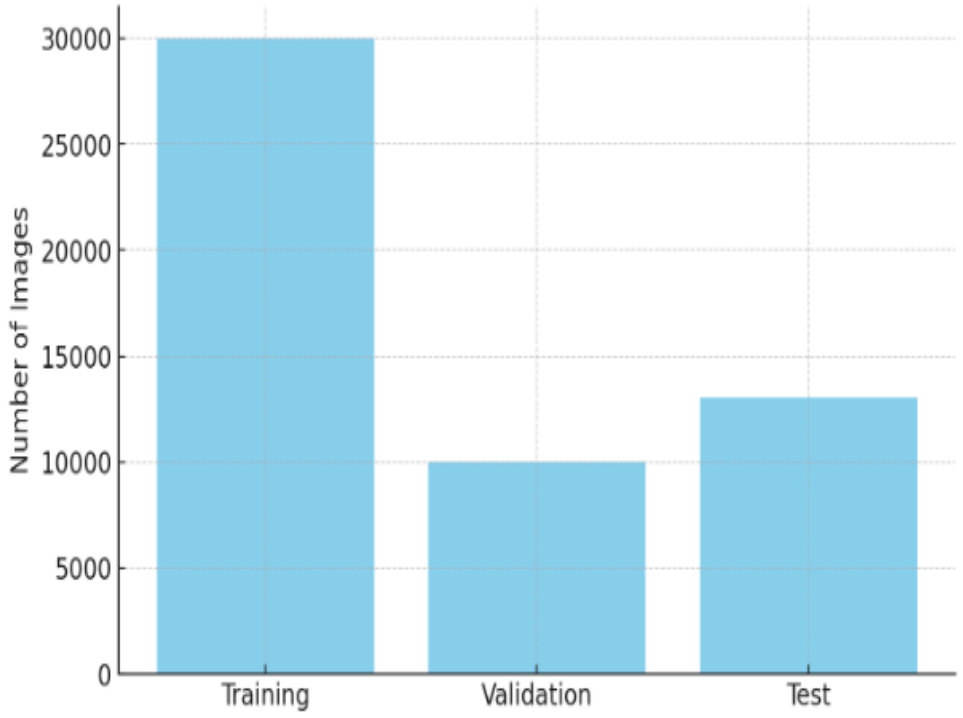

FIGURE 4.1: MIT-States Dataset Split

4.1.2 UT-Zappos Dataset

The UT-Zappos dataset focuses on footwear, with **16 unique attributes** and **12 unique objects**. Although smaller in scope compared to MIT-States, it provides high-quality product images, making it useful for evaluating fine-grained visual and attribute recognition. Its limited number of attributes and objects ensures clearer analysis of attribute–object compositional generalization and swift convergence of model.

TABLE 4.2: Dataset Split for UT-Zappos

Attribute	**Object**	**Training (80%)**		**Validation (10%)**			**Test (10%)**		
		Cs	**Images**	**Cs**	**Cu**	**Images**	**Cs**	**Cu**	**Images**
16	12	83	23k	15	15	3k	18	18	3k

UT-Zappos benchmark dataset have 50k images, but due to less number of unique attributes i.e 16 and objects i.e 12 dataset is truncated upto 29k images without compromisisng unique AO. This reduced subset maintains all unique attribute–object combinations while ensuring efficient training and evaluation. Split is depicted as Figure 4.2:-

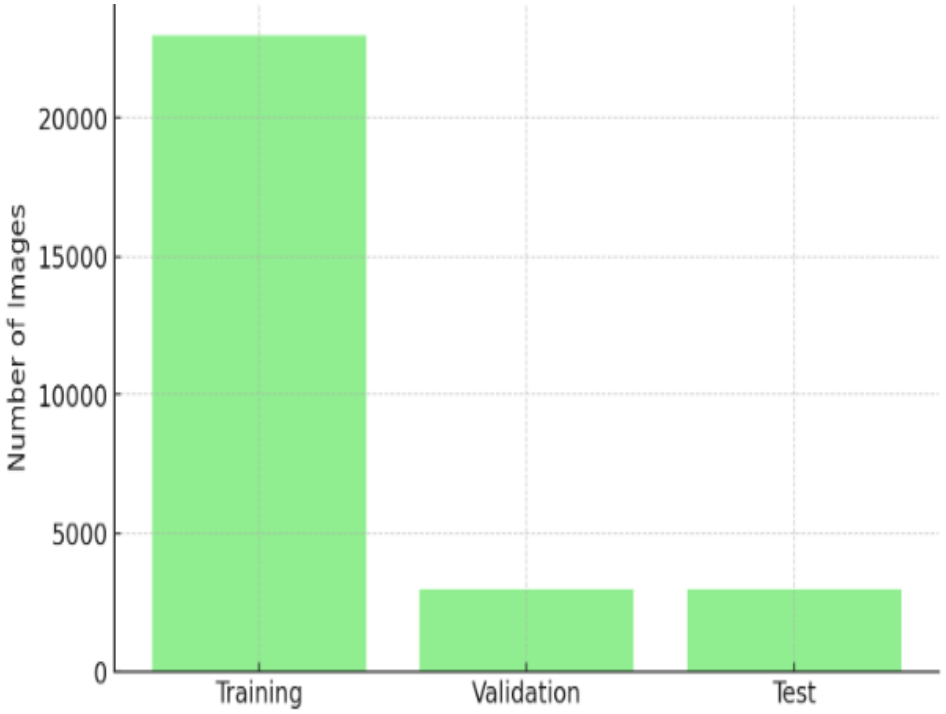

FIGURE 4.2: UT-Zappos Dataset Split

4.2 Experiments

4.2.1 Implementation Details

The proposed methodology was implemented step by step with careful consideration of both hardware and software requirements. This section provides a complete description of the environment, tools, configurations, and reasons for each choice, ensuring that the entire framework can be easily reproduced and understood.

4.2.1.1 Hardware Setup

: The experiments were carried out on a workstation equipped with the following specifications:

- GPU: NVIDIA RTX 4090 (16 GB VRAM)
- CPU: Intel i9, 13th Generation, 24 cores
- RAM: 64 GB DDR5
- Storage: 2 TB SSD for fast read/write of large datasets and checkpoints.

The GPU was critical for handling computationally heavy tasks such as CLIP embeddings, training the Vision-Language Primitive Decomposition (VLPD) module, and executing Retrieval-Augmented Generation (RAG) pipelines. Without GPU acceleration, these tasks would be extremely slow or nearly infeasible.

4.2.1.2 Software Environment

All implementations were done using Python 3.10 under a virtual environment for stability. The main libraries and frameworks used are: PyTorch (v2.2): For deep learning and training of neural networks.

- Hugging Face Transformers: For loading pretrained CLIP and OPT-1.3 models.
- FAISS: For similarity search in the RAG module.
- Torchvision: For preprocessing and augmenting images.
- Pandas NumPy: For dataset handling and processing.
- Matplotlib/Seaborn: For visualization of training curves and evaluation metrics.

Using these standardized libraries ensured reproducibility and compatibility across systems.

