Navigating the Multiverse: Enhancing Robotic Assistance through Multi-Object Navigation and Object Location Optimization

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in ***Computer Science and Engineering*** *by Research*

by

Ahana Datta
2019111007
ahana.datta@research.iiit.ac.in

International Institute of Information Technology, Hyderabad
(Deemed to be University)
Hyderabad - 500 032, INDIA
June, 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Navigating the Multiverse: Enhancing Robotic Assistance through Multi-Object Navigation and Object Location Optimization" by Ahana Datta, has been carried out under my supervision and is not submitted elsewhere for a degree.

June 7, 2024

Adviser: Prof. K. Madhava Krishna

To my parents and maternal grandparents, whose boundless love for education has been my guiding light.

Acknowledgements

I am deeply grateful to Prof. K. Madhava Krishna, my adviser and head of the Robotics Research Centre at IIIT Hyderabad, for serving as an unwavering pillar of support during my research journey, culminating in the completion of this thesis. His guidance, direction, and adept management played a pivotal role in the successful realization of this research work. I extend my sincere appreciation for his mentorship in various projects and for providing me with the opportunity to contribute to the Robotics Research Center.

I extend my heartfelt thanks to Dr. Mohan Sridharan from the University of Edinburgh, UK, for his invaluable guidance at every stage of my research journey, from formulating research problems to writing research papers.

I express my gratitude to Dr. Krishna Murthy Jatavallabhula, a postdoc at MIT CSAIL, for his insightful guidance and for facilitating my understanding of Large Language Models and their applications in the realm of Embodied AI. I also want to express my gratitude to Dr. Ragesh Kumar Ramachandran, who advised me during his time as a post-doctoral scholar at USC. He introduced me to independent research and spent countless hours explaining complex mathematical concepts. I am grateful for having the opportunity to work with him. I would like to thank my teammates in the Embodied AI project, who played a significant role in the rigorous process of formulating ideas, planning, implementation and paper writing.

I also want to express gratitude to my batchmates and friends – Tanvi Narsapur, for her constant support, aiding in balancing academics, research, and social life on campus; Meghna Mishra, for her steadfast companionship throughout my degree; Sriram Devata for his very critical feedback on my work which motivated me to work harder; Gokul Thota for providing feedback on my thesis; and Kawshik Manikantan for clearing my deep learning related queries. To all others who have lent their support, thank you for making my college years truly memorable. Additionally, I extend my heartfelt appreciation to my childhood friend, Stuti Raha, who has been by my side for the past 17 years, offering her invaluable friendship.

Finally, I owe a debt of gratitude to my parents, whose unwavering support throughout the years made all of this possible. I also express my heartfelt thanks to my late maternal grandparents for instilling a culture of education in our family and for their enduring love and support throughout the years.

Abstract

Embodied AI, where artificial agents interact with their environment through sensors and actuators, holds immense potential for real-world applications such as robotic assistance. Efficient object navigation and locating strategies are crucial for robotic assistance in real-world environments. However, existing methodologies often encounter challenges in adapting to dynamic environments and incorporating human-like reasoning for optimal decision-making. This thesis aims to bridge these gaps by addressing two fundamental challenges in embodied AI: Multi-Object Navigation (MultiON) and optimal object location within household environments.

First, we tackle MultiON, where a robot is tasked with localizing multiple instances of diverse object classes in dynamic environments. This is a fundamental task for an assistive robot in a home or a factory. Existing methods for this task have viewed this as a direct extension of Object Navigation (ON), the task of localising a single instance of an object class, and are pre-sequenced, i.e., the sequence in which the object classes are to be explored is provided in advance. This is a strong limitation in practical applications characterized by dynamic changes. We present **S**equence-**A**gnostic **M**ultiON (SAM), which is the task of locating an instance each of multiple objects in a household environment in no pre-defined order. We present a deep reinforcement learning framework for an actor-critic architecture and a reward specification. It exploits past experiences and seeks to reward progress towards individual as well as multiple target object classes. We use photo-realistic scenes from Gibson benchmark dataset in the AI Habitat 3D simulation environment to experimentally show that our method performs better than a pre-sequenced approach and a state-of-the-art ON method extended to MultiON.

Next, we present CLIPGraphs, a novel method for determining the best room to place or find objects within home environments. Existing approaches predominantly rely on large language models (LLMs) or reinforcement learning (RL) policies, neglecting commonsense domain knowledge. CLIPGraphs effectively integrates domain knowledge, data-driven methods, and multimodal learning to ascertain object-room affinities. Specifically, it (a) encodes a knowledge graph of prior human preferences about the room location of different objects in home environments, (b) incorporates vision-language features to support multimodal queries based on images or text, and (c) uses a graph network to learn object-room affinities based on embeddings of the prior knowledge and the vision-language features. We demonstrate that our approach provides better estimates of the most appropriate location of objects from a benchmark set of object categories in comparison with state-of-the-art baselines.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, there has been a surge in research focused on embodied AI and household agents, driven by the quest to revolutionize household environments. These intelligent systems combine artificial intelligence with physical bodies, aiming to address a myriad of challenges within homes autonomously. Advanced robotics, computer vision, and machine learning techniques are being employed in research to imbue these agents with the capability to navigate, understand, and interact with complex domestic settings. From cleaning and organizing to providing personalized assistance, the ongoing advancements in embodied AI for household agents hold promise for transforming daily life, ushering in a new era of smart, responsive, and seamlessly integrated home automation solutions.

Navigating the household presents multifaceted challenges for AI and household agents, encompassing dynamic environments and diverse objects. Agents must discern between objects, handle occlusions, and understand spatial relationships to execute tasks effectively. Additionally, ensuring user safety and privacy, and addressing diverse user preferences adds complexity. Overcoming these challenges necessitates advanced algorithms for perception, path planning, and context-aware decision-making, reflecting the intricate nature of household tasks that demand seamless integration of intelligence and physical action in a dynamic and unpredictable setting. In this thesis, we explore how humans perform simple everyday tasks, the thought process behind them and try to incorporate the same kind of commonsense reasoning in AI agents so that they can attempt to perform mundane daily tasks with an acceptable success rate. Specifically, we explore ways to find multiple objects in a household and utilise commonsense reasoning to encode spatial relationships between objects and rooms.

1.1 Object Goal Navigation and Multi-Object Navigation

Recent advancements in deep learning, computer graphics, and robotics have sparked a growing interest in embodied AI tasks, where artificial agents learn through interaction with their environments. Among these tasks, searching for and locating objects in indoor settings stands out as they are important yet mundane tasks that often consume a significant amount of time. Humans have a natural ability to perform this task effortlessly, locating items like spectacles, car keys, and an umbrella without specifying

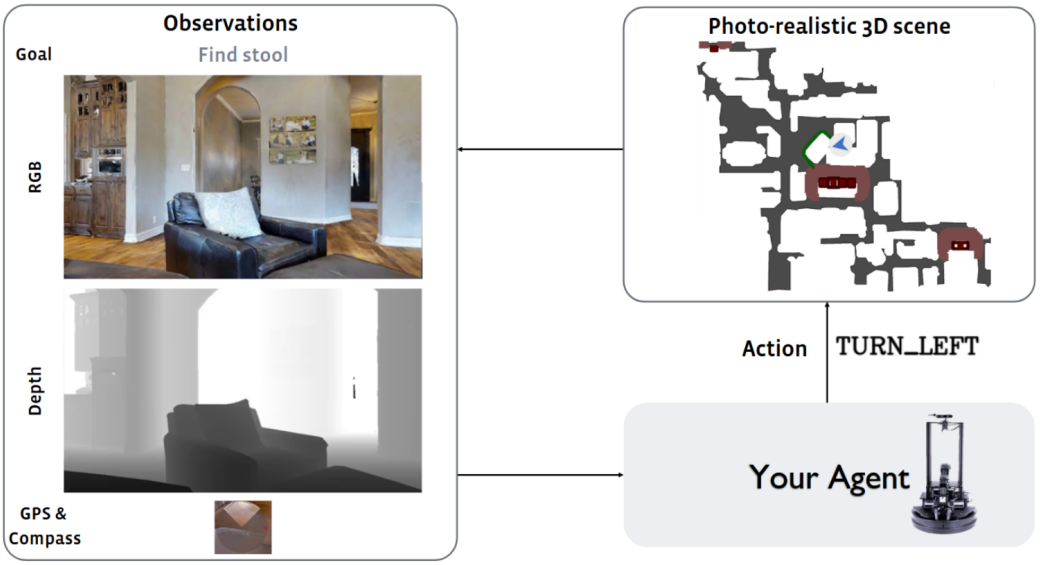
an order. Furthermore, they can seamlessly transfer this skill to new, unseen environments. Therefore, Object Navigation research has surged, motivated by this human ability.

Object goal navigation is a task in the field of embodied AI where an agent, typically an autonomous robot, is tasked with navigating an environment to find a specific object or category of objects. Unlike traditional navigation tasks where the goal might be a particular location or waypoint (Point Goal Navigation), in object goal navigation, the goal is to locate and reach any instance of a particular object category within the environment. The agent begins at a random location within the environment and uses sensors such as cameras and/or depth sensors to perceive its surroundings. It then employs navigation algorithms to plan and execute actions that will lead it towards instances of the specified object category. Figure 1.1 illustrates the Object Navigation task.

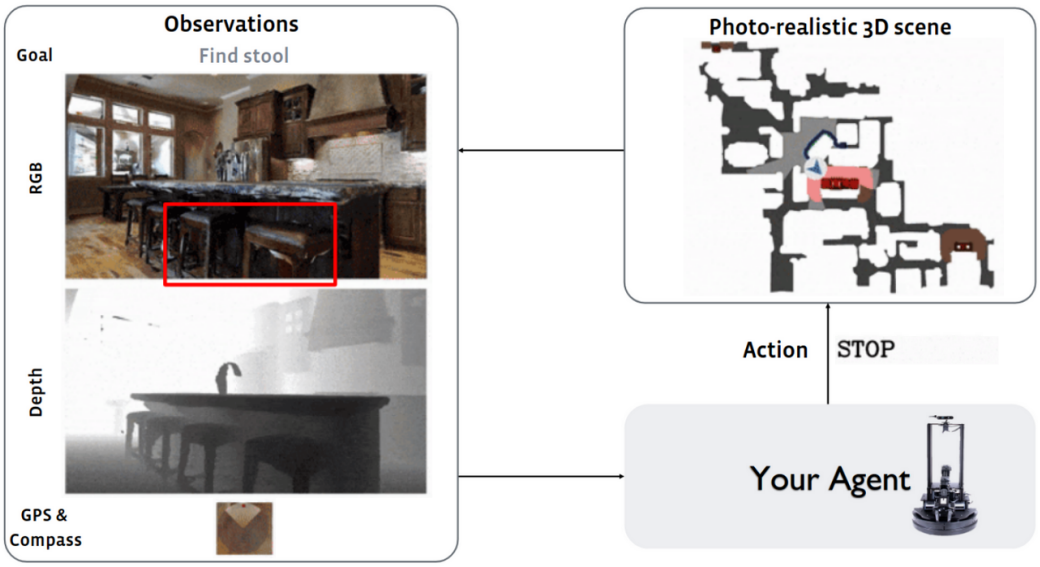
Object goal navigation tasks often involve complex environments with obstacles and varying terrain, requiring the agent to exhibit robust perception, planning, and navigation capabilities. These tasks are common in research aimed at developing intelligent robots capable of executing tasks in real-world settings, including search-and-rescue missions, object retrieval in warehouses, and household assistance. In this thesis, we narrow our focus to the practical application of household assistance. Traditional approaches to solving object navigation tasks usually require extensive domain expertise, such as a detailed understanding of the environment's layout. However, this requirement is unfeasible in numerous real-world scenarios. Therefore, our approach involves constructing an allocentric semantic map of the world as the agent explores previously unseen environments.

A limitation of object navigation strategies is that they are primarily focused on single objects, thus limiting their applicability in various indoor scenarios where multiple categories of objects need to be located within a single search or navigation session. Chapter 3 talks about the Multi-Object Navigation (MultiON) task, which involves a robot localizing instances of multiple object classes - a task crucial for assistive robots in various indoor settings. The MultiON task as proposed by Saim Wani et al. [5], describes the task as "a task framework that involves navigation to an ordered sequence of objects placed within realistic 3D interiors". The MultiON task extends the object goal navigation task initially introduced by Zhu et al. [6] and Anderson et al. [7], offering greater flexibility by enabling the selection of multiple object goals, thus allowing for increased control over task complexity. However, this "ordered sequence" or *pre-sequenced* approach limits adaptability because in real-world household settings, we, as humans, usually tend to locate objects from a list as we find them in the environment, which is often faster than following the order in which the objects are listed. This is another limitation addressed in this thesis.

In chapter 3, we present **S**equence-**A**gnostic **M**ultiON (SAM), where the agent is provided with a list of objects to locate in a household environment without any explicit order specified for finding the objects. We introduce a deep reinforcement learning framework for SAM, employing an actor-critic architecture with a refined reward system. The approach proves more adaptable to dynamic scenarios, emphasizing its potential for practical applications in evolving environments. The motivation for this work is to encode some commonsense reasoning into the agent so that it can find multiple objects faster.



(a) Agent is initialised in the Living Room. According to the agent's observations, it executes actions from {MOVE_FORWARD, TURN_LEFT, TURN_RIGHT}.



(b) After executing some actions, the agent finally reaches the Kitchen where the goal category *stool* is present. After reaching the goal, the agent executes the STOP action.

Figure 1.1: **Object Navigation Task**: Here, the task for the agent is to reach the goal category stool. The agent has access to RGB-D camera and a noiseless GPS+ Compass sensor. We can see the explored area marked in light grey and the goal region marked as red in the occupancy map as the agent moves to the goal in Figure (b). Adapted from the Habitat 2020 Challenge.

The framework was evaluated in the AI Habitat 3D simulation environment using the Gibson benchmark dataset. Our experiments conclusively demonstrate that our framework outperforms pre-sequenced MultiON and state-of-the-art object navigation methods when adapted for MultiON scenarios.

1.2 Object-Room Affinities

Throughout the evolution of society, humans have exhibited a penchant for organizing their surroundings. While the norms governing the arrangement of our living spaces have evolved, certain enduring associations persist within the human collective. When tasked with locating an item such as "Headphones", humans typically rely on intuitive associations with specific areas, such as the "Home Office", "Living Room", or "Entertainment Room." This simple thought experiment underscores the subconscious associations we have formed between objects and the rooms they traditionally occupy.

In Chapter 4, we introduce a novel method for determining the best room to place an object in for scene rearrangement or to find an object in for Multi-Object Navigation. While state-of-the-art approaches rely on large language models (LLMs) or reinforcement learning (RL) policies for this task, our approach, CLIPGraphs, efficiently combines commonsense domain knowledge, data-driven methods, and recent advances in multimodal learning. Specifically, it (a) encodes a knowledge graph of prior human preferences about the room location of different objects in home environments, (b) incorporates vision-language features to support multimodal queries based on images or text, and (c) uses a graph network to learn object-room affinities based on embeddings of the prior knowledge and the vision-language features. We demonstrate that our approach provides better estimates of the most appropriate location of objects from a benchmark set of object categories in comparison with state-of-the-art baselines.

1.2.1 Traces of Utility in Our Decision-Making Process

The mapping of objects to specific spaces often stems from considerations of "utility." For instance, when tasked with finding "an apple" in an unfamiliar house, one instinctively recognizes it as a perishable food item, suggesting it should be located in areas designated for food storage and consumption, such as the Kitchen, Pantry, or Dining Room.

The Familiarity Factor and Decision-Making Time: Examining our inference process in stages can provide further insight. Consider a scenario where the object to be found has an unknown name, and its nature—whether it's a writing instrument, a skincare product, or a magical item from Narnia—is unclear. In such a situation, a systematic approach involves posing questions to ascertain its utility: Is it consumable? Is it a cosmetic? Is it an electronic device? This initial understanding of utility guides the identification of the room where objects with similar utility are typically located, streamlining the decision-making process with increased familiarity.

This thought experiment elucidates two significant insights:

- Successfully addressing these questions allows humans to infer the appropriate location for an object, providing a universal understanding applicable to *any* household.
- Humans not only possess commendable Object-Room mappings in their cognitive processes but also leverage object-object relationships, shaped by the co-occurrence of these objects and influenced by their shared utility.

In Chapter 4, our objective is to employ a graph network to cluster commonly encountered household objects into room categories. The fundamental rationale behind utilizing a Graph Convolutional Network lies in harnessing both Object-Room Relationships and Object-Object Relationships to generate latent representations indicative of object categorization into rooms based on shared features or utility.

