

Understanding Language Referring To The Visual World

Volkan Cirik

CMU-LTI-22-003

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Louis-Philippe Morency (Co-Chair)	Carnegie Mellon University
Taylor Berg-Kirkpatrick (Co-Chair)	UC San Diego
Yonatan Bisk	Carnegie Mellon University
Jason Baldridge	Google Research

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies*

Copyright © 2022 Volkan Cirik

Keywords: grounded language learning, grounded semantics, vision-and-language

I dedicate this thesis to the people who have supported me throughout my education.

Abstract

Artificial Intelligence (AI) technologies affect many facets of our daily lives. AI systems help us manage our shopping lists, type emails faster, and answer our curious search queries. However, current AI systems have limited agency in the world – i.e., they lack the embodied sensory experience of the world that is often referred as embodied AI. Hopefully, in the near future, embodied AI systems will allow autonomous vehicles to mobilize the visually impaired in our communities, enable robots in providing company for our elderly, and facilitate virtual agents in cooperatively teaching our children complex concepts in mixed reality settings. The imminent manifestation of embodied AI in the physical world necessitates research that models the interaction between natural language and physical referents. This technical challenge has been dubbed ‘language grounding’ and is the central focus of this thesis.

In studying language grounding, we identify and define three core challenges, and then describe novel methods, analyses, and experiments that to attempt to address each in turn. First, we address spatial grounding with the goal of linking language mentions of objects with their spatial locations in the world. We study this problem in the context of a fully-observable representation of the world. Second, we study the problem of sequential grounding, where observations of the world are partial (e.g., restricted to a limited field-of-view) and unfold over time as a result of the actions of the system. Partial observation makes language grounding more challenging, increasing the difficulty of accurate interpretation – e.g., an utterance may refer to something not currently in the view of the system. Third, we tackle the problem of imbuing agents with the same types of prior knowledge that humans assume and rely upon to disambiguate linguistic utterances when communicating. Human speakers tend to vastly underspecify spatial information when communicating with others as they omit many details that they expect the listener to know already. This poses a technical challenge for language grounding: how do our agents leverage general and situated prior knowledge of the world? In this thesis, we present contributions in the form of methods, resources, and tools to strengthen our understanding of these technical challenges and to make progress towards their eventual solutions.

Acknowledgments

This thesis would not exist without the help and effort of many people. First and foremost, I thank LP. He spent countless hours on my training. I learned a lot about paying attention to details, reframing concepts, and seeing the bigger picture. He did all these effortlessly and patiently. The more impressive of all is that he ensured we all were having fun along the way. I thank Taylor for his contagious curiosity, excitement, and kindness. He is one of those rare people who deeply understand many complex ideas and make them accessible for everyone. I am proud of being your mentee and friend. I thank my committee members, Jason and Yonatan. Jason was always available, fully supportive, and insightful. He is the perfect mentor. Yonatan is an exceptional human being – super transparent and admirably emphatic. He helped a lot with big decisions in my career. He is an exemplary member of our community; our community is and will be better because of him. I also thank my previous mentors: Deniz, Teruko, and Ed. Because of their bet on me before my PhD, this thesis became possible.

I want to thank heroes and heroines behind the scene to ensure the big research machine in CMU runs smoothly: Corey, John, Julie, Kate, Krista, Laura, MaryJo, Taylor, Tessa and many more. I want to thank Stacey, Nicki, Bob, and Daren especially. Stacey was always patient and helpful and, in many cases, went above and beyond to support students. All students were immensely indebted to her. Nicki was always generous with her help and always lightened the mood with her smile. Bob's door was literally and figuratively always open – I truly appreciate your help for Dec5 socializing SCS students. Without Daren's help, life would be much harder for me when I fought battles on many fronts.

I want to thank all my peers: Adhi, Alexandria, Alok, Amir, Bhuwan, Carla, Chirag, Cindy, Danish, Dheeraj, Dongyeop, Guillaume, Harsh, Hemank, Hubert, Jeff, John, Juneki, Junxian, Korte, Kartik, Maria, Michael, Nico, Nikita, Nikolai, Paul, Prasanna, Shane, Shrimai, Sujay, Swabha, Tadas, Torsten, Victoria, Vidisha, Waleed, Ying, Zhun and many more. They were all brilliant in unique ways, and I learned many things by being around them. I specifically thank Chaitanya for his comradeship, kindness, many fun chess matches, and Daniel for creating a chill vibe on all possible occasions. I especially thank him for a perfect surfing trip in the middle of a pandemic.

I want to thank my support network in Pittsburgh. Life in Pittsburgh was fun, fulfilling, and easier because of you: Burcu B, Burcu K, Cihan, Deniz, Derya, Erhan, Gizem, Ilana, Irem, Marina, Matt, Mert A, Mert T, Onur, Reggie, Sercan, Yigit, my little friends Mia, Libby, and Robin. I especially thank Gunay for his continued support and his generous soul, Meric for being the source of fun and color, and Ozgun & Baris for being there for me no matter what. They were great role models for me – I am a better person because of these people.

I want to thank my support network in the rest of the world: Ali, Ali G, Avni, Baris, Basar, Batu, Duygu, Ecem, Emir, Gamze, Gizem, Irem, Ismail, Melih, Mert, Murat, Mustafa, Onur, Riza, Seval, Sinan. I missed everyone a lot, but I will be much closer soon.

I want to thank Tina for being the source of joy, excitement, support, peace, and love in my life. I admire her for many things, but her authenticity and resilience are on another level. I was able to run the last mile because of her exemplary *sisu*. Kiitos, kulta!

I thank my family for creating a safe space for pursuing my passion. It was a long journey, and we went through lots of ups and downs together. Thanks for everything. I love you all!

Contents

1	Introduction	1
1.1	Technical Challenges	3
1.2	Thesis Contributions	5
2	The Role of Language in Language Grounding	7
2.1	Overview	8
2.2	Analysis by Perturbation	9
2.2.1	Analysis Methodology	9
2.2.2	Syntactic Analysis by Permuting Word Order	9
2.2.3	Lexical Analysis by Discarding Words	10
2.2.4	Bias Analysis by Discarding Referring Expressions	10
2.2.5	Discussion	11
2.3	Neural Sieves	11
2.3.1	Filtering Experiments	12
2.3.2	Results	14
2.4	Related Work	14
2.5	Conclusion	14
3	Modeling Cross-Object Relationships	15
3.1	Introduction	16
3.2	GroundNet	16
3.2.1	Motivation	17
3.2.2	Generating a Computation Graph	18
3.2.3	Neural Modules	19
3.3	Experiments	22
3.4	Results	24
3.5	Related Work	25
3.6	Conclusion	25
4	A Novel Benchmark For Referring Expression Recognition in 360° Images	29
4.1	Overview	30
4.2	Motivation	32
4.3	Refer360° Dataset	33
4.3.1	Annotation Procedure	33

4.3.2	Dataset Statistics	35
4.4	Analyses	37
4.4.1	Target Locations	37
4.4.2	Ablation of Instruction Sentences	39
4.4.3	Linguistic Phenomena Observed	39
4.4.4	Localization Experiments	40
4.5	Related Work	41
4.6	Conclusion	41
5	Reasoning About Alternatives for Vision-and-Language Navigation	45
5.1	Overview	46
5.2	Instruction Following with Speaker-Follower Models	47
5.2.1	Speaker-Driven Data Augmentation	47
5.2.2	Speaker-Driven Route Selection	48
5.2.3	Panoramic Action Space	49
5.3	Experiments	51
5.3.1	Experimental Setup	51
5.3.2	Results and Analysis	52
5.4	Related Work	55
5.5	Conclusion	57
6	General Knowledge of Objects for Referring Expression Recognition in Partially-Observed Scenes	69
6.1	Overview	70
6.2	Dynamic Referring Expression Recognition (dRER) Task	70
6.3	HOLM	72
6.3.1	Affinity Scores from Language Models	73
6.3.2	Object Hallucination	73
6.4	Experiments	74
6.4.1	Experimental Setup	75
6.4.2	Results and Discussion	76
6.5	Related Work	79
6.6	Conclusion	81
7	Situated Knowledge for Remote Embodied Visual Referring Expression	83
7.1	Overview	84
7.2	OSMaN: Object-centric Situated Map Navigator	85
7.2.1	FIND GOAL	86
7.2.2	SCORE OBJECTS	87
7.2.3	NAVIGATE&PREDICT	88
7.2.4	OSMaN-Exhaustive Acquiring Situated Knowledge with Exhaustive Search	89
7.3	Experiments	89
7.3.1	Experimental setup	90
7.3.2	Results and Discussion	91

7.4	Related Work	94
7.5	Conclusion	95
8	Conclusion and Discussion	97
8.1	Thesis Contributions	97
8.2	Broader Impact and Limitations	98
8.3	Future Research Directions	99
	Bibliography	101

List of Figures

- 1.1 An illustration for a complex task for an embodied AI system given four instructions. Colored dashed lines show desired trajectory for the robot starting position (0) for the instruction in the same color. Colored numbers in parentheses represent the expected endpoint for the instruction in the same color. 2
- 2.1 Overview of Neural Sieves. Sieve I filters object types having multiple instances. Sieve II filters objects of one category mentioned in referring expression. Objects of the same category have the same color frames. Best seen in color. 8
- 3.1 An Overview of GroundNet. An referring expression (a) is first parsed (b). Then, the computation graph of neural modules is generated using the parse tree (c). Each node localizes objects present in the image (d). 17
- 3.2 Illustrations of GroundNet’s neural modules. Upper left shows an example referring expression and the input for `Relate` node (upper right, highlighted in red) of a small section of a computation tree. Modules take inputs from module’s text span T , the set of bounding boxes R , and output probabilities of other nodes p_i . Best seen in color. 20
- 3.3 Qualitative Results for GroundNet. Bounding boxes and referring expressions to target object (in green boxes) on the left. GroundNet predictions in the middle and CMN predictions are on the right. GroundNet localizes not only the target object but also supporting objects (e.g. disc and girl in the first row, bench in the second). 27
- 4.1 An example from Refer360° . Orange frames represent the field-of-view (FoV) of the follower after interpreting each instruction. Numbers in the frames represent the sequential order. Green lines show how FoVs change continuously. After each instruction, the follower changes the FoV to align with what the instruction describes. Please see Figure 4.2a to see the correct location of Waldo. 30
- 4.2 Examples are from (a) Refer360° (b) Touchdown-SDR, and (c) Google-Ref datasets. In Refer360° , the target location could be *any random location* on the image. In (b), annotators chose an existing object as the target location. In (c), boxes for objects were used as targets. Refer360° also seeks to increase the complexity of instruction following, making it more realistic by introducing a partial and dynamic FoV rather than providing a holistic oracle-like view of the image. 32

4.3	Screenshot of Amazon Mechanical Turk interface for finding task. We ask annotators to complete each instruction before moving to the next one. To do so change the bullseye where they think the instruction is describing.	43
4.4	Screenshot of Amazon Mechanical Turk interface for describing task. We ask annotators to first find Waldo themselves, then give detailed instructions one by one so that anyone starting from a random field-of-view find it.	43
4.5	Distribution of the number of tokens for vision-and-language datasets similar to Refer360°	44
4.6	Text length for different placement methods for single instruction and instruction sequences. Manual means annotators pick the target location, random means we randomly pick the target location in the scene.	44
5.1	The task of vision-and-language navigation is to perform a sequence of actions (navigate through the environment) according to human natural language instructions. Our approach consists of an instruction <i>follower</i> model (left) and a <i>speaker</i> model (right).	47
5.2	Our approach combines an instruction <i>follower</i> model and a <i>speaker</i> model. (a) The speaker model is trained on the ground-truth routes with human-generated descriptions; (b) it provides the follower with additional synthetic instruction data to bootstrap training; (c) it also helps the follower interpret ambiguous instructions and choose the best route during inference. See Sec. 5.2 for details.	48
5.3	Compared with low-level visuomotor space, our panoramic action space (Sec. 5.2.3) allows the agents to have a complete perception of the scene, and to directly perform high-level actions.	50
5.4	The success rate of our model using different numbers K of route candidates (generated by state-factored search) for pragmatic inference. Stars show the performance of greedy inference (without search, and hence without pragmatics). While performance increases with number of candidates up through 40 on val unseen, the success rate tends to saturate. We note improvements both from the state-factored search procedure (comparing the stars to the circle and triangle points at $K = 1$) as well as from having more candidates to choose from in pragmatic inference (comparing larger values of K to smaller).	54
5.5	Follower without pragmatic inference on val seen. The instruction involves an ambiguous “walk a bit” command. Without pragmatic reasoning by the speaker, the follower failed to predict how much to move forward, stopping at a wrong location without entering the door.	58
5.6	Follower with pragmatic inference on val seen. With the help of the speaker, the follower could disambiguate “walk a bit” to move the right amount to the correct location. It then turned right and walked into the door to stop by the “zebra striped rug”.	59
5.7	Follower without pragmatic inference on val unseen. The command “walk into bedroom” is ambiguous since there are two bedrooms (one on the left and one on the right). The follower could not decide which bedroom to enter, but went into the wrong room with no “table clock”.	60

5.8	Follower with pragmatic inference on val unseen. The speaker model helps resolve the ambiguous “walk into bedroom” command (there are two bedrooms), allowing the follower to enter the correct bedroom on the right, where it could see a “table clock”.	61
5.9	Follower without pragmatic inference on val unseen. Although making a right turn as described, the follower fails to turn right at the correct location, and stopped at the door instead of the mirror. The route taken by the follower would be better described as “... <i>wait near the door</i> ” by a human, which the speaker could learn to capture.	62
5.10	Follower with pragmatic inference on val unseen. Using the speaker to measure how likely a route matches the provided description, the follower made the right turn at the correct location “ <i>the end of the rug</i> ”, and stopped near the mirror. . . .	63
5.11	Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.	64
5.12	Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.	65
5.13	Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.	66
5.14	Navigation examples on unseen environments with and without pragmatic inference from the speaker model (<i>best visualized in color</i>). (a) The follower without pragmatic inference misinterpreted the instruction and went through a wrong door into a room with no bed. It then stopped at a table (which resembles a bed). (b) With the help of a speaker for pragmatic inference, the follower selected the correct route that enters the right door and stopped at the bed.	67
6.1	Illustration of our main contribution: Hallucinating Objects . Knowledge about object relationships is helpful when navigating in an unknown and partially observed environment. In the example above, the TV is not visible, but the couch hints that a TV might be in front of it because usually couches face TVs.	70
6.2	Illustration of the dRER task with an example of language instruction and its recognition in four steps . The agent adjusts its FoV by looking at different directions and navigate on the graph in the spherical view. Note that objects mentioned in bold in the instruction are not visible at all until timestep 4. Thus, the agent needs to reason about possible locations of the mentioned object using its partial view of the scene.	71

6.3	HOLM for the dRER task. (Top) We use language models trained on a large amount of text by prompting with the spatial relationship of objects to calculate co-occurrence statistics of objects. (Bottom) The flow of our hallucination method. We determine objects of interest for each action. Then, we combine objects of interest and co-occurrence table to hallucinate objects, i.e. what might appear after performing an action.	72
7.1	(Left) Situated knowledge in the form of object detections at each location in the environment and the connection between these locations. (Right) Steps OSMaN takes for remote object localization. OSMaN extracts the goal object, scores each candidate object with textual and semantic similarity, chooses the highest-scoring object and location and navigates to the object’s location using known connections, i.e., the environment map.	86
7.2	Dependency parse of an instruction. In FIND GOAL, we use the first noun phrase of the root as the goal object.	87
7.3	OSMaN-Exhaustive’s process of situating itself in the environment. At each location, it detects and records objects. Then explores other locations and keeps track of adjacency information between successive locations. At the end of this process OSMaN-Exhaustive knows what objects are visible and where, and knows how to navigate from any start location to any target location.	88
7.4	Comparison of CLS (i.e., coverage of ground-truth path) scores for OSMaN and RBERT[86]. Purple circles represent a datum in validation unseen split. Numbers at each quadrant show the percentage of the data. The lower-left quadrant in purple square shows REVERIE-DeLiRE instances for which RBERT and OSMaN perform poorly.	94

List of Tables

2.1	Results for Shuffling Word Order for Referring Expressions. Δ is the difference between no perturbation and shuffled version of the same system.	9
2.2	Results with discarded word categories. Numbers in parentheses are Δ , the difference between the best performing version of the original model.	10
2.3	Results with discarded referring expressions. Surprisingly, the top-2 prediction (73.1%) of the “image-only” model is better than the top prediction of the state-of-the-art (70.5%).	11
2.4	Precision@ k accuracy for Neural Sieves and state-of-the-art systems. Note that even without using the referring expression, Sieve I is able to reduce the number of candidate boxes to 3 for 86.6% of the instances. When we further predict the type of objects with Sieve II, the number of candidate boxes is reduced to 2 for 84.2% of the instances.	13
3.1	The accuracy of models with the support of syntax, dynamic computation, modularity, relationship modeling, and supporting object predictions. Our model is the first syntax-based model with successful results and achieves the best results in supporting object localization.	23
4.1	Comparison of referring expression datasets, including our proposed Refer360° dataset. Refer360° poses a more challenging scenario where the system observes only a partial and dynamic FoV. Refer360° also includes explicit alignments between intermediate instruction steps and human follower actions which can be used as an auxiliary evaluation metric or source of supervision.	31
4.2	Statistics for data collection stages. Stage I is for hiring annotators. Stage II is for collecting and verifying the instructions. Last stage is further verifying hard instances that are not verified II.	35
4.3	Statistics for Panoramic Images used in Refer360° dataset.	35
4.4	Statistics for detected objects per image in Touchdown-SDR and Refer360° . On average, Refer360° images contain fewer of objects. However, these objects are from a wider variety of object types.	36
4.5	Language statistics for Refer360° dataset and other referring expression recognition datasets.	36
4.6	Statistics for dataset splits in Refer360° dataset.	37

4.7	Statistics for target locations image in Touchdown-SDR and Refer360° . Target is located on or near the wider variety of objects and further away from other objects.	38
4.8	Results for instruction ablation human study. Annotators need all instructions to complete the task accurately.	38
4.9	Linguistic analysis of 100 randomly sampled examples from Refer360° . We annotate each example for the presence and count of each phenomenon. c is the total number of instructions out of the 100 containing at least one example of the phenomenon. μ is the mean number of times each phenomenon appears per instruction sequence.	39
4.10	Results for the LingUNet on two benchmark datasets. Since LinGUNet designed for observing the full instruction set and the holistic view of the scene, and it performs significantly worse on Refer360°	40
5.1	Ablations showing the effect of each component in our model. Rows 2-4 show the effects of adding a single component to the baseline system (Row 1); Rows 5-7 show the effects of removing a single component from the full system (Row 8). NE is navigation error (in meters); lower is better. SR and OSR are success rate and oracle success rate (%); higher is better. See Sec. 5.3.2 for details.	52
5.2	Performance comparison of our method to previous work. NE is navigation error (in meters); lower is better. SR and OSR are success rate and oracle success rate (%) respectively (higher is better). Trajectory length (TL) on the test set is reported for completeness. *: <i>When submitting to the Vision-and-Language Navigation Challenge, we modified our search procedure to maintain physical plausibility and to comply with the challenge guidelines. The resulting trajectory has higher oracle success rate while being very long.</i>	55
6.1	Spatial Prompt Templates	74
6.2	FoV accuracy results for Refer360° and Touchdown with no hallucination baseline, best performing models, and Next FoV oracle model, i.e. the ability to look ahead for neighbor FoVs, and observing full 360° scenes. Our method outperforms the baseline models from the literature.	76
6.3	FoV accuracy results for Refer360° and Touchdown for methods using beam search or single candidate trajectory. HOLM consistently improves the baseline and does not use multiple trajectories.	77
6.4	FoV accuracy results for Refer360° and Touchdown for different methods for calculating affinity scores for HOLM. XLM-based affinity scores achieve the best performance.	77
6.5	FoV accuracy results for Refer360° and Touchdown when task data is used for object hallucination. The limitation of the domain data can be addressed using external resources such as pre-trained LMs.	78
6.6	Precision (P), Recall (R), and F1 scores for Refer360° and Touchdown for hallucinating objects in neighboring FoVs. Similar to the downstream task results, pre-trained LM performs the best.	78

6.7	FoV accuracy results for Refer360° and Touchdown for models processing unlabeled text. WE and LM are abbreviations for word embeddings and language models. All hallucination-based methods perform better than the baseline. XLM achieves the best performance in both datasets.	79
7.1	Comparison of OSMaN with the-state-of-the-art. Symbols % and 🕒 represent accuracy and efficiency for metrics, respectively. OSMaN achieves the best results for all evaluation settings and metrics. However, OSMaN is the only model that has access to ground-truth situated knowledge (Please see Table 7.2 for more results where all models have access the same sources of information).	92
7.2	Varying the source of situated knowledge for OSMaN. In top-2 rows, OSMaN has access to ground-truth map of the environment. Below that, all models only have access to navigable next locations. Black and <u>underlined</u> numbers denote the best and runner-up results for each column, respectively. Even when OSMaN needs to build situated knowledge from scratch it is more accurate than the state-of-the-art approaches (third row).	92
7.3	Oracle experiments for OSMaN where the ground-truth feature is fed to the system. OSMaN modules are accurate and straightforward, but there is still room for improvement. Each oracle feature improves all metrics significantly. Numbers in green show the difference between OSMaN and the oracle version for each metric in their columns.	93
7.4	Comparison of models on REVERIE and our proposed REVERIE-DeLiRE. We identify REVERIE-DeLiRE with OSMaN and RBERT, where they perform poorly. All models’ performances drop when evaluated in the REVERIE-DeLiRE subset.	94

Chapter 1

Introduction

Let us imagine a language . The language is meant to serve for communication between a builder A and an assistant B. A is building with building-stones; there are blocks, pillars, slabs and beams. B has to pass the stones, and that in the order in which A needs them. For this purpose they use a language consisting of the words 'block', 'pillar', 'slab', 'beam'. A calls them out; –B brings the stone which he has learnt to bring at such-and-such a call. – Conceive this as a complete primitive language.

Ludwig Wittgenstein, *Philosophical Investigations*

Artificial Intelligence (AI) systems have permeated our daily lives over the past decade. We now use voice-activated devices to manage our calendars, playlists, compose emails, or query our favorite restaurants' business hours. Although AI systems can handle these complex tasks, they are *static* devices that operate with *limited knowledge about their environment*. They are static in the sense that they do not move around or change the location of other objects. Their knowledge of the environment around them is limited, using mostly text or speech as input. The next generation of AI systems, namely *embodied AI*, will require new capabilities to surpass these static AI systems. An embodied AI system exists in a physical (or virtual) environment. It perceives *multi-modal sensory stimuli* – e.g. visual input in the form of point clouds of a LIDAR, RGB, infrared or depth input of a camera, and the speech signal. These embodied AI systems are often capable of *acting* on the environment, either by moving through it or moving things in it.

These new embodied AI systems are likely to become a part of our daily lives, just like their older static counterparts. For instance, we have seen great advances in the capabilities of autonomous vehicles for transportation and logistics. In the future, robots could be useful assistants in the household, helping the elderly take care of daily chores. We might use drones to pick up and deliver packages, or even record our memorable experiences from above. Virtual agents might assist in teaching children about geography or biology. One common denominator in all these use cases of embodied AI is the need to communicate with humans about the environment – e.g., giving directions, or referring to a specific physical objects.

The central question this thesis considers is **how can we better enable AI systems to**

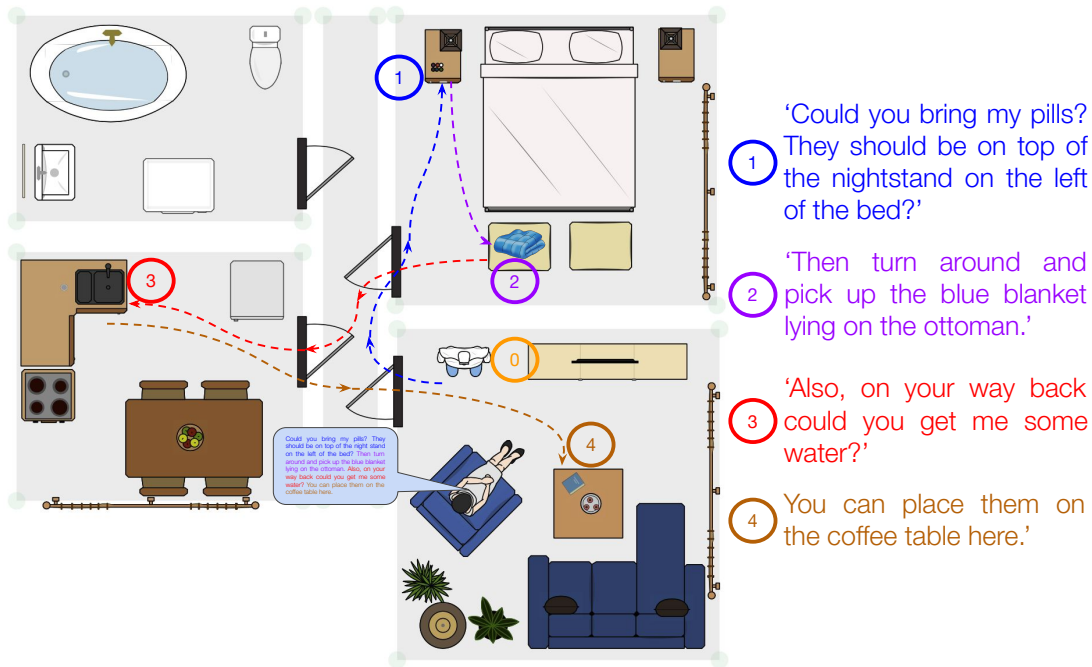


Figure 1.1: An illustration for a complex task for an embodied AI system given four instructions. Colored dashed lines show desired trajectory for the robot starting position (0) for the instruction in the same color. Colored numbers in parentheses represent the expected endpoint for the instruction in the same color.

naturally communicate while referring to the world and acting upon it? We focus on **language grounding**, the process of linking units of language (i.e., words, phrases, sentences) to actions of the system (i.e., moving in the world, changing perceptual field) and visual input (i.e., photographic sensory perception). To highlight the technical challenges involved in addressing this process, we use the following scenario as a motivating example: An older person is communicating with an AI robot to get assistance at home. This scenario is depicted in Figure 1.1. As a starting point, the robot should be able to identify and localize mentioned objects (i.e. *pills, night stand, blue blanket, ottoman, some water, coffee table*). Further, the robot needs to recognize how objects are spatially related to each other (i.e. *on top, on the left, lying on, on*). In the first instruction (1), the user does not mention the room to go to – instead, the robot must infer the room. In the second instruction (2), the robot needs to have a spatial understanding of the environment it operates in – i.e., when facing the nightstand, the ottoman would be behind it. The third instruction (3) also does not mention the destination explicitly. The robot needs to decide where to get water after leaving the room. One path leads to the kitchen, and the other leads to the bathroom; the robot could reason, “would they use the phrase *on the way back* if the path leads to the kitchen?” This reasoning could eliminate an unnecessary visit to the bathroom.

Language grounding has many applications in various fields such as robotics, autonomous driving, and virtual reality. To study this important topic, we use three specific technical tasks, corresponding to each of the high-level challenges outlined above: referring expression recognition, vision-and-language navigation, and remote object localization. The goal of referring expressions

is to unambiguously target one specific object in the world. In the first instruction (1) of Figure 1.1, the second sentence is an example of a referring expression which unambiguously describes the pills. The goal of vision-and-language navigation is to help navigate an embodied AI system in the world with natural language instructions to a target location. The first (1), third (3), and fourth (4) instructions in Figure 1.1 are example instructions for the vision-and-language navigation task such that the robot needs to visit bedroom, kitchen, and living room to complete the instruction. Remote object localization combines the previous two tasks into one: given a high-level natural language instruction sequence (e.g. (1)) the embodied AI system needs to navigate to a remote location in the world and choose the correct object in that remote location.

All the technical challenges illustrated in this scenario are defined in the following section. After describing these technical challenges, Section 8.1 provides an overview of the main contributions.

1.1 Technical Challenges

This thesis aims to study language grounding – the linguistic phenomenon by which words are linked to the world. To better understand and model language grounding, we provide scaffolding for this problem in the form of three core technical challenges, described below.

Spatial Grounding To properly link words with elements of the world, we first need to understand the spatial relationships between objects in the world. Humans naturally take advantage of the spatial structure of the world to unambiguously refer to a specific object. As an example, let us revisit the first instruction (1) mentioned in Figure 1.1. First, to complete this instruction, the system needs to identify all objects mentioned in the text, i.e., *pills*, *nightstand*, *bed*. But how can these pieces of text be grounded to objects detected in visual input? Also, how are all mentioned objects related to each other? Next, the system needs to identify spatial structures between objects, i.e., *on top on the left*. These spatial structures are cross-object relationships that tie each of objects mentions to one another. How can an AI system model cross-object relationships as part of the language grounding process? Given the complexity of this problem, let us start with a scenario where the AI system has access to all the visual information upfront. We do this by studying language grounding in the context of a static visual field – i.e., a single image – using the task of image-based referring expression recognition.

We define two main research questions within this challenge:

Q1.1 First, we want to better understand how recent neural network architectures perform the task of language grounding: How much of the world’s spatial structure is explicitly integrated into these recent models?

Q1.2 We are interested in building new models that explicitly learn to reason about the world’s spatial structure, including physical relationships between objects: How can we model cross-object relationships while referring to the visual world?

Sequential Grounding In real-world scenarios, people do not see the whole world at once but instead explore it sequentially by re-orienting themselves or moving in the world. This challenge is two-fold: First, grounding needs to be done using our current partial view (and potentially, with the history of our previous observations). The partial view of the world will increase the level of ambiguity between words and the world since the words may refer to something out-of-view. For instance, for instruction two (2) in Figure 1.1, after completing the first instruction, the robot faces the nightstand and does not see the ottoman and the blue blanket. Second, we need to link words with the most accurate action to reach our goal (e.g., reaching a specific location in the house) in addition to linking words to the world. The correct interpretation of the natural language input depends on correct sequences of actions. There could be *many alternatives* when grounding natural language to actions and the world, but only a small fraction of action sequences are intended by the user – or lead to the users intended outcome. For instance, in the third instruction (3), the user’s phrase “*on your way back*” make sense only when moving from bedroom to kitchen on the way back to the living room but not moving from bedroom to bathroom. In realistic scenarios, both cases need to be handled together (partial view and relevance to the next action). To get a clearer understanding of both cases, we study them in two steps:

Q2.1 How do people express themselves when only partial views are available to them? What are the defining characteristics of human language when referring to actions in a partially observed world?

Q2.2 How can we build a computational model that can reason about and operate in a partially observed environment?

Prior Knowledge for Grounding When communicating with other people, what is not said can be nearly as informative as what is said. According to Grice’s maxim of quantity [70], we prefer to be as informative as possible, but not more. In communicating with the embodied AI systems, this maxim implies that we would prefer to tell the AI system just enough information. We cannot give an extremely detailed description of how to complete a task or all the knowledge about the world. Instead, we expect the AI system to have some prior knowledge about the world. This phenomenon is sometimes referred to as common-sense knowledge. For instance, in Figure 1.1, the user does not even specify the names of the rooms, based on our common sense we can infer that. What kind of prior knowledge is relevant to interpreting language referring to the world? We focus on two types of prior knowledge. (1) General knowledge refers to the generic common-sense people have when referring to objects and locations (e.g., refrigerators are usually in kitchens). (2) Situated knowledge refers to knowledge about things specific to the situation (e.g., the ottoman being next to the bed in the example figure, but it may very well be in living room in a specific house). To study these two types of prior knowledge, we focus on knowledge related to objects and their spatial locations, given their prominence in human language when referring to the world.

First, we study the *general a priori knowledge of objects*, understanding generic knowledge about objects, their properties, and more specifically, spatial locations of objects. For instance, if we observe a table, we expect to see chairs around it, especially in a dining room. So, if we are looking for chairs, we naturally look close to the table to find them. This kind of generic knowledge is often referred to as common-sense. While humans accumulate this knowledge over long periods of time, can we build methods to help AI systems to start learning as well? To make this tractable, we focus on specific knowledge about objects and their locations.

Second, we study *situated knowledge* which is the understandings specific to the current environment – i.e., which objects appear in the environment and in what spatial configuration. When a person gives directions in their own house, they are likely to take advantage of knowledge about the specific layout of the house. While giving instructions to someone else, they would imagine moving around in the specific environment and use this knowledge to simplify instructions. To interpret such instruction, the AI system needs to ground language to the environment that has not been described explicitly.

To see how these two types of prior knowledge work together, let us revisit instructions in Figure 1.1. The first instruction **(1)** “Could you bring my pills? They should be on top of the nightstand on the left of the bed?” does not mention which room to go. A general knowledge like “nightstand are likely to appear in bedrooms” gives us a hint about the human’s intention. On the other hand, the situated knowledge is about *aligning our understanding* of this specific bedroom and how objects are located specifically in this bedroom. Let us look at the second instruction “Then turn around and pick up the blue blanket lying on the ottoman.” In this example, human *assumes* that the robot is looking at the nightstand after completing the first instruction **(1)** and *know* that the ottoman is behind the robot. Based on their understanding of the bedroom (i.e., situated knowledge), they use the phrase “*turn around.*” Using all this prior knowledge is a seamless process for humans, but how can an embodied AI system learn this general and situated knowledge? Our research questions are then:

Q3.1 How can an AI system acquire generic knowledge about the spatial properties of objects and take advantage of it when grounding language to the world?

Q3.2 How can an AI system use situated knowledge specific to its current environment when interpreting language input?

The following section gives an overview of this thesis’s contributions to address the technical challenges described in this section.

1.2 Thesis Contributions

- **The Role of Language in Language Grounding.** We first build an understanding on how state of the art systems use language for the referring expression recognition (**Q1.1**) by presenting a detailed empirical analyses of the state-of-the-art systems. The goal of this task is to identify the object in an image referred to by a natural language expression. We find

strong evidence that even sophisticated and linguistically motivated models for this task may ignore the important properties of language instead relying on shallow correlations introduced by unintended biases in the data selection and annotation process. This work was published at NAACL 2018 and described in Chapter 2.

- **Modeling Cross-Object Relationships.** We introduce GroundNet to address cross-object relationships for spatial grounding (**Q1.2**). GroundNet is an approach for building neural networks for referring expression recognition tasks based on the input referring expressions' syntactic structure. We show that GroundNet's syntax-based approach aids the localization of all object mentions and all relationships between mentioned objects. This study was published at AAAI 2018 and is explained in Chapter 3.
- **A Novel Benchmark For Referring Expression Recognition in 360° Images.** We build a benchmark with all challenging characteristics of the sequential grounding to study **Q2.1** in more depth. Our dataset is the first large-scale language grounding benchmark that has fine-grained alignments between partial observations of a 3D image, expert actions, and natural language instructions. This work was presented at ACL 2020 and is described in Chapter 4.
- **Reasoning About Alternatives for Vision-and-Language Navigation.** We introduce a pragmatic reasoning model based on the Rational Speech Act. The model can reason about alternative interpretations of natural language input for a sequential grounding task (**Q2.2**). Our work is the first application of the pragmatic reasoning in a photo-realistic vision-and-language environment. This work was published at NeurIPS 2018 and is explained in Chapter 5,
- **General Knowledge of Objects for Referring Expression Recognition in Partially-Observed Scenes.** We introduce HOLM, Hallucinating Objects with Language Models, to address the challenge of partial observability to study **Q3.1**. HOLM uses large pre-trained language models (LMs) to infer object hallucinations for the unobserved part of the environment. This work will appear at ACL 2022 and is described in Chapter 6.
- **Situated Knowledge for Remote Embodied Visual Referring Expression.** We introduce OSMaN, Object-based Simple Map Navigator for remote object localization to study **Q3.2**. OSMAN has the situated knowledge of objects, their locations, and how these locations are connected to each other. OSMAN uses this knowledge for navigating to a target location and predicting a target object. This work is described in Chapter 7.