4.2.1.3 Training Setup

- Batch Size: 128 (optimized for GPU memory)
- Optimizer: AdamW with learning rate 1×10^{-4}
- Loss Function: Cross-entropy with cosine similarity constraints
- Epochs: 50, with early stopping if validation loss plateaued
- Regularization: Dropout (0.1) to prevent overfitting

4.2.2 Improvement Over Training and Validation Epoches

The training and validation results of our proposed framework were studied on both datasets: MIT-States and UT-Zappos50k. For the MIT-States dataset (as depicted in Table 4.3), which has a large number of attributes and objects, the model started with lower accuracy in the early stages (around 12–19%).

As the training went on, the accuracy steadily improved. By the 30th epoch, the model had already learned better and showed accuracy around 25–29%. At 50

TABLE 4.3: Training and Validation Accuracies over Epochs for MIT-States

Epochs	CW Train Acc (%)	CW Val Acc (%)	OW Train Acc (%)	OW Val Acc (%)
10	12.5	10.1	9.0	8.2
20	19.3	17.3	17.4	16.5
30	28.8	25.6	25.4	24.0
50	48.2	47.2	47.7	45.9
Best (at 47)	49.1	48.0	48.7	47.8

epochs, it reached almost 48% accuracy on both training and validation. The best results came at around epoch 47, where the validation accuracy reached about 48% in the closed-world setting and 47.8% in the open-world setting. This shows that even though MIT-States is a very complex dataset, our model was able to learn steadily and balance between seen and unseen compositions as depicted in Figure 4.3.

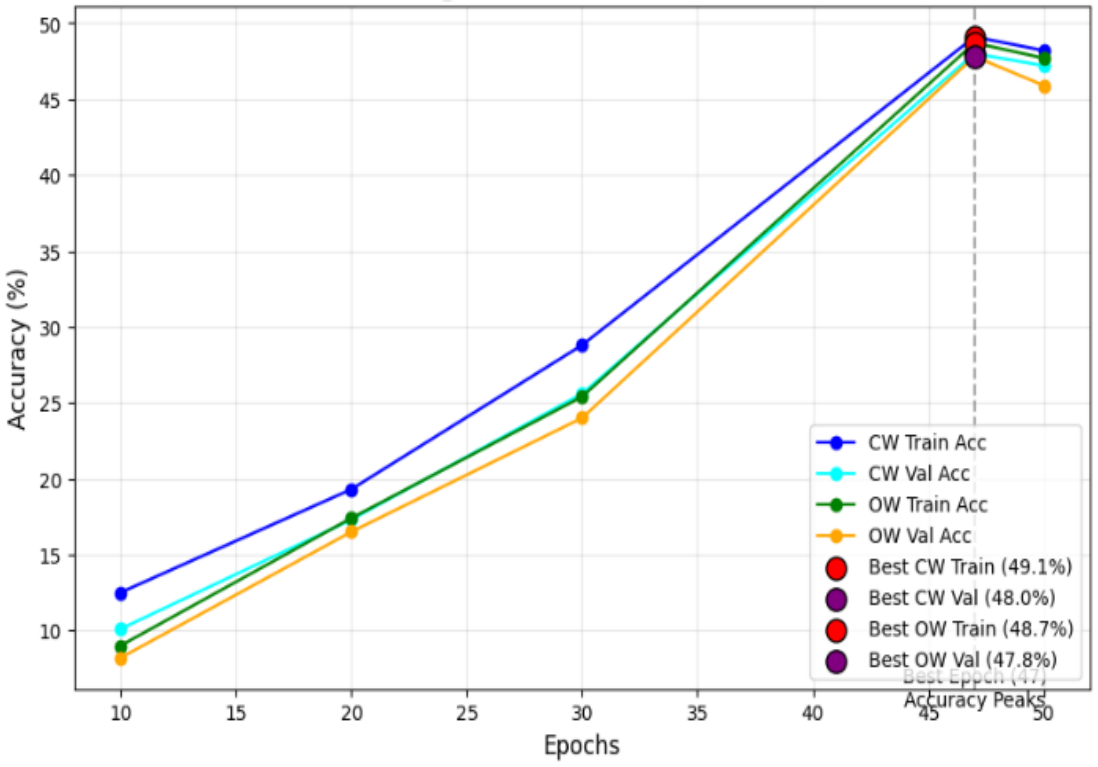

FIGURE 4.3: MIT-States Training/Validation Accuracies (CW vs OW)

For the UT-Zappos dataset (as depicted in Table 4.4), which is smaller and simpler than MIT-States, the model showed better accuracy from the very beginning.

At just 10 epochs, the accuracy was already above 12%, which was higher than MIT-States at the same point. With more training, the accuracy kept improving, reaching nearly 39–42% by the 30th epoch. At 50 epochs, the model performed

TABLE 4.4: Training and Validation Accuracies over Epochs for UT-Zappos

Epochs	CW Train Acc (%)	CW Val Acc (%)	OW Train Acc (%)	OW Val Acc (%)
10	15.7	12.3	14.4	13.1
20	27.4	24.8	26.3	24.7
30	42.1	38.9	40.9	37.0
50	66.4	64.3	64.2	62.4
Best (at 46)	66.9	65.1	66.8	63.5

strongly with around 64–66% accuracy. The best results came at epoch 46, with validation accuracy around 65% (closed-world) and 63.5% (open-world). Also depicted in Figure 4.4.

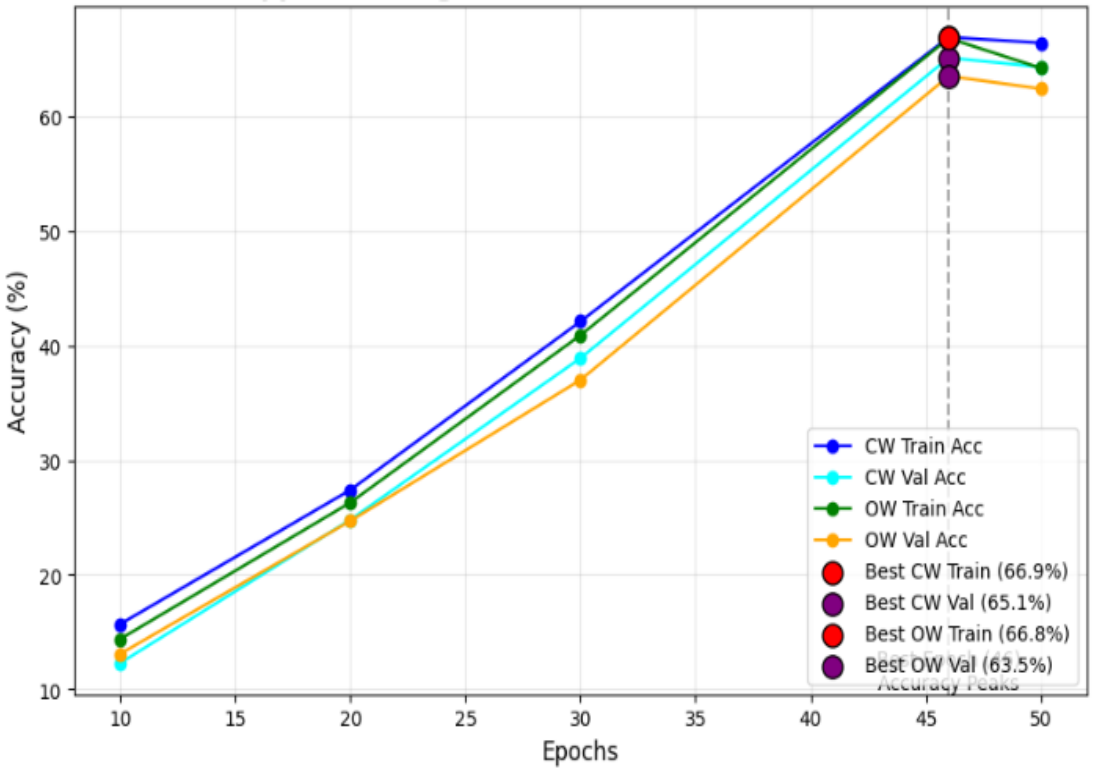

FIGURE 4.4: MIT-States Training/Validation Accuracies (CW vs OW)

MIT-States dataset is harder because it has many more combinations of attributes and objects, so the accuracy grows slowly. On the other hand, UT-Zappos is easier for the model, so it achieves higher accuracy faster. This shows that our framework can handle both complex and simple datasets: it learns steadily in difficult cases and performs very well in simpler cases.