Graph Convolutional Networks utilize not only node features but also incorporate a message-passing mechanism between neighbouring nodes. This mechanism enables the generation of node representations that consider information from neighbouring nodes in addition to the inherent node features.

To accomplish this, we leverage the Human Preference Dataset [8] to identify prevalent and universally accepted object-room mappings. We then integrate this dataset with multimodal CLIP [9] features, employing them to learn utility-based clustering using our graph network. This approach enables us to capture both the inherent relationships between objects and the preferences embedded in human cognition, providing a comprehensive framework for object categorization within domestic spaces.

This categorization, while insightful, is not without its limitations. Human organization of household items isn't solely guided by utility; at times, it's influenced by the "ease of use." For instance, although, from a utility perspective, a "water bottle" is typically associated with the "kitchen", our practical experiences inform us that it's more likely to be found in areas where people commonly spend their time. Retrieving objects in homes with such preferences poses a challenge solely relying on generalized utility-based categorization.

The complexities of object retrieval under these circumstances fall beyond the scope of this current work. Addressing such user-specific behaviours in multi-object navigation is a facet we plan to explore in future endeavours in Multi-Object Navigation.

1.3 Thesis Contributions

The overall contributions of the thesis can be summarized as follows:

1. **Development of SAM Framework**: The thesis introduces the Sequence Agnostic MultiON (SAM) framework inspired by human cognition. SAM allows robots to explore environments without needing to compute a global order of tasks. Instead, it adapts its exploration based on observations, minimizing distance travelled and task completion time. This framework contributes by:

(a) Introducing a method for greedy selection of 'long-term goals' to minimize distance to instances of target object classes.

(b) Extending previous work on Object Navigation (ON) to develop a deep reinforcement learning (RL) framework for SAM, which rewards concurrent progress towards multiple object classes [4].

2. **Introduction of CLIPGraphs Framework**: The thesis presents CLIPGraphs, a framework designed to accurately estimate object-room affinities by leveraging commonsense knowledge, data-driven methods, and multimodal embeddings. This framework incorporates:

   (a) A knowledge graph encoding human preferences for the room location of objects in home environments.

   (b) Joint embeddings of image and text features to facilitate multimodal learning and queries [10].

   (c) A graph network that learns object-room affinities based on latent embeddings of the knowledge graph, including image and text feature embeddings.

3. **Dataset Contribution**: The thesis contributes a dataset of over 8000 image-text pairs, extracted from the Web for the 268 benchmark object categories described in [8]. This dataset is crucial for evaluating the CLIPGraphs framework's ability to accurately estimate the optimal room location for any given object, which is a crucial step in object navigation and scene rearrangement.

Evaluation and Validation. This thesis evaluates both frameworks rigorously. For SAM, task completion time and distance travelled are compared against the previous state-of-the-art PSM method. Experimental results demonstrate significant improvements over baselines, validating the effectiveness of the proposed framework. For CLIPGraphs, the thesis evaluates the ability to estimate the best room location for objects.

Contribution to Robotics and AI. Both frameworks contribute to advancing robotics and artificial intelligence research. SAM enhances robot autonomy by enabling efficient exploration in unknown environments without predefined sequences. CLIPGraphs improves object navigation and scene rearrangement tasks by accurately estimating object-room affinities, leveraging multimodal embeddings and commonsense knowledge. Overall, the thesis significantly advances the state-of-the-art in robotic exploration and object navigation tasks through the development and validation of the SAM and CLIPGraphs frameworks, along with the creation of the associated dataset, contributing to the broader fields of robotics, artificial intelligence, and human-robot interaction.

1.4 Thesis Organisation

The chapters in this thesis are as described as follows:

Ch1: This is the introductory chapter, which discusses the scope of the work carried out in this thesis, introduces the research problems we are tackling, and outlines the motivation behind some of the methods we have adopted to solve them.

Ch2: In this chapter, we undertake a thorough review of the literature related to the problems we are tackling. We also briefly describe a few essential basics which are later used in our proposed methods.

Ch3: We present the first contribution of this thesis - a framework to solve the Multi-Object Navigation problem in a sequence-agnostic way.

Ch4: This chapter presents the second contribution of this thesis - a novel method to determine the best room to place an object in, especially useful for scene rearrangement and object navigation tasks.

Ch5: We conclude with a summary of the methods and results discussed in this thesis and the scope of extension of this work in the future.

Chapter 2

Background

In this chapter, we give a brief overview of the related work in embodied AI tasks – single object and multi-object search tasks, commonsense reasoning and the use of CLIP embeddings to solve embodied AI tasks.

2.1 Embodied AI tasks

Research in Embodied AI has gained momentum with the availability of datasets containing 3D scenes of indoor environments, such as Matterport3D [11], Gibson [12], and Habitat-Matterport 3D [13]. These datasets are utilized in AI simulators like Habitat [14] and GibsonEnv [12] to explore various problems, including PointGoal Navigation [15, 16], where a robot is tasked with moving to a specified point location; ObjectGoal Navigation [4, 17–19], which involves finding a target object instance in an unexplored environment; and Visual Language Navigation [20], which requires navigating to a target scene based on complex instructions and scene descriptions.

Apart from goal navigation tasks, in recent efforts to train embodied agents for human-like activities, the following tasks have also been explored: scene exploration [21, 22]; embodied QA [23–25]; and rearrangement [26–28]. ALFRED [29], TEACh [30], and [31] investigate agents' ability to perform actions based on natural language instructions, while [32–34] utilize knowledge graphs for visual classification and detection. While these works include explicit specification of the goal state by a human agent, recent works have started the inclusion of reasoning with commonsense knowledge to enable agents to perform these tasks intelligently.

2.2 Object search tasks

Object search tasks, such as Object Navigation (ON), are designed to navigate to specific target objects swiftly within previously unseen environments. End-to-end reinforcement learning methods have emerged as the state-of-the-art approach for directly mapping pixels to actions tailored to accomplish such tasks. Mousavian et al. [35] leveraged semantic visual representations from advanced detectors and

segmentors for target-driven visual navigation. Zhu et al. [6] proposed a deep reinforcement learning framework to enhance target and scene generalizability. However, despite their effectiveness, deep reinforcement learning methods often struggle to generalize to novel scenarios due to the lack of a semantic understanding of the environment. In contrast, approaches emphasizing better generalization construct allocentric maps encoding semantic priors [19, 21, 36, 37].

In this thesis, we shift the focus towards searching for multiple objects, a more relevant challenge in household navigation scenarios. Real-world navigation often entails locating several objects concurrently, making multi-object navigation a practical and pertinent challenge.

2.3 Multiple object search tasks

The goal of the MultiON task [5] is for a robot to localize an instance (each) of more than one object class. The original challenge description introduced coloured cylinders in the Habitat environment, with only one instance of each object class, and provided the sequence in which instances of the target object classes were to be localized. This reduced MultiON to a fixed sequence of ON tasks, which we refer to as pre-sequenced MultiON (PSM). Methods for PSM have predominantly developed and evaluated different map representations and memory architectures. For example, the original paper on MultiON explored *NoMap*, *Oracle Map*, and *Learned Map* agents under the assumption of noiseless pose estimation [5]. Also, their RL policy rewarded the agent's progress toward, as well as the localization of, an instance of the target object class under consideration. Kim et al. [38] sought to decouple mapping from localization, with the policy being a combination of exploration and moving towards a particular object class instance. There has also been work exploring the role of auxiliary tasks in improving navigation performance, specifically by taking into account object instances that had already been observed while executing the policy [39]. However, most of these studies only account for one instance per object class in the environment. Essentially, this means that only a single instance of each object class, such as "chair," exists within the environment. Furthermore, these objects are typically synthetic ones introduced artificially into the environment. Another limitation in these approaches is the Pre-Sequenced MultiON (PSM) formulation, where the order of object localization is predetermined for the agent. However, this approach poses practical challenges in real-world scenarios where flexibility is crucial. Ideally, in such scenarios, we should have the ability to identify and mark an object off our list even if it is not found in the sequence originally specified.

In contrast to the existing methodologies, Chapter 3 introduces several advancements. Firstly, we allow for the existence of multiple instances of each object class, thereby better reflecting real-world scenarios. Additionally, we focus on naturally occurring object classes within Gibson indoor scenes, eliminating the need for artificially induced objects. Moreover, we alleviate the limitations of the PSM formulation by shifting our focus to Sequence-Agnostic MultiON, enabling the agent to locate objects without strict sequential constraints. To achieve this, we adapt a previously established deep network

architecture for Object Navigation (ON) [4] to consider embeddings of additional inputs and a novel reward specification.

2.4 Commonsense Reasoning

In the context of rearrangement, Housekeep [8] and TIDEE [40] focus on organising a household using commonsense reasoning, utilising the training of Large Language Models (LLMs). Meanwhile, CSR [41] employs reasoning derived from a scene graph to identify objects and changes in room states. Other works like JARVIS [42], DANLI [43], and LLM-Planner [44] show the effectiveness of prompting LLMs for language understanding and sub-goal planning using natural language instructions. Furthermore, [45] evaluates the performance of various language models and explores their limitations related to commonsense in the physical world.

2.5 CLIP for Embodied AI

CLIP (Contrastive Language-Image Pre-training) [9] uses large-scale text-image pairs for training image and text encoders simultaneously and has shown remarkable performance for object recognition. The effectiveness of CLIP image and text embeddings for Embodied AI tasks has been evidenced by recent studies [27, 46–48] over traditional ResNet-based architectures. [49] demonstrated the use of CLIP to match objects in a cross-instance setting with visual features as a measure of similarity to complete tabletop object rearrangement tasks. A recent work, ZSON [50], proposes a zero-shot object navigation agent that uses CLIP embeddings to localize objects in the environment and navigate towards them without any additional training. The agent leverages the semantic similarity between the object category name and the visual features of the object to guide its exploration. Similarly, CLIP was used by [51, 52] for zero-shot vision and language navigation by using natural language expressions for descriptions of target objects. Recent works [53–57] use pixel-level CLIP features for robotic navigation using language commands. Furthermore, [58, 59] have demonstrated the use of CLIP visual and language embeddings for learning robotic scenes, while [60, 61] use CLIP for generating 3D scene memories from 2D images and natural language.

2.6 Graph Convolutional Networks

Graph Convolutional Networks (GCNs) mark a significant advance in deep learning, transforming how we analyze graph-structured data. Introduced by Kipf and Welling in 2017 [2], GCNs extend principles from CNNs to graphs, offering a versatile framework. Unlike traditional CNNs for grid-structured data, GCNs adapt to irregular graph structures, using the adjacency matrix to capture relationships be-

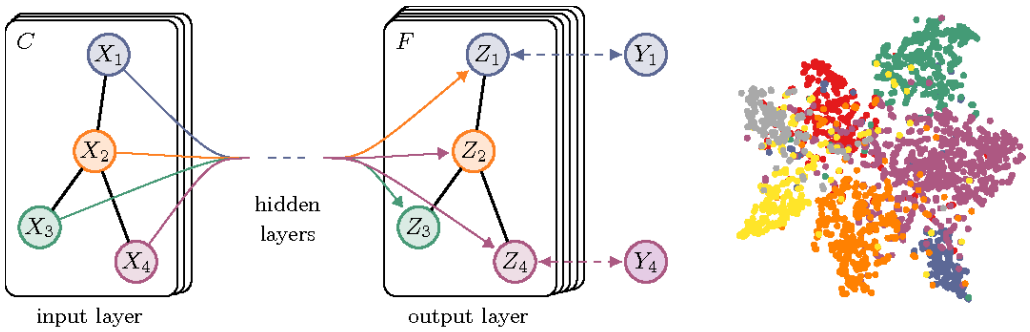


Figure 2.1: Left: Illustration of a multi-layer Graph Convolutional Network (GCN) for semi-supervised learning, featuring C input channels and F feature maps in the output layer. The labels are denoted by Y_i; the black lines show the graph edges. The graph structure is shared across layers. Right: t-SNE representation of hidden layer activations of a two-layer GCN trained on the Cora dataset [1] using 5% of labels. Colours denote document class. Source: [2]

tween nodes. This adaptability makes GCNs effective for tasks like node classification, link prediction, and graph classification.

In GCNs, edges go beyond mere connections; they notably shape node embeddings, especially in weighted graphs. Edge weights add complexity, serving as a quantitative measure of connection strength. During processing, a GCN takes a graph with an adjacency matrix as input, where each edge is associated with a weight. The impact of edge weights on node embeddings becomes evident, with nodes connected by stronger edges exerting a significant influence. This dynamic information propagation shapes how nodes are represented within the graph.

Chapter 3

Sequence-Agnostic Multi-Object Navigation

In this chapter, we introduce a deep reinforcement learning framework for **S**equence-**A**gnostic **M**ultiON, employing an actor-critic architecture with a refined reward system.

(Published and presented at the IEEE International Conference on Robotics and Automation (ICRA) 2023 held in London, UK. [62])

3.1 Introduction

Figure 3.1: Example trajectories of the same episode when agent traverses in Pre-Sequenced MultiON (PSM) and Sequence Agnostic MultiON (SAM). In this particular episode, the goal objects are specified as {chair, toilet, couch}. Paths taken by PSM and SAM are shown in red and blue. Semantic annotations for the Gibson Tiny split [3] have been used here.

Consider the home environment in Figure 3.1 with instances of object classes such as *bed* and *toilet* in different rooms. A core task for an assistive robot in such an environment is to locate instances

of specific object classes. The Object Goal Navigation (ON) task [6, 19] requires a robot to find an instance of a single object class in an unknown environment. In contrast, the Multi-Object Navigation (MultiON) task [5] involves an AI agent being given a list of objects to find in an unknown household environment. The agent's objective is to explore and locate one instance of each specified object within the environment. Thus, the MultiON task is a generalization of the ON task. Both ON and MultiON tasks are relevant to many practical applications.

Humans perform MultiON tasks with seemingly little effort. For example, a human going for a walk may need a pair of socks, their house keys, and their pink umbrella. To locate these objects, they build on prior experience in this and other related (home) environments, to explore a series of locations likely to contain one or more of these objects. Also, humans try to concurrently minimize the distance to all target object classes and adapt their exploration based on observations. For example, if keys are observed unexpectedly on top of the cabinet while searching for socks, the human will stop and confirm whether these are the house keys. State-of-the-art methods for MultiON, on the other hand, focus on ON tasks or perform **P**re-**S**equenced **M**ultiON (PSM) in which the robot is given the sequence in which the target object classes are to be explored [5]. For example, the robot in Figure 3.1 (left sub-figure) is given the sequence $\{chair, toilet, couch\}$ as the MultiON task. It first searches for and finds a chair; although it spots a couch as it moves near the chair to confirm the chair's location. It then searches for a toilet (second object class in the sequence) before coming back to confirm the location of the couch seen earlier.

Inspired by insights from human cognition, we present a framework for **S**equence **A**gnostic **M**ultiON (SAM); the robot is neither provided nor forced to compute a global order in which it locates instances of the target object classes. Instead, the robot explores likely locations of the target objects and automatically adapts its exploration based on observations. In Figure 3.1 (right sub-figure), for example, the robot observes a chair and a couch nearby. It first confirms the location of the couch before confirming the chair's location, then explores further to locate a toilet; the distance traveled and the task completion time are substantially less compared with PSM in Figure 3.1 (left sub-figure). Specifically, our framework makes two key contributions:

1. Instead of computing the globally optimal sequence of trajectories by evaluating all possible paths through locations in the domain, the robot builds on past experience in environments with a similar distribution of regions and objects, greedily choosing a series of 'long -term goals' in an attempt to concurrently minimize the distance to an instance of all target object classes.

2. Extends the previous work on ON [4] to develop a deep reinforcement learning (RL) framework for SAM, introducing a novel reward specification and adapting the actor-critic network to reward the concurrent progress to instances of multiple object classes, instead of only rewarding the progress towards identifying an instance of a single object class.

We experimentally evaluated our framework through ablation studies, and quantitative and qualitative comparisons with relevant baselines using photo-realistic scenes from the Gibson benchmark dataset in

AI Habitat, a 3D simulation environment [14]. We computed five standard performance measures over experimental trials involving different number of object classes. These experiments demonstrated the significantly better performance provided by our framework, compared with the use of a predetermined sequence of object classes, and with two other methods for selecting long-term goals: random search, and a state-of-the-art deep RL method for ON [19] extended to MultiON. In particular, our SAM framework provided $\approx 50\%$ reduction in the number of time steps and path length compared with a PSM baseline.