Next in Chapter 2, we start exploring research questions regarding the spatial grounding.

Chapter 2

The Role of Language in Language Grounding

Science, my boy, is made up of mistakes, but they are mistakes which it is useful to make, because they lead little by little to the truth.

Jules Verne

Spatial referring expressions are part of our social life (“Please drop me at the blue house next to the red mailbox.”) and also part of professional interactions (“Could you pass the small scalpel to the right of the forceps?”). These natural language expressions are uttered to locate an object in the visual world uniquely. We understand such expressions by identifying mentioned objects and resolve described spatial relationships between objects using their visual input. However, how do machines operationalize such a process? This chapter aims to understand how state-of-the-art neural network models address the referring expression recognition task. We do this by probing the performance of models by perturbing their natural language input. We presented the work described in this chapter in the following publication:

- Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency, “Visual Referring Expression Recognition: What Do Systems Actually Learn?”, In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), ACL, pp. 781-787, 2018.

The code for reproducing experiments in this chapter is publicly available on Github¹.

¹<https://github.com/volkancirik/neural-sieves-refexp>

2.1 Overview

There has been increasing interest in modeling natural language in the context of a visual grounding. Several benchmark datasets have recently been introduced for describing a visual scene with natural language [32], describing or localizing specific objects in a scene [105, 146], answering natural language questions about the scenes [13], and performing visually grounded dialogue [46]. Here, we focus on referring expression recognition (RER) – the task of identifying the object in an image that is referred to by a natural language expression produced by a human [41, 93, 97, 105, 146, 160, 183, 239].

Recent work on RER has sought to make progress by introducing models that are better capable of reasoning about linguistic structure [93, 160] – however, since most of the state-of-the-art systems involve complex neural parameterizations, what these models actually learn has been difficult to interpret. This is concerning because several post-hoc analyses of related tasks [1, 49, 67, 99, 243] have revealed that some positive results are actually driven by superficial biases in datasets or shallow correlations without deeper visual or linguistic understanding. Evidently, it is hard to be completely sure if a model is performing well for the right reasons.

To increase our understanding of how RER systems function, we present several analyses inspired by approaches that probe systems with perturbed inputs [102] and employ simple models to exploit and reveal biases in datasets [26]. First, we investigate whether systems that were designed to incorporate linguistic structure actually require it and make use of it. To test this, we perform perturbation experiments on the input referring expressions. Surprisingly, we find that models are robust to shuffling the word order and limiting the word categories to nouns and adjectives. Second, we attempt to reveal shallower correlations that systems might instead be leveraging to do well on this task. We build two simple systems called Neural Sieves: one that completely *ignores* the input referring expression and another that only predicts the category of the referred object from the input expression. Again, surprisingly, both sieves are able to identify the correct object with surprising precision in top-2 and top-3 predictions. When these two simple systems are combined, the resulting system achieves precisions of 84.2% and 95.3% for top-2 and top-3 predictions, respectively. These results suggest that to make meaningful progress on grounded language tasks, we need to pay careful attention to what and how our models are learning, and whether our datasets contain exploitable bias.



Figure 2.1: Overview of Neural Sieves. Sieve I filters object types having multiple instances. Sieve II filters objects of one category mentioned in referring expression. Objects of the same category have the same color frames. Best seen in color.

2.2 Analysis by Perturbation

In this section, we analyze how the state-of-the-art referring expression recognition systems utilize linguistic structure. We conduct experiments with perturbed referring expressions where various aspects of the linguistic structure are obscured. We perform three types of analyses: the first one studying syntactic structure (Section 2.2.2), the second one focusing on the importance of word categories (Section 2.2.3), and the final one analyzing potential biases in the dataset (Section 2.2.4).

2.2.1 Analysis Methodology

To perform our analysis, we take two state-of-the-art systems CNN+LSTM-MIL [160] and CMN [93] and train them from scratch with perturbed referring expressions. We note that the perturbation experiments explained in next subsections are performed on all train and test instances. All experiments are done on the standard train/test splits for the Google-Ref dataset [146]. Systems are evaluated using the precision@ k metric, the fraction of test instances for which the target object is contained in the model’s top- k predictions. We provide further details of our experimental methodology in Section 2.3.1.

2.2.2 Syntactic Analysis by Permuting Word Order

In English, the word order is important for correctly understanding the syntactic structure of a sentence. Both models we analyze use Recurrent Neural Networks (RNN) [53] with Long Short-Term Memory (LSTM) cells [84]. Previous studies have shown that recurrent architectures can perform well on tasks where word order and syntax are important: for example, tagging [123], parsing [205], and machine translation [16]. We seek to determine whether recurrent models for RER depend on syntactic structure.

Premise 1: *Randomly permuting the word order of an English referring expression will obscure its syntactic structure.*

We train CMN and CNN+LSTM-MIL with shuffled referring expressions as input and evaluate their performance.

Model	No Perturbation	Shuffled	Δ
CMN	.705	.675	-.030
LSTM+CNN-MIL	.684	.630	-.054

Table 2.1: Results for Shuffling Word Order for Referring Expressions. Δ is the difference between no perturbation and shuffled version of the same system.

Table 2.1 shows accuracies for models with and without shuffled referring expressions. The column with Δ shows the difference in accuracy compared to the best performing model without shuffling. The drop in accuracy is surprisingly low. Thus, we conclude that these models do not strongly depend on the syntactic structure of the input expression and may instead leverage other, shallower, correlations.

2.2.3 Lexical Analysis by Discarding Words

Following the analysis presented in Section 2.2.2, we are curious to study what other aspects of the input referring expression may be essential for the state-of-the-art performance. If the syntactic structure is largely unimportant, it may be that spatial relationships can be ignored. Spatial relationships between objects are usually represented by prepositional phrases and verb phrases. In contrast, simple descriptors (e.g. green) and object types (e.g. table) are most often represented by adjectives and nouns, respectively. By discarding all words in the input that are not nouns or adjectives, we hope to test whether spatial relationships are actually important to the state-of-the-art models. Notably, both systems we test were specifically designed to model object relationships.

Premise 2: *Keeping only nouns and adjectives from the input expression will obscure the relationships between objects that the referring expression describes.*

Table 2.2 shows accuracies resulting from training and testing these models on only the nouns and adjectives in the input expression. Our first observation is that the accuracies of models drop the most when we discard the nouns (the rightmost column in Table 2.2). This is reasonable since

Models	Noun & Adj (Δ)	Noun (Δ)	Adj (Δ)
CMN	.687 (-.018)	.642 (-.063)	.585 (-.120)
LSTM+CNN-MIL	.644 (-.040)	.597 (-.087)	.533 (-.151)

Table 2.2: Results with discarded word categories. Numbers in parentheses are Δ , the difference between the best performing version of the original model.

nouns define the types of the objects referred to in the expression. Without nouns, it is extremely difficult to identify which objects are being described. Second, although both systems we analyze model the relationship between objects, discarding verbs and prepositions, which are essential in determining the relationship among objects, does not drastically reduce their performance (the second column in Table 2.2). This may indicate the superior performance of these systems does not specifically come from their modeling approach for object relationships.

2.2.4 Bias Analysis by Discarding Referring Expressions

Goyal et al. [67] show that some language and vision datasets have exploitable biases. Could there be a dataset bias that is exploited by the models for RER?

Premise 3: *Discarding the referring expression entirely and keeping only the input image creates a deficient prediction problem: achieving high performance on this task indicates dataset bias.*

We train CMN by removing all referring expressions from train and test sets. We call this model “image-only” since it ignores the referring expression and will only use the input image. We compare the CMN “image-only” model with the state-of-the-art configuration of CMN and a *random baseline*. Table 2.3 shows precision@ k results. The “image-only” model is able to surpass the random baseline by a large margin. This result indicates that the dataset is biased, likely as a result of the data selection and annotation process. During the construction of the dataset, Mao et al. [146] annotate an object box only if there are at least 2 to 4 objects of the

Model	P@1	P@2	P@3	P@4	P@5
CMN	.705	.926	.979	.993	.998
CMN “image-only”	.411	.731	.885	.948	.977
Random Baseline	.204	.403	.557	.669	.750

Table 2.3: Results with discarded referring expressions. Surprisingly, the top-2 prediction (73.1%) of the “image-only” model is better than the top prediction of the state-of-the-art (70.5%).

same type in the image. Thus, only a subset of object categories ever appears as targets because some object types rarely occur multiple times in an image. In fact, out of 90 object categories in MSCOCO, 43 of the object categories are selected as the target objects less than 1% of the time they occur in images. This potentially explains the relative high performance of the “image-only” system.

2.2.5 Discussion

The previous analyses indicate that exploiting bias in the data selection process and leveraging shallow linguistic correlations with the input expression may go a long way towards achieving high performance on this dataset. First, it may be possible to simplify the decision of picking an object to a much smaller set of candidates without even considering the referring expression. Second, because removing all words except for nouns and adjectives only marginally hurt performance for the systems tested, it may be possible to further reduce the set of candidates by focusing only on simple properties like the category of the target object rather than its relations with the environment or with adjacent objects.

2.3 Neural Sieves

We introduce a simple pipeline of neural networks, Neural Sieves, that attempt to reduce the set of candidate objects down to a much smaller set that still contains the target object given an image, a set of objects, and the referring expression describing one of the objects.

Sieve I: Filtering Unlikely Objects. Inspired by the results from Section 2.2.4, we design an “image-only” model as the first sieve for filtering *unlikely* objects. For example in Figure 2.1, Sieve I filters out the backpack and the bench from the list of bounding boxes since there is only one instance of these object types. We use a similar parameterization of one of the baselines (CMN_{LOC}) proposed by Hu et al. [93] for Sieve I and train it by only providing spatial and visual features for the boxes, ignoring the referring expression. More specifically, for visual features r^{vis} of a bounding box of an object, we use Faster-RCNN [181]. We use 5-dimensional vectors for spatial features $r^{spat} = [\frac{x_{min}}{W_V}, \frac{y_{min}}{H_V}, \frac{x_{max}}{W_V}, \frac{y_{max}}{H_V}, \frac{A_r}{A_V}]$ where A_r is the size and $[x_{min}, y_{min}, x_{max}, y_{max}]$ are coordinates for bounding box r and A_V, W_V, H_V are the area, the width, and the height of the input image V . These two representations are concatenated as $r^{vis,spat} = [r^{vis}, r^{spat}]$ for a bounding box r .

We parameterize Sieve I with a list of bounding boxes R as the input with a parameter set Θ_I as follows:

$$s_I = W_I^{score} r^{vis,spat} \quad (2.1)$$

$$f_I(R; \Theta_I) = softmax(s_I) \quad (2.2)$$

Each bounding box is scored using a matrix W_I^{score} . Scores for all bounding boxes are then fed to softmax to get a probability distribution over boxes. The learned parameter Θ_I is the scoring matrix W_I^{score} .

Sieve II: Filtering Based on Objects Categories After filtering *unlikely* objects based only on the image, the second step is to determine which object category to keep as a candidate for prediction, filtering out the other categories. For instance, in Figure 2.1, only instances of suitcases are left as candidates after determining which type of object the input expression is talking about. To perform this step, Sieve II takes the list of object candidates from Sieve I and keeps objects having the same object category as the referred object. Unlike Sieve I, Sieve II uses the referring expression to filter bounding boxes of objects. We again use the baseline model of CMN_{LOC} from the previous work [93] for the parametrization of Sieve II with a minor modification: instead of predicting the referred object, we make a binary decision for each box of whether the object in the box is the same category as the target object.

More specifically, we parameterize Sieve II as follows:

$$\hat{r}^{vis,spat} = W_{II}^{vis,spat} r^{vis,spat} \quad (2.3)$$

$$z_{II} = \hat{r}^{vis,spat} \odot f_{att}(T) \quad (2.4)$$

$$\hat{z}_{II} = z_{II} / \|z_{II}\|_2 \quad (2.5)$$

$$s_{II} = W_{II}^{score} \hat{z}_{s2} \quad (2.6)$$

$$f_{II}(T, R; \Theta_{II}) = sigmoid(s_{II}) \quad (2.7)$$

We encode the referring expression T into an embedding with $f_{att}(T)$ which uses an attention mechanism [16] on top of a 2-layer bidirectional LSTM [188].

We project bounding box features $r^{vis,spat}$ to the same dimension as the embedding of referring expression (Eq 2.3). Text and box representations are element-wise multiplied to get z_{II} as a joint representation of the text and bounding box (Eq 2.4). We L2-normalize to produce \hat{z}_{II} (Eq 2.5, 2.6). Box scores s_{II} are calculated with a linear projection of the joint representation (Eq 2.6) and fed to the sigmoid function for a binary prediction for each box. The learned parameters Θ_{II} are $W_{II}^{vis,spat}$, W_{II}^{score} , and parameters of the encoding module f_{att} .

2.3.1 Filtering Experiments

We are interested in determining how accurate these simple neural sieves can be. High accuracy here would give a possible explanation for the high performance of more complex models.

Dataset. For our experiments, we use Google-Ref [146] which is one of the standard benchmarks for referring expression recognition. It consists of around 26K images with 104K annotations. We use their Ground-Truth evaluation setup where the ground truth bounding box annotations from

Model	precision@ k	Accuracy
CMN	1	.705
CMN	2	.926
CMN	3	.979
LSTM+CNN-MIL	1	.684
LSTM+CNN-MIL	2	.907
LSTM+CNN-MIL	3	.972
Neural Sieve I	1	.401
Neural Sieve I	2	.712
Neural Sieve I	3	.866
Neural Sieve I + II	1	.488
Neural Sieve I + II	2	.842
Neural Sieve I + II	3	.953

Table 2.4: Precision@ k accuracy for Neural Sieves and state-of-the-art systems. Note that even without using the referring expression, Sieve I is able to reduce the number of candidate boxes to 3 for 86.6% of the instances. When we further predict the type of objects with Sieve II, the number of candidate boxes is reduced to 2 for 84.2% of the instances.

MSCOCO [127] are provided to the system as a part of the input. We used the split provided by Nagaraja et al. [160] where splits have disjoint sets of images. We use precision@ k for evaluating the performance of models.

Implementation Details. To train our models, we used stochastic gradient descent for 6 epochs with an initial learning rate of 0.01 and multiplied by 0.4 after each epoch. Word embeddings were initialized using GloVe [169] and finetuned during training. We extracted features for bounding boxes using the fc7 layer output of Faster-RCNN VGG-16 network [181] pre-trained on MSCOCO dataset [127]. Hyperparameters such as hidden layer size of LSTM networks were picked based on the best validation score. For perturbation experiments, we did not perform any grid search for hyperparameters. We used hyperparameters of the previously reported best performing model in the literature.

Baseline Models. We compare Neural Sieves to the state-of-the-art models from the literature. **LSTM + CNN - MIL** Nagaraja et al. [160] score *target object-context object* pairs using LSTMs for processing the referring expression and CNN features for bounding boxes. The pair with the highest score is predicted as the referred object. They use Multi-Instance Learning for training the model. **CMN** [93] is a neural module network with a tuple of object-relationship-subject nodes. The text encoding of tuples is calculated with a two-layer bi-directional LSTM and an attention mechanism [16] over the referring expression.

2.3.2 Results

Table 2.4 shows the precision scores. The referred object is in the top-2 candidates selected by Sieve I 71.2% of the time and in the top-3 predictions 86.6% of the time. Combining both sieves into a pipeline, these numbers further increase to 84.2% for top-2 predictions and to 95.3% for top-3 predictions. Considering the simplicity of Neural Sieve approach, these are surprising results: two simple neural network systems, the first one ignoring the referring expression, the second predicting only object type, are able to reduce the number of candidate boxes down to 2 on 84.2% of instances.

2.4 Related Work

Referring expression recognition and generation is a well studied problem in intelligent user interfaces [23], human-robot interaction [24, 55, 227], and situated dialogue [107]. Kazemzadeh et al. [105] and Mao et al. [146] introduce two benchmark datasets for referring expression recognition. Several models that leverage linguistic structure have been proposed. Nagaraja et al. [160] propose a model where the target and supporting objects (i.e. objects that are mentioned in order to disambiguate the target object) are identified and scored jointly. The resulting model is able to localize supporting objects without direct supervision. Hu et al. [93] introduce a compositional approach for the RER task. They assume that the referring expression can be decomposed into a triplet consisting of the target object, the supporting object, and their spatial relationship. This structured model achieves state-of-the-art accuracy on the Google-Ref dataset. Cirik et al. [41] propose a type of neural modular network [12] where the computation graph is defined in terms of a constituency parse of the input referring expression.

Previous studies on other tasks have found that the state-of-the-art systems may be successful for reasons different than originally assumed. For example, Chen et al. [27] show that a simple logistic regression baseline with carefully defined features can achieve competitive results for reading comprehension on CNN/Daily Mail datasets [82], indicating that more sophisticated models may be learning relatively simple correlations. Similarly, Gururangan et al. [72] reveal bias in a dataset for semantic inference by demonstrating a simple model that achieves competitive results *without looking at the premise*.

2.5 Conclusion

We have analyzed two RER systems by variously perturbing aspects of the input referring expressions: shuffling, removing word categories, and finally, by removing the referring expression entirely. Based on this analysis, we proposed a pipeline of simple neural sieves that captures many of the easy correlations in the standard dataset. Our results suggest that careful analysis is important both while constructing new datasets and while constructing new models for grounded language tasks. The techniques used here may be applied more generally to other tasks to give better insight into what our models are learning and whether our datasets contain exploitable bias.

Chapter 3

Modeling Cross-Object Relationships

Syntax, my lad. It has been restored to the highest place in the republic.

John Steinbeck

In this chapter, we further study spatial grounding in the referring expression recognition task. We build a computational model that explicitly reasons the spatial structure of objects. The work described in this chapter first appeared in the following publication:

- Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency, “Using Syntax To Ground Referring Expressions In Natural Images”, In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1. 2018.

The code for reproducing experiments in this chapter is publicly available on Github¹.

¹<https://github.com/volkancirik/groundnet>

3.1 Introduction

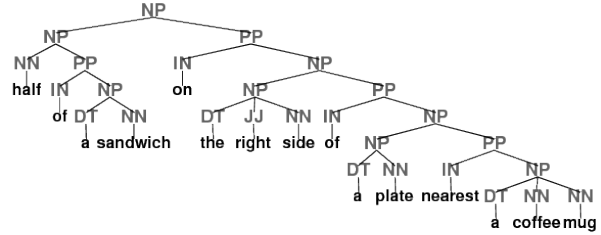
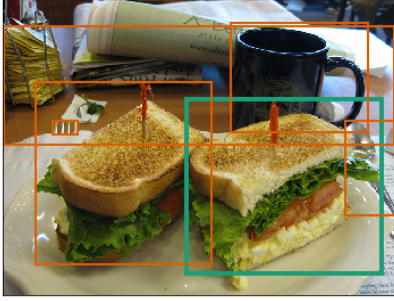
Spatial referring expressions are part of our everyday social life (“Please drop me at the blue house next to the red mailbox.”) and also part of professional interactions (“Could you pass the small scalpel to the right of the forceps?”). These natural language expressions are designed to uniquely locate an object in the visual world. The process of grounding referring expressions into visual scenes involves many intermediate challenges. As a first step, we want to locate all the objects mentioned in the expression. While one of these mentions refers to the target object, the other mentions (i.e. supporting object mentions) are also important because they were included by the author of the referring expression in order to disambiguate the target. In fact, [70] argued that supporting objects will only be mentioned when they are *necessary* for disambiguation. As a second step, we want to identify the spatial relationships between these objects. Is the target to the left of the supporting object? Is it beneath it? To make effective use of an identified supporting object, we must understand how this object is related to the target. And finally, for many natural referring expressions, the process is recursive: a supporting object may itself be identified by a relationship with another supporting object. As a result, models that reason about referring expressions must respect this hierarchy, processing sub-expressions before attacking larger expressions. Modeling this compositionality is critical to designing recognition systems that behave in an interpretable way and can justify their decisions.

In this chapter, we introduce GroundNet, the first dynamic neural architecture for referring expression recognition that takes full advantage of syntactic compositionality. Past approaches, such as the Compositional Neural Network (CMN) model [93], have relied on limited syntactic information in processing referring expressions – for example, CMN tracks a single supporting object – but have not modeled linguistic recursion and therefore is incapable of tracking multiple supporting objects. As shown in Figure 1, our GroundNet framework relies on a syntactic parse of the input referring expression to dynamically create a computation graph that reflects the recursive hierarchy of the input expression. As a result, our approach tracks intermediate localization decisions of all supporting objects. Following the approach of [11, 12], this computation graph is translated into a neural architecture that keeps interpretable information at each step of the way, as can be seen in Figure 3.1d.

We additionally introduce a new set of annotations that specify the correct locations of supporting objects in a portion of the standard benchmark dataset, GoogleRef [146] to evaluate the interpretability of models for referring expression recognition. Using these additional annotations, our empirical evaluations demonstrate that GroundNet substantially outperforms the state-of-the-art at intermediate predictions of the supporting objects, yet maintains comparable accuracy at target object localization. These results demonstrate that syntactic compositionality can be successfully used to improve interpretability in neural models of language and vision.

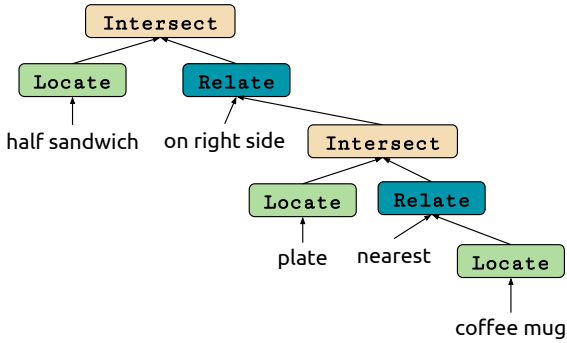
3.2 GroundNet

In this section, we explain the motivation of GroundNet, how we generate the computation graph for GroundNet, and finally detail the neural modules that we use for computing the localization the referring expressions.

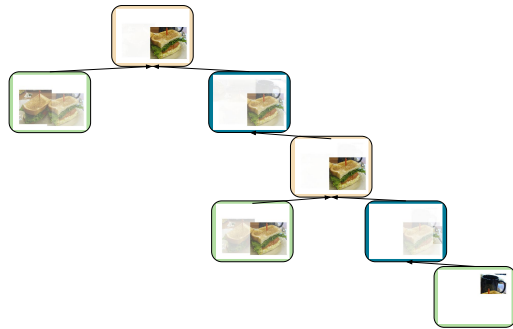


(a) An example referring expression from our validation set “half of a sandwich on the right side of a plate nearest a coffee mug”. Orange boxes are region candidates and green box is the referred bounding box.

(b) The parse tree for the referring expression in (a)



(c) Computation graph for the parse tree in (b)



(d) Grounding of objects in (a) with the computation graph in (c). The more visible objects have higher probabilities. Note that the model is able to ground supporting objects like the coffee mug.

Figure 3.1: An Overview of GroundNet. An referring expression (a) is first parsed (b). Then, the computation graph of neural modules is generated using the parse tree (c). Each node localizes objects present in the image (d).

3.2.1 Motivation

A referring expression disambiguates a target object using the object’s discriminative features such as color, size, texture etc., and their relative position to other *supporting* objects. Figure 3.1a shows a canonical example from our task w one half of a sandwich is referred by “half of a sandwich on the right side of a plate nearest a coffee mug”. Here the sandwich is disambiguated using relative clauses (e.g. “the right side of” , “nearest”) and the *supporting* objects (e.g “plate”, “coffee mug”). We observe that there is a correspondence between the linguistic compositional structure (i.e. the parse tree) of the referring expression and the process of resolving a referring expression. In Figure 3.1b, we see that the target object and supporting objects have a noun phrase (NP) on the parse tree of the referring expression. Also, the relative positioning of objects in the image (e.g. being on the right, or near) correspond to prepositional phrases (PP) on the tree. We design GroundNet based on this observation to localize the target object by modeling the

compositional nature of the language. The compositionality principle states that the meaning of a constituent is a function of (i) its building blocks and (ii) the recursive rules to combine them. In our case, the building blocks for the GroundNet is grounding of objects i.e. the probability of how likely an object is for word phrases. The combining rules are defined by the parse tree describing what these objects are and how they are related to each other.

GroundNet models the processing of a referring expression in a computation graph (see Figure 3.1c) based on the parse tree of the referring expression (see Figure 3.1b). Nodes of the computation graph have 3 different types aiming to capture the necessary computations for localizing the target object. `Locate` nodes ground a noun phrase (“half sandwich”, “plate”, “coffee mug”), i.e. pointing how likely that a given noun phrase refers to an object present in the image. For example, in Figure 3.1d, `Locate` node of the phrase “half sandwich” outputs higher probabilities for both halves of sandwiches compared to other objects. Prepositional phrases (“on right side”, “nearest”) correspond to `Relate` nodes in the computation tree. `Relate` nodes calculate how likely objects are related to the grounding of objects with given prepositional phrase. For instance, in Figure 3.1c, the `Relate` node of “nearest” computes how likely the objects are related to the grounding of “coffee mug” with the relation “nearest”. We convert the phrases coming from branches in the parse tree to `Intersect` nodes. It simply intersects two sets of groundings so that objects that have high likelihood in both branches will have high probabilities for the output (see the root node in Figure 3.1d). Since each node of this computation graph outputs a grounding for its subgraph, GroundNet is interpretable as a whole. At each node, we can visualize how model’s multiple predictions for objects propagates through the computation graph.

In following sections, we detail how we generate the computation graph and the neural modules used in GroundNet.

3.2.2 Generating a Computation Graph

GroundNet processes the referring expression with a computation graph (Figure 3.1c) based on to the parse tree (Figure 3.1b) of the referring expression. First, we parse the referring expression with Stanford Parser [145]. Then, we generate the computation graph (see Figure 3.1b, 3.1c for an example) for a parse tree with a recursive algorithm (see Algorithm 1).

Algorithm 1: Generate Computation Tree

```

1: procedure GenerateComputationTree(tree)
2:   left_NP = FindNP(tree.left)
3:   right_NP = FindNP(tree.right)
4:   if left_NP == "" then
|     return (Locate tree.text)
5:   Relate = FindPP(tree, [left_NP, right_NP])
6:   left_cg = GenerateComputationTree(left_NP)
7:   right_cg = GenerateComputationTree(right_NP)
8:   return (Intersect (left_cg) (Relate right_cg))
9: end procedure

```

Above, the function *FindNP* finds the noun-phrase with the largest word span of given root

node for left and right branches (line 2, 3). If the tree does not have an NP subtree, it returns a `Locate` node (line 4).

`FindPP` extracts the words between noun-phrases to model the relationship between them and returns a `Relate` node (line 5). For both left and right branches of the parse tree, the same algorithm is recursively called (lines 6, 7). Finally, the sub-computation graphs of left and right branches are merged (line 8) into an `Intersect` node.

Each node in the computation graph is decorated with the phrase T using the text span, i.e. constituents, of the corresponding parse tree node. We filter out the function words such as determiners ‘a’ and ‘the’. For instance, the `Locate` on the left in Figure 3.1c has the span of words “half sandwich” from the corresponding noun phrase “the half of a sandwich” in Figure 3.1b.

In the following section, we explain the set of neural modules that we design for performing the localization of the referring expression on a composed computation graph.

3.2.3 Neural Modules

We operationalize the computational graph for a referring expression into an end-to-end neural architecture by designing neural modules that represent each node of our graph. First, let us introduce the notation for referring expression task. For each referring expression, (I, R, X) are inputs where I is an image, R is the set of bounding boxes r_i of objects present in the image I , and X is a referring expression disambiguating a target object in bounding box r^* . Our aim is to predict r^* processing the referring expression in a computational graph with neural modules. In addition to (I, R, X) , neural modules use the output of other neural modules and the text span T of the computation node.

We detail parameterization of neural modules in following subsections and visualize them in Figure 3.2 for clarity.

Attend

This module induces a text representation for `Locate` and `Relate` nodes. It takes the words $\{w_i\}_{i=1}^{|T|}$ and embeds them to a word vector $\{e_i\}_{i=1}^{|T|}$. A 2-layer bidirectional LSTM network [188] processes embedded words. Both forward and backward layer representations are concatenated for both layers into a single hidden representation for each word as follows:

$$h_i = [h_i^{(1, fw)} h_i^{(1, bw)} h_i^{(2, fw)} h_i^{(2, bw)}] \quad (3.1)$$

The attention weights are computed with a linear projection using W^a :

$$a_i = \frac{\exp(W^a h_i)}{\sum_{i=1}^{|T|} \exp(W^a h_i)} \quad (3.2)$$

The output of `Attend` is the weighted average of word vectors e_i where the weights are attentions a_i .

$$f_a(T; \Theta_a) = \sum_{i=1}^{|T|} a_i e_i \quad (3.3)$$

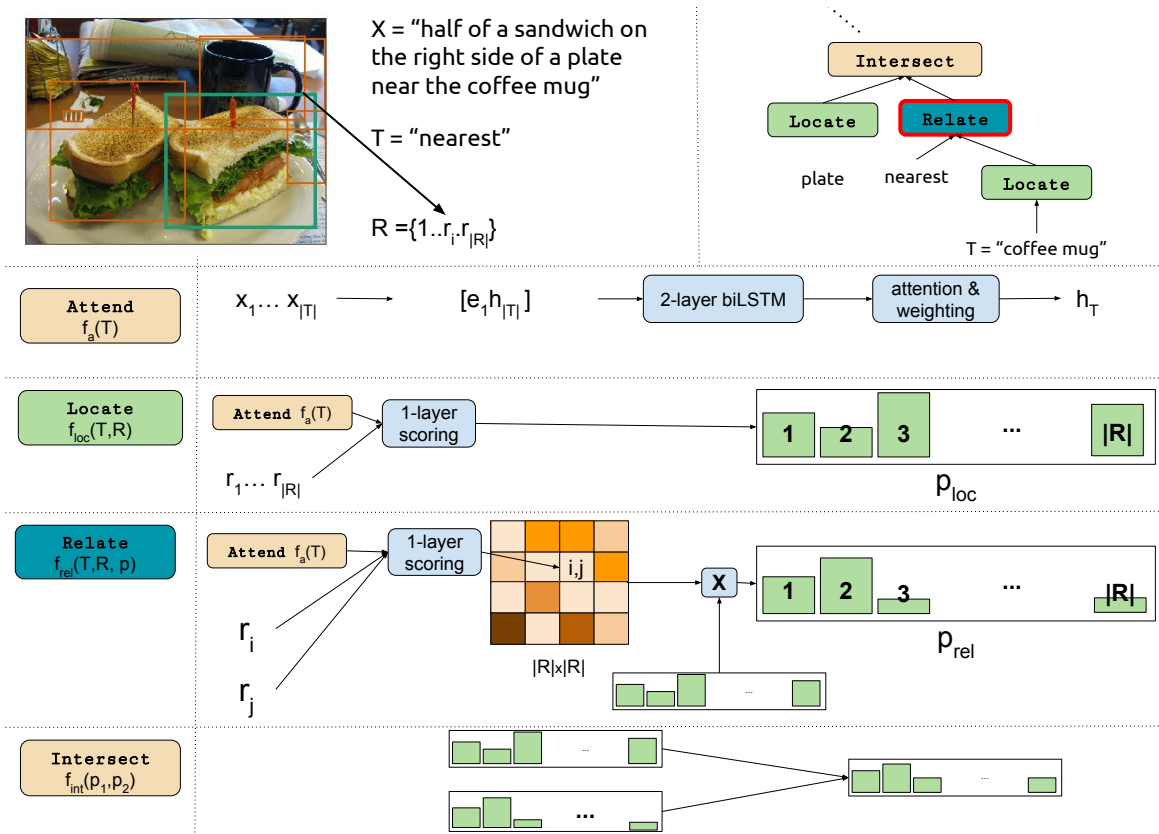


Figure 3.2: Illustrations of GroundNet’s neural modules. Upper left shows an example referring expression and the input for `Relate` node (upper right, highlighted in red) of a small section of a computation tree. Modules take inputs from module’s text span T , the set of bounding boxes R , and output probabilities of other nodes p_i . Best seen in color.

The learned parameters Θ_a of this module are the parameters of 2-layer bidirectional LSTM and scoring matrix W^a .

Locate

This module predicts which object is referred to for a text span, i.e. noun phrase, in the referring expression. It computes the probability distribution over bounding boxes using the output of `Attend` and feature representations of bounding boxes. For instance in Figure 3.1c, `Locate` node with input “half sandwich” localizes objects by scoring each bounding box. `Locate` node does so by scoring how well the text span “half sandwich” matches the content of each bounding box.

To represent a bounding box r , we use spatial and visual features. First, visual features r_{vis} for the bounding box are extracted using a convolutional neural network [181]. Second, spatial features represent position and size of the bounding box. We have 5-dimensional vectors for spatial features $r_{spat} = [\frac{x_{min}}{W_I}, \frac{y_{min}}{H_I}, \frac{x_{max}}{W_I}, \frac{y_{max}}{H_I}, \frac{S_r}{S_I}]$ where S_r is the size and $[x_{min}, y_{min}, x_{max}, y_{max}]$ are coordinates for bounding box r and S_I, W_I, H_I are area, width, and the height of the input image

I. These two representations are concatenated as $r_{vis,spat} = [r_{vis}r_{spat}]$ for a bounding box r .

We follow the previous work [93] for parametrization of `Locate`.

$$\hat{r}_{vis,spat} = W_{vis,spat}^{loc} r_{vis,spat} \quad (3.4)$$

$$z_{loc} = \hat{r}_{vis,spat} \odot f_a(T) \quad (3.5)$$

$$\hat{z}_{loc} = z_{loc} / \|z_{loc}\|_2 \quad (3.6)$$

$$s_{loc} = W_{score}^{loc} \hat{z}_{loc} \quad (3.7)$$

$$p_{loc} = \text{softmax}(s_{loc}) \quad (3.8)$$

$$f_{loc}(T, R; \Theta_{loc}) = p_{loc} \quad (3.9)$$

First, $r_{vis,spat}$ is projected to the same dimension as the text representation coming from the `Attend` (Eq 3.4). Text and box representations are element-wise multiplied to get z_{loc} for a joint representation of the text and bounding box. We normalize with L2-norm into \hat{z}_{loc} (Eq 3.5, 3.6). Localization score s_{loc} is calculated with a linear projection of the joint representation (Eq 3.7). Localization scores are fed to softmax to form a probability distribution p_{loc} over boxes. The learned parameters Θ_{loc} of this module are the matrices $W_{vis,spat}^{loc}$ and W_{score}^{loc} .

Relate

predicts how likely an object *relates* to the other objects with some relation described by the node’s text span. For instance, the relation “nearest” in Figure 3.1d holds for half-sandwich pairs, and a half-sandwich and coffee mug pair. Since the incoming `Locate` node to `Relate` outputs a high probability for the coffee mug, only objects near to coffee mug have a high probability. GroundNet does so by first computing a relationship score matrix for boxes and multiplying the scoring matrix with the grounding input. We do not define a set of relationships for `Relate`, instead, model learns how objects relate to each other using module’s text representation. Specifically, this module computes a relationship score matrix S_{rel} of size $R \times R$ consisting of scores for box i and j as follows:

$$\hat{r}_{i,j} = W_{spat}^{rel} r_{i,j} \quad (3.10)$$

$$z_{rel} = \hat{r}_{i,j} \odot f_a(T) \quad (3.11)$$

$$\hat{z}_{rel} = z_{rel} / \|z_{rel}\|_2 \quad (3.12)$$

$$S_{rel}[i, j] = W_{score}^{rel} \hat{z}_{rel} \quad (3.13)$$

$$p_{rel} = S_{rel} p \quad (3.14)$$

$$f_{rel}(T, R, p; \Theta_{rel}) = p_{rel} \quad (3.15)$$

Above, spatial representations of boxes are concatenated as $r_{i,j} = [r_{i,spat}, r_{j,spat}]$ and projected into the same dimension as text representation (Eq 3.10). Similar to `Locate`, text and box representations are fused with element-wise multiplication and L2-normalization (Eq 3.11, 3.12), then box pair is scored linearly (Eq 3.13).