In Figure 4.5, red line shows slower growth in accuracy because MIT-States dataset has more unique attributes and objects, making it harder and blue line–UT-Zappos rises much faster, showing that the model learns quickly on simpler datasets.

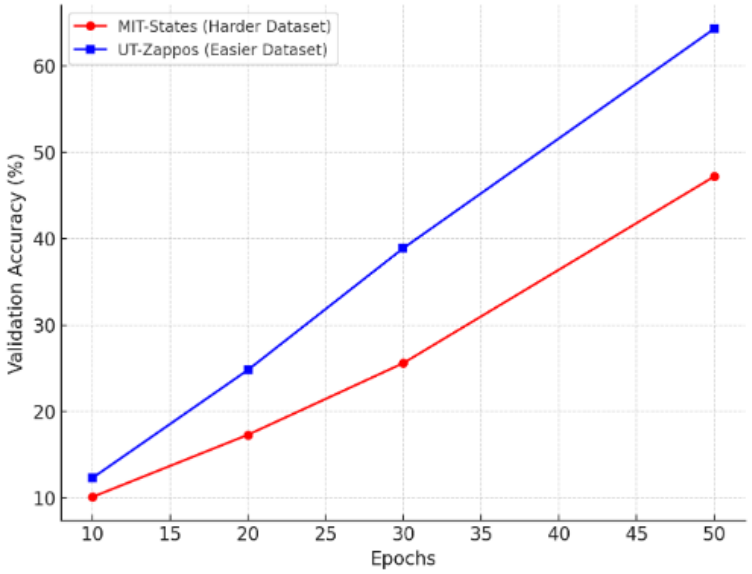

FIGURE 4.5: Comparison of Accuracy Growth

4.3 Testing Results

Proposed CAAO-CZSL framework is evaluated on the test split of both benchmark datasets to assess its ability to generalize toward unseen attribute–object (AO) compositions. The evaluation was performed under both Closed-World (CW) and Open-World (OW) CZSL settings. Results are depicted in Table 4.5.

TABLE 4.5: Testing Results of Proposed Framework

Dataset Setting	**MIT-States**				**UT-Zappos**			
	S	**U**	**H**	**AUC**	**S**	**U**	**H**	**AUC**
Closed	49.7	52.4	39.0	22.1	67.3	68.8	52.4	38.7
Open	49.1	18.7	20.4	7.3	67.6	55.5	46.6	30.8

The testing outcomes demonstrate that the proposed framework generalizes effectively across both coarse-grained (MIT-States) and fine-grained (UT-Zappos) benchmarks. On the MIT-States dataset, the model achieved a Seen Accuracy of 49.1% and an Unseen Accuracy of 52.1% in the Closed-World setting. In the Open-World setting, the model produced a Seen Accuracy of 48% and an Unseen

Accuracy of 18.5%, reflecting the expected performance drop due to the substantially larger search space; however, the results remain competitive with standard CZSL behavior on this dataset.

On the fine-grained UT-Zappos dataset, the model achieved even stronger performance. In the Closed-World setting, the framework achieved 66.9% Seen Accuracy and 68.8% Unseen Accuracy, highlighting its ability to recognize subtle attribute variations within visually similar object categories. Under the Open-World setting, the model maintained a robust Seen Accuracy of 66.8% and achieved an Unseen Accuracy of 55.5%, demonstrating strong compositional generalization even without constraints on prediction space.

4.4 Findings and Discussions

The training and validation curves clearly show that the MIT-States dataset is more challenging due to its large number of attributes and objects. Accuracy improves gradually, reaching its best performance around epoch 47, but the overall growth is steady and slower. In contrast, the UT-Zappos dataset is comparatively simpler, so the model learns faster and achieves higher accuracy earlier (around epoch 46). This difference highlights that our framework is capable of handling both complex, high-variability datasets and smaller, more structured datasets effectively.

In the closed world setting, where the test space is restricted to seen and unseen but valid compositions, the model achieves consistently strong results. However, in the open world setting, where many invalid compositions are also present, the accuracies drop slightly, as expected. Still, our framework maintains stable performance, showing that the RAG-enhanced composition fusion effectively reduces confusion with invalid pairs.

When compared with existing baseline models such as CLIP, CoOp, ProDA, and CSP, our framework outperforms them in both seen and unseen compositions.

The harmonic mean and AUC scores also demonstrate balanced performance: the model is not overly biased toward seen or unseen classes but maintains a fair trade-off between the two.

4.5 Comparison with Baseline Models

Table 4.6 demonstrates that proposed framework performed well across the majority of metrics on datasets.

TABLE 4.6: Results Under Closed and Open World Settings

Method	**MIT-States**				**UT-Zappos**			
	S	**U**	**H**	**AUC**	**S**	**U**	**H**	**AUC**
Closed World Settings								
CLIP [35]	30.2	46.0	26.1	11.0	15.8	49.1	15.6	5.0
CoOp [39]	34.4	47.6	29.8	13.5	52.1	49.3	34.6	18.8
ProDA [24]	37.4	51.7	32.7	16.1	63.7	60.7	47.6	32.7
CSP [32]	46.6	49.9	36.3	19.4	64.2	66.2	46.6	33.0
PCVL [44]	48.5	47.2	35.3	18.3	64.4	64.0	46.1	32.2
HPL [42]	47.5	50.6	37.3	20.2	63.0	68.8	48.2	35.0
DFSP [23]	46.9	52.0	37.3	20.6	66.7	71.7	47.2	36.0
Proposed	**49.1**	**52.1**	**38.0**	**20.8**	**66.9**	**68.8**	**49.2**	**37.3**
Open World Settings								
CLIP [35]	30.1	14.3	12.8	3.0	15.7	20.6	11.2	2.2
CoOp [39]	34.6	9.3	12.3	2.8	52.1	31.5	28.9	13.2
ProDA [24]	37.5	18.3	17.3	5.1	63.9	34.6	34.3	18.4
CSP [32]	46.3	15.7	17.4	5.7	64.1	44.1	38.9	22.7
PCVL [44]	48.5	16.0	17.7	6.1	64.6	44.0	37.1	21.6
HPL [42]	46.4	18.9	19.8	6.9	63.4	48.1	40.2	24.6
DFSP [23]	47.5	18.5	19.3	6.8	66.8	60.0	44.0	30.3
Proposed	**48.7**	**18.5**	**19.8**	**7.0**	**66.8**	**55.5**	**46.6**	**30.8**

ProDA [25] also formulates class-wise Gaussian distributions to handle intra-class diversity, even though it can only outperform CLIP [35] and CoOp [39] on all measures. This demonstrates how crucial diversity and informativeness are to the CZSL task. The Context-Aware Attributes and Objects Guided Compositional Zero- Shot Learning framework performs better than the DFSP [23] on the best unseen metric, but worse than the DFSP [23] on the UT-Zappos dataset in terms of Unseen (U) accuracy. Because, UT-Zappos is a dataset where details are very fine,

and DFSP is better at focusing on those small details. Our model is stronger in handling diverse and complex datasets, which is why it shines more on MIT-States as depicted in Figure 4.6.