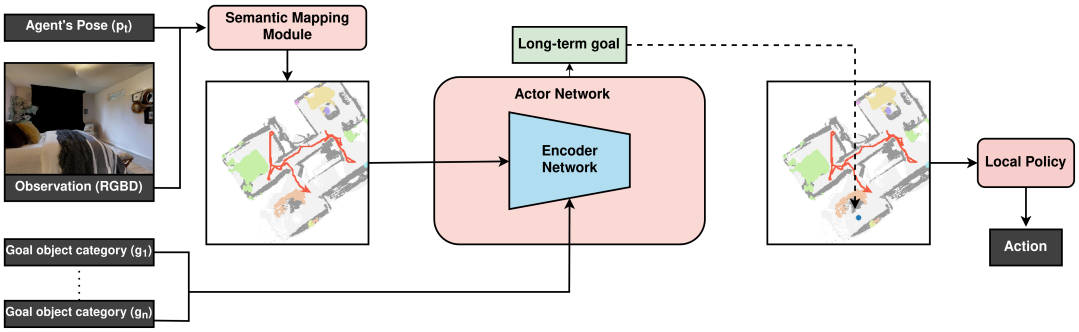


Figure 3.2: Our framework comprises three primary elements. The *Semantic Mapping* module utilizes odometry pose readings and RGB-D observations to create an allocentric semantic map of the local environment. The *Encoder Network* receives the semantic map and encodings of the target object classes and extracts the high-level feature embeddings from the semantic map. An actor-critic network is then trained using the feature embeddings. This network outputs a long-term goal to search and find an instance of the target object classes. The deterministic *Local Policy* uses analytical planners to compute low-level navigation actions to reach the specified long-term goal.

3.2 Problem Formulation and Framework

This section describes the Sequence-Agnostic MultiON (SAM) task that we are solving and our proposed framework for the same.

3.2.1 Task Description

In each episode of SAM, the robot must locate an instance (each) of a set G of one or more target object classes in the environment. The environment consists of at least one (and often two or more) instance(s) of each object class G_i. At each timestep t in the episode, the robot receives: (a) egocentric RGB and depth observations of the scene within the robot's view; (b) the robot's pose in the domain; and (c) an one-hot encoding for each object class (out of $N = 16$ classes) whose instance is to be located. We use k-ON to refer to an episode with k target object classes. The robot does not have any prior map

of the domain or the sequence in which the target object classes are to be explored; if such a sequence is provided, it is a PSM task. An object class is considered to be $found$ when the robot navigates close (distance to success or target, $d_s \leq 1m$) to an instance of the class; if no such instance is found within a maximum number of timesteps, the episode is said to be unsuccessful. The robot can execute one of these four actions: $\{$*MOVE_FORWARD, TURN_LEFT, TURN_RIGHT, STOP*$\}$. The `MOVE_FORWARD` action moves the robot forward by $0.25m$, whereas the `TURN` actions cause the robot to rotate by 30° in the appropriate direction (`LEFT, RIGHT`).

3.2.2 Proposed Framework

As stated earlier, we pursue a deep RL formulation of SAM and present a modular approach based on the actor-critic architecture. Figure 3.2 provides an overview of this framework, which extends the prior work on ON [4], and comprises three modules:

1. **Semantic Mapping:** Builds a map of the domain (i.e., metric arrangement of space with obstacles and empty space) from the RGB-D (RGB and depth) data and pose observations. Robot uses the map to localize itself and processes the RGB-D data to identify and localize specific object instances in this map.

2. **Encoder Network:** Receives as input the semantic map (above) and the encodings of target object classes and extracts high-level features. These features are sent to an Actor-Critic network that repeatedly computes a 'long-term goal', i.e., a region the robot should travel to in search of an instance of a target object class.

3. **Deterministic Local Policy:** Uses analytical planners to compute the low-level navigational actions that need to be taken to reach the current long-term goal region.

The semantic map constructed by the Semantic Mapping module is a matrix of dimension $K \times M \times M$. This contains K channels of $M \times M$ size maps where $K = C + 2$ and C is the total number of semantic categories. The first two channels contain the obstacles and the explored areas respectively, while the remaining channels contain the C object categories. We use the mapping procedure from a state-of-the-art method for ON [19]. We leverage a pretrained Mask R-CNN model [63] to infer semantic categories from the RGB data observed. Utilizing the depth observations, we generate point clouds, associating each point in the point cloud with its corresponding estimated semantic category. With the help of differentiable geometric computations on each point within the point cloud, we build a voxel representation that is then transformed into a semantic map of dimension $(C+2) \times M \times M$. The semantic mapping module is represented in Figure 3.3. We also use random shift augmentation on the predicted semantic map to promote generalization.

The encoder network takes as input the estimated semantic map from the previous module, the robot's current and past locations, the objects found and localized so far, and the encoding of the target object

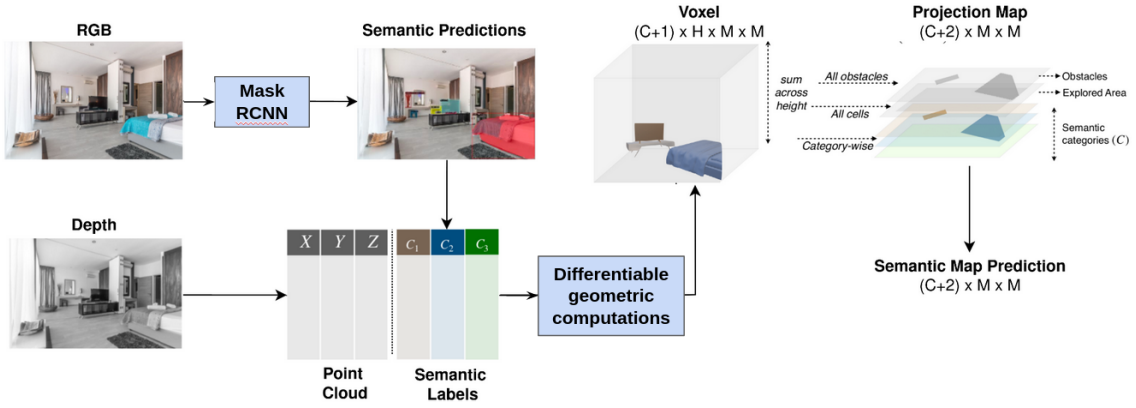


Figure 3.3: **Semantic Mapping Module**: RGB and Depth images undergo a series of operations to generate a top-down Semantic Map. Source: [4]

classes. *The one-hot encoding of multiple object classes was not considered in earlier works* [4]. High-level feature embeddings are extracted by the encoder and then used by the actor network to obtain a long-term goal, i.e., the next location the robot should move to for the target object search. The encoder network consists of 4 convolutional layers with 3×3 kernels and 32 channels [64]. After each convolutional layer, we apply the ReLU activation. A stride length of 1 is used everywhere. A single fully-connected layer, normalized by the LayerNorm operation [65], receives the output of these layers. Also, a hyperbolic tangent non-linear transform is applied to the 50-dim output of the fully-connected layer. We use orthogonal initialization [66] to initialize the weight matrix of the fully connected layer and the convolutional layers. The bias is set as zero.

The output of the encoder network is used by the actor-critic network. The actor and critic components work with the same weights in the convolutional layers but have separate encoders. The weights in the convolutional layers can only be updated by the optimizer in the critic network. We apply the clipped double Q-learning approach introduced by Van Hasselt et al. [67] to the critic network. This strategy involves representing each Q-function with a three-layer multi-layer perceptron (MLP), incorporating ReLU activations after each layer except the last. All the transition states are stored using a replay buffer. The transition states include the semantic map, target object classes, action, reward, next semantic map, and the subsequent target object classes. The encoder and actor-critic networks receive two inputs: (i) a set of transition states obtained from the replay buffer and (ii) the augmented semantic map. Every 25 timesteps, a new long-term goal is sampled. Note that the parameters of both the actor and critic networks are revised during training. Once trained, only the actor network is used for testing. For ease of understanding, only the actor network is labelled in Figure 3.2. The specification of the reward function, a key contribution of this chapter, is described in Section 3.2.3.

When the actor-critic network provides a long-term goal, the local policy module uses the Fast Marching Method [68] to guide the robot to this region. Specifically, from the semantic map generated in the semantic mapping module, the obstacle channel is used to compute the shortest path from

the current location to the current long-term goal. The robot then computes the low-level navigational actions to navigate along the computed shortest path.

3.2.3 Reward Function

A key contribution of this work is the reward specification used to train our deep network. Recall that our objective is to mimic the intuitively appealing sequence-agnostic behaviour of humans engaged in MultiON tasks. In order to do so, we identified three desired characteristics that we wanted our reward function to capture:

1. We wanted to encourage the robot to find an instance of each target object class.
2. We wanted to motivate the robot to concurrently reduce the distance to an instance of more than one object class. However, we did not want it to compute the globally optimal exploration sequence by considering all possible sequences because that could be computationally intractable in practical deployment.
3. We wanted to encode *non-procrastination*, i.e., minimize the time spent looking for object instances.

Based on these desired characteristics, we formulated the reward function as follows:

$$Reward = R_{\text{sub-goal}} + \alpha_{process} * R_{process} + CNR \tag{3.1}$$

where $R_{sub-goal}$ is the reward for achieving a 'sub-goal', i.e., an instance of one of the target object classes; $R_{process}$ is the process reward; and CNR is the negative reward, i.e., cost (currently -0.01) accumulated at every timestep. The value of CNR was set such that the penalty accrued over a typical episode was small relative to the other parts of the reward function. We used scaling factor $\alpha_{process} = 0.1$ (determined experimentally) to vary the relative influence of $R_{process}$ on the overall behaviour of the robot.

3.2.3.1 Sub-goal reward

$R_{\text{sub-goal}}$ is the standard reward the robot receives when it localizes an instance of any target object class. To ensure that this reward is only received at the corresponding timestep, we modelled it as:

$$R_{sub-goal} = 1_{sub-goal} * r_{\text{sub-goal}} \tag{3.2}$$

where $1_{sub-goal}$ is an indicator function that is equal to 1 *iff* the robot reaches an instance of one of the target object classes, and $r_{sub-goal}$ is the instantaneous real-valued reward. This can be restated as:

$$R_{sub-goal} = \begin{Bmatrix} r_{sub-goal} & \text{if a sub-goal is reached} \\ 0 & \text{otherwise} \end{Bmatrix} \tag{3.3}$$

We experimentally set $r_{sub-goal} = 2$ to be relatively higher than the other two parts of our reward, in order to enable the robot to reach the sub-goal with higher priority.

3.2.3.2 Process reward

We recognized that this part of the reward function may require a trade-off with the first part of the reward ($R_{\text{sub-goal}}$), e.g., focusing on the shortest path to a particular region may help the robot obtain $r_{\text{sub-goal}}$ as soon as possible but it may make sense to deviate from this region to another region nearby if an instance of another target object class is likely to be found there.

We first used the known (i.e., ground truth) location of object instances during training to compute the distance to the closest instance of each target object class at each timestep. We used this information to compute, at each timestep t, the total decrease in the geodesic distance to the nearest instance of each target object class g_i:

$$d_t = \sum_{i}^{N} (dtg_{i,t-1} - dtg_{i,t}) \tag{3.4}$$

where $dtg_{i,t}$ refers to the shortest distance to an instance of object class i at timestep t; and N is the number of target object classes whose instance remains to be localized in this episode. Once d_t is computed, $R_{process}$ is computed as:

$$R_{process} = \begin{cases} \frac{n}{N} + d_t & \text{if } dtg \text{ of } n \text{ classes decreases} \\ d_t & \text{otherwise} \end{cases} \tag{3.5}$$

where the additional reward received depends on the fraction of the target object classes to whose instances the robot was able to reduce its distance during the training episode. This part of the reward thus encourages the robot to greedily attempt to concurrently localize more than one object based on prior experience in similar environments and on the observations received in the current episode. Note that *the reward function's components remain the same irrespective of the number of target object classes*. Experimental results (below) demonstrate the benefits of this reward specification and our SAM framework.

3.3 Experimental Setup

We evaluated our SAM framework's capabilities using photo-realistic benchmark scenes (3D reconstructions of home environments) from the Gibson dataset in the AI Habitat simulator [14]. We used the standard ObjectNav Challenge's 25 scenes during training by setting up ≈ 1000 episodes (total) of k-ON task, randomly selecting $k \in [2, 3]$ target object classes in each episode. The robot's starting position is randomly sampled from navigable points in the environment in each episode. For testing, we generated datasets for 2-ON and 3-ON tasks using five scenes from the Gibson dataset that were not considered during training. This testing dataset included 200 episodes for each of the five scenes, resulting in a total of 1000 episodes. We did not explore k-ON tasks for $k > 3$ in this benchmark dataset, as the number of scenes containing one or more instances of the target object classes was less for higher k. Figure 3.4 shows the distribution of object goals in each scene. Every object was chosen almost an equal number of times in every scene, which shows that our dataset is well randomized and not biased.

Scene	Chair	Couch	Potted Plant	Bed	Toilet	TV
Collierville	79	82	80	0	88	71
Corozal	61	70	59	65	70	75
Darden	71	81	79	84	85	0
Markleeville	74	91	73	80	82	0
Wiconisco	91	105	100	0	104	0

(a) Validation dataset distribution for 2-ON

Scene	Chair	Couch	Potted Plant	Bed	Toilet	TV
Collierville	105	121	124	0	124	126
Corozal	95	102	106	109	90	98
Darden	114	121	115	126	124	0
Markleeville	112	119	129	113	127	0
Wiconisco	150	140	155	0	155	0

(b) Validation dataset distribution for 3-ON

Figure 3.4: The tables above depict the distribution of goal objects across the 5 unseen scenes utilized as a validation dataset. Objects with a count of 0 indicate their absence in the respective scene.

As stated in Section 3.2.1, the robot's observations were in the form of $4 \times 640 \times 480$ RGB-D images, and the success threshold $d_s = 1m$. The maximum episode length was 1000 and 600 timesteps for the 3-ON and 2-ON episodes, respectively. The target object classes were those considered by the state-of-the-art approaches for ON [19]: 'chair', 'couch', 'potted plant', 'bed', 'toilet' and 'tv'. We experimentally evaluated the following hypotheses about our framework:

H1: Our SAM framework traverses shorter paths and takes fewer timesteps when compared with PSM.

H2: Our SAM framework provides better long-term goals than the state-of-the-art object navigation baseline extended to multi-object settings.

We considered three baselines for evaluation:

1. **Random:** The robot pursued the SAM task (i.e., no fixed object class sequence) but it chose an action randomly in each timestep of each episode. This is a standard comparative benchmark in literature to understand the margin of improvement by other methods.

2. **PSM:** The robot was given an order (i.e., sequence) in which it had to explore the target object classes; the underlying framework was the deep RL approach we used for ON in the prior work [4].

3. **Multi-Semantic Exploration (M-SemExp):** The robot pursued the SAM task, but the underlying deep RL method extended a state-of-the-art method developed for ON [19] to the MultiON setting. In particular, the reward was specified as the total decrease in geodesic distance to the nearest instance of each object g_i.

$$R_{SemExp} = \alpha_{SemExp} * \sum_{i}^{N} (dtg_{i,t-1} - dtg_{i,t}) \tag{3.6}$$

where $dtg_{i,t}$ is the shortest distance to an instance of object class g_i at timestep t; N is the number of target object classes whose instance remains to be localized.

We have considered five standard performance measures:

(i) **Success (%):** Fraction of episodes in which the robot successfully localizes an instance of each target object class within the maximum number of steps allowed.

(ii) **Sub-success (%):** Ratio of the number of target object classes whose instance has been localized to the total number of target object classes in an episode.

(iii) **Timesteps:** The total number of timesteps taken to successfully complete a particular episode. Any move, such as moving forward or a rotation move, adds a unit to the timestep count.

(iv) **Global Path Length (m):** The length of the path (in meters) traversed by the robot to successfully identify an instance of each target object class in an episode.

(v) **Global-SPL (G-SPL, %):** Ratio of the globally minimum path length (g) and the length of the actual path taken by our robot in a particular episode; g is computed as the shortest path that visits one instance of each target object class.

$$\text{G-SPL} = (success) * \frac{g}{max(g, p)} \tag{3.7}$$

where p is the length of the actual path traversed by the robot to localize an instance of each target object class.

Furthermore, we included some qualitative results in Section 3.4.1.