Finally, the probability distribution p_{rel} over bounding boxes is calculated as $p_{rel} = S_{rel} p_{loc}$. The learned parameters Θ_{rel} of this module are the matrices W_{spat}^{rel} and W_{score}^{rel} .

Intersect

This module combines groundings coming from two branches of the computation graph by simply multiplying object probabilities and normalizing it to form a probability distribution. In the following section, we explain our experimental setup.

3.3 Experiments

Now, we detail our experimental setup. In our experiments, we are interested in following research questions:

- **(RQ1)** How successful models are incorporating the syntax and how important the dynamic and modular computation in exploiting the syntactic information?
- **(RQ2)** What are the accuracies of models for supporting objects and how these accuracies change depending on the syntactic information?

Now, we explain datasets used for our experiments.

Referring Expression Dataset. We use the standard Google-Ref [146] benchmark for our experiments. Google-Ref is a dataset consisting of around 26K images with 104K annotations. We use "Ground-Truth" evaluation setting where the ground truth bounding box annotations from MSCOCO [127] are used.

Supporting Objects Dataset. We also investigate the performances of models in terms of interpretability. We measure the interpretability of a model by its accuracy on both target and supporting objects. To this end, we introduce a new set of annotations on Google-Ref dataset. First, we run a pilot study on MTurk where all bounding boxes and the referring expression present to annotators². Our in-house annotator has an agreement of 0.75 - a standard metric in word alignment literature [68, 164] with three turkers on a small validation set of 50 instances. Overall, our annotator labeled 2400 instances – but only 1023 had at least one supporting object bounding box.

We remind that the training data does not have any annotations for supporting objects. Models should be able to predict supporting objects using only target object supervision and text input. We should emphasize that our work is the first to report quantitative results on supporting object for the referring expression task and we will release our annotation for future studies. Next, we provide details of our implementation.

Implementation Details. We trained GroundNet with backpropagation. We used stochastic gradient descent for 6 epochs with an initial learning rate of 0.01 and multiplied by 0.4 after each epoch. Word embeddings were initialized with GloVe [169] and finetuned during training. We extracted features for bounding boxes using fc7 layer output of Faster-RCNN VGG-16 network [181] pre-trained on MSCOCO dataset [127]. Hidden layer size of LSTM networks was searched over the range of {64,128,...,1024} and picked based on best validation split which is 2,5% of

²We did not provide the parse trees to not bias the annotators.

Model	Syntax	Dynamic Computation	Modularity	Object Relationship	Supporting(%)	Accurac(%)
LSTM+CNN - MMI						60.7
LSTM+CNN - MMI+visdif				✓		64.0
LSTM+CNN - MIL				✓	15.0	67.3
CMN			✓	✓	11.1	69.7
Recursive NN	✓	✓				51.5
CMN-syntax guided	✓		✓	✓		53.5
GroundNet	✓	✓	✓	✓	60.6	65.7
GroundNet-syntax-guided <code>Locate</code>	✓	✓	✓	✓	60.0	66.7
GroundNet-free-form	✓	✓	✓	✓	10.6	68.9

Table 3.1: The accuracy of models with the support of syntax, dynamic computation, modularity, relationship modeling, and supporting object predictions. Our model is the first syntax-based model with successful results and achieves the best results in supporting object localization.

training data separated from training split. Following the previous work [93], we used official validation split as the test. We initialized all parameters of the model with Xavier initialization [64] and used weight decay rate of 0.0005 as regularization. We implemented our model using PyTorch³ and plan to release our code for public use. Next, we explain models used in our experiments.

Baseline Models. We compare GroundNet to the recent models from the literature. **RecursiveNN** [197] uses the recursive structure of syntactic parses of sentences to retrieve images described by the input sentence. The text representation of a referring expression is recursively calculated following the parse tree of the referring expression. The text representation at root node is jointly scored with bounding box representations and the highest scoring box is predicted. **LSTM + CNN - MMI** [146] uses LSTMs processing the referring expression and CNN for extracting features for bounding boxes and the whole image. Model is trained with Maximum Mutual Information training. **LSTM + CNN - MMI+visdif** [239] introduce contextual features for a bounding box by calculating differences between visual features for object pairs. **LSTM + CNN - MIL**⁴ [160] scores object-supporting object pairs. The pair with the highest score is predicted. They use Multi Instance Learning for training the model. **CMN**⁵ [93] is a neural module network with a tuple of object-relationship-subject nodes. The text representation of tuples are calculated with an attention mechanism [16] over the referring expression. We also report results for **CMN - syntax guided** when a parse tree is used for extracting the object-relationship-subject tuples.

GroundNet with varying level of syntax. We investigate the effect of the syntax varying the level of use of the syntactic structure for GroundNet. **GroundNet** is the original model presented in the previous section where each node in computation graph uses the node’s text span for `Attend`. For **GroundNet-syntax-guided `Locate`** model, `Locate` nodes use the node’s

³<https://pytorch.org>

⁴Originally the authors use a new test split, whereas, we report results for the standard split of the dataset for this model.

⁵We report results for our reimplementation of this model where we did hyperparameter search the same as our model.

text span as an input to the `Attend` module. Whereas for `Relate` nodes can use all referring expression for inducing the text representation. For **GroundNet-free-form** model, Both `Locate` and `Relate` nodes use all of the referring expression as the input to `Attend`. Next, we explain our evaluation metrics used in our experiments.

Evaluation. To evaluate models for referring expression task we use the standard metric of accuracy. For evaluation of supporting objects, when there are multiple supporting objects, we consider a supporting object prediction as accurate only if at least one supporting object is correctly classified. To evaluate approaches modeling the supporting objects we use following methods. For LSTN+CNN-MIL, we use the context object of the maximum scoring target-context object pair as the supporting object. For CMN, we use the object with the maximum object score of a subject-relation-object tuple as the prediction for the supporting object. For GroundNet, we use the object with maximum probability as a prediction for intermediate nodes in the computation graph. In the following section, we discuss results of our experiments.

3.4 Results

We presented overall results in Table 3.1 for the compared models. We now discuss columns of the Table 3.1.

(RQ1) Syntax, Dynamic Computation, and Modularity. GroundNet variations achieve the best results among syntax-based models. “Recursive NN” homogeneously processes the referring expression throughout the parse tree structure. On the other hand, GroundNet modularly parameterizes multi-modal processing of localization and relationships. “CMN - syntax guided” has a fixed computation graph of a subject-relation-object tuple, whereas, GroundNet has a dynamic computation graph for each instance, thus, a varying number of computation nodes are induced. When compared to other syntax-based approaches, GroundNet results show that a dynamic *and* modular architecture is essential to achieve competitive results with a syntax-based approach.

(RQ2) Syntax for Supporting Objects. Our model achieves the highest accuracy on localizing the supporting objects when its modules are guided by syntax. “LSTM+CNN-MIL” and CMN does not exploit the syntax of the referring expression and poorly perform in localizing supporting objects. When we relax the syntactic guidance of GroundNet by letting all modules to attend to all of the referring expression, “GroundNet-free-form” also performs poorly on localizing supporting objects. These results suggest that leveraging syntax is essential in localizing supporting objects and there might be a tradeoff between being interpretable and being accurate for models. We qualitatively show a couple of instances from test set GroundNet and CMN in Figure 3.3. As an example, for the first instance, both GroundNet and CMN successfully predict the target object. GroundNet is able to localize both supporting objects (i.e. the girl and the disc) mentioned in the referring expression, whereas, CMN fails to localize the supporting objects. Next, we review the previous work related to GroundNet.

3.5 Related Work

Grounding Referential Expressions. The most of the recent work [61, 97, 146, 160, 183, 239] addresses grounding referential expression task with a fixed computation graph. In earlier studies [61, 97, 146, 183], the bounding boxes are scored based on their CNN and spatial features along with features for the whole image. Since each box is scored in isolation, these methods ignore the object relationships. More recent studies [93, 160, 239] show that modeling relationship between objects improves the accuracy of models. GroundNet has a dynamic computation graph and models the relationship between objects.

Modular Neural Architectures. Neural Module Networks (NMN) [11, 12] is a general framework for modeling compositionality of language using neural modules. A computation graph with neural modules as nodes is generated based on a parse tree of the input text. GroundNet shares the principles of this framework. We design GroundNet for referring expression task restricting each node grounded in the input image which keeps network interpretable throughout the computation.

Compositional Neural Network (CMN) [93] is also an instant of NMN aiming to remove language parser from the generation of computation graph by inducing text representations to localization and relationship modules using an attention mechanism. Their computation graph is fixed to the subject-relation-subject tuple but the input is dynamically constructed for modules. Our model, on the other hand, can handle multiple relationships mentioned in referring expressions (see the first row of Figure 3.3).

Syntax for Vision. Golland et al. [65] introduce a game-theoretic model successfully leverages syntax for grounding reference expressions for synthetic scenes. [39] use the visual context for solving prepositional phrase attachment resolution (PPAR) for sentences describing a scene. Unlike our model, their model relies on multiple parse trees and multiple segmentations of an image coming from a black-box image segmenter. Our model can also be extended to address PPAR setting where we only need to ground-truth object annotations for roots of multiple parse trees for the input sentences. [220] introduce a model localizing phrases in sentences that describe an image. However, their model relies on the annotation of phrase-object pairs. GroundNet only uses target object annotations and there is no supervision for supporting objects. [230] aim to address localization of phrases on region masks. Similar to our approach, they do not rely on ground-truth masks during training. However, unlike GroundNet, their model does not model relationship between objects.

3.6 Conclusion

In this chapter, we present GroundNet, a compositional neural module network designed for the task of grounding referring expressions. We also introduce a new auxiliary task and an annotation for localizing the supporting objects.

Our experiments on a standard benchmark show that GroundNet is the first model that successfully incorporates syntactic information for the referring expression task. This syntactic information helps GroundNet achieve state-of-the-art results in localizing supporting objects.

Our results show that recent models are unsuccessful at localizing supporting objects. This suggests that current solutions to referring expression task come with an interpretability-accuracy trade-off. Our approach substantially improves supporting object localization, while maintaining high accuracy, thus representing a new and more desirable point along the trade-off trajectory.

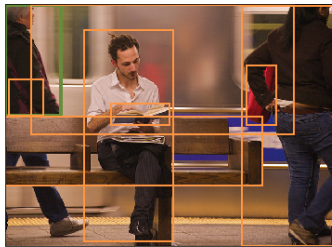
Acknowledgements

This project was partially supported by Oculus and Yahoo InMind research grants. The authors would like to thank anonymous reviewers and the members of MultiComp Lab at CMU for their valuable feedback.

Referring Expression



“a white color car behind a girl catching a disc”

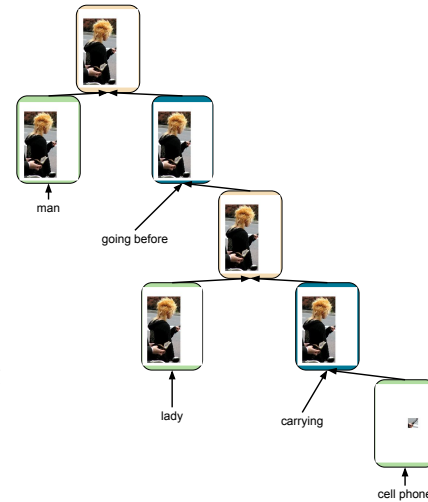
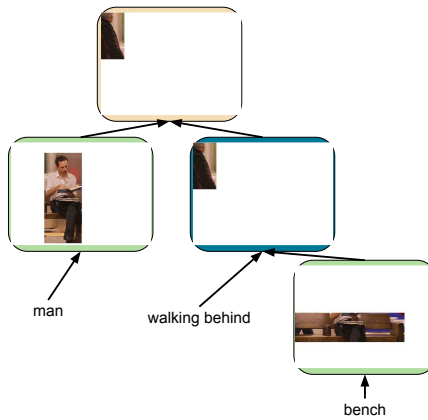
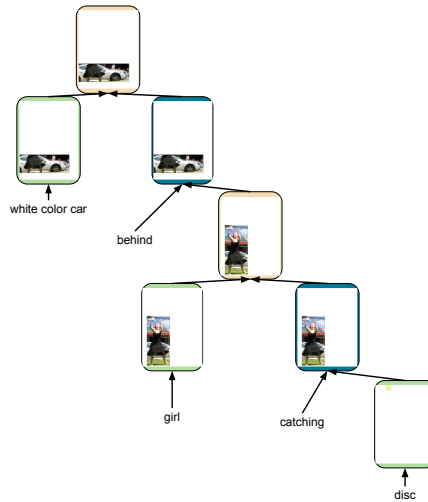


“the man walking behind the bench”



“a man going before a lady carrying a cellphone”

GroundNet



CMN

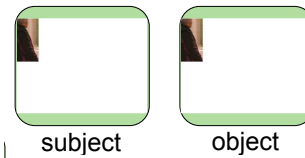
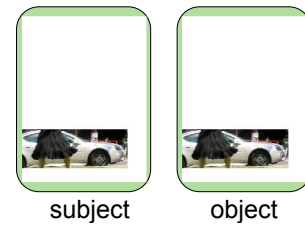


Figure 3.3: Qualitative Results for GroundNet. Bounding boxes and referring expressions to target object (in green boxes) on the left. GroundNet predictions in the middle and CMN predictions are on the right. GroundNet localizes not only the target object but also supporting objects (e.g. disc and girl in the first row, bench in the second).

Chapter 4

A Novel Benchmark For Referring Expression Recognition in 360° Images

Nothing has such power to broaden the mind as the ability to investigate systematically and truly all that comes under thy observation in life.

Marcus Aurelius

In the previous two chapters, we studied spatial grounding in static images. However, in the real world, humans explore the world sequentially by moving in the world or changing our heads' pitch and yaw. In this chapter, we aim to understand how people use language when they process partial visual information about the world. To this end, we build a novel large-scale language grounding benchmark with human participants using 360° images. The work described in this chapter was presented in the following publication:

- Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency, “Refer360°: A Referring Expression Recognition Dataset in 360° Images”, In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL, pp.7189-7202, 2020.

The code for reproducing experiments in this chapter is publicly available on Github¹.

¹<https://github.com/volkancirik/refer360>

4.1 Overview

Imagine a scenario in which you are asked to retrieve medication from a bathroom. “First, face the sink, then find the second drawer in the cabinet to your left. The pills should be inside that drawer behind the toothbrush.” Interpreting instruction sequences in order to locate targets in novel

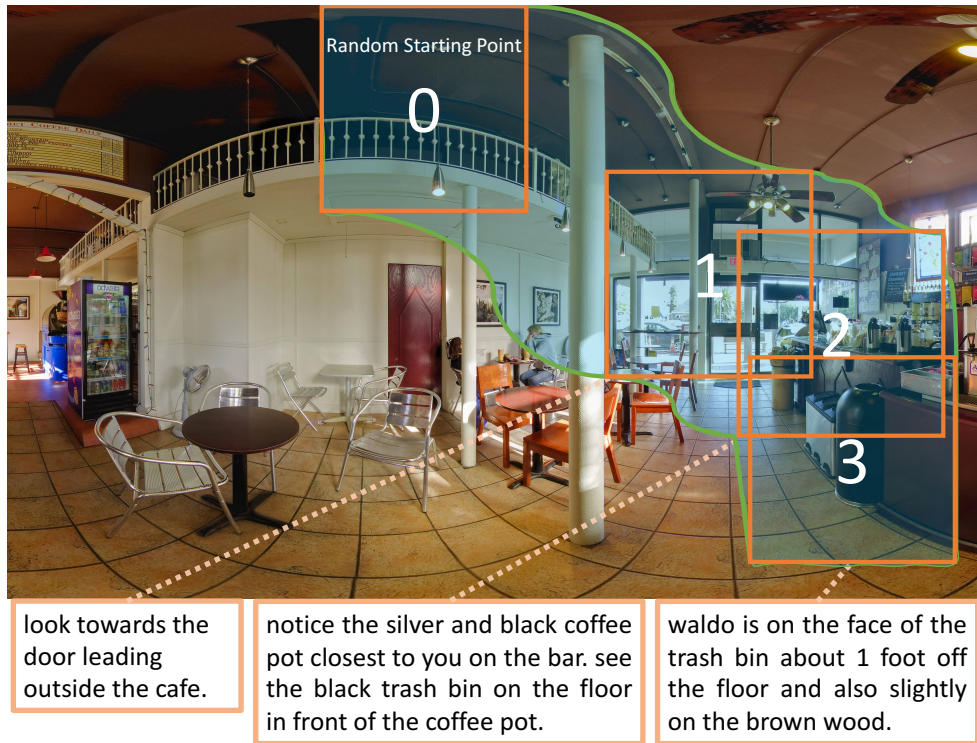


Figure 4.1: An example from Refer360°. Orange frames represent the field-of-view (FoV) of the follower after interpreting each instruction. Numbers in the frames represent the sequential order. Green lines show how FoVs change continuously. After each instruction, the follower changes the FoV to align with what the instruction describes. Please see Figure 4.2a to see the correct location of Waldo.

environments is challenging for AI systems (e.g. personal robots and self-driving cars). First, the system needs to ground the instructions into visual perception [7, 90]. This often requires identification of the mentioned object [172] through physical relationships with surrounding objects [41, 93]. Second, since human visual perception has limited field-of-view, instructions are often sequential: First, the correct FoV should be identified before searching for the final target. In many situations, the target location is not visually unique (e.g. in the middle of a plain wall), and several intermediate instructions are required. To study these challenges, we introduce a novel dataset, named Refer360°, for the task of localizing a target in 360° scenes given a sequence of instructions.

Figure 4.1 presents an example scenario from Refer360°. For this scenario, finding the target location requires first finding the door leading outside, then looking at the coffee pot, and finally

Dataset	Target Location Selection	Field of View (FoV)	Action Space	Intermediate Steps
Refer360°	Random Points	Dynamic with partial FoV	4 Directions	✓
Touchdown-SDR [30]	Human Selected Points	Oracle: Holistic & Static, 360°scenes	✗	✗
Google-Ref [146]	Annotated Objects	2D Images	✗	✗
Ref-UNC [105]	Annotated Objects	2D Images	✗	✗

Table 4.1: Comparison of referring expression datasets, including our proposed Refer360° dataset. Refer360° poses a more challenging scenario where the system observes only a partial and dynamic FoV. Refer360° also includes explicit alignments between intermediate instruction steps and human follower actions which can be used as an auxiliary evaluation metric or source of supervision.

finding the trash can, which is the nearest object to the target. Here, instructions are given from the perspective of a partial field of view (FoV) of the scene, and these FoVs can dynamically be changed. Thus, the correct interpretation of the sequence of instructions will require reasoning about what is currently visible in the FoV (e.g., grounding of objects) but also what is not visible yet. These scenarios will often require adjusting the FoV based on intermediate instructions. An important feature of the Refer360° dataset is that the target location is not an object; instead, it can be any point in the scene, which makes the grounding task more challenging since it is harder to describe a location when we cannot readily refer to it with the name of an object.

Refer360° consists of 17,137 instruction sequences with ground-truth actions to complete these instructions in 360° scenes. Refer360° has some unique characteristics which differentiate it from prior work. First, Refer360° allows the scene to be viewed through a *partial* FoV that can be *dynamically* changed as instructions are followed. This is in contrast with existing 360° scene-based datasets such as Touchdown-SDR [30] and 2D image-based referring expression datasets [97, 105, 146], where the visual input is either fixed, corresponding to a holistic, oracle-like view, or consists of fixed, cardinal FoVs. The partial and dynamic FoV in Refer360° poses new challenges for language grounding (see Figure 4.2a, 4.2b, and 4.2c for an illustrative comparison). For instance, the mentioned objects may not be visible in the current FoV, and language may refer to the FoV itself. Further, since our annotators generate instructions while observing a partial and dynamic FoV, and do so for a follower whose first FoV will be initially located at random, the instruction following task is strongly sequential. To interpret the sequence of instructions to find the target correctly, a follower must reason about the sequence of FoVs referenced by the instructor.

Second, unlike other datasets, the target locations in Refer360° are randomly distributed and thus may occur anywhere – not just on predetermined objects. As a result, target locations are less prone to bias [1, 42, 49, 67, 99]. These random locations lead to more linguistically complex instructions, as shown in our analyses – when instead annotators choose the target location, they are likely to be biased towards locations that are more easily described (e.g. on top of a named object). Table 4.1 shows a comparison of similar datasets. In the following section, we motivate Refer360° dataset in more detail.



(a) An example scene from the Refer360° dataset. Note that both annotators and systems cannot observe the shaded area. They only observe a partial field of view which can be updated dynamically.



(b) An example scene from Touchdown-SDR where the bullseye is pointing to the target location. Instructions for this instance are “a black doorway with red brick to the right of it, and green brick to the left of it. it has a light just above the doorway, and on that light is where you will find touchdown.”



(c) An example image from Google-Ref dataset with the referring expression “a young elephant nudges its head into that of a slightly taller one.”.

Figure 4.2: Examples are from (a) Refer360° (b) Touchdown-SDR, and (c) Google-Ref datasets. In Refer360° , the target location could be *any random location* on the image. In (b), annotators chose an existing object as the target location. In (c), boxes for objects were used as targets. Refer360° also seeks to increase the complexity of instruction following, making it more realistic by introducing a partial and dynamic FoV rather than providing a holistic oracle-like view of the image.

4.2 Motivation

The vision behind Refer360° is to build systems that perform localization of any point in 3D space, bringing us closer to human-like reasoning. This is an important milestone towards better collaboration between AI systems (e.g. personal robots) and humans, allowing them to act within the same space. It might also pave the way for AI-agents interacting with virtual worlds. The Refer360° dataset was designed to address three technical challenges towards this vision.

First, learning environments we create need to reflect the characteristics of human’s perception of 3D space. In such an environment, the agent only observes a partial FoV of the scene. This requires adjusting the FoV in accordance with instructions so that current view and instructions are aligned. The agent’s FoV can be changed in a continuous manner, moving smoothly left, right, up, and down. This is analogous to a real-world robot performing motor actions to change its

camera position, or a human changing their head’s pitch and yaw. Further, real scenes are 3D, but the FoV is represented in 2D in our task. Thus, interpreting some instructions will require inferences about depth.

Second, the paradigm of 360° scenes with partial FoV will almost always necessitate instructions that consist of multiple intermediate steps. As the first intermediate step, the follower and instructor need to find a common referential FoV. Then, the instructor can continue giving guidance towards the target location, often by identifying objects that are physically related to the target location. This multi-step process can serve as a natural benchmark for measuring whether systems achieve localization through a human-like process of progressively getting closer to the target location by interpreting intermediate steps. In other words, this setup may help researchers make sure that our systems are arriving at the referred location for the right reasons.

Third, since any point in the scene could be of interest, instructions will be more complex: many points in the scene will not correspond to easily named objects, and thus, when such points are allowed as targets, more sophisticated instructions will be required to unambiguously refer to them. The instructor may rely on description of physical relationships with the closest easily named locations in the scene [42, 93, 160]. For instance, in Figure 4.2a, the target location is on the side of a trash bin, which is difficult to uniquely describe with a single word or a short phrase. In this case, the instructor may use the distance to the floor or to another object in the scene in order to describe the exact location of the target. This will additionally introduce description of degree (e.g. ‘slightly above’, ‘a few inches away from’) rather than more discrete spatial relationships (e.g. ‘on top of the desk’).

4.3 Refer360° Dataset

In this section, we describe the details of the Refer360° dataset, a vision-and-language benchmark for localizing a target point in a panoramic image. Refer360° consists of 17137 instruction sequences that describe randomly distributed target locations in 2,000 panoramic scenes from the SUN360 [231] dataset. We first explain the annotation procedure for collecting and validating the instruction sequences. Later, we discuss the statistics of the Refer360° dataset.

4.3.1 Annotation Procedure

Annotation of the Refer360° dataset was carried out in three stages on Amazon Mechanical Turk with two tasks, namely a description task and a finding task. First we describe the two tasks in more detail.

Description Task. Our main goal is to collect instructions for finding any point in a 360° image. Annotators started this task looking at the ceiling of the 360° image with a random yaw². We asked them to find the target location for which we use an icon of Waldo. Target locations are chosen randomly – we discuss the details of this design choice in Section 4.3.2. The target can be at any longitude and can have a latitude within a range of 45 degrees from the top and bottom

²We wanted to avoid introducing any bias by beginning the same position each time for each scene.

of the 360 image. This restriction in latitude is made for two reasons: (1) visual distortions happen at extreme points, and (2) during the finding task, the starting point is the “ceiling” of the 360 image. Annotators were asked to give instructions to find the target location using at least three instructions. Please see Figure 4.3 to see a screenshot of the user interface we build for this task.

Finding Task. We design this task to verify the quality of instruction sequences provided by annotators in the description task. We asked annotators to complete the instruction sequences sentence by sentence. The initial field of view of annotators is always pointing at the ceiling of the 360° image with a random yaw. We asked annotators to change the FoV after each instruction so that the center of the FoV points to the location the intermediate instruction is describing. After moving the FoV to the correct position, annotators clicked a button to read the next instruction. We recorded the spherical coordinates of the center of the FoV after each instruction. As a result, our annotations include aligned intermediate steps that find the target location. After the final instruction, the annotators predicted the target location by changing the center of FoV or clicking on the FoV.

We collected and verified the quality of our data in three stages using description and finding tasks. In the first stage, we sought a pool of annotators providing high-quality annotations. For the second, aimed to collect a large number of annotations and verify their quality. In the third stage, we further verified instruction sequences that were not verified in the second stage.

Stage I. In this stage, we asked annotators to complete the finding task for four different scenes. We wrote the instruction sequences for this stage’s finding task to give annotators an example of instruction sequences for describing the target location. Then, annotators completed the description task for 4 different scenes. A total of 256 annotators participated in this first stage. We manually inspected each instruction sequences provided by these annotators for their quality of descriptions of the target location and reduced the pool of annotators to 86.

Stage II. In this stage, for each annotation session, we asked annotators first to find the target location for four different scenes, and later, describe the target location four times for different scenes³. We used the finding task to verify the quality of the instruction sequences. If an annotator predicts the target location within a radius of 11 degrees in spherical coordinates, which is roughly equal to the size of the Waldo icon we used, we counted that instance as verified.

Stage III. After the second stage, we have some instructions where the annotators could not find the target accurately. This could mean either the instructions are not clear, or it is actually harder to find the target location with these instruction sequences. In the third stage, we did another round of the finding tasks to verify these harder instruction sequences.

After these three stages, we have a total of 17137 instruction sequences in which at least one annotator was able to find the target location accurately.

³Annotators never observed their own instruction sequences while doing the finding tasks.

Annotation Stage	# of Annotators	# of Collected Instructions	# of Verified Instructions
Stage I: Hiring	256	854	n\a
Stage II: Collection & Verification	86	20630	14062
Stage III: Verification	86	n\a	3073

Table 4.2: Statistics for data collection stages. Stage I is for hiring annotators. Stage II is for collecting and verifying the instructions. Last stage is further verifying hard instances that are not verified II.

Scene Type	Scene Location	# of Images
Restaurant	Indoor	500
Shop	Indoor	250
Expo Showroom	Indoor	250
Living Room	Indoor	250
Bedroom	Indoor	250
Street	Outdoor	250
Plaza Courtyard	Outdoor	250

Table 4.3: Statistics for Panoramic Images used in Refer360° dataset.

Payment and Incentive Structure One session of annotation consisted of finding task for 4 scenes and describing task for 4 scenes which took about 15 minutes to complete on average. The base pay for one session was \$2.25. For each instruction sequence that was accurately found by another annotator, we paid a bonus of \$0.10 to both the annotator who found the location and the annotator who wrote the instruction sequence. Thus, for both the finding and describing task annotators have an interest in performing the task accurately. Next, we provide statistics of the Refer360° dataset.

4.3.2 Dataset Statistics

We split our presentation of dataset statistics into two parts: namely, scene statistics and language statistics.

Scene Statistics: To investigate the challenges in localizing a target location for both indoor and outdoor scenes as well as for different kinds of indoor and outdoor scene categories, we use seven scene categories from the SUN360 [231] dataset. We use total of 2,000 scenes. Table 4.3 shows the distribution of scene categories that comprise the Refer360° dataset.

We want to analyze the richness of the scenes in the Refer360° dataset and compare it with Touchdown-SDR. The domain of the scenes will affect the instruction one needs to use to describe a target location. To be more specific, when annotators give instructions, they use supporting objects as anchor points to help guide the attention of the follower. Thus, the availability of a rich set of objects is essential for describing the target location. Since the annotation of objects in 360° images is a laborious task itself, we use an off-the-shelf object detection method [6] to annotate scenes with objects. We split 360° images into 12 different 2D images covering the 360°

view⁴. This provides us a proxy to analyze the kind of objects usually observed in 360° images in Touchdown-SDR and Refer360° .

Dataset	Avg # of Objects	Object Type PPL
Touchdown-SDR	93.81	15.93
Refer360°	62.44	42.93

Table 4.4: Statistics for detected objects per image in Touchdown-SDR and Refer360° . On average, Refer360° images contain fewer of objects. However, these objects are from a wider variety of object types.

Table 4.4 shows the average number of objects and the perplexity of the distribution of detected objects per 360° scene used in Refer360° and Touchdown-SDR datasets. As expected, the average number of detected objects in Touchdown-SDR scenes is higher than in Refer360° because all scenes depict outdoor settings from Google’s StreetView API. However, this analysis shows that Refer360° has much larger diversity of object types and therefore will likely have greater lexical diversity in instructions.

Scenes	Dataset	Avg. Text Length	Vocab. Size	Size
360°	Refer360°	43.80	11220	17137
360°	Touchdown-SDR	26.97	5705	9325
2D	Guess What?!	24.99	27713	160745
2D	Google-Ref	8.46	12108	142210
2D	Refer-UNC	3.51	21305	414138

Table 4.5: Language statistics for Refer360° dataset and other referring expression recognition datasets.

Language Statistics: Refer360° contains a total of 17137 instruction sequences (8.57 per scene) describing target locations. Table 4.5 shows language statistics for Refer360° and other referring expression recognition datasets. Refer360° is bigger than Touchdown-SDR, yet, smaller than other datasets. This is because it is a more time-intensive and costly process to annotate and validate 360° images compared with 2D images.

Figure 4.5 shows the distribution of text length for the instructions. Compared to other referring expression recognition and image captioning datasets, Refer360° contains the longest instructions on average. This is a result of two differences with previous tasks. First, previous datasets use the entire scene as a single field of view. Thus, there is reduced need to describe how to find the target location sequentially. In Touchdown-SDR, the recognition system or human annotator needs to find an FoV that includes the target location. In Refer360° , the finding task is carried out sequentially; thus, each instruction needs to be completed accurately to be able

⁴We fixed the confidence threshold for detection of objects to 0.5 and maximum number of objects to 20.

Split	# of Instances
Train	13287
Validation Seen	900
Validation Unseen	1009
Test Seen	900
Test Unseen	1041

Table 4.6: Statistics for dataset splits in Refer360° dataset.

to find the target location. Second, in Refer360° , the target location is randomly distributed in scenes. As seen in Table 4.7, when the target location is randomly selected, the target location is on average further from other objects (we discuss this in more detail in Section 4.4.1).

Dataset Splits: We use a similar train, validation, and test split strategy as the Room-to-Room dataset [7]. We reserve a subset of images from each scene category for validation and test splits for unseen scene evaluation i.e. these scenes are not observed in the train split to study generalization capabilities of models. The remaining scenes are pooled together for training, validation, and test splits for seen scenes evaluation. Table 4.6 shows statistics for the splits. Following the previous studies, the ground-truth annotations for test splits will not be released.

4.4 Analyses

We conduct four analyses of the Refer360° dataset. First, we investigate if the random selection process of target locations can mitigate possible bias issues. Recent studies [1, 49, 67, 99] show that design decisions for collecting annotations may introduce bias into datasets. High-capacity machine learning models can exploit these issues which hinders the meaningful progress towards real language understanding [42, 243]. Second, we study whether each instruction in an instruction sequence is critical in finding the target location, or whether some instruction sequences are overcomplete. It may be very well the case that, by just understanding the last instruction, one can easily locate the target location. Third, we perform a qualitative analysis of Refer360° to provide the types of linguistic reasoning required to find the target location accurately. Finally, we analyze the performance of the state-of-the-art on Refer360° .

4.4.1 Target Locations

The selection method for the target location plays a crucial role in the kind of language one needs to use to describe that location. Earlier studies on referring expression recognition datasets [97, 105, 146, 198] select the target location as object boxes annotated by humans. In Touchdown-SDR [30] instead, annotators decide the location of the target rather than choosing one of the pre-defined lists of object boxes.

In our initial iterations for the data collection, we followed this procedure. However, we observed that in many cases, annotators chose the most salient, or unique object or region in

the image. Figure 4.6 compares the distribution of instruction sequence lengths for random and manual selection of targets.

This could introduce a location bias to the dataset – i.e. if annotators get to select the target location, they may choose targets that are easy to describe, sometimes leading to trivial or uninteresting examples, and more broadly to artificially simple language overall. For instance, if there is only one pink object in the scene, annotators usually preferred describing that region rather than some other obscure location in the scene. Instead of letting annotators decide where to place targets in the scene, we *randomly* picked a target location in the scene and asked them to describe how to find that location. As a result, our instruction sequences are complex as we show next.

Comparison	Touchdown-SDR	Refer360°
The perplexity of the distribution of an object that the target is located on	9.53	17.86
The perplexity of the distribution of the closest objects	17.80	46.84
The average distance to the closest objects	8.64	23.88

Table 4.7: Statistics for target locations image in Touchdown-SDR and Refer360° . Target is located on or near the wider variety of objects and further away from other objects.

To measure the differences in instructions for randomly or manually chosen targets, we compute three quantities. First, we compute the variety of objects that the target is located on using the perplexity of object frequencies. Similarly, we also compute the variety of objects closest to the target objects. Since we use objects near to the target location as anchor points, this is also another useful metric. The higher the perplexity of both metrics, the harder it is to predict the target location using just the object type or the closest object. Third, we measure the average distance between the target location and the nearest object. The closer the target location to another object, the easier it is to describe using the closest object as an anchor point.

Instructions	Average Distance	Accuracy
Last Sentence	73.01	0.37
Last 2 Sentences	42.32	0.63
All Sentences	11.35	0.88

Table 4.8: Results for instruction ablation human study. Annotators need all instructions to complete the task accurately.

Table 4.7 shows statistics for target locations in Touchdown-SDR and Refer360° . For both perplexity metrics, we observe that the target is located near or inside a wider variety of objects in Refer360° . Also, on average, the target location is further away from other objects for Refer360° . These statistics show that randomly choosing the target location helps us address possibly bias towards simple instructions and makes recognition more challenging.