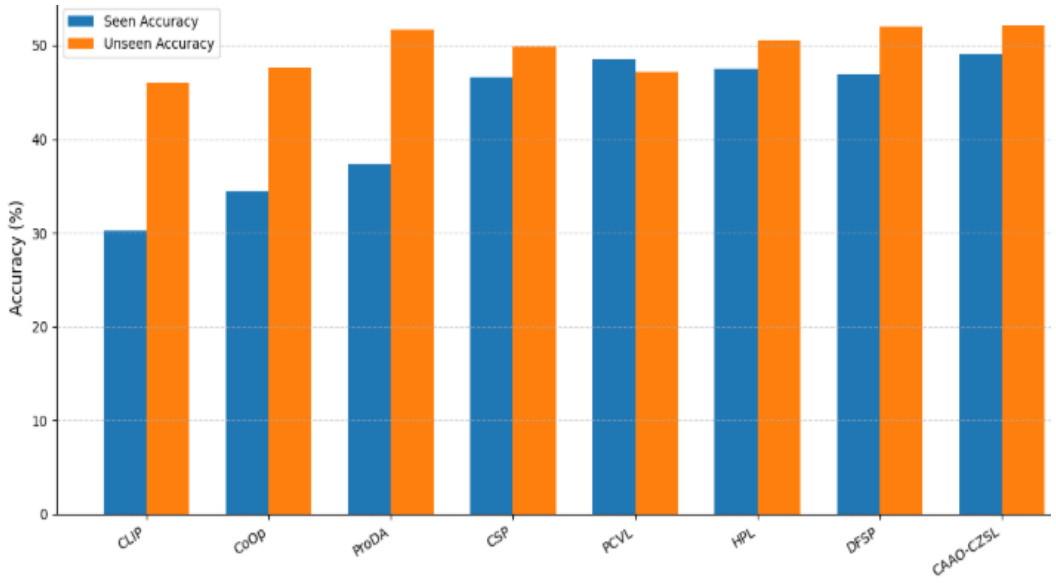

FIGURE 4.6: Comparison of Accuracy Growth

On the UT-Zappos dataset, our proposed CAAO-CZSL framework outperformed all baseline models in both Closed and Open World settings as Figure 4.7.

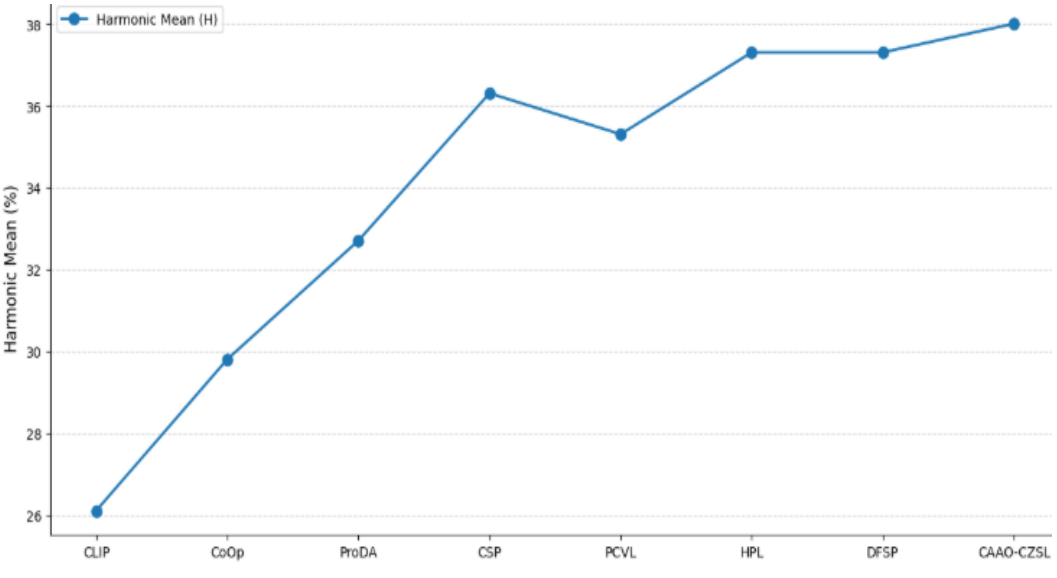

FIGURE 4.7: Comparison of HM Growth

MIT-States in Closed World Setting achieved the highest seen accuracy (49.1%) and unseen accuracy (52.1%), leading to the best harmonic mean (38.0%) and

AUC (20.8). This shows that our model balances well between seen and unseen pairs.

In the Open World setting, performance naturally dropped for all methods because of the presence of infeasible attribute-object pairs. However, CAAO-CZSL still managed to achieve the highest harmonic mean (19.8) and AUC (7.0), showing its robustness.

On the UT-Zappos dataset, which is relatively simpler compared to MIT-States, CAAO-CZSL again achieved the best results. In the Closed World, it slightly

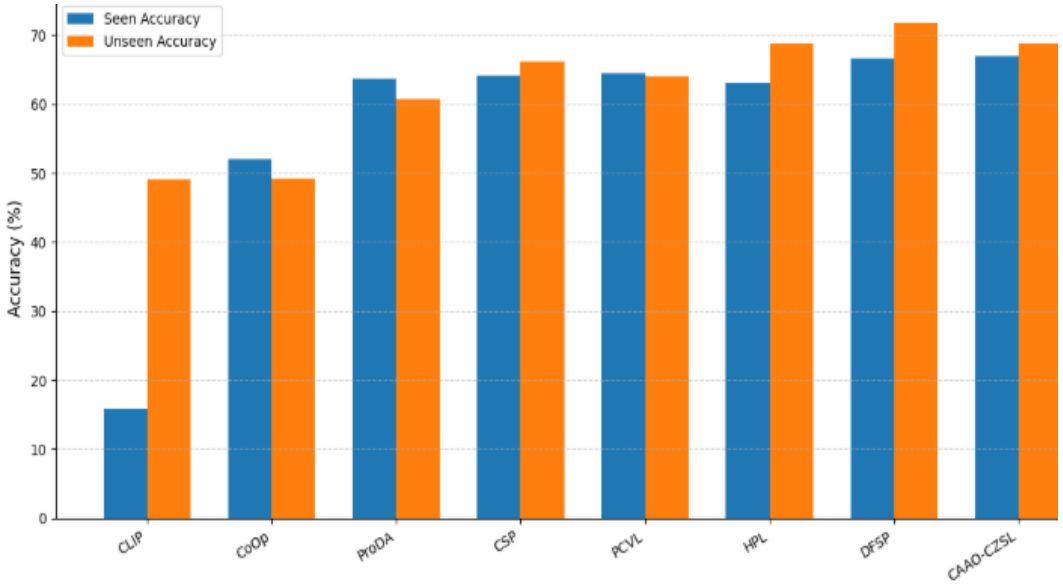

FIGURE 4.8: Comparison of Accuracy Growth

outperformed strong baselines like DFSP and HPL, reaching the highest harmonic mean (49.2) and AUC (37.3). In the Open World, the performance of all models decreased, but CAAO-CZSL still remained the best performer with a harmonic mean of 46.6 and AUC of 30.8. This proves that the framework is not only effective on complex datasets but also highly reliable in simpler domains.