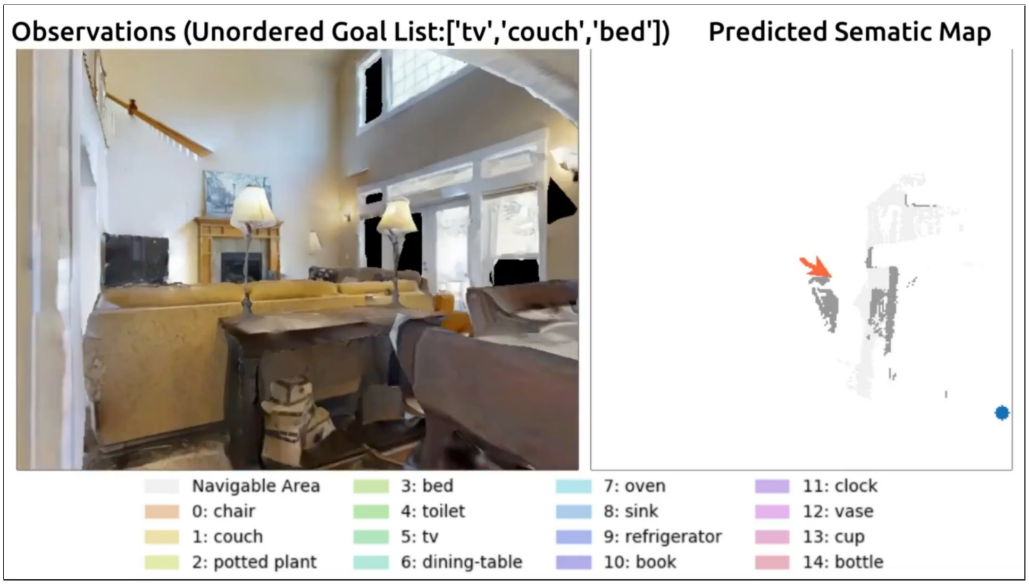
3.4 Experimental Results

This section describes the qualitative and quantitative results of the experimental evaluation of our SAM framework. Figure 3.5 describes two snapshots of the output we get on executing our framework.

3.4.1 Qualitative Results

Figure 3.6 shows a qualitative comparison of our SAM framework with the PSM baseline in the form of snapshots for a specific episode. Recall that PSM is provided with the sequence {couch, tv, toilet} in which the target object classes are to be explored, whereas only the target object classes are known in the SAM formulation. With our SAM framework, the robot quickly moves toward and localizes a toilet in timestep 21, a couch by timestep 72, and a TV by timestep 81. With PSM, on the other hand, the robot takes more than 200 timesteps to complete this task. These results partially support **H1**.

Figure 3.7 shows a qualitative comparison of the proposed SAM framework and the M-SemExp baseline. For both methods, a snapshot of the episode was taken when a new long-term goal was chosen.



(a) Snapshot after the first few timesteps of an episode showing the long-term goal (blue dot)

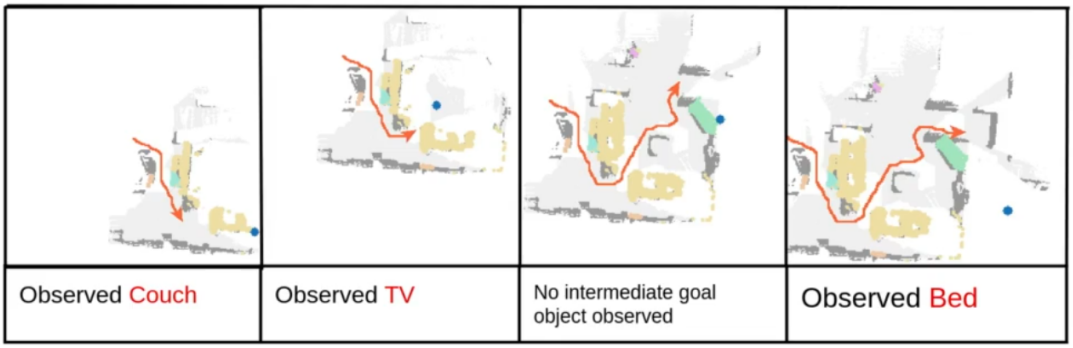
(b) Snapshot showing the semantic map when a goal object is found. The goal is marked in blue on the semantic map.

Figure 3.5: The figures show different timestamps of the same episode. In each snapshot, we have the remaining goal objects to be found on the top as "Unordered Goal List". The image on the left of each snapshot shows the agent's view of the scene at that timestep. The semantic map created so far is shown on the right side of each snapshot. Additionally, a legend for the semantic map is provided at the bottom of each snapshot. The semantic map also illustrates the agent's path in red, with an arrow indicating the direction of the agent's movement.

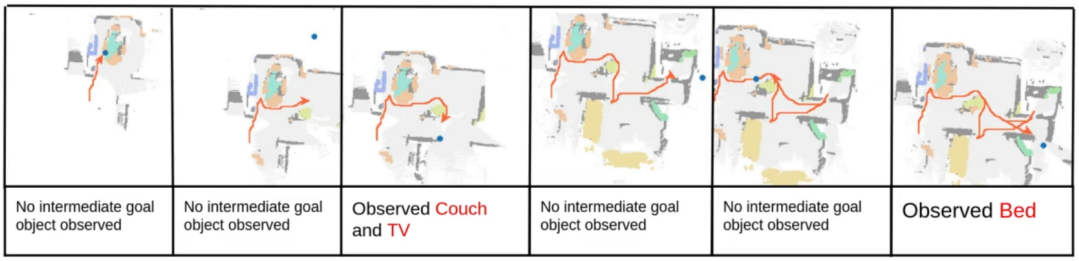
(a) Sequence-Agnostic MultiON

(b) Pre-Sequenced MultiON

Figure 3.6: Qualitative comparison of the performance of (a) our SAM framework with (b) PSM in an episode of the MultiON task with three target object classes (couch, tv, toilet) using experiment snapshots. In every snapshot, we show the scene observation (above) and the semantic map generated (below). Over the scene image, we have mentioned the timesteps (t) completed and the object that was found in that snapshot. We can see that our SAM framework results in a smaller number of timesteps.



(a) SAM framework



(b) Multi-Semantic Exploration

Figure 3.7: **Analysis of long-term goals**: Example trajectories of the same evaluation episode are shown, (a) using the SAM framework and (b) using the M-SemExp framework, to demonstrate the performance of our reward function compared to M-SemExp's reward function. Comments below each snapshot describe the events during the pursuit of each specific long-term goal.

Both frameworks were used to search for an unordered list of objects. With the SAM framework, the agent finds the objects {tv, couch, bed}, with only 4 long-term goals, whereas M-SemExp requires 6 long-term goals. Our SAM framework resulted in a shorter path $11.3m$ and also completed the task in 86 timesteps, while M-SemExp resulted in a total path length of 19.3m and 182 timesteps. This result can be summarised in 3.8. These results partially support **H2**.

3.4.2 Quantitative Results

We then evaluated **H1** quantitatively, comparing our SAM framework with the PSM baseline, with the results averaged over the successful episodes in 200 *paired* episodes of five scenes summarized in Table 3.1. Note that both frameworks had the same environment, the same robot starting position and the same three target object classes in each paired episode. To facilitate a fair comparison, we first ran each episode with SAM, i.e., with no constraint on the order in which an instance of the target object classes is to be found. The order in which the robot ended up localizing the target object classes was

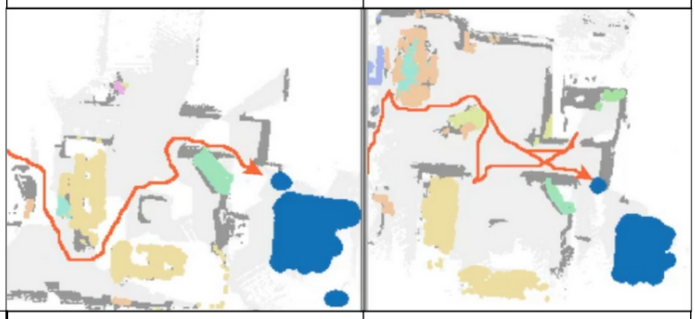
	Our SAM Framework	M-SemExp
		
Total timesteps	86	182
Total Path Length	11.3m	19.3m
Total Long-term goals	4	6

Figure 3.8: Summary of long-term goal analysis of SAM and M-SemExp

then provided as the target sequence for the PSM approach. As shown in Table 3.1, our SAM framework provided an $\approx 50\%$ reduction in the average number of timesteps and the average path length compared with the PSM framework. This improvement in performance with our SAM framework was strongly influenced by our design of the reward function for the deep network architecture; see Section 3.2.3. These results strongly support **H1**.

To evaluate **H2**, we compared our SAM framework with the Random and M-SemExp baselines; recall that the latter was obtained by adapting a state-of-the-art deep RL framework for ON [19]. Table 3.2 summarizes the corresponding results for 2-ON and 3-ON tasks. We observed that our SAM framework provided substantially better performance in all three measures considered. Also, performance improved when the number of target object classes increased from 2-ON to 3-ON because of the associated increase in the maximum number of timesteps and our framework's attempt to concurrently reduce the distance to an instance of all target object classes. These results strongly support **H2**.

3.4.3 Ablation studies

Next, we performed ablation studies to further explore the effect of the maximum number of allowed timesteps on the performance of our framework compared with the M-SemExp baseline. Specifically, we varied the maximum permissible number of timesteps from 200-600 for the 2-ON task, and from 300-1000 for the 3-ON task, with the results summarized in Table-3.3 and Table-3.4 respectively. We observed that the degradation in performance as the maximum number of timesteps is reduced was less with our framework than with M-SemExp. These results further reinforced the fact that objects are localized in fewer steps as a result of pursuing a sequence-agnostic approach, and of encouraging the robot to concurrently reduce the distance to an instance of multiple target object classes. Due to the

Scene Name	Timesteps ↓		Global Path Length (m) ↓	
	PSM	SAM	PSM	SAM
Collierville	242	**122**	29.53	**16.16**
Corozal	336	**179**	46.23	**27.28**
Darden	248	**117**	31.43	**16.08**
Markleeville	272	**140**	35.41	**18.87**
Wiconisco	389	**224**	52.81	**33.94**

Table 3.1: Our SAM framework provides significantly better performance than the PSM framework, with numbers averaged over 200 paired episodes for each of the five testing scenes.

Method	Success (%) ↑		Sub-success (%) ↑		G-SPL (%) ↑	
	2-ON	3-ON	2-ON	3-ON	2-ON	3-ON
Random	3.3	4.7	11.5	14.2	0	0
M-SemExp	60.5	61.7	73.1	76.6	30.5	29.8
SAM (ours)	**70.7**	**72.3**	**82.5**	**86.9**	**39.3**	**39.3**

Table 3.2: Our SAM framework results in significantly better performance than the *Random* and *M-SemExp* baselines on three key measures; results averaged over 200 episodes of each of the five testing scenes.

lack of varied objects in the existing data sets (i.e., scenes), for 4-ON, a subset of the data was used and our framework still provided better results than the M-SemExp baseline in each of the different timesteps tested upon. For example, the success rate in 800 timesteps for the M-SemExp baseline and our SAM framework were 61% and 64.6% (respectively) for four target object classes. These results provide further support for hypothesis **H2**.

3.5 Summary

Object Navigation (ON) methods aim to locate instances of a single object class, while state-of-the-art Multi-Object Navigation (MultiON) methods are pre-sequenced (PSM), meaning they follow a predetermined order for exploring target object classes. This chapter introduces Sequence-Agnostic

Timesteps	M-SemExp		SAM (our method)	
	Success (%)	Sub-success (%)	Success (%)	Sub-success (%)
600	60.5	73.1	**70.7**	**82.5**
300	49.6	67.2	**60.3**	**77.1**
200	34.7	56.5	**48.6**	**70.2**

Table 3.3: **2-ON Task:** Comparison of our SAM framework with the M-SemExp baseline for different values of the maximum number of timesteps allowed in each episode.

Timesteps	M-SemExp		SAM (our method)	
	Success (%)	Sub-success (%)	Success (%)	Sub-success (%)
1000	61.71	76.6	**72.3**	**86.9**
500	41.34	64.6	**63.7**	**84.3**
300	32.49	55.0	**45.3**	**76.4**

Table 3.4: **3-ON Task:** Comparison of our SAM framework with the M-SemExp baseline for different values of the maximum number of timesteps allowed in each episode.

Multi-object Navigation (SAM), explaining the concept and presenting a deep reinforcement learning-based framework for its implementation. We describe a tailored reward function for the actor-critic network, designed to promote sequence-agnostic exploration. Through experiments conducted in the Habitat 3D simulation environment using scenes from the Gibson dataset, we demonstrate a significant performance enhancement over three baselines: random action selection, PSM, and an extension of the SemExp method [19] to MultiON (M-SemExp). Our approach relies solely on imparting commonsense to a robotic agent through prior experience and reinforcement learning. However, we recognize that humans benefit from the correlation between objects and rooms in real-world environments, a topic we will explore in the following chapter.

Chapter 4

CLIPGraphs

In the previous chapter, we discussed the Multi-Object Navigation task in a household setting. We also discussed a modular approach based on the actor-critic architecture for this particular task. In this chapter, we introduce CLIPGraphs, a novel method for determining the best room to place/find an object in, a precursor for the task of embodied scene rearrangement and object navigation.

(Published and presented at the IEEE International Conference on Robot and Human Interactive Communication (RO-MAN) 2023, also accepted for the International Conference on Robotics and Automation (ICRA) 2023 Workshop on Pretraining4Robotics.)

4.1 Introduction

Imagine a robot being tasked with tidying up an unfamiliar house. This task is a variant of the *scene rearrangement* challenge for embodied AI [26]. To perform this task, the robot must first determine what tidying up means in this specific house, which requires constructing a representation of the current state of the house and inferring a possible goal state (i.e., a configuration in which the house is deemed *tidy*). Any errors in this step can influence downstream planning and control, resulting in irrecoverable failure. Computing the most appropriate room location for specific object categories is thus critical to the successful completion of such tasks.

Human-inhabited environments such as homes and offices are designed to be functional and aesthetically pleasing. A key characteristic of such environments is the semantic organization, i.e., objects are placed in locations based on their purpose. This enables humans to adapt efficiently to new environments designed to serve the same purpose. For example, when a person enters a new home and wants to find sugar to make a cup of coffee, they instinctively look in the kitchen or pantry. We leverage this semantic organization to enable robots to predict the likely locations of any given object. Specifically, we leverage recent developments in multimodal (vision-language) representation learning to propose a flexible approach for learning *object-room affinities*, i.e., the relative likelihood of any given object belonging to a particular room in a house, based on image and text input.

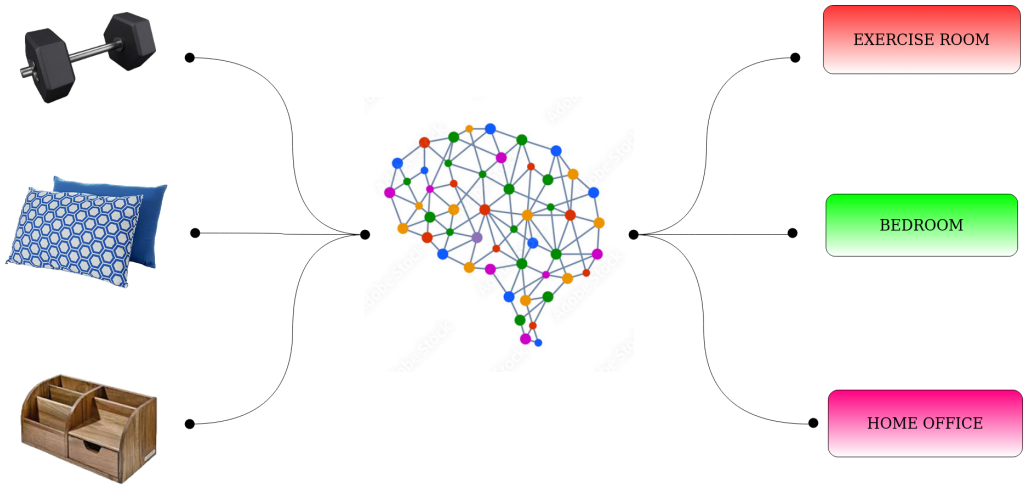


Figure 4.1: Our method leverages semantic organization (e.g., "dumbbells are usually in the exercise room") to better compute the most suitable location for any given object.