Phenomenon	c	μ	Example from Refer360°
Coreference	96	1.6	<i>on the very upper left corner of the blue part of that window</i>
Comparison	15	0.1	<i>the smaller building to the right of the spire</i>
Sequencing	13	0.1	<i>go right just a smidge and then go up above</i>
Counting	30	0.3	<i>shaped like a football and has 3 silver legs</i>
Egocentric Spatial Mention	46	0.6	<i>find the shelves with books nearest to you</i>
Allocentric Spatial Mention	35	0.5	<i>waldo is sitting on the right side of the window</i>
Direction	92	1.6	<i>look at the knife on the wall to the left</i>
Temporal Condition	13	0.1	<i>turn right until you see a mirror on the wall</i>
3D understanding	22	0.2	<i>counter with the two bar stools sitting in front of it</i>
Inexact/Approximate Language	28	0.2	<i>in front of the white strip at the bottom slightly off center</i>
More than 2 Supporting Objects	47	0.5	<i>now look on the floor in between the table and the chair</i>

Table 4.9: Linguistic analysis of 100 randomly sampled examples from Refer360°. We annotate each example for the presence and count of each phenomenon. c is the total number of instructions out of the 100 containing at least one example of the phenomenon. μ is the mean number of times each phenomenon appears per instruction sequence.

4.4.2 Ablation of Instruction Sentences

While collecting instructions, we asked annotators to describe the target location using at least three and at most five sentences. It might be possible to find the target location using only the last instruction, which may make the first sentences unnecessary. Such redundancy makes it harder to study the core challenges of grounding instructions to visual perception and actions. Thus, we conducted an ablation study with the same pool of annotators using 1K instructions from the dataset. Here we check whether Refer360° has strong dependencies between instructions.

We ran two ablation studies to examine the necessity of using all instruction sentences. For the first study, we ran a finding task with the same pool of annotators, where we provided only the final instruction. For the second study, similarly, we ran another finding task where we provided only the penultimate and the final instruction. We compare the average euclidean distance between the predicted locations and the target location, and the accuracy, i.e. for what percentage of the time the distance between the predicted location and the target location is less than 11 degrees.

Table 4.8 shows the result of our ablation analysis. Annotators’ performance significantly dropped when they can only read the last instruction. They could find the target object only 37% of the time. Using the penultimate instruction helped them a lot, and they achieved 63% accuracy. The best performance is achieved when they observe the full instructions. These results show that each instruction is necessary for accurately finding the target location.

4.4.3 Linguistic Phenomena Observed

Before designing a system to address a language-related task, it is important to understand different kinds of linguistic phenomena observed in the task. We follow the procedure described in Touchdown-SDR [30], and added a few novel phenomena including 3D understanding, inexact language, and the use of more than two supporting objects as linguistic phenomena. Table 4.9

shows the result of our analyses for 100 randomly sampled instances. Refer360° requires reasoning for a rich set of linguistic phenomenon including the resolution of the coreference chains, counting objects, a rich set of spatial language phenomena such as multiple-supporting object mentions and 3D scene understanding.

4.4.4 Localization Experiments

Our analyses in the previous subsections suggest that Refer360° poses several challenges. In Section 4.4.1, we show that since the target locations are randomly chosen, it is harder to exploit possible location bias. In Section 4.4.2, we show that it is essential to model the sequential nature of the instructions. Section 4.4.3 shows that there are lots of interesting linguistic phenomena observed in Refer360° . We want to verify these claims by training the state-of-the-art model and measure its performance on our Refer360° dataset.

We use the same experimental setup in Touchdown-SDR using the scenes provided in the concurrent work [150], where we slice 360scene into 8 FoVs covering the scene. We pass each of these FoVs to a pre-trained model [80], and extract features from fourth to the last layer before classification to get a feature map representation of the FoVs. We concatenate 8 FoV slices to a single tensor to represent the 360° scene.

We use the LingUNet model [19, 30, 156], which performs the state-of-the-art results on TouchDown-SDR dataset. LingUNet is an image-to-image encoder-decoder model where a language and image representations are fused to predict a probability over the input image. Instructions are fed to bi-directional Long Short-Term Memory (LSTM) recurrent neural network to induce a language representation. To induce fused image-text representations, the input image tensor is passed to a convolutional neural network conditioned on the text representations. The fused representation is then fed to deconvolution layers to predict the location of the target. We use the same accuracy and distance metrics described in Section 4.4.2.

Dataset	Accuracy (%)	Distance
Touchdown-SDR (reported)	26.1	708
Touchdown-SDR (replication)	23.5	715
Refer360°	13.0	1235

Table 4.10: Results for the LingUNet on two benchmark datasets. Since LingUNet designed for observing the full instruction set and the holistic view of the scene, and it performs significantly worse on Refer360° .

As we can see in Table 4.10, LingUNet performs significantly worse on Refer360° ⁵. This might be due to the difference we highlighted in earlier sections. First and foremost, instructions must be completed sequentially. However, LingUNet does not model the sequential nature of the task for Refer360° , rather uses all instruction sequence and oracle-view of the 360° scene.

⁵We used publicly available code provided by authors to run the experiments. We could not replicate the exact numbers reported in the paper, yet, we use exactly the same setup for both Refer360° and Touchdown-SDR for a fair comparison.

Second, the scenes in Touchdown-SDR is from a single domain, but in Refer360° , we have a richer set of scenes for both indoor and outdoor.

4.5 Related Work

Referring expression recognition. Grounding a short phrase or a sentence into a visual modality such as video [4, 109] or imagery [114, 171, 172, 238] is a well studied problem in intelligent user interfaces [23], human-robot interaction [24, 55, 227], and situated dialogue [107]. Kazemzadeh et al. [105], Hu et al. [89], and Mao et al. [146] introduce two benchmark datasets for the real-world 2D images. Nagaraja et al. [160] propose a model where the target and supporting objects (i.e. objects that are mentioned in order to disambiguate the target object) are identified and scored jointly. Hu et al. [93] introduce a compositional approach where they assume that the referring expression can be decomposed into a triplet consisting of the target object, the supporting object, and their spatial relationship. Similarly, Cirik et al. [41] propose a type of neural modular network [12] where the grounding of referring expression depends on the parse tree of the input referring expression to learn to ground an unconstrained number of supporting objects.

360° Scenes. Although 360° scenes are well studied in the computer vision domain [200, 226, 231, 232, 233, 236, 241], few studies explore the challenges of 360°scenes in the context of language grounding. Chou et al. [37] introduce a dataset where 360° videos are narrated. They address the task of predicting the field of view for the given narration. Anderson et al. [7] introduce the vision and language navigation task for simulated indoor environments where an agent is placed in a location in a house and follows the instructions to go to a target location. Here the agent observes a discretized view of the current location (i.e. the 360° scene is split into a fixed number of field of views). The most related work to Refer360° is Touchdown [30] which introduces two tasks: a vision and language navigation task and a spatial description resolution (SDR) task (i.e. a referring expression recognition task for a simulated outdoor environment). In contrast with Touchdown, in our setup instructors, followers, and learning systems observe a partial FoV of the scene, but they can change the FoV continuously to explore the scene. This approach yields instructions with a stronger sequential dependencies and with stronger reference to the FoV itself. We demonstrate some of these differences in analysis in Section 4.4. Concurrent work studies visual question answering [36] and object detection [38] for 360°scenes. Another concurrent study [175] combines vision-and-language navigation and referring expression recognition into one task where the system is asked to localize the referred object after navigating to another point in a real images of rendered buildings.

4.6 Conclusion

We designed Refer360° to study 3D spatial language understanding for real scenes. We collected a fine-grained set of annotations that support study at many levels of language grounding. Refer360° is a versatile dataset and enables investigation along three axes:

- **Language:** Refer360° enables modeling tasks that study single instruction, multiple instructions, or interactive language where the next instruction is revealed only after reaching an intermediate milestone.
- **Vision:** Refer360° enables modeling tasks that try to predict targets at different granularities: at the object level if trying to identify the closest object to the target, at the region level in a similar style to Touchdown-SDR, and finally, at the pixel level.
- **Action:** Refer360° enables modeling tasks where the action space is static with the whole 360 image given upfront, where the action space consists of a sequence of discrete choices between fixed views, and when the action space is continuous, consisting of angles for rotation.

In our experiments, we presented one of these scenarios (single instruction, static, and pixel-level) since it was the closest to the pre-existing Touchdown-SDR system. However, one can also study a much larger number of scenarios and modeling tasks using Refer360° .

Acknowledgements

We are thankful to anonymous ACL conference reviewers for providing valuable feedback. We thank members of MultiComp Lab at CMU and Berg Lab at UCSD for useful discussions. We thank Howard Chen for helping us replicate their experiments. This material is partially supported by Siemens and the National Science Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of Siemens or the National Science Foundation, and no official endorsement should be inferred.

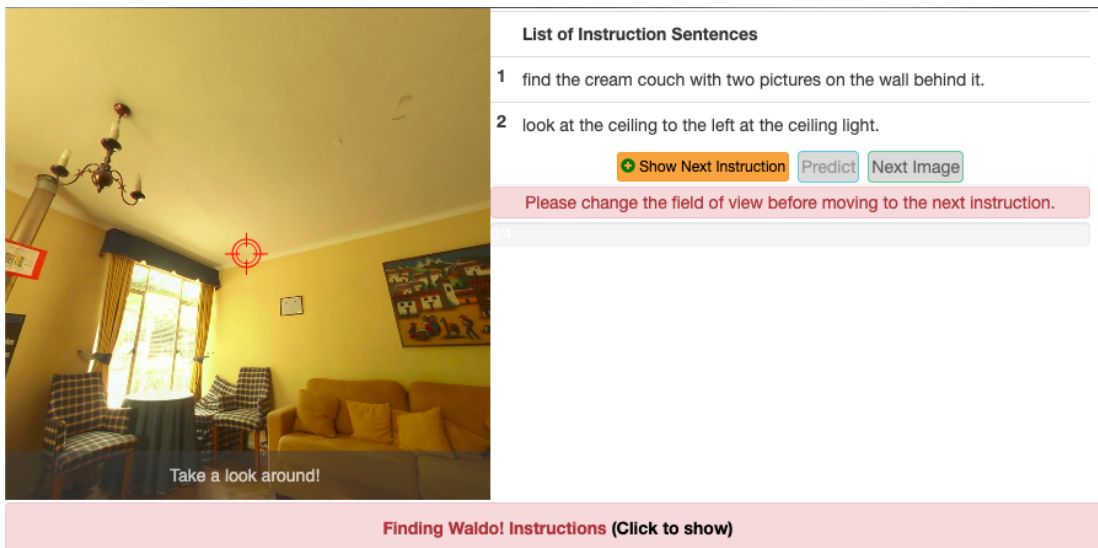


Figure 4.3: Screenshot of Amazon Mechanical Turk interface for finding task. We ask annotators to complete each instruction before moving to the next one. To do so change the bullseye where they think the instruction is describing.

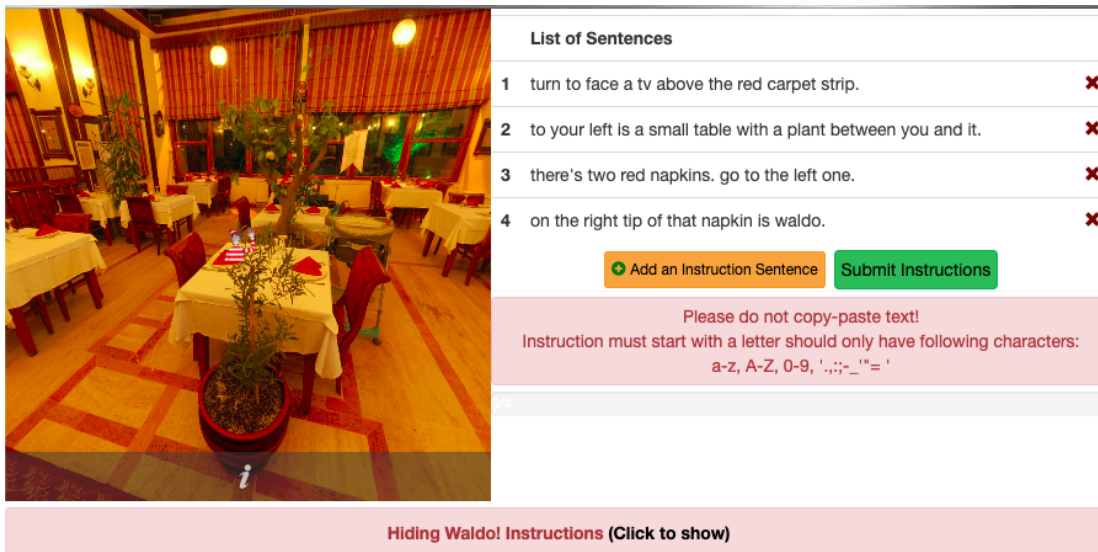


Figure 4.4: Screenshot of Amazon Mechanical Turk interface for describing task. We ask annotators to first find Waldo themselves, then give detailed instructions one by one so that anyone starting from a random field-of-view find it.

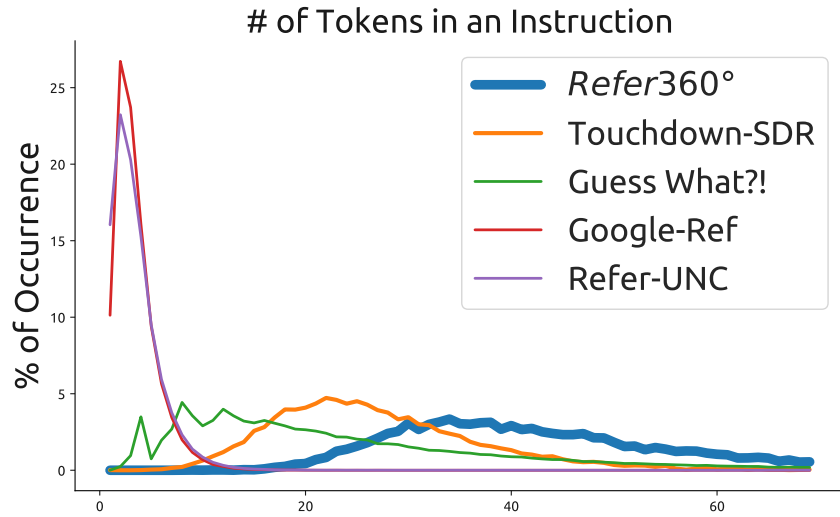


Figure 4.5: Distribution of the number of tokens for vision-and-language datasets similar to Refer360°

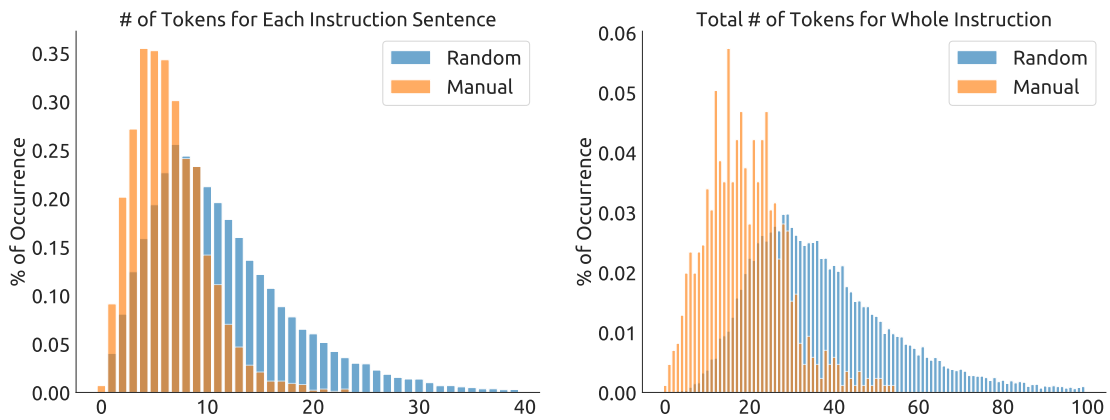


Figure 4.6: Text length for different placement methods for single instruction and instruction sequences. Manual means annotators pick the target location, random means we randomly pick the target location in the scene.

Chapter 5

Reasoning About Alternatives for Vision-and-Language Navigation

In our reasonings concerning matter of fact, there are all imaginable degrees of assurance, from the highest certainty to the lowest species of moral evidence. A wise man, therefore, proportions his belief to the evidence.

David Hume

In a partially observed world, natural language instructions only identify few coarse-grained actions (e.g., go to kitchen take a left after the table) rather than providing fine-grained actions (e.g., turn 37° to the left and move 3.21 meters). Thus, there could be many interpretations of the given instruction, but the human user intends only a few. An embodied AI system then needs to reason about its actions and their possible consequences. In this chapter, we study this challenging reasoning problem. The work described in this chapter appeared in the following publication:

- Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, Trevor Darrell, , “Speaker-Follower Models for Vision-and-Language Navigation”, In Advances in Neural Information Processing System, Curran Associates, Inc., volume 31, 2018.

This is a equal contribution publication where DF, RH, VC led and worked on developing ideas, running experiments, and writing the manuscript. The code for reproducing experiments in this chapter is publicly available on Github¹.

¹https://github.com/ronghanghu/speaker_follower

5.1 Overview

In the vision-and-language navigation task [8], an agent is placed in a realistic environment, and provided a natural language instruction such as “*Go down the stairs, go slight left at the bottom and go through door, take an immediate left and enter the bathroom, stop just inside in front of the sink*”. The agent must follow this instruction to navigate from its starting location to a goal location, as shown in Figure 5.1 (left). To accomplish this task the agent must learn to relate the language instructions to the visual environment. Moreover, it should be able to carry out new instructions in unseen environments.

Even simple navigation tasks require nontrivial *reasoning*: the agent must resolve ambiguous references to landmarks, perform a counterfactual evaluation of alternative routes, and identify incompletely specified destinations. While a number of approaches [151, 157, 223] have been proposed for the various navigation benchmarks, they generally employ a single model that learns to map directly from instructions to actions from a limited corpus of annotated trajectories.

In this chapter we treat the vision-and-language navigation task as a trajectory search problem, where the agent needs to find (based on the instruction) the best trajectory in the environment to navigate from the start location to the goal location. Our model involves an instruction interpretation (*follower*) module, mapping instructions to action sequences; and an instruction generation (*speaker*) module, mapping action sequences to instructions (Figure 5.1), both implemented with standard sequence-to-sequence architectures. The speaker learns to give textual instructions for visual routes, while the follower learns to follow routes (predict navigation actions) for provided textual instructions. Though explicit probabilistic reasoning combining speaker and follower agents is a staple of the literature on computational pragmatics [56], application of these models has largely been limited to extremely simple decision-making tasks like single forced choices.

We incorporate the speaker both at training time and at test time, where it works together with the learned instruction follower model to solve the navigation task (see Figure 5.2 for an overview of our approach). At training time, we perform speaker-driven data augmentation where the speaker helps the follower by synthesizing additional route-instruction pairs to expand the limited training data. At test time, the follower improves its chances of success by looking ahead at possible future routes and pragmatically choosing the best route by scoring them according to the probability that the speaker would generate the correct instruction for each route. This procedure, using the external speaker model, improves upon planning using only the follower model. We construct both the speaker and the follower on top of a panoramic action space that efficiently encodes high-level behavior, moving directly between adjacent locations rather than making low-level visuomotor decisions like the number of degrees to rotate (see Figure 5.3).

To summarize our contributions: We propose a novel approach to vision-and-language navigation incorporating a visually grounded speaker–follower model, and introduce a panoramic representation to efficiently represent high-level actions. We evaluate this speaker–follower model on the Room-to-Room (R2R) dataset [8], and show that each component in our model improves performance at the instruction following task. Our model obtains a final success rate of 53.5% on the unseen test environment, an absolute improvement of 30% over existing approaches. Our code and data are available at http://ronghanghu.com/speaker_follower.



Figure 5.1: The task of vision-and-language navigation is to perform a sequence of actions (navigate through the environment) according to human natural language instructions. Our approach consists of an instruction *follower* model (left) and a *speaker* model (right).

5.2 Instruction Following with Speaker-Follower Models

To address the task of following natural language instructions, we rely on two models: an instruction-*follower* model of the kind considered in previous work, and a *speaker* model—a learned instruction generator that models how humans describe routes in navigation tasks.

Specifically, we base our follower model on the sequence-to-sequence model [8], computing a distribution $P_F(r | d)$ over routes r (state and action sequences) given route descriptions d . The follower encodes the sequence of words in the route description with an LSTM [84], and outputs route actions sequentially, using an attention mechanism [16] over the description. Our speaker model is symmetric, producing a distribution $P_S(d | r)$ by encoding the sequence of visual observations and actions in the route using an LSTM, and then outputting an instruction word-by-word with an LSTM decoder using attention over the encoded input route (Figure 5.1).

We combine these two base models into a speaker-follower system, where the speaker supports the follower both at training time and at test time. An overview of our approach is presented in Figure 5.2. First, we train a speaker model on the available ground-truth navigation routes and instructions. (Figure 5.2 (a)). Before training the follower, the speaker produces synthetic navigation instructions for novel sampled routes in the training environments, which are then used as additional supervision for the follower, as described in Sec. 5.2.1 (Figure 5.2 (b)). At follower test time, the follower generates possible routes as interpretations of a given instruction and starting context, and the speaker pragmatically ranks these, choosing one that provides a good explanation of the instruction in context (Sec. 5.2.2 and Figure 5.2 (c)). Both follower and speaker are supported by the panoramic action space in Sec. 5.2.3 that reflects the high-level granularity of the navigational instructions (Figure 5.3).

5.2.1 Speaker-Driven Data Augmentation

The training data only covers a limited number of navigation instruction and route pairs, $\mathcal{D} = (d_1, r_1) \dots (d_N, r_N)$. To allow the agent to generalize better to new routes, we use the speaker to generate synthetic instructions on sampled new routes in the training environments. To create a synthetic training set, we sample a collection of M routes $\hat{r}_1, \dots, \hat{r}_M$ through the training environments, using the same shortest-path approach used to generate the routes in the original training set [8]. We then generate a human-like textual instruction \hat{d}_k for each instruction \hat{r}_k by

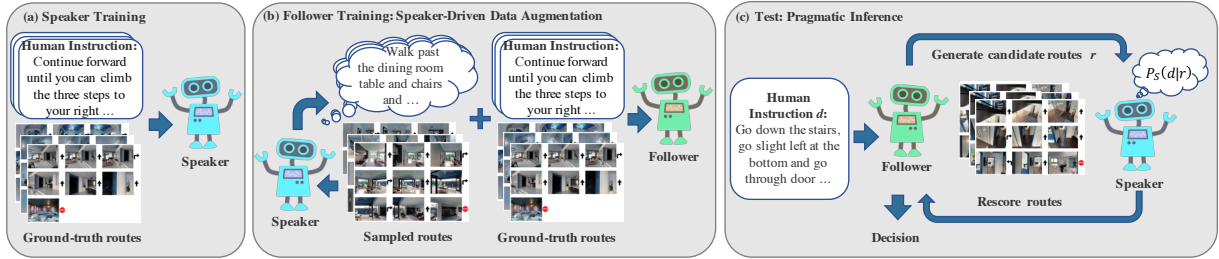


Figure 5.2: Our approach combines an instruction *follower* model and a *speaker* model. (a) The speaker model is trained on the ground-truth routes with human-generated descriptions; (b) it provides the follower with additional synthetic instruction data to bootstrap training; (c) it also helps the follower interpret ambiguous instructions and choose the best route during inference. See Sec. 5.2 for details.

performing greedy prediction in the speaker model to approximate $\hat{d}_k =_d P_S(d | \hat{r}_k)$.

These M synthetic navigation routes and instructions $\mathcal{S} = (\hat{d}_1, \hat{r}_1), \dots, (\hat{d}_M, \hat{r}_M)$ are combined with the original training data \mathcal{D} into an augmented training set $\mathcal{S} \cup \mathcal{D}$ (Figure 5.2(b)). During training, the follower is first trained on this augmented training set, and then further fine-tuned on the original training set \mathcal{D} . This speaker-driven data augmentation aims to overcome data scarcity issue, allowing the follower to learn how to navigate on new routes following the synthetic instructions.

5.2.2 Speaker-Driven Route Selection

We use the base speaker (P_S) and follower (P_F) models described above to define a *pragmatic follower* model. Drawing on the Rational Speech Acts framework [56, 66], a pragmatic follower model should choose a route r through the environment that has high probability of having caused the speaker model to produce the given description d : ${}_r P_S(d | r)$ (corresponding to a rational Bayesian follower with a uniform prior over routes). Such a follower chooses a route that provides a good explanation of the observed description, allowing counterfactual reasoning about instructions, or using global context to correct errors in the follower’s path, which we call *pragmatic inference*.

Given the sequence-to-sequence models that we use, exactly solving the maximization problem above is infeasible; and may not even be desirable, as these models are trained discriminatively and may be poorly calibrated for inputs dissimilar to those seen during training. Following previous work on pragmatic language generation and interpretation [10, 58, 158, 196], we use a rescoring procedure: produce candidate route interpretations for a given instruction using the base follower model, and then rescore these routes using the base speaker model (Figure 5.2(c)).

Our pragmatic follower produces a route for a given instruction by obtaining K candidate paths from the base follower using a search procedure described below, then chooses the highest scoring path under a combination of the follower and speaker model probabilities:

$$r \in R(d) P_S(d | r)^\lambda \cdot P_F(r | d)^{(1-\lambda)} \quad (5.1)$$

where λ is a hyper-parameter in the range $[0, 1]$ which we tune on validation data to maximize the

accuracy of the follower.²

Candidate route generation To generate candidate routes from the base follower model, we perform a search procedure where candidate routes are produced incrementally, action-by-action, and scored using the probabilities given by P_F . Standard beam search in sequence-to-sequence models (e.g. [205]) forces partial routes to compete based on the number of actions taken. We obtain better performance by instead using a *state-factored* search procedure, where partial output sequences compete at the level of states in the environment, where each state consists of the agent’s location and discretized heading, keeping only the highest-scoring path found so far to each state. At a high-level, this search procedure resembles graph search with a closed list, but since action probabilities are non-stationary (potentially depend on the entire sequence of actions taken in the route), it is only approximate, and so we allow re-expanding states if a higher-scoring route to that state is found.

At each point in our state-factored search for searching and generating candidates in the follower model, we store the highest-probability route (as scored by the follower model) found so far to each state. States contain the follower’s discrete location and heading (direction it is facing) in the environment, and whether the route has been completed (had the STOP action predicted). The highest-scoring route, which has not yet been expanded (had successors produced), is selected and expanded using each possible action from the state, producing routes to the neighboring states. For each of these routes r with final state s , if s has not yet been reached by the search, or if r is higher-scoring under the model than the current best path to s , r is stored as the best route to s . We continue the search procedure until K routes ending in distinct states have predicted the STOP action, or there are no remaining unexpanded routes.

Since route scores are products of conditional probabilities, route scores are non-increasing, and so this search procedure generates routes that do not pass through the same state twice—which we found to improve accuracy both for the base follower model and the pragmatic rescoreing procedure, since instructions typically describe acyclic routes.

We generate up to $K = 40$ candidate routes for each instruction using this procedure, and rescore using Eq. 5.1. In addition to enabling pragmatic inference, this state-factored search procedure improves the performance of the follower model on its own (taking the candidate route with highest score under the follower model), when compared to standard greedy search (see Fig 5.4).

5.2.3 Panoramic Action Space

The sequence-to-sequence agent in [8] uses low-level visuomotor control (such as turning left or right by 30 degrees), and only perceives frontal visual sensory input. Such fine-grained visuomotor control and restricted visual signal introduce challenges for instruction following. For example in Figure 5.3, to “turn left and go towards the sofa”, the agent needs to perform a series of turning

²In practice, we found best performance with values of λ close to 1, relying mostly on the score of the speaker to select routes. Using only the speaker score (which corresponds to the standard RSA pragmatic follower) did not substantially reduce performance compared to using a combination with the follower score, and both improved substantially upon using only the follower score (corresponding to the base follower).

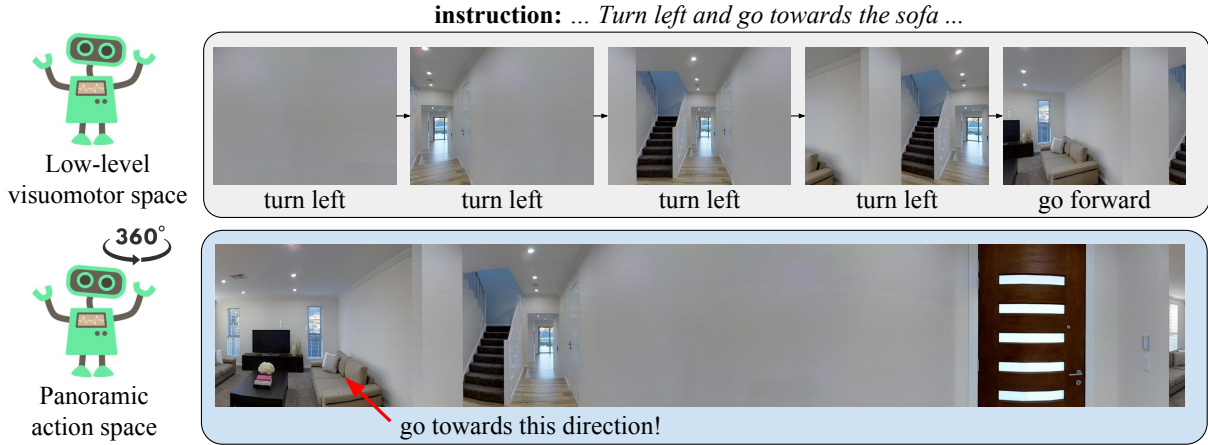


Figure 5.3: Compared with low-level visuomotor space, our panoramic action space (Sec. 5.2.3) allows the agents to have a complete perception of the scene, and to directly perform high-level actions.

actions until it sees a sofa in the center of its view, and then perform a “go forward” action. This requires strong skills of planning and memorization of visual inputs. While a possible way to address this challenge is to learn a hierarchical policy such as in [45], in our work we directly allow the agent to reason about high-level actions, using a panoramic action space with panoramic representation, converted with built-in mapping from low-level visuomotor control.

As shown in Figure 5.3, in our panoramic representation, the agent first “looks around” and perceives a 360-degree panoramic view of its surrounding scene from its current location, which is discretized into 36 view angles (12 headings \times 3 elevations with 30 degree intervals – in our implementation). Each view angle i is represented by an encoding vector v_i . At each location, the agent can only move towards a few navigable directions (provided by the navigation environment) as other directions can be physically obstructed (e.g. blocked by a table). Here, in our action space the agent only needs to make high-level decisions as to which navigable direction to go to next, with each navigable direction j represented by an encoding vector u_j . The encoding vectors v_i and u_j of each view angle and navigable direction are obtained by concatenating an appearance feature (ConvNet feature extracted from the local image patch around that view angle or direction) and a 4-dimensional orientation feature $[\sin \psi; \cos \psi; \sin \theta; \cos \theta]$, where ψ and θ are the heading and elevation angles respectively. Also, we introduce a STOP action encoded by $u_0 = \vec{0}$. The agent can take this STOP action when it decides it has reached the goal location (to end the episode).

To make a decision on which direction to go, the agent first performs one-hop visual attention to look at all of the surrounding view angles, based on its memory vector h_{t-1} . The attention weight $\alpha_{t,i}$ of each view angle i is computed as $a_{t,i} = (W_1 h_{t-1})^T W_2 v_{t,i}$ and $\alpha_{t,i} = \exp(a_{t,i}) / \sum_i \exp(a_{t,i})$.

The attended feature representation $v_{t,att} = \sum_i \alpha_{t,i} v_{t,i}$ from the panoramic scene is then used as visual-sensory input to the sequence-to-sequence model (replacing the 60-degree frontal appearance vector in [8]) to update the agent’s memory. Then, a bilinear dot product is used to obtain the probability p_j of each navigable direction j : $y_j = (W_3 h_t)^T W_4 u_j$ and $p_j = \exp(y_j) / \sum_j \exp(y_j)$.

The agent then chooses a navigable direction u_j (with probability p_j) to go to the adjacent location along that direction (or u_0 to stop and end the episode). We use a built-in mapping that seamlessly translates our panoramic perception and action into visuomotor control such as turning and moving.

5.3 Experiments

5.3.1 Experimental Setup

Dataset We use the Room-to-Room (R2R) vision-and-language navigation dataset [8] for our experimental evaluation. In this task, the agent starts at a certain location in an environment and is provided with a human-generated navigation instruction, that describes a path to a goal location. The agent needs to follow the instruction by taking multiple discrete actions (e.g. turning, moving) to navigate to the goal location, and executing a “stop” action to end the episode. Note that differently from some robotic navigation settings [168], here the agent is not provided with a goal image, but must identify from the textual description and environment whether it has reached the goal.

The dataset consists of 7,189 paths sampled from the Matterport3D [25] navigation graphs, where each path consists of 5 to 7 discrete viewpoints and the average physical path length is 10m. Each path has three instructions written by humans, giving 21.5k instructions in total, with an average of 29 words per instruction. The dataset is split into training, validation, and test sets. The validation set is split into two parts: *seen*, where routes are sampled from environments seen during training, and *unseen* with environments that are not seen during training. All the test set routes belong to new environments unseen in the training and validation sets.

Evaluation metrics Following previous work on the R2R task, our primary evaluation metrics are navigation error (NE), measuring the average distance between the end-location predicted by the follower agent and the true route’s end-location, and success rate (SR), the percentage of predicted end-locations within 3m of the true location. As in previous work, we also report the oracle success rate (OSR), measuring success rate at the closest point to the goal that the follower has visited along the route, allowing the agent to overshoot the goal without being penalized.

Implementation details Following [8] and [223], we produce visual feature vectors v using the output from the final convolutional layer of a ResNet [80] trained on the ImageNet [186] classification dataset. These visual features are fixed, and the ResNet is not updated during training. To better generalize to novel words in the vocabulary, we also experiment with using GloVe embeddings [169], to initialize the word-embedding vectors in the speaker and follower.

In the baseline without using synthetic instructions, we train follower and speaker models using the human-generated instructions for routes present in the training set. The training procedure for the follower model follows [8] by training with *student-forcing* (sampling actions from the model during training, and supervising using a shortest-path action to reach the goal state). We use the training split in the R2R dataset to train our speaker model, using standard maximum likelihood training with a cross-entropy loss.

#	Data	Pragmatic	Panoramic	Validation-Seen			Validation-Unseen		
	Augmentation	Inference	Space	NE ↓	SR ↑	OSR ↑	NE ↓	SR ↑	OSR ↑
1				6.08	40.3	51.6	7.90	19.9	26.1
2	✓			5.05	46.8	59.9	7.30	24.6	33.2
3		✓		5.23	51.5	60.8	6.62	34.5	43.1
4			✓	4.86	52.1	63.3	7.07	31.2	41.3
5	✓	✓		4.28	57.2	63.9	5.75	39.3	47.0
6	✓		✓	3.36	66.4	73.8	6.62	35.5	45.0
7		✓	✓	3.88	63.3	71.0	5.24	49.5	63.4
8	✓	✓	✓	3.08	70.1	78.3	4.83	54.6	65.2

Table 5.1: Ablations showing the effect of each component in our model. Rows 2-4 show the effects of adding a single component to the baseline system (Row 1); Rows 5-7 show the effects of removing a single component from the full system (Row 8). NE is navigation error (in meters); lower is better. SR and OSR are success rate and oracle success rate (%); higher is better. See Sec. 5.3.2 for details.

In speaker-driven data augmentation (Sec. 5.2.1), we augment the data used to train the follower model by sampling 178,000 routes from the training environments. Instructions for these routes are generated using greedy inference in the speaker model (which is trained only on human-produced instructions). The follower model is trained using student-forcing on this augmented data for 50,000 iterations, and then fine-tuned on the original human-produced data for 20,000 iterations. For all experiments using pragmatic inference, we use a speaker weight of $\lambda = 0.95$, which we found to produce the best results on both the seen and unseen validation environments.