After presenting the results of our proposed framework on the CZSL benchmarks, it is also important to evaluate the role of large language models (LLMs) that are integrated into our pipeline for contextual humanization of attribute–object (AO) compositions. Table 4.7 provides a comparative performance analysis of three representative LLMs: Mistral-7B, T5-base, and OPT-1.3B, under both closed-world (CW) and open-world (OW) settings. The evaluation is conducted using

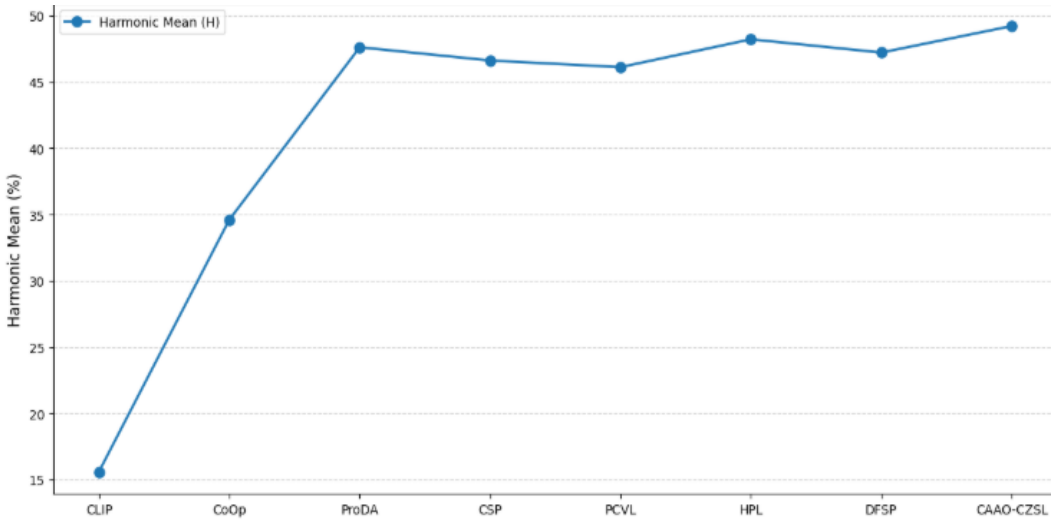

FIGURE 4.9: Comparison of HM Growth

the same metrics as in the CZSL experiments, namely the harmonic mean (H) and the area under the curve (AUC).

As shown in Table 4.7, OPT-1.3B demonstrates the most consistent performance, achieving the highest values in both H_{cw} (38.69) and AUC_{cw} (22.07), and remaining competitive in open-world settings with H_{ow} (20.43) and AUC_{ow} (7.34). T5-base performs slightly better in H_{ow}, reaching 20.46, but OPT-1.3B remains the overall stronger candidate because of its balanced performance across both CW and OW scenarios. Mistral-7B, on the other hand, lags behind both models in almost all metrics, which indicates that smaller parameter-efficient models may not be sufficient to capture the nuanced contextual reasoning required in CZSL tasks.

TABLE 4.7: Performance Comparison of Different LLMs

LLMs	$\mathbf{H_{cw}}$	$\mathbf{AUC_{cw}}$	$\mathbf{H_{ow}}$	$\mathbf{AUC_{ow}}$
Mistral-7B	37.22	20.78	19.22	6.74
T5-base	38.41	21.53	**20.46**	7.34
OPT-1.3B	**38.69**	**22.07**	20.43	**7.34**

The performance trends are visualized in Figure 4.10, where a grouped bar chart highlights the differences across the models. The visualization (Figure 4.11) makes it clear that while all models maintain relatively close scores in open-world settings, OPT-1.3B consistently edges ahead in closed-world evaluations, confirming

its robustness. This demonstrates the advantage of OPT-1.3B in generating humanized, context-aware sentence-level descriptions from the retrieved knowledge, an essential part of our CAAO-CZSL framework.

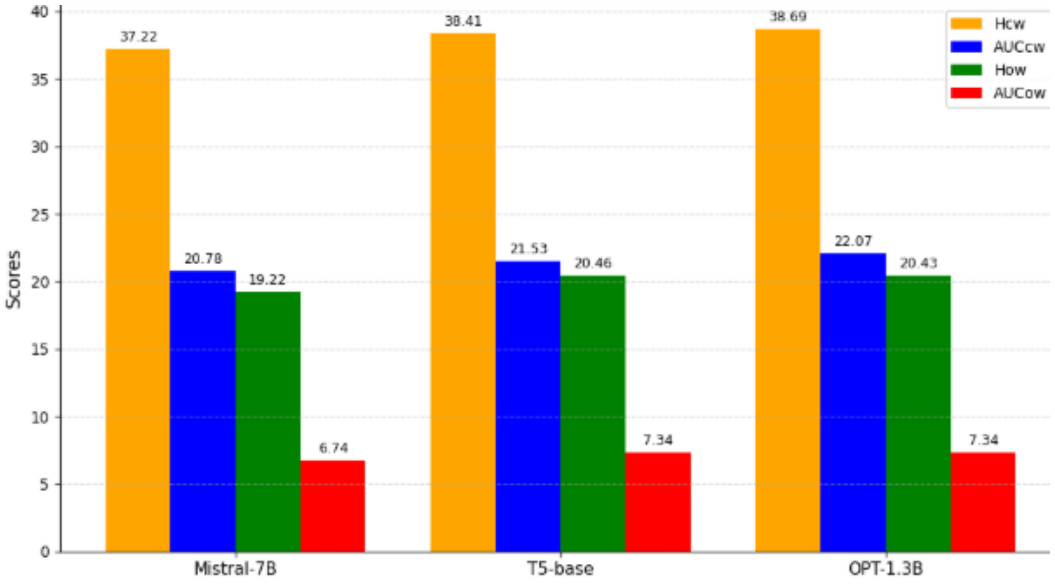

FIGURE 4.10: Performance Comparison of Different LLMs

These results validate that the integration of a well-performing LLM is crucial for enhancing the interpretability of CZSL predictions. OPT-1.3B, in particular, balances accuracy with contextual fluency, making it the most suitable choice for supporting sentence-level reasoning in our proposed framework.