State-of-the-art methods have used Large Language Models (LLMs) as *commonsense* reasoning machinery for this *tidy up* task [8]. These methods are limited to textual descriptors, which can be challenging to ground to a specific scene. Moreover, they use ground truth object labels for generating object-room affinities, which limits their operation outside of the training data distribution. Others have used reinforcement learning (RL) to compute policies for related tasks such as visual semantic navigation [4, 19, 50, 55], and Multi-Object Navigation [62, 69, 70], but do not fully leverage knowledge from different sources in the learning process.

Our framework, *CLIPGraphs*, seeks to leverage the complementary strengths of commonsense knowledge, data-driven methods, and multimodal embeddings to estimate object-room affinities accurately. It does so by incorporating:

1. A *knowledge graph* that encodes human preferences of the room location of objects in home environments;

2. Joint embeddings of image and text features [10] to support multimodal learning and queries in the form of images or text; and

3. A graph network that learns object-room affinities over a dataset of common household objects based on latent embeddings of the knowledge graph that includes the image and text feature embeddings.

The novelty lies in the combination of these components to achieve the desired objective. We evaluate our framework's ability to accurately estimate the optimal room location for any given object, which is a crucial step in object navigation and scene rearrangement. This is done by creating a dataset of over

8000 image-text pairs, extracted from the Web for the 268 benchmark object categories described in [8]. Experimentally, we demonstrate that our framework significantly improves performance compared to state-of-the-art baselines, which include Large Language Models (LLMs) and language embeddings encoding common sense knowledge of object locations.

4.2 Datasets

We employed object and room categories, along with their corresponding human annotations, sourced from Housekeep [8]. In addition, we meticulously curated a visual counterpart for the 268 object categories mentioned in Housekeep [8].

4.2.1 Human Preference Dataset

The Human Preference Dataset, crafted by Housekeep [8], aims to comprehend individuals' preferences for organizing everyday household objects in both orderly and disorderly settings. The dataset resulted from a study conducted on Amazon MTurk with 372 participants. For each object-room pair, participants were tasked with classifying the available receptacles of that room into three categories:

- **Misplaced:** Subset of receptacles where the object is found in untidy houses.
- **Correct:** Subset of receptacles where the object is found in tidy houses.
- **Implausible:** Subset of receptacles where the object is unlikely to be found, either in a tidy or untidy house.

Participants were also asked to rank all the receptacles present in that room. Interested readers are referred to their paper for detailed information on data collection, filtering processes, and its application to Scene Rearrangement.

4.2.2 IRONA Dataset

To perform MultiON or scene rearrangement tasks, a robot needs the key ability to accurately compute the appropriate location for any given object. To explore this ability, we created the *Images for Room-Object Nexus through Annotations* (IRONA) dataset of 30 RGB images from the Web per object category. These images consist of white-background catalogue shots corresponding to the same 268 categories of household objects used by Housekeep [8] for benchmarking the Scene-Rearrangement task. For each image in the IRONA dataset, the robot is tasked with computing the probability of the object belonging to each of the 17 room categories.

4.2.2.1 Rooms

We considered the same 17 room categories as employed by Housekeep [8]:

Rooms		
bathroom	bedroom	childs_room
closet	corridor	dining_room
exercise_room	garage	home_office
kitchen	living_room	lobby
pantry_room	playroom	storage_room
television_room	utility_room	

4.2.2.2 Object Categories

The list of 268 object categories we utilized can be accessed here. A representative image of our IRONA dataset can be seen in Figure 4.2.

4.3 Problem Formulation and Framework

Our framework, called CLIPGraphs, trains a Graph Convolutional Network (GCN) [2] to compute embeddings that are used to estimate these object-room affinities. Figure 4.3 shows the training pipeline. It uses a knowledge graph to encode existing information of human preferences (of room location of objects) for the object categories [8], and incorporates a modified contrastive loss function to compute better latent embeddings of the image and language encoder features provided by CLIP [9] for the nodes of the knowledge graph. The resultant node embeddings model the information about the room location of various objects in the latent space. During inference, the CLIP features generated for any (test) RGB images are processed by the GCN, with the cosine similarity between the embeddings of the rooms and the image providing the desired estimate of object-room affinities. We describe the individual components of our framework below.

4.3.1 Knowledge Graph

In the initial step of our framework, we utilize human-annotated preferences from the Housekeep data [8]. For each object-room pair, 10 human annotators ranked receptacles based on the likelihood of correct or incorrect object placement. Consequently, there are 10 opinions (positive, negative, or zero) for each object-room-receptacle tuple. To ensure reliability, we filter the dataset for good annotator agreement.

Figure 4.2: Representative Image Of Our Web Scraped Dataset; 1 image per object category

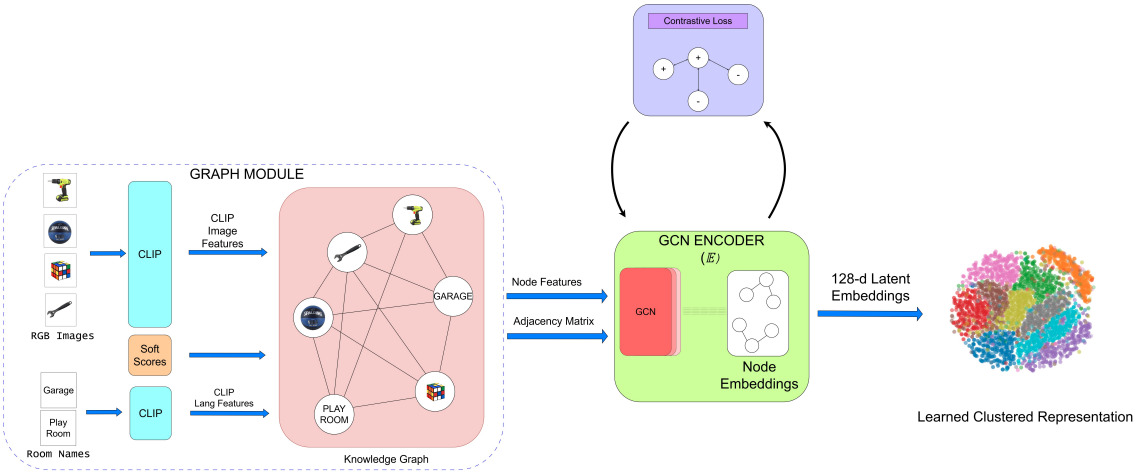


Figure 4.3: *CLIPGraphs* constructs a graph module (bottom-left) using CLIP encoders and passes that to a GCN Encoder($\mathbb{E}$) module. The encoder is trained using contrastive loss to create better node embeddings that bring similar embeddings closer. Visualization of final layer activations confirms the formation of well-defined node clusters.

Positive (negative) soft scores are computed as the mean of positive (negative) reciprocal preferences for all receptacles related to a specific object-room pair. We select the room with the highest positive-scored receptacle to determine ground truth object-room mappings. Other rooms in the domain are assigned the mean negative soft score of receptacles in that room.

To populate a knowledge graph using available annotated information, we partition the IRONA web-scraped dataset into training, validation, and test sets with a 15:5:10 ratio of images per object category. The knowledge graph is instantiated with each image of the training set as a node, along with room names, resulting in 4037 nodes (268*15 + 17). Five types of edges connect the nodes (see Figure 4.4): (1) self-edge (edge weight=1); (2) edge between images of the same object (edge weight=1); (3) edge between two objects in the same ground truth room; (4) edge between an object and its correct room node; and (5) edge between an object and its incorrect room nodes. Weights for edges of types 4 and 5 are based on the object-room soft scores. Edges of type 3 receive a randomly chosen weight between 0.5 to 0.7, while edges of type 1 and 2 are assigned a weight of 1. Table 4.1 summarizes various information about our knowledge graph that is being used for training purposes.

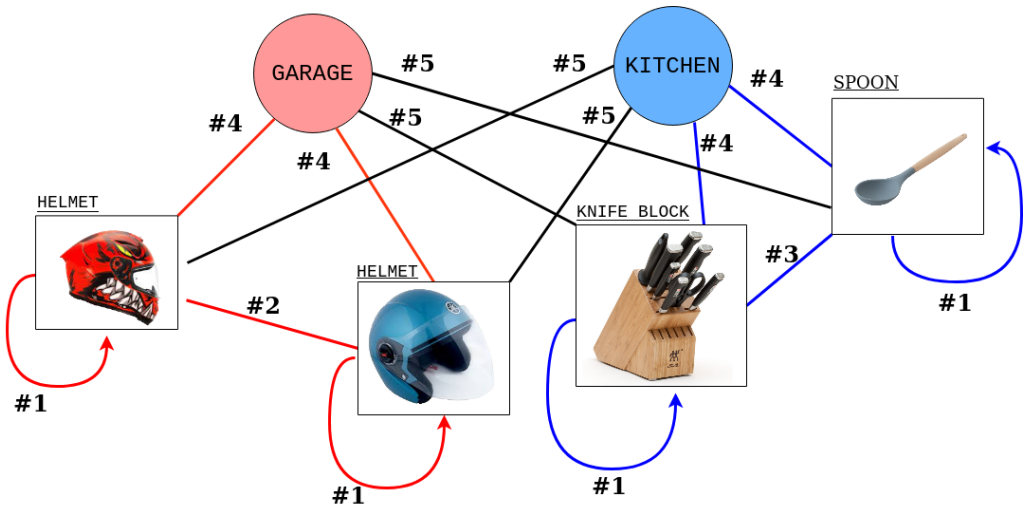


Figure 4.4: An illustration of the five types of edges in our knowledge graph. Coloured edges denote positive weights, while black ones denote negative weights. Numbers on the edges indicate the edge types.

Statistics About our Knowledge Graph	**Value**
#Nodes	4020 Object Images Nodes & 17 Room Nodes
#Edges	7,66,649
Self Loops	Yes
Train Images	268*15 Images
Test Images	268*10 Images
Val Images	268*5 Images
Types of edges	weighted undirected

Table 4.1: Statistics for the Knowledge Graph created using the web scraped dataset

4.3.1.1 Nodes

There are currently two types of nodes in our knowledge graph

- **Room Nodes:** Since we have 17 room categories, there are 17 such nodes. For our proposed method, the node features for these nodes are generated using the CLIP language encoders.
- **Object Nodes:** In our proposed method, for each of the 268 object categories, 15 images are chosen for training, and for each of the 268*15 nodes, we generate node features using the CLIP image encoders [9]. However, we also create CLIP Language Embeddings for these 268 Object Categories for baseline comparisons.

4.3.2 Edge Weights

For assigning edge weights to various edges between our nodes, we use the ranks provided by Housekeep Human Preference Dataset for each *object-room-receptacle*[1] combination. We further calculate soft scores using algorithm 1, which takes these ranks as input.

For each object-room-receptacle combination, 10 annotators were given just 3 categories to classify the combinations. Therefore for each object-room-receptacle, there could either be a majority of *positive ranks*, *negative ranks*, or *implausible ranks*:

1. For any object-room-receptacle combination if a majority of annotators (> 5) gave positive ranks[2]; we calculate a $positive\ score$

2. For any object-room-receptacle combination if a majority of annotators (> 5) gave negative ranks[3]; we calculate a $negative\ score$

3. However, if *implausible* ranks were in majority then we need to modify those a bit. According to Housekeep [8], an *implausible* combination was such a combination that could neither occur in an *untidy* house nor a *tidy* house. Therefore, it accounts for the maximum negative object-room-receptacle affinity. Thus, we assign -1, i.e., the maximum possible negative score to such object-room-receptacle pairs.

Finally, we would have a dictionary that records whether the majority opinion for every object-room-receptacle was *positive*, *negative* or *implausible*(-1).

4.3.2.1 Correct Object-Room Mappings

To get the correct object-room mappings, we compare which object-room-receptacle has the highest *positive* score for a given object. We assign that room as the correct object-room mapping.[4]

[1] *"receptacle" was a categorization used by Housekeep [8] to define 128 flat horizontal surfaces in a household where objects can be found - misplaced or correctly placed*

[2] A receptacle with "+1" rank is a more appropriate for an object as compared to a receptacle with a "+2" rank for the same object-room pair.

[3] A receptacle with "-1" rank is a receptacle where humans are more prone to keep objects in an *untidy* state of the house as compared to a receptacle with a "-2" rank for the same object-room pair.

[4] Our future extension would be to extend this mapping to top-K correct rooms

4.3.2.2 Incorrect Object-Room Mappings

Once we've created the correct object-room GT mapping, the edge weights of the other 16 rooms will need to be allotted. So, for each such object-room pair, there can be 3 cases:

1. **All/majority positive receptacles**: We assign $-\epsilon$ as edge weights
2. **All/majority negative receptacles**: We assign the mean of all the negative scored receptacles
3. **All/majority implausible receptacles**: We assign the mean of all the negative scored receptacles. (since we had already assigned implausible receptacles with -1)

Algorithm 1 Generating ORR Positive Soft Scores

for each object in objects **do**
  for each room in rooms **do**
    $ranks \leftarrow object - room - receptacle - ranks$
    $score\_dict \leftarrow \{\}$
    for each receptacle in the room **do**
      $combination\_score \leftarrow []$
      for each rank given for the combination **do**
        if $rank > 0$ **then**
          $combination\_score.append(1/rank)$
        end if
      end for
      if $len(combination\_score) >= 5$ **then**
        $score\_dict[combination] = sum(combination\_score)/max\_len\_pos$[5]
      else
        $score\_dict[combination] = 0$
      end if
    end for
  end for
end for

[5]For a given object, max_len_pos/neg/imp is the maximum number of annotators that gave positive/negative/implausible ranks across all room-receptacle pairs.

Algorithm 2 Generating ORR Negative Soft Scores

for each object in objects **do**
  for each room in rooms **do**
    $ranks \leftarrow object - room - receptacle - ranks$
    $score\_dict \leftarrow \{\}$
    for each receptacle in the room **do**
      $combination\_score \leftarrow []$
      $min\_neg\_rank = min(all\ ranks\ for\ the\ ORR\_combination)$
      for each rank given for the combination **do**
        if $rank < 0$ **then**
          $rank = rank + min\_neg\_rank + 1$
          $combination\_score.append(1/rank)$
        end if
      end for
      if $len(combination\_score) >= 5$ **then**
        $score\_dict[combination] = sum(combination\_score)/max\_len\_neg^5$
      else
        $score\_dict[combination] = 0$
      end if
    end for
  end for
end for

Algorithm 3 Generating ORR Implausible Soft Scores

for each object in objects **do**
  for each room in rooms **do**
    $ranks \leftarrow object - room - receptacle - ranks$
    $score\_dict \leftarrow \{\}$
    for each receptacle in the room **do**
      $combination\_score \leftarrow []$
      for each rank given for the combination **do**
        if $rank == 0$ **then**
          $combination\_score.append(-1)$
        end if
      end for
      if $len(combination\_score) >= 5$ **then**
        $score\_dict[combination] = sum(combination\_score)/max\_len\_imp$[5]
      else
        $score\_dict[combination] = 0$
      end if
    end for
  end for
end for

For example, for a particular object-room-receptacle combination, the calculation of positive soft scores is shown:

$$\begin{aligned}
\text{knife-bottom_cabinet ranks:} &\quad -1, -3, 0, 1, 3, 2, -2, 5, -1, 4 \\
\text{Filter the negative ranks:} &\quad 1, 3, 2, 5, 4 \\
\text{Take the reciprocal of ranks:} &\quad \frac{1}{1} + \frac{1}{3} + \frac{1}{2} + \frac{1}{5} + \frac{1}{4} = 2.367 \\
\text{Calculate the mean of reciprocal ranks:} &\quad 2.283/5 = 0.45
\end{aligned}$$

For each object, the ground-truth room is decided by choosing the room containing the highest-positively scored receptacle for that object. Every other room in the domain is assigned the mean negative soft score(for a given object-room pair) of all the receptacles present in that room.

4.3.2.3 Node embeddings

After creating the basic knowledge graph, we initialize node embeddings using the pretrained CLIP model's high-dimensional embeddings. Object nodes are initialized with corresponding CLIP image encoder embeddings, and room nodes with corresponding CLIP language encoder embeddings. This captures the appearance of objects and known associations between objects and rooms based on the large dataset used to train CLIP embeddings.

We experiment with three pretrained CLIP architectures: Vision Transformer (ViT), ResNet-50, and ConvNeXt. ViT-H/14 [71] is trained on LAION-2B, a 2.3 billion subset of LAION-5B [72] with English captions. ResNet-50 [73] uses OpenAI's pretrained weights [9], and ConvNeXt base [74] is pre-trained on LAION-400m [75], which contains 400 million image-text pairs[6]. The three CLIP architectures we utilized, as given by OpenCLIP [10], have the following dimensionalities: ViT-H/14 and RN50, each with a dimensionality of 1024 features, and ConvNeXt-base with a dimensionality of 512 features. These were chosen after taking into consideration the datasets they were trained on and their performance in other embodied AI tasks. Once CLIP embeddings are associated with knowledge graph nodes, we proceed to the next steps in our training pipeline.