5.3.2 Results and Analysis

We first examine the contribution from each of our model’s components on the validation splits. Then, we compare the performance of our model with previous work on the unseen test split.

Component Contributions

We begin with a baseline (Row 1 of Table 5.1), which uses only a follower model with a non-panoramic action space at both training and test time, which is equivalent to the student-forcing model in [8].³

Speaker-driven data augmentation We first introduce the speaker at training time for data augmentation (Sec. 5.2.1). Comparing Row 1 (the baseline follower model trained only with the

³Note that our results for this baseline are slightly higher on val-seen and slightly lower on val-unseen than those reported by [8], due to differences in implementation details and hyper-parameter choices.

original training data) against Row 2 (training this model on augmented data) in Table 5.1, we see that adding the augmented data improves success rate (SR) from 40.3% to 46.8% on validation seen and from 19.9% to 24.6% on validation unseen, respectively. This higher relative gain on unseen environments shows that the follower can learn from the speaker-annotated routes to better generalize to new scenes.

Note that given the noise in our augmented data, we fine-tune our model on *the original training data* at the end of training as mentioned in Sec. 5.2.1. We find that increasing the amount of augmented data is helpful in general. For example, when using 25% of the augmented data, the success rate improves to 22.8% on validation unseen, while with all the augmented data the success rate reaches 24.6% on validation unseen, which is a good balance between performance and computation overhead.

Pragmatic inference We then incorporate the speaker at test time for pragmatic inference (Sec. 5.2.2), using the speaker to rescore the route candidates produced by the follower. Adding this technique brings a further improvement in success rate on both environments (compare Row 2, the data-augmented follower without pragmatic inference, to Row 5, adding pragmatic inference). This shows that when reasoning about navigational directions, large improvements in accuracy can be obtained by scoring how well the route explains the direction using a speaker model. Importantly, when using only the follower model to score candidates produced in search, the success rate is 49.0% on val-seen and 30.5% on val-unseen, showing the importance of using the speaker model to choose among candidates (which increases success rates to 57.2% and 39.3%, respectively).

In Figure 5.4, we also investigate the effect of beam size K and compare our state factored and greedy search. The success rate increases but saturates when we increase the number of candidate routes. Also, our state-factored search (indicated in triangle points) show superior performance compared to standard greedy search (indicated in star points).

Panoramic action space Finally, we replace the visuomotor control space with the panoramic representation (Sec. 5.2.3). Adding this to the previous system (compare Row 5 and Row 8) shows that the new representation leads to a substantially higher success rate, achieving 70.1% and 54.6% success rate on validation seen and validation unseen, respectively. This suggests that directly acting in a higher-level representation space makes it easier to accurately carry out instructions. Our final model (Row 8) has over twice the success rate of the baseline follower in the unseen environments.

Importance of all components Above we have shown the gain from each component, after being added incrementally. Moreover, comparing Rows 2-4 (adding each component independently to the base model) to the baseline (Row 1) shows that each component in isolation provides large improvements in success rates, and decreases the navigation error. Ablating each component (Rows 5-7) from the full model (Row 8) shows that each of them is important for the final performance.

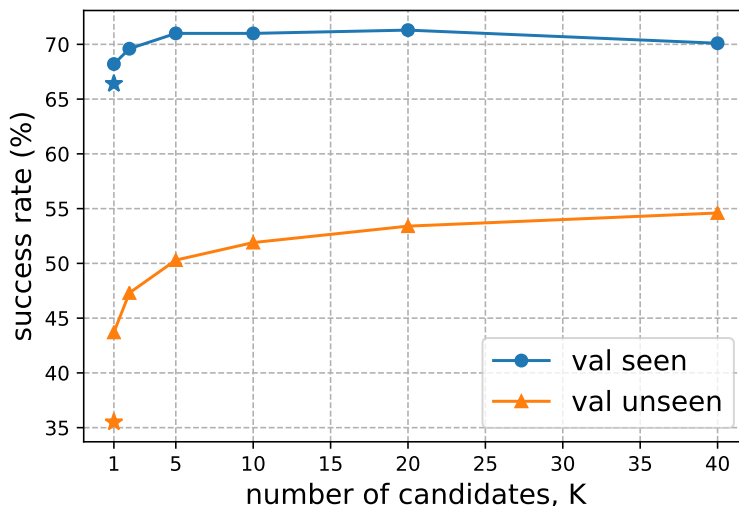


Figure 5.4: The success rate of our model using different numbers K of route candidates (generated by state-factored search) for pragmatic inference. Stars show the performance of greedy inference (without search, and hence without pragmatics). While performance increases with number of candidates up through 40 on val unseen, the success rate tends to saturate. We note improvements both from the state-factored search procedure (comparing the stars to the circle and triangle points at $K = 1$) as well as from having more candidates to choose from in pragmatic inference (comparing larger values of K to smaller).

Qualitative results Here we provide qualitative examples further explaining how our model improves over the baseline. The intuition behind the speaker model is that it should help the agent more accurately interpret instructions specifically in ambiguous situations. Figure 5.14 shows how the introduction of a speaker model helps the follower with pragmatic inference.

Figure 5.5 v.s. 5.6 show the step-wise navigation trajectory of the base follower (without pragmatic inference) and the follower model with pragmatic inference, on the val seen split. Figure 5.7 v.s. Figure 5.8 and Figure 5.9 v.s. Figure 5.10 show the trajectory of the agent without and with pragmatic inference (using the speaker model) on the val unseen split. The speaker helps disambiguate vague instructions by globally measuring how likely a route can be described by the instruction.

We also visualize the image attention (attention weights $\alpha_{t,i}$ of each view angle i in our panoramic action space in Section 5.2.3), and the textual attention on the input instructions from the sequence-to-sequence model in Figures 5.11, 5.12 and 5.13.

Comparison to Prior Work

We compare the performance of our final model to previous approaches on the R2R held-out splits, including the test split which contains 18 new environments that do not overlap with any training or validation splits, and are only seen once at test time.

The results are shown in Table 5.2. In the table, “Random” is randomly picking a direction and

Method	Validation-Seen			Validation-Unseen			Test (unseen)			
	NE ↓	SR ↑	OSR ↑	NE ↓	SR ↑	OSR ↑	NE ↓	SR ↑	OSR ↑	TL ↓
Random	9.45	15.9	21.4	9.23	16.3	22.0	9.77	13.2	18.3	9.89
Student-forcing [8]	6.01	38.6	52.9	7.81	21.8	28.4	7.85	20.4	26.6	8.13
RPA [223]	5.56	42.9	52.6	7.65	24.6	31.8	7.53	25.3	32.5	9.15
ours	3.08	70.1	78.3	4.83	54.6	65.2	4.87	53.5	63.9	11.63
ours (challenge participation)*	–	–	–	–	–	–	4.87	53.5	96.0	1257.38
Human	–	–	–	–	–	–	1.61	86.4	90.2	11.90

Table 5.2: Performance comparison of our method to previous work. NE is navigation error (in meters); lower is better. SR and OSR are success rate and oracle success rate (%) respectively (higher is better). Trajectory length (TL) on the test set is reported for completeness. *: *When submitting to the Vision-and-Language Navigation Challenge, we modified our search procedure to maintain physical plausibility and to comply with the challenge guidelines. The resulting trajectory has higher oracle success rate while being very long.*

going towards that direction for 5 steps. “Student-forcing” is the best performing method in [8], using exploration during training of the sequence-to-sequence follower model. “RPA” [223] is a combination of model-based and model-free reinforcement learning (see also Sec. 5.4 for details). “ours” shows our performance using the route selected by our pragmatic inference procedure, while “ours (challenge participation)” uses a modified inference procedure for submission to the Vision-and-Language Navigation Challenge. Prior work has reported higher performance on the seen rather than unseen environments [8, 223], illustrating the issue of generalizing to new environments. Our method more than doubles the success rate of the state-of-the-art RPA approach, and on the test set achieves a final success rate of 53.5%. This represents a large reduction in the gap between machine and human performance on this task.

5.4 Related Work

Natural language instruction following Systems that learn to carry out natural language instructions in an interactive environment include approaches based on intermediate structured and executable representations of language [14, 29, 73, 137, 210] and approaches that map directly from language and world state observations to actions [9, 21, 151, 157]. The embodied vision-and-language navigation task studied in this chapter [8] differs from past situated instruction following tasks by introducing rich visual contexts. Recent work [223] has applied techniques from model-based and model-free reinforcement learning [225] to the vision-and-language navigation problem. Specifically, an environment model is used to predict a representation of the state resulting from an action, and planning is performed with respect to this environment model. Our work differs from this prior work by reasoning not just about state transitions, but also about the relationship between states and the language that describes them—specifically, using an external speaker model to predict how well a given sequence of states explains an instruction.

Pragmatic language understanding A long line of work in linguistics, natural language processing, and cognitive science has studied *pragmatics*: how linguistic meaning is affected

by context and communicative goals [69]. Our work here makes use of the Rational Speech Acts framework [56, 66], which models the interaction between speakers and listeners as a process where each agent reasons probabilistically about the other to maximize the chances of successful communicative outcomes. This framework has been applied to model human use of language [57], and to improve the performance of systems that generate [10, 43, 146, 216] and interpret [142, 214, 240] referential language. Similar modeling tools have recently been applied to generation and interpretation of language about sequential decision-making [58]. The present work makes use of a pragmatic instruction follower in the same spirit. Here, however, we integrate this with a more complex visual pipeline and use it not only at inference time but also at *training* time to improve the quality of a base listener model.

Semi- and self-supervision The semi-supervised approach we use is related to model bootstrapping techniques such as self-training [149, 190] and co-training [20] at a high-level. Recent work has used monolingual corpora to improve the performance of neural machine translation models structurally similar to the sequence-to-sequence models we use [71, 80, 191]. In a grounded navigation context, [81] use a word-prediction task as training time supervision for a reinforcement learning agent. The approach most relevant to our work is the SEQ4 model [116], which applies semi-supervision to a navigation task by sampling new environments and maps (in synthetic domains without vision), and training an autoencoder to reconstruct routes, using language as a latent variable. The approach used here is much simpler, as it does not require constructing a differentiable surrogate to the decoding objective.

Semi-supervised data augmentation has also been widely used in computer vision tasks. In Data Distillation [179], additional annotation for object and key-point detection is obtained by ensembling and refining a pretrained model’s prediction on unannotated images. Self-play in adversarial groups of agents is common in multi-agent reinforcement learning [195, 201]. In actor-critic approaches [206, 207] in reinforcement learning, a critic learns the value of a state and is used to provide supervision to the actor’s policy during training. In this chapter, we use a speaker to synthesize additional navigation instructions on unlabeled new routes, and use this synthetic data from the speaker to train the follower.

Grounding language in vision Existing work in visual grounding [98, 146, 161, 170, 184] has addressed the problem of *passively* perceiving a static image and mapping a referential expression to a bounding box [98, 146, 170] or a segmentation mask [95, 128, 238], exploring various techniques including proposal generation [31] and relationship handling [41, 94, 161, 221]. In our work, the vision-and-language navigation task requires the agent to *actively* interact with the environment to find a path to the goal following the natural language instruction. This can be seen as a grounding problem in linguistics where the language instruction is grounded into a trajectory in the environment but requires more reasoning and planning skills than referential expression grounding.

5.5 Conclusion

The language-and-vision navigation task presents a pair of challenging reasoning problems: in language, because agents must interpret instructions in a changing environmental context; and in vision, because of the tight coupling between local perception and long-term decision-making. The comparatively poor performance of the baseline sequence-to-sequence model for instruction following suggests that more powerful modeling tools are needed to meet these challenges. In this chapter, we have introduced such a tool, showing that a follower model for vision-and-language navigation is substantially improved by carefully structuring the action space and integrating an explicit model of a *speaker* that predicts how navigation routes are described. We believe that these results point toward further opportunities for improvements in instruction following by modeling the global structure of navigation behaviors and the pragmatic contexts in which they occur.

Acknowledgements

This work was partially supported by US DoD and DARPA XAI and D3M, NSF awards IIS-1833355, Oculus VR, and the Berkeley Artificial Intelligence Research (BAIR) Lab. DF was supported by a Huawei / Berkeley AI fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors, and no official endorsement should be inferred.

Instruction:

Walk down and turn right. Walk a bit, and turn right towards the door. Enter inside, and stop in front of a zebra striped rug.

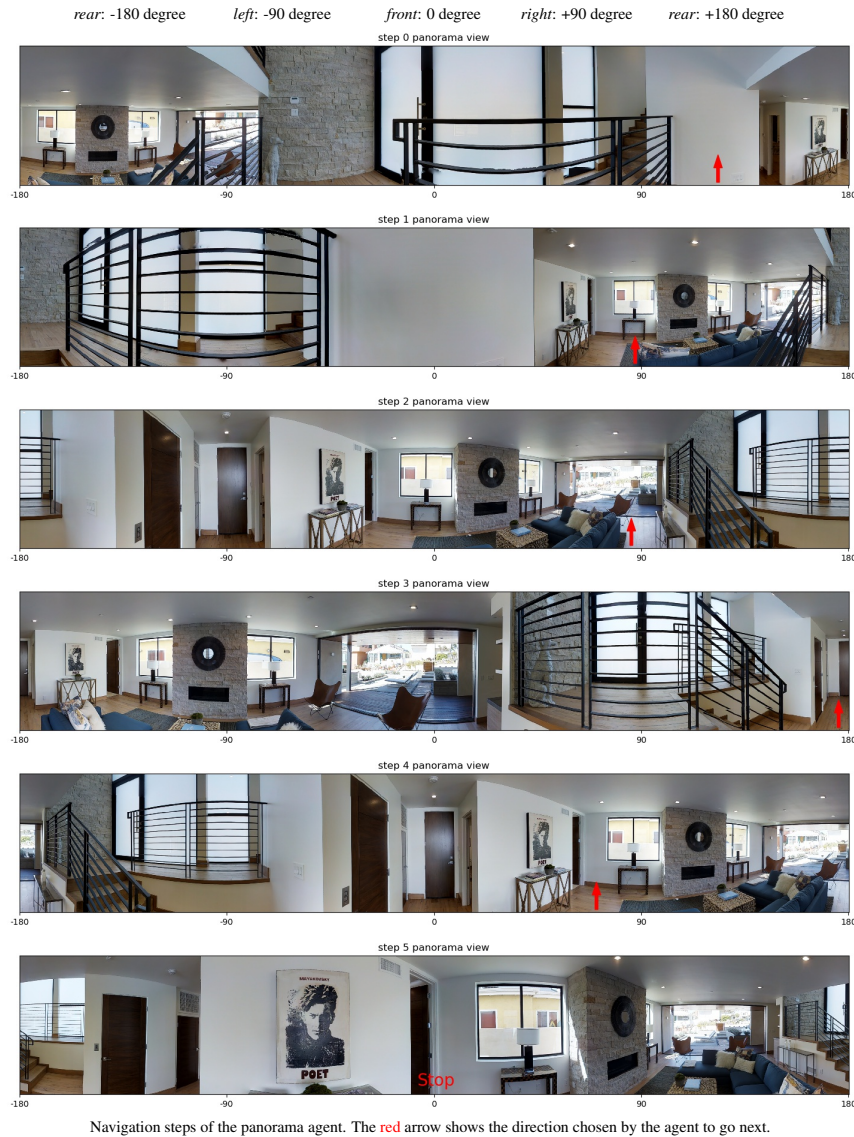


Figure 5.5: Follower **without pragmatic inference** on val seen. The instruction involves an ambiguous “walk a bit” command. Without pragmatic reasoning by the speaker, the follower failed to predict how much to move forward, stopping at a wrong location without entering the door.

Instruction:

Walk down and turn right. Walk a bit, and turn right towards the door. Enter inside, and stop in front of a zebra striped rug.

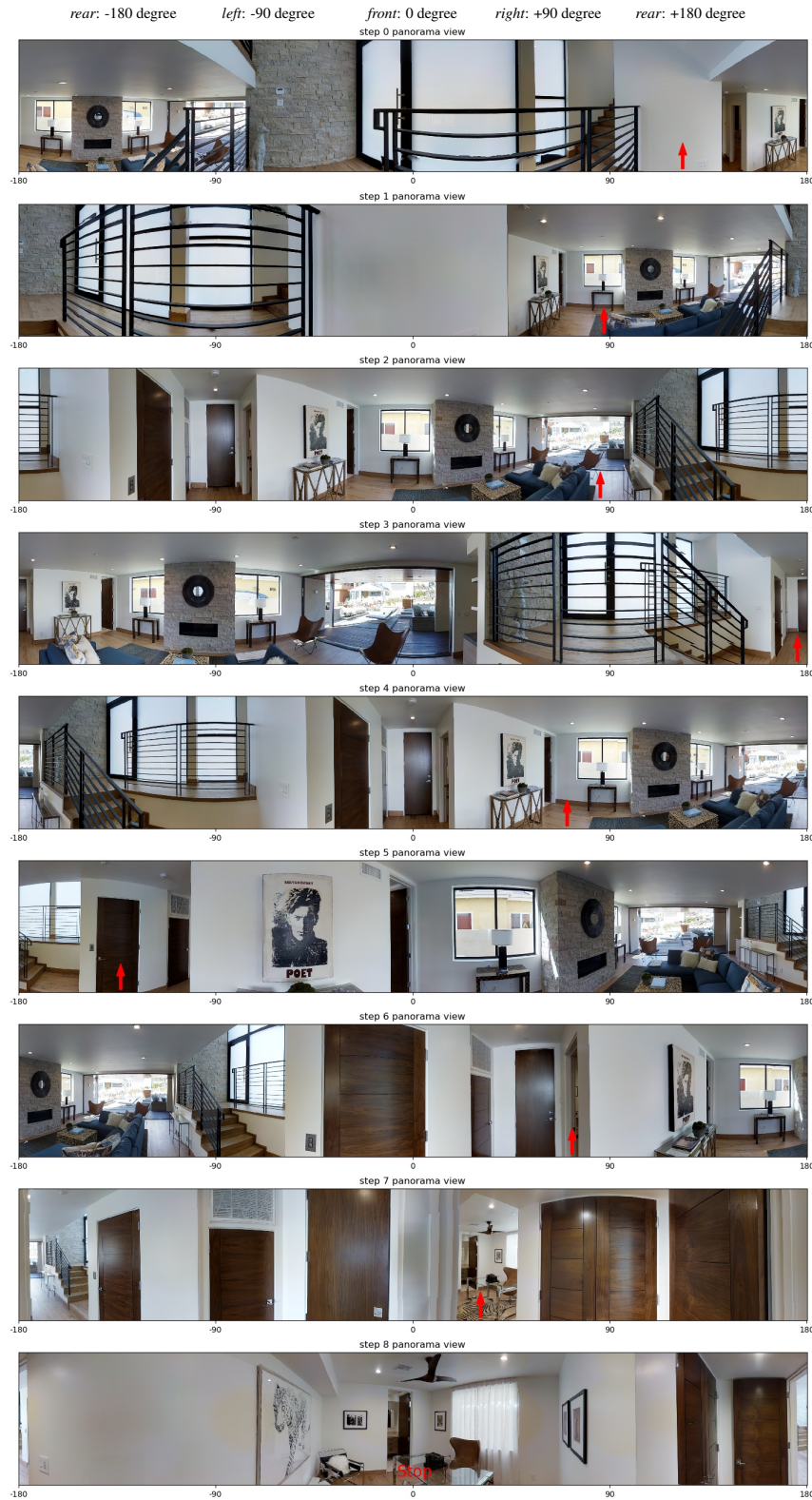
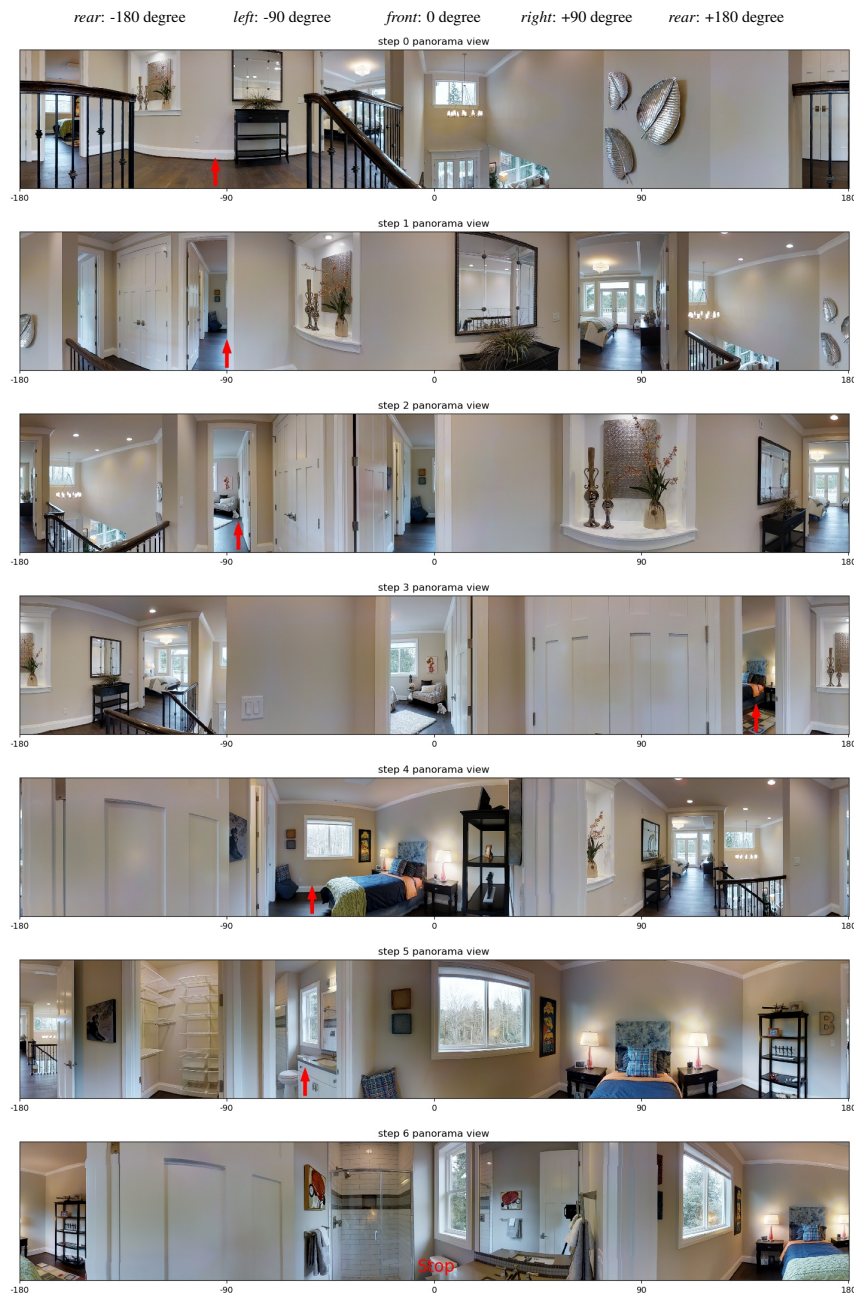


Figure 5.6: Follower **with pragmatic inference** on val seen. With the help of the speaker, the follower could disambiguate “walk a bit” to move the right amount to the correct location. It then turned right and walked into the door to stop by the “zebra striped rug”.

Instruction:

Walk past hall table. Walk into bedroom. Make left at table clock. Wait at bathroom door threshold.

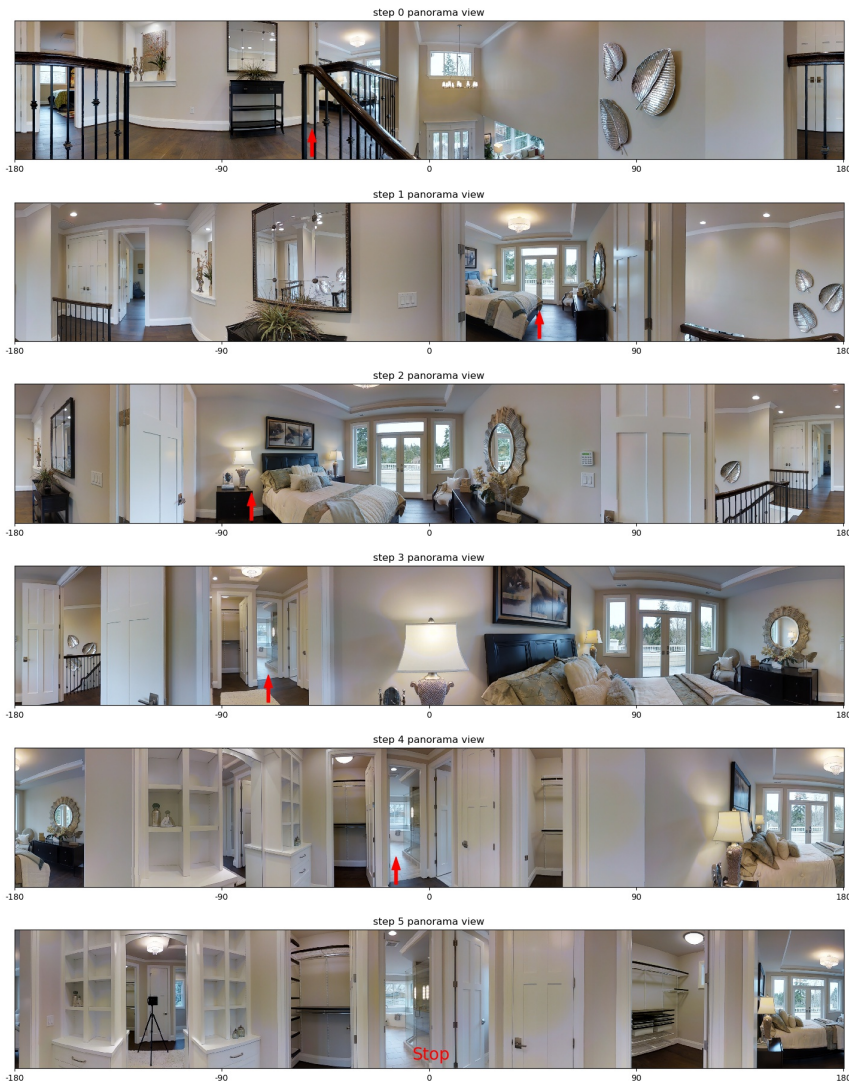


Navigation steps of the panorama agent. The red arrow shows the direction chosen by the agent to go next. Figure 5.7: Follower **without pragmatic inference** on val unseen. The command “walk into bedroom” is ambiguous since there are two bedrooms (one on the left and one on the right). The follower could not decide which bedroom to enter, but went into the wrong room with no “table clock”.

Instruction:

Walk past hall table. Walk into bedroom. Make left at table clock. Wait at bathroom door threshold.

rear: -180 degree left: -90 degree front: 0 degree right: +90 degree rear: +180 degree



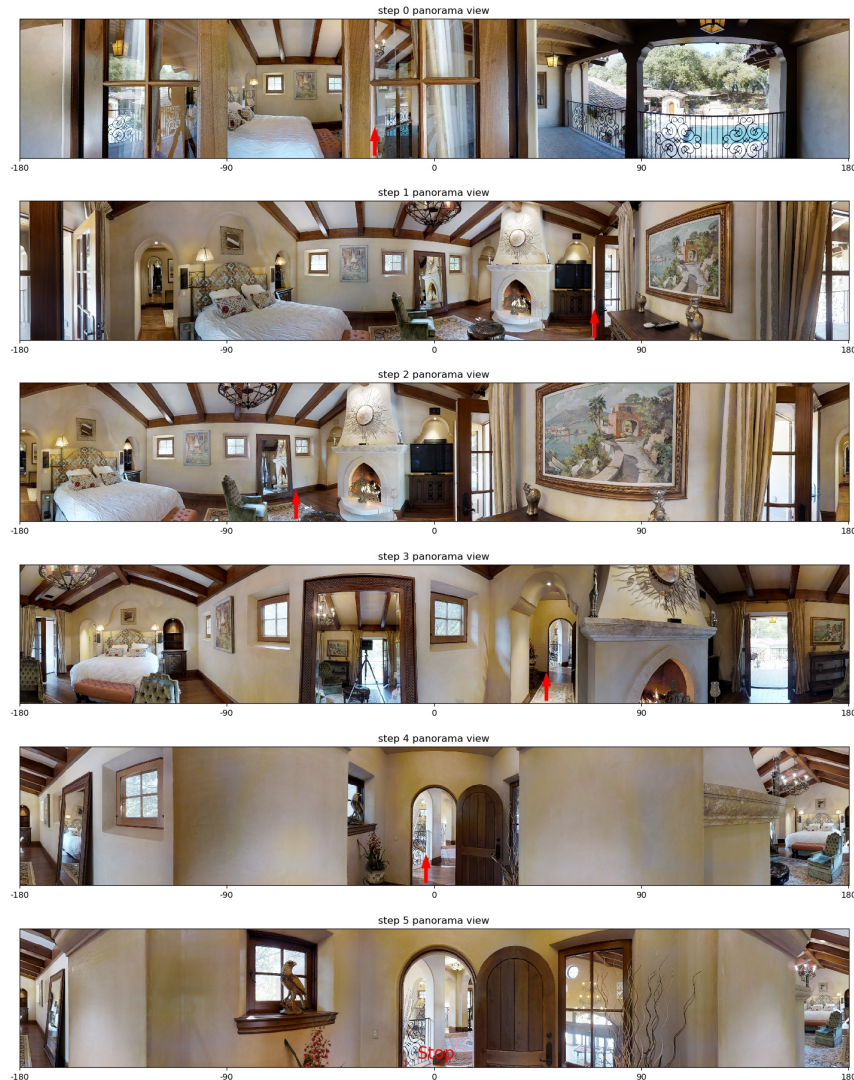
Navigation steps of the panorama agent. The red arrow shows the direction chosen by the agent to go next.

Figure 5.8: Follower with pragmatic inference on val unseen. The speaker model helps resolve the ambiguous “walk into bedroom” command (there are two bedrooms), allowing the follower to enter the correct bedroom on the right, where it could see a “table clock”.

Instruction:

Enter the bedroom and make a slight right. Walk across the room near the foot of the bed. Turn right at the end of the rug. Wait near the mirror.

rear: -180 degree left: -90 degree front: 0 degree right: +90 degree rear: +180 degree



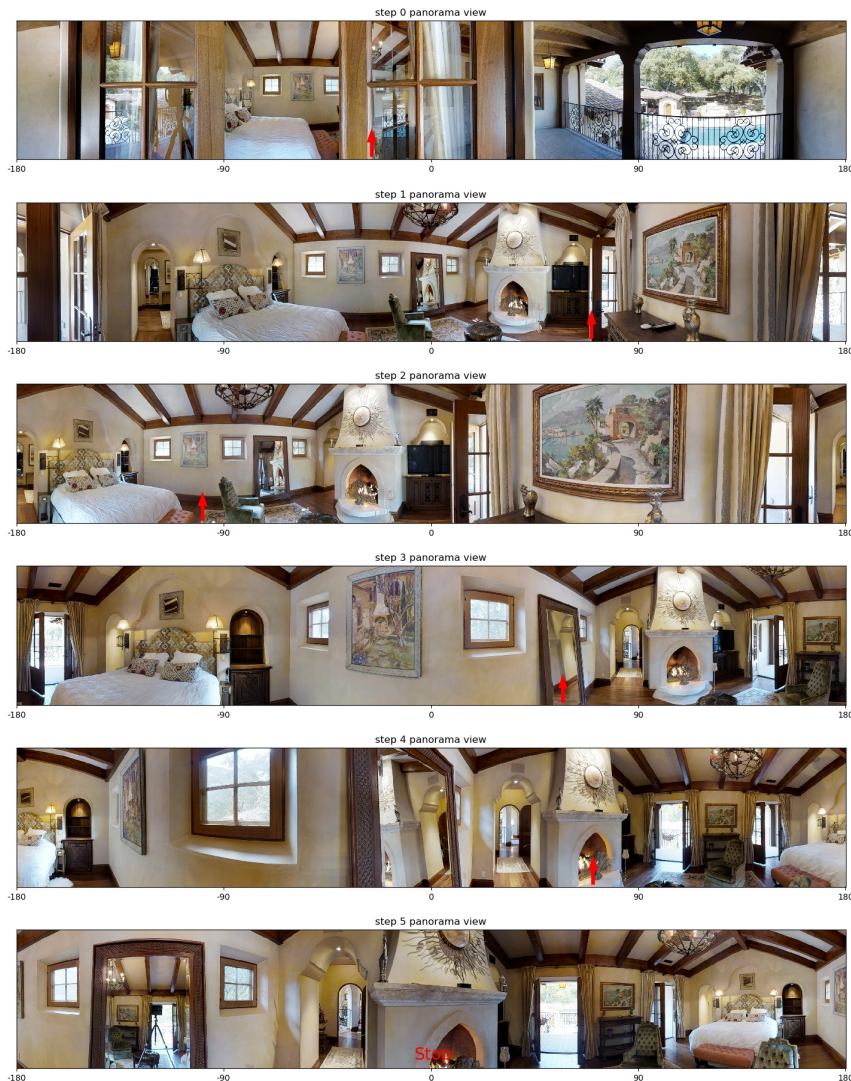
Navigation steps of the panorama agent. The red arrow shows the direction chosen by the agent to go next.

Figure 5.9: Follower **without pragmatic inference** on val unseen. Although making a right turn as described, the follower fails to turn right at the correct location, and stopped at the door instead of the mirror. The route taken by the follower would be better described as “...wait near the door” by a human, which the speaker could learn to capture.

Instruction:

Enter the bedroom and make a slight right. Walk across the room near the foot of the bed. Turn right at the end of the rug. Wait near the mirror.

rear: -180 degree left: -90 degree front: 0 degree right: +90 degree rear: +180 degree



Navigation steps of the panorama agent. The red arrow shows the direction chosen by the agent to go next.

Figure 5.10: Follower **with pragmatic inference** on val unseen. Using the speaker to measure how likely a route matches the provided description, the follower made the right turn at the correct location “the end of the rug”, and stopped near the mirror.

Instruction: *Go through the doorway on the right, continue straight across the hallway and into the room ahead. Stop near the shell.*

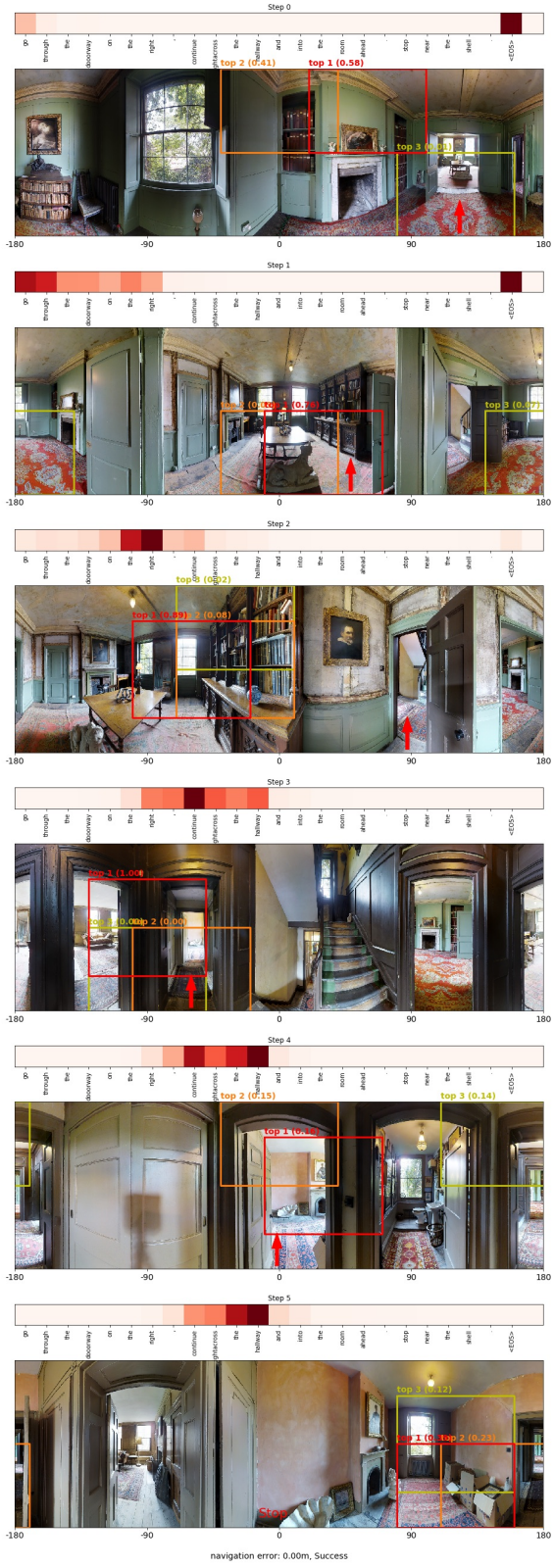


Figure 5.11: Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.