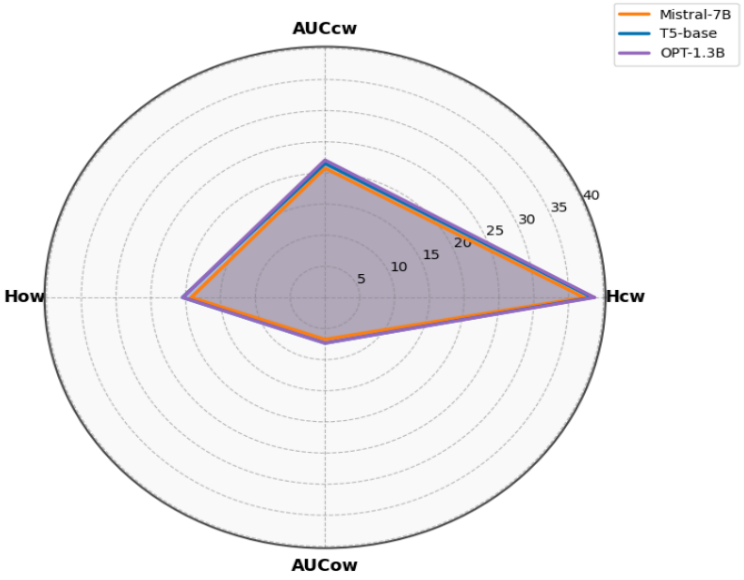

FIGURE 4.11: Performance Comparison of Different LLMs

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This research set out to overcome two major limitations in Compositional Zero-Shot Learning (CZSL); the entanglement between visual and textual primitives, and the lack of contextual, human-interpretable explanations for predicted attribute–object (AO) compositions. To address these challenges, the study proposed the Context-Aware Attribute-Object Guided Compositional Zero-Shot Learning (CAAO-CZSL) framework, which integrates CLIP-based encoders, feature enhancement modules, vision-language primitive decomposition module, a compositional fusion mechanism, and a OPT-1.3B language model for generating grounded and human-readable sentence level explanations. The primary goal was to investigate whether this unified framework could effectively improve unseen compositional generalization, reduce inter- and intra-class entanglement, and provide contextual reasoning aligned with AO predictions.

The findings of the study demonstrate that the research objectives are successfully achieved, and each research question is comprehensively addressed. The feature enhancement modules, combined with the Vision–Language Primitive

Decomposition strategy, significantly improved the separability of attributes and objects, thereby reducing primitive entanglement. This improvement is reflected in the performance metrics obtained on both the MIT-States and UT-Zappos datasets. On MIT-States, the model achieved a Seen Accuracy of 49.1% and an Unseen Accuracy of 52.1% in the Closed-World setting. In the more challenging Open-World setting, the model obtained 48% Seen Accuracy and 18.5% Unseen Accuracy. These results indicate that the proposed architectural design successfully balances seen and unseen performance in a dataset with a large number of diverse attribute–object compositions. On the UT-Zappos dataset, the model further demonstrated its effectiveness by achieving 66.9% Seen Accuracy and 68.8% Unseen Accuracy in the Closed-World setting, and 66.8% Seen Accuracy with 55.5% Unseen Accuracy in the Open-World setting. These strong results highlight that the proposed model is particularly well-suited to fine-grained attribute recognition tasks and can generalize effectively to unseen AO compositions.

The integration of a Retrieval-Augmented Generation module enhanced the contextual grounding of the model by enabling it to generate coherent and semantically aligned explanations for each prediction. This component addressed the research objective related to interpretability and successfully answered the research question regarding the integration of natural language explanation in a CZSL pipeline. Furthermore, comparative evaluations with state-of-the-art methods confirmed that the proposed framework delivers superior performance across multiple metrics, thereby validating its contribution to both compositional generalization and explainability.

5.2 Future Work

Although the proposed CAAO-CZSL framework delivers encouraging results, several opportunities remain for further advancement. Future research can explore the integration of larger multimodal reasoning models, which may further refine both compositional understanding and contextual explanation quality. Extending

the current design beyond static image datasets toward video-based or temporal attribute–object compositions may contribute to a more dynamic and realistic understanding of complex scenes. Another important direction involves broadening the scope of attributes to include non-visual or abstract characteristics such as emotional, functional, or contextual attributes, which would require deeper semantic alignment between visual cues and external knowledge sources.

Additionally, improving the transparency of the fusion mechanism represents a valuable line of inquiry, as a more interpretable or disentangled compositional process would strengthen the reliability of CZSL systems in high-stakes applications. Scaling the framework to larger, more diverse datasets, including those used in industrial, retail, or robotic environments, would further test its generalization abilities in real-world deployment. Beyond performance improvements, reducing computational overhead through parameter-efficient fine-tuning, model compression, or knowledge distillation may help make the system suitable for resource-constrained devices. Finally, deploying the framework in practical settings such as e-commerce product search, assistive visual systems, or robotic perception could provide deeper insights into system behavior under real operating conditions and help shape the next generation of context-aware compositional learning systems.

Bibliography

[1] M. T. Law, A. L. Traboulsee, D. K. Li, R. L. Carruthers, M. S. Freedman, S. H. Kolind, and R. Tam, "Machine learning in secondary progressive multiple sclerosis: an improved predictive model for short-term disability progression," *Multiple Sclerosis Journal–Experimental, Translational and Clinical*, vol. 5, no. 4, p. 2055217319885983, 2019.

[2] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[3] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International Conference on Machine Learning*, pp. 843–852, 2015.

[4] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[5] I. C. Udousoro, "Machine learning: A review," *Bilingual Publisher Semiconductor Science and Information Devices Journal*, 2020.

[6] P. Lison, "An introduction to machine learning." Language Technology Group (LTG), 2015.

[7] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014.

[8] D. Chatterjee, S. Ghosh, P. R. Brady, S. J. Kapadia, A. L. Miller, S. Nissanke, and F. Pannarale, 2019.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

[10] R. E. Flory, "Image acquisition technology," *Proceedings of the IEEE*, vol. 73, pp. 613–637, 1985.

[11] R. C. Gonzalez and R. E. Woods, *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008.

[12] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *The European Conference on Computer Vision (ECCV)*, 2018.

[13] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *The European Conference on Computer Vision (ECCV)*, 2018.

[14] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, "A guide to convolutional neural networks for computer vision," *Synthesis Lectures on Computer Vision*, vol. 8, no. 1, pp. 1–207, 2018.

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, pp. 740–755, Springer, 2014.

[17] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychological Review*, vol. 94, no. 2, pp. 115–147, 1987.

[18] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, 2019.

[19] A. Paivio, "Dual coding theory: Retrospect and current status," *Canadian Journal of Psychology / Revue canadienne de psychologie*, vol. 45, no. 3, p. 255, 1991.

[20] S. Rahman, S. Khan, and N. Barnes, "Polarity loss: Improving visual-semantic alignment for zero-shot detection," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[21] J. Yang, X. Ma, Y. Wu, C. Li, Z. Su, J. Xu, and Y. Feng, "Aogn-czsl: An attribute-and object-guided network for compositional zero-shot learning," *Information Fusion*, 2025.

[22] F. Pourpanah, M. Abdar, Y. Luo, X. Zhou, R. Wang, C. P. Lim, X.-Z. Wang, and Q. J. Wu, "A review of generalized zero-shot learning methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4051–4070, 2023.

[23] Z. Han, Z. Fu, S. Chen, and J. Yang, "Contrastive embedding for generalized zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2371–2381, 2021.

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

[25] Y. Atzmon, J. Berant, V. Kezami, A. Globerson, and G. Chechik, "Learning to generalize to new compositions in image understanding," 2016.

[26] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5542–5551, 2018.

[27] X. Hu, J. Jiang, X. Liu, and J. Ma, "Zmff: Zero-shot multi-focus image fusion," *Information Fusion*, vol. 92, pp. 127–138, 2023.