4.3.3 GCN Training

The next step of training involves feeding node embeddings, each with 512 or 1024 dimensions depending on the chosen CLIP architecture, and the adjacency matrix (representing the knowledge graph structure) into a Graph Convolutional Network (GCN) [2]. This step aims to acquire improved latent space embeddings for our knowledge graph. GCNs excel in capturing non-linear relationships between nodes and learning from both local and global graph structures. Consequently, nodes with higher similarity are mapped to points that are closer in the latent embedding space, while dissimilar nodes are mapped to more distant points in the latent space. For instance, the output 128-dimensional

[6]Implementation used existing code [76]

#	Hyperparameter	Value
1	Node Feature Size	512(ConvNeXt) / 1024 (Others)
2	Output Node Embedding	128
3	GCN [2] Layers	3
4	Learning Rate	10^{-3}
5	Learning Rate Schedule	StepLR: step size=1000 , $\gamma = 0.25$
6	Temperature	0.01
7	Batch Size [Loss]	15
8	Negatives Per Batch	40
9	Epochs	5k

Table 4.2: Hyperparameter choices for our Graph Based Network to learn latent representations of CLIP Visual Encoder Features

GCN (object) embedding for a microwave will have a higher cosine similarity with the output 128-dimensional GCN (language) embedding for the kitchen. Table 4.2 shows the hyperparameters used for our GCN model.

4.3.3.1 Loss Function

An essential aspect of the training process involves selecting an appropriate loss function. Previous research has extensively explored functions such as contrastive loss [77], triplet loss [78], and multi-class N-pair loss [79]. Recent studies have highlighted the advantages of employing the loss function introduced in the CLIP-Fields method [61]. We adapt this loss function to better leverage the knowledge graph created using the IRONA dataset and human preference annotations.

Our GCN is trained using a contrastive loss function similar to the one described in the CLIP-Fields method [61]. The objective is to cluster similar embeddings closer in the latent space and map dissimilar embeddings to points that are further away in the latent space. We tailor the basic loss function to our problem formulation and incorporate additional information from edge weights. The loss L is defined as:

$$L = -e^{-weight_{+\bullet}} \log \left(\frac{e^{(sim_{+\bullet}/T)}}{\sum_{i=1}^{K} e^{(sim_{-\bullet,i}/T)}} \right) \tag{4.1}$$

where $weight_{+\bullet}$ is the edge weight between the positive node and the anchor node, $sim_{+\bullet}$ is the cosine similarity between the anchor and a positive node embedding, and $sim_{-\bullet,i}$ is the cosine similarity between the anchor node embedding and the i^{th} negative node embedding. T is a temperature term tuned over a validation set. Section 4.4.5 describes the loss function ablations.

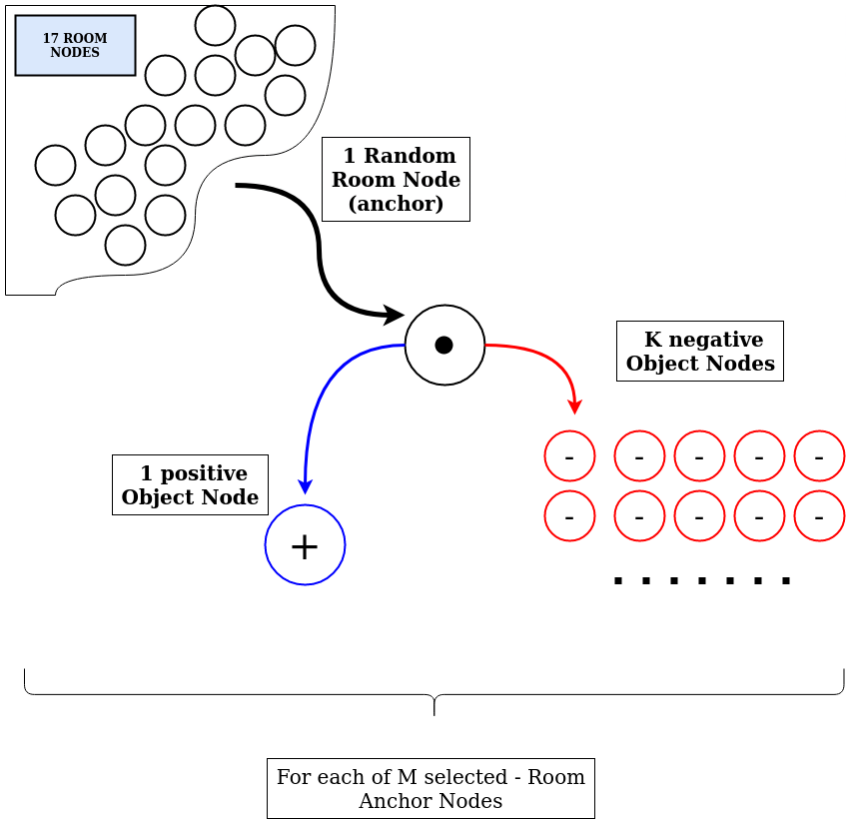


Figure 4.5: Sampling method used in the loss function; shown for $K = 10$ and $M = 1$; we average the loss over M batches.

We randomly select one of the 17 rooms as our anchor node, choose a positive node (for the numerator in Equation 4.1) by randomly selecting an object within that room, and sample k negative nodes for the denominator of the loss function from objects located outside the room; Figure 4.5 illustrates this process. The sampling method is repeated for a batch of samples, and the mean loss is calculated. This formulation of the loss function minimizes the distance between the anchor node and the positive node while maximizing the distance with each of the negative nodes, leading to distinct clusters in the graph embeddings. As mentioned previously, the training pipeline is outlined in Figure 4.3.

4.3.4 Testing

Once the GCN has been trained, the testing pipeline, illustrated in Figure 4.6, is employed for inference. Similar to the training process, we compute the CLIP image encoder embedding for the test image and the CLIP language encoder embedding for potential rooms. These embeddings are then fed into the GCN with only self-edges (in the absence of a knowledge graph) to obtain the output, which gives the latent space embedding for the test image and potential rooms.

Subsequently, similarity scores are computed between each image node $\vec{x}$ and each of the room(s) $\vec{y}$ using the cosine similarity function:

$$\cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{||\vec{x}|| \cdot ||\vec{y}||}$$

.

We then average the similarity scores over different images of each object category to get the affinity score between that object category and each of the candidate rooms.

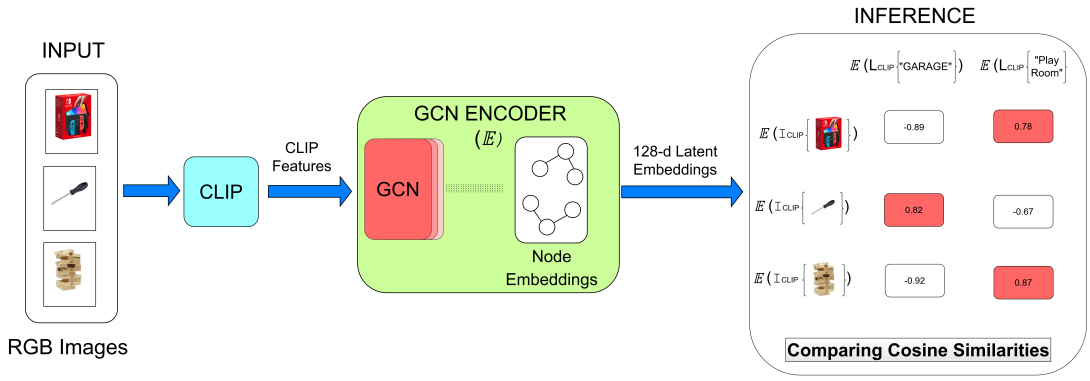


Figure 4.6: Our inference pipeline processes input RGB images to generate CLIP image embeddings. These embeddings are processed by the GCN Encoder to produce latent image embeddings. Cosine similarity between these latent embeddings and previously learned room embeddings determines object-room affinities.

4.4 Experimental Setup and Results

This section describes the experiments we conducted and discusses the corresponding results. We experimentally evaluate the following hypothesis:

H1: CLIP language embeddings result in better performance than other language encoder embeddings;

H2: Multimodal CLIP embeddings, by themselves, do not perform better than language-based embeddings;

H3: Our framework leads to better performance than (i) the underlying CLIP embedding, (ii) just the language-based encodings, and (iii) the GPT-3 LLM;

H4: Our framework provides robustness to previously unseen noisy backgrounds.

We evaluate **H1-H3** quantitatively and **H4** qualitatively. The performance task was to compute estimates of object-room affinities for all 268 object categories and 17 rooms in the test split of the IRONA dataset.

4.4.1 Experimental Setup

Object-room affinities have predominantly been determined by language-based embeddings or human input in prior work. Since our work combines prior knowledge and multimodal (vision, language) inputs, our chosen baselines were off-shelf language encoders and the GPT-3 LLM.

We considered two performance measures:

1. **mAP:** The mean average precision (mAP) is the average of precision scores at different recall values for each instance of an object category, and the mean over all the object categories.

2. **Top k Hit Ratio:** The average fraction of object categories for which the ground truth correct room was among the Top k estimates from our framework.

All claims are statistically significant unless stated otherwise.

4.4.2 Baseline

To obtain baseline results, we query **GPT-3** to rank the 17 rooms for each object like this:

> Which of the following rooms would you expect to find a **knife block** in? Please rank in decreasing order of likelihood: bathroom, bedroom, child room, closet, corridor, dining room, exercise room, garage, home office, kitchen, living room, lobby, pantry room, playroom, storage room, television room, utility room.

We use such queries to obtain room rankings for each of 268 objects and use that to obtain baseline mAP and hit ratio for GPT-3, and the results are shown in Table 4.3

	Test mAP ⇑	**Hit-Ratio** ⇑		
		Top-1	Top-3	Top-5
GPT-3	0.66	0.52	0.76	0.81

Table 4.3: Results for object-room mappings based on queries to GPT-3

The room rankings for each object category by querying GPT-3 is compiled in the file: gpt_pred.txt

Language Model	Test mAP ⇑	Hit-Ratio ⇑		
		Top-1	Top-3	Top-5
ConvNeXt	0.405	0.223	0.472	0.632
ViT	**0.456**	0.256	**0.576**	**0.710**
RN50	0.453	**0.275**	0.546	0.643
RoBerta	0.417	0.238	0.491	0.636
GloVE	0.148	0.123	0.208	0.278

Table 4.4: CLIP-based language embeddings perform better than other popular language encoders. These results support **H1**.

UnTuned-CLIP	Test mAP ⇑	Hit-Ratio ⇑		
		Top-1	Top-3	Top-5
ConvNeXt	0.41	0.24	0.46	0.62
ViT	**0.42**	**0.25**	**0.49**	**0.65**
RN50	0.39	0.19	0.45	0.67

Table 4.5: Multimodal CLIP embeddings, by themselves, do not improve performance compared with just the CLIP-based language embeddings (see Table 4.4). Results support **H2**.

4.4.3 Quantitative Results

To evaluate **H1**, we first compared two existing language encoder embeddings (RoBerta [80], GloVE [81]) with just the CLIP-based language embeddings with each of the three CLIP architectures. As shown in Table 4.4, the CLIP-based language embeddings (particularly the ViT architecture) resulted in better performance, supporting **H1**.

Next, we compared the performance of the multimodal (vision, language) CLIP embeddings for each of the three CLIP architectures. As shown in Table 4.5, performance is comparable but slightly worse than that in Table 4.4. These results support **H2** and motivate the use of GCNs.

Next, we computed the performance of our architecture, i.e., with GCNs trained using the contrastive loss function and the underlying multimodal CLIP embeddings, with the corresponding results shown in Table 4.6. The best performance was (once again) with the ViT version of the CLIP architecture. Also, performance was substantially better than with the multimodal CLIP embeddings (Table 4.5) or

GCN-CLIP	Test mAP ⇑	Hit-Ratio ⇑		
		Top-1	Top-3	Top-5
ConvNeXt	0.73	0.62	0.81	0.88
ViT	**0.85**	**0.76**	**0.93**	**0.97**
RN50	0.66	0.53	0.75	0.81

Table 4.6: CLIPGraphs' use of GCN embeddings of multimodal CLIP features and commonsense knowledge results in substantially better performance compared with just the CLIP embeddings in Tables 4.4 and 4.5. Results support **H3**.

CLIP's language encoder embeddings (Table 4.4). For example, there is an $\approx 40\%$ increase in mAP score compared with not using the GCNs. These results partially support **H3**.

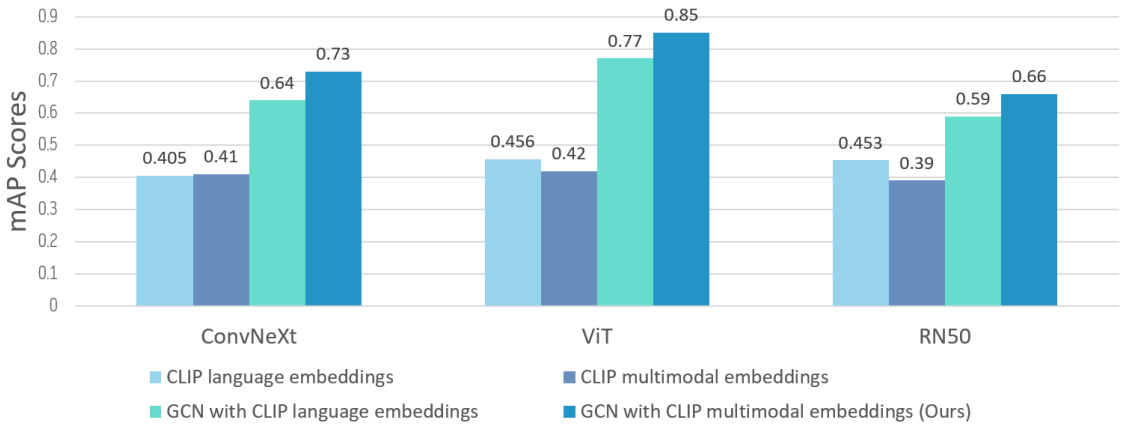


Figure 4.7: Graphical representation of mAP scores for CLIPGraphs and CLip multimodal and language-based embeddings. Summarized from Tables 4.4, 4.5, 4.6 and 4.7.

We conducted experiments with our framework to further explore the benefits of a multimodal CLIP representation but with GCN embeddings of only the language-based encoding of CLIP. The results reported in Table 4.7 show the benefits of using the multimodal CLIP embeddings, which demonstrates the effectiveness of our model in cross-modal commonsense reasoning. Further, Figure 4.7 shows a graphical representation comparing the mAP for CLIPGraphs and the underlying CLIP language-based and multimodal features.

The next experiment compared our framework's performance with the GPT-3 LLM and a state-of-the-art language encoder that provided the best performance among language-based encoders. The results summarized in Table 4.8 show that our framework provides substantially better performance

GCN-CLIP[Lang]	**Test mAP** ⇑	**Hit-Ratio** ⇑		
		Top-1	Top-3	Top-5
ConvNeXt	0.64	0.53	0.69	0.76
ViT	**0.77**	**0.68**	**0.77**	**0.83**
RN50	0.59	0.46	0.63	0.74

Table 4.7: Using our GCN-based embedding with just the underlying language-based CLIP encoding results in better performance than in the absence of the GCN embedding, but performance is not as good as when GCNs are used with the multimodal CLIP embeddings (in Table 4.6).

	Test mAP ⇑	**Hit-Ratio** ⇑		
		Top-1	Top-3	Top-5
Our [GCN-CLIP] [4.6]	**0.85**	**0.76**	**0.93**	**0.97**
GPT-3	0.66	0.52	0.76	0.81
Best untrained CLIP model [4.5]	0.42	0.25	0.49	0.65
Best Language Encoder [4.4]	0.456	0.275	0.576	0.71

Table 4.8: Our framework, with GCN and underlying multimodal CLIP embeddings, substantially improves performance compared with standalone GPT-3 LLM, just the underlying multimodal CLIP embeddings and language-based encoders; hence, the results strongly support **H3**.

by fully leveraging prior commonsense knowledge and multimodal CLIP embeddings. These results strongly support **H3**.