Instruction: Walk through the kitchen, enter the dining room, walk to the doorway to the right of the dining room table, wait at the glass table.



Figure 5.12: Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.

Instruction: *Walk up stairs. Turn left and walk to the double doors by the living room.*

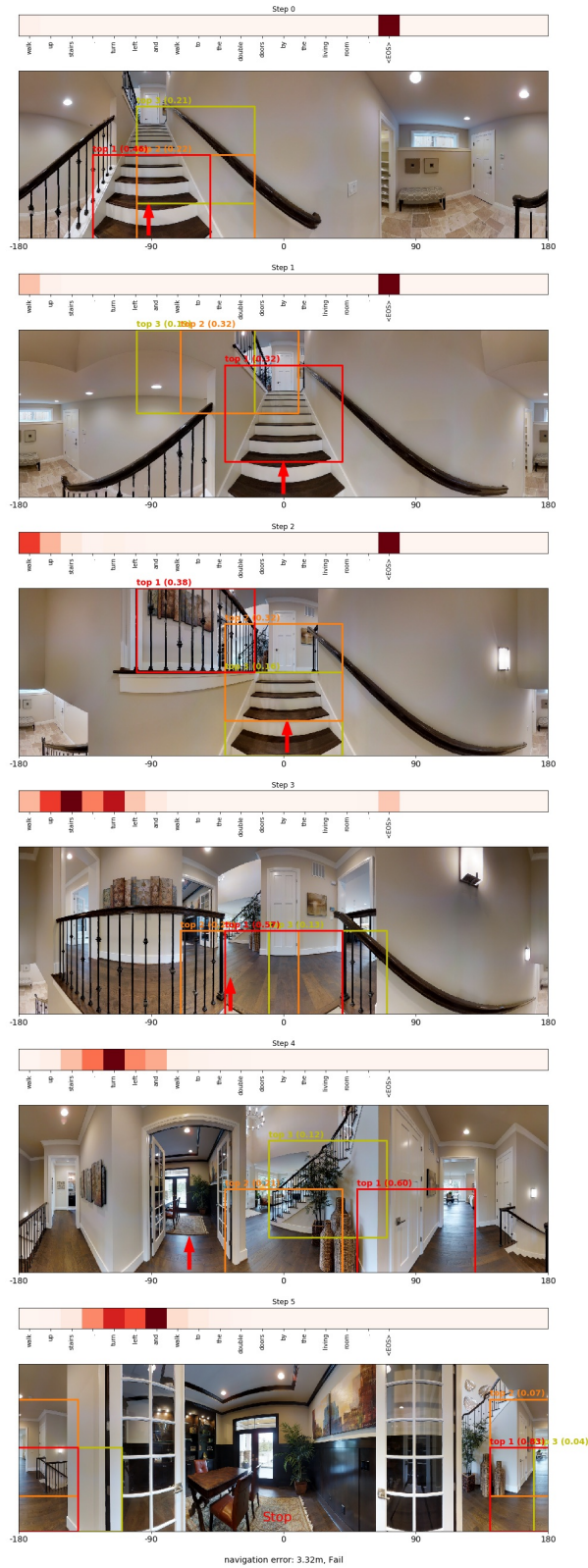
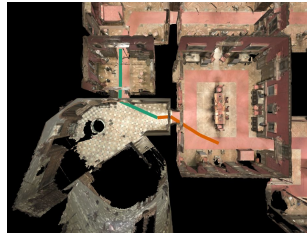


Figure 5.13: Image and textual attention visualization on val unseen (best viewed at 200%). At each step, the textual attention is shown at the top, and the 1st, 2nd and 3rd most attended view angles are shown in red, orange and yellow boxes, respectively (the number in the parenthesis shows the attention weight). The red arrow shows the direction chosen by the agent to go next.

instruction:

Go through the door on the right and continue straight. Stop in the next room in front of the bed.

top-down
overview of
trajectories



(a) **orange**: trajectory without pragmatic inference

(b) **green**: trajectory with pragmatic inference

Step
1



Step
2



Step
3



Step
4

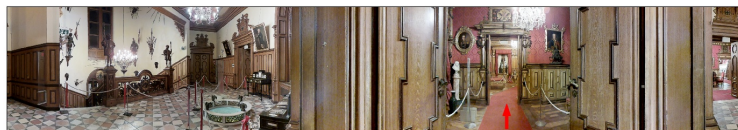


(a) navigation steps **without pragmatic inference**; **red** arrow: direction to go next

Step
1



Step
2



Step
3



Step
4



(b) navigation steps **with pragmatic inference**; **red** arrow: direction to go next

Figure 5.14: Navigation examples on unseen environments with and without pragmatic inference from the speaker model (*best visualized in color*). (a) The follower without pragmatic inference misinterpreted the instruction and went through a wrong door into a room with no bed. It then stopped at a table (which resembles a bed). (b) With the help of a speaker for pragmatic inference, the follower selected the correct route that enters the right door and stopped at the bed.

Chapter 6

General Knowledge of Objects for Referring Expression Recognition in Partially-Observed Scenes

A hallucination is a fact, not an error; what is erroneous is a judgment based upon it.

Bertrand Russell

In this chapter, we study how an embodied AI system could acquire general knowledge about objects and use this knowledge for language grounding. We introduce Hallucinating Object with Language Models, or HOLM, that extracts spatial knowledge about objects using large pre-trained language models for language grounding systems.

The work described in this chapter will appear in the following publication:

- Volkan Cirik, Louis-Philippe Morency, and Taylor Berg-Kirkpatrick, “HOLM: Hallucinating Objects with Language Models for Referring Expression Recognition in Partially-Observed Scenes”, In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL, 2022.

The code for reproducing experiments in this chapter is publicly available on Github¹.

¹<https://github.com/volkancirik/holm>

6.1 Overview

One of the fundamental challenges in building AI systems physically present in the world is addressing the issue of partial observability, the phenomenon where the entire state of the environment is not known or available to the system. People cope with partial observability by reasoning about what is not immediately visible (see example in Figure 6.1). People use their general knowledge about the world and adapt their knowledge to specific situations. General knowledge about kitchens can help to know approximately where to look for pans or utensils in a kitchen that has never been seen before. How can an AI system build general knowledge about objects and their environment to help with a similar task? Even more interestingly, can we gather this information from language, using readily available resources such as language models trained on a large collection of unlabeled text?

In this chapter, we introduce a method called HOLM, **H**allucinating **O**bjects with **L**anguage **M**odels, for reasoning about the unobserved parts of the environment. Inspired by the recent successes of large pre-trained language models (LM) extracting knowledge about the real world, we propose a methodology based on spatial prompts to extract knowledge from language models about object. HOLM extracts spatial knowledge about objects in the form of affinity scores of objects, i.e., how often a pair of objects are observed together. This knowledge of objects are combined with observed spatial layout to hallucinate what might appear in the unobserved part of the scene. We evaluate our HOLM approach on Dynamic Referring Expression Recognition (dRER) task where the goal is to find a target location by dynamically adjusting the field of view (FoV) in partially observed 360° scenes. We examine how HOLM compares with the state-of-the-art approaches on two publicly available datasets to study generalization for both indoor and outdoor settings.

Instruction: "Find the *tv*. The target is above the *tv* next to the *standing lamp*."

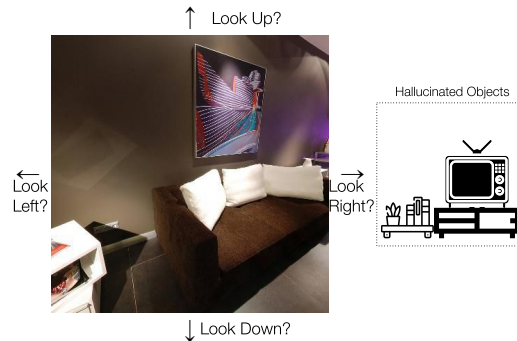


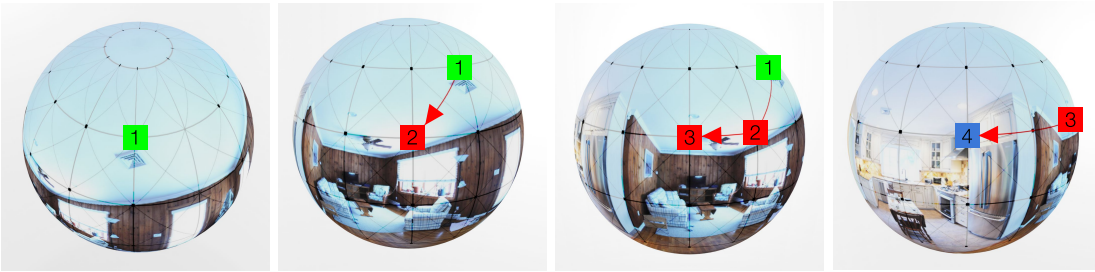
Figure 6.1: Illustration of our main contribution: **Hallucinating Objects**. Knowledge about object relationships is helpful when navigating in an unknown and partially observed environment. In the example above, the TV is not visible, but the couch hints that a TV might be in front of it because usually couches face TVs.

6.2 Dynamic Referring Expression Recognition (dRER) Task

dRER task is designed to localize a target location in a dynamically observed 360° scene given natural language instruction. Unlike conventional referring expression recognition, which refers to an object in a static visual input, in dRER, only a partial view of the scene is provided in a field of view. However, the system can adjust the field of view to find the described point in the scene. In Figure 6.2, we illustrate the dRER task and motivate our method. On top, natural language instruction is given. In the middle, the spherical view of the scene is illustrated – the

Instruction: “Find the **oven**. The target is above the **oven** on the *range hood*.”

360° Views of a Scene with Spherical Projection



Agent's Field of Views

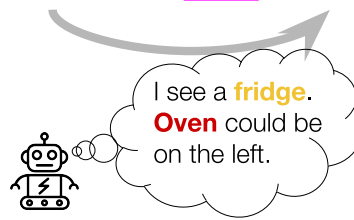
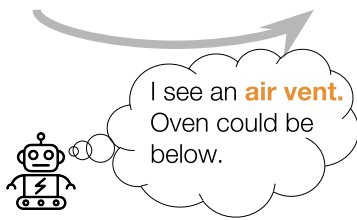


Figure 6.2: **Illustration of the dRER task with an example of language instruction and its recognition in four steps.**

The agent adjusts its FoV by looking at different directions and navigate on the graph in the spherical view. Note that objects mentioned in bold in the instruction are not visible at all until timestep 4. Thus, the agent needs to reason about possible locations of the mentioned object using its partial view of the scene.

agent explores only some portion of a 360° scene. FoVs on the sphere represented as square nodes form a graph. By *navigating to a neighboring node*, the agent adjusts its FoV and observes a different view of the scene. Note that objects mentioned in the instruction “oven” and “range hood” are not visible until the fourth timestep. Thus, to perform well on this task, it is essential to reason about where objects might appear.

The dRER task can be formulated as a Markov Decision Process (MDP) [88] $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P_s, r \rangle$ where \mathcal{S} is the visual state space, \mathcal{A} is the discrete action space ², P_s is the unknown environment probability distribution from which the next state is drawn, and $r \in \mathbb{R}$ is the reward function. For a time step t , the agent observes an image $s_t \in \mathcal{S}$, and performs an action $a_t \in \mathcal{A}$. As a result of this action, the environment generates a new observation $s_{t+1} \sim P_s(\cdot | s_t, a_t)$ as the next state.

²For computational efficiency, we picked discrete action space. It could be continuous as well.

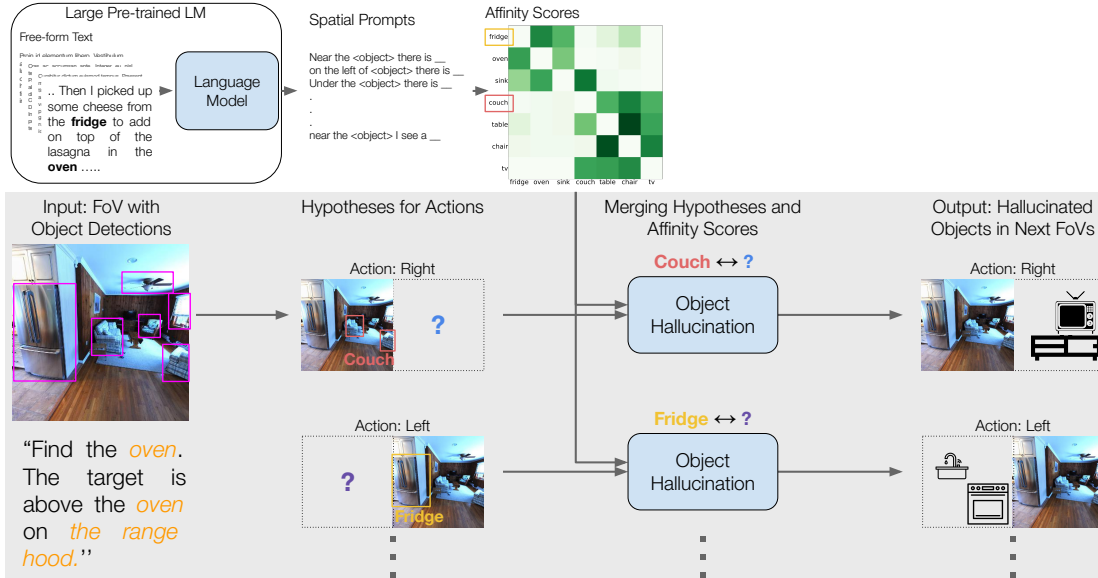


Figure 6.3: **HOLM for the dRER task.** (Top) We use language models trained on a large amount of text by prompting with the spatial relationship of objects to calculate co-occurrence statistics of objects. (Bottom) The flow of our hallucination method. We determine objects of interest for each action. Then, we combine objects of interest and co-occurrence table to hallucinate objects, i.e. what might appear after performing an action.

This interaction continues sequentially and ends when the agent performs a special `STOP` action or a pre-defined maximum episode length is reached. The resolution process is successful if the agent ends the episode at the target location.

In dRER, instructions are represented as N sequence of sentences represented as $x = \{x_i\}_{i=1}^N$. Each instruction sentence x_i consists of a sequence of L_i words, $x_i = [x_{i,1}, x_{i,2}, \dots, x_{i,L_i}]$. The training dataset $\mathcal{D}_E = \{\mathcal{X}, \mathcal{T}\}$ consists of M pairs of the instruction sequence $x \in \mathcal{X}$ and its corresponding expert trajectory $\tau \in \mathcal{T}$. The agent learns to navigate by learning a policy π via maximum likelihood estimation (MLE):

$$\begin{aligned}
 & \max_{\theta} \mathcal{L}_{\theta}(\mathcal{X}, \mathcal{T}), \text{ where} \\
 & \mathcal{L}_{\theta}(\mathcal{X}, \mathcal{T}) = \log \pi_{\theta}(\mathcal{T}|\mathcal{X}) \\
 & \mathcal{L}_{\theta}(\mathcal{X}, \mathcal{T}) = \frac{1}{M} \sum_{k=1}^M \log \pi_{\theta}(\tau^k|x^k)
 \end{aligned} \tag{6.1}$$

6.3 HOLM

In dRER, the system only observes the current FoV and does not see the resulting FoV before taking any actions. Thus, it is essential to reason what might appear in a future observation using what is currently visible to the system. Our core intuition is that objects visible in the current FoV and their locations in the FoV give us a clue about what might appear if a particular action is taken.

Here, we propose an approach for reasoning about future observations using what is visible and some general knowledge of objects. Let us go through the illustration in Figure 6.3 to explain our HOLM method. In the top panel, we feed spatial prompts to pre-trained language models to extract knowledge about objects in the form of affinity scores. In the bottom panel, we see the input of the system where there are natural language instructions, an FoV of the scene, and detected objects. Next, we calculate which objects are relevant to each action. For instance, couch detections are on the right side; thus, they are relevant to the right action. Similarly, the fridge is relevant for the left action because it is on the left side. Then on the third step, using the affinity score of a pair of objects, we predict what might appear after performing an action. For right action, our model hallucinates a tv and tv-stand might appear because the couch and tv have a high affinity score according to the LM.

6.3.1 Affinity Scores from Language Models

Language models process a large amount of text to learn regularities in natural language. They do so by predicting the next word or masked token given a sequence of words. Our intuition is that objects that frequently appear in an environment close to each other will have similar language usage. Thus, we hypothesize that language models’ capability of learning affinity scores of words in language also reflects objects’ spatial properties. In Figure 6.3’s top panel, we illustrate how we extract this capability. We query language models trained on a large amount of free-form text with spatial relationship prompts. These spatial prompts aim to capture the usage of words when they appear together in the world. An example of these prompt templates is “Near the o_1 , there is ___” where $o_1 \in O$ is an object label where O is a set of object labels. If object o_1 co-occurs with o_2 with high frequency, the language model would provide a high probability for the phrase “Near the o_1 , there is o_2 ”. Using all pairs in O and K spatial prompt templates, we generate queries q . Please see Table 6.1 for the full list of spatial prompt templates.

We then calculate affinity scores C_{o_1, o_2} , i.e., observing o_2 when o_1 is present as follows:

$$C_{o_1, o_2} = \sum_{i=1}^K p_{\text{LM}}(o_2 | q_i) \quad (6.2)$$

Where $p_{\text{LM}}(o_2 | q)$ is a language model that calculates the probability of observing a token o_2 given a prefix sequence of tokens q .

6.3.2 Object Hallucination

Our main idea behind HOLM is to reason about what might be observed in a future observation by combining (1) which objects are visible in the current observation and (2) what we know about the spatial properties of those objects. We explain the details of our approach in this section.

Let $p_a \in \mathbb{R}^{|O|}$ be the vector of probabilities of observing an object among a set of all objects O after performing an action a . We calculate p_a as follows:

$$p_a = (p_{\text{FoV}} \odot 1_a) C \quad (6.3)$$

near the object there is
 near the object I see a
 near the object there should be a
 the object near the object is
 on the left of object there is
 on the right of object there is
 on top of object there is
 under the object there is
 across the object there is
 close the object there is

Table 6.1: Spatial Prompt Templates

Where $p_{\text{FoV}} \in \mathbb{R}^{|O|}$ is a vector of confidence values for objects detected in the current FoV. We use an off-the-shelf object detection system [6] to calculate p_{FoV} . C is the affinity scores of size $|O| \times |O|$. C represents how often a pair of object appear in a spatial relationship and represents the general knowledge of objects. $1_a \in \{0, 1\}^{|O|}$ is a binary vector representing spatially related objects for a direction a . This vector is calculated with an indicator function to determine whether an object is spatially related to action a .

We calculate the indicator function as follows. First, we separate the FoV into 4 imaginary regions called quadrants where each quadrant determines how a region in observed FoV is spatially relevant for canonical directions (i.e., up, down, left, right). In other words, quadrants are “hot-spots” for each direction i.e., the left side of the image is more relevant to the right side of the image if we are interested in what might appear on the left. For 8 directions (left, right, down, up, down-left, down-right, up-left, up-right), we calculate how much each objects’ bounding box overlaps with these quadrants. If intersection-over-union is above a fixed threshold we keep this object for the hallucination process.

6.4 Experiments

We designed our experiments to study and evaluate our proposed HOLM approach under four different research questions. **RQ1:** What is the performance of HOLM when compared to other state-of-the-art approaches? **RQ2:** what is the impact of LM as a source of knowledge for HOLM when compared to other more conventional sources (e.g., images)? **RQ3:** How essential are external sources of data for learning knowledge about objects compared to in domain data? **RQ4:** How accurate is HOLM for predicting objects in future observations? **RQ5:** How do annotation-free language-based knowledge sources i.e., LMs and word embeddings compare for HOLM?

The following section explains the details of experimental setup. Our results are presented and discussed in Section 6.4.2.

6.4.1 Experimental Setup

To study the research questions previously mentioned, we identified two publicly available datasets and specific state-of-the-art methods as baselines to compare with.

Datasets. We selected the following two datasets to see if our method generalizes to both indoor and outdoor settings. The Refer360° dataset [40] consists of 17K natural language instructions and ground-truth trajectory pairs for localizing a target point in 360° scenes. The ground-truth trajectories are annotated by human annotators in the form of successive FoVs in partially observed 360° scenes. The dataset uses a subset of the SUN360 dataset [231] as the source of scenes and these scenes are from both indoor and two outdoor locations.

Touchdown [30] consists of 9K natural language instruction and ground-truth location pairs for 360° scenes on Google Streetview. Unlike the Refer360° dataset, Touchdown does not have expert trajectories – only expert predictions for the target location are provided. Thus, we generated ground-truth trajectories by calculating shortest path trajectories between a randomly selected starting point ³ and the target location.

Baselines Models. We compare our method with the state-of-the-art models and also few simple baselines (i.e., no parameter learning).

- The Self Monitoring Navigation Agent (SMNA) [143] model is trained with a co-grounding module where both visual and textual input is attended at the same time. The agent also measures its progress with a progress monitor module.
- FAST [106] stands for Frontier Aware Search with backTracking. The FAST model learns to score partial trajectories of an agent for efficiently backtracking to a previous location after a mistake.
- Speaker-Follower [59] uses a sequence-to-sequence speaker model to re-rank a follower model’s candidate trajectories. This pragmatic reasoning model has been shown to improve navigation agents’ performance significantly.
- LingUNet [156] is an image-to-image encoder-decoder model for learning image-to-image mappings conditioned on language. We should emphasize that, unlike the previous methods, LingUNet is not a navigation model; instead, it predicts regions over an image.
- RANDOM agent randomly picks an action.
- STOP agent predicts the starting FoV as the target FoV.

For a fair comparison, the same model was used as the basis for all the compared model. For our proposed approach HOLM is used to enhance the SMNA baseline by hallucinating objects for unseen regions. After getting object hallucinations for each neighboring FoVs, we use the sum of word embeddings for object labels as the input representation for the neighboring FoV. In the oracle “Next FoV” scenario, we use ground-truth FoVs to do the same process. For a fair comparison, we use SMNA as the base agent for learning to recover from a mistake during navigation process with FAST and as the follower model for pragmatic reasoning with Speaker-Follower.

³Following [40], we set the initial random point to be a fix heading and random yaw.

Method	Oracle	Refer360°	Touchdown
Stop Agent		14.1	0.0
Random Agent		12.1	6.8
SMNA [143]		27.1	45.9
+ HOLM (this work)		32.2	49.8
SMNA [143]	Next FoV	33.5	50.2
LingUNet* [30]	Full Panorama	21.4	47.2

Table 6.2: FoV accuracy results for Refer360° and Touchdown with no hallucination baseline, best performing models, and Next FoV oracle model, i.e. the ability to look ahead for neighbor FoVs, and observing full 360° scenes. Our method outperforms the baseline models from the literature.

Evaluation Metrics. Our main evaluation metric for methods is FoV accuracy: the percentage of the time the target location is visible in the final FoV. The FoV accuracy sets an upper bound on the localization accuracy for predicting the pixel location of the target point, i.e., if the target is not visible, it is impossible to predict the exact location. Thus, we focus on this metric to compare systems. Since the main training objective is to find the *any* FoV where the target object is visible, unlike Touchdown [30] we do not report pixel-level distance to the target location.

Implementation. All models are trained for 100K iterations. We use Adam [112] for optimization with a learning rate 0.0001 and weight decay parameter 0.0005 [119]. For each model, we perform a grid-search over their hyperparameters (e.g., number of hidden units, number of layers, dropout rate) and pick the best performing model based on validation score⁴. All models are implemented using PyTorch [167] and publicly available.

To speed up the training procedure, we used fixed a grid of FoVs for all 360° images where each FoV is connected to its neighboring FoVs. This grid forms the navigation graph depicted in the Figure 6.2. We use 30° of separation between successive FoVs which provides enough overlap to reveal relevant information about successive FoVs yet distant enough so that the model needs to reason about future steps. We then pre-calculated the rectilinear projection of each of the FoVs on the grid for all scenes.

6.4.2 Results and Discussion

In this section we present and discuss experimental results and analyses.

(RQ1) HOLM Improves performance. Our main results are presented in Table 6.2. In the first row block, we see that simple non-learning baselines fail to perform on the dRER. In the second row block, we compare our method with the baseline where the agent does not have any visual input from the next FoVs. HOLM improves the baseline by hallucinating objects for the next FoVs. In the third row block, we provide results for oracle scenarios. For SMNA, we feed ground-truth FoV as the input of the system. This result sets the upper bound on HOLM,

⁴For Refer360° we use validation unseen split. Touchdown does not have seen-unseen distinction.

Method	Beam Search	Refer360°	Touchdown
Baseline SMNA [143]		27.1	45.9
+ HOLM (this work)		+5.1	+3.9
+ FAST [106]		-6.4	+4.7
+ Speaker-Follower [59]		-4.6	-11.1

Table 6.3: FoV accuracy results for Refer360° and Touchdown for methods using beam search or single candidate trajectory. HOLM consistently improves the baseline and does not use multiple trajectories.

because it cannot achieve better hallucination than the ground-truth FoVs. However, HOLM achieves pretty close to this upper bound and show that it can provide useful predictions for this task. For LinGUNet, we feed the full 360° scenes as the visual input. Since LingUNet is not a navigation agent i.e. predicts the target location using full 360° scenes, we calculate FoV accuracy by drawing an FoV around the prediction, which explains ‘*’.

In Table 6.3, we compare HOLM with FAST and Speaker-Follower methods, both of which use beam search. During the beam search, these methods use multiple trajectories while deciding on a trajectory. However, this is not plausible in a real-world scenario, i.e. a robot would not generate many trajectories before performing action. HOLM, on the other hand completes the task on a single trajectory while predicting possible future states. FAST improves SMNA for Touchdown but not for Refer360°, which might be due to the richness of scenes in Refer360° whereas in Touchdown, the scenes are always in the same domain. Speaker-Model’s decreases the score for SMNA possibly due to the Speaker models’ poor performance where the BLEU score is around 6. HOLM consistently improves for both datasets and does not perform any expensive look-ahead operations such as beam search.

Knowledge Type	Human Annotation	Affinity Scores	Refer360°	Touchdown
Baseline		Uniform	27.8	45.2
Baseline		Diagonal	29.3	45.9
Visual		VisualGenome	30.8	48.4
Knowledge Base		WordNet	29.5	48.4
Pre-trained LM		XLM	32.2	49.8

Table 6.4: FoV accuracy results for Refer360° and Touchdown for different methods for calculating affinity scores for HOLM. XLM-based affinity scores achieve the best performance.

(RQ2) Pre-trained LM produces better affinity scores compared to other sources. In Table 6.4, we compare several baseline methods for calculating the affinity scores. First, we use uniform (i.e., each object pair has the same affinity score) and identity matrix (i.e., object x can only have affinity score with itself) baselines. We also study calculating affinity scores using data annotated by humans. First, we use object annotations in VisualGenome [118]. VisualGenome provides a large collection of fine-grained annotations for objects and their spatial relationships. Second, ideally we would like to use human annotations for calculating the affinity score. However, this requires annotation of $|O|^2$ annotations. Instead, as a proxy, we use WordNet

[155], a knowledge-base hierarchy annotated by experts. We use NLTK [18] to calculate the WordNet similarity to extract the affinity scores between objects. XLM-based HOLM achieves the best results among these baselines. This result shows that without using human annotations, we can extract useful knowledge about objects using pre-trained LMs.

Method	Data Source	Refer360°	Touchdown
HOLM with XLM	External	32.2	49.8
HOLM with Objects Counts	Internal	30.3	48.7
Hallucinating with 3-Layer MLP	Internal	27.5	46.3

Table 6.5: FoV accuracy results for Refer360° and Touchdown when task data is used for object hallucination. The limitation of the domain data can be addressed using external resources such as pre-trained LMs.

(RQ3) External sources may provide better information compared to task data. In Table 6.5, we compare methods that only use task data for object hallucination and HOLM with external sources such as pre-trained LM. For the second row in the table), we use the BUTD model [6] to annotate training images with object bounding boxes. Using bounding boxes of objects, we calculate affinity scores. For the third row in the table, we design a model that takes FoV and an object type as an input and predicts a direction (i.e., hallucinate where it might appear) as output. We pass the final feature map layer of 152-layer ResNet[80] as input to a 3-layer feed-forward neural network to predict objects that might appear in neighboring FoVs. This model achieves an F1 score of 40.3 for direction prediction. Both of these methods improve over the SMNA baseline but are worse than the pre-trained LM. This result indicates that task data may have limitations, and external sources such as a pre-trained LM may provide a signal for knowledge about objects.

Knowledge Type	Affinity Scores	Refer360°	Touchdown
Visual	VisualGenome	P 1.4 R 55.3 F1 2.7	P 1.5 R 55.2 F1 2.9
Knowledge Base	WordNet	P 1.3 R 55.4 F1 2.6	P 1.4 R 55.3 F1 2.8
Pre-trained LM	XLM	P 2.0 R 49.5 F1 3.9	P 2.2 R 63.2 F1 4.3

Table 6.6: Precision (P), Recall (R), and F1 scores for Refer360° and Touchdown for hallucinating objects in neighboring FoVs. Similar to the downstream task results, pre-trained LM performs the best.

(RQ4) Accuracy of HOLM translates to dRER So far, we measure the performance of HOLM for the downstream dRER task. We can also measure how accurate HOLM is at predicting the presence of an object in neighboring FoVs. We annotate each neighboring ground-truth FoVs with detections from BUTD. If the p_a^i for object $o_i \in O$ is above $\frac{1}{|O|}$, we count that as a prediction of an object in the neighboring FoV after performing action a . In Table 6.6, we provide precision, recall, and F1 score for the performance of different methods for calculating affinity scores for HOLM. XLM achieves the best performance among the methods we compare. We conclude that the performance for the intrinsic task (i.e., predicting the presence of objects) translates to dRER performance.

Method	Model	Refer360°	Touchdown
Baseline	SMNA	27.1	45.9
WE	+ HOLM with FastText [153]	31.6	46.8
	+ HOLM with GloVe [169]	31.0	49.2
	+ HOLM with word2vec [154]	29.3	46.2
LM	+ HOLM with GPT3 [22]	31.1	46.3
	+ HOLM with Roberta [135]	30.3	46.0
	+ HOLM with XLM [44]	32.2	49.8

Table 6.7: FoV accuracy results for Refer360° and Touchdown for models processing unlabeled text. **WE** and **LM** are abbreviations for word embeddings and language models. All hallucination-based methods perform better than the baseline. XLM achieves the best performance in both datasets.

(RQ5) Both word embeddings and LMs are good sources of general knowledge of objects

In Table 6.7, we compare word embedding methods and different language models. We use cosine similarities between pairs of objects to calculate the affinity scores. For language models, we compare Open AI’s GPT3 [22] using their online API⁵. We use Transformers Library [228] for RoBERTa [135] and XLM [44]. All methods consistently improve over the baseline SMNA model, however, we achieve the best performance using XLM. This result indicates that we can extract useful knowledge about objects with methods relying on large amount of unlabeled text.

6.5 Related Work

Our work on dRER is closely related to previous studies focusing on Referring Expression Recognition (RER), Vision-and-Language Navigation (VLN), and methods we propose are related to pre-training language models for vision-and-language tasks, model-based reinforcement learning, and co-occurrence modeling for computer vision. We review these studies in this section.

RER is the task of localizing a target object or a point in an image described by a natural language expression. The most of existing datasets poses the task in 2D images with objects as being the target [2, 35, 105, 133, 146, 198, 239]. Several lines of work are proposed to address RER. In joint embedding approaches [47, 61, 98, 132, 134, 142, 146, 160, 239, 240, 242, 244], object representations and referring expression representations separately learned and projected into a joint space. Modular approaches learn specialized neural network modules for object localization or relationships among objects. They computationalize the recognition process either with a pre-defined composition of modules [92, 238] or with external parsers [41, 129]. Another line of work is to learn inter-object relationships representations via graph neural networks [136, 222, 234, 235]. Recent success in masked pre-training approaches [48] also showed progress in the RER task and achieved state-of-the-art results [34, 104, 139, 140].

In Touchdown [30] and Refer360° [40] the target is a point not an object in a 360° image. In

⁵<https://beta.openai.com/>

the dRER setup, we also use 360° images of Touchdown and Refer360°, but we do not provide the full panoramic view of the scene. Instead, in a more realistic scenario, the agent observes a partial and dynamic view of the scene, i.e. the agent needs to adjust its FoV to find the target location. Closer to our work, in REVERIE [175] an embodied setup is proposed where the agent needs to first navigate to a location where the target object is visible. Similar to Touchdown and Refer360°, at the final position, the full 360° view is visible to the agent. Unlike ours and similar to 2D image-based RER, the target is an object rather than a point in the scene.

VLN is a vision-and-language task where an agent in a simulated environment observes a visual input and is given a natural language instruction to navigate to a target location. The earlier work [28, 144, 193] studies the task with synthetic images or in a very small scale [219]. Anderson et al. [8] proposes Room-to-room (R2R) benchmark and revisit VLN task with a modern look. In R2R, the agent observes panoramic scans of a house [25] and needs to carry out the natural language instruction. EnvDrop [209] model shows generalization to unseen environments by dropping visual features. PREVALENT [78] tackles the data sparsity problem with a pre-training scheme. Hong et al. [85] show that a pre-trained multi-modal can be enhanced with a memory state for the VLN task by recurrently feeding a contextualized state feature after each time step. dRER also poses a navigation task where locations in physical space in VLN correspond to FoVs in a fixed location. In dRER, a trajectory of the agent corresponds to its resolution process for finding the goal location.

Pre-trained models for Vision-and-Language has been recently studied after the huge success of transformer-based models [215] in NLP [22, 44, 48, 135, 173, 180, 204]. Numerous studies extend these approaches to the multimodal domain [96, 126, 139, 174, 199, 203, 208]. They achieve the-state-of-the-art results in several tasks such as image captioning, text-to-image retrieval, or referring expression recognition. Our work differs from these studies in the sense that the previous approaches use large scaled paired image-text data [33, 51, 101, 178, 187] to learn efficient representations [60, 115] for visual and textual modalities whereas we are interested in spatial information learned in unimodal text representations.