[28] J. Chen, Q. Li, M. Gao, W. Zhai, G. Jeon, and D. Camacho, "Towards zero-shot object counting via deep spatial prior cross-modality fusion," *Information Fusion*, p. 102537, 2024.

[29] Y. Yu, Z. Ji, J. Guo, and Y. Pang, "Transductive zero-shot learning with adaptive structural embedding," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 9, pp. 4116–4127, 2017.

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

[31] M. F. Naeem, Y. Xian, F. Tombari, and Z. Akata, "Learning graph embeddings for compositional zero-shot learning," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 953–962, 2021.

[32] M. Mancini, M. F. Naeem, Y. Xian, and Z. Akata, "Learning graph embeddings for open world compositional zero-shot learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[33] X. Ma, J. Yang, J. Lin, Z. Zheng, S. Li, B. Hu, and X. Tang, "Lvarczsl: Learning visual attributes representation for compositional zero-shot learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 12 Part 2, pp. 13311–13323, 2024.

[34] A. Nenkova and K. McKeown, "Text summarization: A review," *Foundations and Trends in Information Retrieval*, vol. 5, no. 2, pp. 103–222, 2011.

[35] Y. Atzmon, F. Kreuk, U. Shalit, and G. Chechik, "A causal view of compositional zero-shot recognition," in *Advances in Neural Information Processing Systems*, 2020.

[36] Y. Zou and K. T. Y. W. Y. M. J. M. S. C. K. T. Zhang, S. Chen, "Compositional few-shot recognition with primitive discovery and enhancing," in *ACM Multimedia Association for Computing Machinery's annual conference on multimedia*, 2020.

[37] G. Xu, P. Kordjamshidi, and J. Chai, "Prompting large pre-trained vision-language models for compositional concept learning." arXiv preprint arXiv:2211.05077, 2022.

[38] N. V. Nayak, P. Yu, and S. H. Bach, "Learning to compose soft prompts for compositional zero-shot learning," in *ICLR*, 2023.

[39] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, 2022.

[40] Y. Lu, J. Liu, Y. Zhang, Y. Liu, and X. Tian, "Prompt distribution learning," in *CVPR*, 2022.

[41] 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 *ICML*, 2021.

[42] S. Karthik, M. Mancini, L. Akata, Z., J. Yao, Y. Ge, Y. Wang, and Bi, "Kg-sp: Knowledge guided simple primitives for open world compositional zero-shot learning," in *CVPR*, 2022.

[43] S. Huang, B. Gong, Y. Feng, Y. Lv, and D. Wang, "Troika: Multi-path cross-modal traction for compositional zero-shot learning," in *CVPR*, 2024.

[44] H. Patel, L. Parmar, Shivam, J. Yao, Y. Ge, Y. Wang, and Bi, "Prompt engineering for large language model," in *Proceedings of the International Conference on Artificial Intelligence and Applications*, 2023.

[45] L. Mei, J. Yao, Y. Ge, Y. Wang, B. Bi, Y. Cai, J. Liu, M. Li, Z.-Z. Li, and D. Zhang, "A survey of context engineering for large language models," in *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering*, 2023.

[46] I. Misra, M. L. Gupta, A. Hebert, J. Yao, Y. Ge, Y. Wang, and Bi, "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.

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

[48] C. Chen and K. Grauman, "Inferring analogous attributes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 200–207, 2014.

[49] Y.-L. Li, Y. Xu, X. Mao, and C. Lu, "Symmetry and group in attribute-object compositions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11316–11325, 2020.

[50] H. Wei, Patel, L. Parmar, Shivam, J. Yao, Y. Ge, Y. Wang, and Bi, "Prompt engineering for large language model," *arXiv preprint arXiv:2304.11731*, 2023.

[51] 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*, pp. 13658–13667, 2022.

[52] S. Purushwalkam, M. Nickel, A. Gupta, L. Ranzato, M., J. Yao, Y. Ge, Y. Wang, and Bi, "Task-driven modular networks for zero-shot compositional learning," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3592–3601, 2019.

[53] M. Yang, C. Deng, J. Yan, X. Liu, L. Tao, D., J. Yao, Y. Ge, Y. Wang, and Bi, "Learning unseen concepts via hierarchical decomposition and composition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10248–10256, 2020.

[54] S. Reed, Z. Akata, H. Lee, and B. Schiele, "Learning deep representations of fine-grained visual descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–58, 2016.

[55] M. F. Naeem, Y. Xian, F. Tombari, and Z. Akata, "Learning graph embeddings for compositional zero-shot learning," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 953–962, 2021.

[56] S. Hao, K. Han, and K.-Y. K. Wong, "Learning attention as disentangler for compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15315–15324, 2023.

[57] P. Isola, J. J. Lim, and E. H. Adelson, "Discovering states and transformations in image collections," in *CVPR*, 2015.

[58] M. Yang, C. Deng, J. Yan, X. Liu, and D. Tao, "Learning unseen concepts via hierarchical decomposition and composition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10248–10256, 2020.

[59] S. Karthik, M. Mancini, and Z. Akata, "Kg-sp: Knowledge guided simple primitives for open world compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9336–9345, 2022.

[60] M. Mancini, M. F. Naeem, Y. Xian, and Z. Akata, "Learning graph embeddings for open world compositional zero-shot learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[61] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks." arXiv preprint, 2016.

[62] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *International Conference on Machine Learning*, pp. 1725–1735, PMLR, 2020.

[63] L. Mei, J. Yao, Y. Ge, Y. Wang, B. Bi, Y. Cai, J. Liu, M. Li, Z.-Z. Li, and D. Zhang, "A survey of context engineering for large language models," *arXiv preprint arXiv:2312.12149*, 2023.

[64] C. Jing, Y. Li, H. Chen, L. Shen, C., J. Yao, Y. Ge, Y. Wang, and Bi, "Retrieval-augmented primitive representations for compositional zero-shot learning," in *Proceedings of the AAAI (Association for the Advancement of Artificial Intelligence) Conference on Artificial Intelligence*, vol. 38, pp. 2652–2660, 2024.

[65] S. Huang, B. Gong, Y. Feng, M. Zhang, Y. Lv, and D. Wang, "Troika: Multi-path cross-modal traction for compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24005–24014, 2024.

[66] Y. Li, Z. Liu, H. Chen, and L. Yao, "Context-based and diversity-driven specificity in compositional zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17037–17046, 2024.

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

[68] Y. L. Li, Y. Xu, X. Mao, and C. Lu, "Symmetry and group in attribute-object compositions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[69] T. Nagarajan and K. Grauman, "Attributes as operators: factorizing unseen attribute-object compositions," in *ECCV*, 2018.

[70] D. Huynh and E. Elhamifar, "Compositional zero-shot learning via fine-grained dense feature composition," in *Advances in Neural Information Processing Systems*, 2020.

[71] M. Lewis, Q. Yu, J. Merullo, and E. Pavlick, "Does clip bind concepts? probing compositionality in large image models." arXiv preprint arXiv:2212.10537, 2022.

[72] Z. Ma, J. Hong, M. O. Gul, M. Gandhi, I. Gao, and R. Krishna, "Crepe: Can vision language foundation models reason compositionally?," in *CVPR*, 2023.

[73] M. Yuksekgonul, F. Bianchi, P. Kalluri, D. Jurafsky, and J. Zou, "When and why vision-language models behave like bags-of-words, and what to do about it?," in *ICLR*, 2023.