4.4.4 Qualitative Results

Our model was trained on clean white background images to predict the most appropriate room for the object in the image. To evaluate its performance on real-world images, we conducted tests on photographs captured on a mobile phone. These photographs were fed into our model. Additionally, we tested the model's ability to generalize by feeding it images of objects not present in the training set. We present both successful and unsuccessful cases to illustrate its performance.

Figure 4.8 shows the result of using our framework with images of 3 previously seen object categories but in noisy, previously unseen backgrounds to test our model's generalization capabilities on

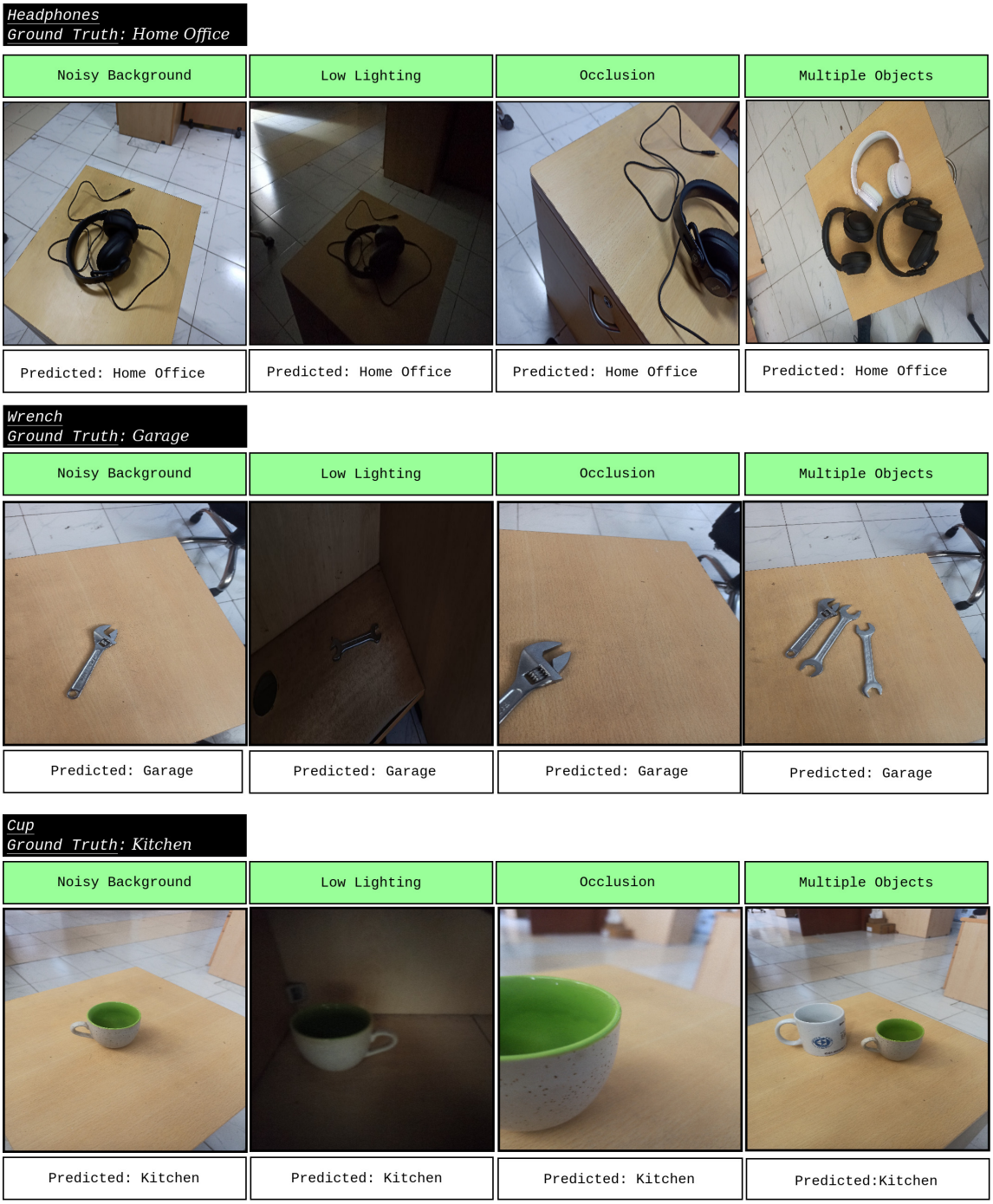


Figure 4.8: Qualitative result of using our framework with images of previously *seen* objects but in noisy backgrounds. In each case, the object's room association was estimated correctly, demonstrating the broad applicability of our method. These results support **H4**.

Figure 4.9: Successful placement of previously unseen object categories (absent in the training set) by leveraging commonsense domain knowledge. Since these categories were unseen, we didn't have any ground truth available. Thus, we mention the *expected* room on the basis of our commonsense.

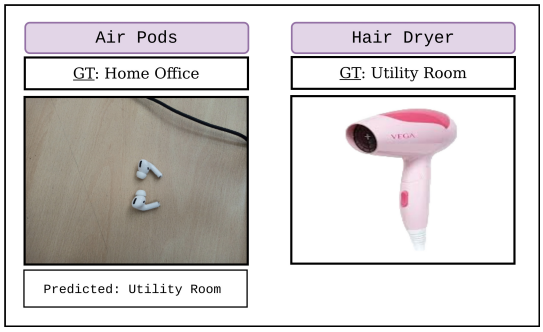


Figure 4.10: Failure to determine correct room for object category *earpods* (not in our train set) because it was structurally similar to *hair dryer* category that was in our training set.

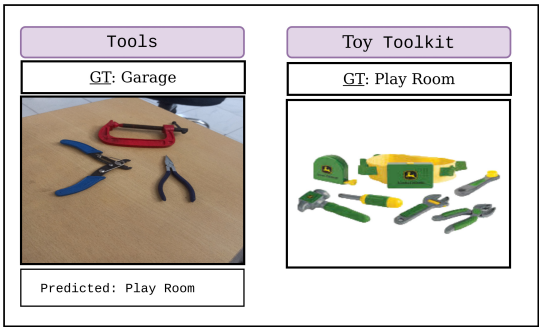


Figure 4.11: Failure with composite object categories; *tools* was not a category in our training set, but they were incorrectly associated with the *play room* because they were structurally similar to the *toy toolkit* that was in the training set.

real-world noisy images. For this experiment, we used 4 different scenarios - noisy background, low lighting, occlusion and multiple objects. In each case, the object's room association was estimated correctly. Next, Figure 4.9 shows the success cases when our trained framework was used with objects from previously unseen object categories. Success (i.e., estimating the correct room association for the objects) can be attributed to leveraging commonsense knowledge extracted from similar images.

Figures 4.10 and 4.11 show some examples of our framework's limitations. In Figure 4.10, an input image of earpods (not present in the training set) was mapped to the *utility room* because our model has no knowledge of scale, and the image of earpods was similar in appearance to hair dryers that was known to our framework. Figure 4.11 shows another failure case in which our framework estimated the room association for actual tools (which it has not seen before) as *playroom* because the training set

contained an image of a toy tool kit in a playroom. However, when considered individually, each tool is associated with the correct room location. These results support hypothesis **H4**.

4.4.5 Loss Function Ablations

We considered two performance measures:

1. **mAP:** The average precision is the average of precision scores at different recall values for each instance of an object category. For a given object, Average Precision is calculated by:

$$AP = \sum_{n} (R_n - R_{n-1}) P_n$$

   Where P_n and R_n are Precision and Recall values at the n^{th} threshold. Taking a mean over all the object categories gives us the final mean Average Precision (mAP).

2. **Top k Hit Ratio:** This represents the average proportion of object categories wherein the ground truth correct room was included among the Top k estimates produced by our framework. This measure can be computed as:

$$\text{Top-k HR} = \frac{1}{|O|} \sum_{o \in O} 1(R_o \cap T_o \neq \emptyset)$$

   where O is the set of objects, R_o is the set of top-k rooms recommended for object o, T_o is the ground truth room for object o, and 1 is the indicator function that returns 1 if the condition is true and 0 otherwise.

4.4.5.1 Comparison with existing loss functions

We compare the mAP of existing loss functions with the loss function we use as described in Equation 4.1 in the following table:

Loss Function	**mAP**
Margin [78]	0.371
Triplet [77]	0.51
Ours	**0.85**

4.4.5.2 Hyperparameters

We adopted a modified version of the loss function proposed by [61]. The efficacy of our loss function was contingent on the refinement of our sampling technique. We fine-tuned the following parameters:

1. **Batch Size**

   We conducted experiments varying the number of batches containing anchor, positive, and negative nodes sampled per epoch for loss computation. The figure illustrates that optimal performance is achieved with a batch size of 15. Consequently, we proceeded with further experiments using this value. The comparison is presented in Figure 4.12.

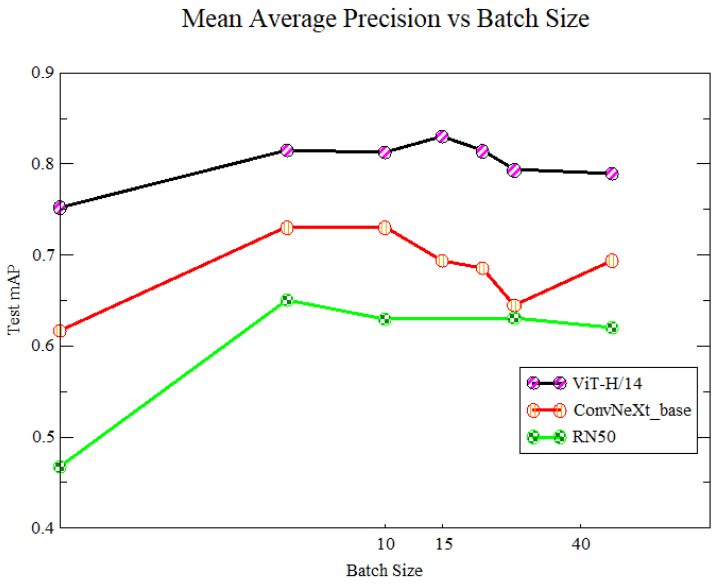


Figure 4.12: Variation of testing metrics with number of batches used for contrastive loss

2. **Number of Negative Nodes Sampled per Batch for Each Epoch**

   To assess the impact on model performance, we investigated variations in the number of negative nodes sampled per anchor point to compute the contrastive loss. The results are depicted in Figure 4.13. The figure shows that the highest performance was observed when the number of negative samples was set to 40. Therefore, we established a fixed total of 40 negatives per anchor for subsequent experiments.

4.4.6 Additional plots

Using t-SNE to visualize the high-dimensional embeddings, we observe initial random nodes in Figure 4.14, where each colour represents objects of a unique room. As the model trains, we observe clustering in the embedding space to cluster objects belonging to the same room. This is shown in Figure 4.15. We further zoom into the objects clustered in “Bathroom” in Figure 4.16 and “Pantry” in Figure 4.17.

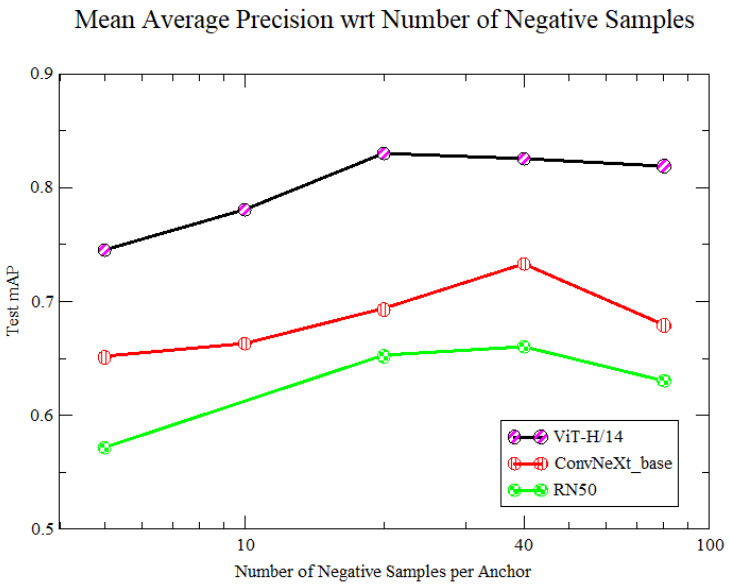


Figure 4.13: Variation of testing metrics with different numbers of negative samples used per anchor.

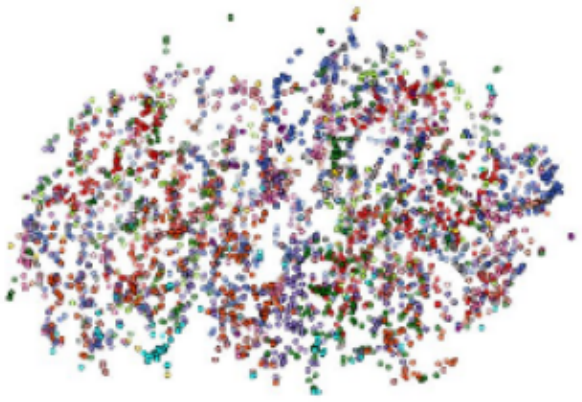


Figure 4.14: Untrained TSNE

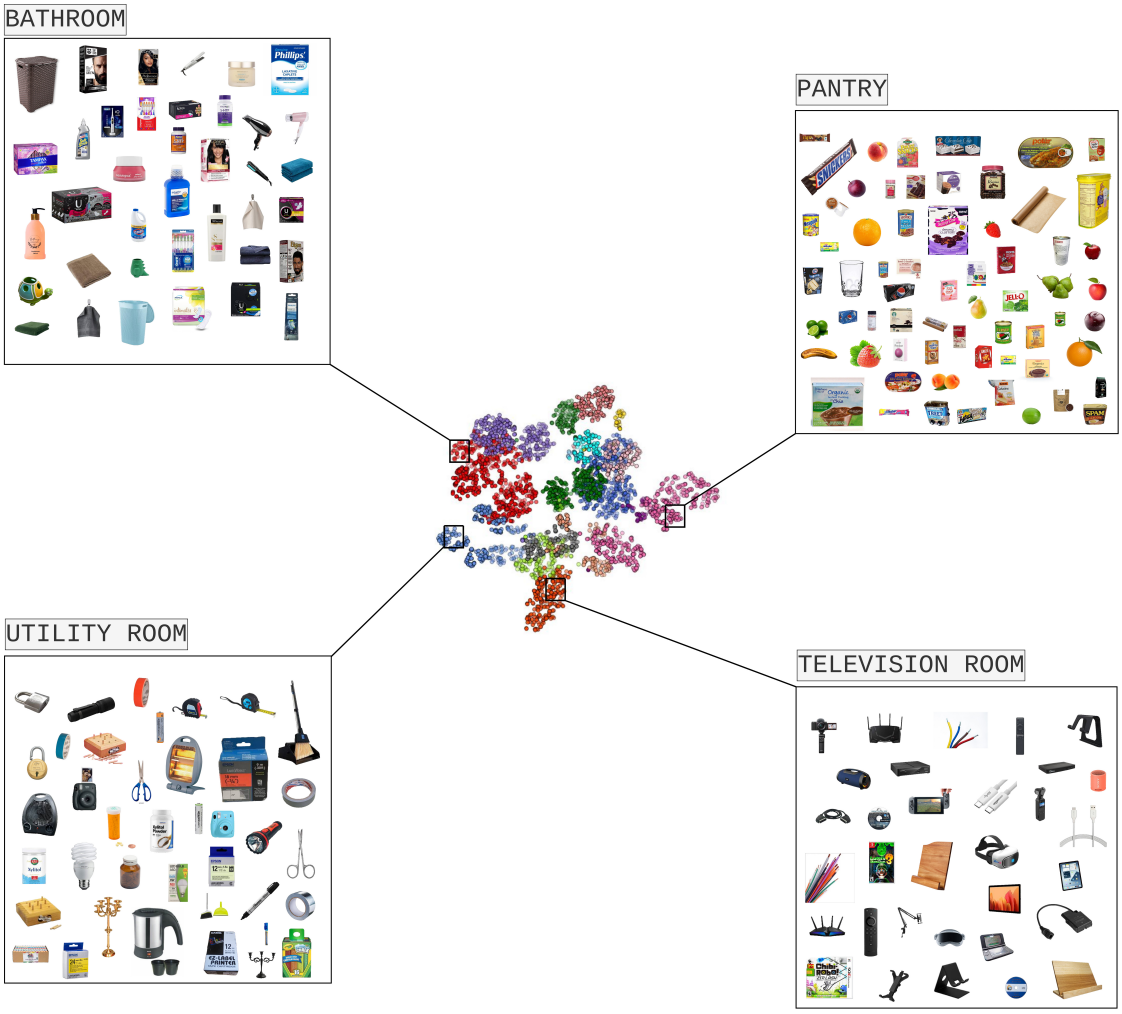


Figure 4.15: t-SNE visualization of our embeddings on the test split of the Web Scraped Dataset. The boxes show images of objects belonging to the same rooms getting clustered. For a more interactive view of this figure, check out our website: https://clipgraphs.github.io

.