Language priors for vision were explored in recent studies. Lu et al. [138] use word embeddings in a language module to learn a representation for a object-predicate-object triplet for visual relationship detection task. Kiela et al. [111] propose an approach to extend pre-trained transformer-based LMs for multimodal tasks. Similarly, Lu et al. [141], Tsimpoukelli et al. [213] show that pre-trained LMs can be finetuned to perform well in few-shot settings for image classification and open-domain Visual Question Answering [148]. Marino et al. [147] also show that multimodal transformer architectures capture implicit knowledge for a pair of objects. Our work differs from these studies (1) we use only unimodal models, (2) we do not finetune models – we do not update models during training. The most similar work to ours, Scialom et al. [189] show that pre-trained LMs can perform reasonably well on Visual Question Generating [159, 237] out of the box. One difference is that we use object labels rather than object features or the appearance of objects to query the language model; however, they use object features as a visual token to the language model. Prompts we use in our work shares similarities with prompts designed in PIQA [166], but our work is evaluated in a multimodal setup. In contrast, PIQA is evaluated for textual commonsense reasoning tasks.

Hallucination idea is also related the work on predicting future observations in long horizons [217] which has been studied in the context of learning planning [76] and acquiring skills for

control problems [75], and efficient policy learning [74], and vision-and-language navigation [113]. All these approaches are interested in longer horizons; however, in our work, we study predicting single-step future observation. More recent work [91, 182, 185] study view synthesis from a single visual observation. Unlike these approaches, HOLM does not generate pixel-level views rather abstractions of views with object labels.

Affinity scores are mainly studied in computer vision tasks in the form of object co-occurrences. Previous studies have shown that object co-occurrences are efficient representations of visual prior for object categorization for object segmentation [62, 122, 177] and zero shot object-recognition [152]. Our work differs from these studies: we do not calculate co-occurrence statistics, i.e. we do not count the frequency of times they appear together; instead, we calculate a probability measure using language models.

6.6 Conclusion

In this chapter, we showed that prior knowledge about objects extracted from LMs and used as affinity scores for predicting future observations. Our experiments showed that our HOLM approach improves over various baselines from the literature. Surprisingly, our model which used general knowledge from LMs outperformed models with knowledge from human-annotated data showing that LMs learn useful knowledge about the world without requiring any visual observation from the real world. We also showed that our approach generalizes to both indoor and outdoor scenarios.

Future work will explore the use of general knowledge in other domains such as vision-and-language navigation [7] and dialog [212]. We also believe general knowledge of objects would be handy in complex scenarios such as manipulating objects in a simulated environment [194]. Another interesting direction would be to study the capability of transferring knowledge from indoor to outdoor settings and vice versa. Finally, the success of PREVALENT [78] and other pre-training approaches for VLN could stem from their ability to implicitly encode prior knowledge about objects. Hopefully, future studies examines this phenomenon.

Acknowledgements

This material is based upon work partially supported by National Science Foundation awards 1722822 and 1750439, and National Institutes of Health awards R01MH125740, R01MH096951 and U01MH116925. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors, and no official endorsement should be inferred.

Chapter 7

Situated Knowledge for Remote Embodied Visual Referring Expression

Neo : Are you saying I have to choose whether Trinity lives or dies?

The Oracle : No, you've already made the choice. Now you have to understand it.

The Matrix Reloaded, 2003

In the previous chapter, we studied from a more general perspective how the prior knowledge of objects gives the AI system a hint on which objects *might appear* in the environment. In this chapter, we explore a different type of prior knowledge. This prior knowledge is the one the AI system has from its previous exploration of the current environment. We call this second type *situated knowledge*; the knowledge belongs to a particular situation or context. Our first goal in this chapter is to show how an AI system situated in a specific environment with the knowledge of previously seen objects and their locations can be more efficient and accurate in completing natural language instructions. Second, we show how to acquire situated knowledge when it is not available with an exhaustive search. Our analyses will also demonstrate that complex models are not always needed when situated knowledge is available. These analyses help us better understand what challenges still require future research.

The code for reproducing experiments described in this chapter is publicly available on Github¹.

¹<https://github.com/volkancirik/OSMaN>

7.1 Overview

Imagine giving some directions to a tourist in your hometown. Before the GPS-augmented cell phone era, we would draw a map on paper highlighting all critical landmarks and showing directions about what to do when they see those landmarks. The tourist can then follow the instructions we give by matching what they observe in the city with what was written on the map. Our prior knowledge about our neighborhood (i.e., *where things are*) would help us develop a navigation route without having to explore the neighborhood again. We call this type of prior knowledge *situated knowledge*. Situated knowledge is about a specific situation or context. In this thesis, the embodied AI system’s environment sets the situation and context. Thus, for embodied AI systems, situated knowledge consists of knowing “what” and “where” in an environment – which would be a powerful tool when navigating and interacting with humans. Our first goal in this chapter is to understand the role of situated knowledge for the vision-language task. For instance, a prior on where the ottomans are usually might hint that they can often be in living rooms near a chair, as we discussed in Chapter 6, but ottomans could also be used in almost any room in the house. Thus an ottoman could be in the bedroom for a specific home. The knowledge about this would simplify the navigation for the AI system. But, this type of knowledge may not be readily available or too expensive to acquire. In that case, the AI system should situate itself in the environment to acquire the knowledge needed to complete its goal. This brings us to our first research question for this chapter: *(Q1) How can an AI system use situated knowledge about its environment for language grounding, and how can it acquire such knowledge?*

A second important goal of this chapter is to understand the limitation and potential of situated knowledge for vision-language navigation. To that end, we keep our design of the situated agent (1) modular and (2) simple. Modularity (1) helps us categorize different phenomena required to build a performant vision-language system. Simplicity (2) gives us the flexibility to change variables for the vision-language system. With a modular and simple design, we use our model as a lens to analyze and understand challenging phenomena for vision-language tasks. This analysis will help researchers gear their efforts towards future directions, which brings us to the second research question for this chapter: *(Q2) How can we build a simple situated vision-language navigation system to uncover some insights about the vision-language navigation tasks?*

This chapter studies these research questions and shows how an AI system can benefit from situated knowledge during navigation and remote object localization tasks and identify potential future directions using a simple-yet performant baseline model. To study situated knowledge in vision-language navigation benchmarks, we focus on the dataset known as Remote Embodied Visual referring Expression in Real Indoor Environments, or REVERIE[175]. This task conveniently ties the previous chapters in the following ways. In the REVERIE task, the agent is placed in the same simulated environment as in Chapter 5 and instructed to identify an object that is remotely located in the environment. To complete a REVERIE instruction, the agent moves in this simulated environment by performing navigation actions. At the end of the navigation, the agent recognizes the referring expression by predicting a bounding box for an object as in Chapters 2 and 3. However, the agent observes a 360° view of the house in that terminal location, similar to Chapters 4 and 6.

Our study has three main contributions.

1. We introduce **Object-centric Situated Map Navigator**, or **OSMaN**², a simple yet strong baseline for the REVERIE. Our empirical analyses show that an agent with situated knowledge of objects achieves state-of-the-art performance. It does so without any training by just using simple and intuitive heuristics we designed for vision-language navigation tasks.
2. We show how to acquire situated knowledge when it is not readily available. Our experiments show that our exhaustive search method that only relies on an object detector is more accurate than the complex neural networks for remote object localization.
3. We introduce REVERIE-DeLiRE³, a subset of the REVERIE for which OSMaN and other systems from the literature struggle the most. We identify the key aspects of REVERIE-DeLiRE which points to the future directions for vision-language navigation and remote object localization.

7.2 OSMaN: Object-centric Situated Map Navigator

The REVERIE task is designed to identify an object in a remote location in a simulated house given natural language instruction. An example instruction for this task could be “go upstairs, enter the bedroom, and bring my water bottle which is on the nightstand”.⁴ To accomplish this task, an embodied AI system should have very sophisticated capabilities: processing potential ambiguity in natural language instruction, recognizing complex and diverse visual environments, and challenging long-horizon action planning. While a complex problem, our intuition is that simple capabilities may bring us a long way in addressing this task in the presence of situated knowledge. In this task, situated knowledge consists of (1) where objects are located and (2) how locations of these objects are connected. We highlight three capabilities required to solve the task with situated knowledge:

1. The system should identify what the goal is from the natural language instruction. For example, the system needs to extract “water bottle” from the full instruction. In many navigation tasks, this goal may be a location or an object.
2. The system must decide whether the goal (e.g., the water bottle) is present in its situated knowledge. Thus, the system needs to compare the goal object with objects present in the situated knowledge.
3. Once the goal object is found in the situated knowledge, the AI system navigates efficiently using situated knowledge of the environment (i.e., how locations of objects are connected) to reach the object’s location and identify the object.

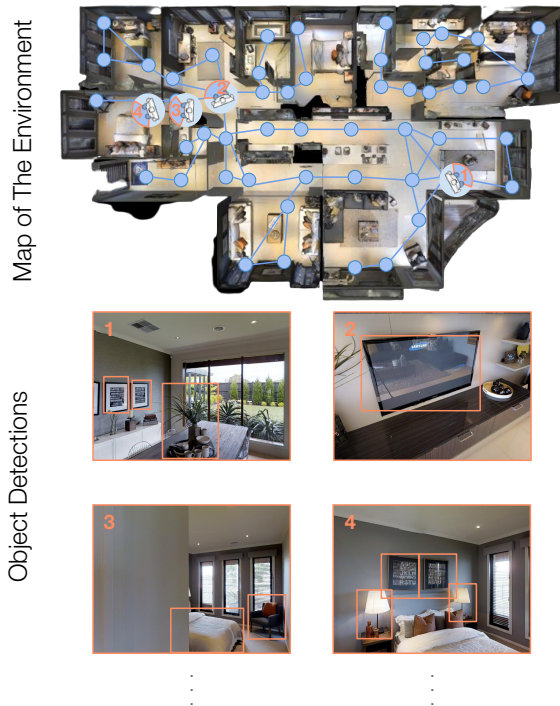
We argue that having situated knowledge of the environment – i.e., where objects are and how to go from one location to another – would simplify the complex remote object localization task into a simple search problem. We propose OSMaN, a heuristics-based model operationalizing these three capabilities in three modules by relying on situated knowledge of the environment.

²Osman is a common name in Turkey. Ottoman Empire means Osman’s Empire in Turkish.

³**Derailing and Limiting Referring Expressions.** Rêverie is French word for dreaming and “délire” means delirium, wild idea, or lunacy.

⁴Note that the agent only needs to correctly identify the object in the REVERIE rather than performing object manipulation actions.

Situated Knowledge



Remote Object Localization with OSMaN

Instruction:

“Go to the living room and find the tv across the couch.”

1) FIND GOAL

“Go to the living room and find the tv across the couch.”
→ find (VERB) → dobj (the tv, NOUN) → prep (across, ADP) → pobj (the couch, NOUN)

2) SCORE OBJECTS

Location1 vase#1 picture#1 picture#2 ...

Location2 television#1

Location3 bed#1 chair#1 ...

Location4 lamp#1 lamp#2 picture#1

...

...

3) NAVIGATE & PREDICT

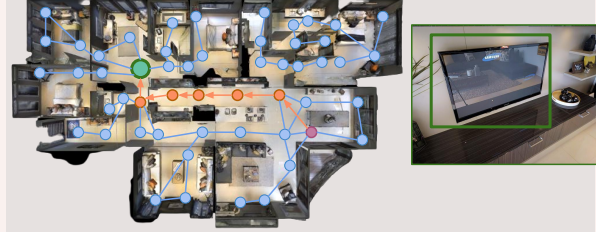


Figure 7.1: (Left) Situated knowledge in the form of object detections at each location in the environment and the connection between these locations. (Right) Steps OSMaN takes for remote object localization. OSMaN extracts the goal object, scores each candidate object with textual and semantic similarity, chooses the highest-scoring object and location and navigates to the object’s location using known connections, i.e., the environment map.

Let us go through Figure 7.1 to illustrate how OSMaN uses situated knowledge for remote object localization. We illustrate OSMaN’s situated knowledge of the environment on the left: it knows how locations are connected with a map of the environment and which objects appear in those locations. On the right, we show OSMaN’s steps for remote object localization. OSMaN first extracts the goal object from the full instruction as a word token. Second, this goal token is used for scoring the similarity between all objects in the environment. The object with the highest similarity score is chosen as the goal object. Third, OSMaN navigates to the location of the goal object and predicts the goal object.

The following sections explain how we implement this process via three modules. We want to keep our design as simple as possible for each of these modules. With simplification, OSMaN helps us understand the limitations and potentials of each of these modules for designing a situated agent for vision-language navigation tasks.

7.2.1 FIND GOAL

Natural language instructions describe a target location or object in detail in vision-language navigation tasks. Our intuition is that complex instruction can be summarized as a high-level goal. This high-level goal could be enough to complete the task in the presence of situated

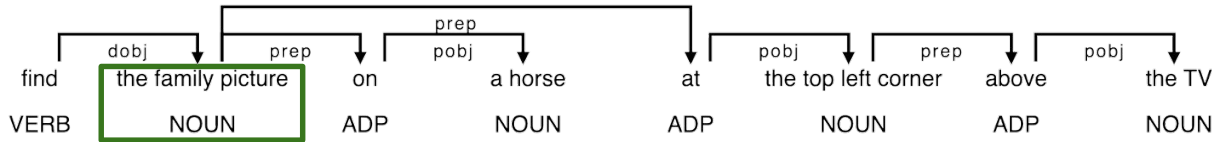


Figure 7.2: Dependency parse of an instruction. In `FIND GOAL`, we use the first noun phrase of the root as the goal object.

knowledge. For instance, an instruction like “Go upstairs, then go to the blue family room and find the family picture on a horse at the top left corner above the TV” has detailed information about the intermediate steps but could possibly be summarized with the end goal of finding “the family picture.” If the AI system already knows the location of “the family picture,” this high-level goal could be enough.

To extract the end goal, we design a simple module `FIND GOAL`. First, we discard the intermediate steps and find the goal object’s clause. For above example, we are left with “find the family picture on a horse at the top left corner above the TV” after this discarding. Then, we use an off-the-shelf dependency parsing tool[87] to parse the instruction. This gives us the dependency relationship between each word in the instruction. An example of how `FIND GOAL` uses the dependency parse tree is in Figure 7.2. We return the root token’s first noun phrase dependent. We find that the first noun phrase is the target object due to the imperative nature of the task. Our `FIND GOAL` module correctly identifies the target object as “the family picture” for the above example. We annotated a small subset of 100 instances in the training data and observed that this module is accurate for 77% of the time, which is surprisingly accurate considering its simplicity

7.2.2 SCORE OBJECTS

Algorithm 1: Scoring Objects’ Similarity to The Goal Phrase

```

1: procedure SCORE_OBJECT(goal, object)
2:   if goal = object then
|     return 3           //Exact Match Score
3:   else if substring(goal, object)=True then
|     return 2           //Sub-string Score
4:   else if edit_distance(goal, object) >= edit_threshold then
|     return 1           //Edit-Distance Score
5:
6:   emb_goal = embedding_lookup(goal)
7:   emb_obj = embedding_lookup(object)
8:   cosine_sim = cosine_distance(emb, emb_goal)
9:   if cosine_sim >= emb_threshold then
|     return cosine_sim //Semantic Similarity Score Score
10:  return 0             //Non-Matching Score
11: end procedure
  
```

Situating OSMaN-Exhaustive in the Environment

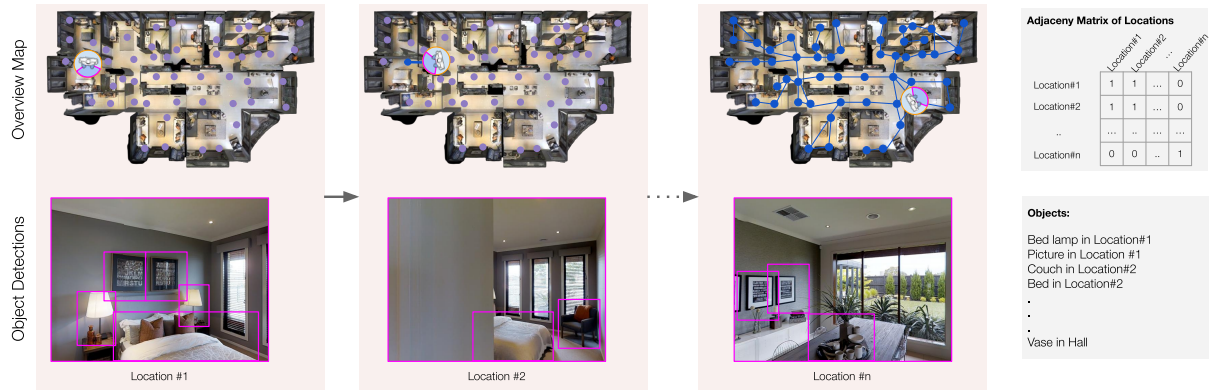


Figure 7.3: OSMaN-Exhaustive’s process of situating itself in the environment. At each location, it detects and records objects. Then explores other locations and keeps track of adjacency information between successive locations. At the end of this process OSMaN-Exhaustive knows what objects are visible and where, and knows how to navigate from any start location to any target location.

Once OSMaN has a high-level goal described in a short phrase, it scores all objects present in the environment. We use the text and semantic similarity of the goal phrase and the object label. In the SCORE module, we use a simple multi-step scoring function described in Algorithm 1. The scoring function outputs the highest score when an object label and the goal phrase are exact matches. The second highest score is given when the goal phrase and the object label are sub-strings. The third highest score is for the edit distance between the goal phrase and the object label is above some fixed threshold. Finally, we use the cosine similarity between the word embeddings [169] of the goal phrase and the object label to capture synonyms or words capturing similar semantics in different word forms.

7.2.3 NAVIGATE&PREDICT

Another component of the situated knowledge is knowing ‘where’, i.e., where objects are located in the environment and how these locations are connected. Here, we assume that OSMaN has this knowledge of the environment. In practice, this means that an embodied AI system like a household robot would be pre-loaded with the information of where objects are in the house and the map of the house. Later, we also show that OSMaN can acquire this situated knowledge even when it is unavailable.

In the previous step, SCORE OBJECT outputs the highest-scoring object to predict remote object localization. Since OSMaN has the situated knowledge of the environment, it has access to the location of this object. Also, OSMaN knows how locations in the environment are connected. Thus, the navigation trajectory to the goal location can easily be calculated via a shortest path algorithm [50].

7.2.4 OSMaN-Exhaustive Acquiring Situated Knowledge with Exhaustive Search

When the agent has situated knowledge of the environment (i.e., where objects are and the map of the environment), OSMaN’s process of a vision-language task boils down to calculating the shortest path between starting location and the goal location. Since this knowledge may not be known a priori, we propose a simple extension of OSMaN, namely OSMaN-Exhaustive, which can obtain situated knowledge required for remote object localization by exploration.

To learn where objects are and how locations of these objects are connected, OSMaN-Exhaustive builds the map of the environment. This process is visualized in Figure 7.3. OSMaN-Exhaustive explores the environment using the depth-first search algorithm. At each location, the model does two things: (1) it records the objects present in those locations using either ground-truth objects or an object detector, and (2) While exploring, OSMaN-Exhaustive keeps track of the connections between successive locations in an adjacency graph to create a map of the environment. After visiting all locations in the environment, the agent returns to the original location, having obtained situated knowledge of the environment. We should emphasize that this is an exhaustive search baseline – i.e., OSMaN-Exhaustive visits *all locations* in the environment. It is practical and straightforward but not maximally efficient, which aligns well to keep OSMaN as simple as possible.

7.3 Experiments

Our primary motivation in this section is to empirically analyze the use of situated knowledge – both its potential and limitations. We also want to use OSMaN to uncover potential future directions through its simplicity. Thus, we designed our experiments to answer the following research questions(RQ):

- **RQ1:** Does situated knowledge of objects and their locations help? To answer this question, we compare the performance of our simple model OSMaN empowered with the situated knowledge of the environment with state-of-the-art approaches.
- **RQ2:** Can a system obtain situated knowledge, and how does it compare to providing ground-truth situated knowledge upfront? To address this question, we compare different configurations of OSMaN where situated knowledge is acquired through exploring the environment.
- **RQ3:** What are the potential and limitations of situated knowledge? We make several simplifying assumptions for OSMaN, but these could be limiting factors. To answer this question, we design oracle experiments where ground-truth information is provided to OSMaN to demonstrate an upper bound on performance.
- **RQ4:** What are the challenging phenomena in vision-language systems for existing approaches and OSMaN? We do pair-wise analyses of a state-of-the-art system and OSMaN to identify such cases. We find a challenging subset where both of these models fail similarly, namely REVERIE-DeLiRE.

7.3.1 Experimental setup

Dataset. To examine the research questions previously mentioned, we work on the REVERIE dataset [175]. The REVERIE has 10.5K instructions in 60 scenes for training. The validation set includes seen and unseen splits to evaluate generalization and has 4.9K instruction in 56 scenes for seen and 3.5K instructions for 10 scenes. The test set includes 6.3K instructions in 16 scenes. There are total of 4653 objects in the dataset. The ground-truth trajectories and annotations of objects are collected via crowdsourcing human annotators. The REVERIE is built on top of the previous vision-language navigation benchmark, Room2Room [7], and shares the same simulation and evaluation setup. At each location in a scene, the agent observes a panoramic view of the environment. Following the literature[7], for each location in the environment, we divide the panoramic view of the agent into 36 field-of-views. The agent needs to predict one of the navigable views for moving. The agent predicts a bounding box for the goal object for object localization.

Baseline Models. We compare OSMaN with best-performing models from the literature, where each of them uses complex sequence-to-sequence neural architectures [163, 205].

- Navigator-Pointer [175] is the first model proposed for REVERIE. The main idea of the Navigator-Pointer model is that the navigation and referring expression recognition processes should interact with each other. The Pointer module predicts top-3 objects at a location in the environment, and these predictions are fed to the Navigator module.
- RBERT [86] is a Transformer-based navigation model. First, the model encodes language instruction via a language-based Transformer which generates an embedding for each word token. To get a representation for visual input, it uses feature maps of Resnet-152 [80]. For extracting bounding boxes of objects, it uses Faster-RCNN [181]. To keep track of the state of the agent, a state embedding from the previous timestep is also used. Textual, visual, and state representations are then fed to a multi-layer transformer. The final layer of transformer outputs predictions for actions and objects and the agent’s state.
- The Self Monitoring Navigation Agent (SMNA) [143] model has two modules: one for language and the other for vision for encoding textual and visual information, respectively. On top of these, SMNA uses a co-grounding module where both visual and textual input is attended at the same time. The agent also measures its progress with a progress monitor module.
- Frontier Aware Search with backtracking (FAST) [106] model is trained to score partial trajectories of the model to backtrack to a previous location. We use SMNA as the base agent for training FAST. FAST has two versions. In “short” version, the agent backtracks only when it revisits a location. In “long” version, the agent always backtracks to the highest scoring partial trajectory using a re-ranker.

Evaluation Metrics. We measure accuracy and efficiency for comparing systems both of them are desired properties of a vision-language system, i.e., they need to be accurate and efficient at completing the task. We use symbols % and 🕒 represent accuracy and efficiency for metrics,

respectively, in our results. We report the mean scores of 5 runs for each setup and the variances for any metrics do not exceed 0.005 due to the deterministic nature of our heuristic-based modules.

- For navigation, we use Success Rate (SR), i.e., the percentage of the time the system ends up in a location where the target object is visible.
- We use Success Rate Weighted by Path Length (SPL) to measure the efficiency of the system by weighing the length of the navigation path [5].
- Coverage weighted by Length Score (CLS) [100] measures how well the predicted path covers the ground-truth path. We use this metric for pairwise analyses of two systems.
- For remote object localization, we use Remote Grounding Success (RGS), i.e., the percentage of the time the predicted object ID is the same as the ground-truth object.
- Similar to SPL, with the Remote Grounding Success rate weighted by navigation Path Length (RGSPL), we calculate the grounding efficiency by weighing the RGS with the path length of the agent.

Implementation. OSMaN does not have any parameters to train. However, we find hyperparameters such as similarity thresholds for SCORE OBJECT using validation seen via grid search.

To equip OSMaN-Exhaustive with situated knowledge of objects, we use the state-of-the-art object detection system Detectron2[229]. Following the literature[7], for each location in the environment, we divide the panoramic view of the agent into 36 field-of-views. We feed these field-of-views to the Detectron2 and extract bounding boxes of objects and their predicted labels. Since REVERIE has its own object label set, we fine-tuned a model trained on the Visual Genome dataset[118] for 330K iterations. We excluded scenes in validation unseen and test sets to prevent contamination. This model has an accuracy of 58.5% in predicting correct labels for the ground-truth labels for REVERIE object annotations. We provide the situated knowledge to OSMaN in two forms. For the map of the environment, we use adjacency matrix for each location in the house. We either provide the ground-truth in the case of OSMaN or the adjacency matrix learned during the exhaustive search for OSMaN-Exhaustive. For situated knowledge of objects, we provide a dictionary to the system where each location in the map has a list of objects visible in each locations in the environment. We both report results in the original studies and our replications for the main results. We train SMNA, FAST, and RBERT agents from scratch for our replications. All models are trained for 100K iterations. We use Adam [112] for optimization with a learning rate 0.0001 and weight decay parameter 0.0005 [119].

7.3.2 Results and Discussion

(RQ1) Situated knowledge helps OSMaN achieve the state-of-the-art performance. We present results for the experiments where we compared OSMaN with the state-of-the-art in Table 7.1. OSMaN surpasses complex neural architectures by a large margin without training a single parameter for navigation and remote object localization. As of February 2022, OSMaN holds the first position on the official leaderboard of the dataset⁵. We should also note that

⁵<https://eval.ai/web/challenges/challenge-page/606/leaderboard/1683>

Model	Validation Seen				Validation Unseen				Test			
	Navigation		Grounding		Navigation		Grounding		Navigation		Grounding	
	SR	SPL	RGS	RGSP	SR	SPL	RGS	RGSP	SR	SPL	RGS	RGSP
	%	🕒	%	🕒	%	🕒	%	🕒	%	🕒	%	🕒
SMNA [143]	41.3	39.6	30.1	29.0	8.2	6.4	4.5	3.6	5.8	4.5	3.1	2.4
FAST-Short [106]	45.1	40.2	31.4	28.1	10.1	6.2	6.2	4.0	14.2	8.7	7.1	4.5
Navigator-Pointer [175]	50.5	45.5	32.0	28.8	14.4	7.2	7.8	3.9	19.9	11.6	11.3	6.1
RBERT [86]	51.8	48.0	38.2	35.6	30.7	24.9	18.8	15.3	29.6	24.0	16.5	13.5
OSMaN (this work)	60.9	58.9	43.5	41.8	53.4	51.3	36.0	34.6	50.2	47.8	34.7	33.0

Table 7.1: Comparison of OSMaN with the-state-of-the-art. Symbols % and 🕒 represent accuracy and efficiency for metrics, respectively. OSMaN achieves the best results for all evaluation settings and metrics. However, OSMaN is the only model that has access to ground-truth situated knowledge (Please see Table 7.2 for more results where all models have access the same sources of information).

Model	Situated Knowledge		Validation Seen				Validation Unseen			
	Objects	Map	Navigation		Grounding		Navigation		Grounding	
			SR	SPL	RGS	RGSP	SR	SPL	RGS	RGSP
			%	🕒	%	🕒	%	🕒	%	🕒
OSMaN	Ground-Truth Labels	Ground-Truth	60.9	58.9	43.5	41.8	53.4	51.3	36.0	34.6
OSMaN-Predicted	Predicted Labels	Ground-Truth	53.8	<u>52.1</u>	37.0	<u>35.8</u>	37.6	<u>36.2</u>	20.2	<u>19.5</u>
OSMaN-Exhaustive	Predicted Labels	Next Location	<u>53.9</u>	1.4	37.0	1.0	<u>37.8</u>	1.0	<u>20.4</u>	1.0
SMNA [143]	Predicted Labels	Next Location	41.3	39.6	30.1	29.0	8.2	6.4	4.5	3.6
FAST-Short [106]	Predicted Labels	Next Location	45.1	40.2	31.4	28.1	10.1	6.2	6.2	4.0
Navigator-Pointer [175]	Predicted Labels	Next Location	50.5	45.5	32.0	28.8	14.4	7.2	7.8	3.9
RBERT [86]	Predicted Labels	Next Location	51.8	48.0	<u>38.2</u>	35.6	30.7	24.9	18.8	15.3

Table 7.2: Varying the source of situated knowledge for OSMaN. In top-2 rows, OSMaN has access to ground-truth map of the environment. Below that, all models only have access to navigable next locations. **Black** and underlined numbers denote the best and runner-up results for each column, respectively. Even when OSMaN needs to build situated knowledge from scratch it is more accurate than the state-of-the-art approaches (third row).

OSMaN does not suffer from overfitting as much as learning-based agents and its performance is consistent for all evaluation setups. There is more than a 50% decline in their performance for learning-based agents when evaluated in validation unseen and test splits. We designed OSMaN as a simple heuristics-based baseline system to process situated knowledge and it performs surprisingly well. However, OSMaN is the only model that has access to the ground-truth situated knowledge, in the next experimental result we address this by comparing other models with OSMaN and its variants where all models have access to the same sources of information in the environment.

(RQ2) OSMaN-Exhaustive can acquire situated knowledge and is competitive with OSMaN in terms of grounding accuracy. We compare variants of OSMaN where it uses ground-truth and acquired or predicted situated knowledge in Table 7.2. In the first row, OSMaN performs best accuracy and efficiency with readily available situated knowledge of objects and a map of the





Model	Intermediate Output			Validation Seen				Validation Unseen			
				Navigation		Grounding		Navigation		Grounding	
	Object Label	Object Location		SR	SPL	RGS	RGSPL	SR	SPL	RGS	RGSPL
				%		%		%		%	
OSMaN	FIND GOAL	SCORE	OBJECTS	60.9	58.9	43.5	41.8	53.4	51.3	36.0	34.6
OSMaN-Oracle	Oracle	SCORE	OBJECTS	73.6 ^{+12.7}	71.3 ^{+12.4}	56.4 ^{+12.9}	54.3 ^{+12.5}	71.0 ^{+17.6}	68.3 ^{+17.0}	55.8 ^{+19.8}	53.7 ^{+19.1}
OSMaN-Oracle	FIND GOAL	Oracle		89.1 ^{+28.2}	89.1 ^{+30.2}	65.6 ^{+22.1}	65.6 ^{+23.8}	83.4 ^{+30.0}	83.4 ^{+32.1}	58.3 ^{+22.3}	58.3 ^{+23.7}
OSMaN-Oracle	Oracle	Oracle		99.6 ^{+38.7}	99.6 ^{+40.7}	76.3 ^{+32.8}	76.3 ^{+34.5}	98.9 ^{+45.5}	98.9 ^{+47.6}	76.7 ^{+40.7}	76.7 ^{+42.1}

Table 7.3: Oracle experiments for OSMaN where the ground-truth feature is fed to the system. OSMaN modules are accurate and straightforward, but there is still room for improvement. Each oracle feature improves all metrics significantly. Numbers in green show the difference between OSMaN and the oracle version for each metric in their columns.

environment. In the second row, we have OSMaN-Predicted, where the system uses predicted labels of objects. Both the navigation and remote grounding accuracies drop as expected in this setup. However, we should note that it still performs better than neural architectures. In the third row, OSMaN-Exhaustive uses predicted labels for objects and builds the environment map by exploration.

This results in low-efficiency numbers since this is a brute-force approach and the system needs to visit all nodes in the house to build the map of the environment. This result shows that OSMaN-Exhaustive can acquire situated knowledge but not efficiently. However, it still achieves state-of-the-art results in SR and RGS *with only an object detector and heuristics and there is no learning involved*.

(RQ3) There is still a lot of room to use situated knowledge. In Table 7.3, we report results for oracle experiments. We feed oracle features to OSMaN instead of using its modules. These setups correspond to OSMaN-Oracle models, which set the upper bound on OSMaN’s performance. As expected for all OSMaN-Oracle models, we observe an improvement over the baseline OSMaN. This suggests that if we design more accurate FIND GOAL or SCORE OBJECTS modules, OSMaN performance will improve substantially. We should note that even when we have oracle features of goal object label and its location, the upper bound on RGS is less than 100% due to ambiguity of objects in end locations – i.e., there are many objects of the same type. Thus, OSMaN has limited capacity to reach the perfect performance, and it needs to employ a referring expression recognition system such as we discussed in Chapter 3.

(RQ4): A pair of models uncover a challenging subset of the REVERIE. In Figure 7.4, we plot the CLS scores of OSMaN on the x-axis and RBERT on the y-axis for data points in validation unseen split. We choose RBERT because of its high performance, and it is orthogonal to our approach (heuristics-based vs. neural architecture). We also overlay the percentage of the cases for four quadrants. For instance, the top-right quadrant shows cases where both RBERT and OSMaN perform better than the 0.5 CLS score, which corresponds to 27% of the validation unseen split. However, we are interested in the lower-left quadrant where *both* RBERT and OSMaN fail to achieve good performance. RBERT, a high-capacity neural architecture, and OSMaN, a heuristic-based situated agent, both fail on a rel-

	SMNA				FAST				RBERT				OSMaN (this work)			
	SR	SPL	RGS	RG SPL	SR	SPL	RGS	RG SPL	SR	SPL	RGS	RG SPL	SR	SPL	RGS	RG SPL
REVERIE	13.7	6.0	7.9	3.1	15.0	5.4	7.1	2.0	29.4	24.4	19.4	16.0	53.4	51.3	36.0	34.6
REVERIE-DeLiRE	11.8	4.6	5.6	1.9	11.7	3.7	5.5	1.5	9.0	4.4	5.6	2.9	3.5	3.4	2.3	2.3
Performance Δ	1.9	1.4	2.3	1.2	3.3	1.7	1.6	0.5	20.4	20.0	13.8	13.1	49.9	47.9	33.7	32.3

Table 7.4: Comparison of models on REVERIE and our proposed REVERIE-DeLiRE. We identify REVERIE-DeLiRE with OSMaN and RBERT, where they perform poorly. All models’ performances drop when evaluated in the REVERIE-DeLiRE subset.

atively large subset of the validation unseen split. We call this subset REVERIE-DeLiRE.

We also evaluate the performance of FAST and SMNA on the REVERIE-DeLiRE. In Table 7.3, we report the results. For all systems their performance drops significantly for this subset. As expected the performance drops more for both OSMaN and RBERT. Two orthogonal approaches both fail in this subset. High-capacity Transformers and situated knowledge of the environment are not good enough for performing well on this subset. Curiously we ask what do we need then? Hopefully, this result inspires future research to focus on this challenging subset.

7.4 Related Work

The work presented in this chapter is related to the previous work on Referring Expression Recognition (RER), Vision-and-Language Navigation (VLN).

RER poses the task of localizing a target point or object described by a natural language phrase in a visual input. The majority of existing benchmarks studies this task with object bounding boxes as target in 2D images [2, 35, 105, 133, 146, 198, 239]. More recently, in Touchdown [30] and Refer360° [40] the target is a pixel point in a 360° image. In Chapter 6, we studied dRER, where the system observes a partial and dynamic view of the scene and needs to change its view to predict the target pixel point.