[74] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, 2022.

[75] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Conditional prompt learning for vision language models," 2022.

[76] A. Razdaibiedina, Y. Mao, R. Hou, M. Khabsa, M. Lewis, J. Ba, and A. Almahairi, "Residual prompt tuning: Improving prompt tuning with residual reparameterization," in *ACL*, 2023.

[77] S. Khan, Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[78] A. Lu, H. Zhang, Y. Zhang, X. Wang, and D. Yang, "Bounding the capabilities of large language models in open text generation with prompt constraints," 2023.

[79] Y. Kwon Li, Z. Liu, H. Chen, and L. Yao, "Context-based and diversity-driven specificity in compositional zero-shot learning," in *CVPR*, 2024.

[80] S. Reed, Z. Akata, H. Lee, and B. Schiele, "Learning deep representations of fine-grained visual descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–58, 2016.

[81] 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*, pp. 1721–1731, 2024.

[82] S. Huang, B. Gong, Y. Feng, Y. Lv, and D. Wang, "Troika: Multi-path cross-modal traction for compositional zero-shot learning," in *CVPR*, 2024.

[83] X. Lu, Z. Liu, S. Guo, and J. Guo, "Decomposed soft prompt guided fusion enhancing for compositional zero-shot learning," in *CVPR*, 2023.

[84] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 7, pp. 1425–1438, 2015.

[85] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model," in *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[86] Y. Li, F. Liang, L. Zhao, Y. Cui, W. Ouyang, J. Shao, F. Yu, and J. Yan, "Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm," 2021.

[87] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[88] V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," in *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[89] F. Yu, J. Tang, W. Yin, Y. Sun, H. Tian, H. Wu, and H. Wang, "Ernie-vil: Knowledge enhanced vision-language representations through scene graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 3208–3216, 2021.

[90] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[91] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," 2014.

[92] B. Meskó, "Prompt engineering as an important emerging skill for medical professionals: Tutorial," *Journal of Medical Internet Research*, vol. 25, no. Suppl 1, p. e50638, 2023.

[93] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[94] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," 2022 Association for the Advancement of Artificial Intelligence.

[95] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen, "Large language models as optimizers," 2023.

[96] H. S. Zheng, S. Mishra, X. Chen, H.-T. Cheng, E. H. Chi, Q. V. Le, and D. Zhou, "Take a step back: Evoking reasoning via abstraction in large language models," 2023.

[97] M. Yasunaga, X. Chen, Y. Li, P. Pasupat, J. Leskovec, P. Liang, E. H. Chi, and D. Zhou, "Large language models as analogical reasoners," 2023.

[98] Y. Zhou, X. Geng, T. Shen, C. Tao, G. Long, J.-G. Lou, and J. Shen, "Thread of thought: Unraveling chaotic contexts," 2023.

[99] Z. M. Wang, Z. Peng, H. Que, J. Liu, W. Zhou, Y. Wu, H. Guo, R. Gan, Z. Ni, M. Zhang, Z. Zhang, W. Ouyang, K. Xu, W. Chen, J. Fu, and J. Peng, "Rolellm: Benchmarking, eliciting, and enhancing role-playing abilities of large language models," 2023.

[100] C. Li, J. Wang, Y. Zhang, K. Zhu, W. Hou, J. Lian, F. Luo, Q. Yang, and X. Xie, "Large language models understand and can be enhanced by emotional stimuli," 2023.

[101] J. Weston and S. Sukhbaatar, "System 2 attention (is something you might need too)," 2023.

[102] A. Wilf, S. S. Lee, P. P. Liang, and L.-P. Morency, "Think twice: Perspective-taking improves large language models' theory-of-mind capabilities," 2023.

[103] Y. Deng, W. Zhang, Z. Chen, and Q. Gu, "Rephrase and respond: Let large language models ask better questions for themselves," 2023.

[104] X. Xu, C. Tao, T. Shen, C. Xu, H. Xu, G. Long, and J.-G. Lou, "Re-reading improves reasoning in language models," 2023.

[105] O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, and M. Lewis, "Measuring and narrowing the compositionality gap in language models," in *Proceedings of NeurIPS*, 2022 Association for the Advancement of Artificial Intelligence AAAI.

[106] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. T. Yih, T. Rocktäschel, S. Riedel, and L. Zettlemoyer, "Retrieval-augmented generation for knowledge-intensive nlp," 2020.

[107] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models (fid)," 2021.

[108] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "Realm: Retrieval-augmented language model pre-training," 2020.

[109] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. T. Yih, "Dense passage retrieval for open-domain question answering," 2020.

[110] O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, M. Lewis, W. T. Yih, D. Khashabi, and T. Khot, "Measuring and narrowing the compositionality gap in lms (self-ask with search)," 2022.

[111] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Ircot: Iterative retrieval with chain-of-thought for knowledge-intensive reasoning," 2023.

[112] K. Santhanam, O. Khattab, C. Hsieh, M. Zaharia, and C. Potts, "Colbertv2: Effective and efficient retrieval via lightweight late interaction," 2021.

[113] G. Fang, X. Dong, Z. Liu, and P. Luo, "Learning to prompt by watching movies," in *CVPR*, 2023.

[114] Y. Ge, R. Zhang, Y. Wu, X. Wang, and Y. Shan, "Domain adaptation via prompt learning," in *CVPR*, 2023.

[115] Y. Xiang, X. Wu, C. Jiang, and J. Zhang, "Clipa-vqa: Contrastive language-image pretraining for video question answering," in *CVPR*, 2023.

[116] C. Wu, Y. Ge, R. Zhang, and Y. Shan, "Godiva: Generative pretraining for multimodal video captioning," in *CVPR*, 2023.

[117] Y. Du, W. Huang, H. Xu, and L. Chen, "Learning to prompt for continual learning," in *CVPR*, 2023.

[118] Z. Chen, X. Jia, X. Shen, J. Sun, and J. Wang, "Learning to prompt for open-vocabulary object detection with vision-language model," in *CVPR*, 2023.

[119] S. Yang, H. Zhang, Z. Zheng, L. Li, and H. Xu, "Prompt tuning for generative multimodal pretrained models," in *CVPR*, 2023.

[120] J. Liu, Y. Liu, H. Yuan, Y. Zhang, and Q. Sun, "Prompt-aligned gradient for prompt tuning," in *CVPR*, 2023.

[121] Y. Sun, Y. Qin, J. Ji, and J. Ma, "Black vip: Black-box visual prompting," 2023.

[122] Z. Jia, H. Xu, X. Wang, Y. Wang, and J. Gao, "Visual prompt tuning," in *CVPR*, 2023.

[123] J. Gu, T. Han, Y. Wu, J. Tang, and D. Lian, "Ppt: Pre-trained prompt tuning for vision-language models," in *CVPR*, 2023.

[124] J. Zhou, H. Xu, X. Wang, Q. Sun, and W. Zhang, "Exploring compositional visual prompting for vision-language models," in *CVPR*, 2023.

[125] K. Wang, M. Song, Y. Li, H. Zhu, Z. Ma, and J. Zhang, "Prompted visual context learning for compositional zero-shot learning," in *CVPR*, 2023.

[126] Z. Zheng, H. Zhu, and R. Nevatia, "Concept-based prompt learning for compositional zero-shot learning," in *CVPR*, 2023.