Figure 4.16: Image showing objects that got clustered in the t-SNE corresponding to **Bathroom** room category

Figure 4.17: Image showing objects that got clustered in the t-SNE corresponding to **Pantry** room category

4.5 Summary

Understanding the relationship between objects and their corresponding rooms is pivotal for tasks like locating objects in a household (object navigation task) and organising a cluttered space (scene rearrangement). In this chapter, we introduced *CLIPGraphs*, a framework that harnesses the combined strengths of commonsense knowledge, data-driven techniques, and multimodal embeddings to accurately determine object-room affinities. We outlined a knowledge graph that encodes human preferences regarding the spatial arrangement of objects within domestic environments. We used joint image and text feature embeddings to support multimodal learning and queries in the form of image or text. Subsequently, we employed a graph network to learn object-room affinities based on latent embeddings extracted from the knowledge graph, encompassing image and text feature embeddings. Further, we introduced a new dataset, *IRONA*, comprising over 8000 image-text pairs sourced from the Web. We utilized this dataset to assess the efficacy of the CLIPGraphs framework in accurately predicting the optimal room placement for various objects. Through experimentation, we demonstrated that our framework significantly outperforms state-of-the-art baselines, including Large Language Models (LLMs) and language embeddings encoding common sense knowledge of object locations.

Chapter 5

Conclusion

This dissertation undertakes a comprehensive exploration of two critical challenges in the domain of embodied AI and intelligent household agents. In Chapter 3, we concentrate on addressing the intricate problem of locating multiple objects within a household environment. We described a deep reinforcement learning-based framework for Sequence-Agnostic Multi-object Navigation (SAM). Traditional methods for solving Multi-Object Navigation (MultiON) suffer from several limitations: (i) they consider only one instance of each object class in the environment; (ii) they typically work with artificial objects; and (iii) they are pre-sequenced (PSM), meaning the order of exploration for target object classes is predetermined. Our proposed SAM framework aims to overcome these limitations. It extends the prior deep (actor-critic) reinforcement learning framework for Object Navigation [4], and incorporates a tailored reward specification that encourages the desired sequence-agnostic operation. It enables the robot to build on prior experiences in environments similar to the target environment and concurrently (and greedily) reduce the distance to an instance of each target object class. The experimental evaluation, conducted within the Habitat 3D simulation environment using realistic scenes from the Gibson dataset, demonstrates a significant performance enhancement compared to various other methods tailored for the MultiON problem. These methods include a baseline that selects actions randomly, PSM, and an extension of the SemExp method for Object Navigation [19] adapted for Multi-Object Navigation.

Expanding on the MultiON problem, Chapter 4 advances the exploration into scene rearrangement, a logical progression once objects can be reliably located in a household. In this chapter, we introduce CLIPGraphs, a novel method designed to accurately determine object-room affinities—a fundamental prerequisite for scene rearrangement tasks. CLIPGraphs leverages a combination of commonsense knowledge, data-driven methods, and multimodal embeddings (vision, language) to estimate these affinities. The framework encodes prior human preferences in a knowledge graph and considers CLIP-based image and language embeddings of nodes in this graph. It utilizes Graph Convolutional Network (GCN)-based embeddings of the CLIP embeddings to learn and estimate the object-room affinities. Experimental evaluation, conducted on our IRONA dataset comprising 8040 images covering 268 benchmark object categories, showcases the superior performance of CLIPGraphs in estimating object-room

affinities compared with language encoder embeddings and the GPT-3 LLM. Additionally, we showed qualitatively that the framework exhibits robustness in handling previously unseen noisy backgrounds in object images.

In future endeavours, we aim to expand the application of the MutliON framework to more complex environments and a broader range of target object classes. Additionally, we plan to explore the integration of clustering techniques as the number of object classes increases. While our current experiments have focused on household scenes, we anticipate the relevance of this project in other settings such as supermarkets or hospitals. We plan to explore the practical implementation of such frameworks on physical robots, particularly in real-world applications such as Telepresence [82].

The CLIPGraphs framework also opens avenues for further research. We plan to train our model with top-k correct rooms to generate object-room affinities that would be useful in downstream tasks such as multi-object navigation and scene rearrangement. This approach mirrors human behaviour, leveraging semantic and utility-based organization principles observed in how we arrange our living spaces. Additionally, we aspire to develop personalized or task-specific embeddings, enabling the framework to calculate object-room affinities tailored to individual users, homes, or tasks. This personalized approach empowers physical robots to tackle complex scene rearrangement tasks and other embodied AI scenarios characterized by semantic organization. By leveraging the ability of Large Language Models (LLMs) to incorporate commonsense reasoning derived from extensive and diverse internet datasets, we aim to fine-tune them for personalized object-room affinities. This involves fine-tuning an LLM with house arrangement patterns and expecting it to identify patterns that can help personalize the object-room affinities.

The approaches elucidated in this dissertation lay a robust foundation for addressing embodied AI challenges with increasing intelligence as AI technologies progress. We anticipate that this work will not only contribute to the scholarly discourse but also serve as a catalyst for further research in the burgeoning field of intelligent household agents.

Related Publications

1. Nandiraju Gireesh*, Ayush Agrawal*, **Ahana Datta***, Snehasis Banerjee, Mohan Sridharan, B. Bhowmick and M. Krishna. "Sequence-Agnostic Multi-Object Navigation." *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2023): 9573-9579.

2. Ayush Agrawal*, Raghav Arora*, **Ahana Datta**, Snehasis Banerjee, B. Bhowmick, Krishna Murthy Jatavallabhula, Mohan Sridharan and M. Krishna. "CLIPGraphs: Multimodal Graph Networks to Infer Object-Room Affinities." *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)* (2023): 2604-2609.

Other Publications

1. Raghav Arora*, Shivam Singh*, Karthik Swaminathan, **Ahana Datta**, Snehasis Banerjee, Brojeshwar Bhowmick, Krishna Murthy Jatavallabhula, Mohan Sridharan and Madhava Krishna. "Anticipate & Act : Integrating LLMs and Classical Planning for Efficient Task Execution in Household Environments." *2024 IEEE International Conference on Robotics and Automation (ICRA)* (2024): 14038-14045. (Not used in this thesis.)

Bibliography

[1] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classifcation in network data. 2008.

[2] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[3] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5664–5673, 2019.

[4] Nandiraju Gireesh, D. A. Sasi Kiran, Snehasis Banerjee, Mohan Sridharan, B. Bhowmick, and M. Krishna. Object goal navigation using data regularized q-learning. *International Conference on Automation Science and Engineering*, pages 1092–1097, 2022.

[5] Saim Wani, Shivansh Patel, Unnat Jain, Angel Chang, and Manolis Savva. Multion: Benchmarking semantic map memory using multi-object navigation. *Advances in Neural Information Processing Systems*, 33:9700–9712, 2020.

[6] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *International Conference on Robotics and Automation*, May 2017.

[7] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Roshan Zamir. On evaluation of embodied navigation agents. *ArXiv*, abs/1807.06757, 2018.

[8] Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. Housekeep: Tidying virtual households using commonsense reasoning. In *European Conference on Computer Vision*, 2022.

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

[10] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. *ArXiv*, abs/2212.07143, 2022.

[11] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[12] Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson ENV: Real-world Perception for Embodied Agents. In *International Conference on Computer Vision and Pattern Recognition*, 2018.

[13] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-Matterport 3D Dataset (HM3D): 1000 Large-scale 3D Environments for Embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[14] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9338–9346, 2019.

[15] Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. Auxiliary Tasks Speed Up Learning Point Goal Navigation. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *International Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 498–516, 16–18 Nov 2021.

[16] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations*, 2019.

[17] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv*, 2006.13171, 2020.

[18] Snehasis Banerjee, Brojeshwar Bhowmick, and Ruddra Dev Roychoudhury. Object goal navigation based on semantics and rgb ego view. *arXiv preprint arXiv:2210.11543*, 2022.

[19] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Neural Information Processing Systems*, 2020.

[20] Sang-Min Park and Young-Gab Kim. Visual language navigation: A survey and open challenges. *Artificial Intelligence Review*, 56(1):365–427, 2023.

[21] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations*, Apr 26-May 1, 2020.

[22] Devendra Singh Chaplot, Helen Jiang, Saurabh Gupta, and Abhinav Kumar Gupta. Semantic Curiosity for Active Visual Learning. In *European Conference on Computer Vision*, 2020.

[23] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2135–213509, 2017.

[24] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. IQA: Visual Question Answering in Interactive Environments. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4089–4098, 2017.

[25] Cătălina Cangea, Eugene Belilovsky, Pietro Lio', and Aaron C. Courville. VideoNavQA: Bridging the Gap between Visual and Embodied Question Answering. In *British Machine Vision Conference*, 2019.

[26] Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A Challenge for Embodied AI, 2020.

[27] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5918–5927, 2021.

[28] Brandon Trabucco, Gunnar A Sigurdsson, Robinson Piramuthu, Gaurav S. Sukhatme, and Ruslan Salakhutdinov. A Simple Approach for Visual Room Rearrangement: 3D Mapping and Semantic Search. In *The Eleventh International Conference on Learning Representations*, 2023.

[29] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10737–10746, 2019.

[30] Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, P. Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramithu, Gokhan Tur, and Dilek Z. Hakkani-Tür. Teach: Task-driven embodied agents that chat. In *AAAI Conference on Artificial Intelligence*, 2021.

[31] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2017.

[32] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Kumar Gupta. Iterative visual reasoning beyond convolutions. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7239–7248, 2018.

[33] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Kumar Gupta. The more you know: Using knowledge graphs for image classification. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20–28, 2016.

[34] X. Wang, Yufei Ye, and Abhinav Kumar Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6857–6866, 2018.

[35] Arsalan Mousavian, Alexander Toshev, Marek Fiser, Jana Kosecka, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. *arXiv*, 1805.06066, 2019.

[36] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *International Conference on Computer Vision and Pattern Recognition*, 2017.

[37] Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18890–18900, 2022.

[38] Junho Kim, Eun Sun Lee, Mingi Lee, Donsu Zhang, and Young Min Kim. SGoLAM: Simultaneous Goal Localization and Mapping for Multi-Object Goal Navigation. *arXiv preprint arXiv:2110.07171*, 2021.

[39] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Teaching agents how to map: Spatial reasoning for multi-object navigation. *arXiv preprint arXiv:2107.06011*, 2021.

[40] Gabriel Sarch, Zhaoyuan Fang, Adam W. Harley, Paul Schydlo, Michael J. Tarr, Saurabh Gupta, and Katerina Fragkiadaki. TIDEE: Tidying Up Novel Rooms using Visuo-Semantic Commonsense Priors. In *European Conference on Computer Vision*, 2022.

[41] Samir Yitzhak Gadre, Kiana Ehsani, Shuran Song, and Roozbeh Mottaghi. Continuous Scene Representations for Embodied AI. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14829–14839, 2022.

[42] Kai Zheng, KAI-QING Zhou, Jing Gu, Yue Fan, Jialu Wang, Zong xiao Li, Xuehai He, and Xin Eric Wang. JARVIS: A Neuro-Symbolic Commonsense Reasoning Framework for Conversational Embodied Agents. *ArXiv*, abs/2208.13266, 2022.

[43] Yichi Zhang, Jianing Yang, Jiayi Pan, Shane Storks, Nikhil Devraj, Ziqiao Ma, K. Yu, Yuwei Bao, and Joyce Yue Chai. DANLI: Deliberative Agent for Following Natural Language Instructions. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

[44] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models, 2023.

[45] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. *ArXiv*, abs/1911.11641, 2019.

[46] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but Effective: CLIP Embeddings for Embodied AI, 2022.

[47] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How Much Can CLIP Benefit Vision-and-Language Tasks? *ArXiv*, abs/2107.06383, 2021.

[48] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, Ayush Tewari, Joshua B. Tenenbaum, Celso Miguel de Melo, Madhava Krishna, Liam Paull, Florian Shkurti, and Antonio Torralba. ConceptFusion: Open-set Multimodal 3D Mapping. *Robotics: Science and Systems (RSS)*, 2023.

[49] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Semantically grounded object matching for robust robotic scene rearrangement. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11138–11144, 2022.

[50] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *ArXiv*, abs/2206.12403, 2022.

[51] Vishnu Sashank Dorbala, Gunnar A. Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S. Sukhatme. CLIP-Nav: Using CLIP for Zero-Shot Vision-and-Language Navigation. *ArXiv*, abs/2211.16649, 2022.

[52] Vishnu Sashank Dorbala, James F. Mullen, and Dinesh Manocha. Can an Embodied Agent Find Your "Cat-shaped Mug"? LLM-Based Zero-Shot Object Navigation. *ArXiv*, abs/2303.03480, 2023.

[53] Chen Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual Language Maps for Robot Navigation. *ArXiv*, abs/2210.05714, 2022.

[54] Dhruv Shah, Blazej Osinski, Brian Ichter, and Sergey Levine. LM-Nav: Robotic Navigation with Large Pre-Trained Models of Language, Vision, and Action. In *Conference on Robot Learning*, 2022.

[55] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *ArXiv*, abs/2203.10421, 2022.

[56] Boyuan Chen, F. Xia, Brian Ichter, Kanishka Rao, Keerthana Gopalakrishnan, Michael S. Ryoo, Austin Stone, and Daniel Kappler. Open-vocabulary Queryable Scene Representations for Real World Planning. *ArXiv*, abs/2209.09874, 2022.

[57] Kanishk Jain, Varun Chhangani, Amogh Tiwari, K. Madhava Krishna, and Vineet Gandhi. Ground then navigate: Language-guided navigation in dynamic scenes, 2022.

[58] Jesse Thomason, Mohit Shridhar, Yonatan Bisk, Chris Paxton, and Luke Zettlemoyer. Language Grounding with 3D Objects. In *5th Annual Conference on Robot Learning*, 2021.

[59] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and Where Pathways for Robotic Manipulation. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[60] Huy Ha and Shuran Song. Semantic Abstraction: Open-World 3D Scene Understanding from 2D Vision-Language Models. In *Conference on Robot Learning*, 2022.

[61] Nur Muhammad Mahi Shafiullah, Chris Paxton, Lerrel Pinto, Soumith Chintala, and Arthur Szlam. CLIP-fields: Weakly supervised semantic fields for robotic memory. In *Workshop on Language and Robotics at CoRL 2022*, 2022.

[62] Nandiraju Gireesh, Ayush Agrawal, Ahana Datta, Snehasis Banerjee, Mohan Sridharan, Brojeshwar Bhowmick, and Madhava Krishna. Sequence-agnostic multi-object navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9573–9579, 2023.

[63] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision*, Oct 2017.

[64] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021.

[65] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv*, 1607.06450, 2016.

[66] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *International Conference on Learning Representations*, 2014.

[67] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2094–2100. AAAI Press, 2016.

[68] J A Sethian. A fast marching level set method for monotonically advancing fronts. *National Academy of Sciences*, 93(4):1591–1595, 1996.

[69] Kirsty Ellis, Denis Hadjivelichkov, Valerio Modugno, Danail Stoyanov, and D. Kanoulas. Navigation among movable obstacles via multi-object pushing into storage zones. *IEEE Access*, 11:3174–3183, 2023.

[70] Pierre Marza, Laëtitia Matignon, Olivier Simonin, and Christian Wolf. Teaching agents how to map: Spatial reasoning for multi-object navigation. *International Conference on Intelligent Robots and Systems*, pages 1725–1732, 2021.

[71] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[72] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022.

[73] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

[74] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, Los Alamitos, CA, USA, June 2022. IEEE Computer Society.

[75] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs, 2021.

[76] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. OpenCLIP, 2021.

[77] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141, 2017.

[78] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.

[79] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

[80] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[81] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[82] Abhijan Bhattacharyya, Ashis Sau, Ruddra Dev Roychoudhury, Snehasis Banerjee, Chayan Sarkar, Pradip Pramanick, Madhurima Ganguly, Brojeshwar Bhowmick, and Balamuralidhar Purushothaman. Teledrive: An intelligent telepresence solution for "collaborative multi-presence" through a telerobot. In *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 433–435. IEEE, 2022.