VLN is the task of following natural language instructions in a simulated environment. The earlier work [28, 144, 193] is based on synthetic images in small environments. More recently, Anderson et al. [8] introduce Room-to-room (R2R) where observes panoramic photo-realistic scans of a house [25] and a large-scale human annotations for natural language instructions. Variations of this setup study VLN in multi-lingual [120], outdoor [30, 83], dialog [77, 162, 212], and continuous settings [117]. In addition to the methods we described in Section 7.3, many other studies are proposed for VLN. Several of these are based on the sequence-to-sequence neural architectures [16, 205, 218]. As we present in Chapter 5, we

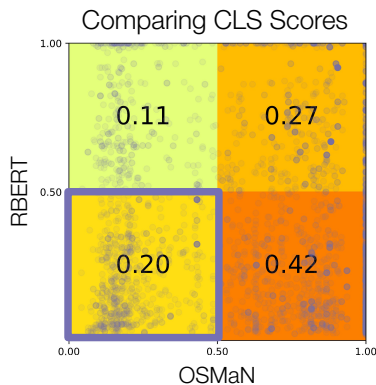


Figure 7.4: Comparison of CLS (i.e., coverage of ground-truth path) scores for OSMaN and RBERT[86]. Purple circles represent a datum in validation unseen split. Numbers at each quadrant show the percentage of the data. The lower-left quadrant in purple square shows REVERIE-DeLiRE instances for which RBERT and OSMaN perform poorly.

study this task with two sequence-to-sequence models: Speaker and Follower. The Speaker model is trained to generate instruction given a trajectory. The Follower model’s possible trajectories are re-ranked using the speaker model. EnvDrop [209] introduces the concept of modality dropping for generalization to unseen environments. PREVALENT [78] uses a variety of pre-training tasks to address the data sparsity problem.

7.5 Conclusion

This chapter studied how situated knowledge of objects and the environment could be helpful for the remote object localization task. We introduced OSMaN, a performant yet straightforward heuristics-based model that relies on situated knowledge, and empirically analyzed this model in the REVERIE benchmark. OSMaN achieves the state-of-the-art remote object localization performance when situated knowledge is readily available to the system. Even when OSMaN acquires the situated knowledge with a brute force algorithm, it still surpasses complex neural architectures in remote grounding accuracy. We also identify potential avenues for improving shortcomings of OSMaN. First, `FIND GOAL` could be improved by using an end-to-end approach rather than a heuristic-based approach. For instance, T5 [180] can be fine-tuned for predicting the goal object. Second, with a state-of-the-art referring expression recognition system [104], OSMaN could improve both accuracy and efficiency by producing more accurate intermediate outputs. Third, with a better search algorithm OSMaN could be more efficient when building the map of the environment. Using OSMaN, we also introduced REVERIE-DeLiRE, a challenging subset for the benchmark task. We show that OSMaN and the state-of-the-art models struggle in REVERIE-DeLiRE. Our analyses on this subset help us better understand the challenges of the task and point the way for future research.

Acknowledgements

This material is based upon work partially supported by National Science Foundation awards 1722822 and 1750439, and National Institutes of Health awards R01MH125740, R01MH096951 and U01MH116925. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors, and no official endorsement should be inferred.

Chapter 8

Conclusion and Discussion

In this thesis, we studied language grounding – i.e., the linguistic phenomena of linking language to the world. This research on language grounding was scaffolded in three technical challenges. First, in spatial grounding, we linked mentions of objects and spatial relationships to referents in visual input. Second, in sequential grounding, we studied the process of relating language units to spatial relationships and actions the AI system needs to take in the world in partially observed environments. Third, we examined how prior knowledge, either general knowledge or situated knowledge, can benefit language grounding systems. To conclude, we first give a brief overview of the contributions of this thesis. Then we discuss the broader impact of our work and its limitations. Lastly, we talk about the future work that can build on top of this thesis.

8.1 Thesis Contributions

Here are the main contributions of this thesis:

- We studied spatial grounding in the referring expression recognition (RER) task. We introduced a technique to uncover biases in a popular RER dataset. We showed that state-of-the-art approaches could exploit this annotation bias rather than modeling cross-object spatial relationships of objects. To model cross-object spatial relationships, we introduced GroundNet. GroundNet uses syntax to detect object mentions and the relationship between objects. We showed that GroundNet accurately identifies all object mentions and uses the spatial relationship of objects effectively
- To study sequential grounding, we introduced a versatile benchmark: Refer360° dataset. We examined annotated data from visual (e.g. where the target locations are located and their spatial relationship to neighboring objects) and linguistic phenomena (e.g. frequency of coreference, egocentric mentions). We also introduced a general model Speaker-Follower for sequential grounding tasks and examined this model in vision-language navigation. Speaker-Follower is built on the Rational Speech Act framework[56, 66] and has substantially improved in vision-language navigation.
- We studied the use of a priori knowledge for grounding. For general knowledge of objects, we proposed HOLM. HOLM relies on large pre-trained LMs to hallucinate objects using the

spatial locations of objects. We showed that LMs capture valuable spatial knowledge about objects. More importantly, vision-language models benefit from this spatial knowledge extracted from LMs with HOLM. We also studied the use of situated knowledge for language grounding. We demonstrated that a complex vision-language task becomes a simple search problem in the presence of situated knowledge. We also showed how to acquire such situated knowledge when it is not readily available.

- In addition to ideas, findings, conceptual frameworks, our work produced a lot of valuable artifacts for the research community. We open-sourced all our empirical setups for all studies: machine learning models, evaluation scripts, simulators, data annotations, tools to annotate data, and more. Hopefully, further research can build on top of our findings and tools we open-sourced.

8.2 Broader Impact and Limitations

In this section, we would like to contextualize our work in its impact on the research field and its limitations and discuss the future work that might address these limitations.

In Chapter 2, we introduced perturbation analyses for referring expression recognition task (RER) [176]. The follow-up and the concurrent work showed that we could design challenging baselines to uncover issues in benchmarks [72, 100, 211]. Similarly, more recent work explores the idea of dynamic benchmarking [110] i.e., benchmarks that evolve in time to capture long-tail phenomena to improve models. However, design of such baselines or analyses all relies on a human expert. Future work may address this issue by *automatically* identifying and categorizing linguistic or visual phenomena that a model exploits or fails to capture and create specific baseline systems or analyses for each of these automatically detected categories.

In Chapter 3, we introduced GroundNet, a syntax-based approach for linking objects mentions and spatial relationships to visual input. The main limitations of our work were (1) its reliance on an off-the-shelf parser and (2) its fixed lexicon of neural modules. Akula et al. [3] showed this limitation and highlighted that modern architectures surpass our syntax-based approach due to their flexibility in forming multi-modal representations. The follow-up work [103, 130] addressed the first limitation and studied learning computation graphs for neural modules rather than using computation graphs determined by a parse-tree. However, the fixed lexicon of human-designed neural modules is still yet to be addressed. Instead, future work could address this by designing a non-parametric method where the number of modules and their designs are learned during training. Finally, a recent study [125] shows that syntax could be helpful in sequential grounding tasks, thus, we hope that GroundNet will be explored in vision-language navigation [7] or vision-language dialog [212] tasks.

In Chapter 4, we proposed a novel benchmark, Refer360° which is a versatile dataset to study natural language understanding, computer vision, and language grounding from different perspectives. An open research question relevant to this work is how a model can form a 360° representation of its environment. Such representation would require an understanding of the depth of space and continuity of 2D visual observation.

In Chapter 5, we introduced the Speaker-Follower model, a pragmatic reasoning model for vision-language navigation. Many follow-up approaches [78, 143, 209] adopted our proposed

high-level action space and have used our data augmentation technique. Kurita and Cho [121] extended our approach and showed that the Speaker model can be used at each timestep during navigation for Bayesian reasoning for alternative actions. The main limitation of the Speaker-Follower model is that it relies on the instructions generated for data augmentations and scored by the Speaker model for reasoning. The Speaker model often hallucinates objects i.e., generating objects mentions that are not present. Future work can address this by enforcing object mentions that are faithful to the environment. Finally, future work could explore the Speaker-Follower model in more interactive language grounding tasks [162, 165, 212].

In Chapter 6, we introduced HOLM for enhancing language grounding systems with general knowledge of objects. HOLM relies on the existing pre-trained language model. The main limitation of HOLM is the hallucinations are conditioned on only current visual information. In future work, HOLM could generate these hallucinations conditioned on the previous visual observations, actions the system has taken, and language instructions.

In Chapter 7, we proposed OSMaN as a strong baseline for remote object localization. OSMaN relies on situated knowledge of the environment. The main limitation of OSMaN is that if situated knowledge is not present, OSMaN-Brute acquires situations itself in the environment with a brute force approach. Future work could address this issue with a heuristic-based search algorithm [79] which could potentially increase the efficiency of OSMaN.

8.3 Future Research Directions

We identified and studied three core challenges in language grounding research. However, future work could explore other aspects that could advance the field. Here, we give a brief overview of potential avenues for advancing language grounding.

Multi-lingual Language Grounding. One shared limitation of the work presented in this thesis is that all benchmarks we used are in English. However, further research is needed to adapt and improve the models we introduced in other languages. For instance, GroundNet relies on parse trees, but much of the spatial relationships are in suffixes for morphologically rich languages such as Turkish. To address this issue, models could extract spatial relationships from sub-word units. Another example would be for OSMaN we designed our heuristics for instructions in English, however, for other languages, we need to adapt our heuristics to capture the grammatical and sub-word structures. Several multi-lingual multimodal benchmarks are available to extend our work to other languages [17, 52, 63, 108, 120, 131, 202]. Another potential limitation is that object detectors we use for visual input rely on (1) annotators in English-speaking western cultures and (2) a fixed lexicon for the labels of objects. However, both objects and how people refer to those objects change in different languages, cultures, and demographics. Further research could focus on these currently invisible dimensions and potentially uncover a more inclusive and holistic view of challenges in language grounding.

Complex Interactions. With the recent advances in building more sophisticated simulated environments [165, 194], we can study more complex interactions with the environment. In

these novel scenarios, the agent can manipulate objects in the environment. Thus, the non-stationary nature of the environment (i.e., objects can be manipulated and moved) poses exciting challenges for embodied AI systems. The future work can build on top the work we presented in this thesis to address complex interactions for language grounding. Potential future directions include reasoning about alternative action spaces with novel techniques for pragmatic reasoning (Chapter 5), hallucinating potential outcome of more complex actions (Chapter 6), or acquiring situated knowledge of affordances of objects (Chapter 7).

Data Augmentation For Language Grounding. The abundance of compute power and data resulted in the recent boom in AI research [108]. Even though recent efforts aim to scale up the vision-language benchmarks [120, 192], we are still far from reaching diminishing returns in terms for the amount of data. Our work in Chapter 5 shows that data augmentation is a viable approach in improving language grounding systems’ performance. Thus, we can address the data scarcity issue with the advances in generative modeling. Recent studies show substantial improvement in natural language generation [22, 180] and image generation [54]. Combining the large collection of unlabeled multi-modal data available on the web and better multi-modal generative models could result in abundance of multi-modal data to augment the datasets we train language grounding models on.

More Prior Knowledge In this thesis, we used focused on a narrow definition of knowledge. In Chapter 6, the general knowledge was based on the spatial properties of objects and their co-occurrences. In Chapter 7, we defined situated knowledge as the spatial location of objects and how these locations are connected to each other. We showed that we could improve or design language grounding systems with prior knowledge. Further research can extend our definitions of knowledge to different types such as physics, social, or historical. As a starting point, knowledge of the world’s physics [15, 124, 224] could be helpful when a language grounding agent interacts in a more complex environment. An accurate model of the physics of objects would reduce the search space when reasoning about actions described in natural language instructions.

Bibliography

- [1] Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the behavior of visual question answering models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960. Association for Computational Linguistics.
- [2] Arjun R Akula, Spandana Gella, Yaser Al-Onaizan, Song-Chun Zhu, and Siva Reddy. 2020. Words aren’t enough, their order matters: On the robustness of grounding visual referring expressions. *arXiv preprint arXiv:2005.01655*.
- [3] Arjun Reddy Akula, Spandana Gella, Yaser Al-Onaizan, Song-Chun Zhu, and Siva Reddy. 2020. Words aren’t enough, their order matters: On the robustness of grounding visual referring expressions. In *ACL*.
- [4] Hazan Anayurt, Sezai Artun Ozyegin, Ulfet Cetin, Utku Aktas, and Sinan Kalkan. 2019. Searching for ambiguous objects in videos using relational referring expressions. In *Proceedings of the British Machine Vision Conference (BMVC)*.
- [5] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.
- [6] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.
- [7] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.
- [8] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [9] Jacob Andreas and Dan Klein. 2015. Alignment-based compositional semantics for instruction following. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

- [10] Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- [11] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to compose neural networks for question answering. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1545–1554, San Diego, California. Association for Computational Linguistics.
- [12] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48.
- [13] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433.
- [14] Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62.
- [15] Tayfun Ates, Muhammed Samil Atesoglu, Cagatay Yigit, Ilker Kesen, Mert Kobas, Erkut Erdem, Aykut Erdem, Tilbe Goksun, and Deniz Yuret. 2020. Craft: A benchmark for causal reasoning about forces and interactions. *arXiv preprint arXiv:2012.04293*.
- [16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- [17] Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. 2018. Findings of the third shared task on multimodal machine translation. In *THIRD CONFERENCE ON MACHINE TRANSLATION (WMT18)*, volume 2, pages 308–327.
- [18] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. ” O’Reilly Media, Inc.”.
- [19] Valts Blukis, Dipendra Misra, Ross A Knepper, and Yoav Artzi. 2018. Mapping navigation instructions to continuous control actions with position-visitation prediction. *arXiv preprint arXiv:1811.04179*.
- [20] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM.
- [21] S.R.K. Branavan, Harr Chen, Luke S. Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 82–90. Association for Computational Linguistics.
- [22] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya

- Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.
- [23] Joyce Y Chai, Pengyu Hong, and Michelle X Zhou. 2004. A probabilistic approach to reference resolution in multimodal user interfaces. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 70–77. ACM.
- [24] Joyce Y Chai, Lanbo She, Rui Fang, Spencer Ottarson, Cody Littlely, Changsong Liu, and Kenneth Hanson. 2014. Collaborative effort towards common ground in situated human-robot dialogue. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 33–40. ACM.
- [25] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*.
- [26] Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367. Association for Computational Linguistics.
- [27] Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.
- [28] David Chen and Raymond Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the AAI Conference on Artificial Intelligence*, volume 25.
- [29] David L. Chen. 2012. Fast online lexicon learning for grounded language acquisition. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1, ACL '12*, pages 430–439, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [30] Howard Chen, Alane Shur, Dipendra Misra, Noah Snaveley, and Yoav Artzi. 2018. Touchdown: Natural language navigation and spatial reasoning in visual street environments. *arXiv preprint arXiv:1811.12354*.
- [31] Kan Chen, Rama Kovvuri, and Ram Nevatia. 2017. Query-guided regression network with context policy for phrase grounding. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- [32] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.
- [33] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. 2013. Neil: Extracting visual knowledge from web data. In *Proceedings of the IEEE international conference on*

computer vision, pages 1409–1416.

- [34] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.
- [35] Zhenfang Chen, Peng Wang, Lin Ma, Kwan-Yee K Wong, and Qi Wu. 2020. Cops-ref: A new dataset and task on compositional referring expression comprehension. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10086–10095.
- [36] Shih-Han Chou, Wei-Lun Chao, Wei-Sheng Lai, Min Sun, and Ming-Hsuan Yang. 2020. Visual question answering on 360° images. In *The IEEE Winter Conference on Applications of Computer Vision*.
- [37] Shih-Han Chou, Yi-Chun Chen, Kuo-Hao Zeng, Hou-Ning Hu, Jianlong Fu, and Min Sun. 2018. Self-view grounding given a narrated 360 video. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- [38] Shih-Han Chou, Cheng Sun, Wen-Yen Chang, Wan-Ting Hsu, Min Sun, and Jianlong Fu. 2020. 360-indoor: Towards learning real-world objects in 360° indoor equirectangular images. In *The IEEE Winter Conference on Applications of Computer Vision*.
- [39] Gordon Christie, Ankit Laddha, Aishwarya Agrawal, Stanislaw Antol, Yash Goyal, Kevin Kochersberger, and Dhruv Batra. 2016. Resolving language and vision ambiguities together: Joint segmentation & prepositional attachment resolution in captioned scenes. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1493–1503. Association for Computational Linguistics.
- [40] Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency. 2020. Refer360: A referring expression recognition dataset in 360: A referring expression recognition dataset in 360 images images. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7189–7202.
- [41] Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Phillippe Morency. 2018. Using syntax to ground referring expressions in natural images. In *32nd AAAI Conference on Artificial Intelligence (AAAI-18)*.
- [42] Volkan Cirik, Louis-Philippe Morency, and Taylor Berg-Kirkpatrick. 2018. Visual referring expression recognition: What do systems actually learn? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 781–787. Association for Computational Linguistics.
- [43] Reuben Cohn-Gordon, Noah Goodman, and Chris Potts. 2018. Pragmatically informative image captioning with character-level reference. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- [44] Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems*, 32:7059–7069.
- [45] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra.

2018. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [46] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2016. Visual dialog. *arXiv preprint arXiv:1611.08669*.
- [47] Chaorui Deng, Qi Wu, Qingyao Wu, Fuyuan Hu, Fan Lyu, and Mingkui Tan. 2018. Visual grounding via accumulated attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7746–7755.
- [48] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- [49] Jacob Devlin, Saurabh Gupta, Ross Girshick, Margaret Mitchell, and C Lawrence Zitnick. 2015. Exploring nearest neighbor approaches for image captioning. *arXiv preprint arXiv:1505.04467*.
- [50] Edsger W Dijkstra et al. 1959. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271.
- [51] Santosh K Divvala, Ali Farhadi, and Carlos Guestrin. 2014. Learning everything about anything: Webly-supervised visual concept learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3270–3277.
- [52] Desmond Elliott, Stella Frank, Khalil Sima’an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.
- [53] Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.
- [54] Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883.
- [55] Rui Fang, Changsong Liu, and Joyce Yue Chai. 2012. Integrating word acquisition and referential grounding towards physical world interaction. In *Proceedings of the 14th ACM international conference on Multimodal interaction*, pages 109–116. ACM.
- [56] Michael C Frank and Noah D Goodman. 2012. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998.
- [57] Michael C Frank, Noah D Goodman, Peter Lai, and Joshua B Tenenbaum. 2009. Informative communication in word production and word learning. In *Proceedings of the Annual Conference of the Cognitive Science Society*.
- [58] Daniel Fried, Jacob Andreas, and Dan Klein. 2018. Unified pragmatic models for generating and following instructions. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- [59] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural*

Information Processing Systems 31, pages 3314–3325. Curran Associates, Inc.

- [60] Andrea Frome, Greg S Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc’ Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: a deep visual-semantic embedding model. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pages 2121–2129.
- [61] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 457–468, Austin, Texas. Association for Computational Linguistics.
- [62] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. 2008. Object categorization using co-occurrence, location and appearance. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.
- [63] Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. 2015. Are you talking to a machine? dataset and methods for multilingual image question. *Advances in neural information processing systems*, 28.
- [64] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256.
- [65] Dave Golland, Percy Liang, and Dan Klein. 2010. A game-theoretic approach to generating spatial descriptions. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 410–419. Association for Computational Linguistics.
- [66] Noah D Goodman and Andreas Stuhlmüller. 2013. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science*, 5(1):173–184.
- [67] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. *arXiv preprint arXiv:1612.00837*.
- [68] Joao Graca, Joana Paulo Pardal, Luísa Coheur, and Diamantino Caseiro. 2008. Building a golden collection of parallel multi-language word alignment. In *LREC*.
- [69] H. P. Grice. 1975. Logic and conversation. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics: Vol. 3: Speech Acts*, pages 41–58. Academic Press, San Diego, CA.
- [70] H Paul Grice. 1975. Logic and conversation. *1975*, pages 41–58.
- [71] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Hui-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.
- [72] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.
- [73] Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. 2017. From language

to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

- [74] David Ha and Jürgen Schmidhuber. 2018. Recurrent world models facilitate policy evolution. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2455–2467.
- [75] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2020. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*.
- [76] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. v. In *International Conference on Machine Learning*, pages 2555–2565. PMLR.
- [77] Meera Hahn, Jacob Krantz, Dhruv Batra, Devi Parikh, James M Rehg, Stefan Lee, and Peter Anderson. 2020. Where are you? localization from embodied dialog. *arXiv preprint arXiv:2011.08277*.
- [78] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146.
- [79] Peter E Hart, Nils J Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107.
- [80] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.
- [81] Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, Marcus Wainwright, Chris Apps, Demis Hassabis, and Phil Blunsom. 2017. Grounded language learning in a simulated 3d world. *CoRR*, abs/1706.06551.
- [82] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.
- [83] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. 2020. Learning to follow directions in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11773–11781.
- [84] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- [85] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. A recurrent vision-and-language bert for navigation. In *CVPR*.

- [86] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vlnbert: A recurrent vision-and-language bert for navigation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1643–1653.
- [87] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python, 2020. URL [https://doi.org/10.5281/zenodo.1212303\(6\)](https://doi.org/10.5281/zenodo.1212303(6)).
- [88] Ronald A Howard. 1960. Dynamic programming and markov processes.
- [89] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. *arXiv preprint arXiv:1704.05526*.
- [90] Ronghang Hu, Daniel Fried, Anna Rohrbach, Dan Klein, Trevor Darrell, and Kate Saenko. 2019. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6551–6557, Florence, Italy. Association for Computational Linguistics.
- [91] Ronghang Hu, Nikhila Ravi, Alexander C Berg, and Deepak Pathak. 2021. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12528–12537.
- [92] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2016. Modeling relationships in referential expressions with compositional modular networks. *arXiv preprint arXiv:1611.09978*.
- [93] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2017. Modeling relationships in referential expressions with compositional modular networks.
- [94] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2017. Modeling relationships in referential expressions with compositional modular networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [95] Ronghang Hu, Marcus Rohrbach, and Trevor Darrell. 2016. Segmentation from natural language expressions. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [96] Ronghang Hu and Amanpreet Singh. 2021. Unit: Multimodal multitask learning with a unified transformer. *arXiv preprint arXiv:2102.10772*.
- [97] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564.
- [98] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [99] Allan Jabri, Armand Joulin, and Laurens van der Maaten. 2016. Revisiting visual question answering baselines. In *European conference on computer vision*, pages 727–739. Springer.
- [100] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1862–1872, Florence, Italy. Association for Computational Linguistics.

- [101] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. *arXiv preprint arXiv:2102.05918*.
- [102] Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.
- [103] Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. *arXiv preprint arXiv:1909.05803*.
- [104] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. 2021. Mdetr–modulated detection for end-to-end multi-modal understanding. *arXiv preprint arXiv:2104.12763*.
- [105] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara L. Berg. 2014. Referit game: Referring to objects in photographs of natural scenes. In *EMNLP*.
- [106] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6741–6749.
- [107] Casey Kennington and David Schlangen. 2017. A simple generative model of incremental reference resolution for situated dialogue. *Computer Speech & Language*, 41:43–67.
- [108] Humair Raj Khan, Deepak Gupta, and Asif Ekbal. 2021. Towards developing a multilingual and code-mixed visual question answering system by knowledge distillation. *arXiv preprint arXiv:2109.04653*.
- [109] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. 2018. Video object segmentation with language referring expressions. In *Asian Conference on Computer Vision*, pages 123–141. Springer.
- [110] Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. 2021. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*.
- [111] Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv preprint arXiv:1909.02950*.
- [112] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.
- [113] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. 2021. Pathdramer: A world model for indoor navigation.
- [114] Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. 2014. What are you talking about? text-to-image coreference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3558–3565.

- [115] Satwik Kottur, Ramakrishna Vedantam, José MF Moura, and Devi Parikh. 2016. Visual word2vec (vis-w2v): Learning visually grounded word embeddings using abstract scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4985–4994.
- [116] Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. 2016. Semantic parsing with semi-supervised sequential autoencoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1078–1087, Austin, Texas. Association for Computational Linguistics.
- [117] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer.
- [118] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.
- [119] Anders Krogh and John A Hertz. 1992. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957.
- [120] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. *arXiv preprint arXiv:2010.07954*.
- [121] Shuhei Kurita and Kyunghyun Cho. 2021. Generative language-grounded policy in vision-and-language navigation with bayes’ rule. *ArXiv*, abs/2009.07783.
- [122] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip HS Torr. 2010. Graph cut based inference with co-occurrence statistics. In *European conference on computer vision*, pages 239–253. Springer.
- [123] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.
- [124] Adam Lerer, Sam Gross, and Rob Fergus. 2016. Learning physical intuition of block towers by example. In *International conference on machine learning*, pages 430–438. PMLR.
- [125] Jialu Li, Hao Hao Tan, and Mohit Bansal. 2021. Improving cross-modal alignment in vision language navigation via syntactic information. In *NAACL*.
- [126] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.
- [127] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.

- [128] Chenxi Liu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, and Alan Yuille. 2017. Recurrent multimodal interaction for referring image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- [129] Daqing Liu, Hanwang Zhang, Feng Wu, and Zheng-Jun Zha. 2019. Learning to assemble neural module tree networks for visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4673–4682.
- [130] Daqing Liu, Hanwang Zhang, Feng Wu, and Zheng-Jun Zha. 2019. Learning to assemble neural module tree networks for visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
- [131] Fangyu Liu, Emanuele Bugliarello, Edoardo Maria Ponti, Siva Reddy, Nigel Collier, and Desmond Elliott. 2021. Visually grounded reasoning across languages and cultures. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10467–10485, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- [132] Jingyu Liu, Liang Wang, and Ming-Hsuan Yang. 2017. Referring expression generation and comprehension via attributes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4856–4864.
- [133] Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L Yuille. 2019. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4185–4194.
- [134] Xihui Liu, Zihao Wang, Jing Shao, Xiaogang Wang, and Hongsheng Li. 2019. Improving referring expression grounding with cross-modal attention-guided erasing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1950–1959.
- [135] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- [136] Yongfei Liu, Bo Wan, Xiaodan Zhu, and Xuming He. 2020. Learning cross-modal context graph for visual grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11645–11652.
- [137] Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.
- [138] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 2016. Visual relationship detection with language priors. In *European Conference on Computer Vision*, pages 852–869. Springer.
- [139] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*.
- [140] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-

- 1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446.
- [141] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*.
- [142] Ruotian Luo and Gregory Shakhnarovich. 2017. Comprehension-guided referring expressions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [143] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019. Self-monitoring navigation agent via auxiliary progress estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- [144] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: Connecting language, knowledge, and action in route instructions. *Def*, 2(6):4.
- [145] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.
- [146] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11–20.
- [147] Kenneth Marino, Xinlei Chen, Devi Parikh, Abhinav Gupta, and Marcus Rohrbach. 2021. Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14111–14121.
- [148] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3195–3204.
- [149] David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Association for Computational Linguistics.
- [150] Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. 2020. Retouch-down: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view. *arXiv preprint arXiv:2001.03671*.
- [151] Hongyuan Mei, Mohit Bansal, and Matthew Walter. 2016. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*.
- [152] Thomas Mensink, Efstratios Gavves, and Cees GM Snoek. 2014. Cosrarta: Co-occurrence statistics for zero-shot classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2441–2448.
- [153] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhersch, and Armand Joulin.

2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.
- [154] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- [155] George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- [156] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. 2018. Mapping instructions to actions in 3d environments with visual goal prediction. *arXiv preprint arXiv:1809.00786*.
- [157] Dipendra Misra, John Langford, and Yoav Artzi. 2017. Mapping instructions and visual observations to actions with reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- [158] Will Monroe, Robert Hawkins, Noah Goodman, and Christopher Potts. 2017. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 5:325–338.
- [159] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany. Association for Computational Linguistics.
- [160] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. 2016. Modeling context between objects for referring expression understanding. In *ECCV*.
- [161] Varun K Nagaraja, Vlad I Morariu, and Larry S Davis. 2016. Modeling context between objects for referring expression understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 792–807. Springer.
- [162] Khanh Nguyen and Hal Daumé III. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. *arXiv preprint arXiv:1909.01871*.
- [163] Samy Bengio Oriol Vinyals, Alexander Toshev and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164.
- [164] Sylwia Ozdowska. 2008. Cross-corpus evaluation of word alignment. In *LREC*.
- [165] Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. 2021. Teach: Task-driven embodied agents that chat. *arXiv preprint arXiv:2110.00534*.
- [166] Bhargavi Paranjape, Julian Michael, Marjan Ghazvininejad, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Prompting contrastive explanations for commonsense reasoning tasks. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4179–4192, Online. Association for Computational Linguistics.
- [167] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory

- Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- [168] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. 2018. Zero-shot visual imitation. *arXiv preprint arXiv:1804.08606*.
- [169] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.
- [170] Bryan Plummer, Liwei Wang, Chris Cervantes, Juan Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- [171] Bryan A. Plummer, Kevin J. Shih, Yichen Li, Ke Xu, Svetlana Lazebnik, Stan Sclaroff, and Kate Saenko. 2018. Revisiting image-language networks for open-ended phrase detection. *arXiv:1811.07212*.
- [172] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649.
- [173] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-bert: Efficient-yet-effective entity embeddings for bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 803–818.
- [174] Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sacheti. 2020. Imagebert: Cross-modal pre-training with large-scale weak-supervised image-text data. *arXiv preprint arXiv:2001.07966*.
- [175] Yuanka Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.
- [176] Yanyuan Qiao, Chaorui Deng, and Qi Wu. 2020. Referring expression comprehension: A survey of methods and datasets. *IEEE Transactions on Multimedia*.
- [177] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. 2007. Objects in context. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE.
- [178] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

- [179] Ilija Radosavovic, Piotr Dollár, Ross Girshick, Georgia Gkioxari, and Kaiming He. 2017. Data distillation: Towards omni-supervised learning. *arXiv preprint arXiv:1712.04440*.
- [180] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.
- [181] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.
- [182] Chris Rockwell, David F. Fouhey, and Justin Johnson. 2021. Pixelsynth: Generating a 3d-consistent experience from a single image. In *ICCV*.
- [183] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. 2016. Grounding of textual phrases in images by reconstruction. In *European Conference on Computer Vision*, pages 817–834. Springer.
- [184] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. 2016. Grounding of textual phrases in images by reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [185] Robin Rombach, Patrick Esser, and Björn Ommer. 2021. Geometry-free view synthesis: Transformers and no 3d priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14356–14366.
- [186] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.
- [187] Fereshteh Sadeghi, Santosh K Kumar Divvala, and Ali Farhadi. 2015. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1456–1464.
- [188] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.
- [189] Thomas Scialom, Patrick Bordes, Paul-Alexis Dray, Jacopo Staiano, and Patrick Gallinari. 2020. What bert sees: Cross-modal transfer for visual question generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 327–337.
- [190] H Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.
- [191] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 86–96.
- [192] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning.

In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

- [193] Nobuyuki Shimizu and Andrew Haas. 2009. Learning to follow navigational route instructions. In *Twenty-First International Joint Conference on Artificial Intelligence*.
- [194] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.
- [195] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.
- [196] Nathaniel J Smith, Noah Goodman, and Michael Frank. 2013. Learning and using language via recursive pragmatic reasoning about other agents. In *Advances in neural information processing systems*, pages 3039–3047.
- [197] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218.
- [198] Florian Strub, Harm de Vries, Jeremie Mary, Bilal Piot, Aaron C. Courville, and Olivier Pietquin. 2017. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*.
- [199] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VLBert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.
- [200] Yu-Chuan Su, Dinesh Jayaraman, and Kristen Grauman. 2016. Pano2vid: Automatic cinematography for watching 360 videos. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*.
- [201] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. 2017. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*.
- [202] Umut Sulubacak, Ozan Caglayan, Stig-Arne Grönroos, Aku Rouhe, Desmond Elliott, Lucia Specia, and Jörg Tiedemann. 2020. Multimodal machine translation through visuals and speech. *Machine Translation*, 34(2):97–147.
- [203] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473.
- [204] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

- [205] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.
- [206] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.
- [207] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.
- [208] Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.
- [209] Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2610–2621.
- [210] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *AAAI*, volume 1, page 2.
- [211] Jesse Thomason, Daniel Gordon, and Yonatan Bisk. 2019. Shifting the baseline: Single modality performance on visual navigation & qa. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- [212] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR.
- [213] Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. *arXiv preprint arXiv:2106.13884*.
- [214] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. 2018. Object referring in visual scene with spoken language. In *Proc. IEEE Winter Conf. on Applications of Computer Vision (WACV)*.
- [215] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- [216] Ramakrishna Vedantam, Samy Bengio, Kevin Murphy, Devi Parikh, and Gal Chechik. 2017. Context-aware captions from context-agnostic supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3.
- [217] Ruben Villegas, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V Le, and Honglak Lee. 2019. High fidelity video prediction with large stochastic recurrent neural networks. *Advances in Neural Information Processing Systems*, 32:81–91.
- [218] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer*

vision and pattern recognition, pages 3156–3164.

- [219] Adam Vogel and Dan Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 806–814.
- [220] Mingzhe Wang, Mahmoud Azab, Noriyuki Kojima, Rada Mihalcea, and Jia Deng. 2016. Structured matching for phrase localization. In *European Conference on Computer Vision*, pages 696–711. Springer.
- [221] Mingzhe Wang, Mahmoud Azab, Noriyuki Kojima, Rada Mihalcea, and Jia Deng. 2016. Structured matching for phrase localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 696–711. Springer.
- [222] Peng Wang, Qi Wu, Jiewei Cao, Chunhua Shen, Lianli Gao, and Anton van den Hengel. 2019. Neighbourhood watch: Referring expression comprehension via language-guided graph attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1960–1968.
- [223] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. *arXiv:1803.07729*.
- [224] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. 2017. Visual interaction networks: Learning a physics simulator from video. *Advances in neural information processing systems*, 30.
- [225] Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. 2017. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*.
- [226] Erik Wijmans and Yasutaka Furukawa. 2017. Exploiting 2d floorplan for building-scale panorama rgb-d alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 308–316.
- [227] Tom Williams, Saurav Acharya, Stephanie Schreitter, and Matthias Scheutz. 2016. Situated open world reference resolution for human-robot dialogue. In *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*, pages 311–318. IEEE Press.
- [228] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- [229] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. <https://github.com/facebookresearch/detectron2>.
- [230] Fanyi Xiao, Leonid Sigal, and Yong Jae Lee. 2017. Weakly-supervised visual grounding of

phrases with linguistic structures.

- [231] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. 2012. Recognizing scene viewpoint using panoramic place representation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2695–2702. IEEE.
- [232] Yanyu Xu, Yanbing Dong, Junru Wu, Zhengzhong Sun, Zhiru Shi, Jingyi Yu, and Shenghua Gao. 2018. Gaze prediction in dynamic 360 immersive videos. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5333–5342.
- [233] Hao Yang and Hui Zhang. 2016. Efficient 3d room shape recovery from a single panorama. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5422–5430.
- [234] Sibeiyang, Guanbin Li, and Yizhou Yu. 2019. Cross-modal relationship inference for grounding referring expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4145–4154.
- [235] Sibeiyang, Guanbin Li, and Yizhou Yu. 2019. Dynamic graph attention for referring expression comprehension. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4644–4653.
- [236] Yang Yang, Shi Jin, Ruiyang Liu, Sing Bing Kang, and Jingyi Yu. 2018. Automatic 3d indoor scene modeling from single panorama. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3926–3934.
- [237] Yezhou Yang, Yi Li, Cornelia Fermuller, and Yiannis Aloimonos. 2015. Neural self talk: Image understanding via continuous questioning and answering. *arXiv preprint arXiv:1512.03460*.
- [238] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. 2018. MATTNET: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [239] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. 2016. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85. Springer.
- [240] Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L Berg. 2017. A joint speaker-listener-reinforcer model for referring expressions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [241] Youngjae Yu, Sangho Lee, Joonil Na, Jaeyun Kang, and Gunhee Kim. 2018. A deep ranking model for spatio-temporal highlight detection from a 360° video. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- [242] Hanwang Zhang, Yulei Niu, and Shih-Fu Chang. 2018. Grounding referring expressions in images by variational context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4158–4166.
- [243] Bolei Zhou, Yuandong Tian, Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. 2015. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*.
- [244] Bohan Zhuang, Qi Wu, Chunhua Shen, Ian Reid, and Anton Van Den Hengel. 2018. Parallel

attention: A unified framework for visual object discovery through dialogs and queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4252–4261.