Procedural Semantics for Human-like Language Understanding in Situated Environments

Verheyen, Lara

Publication date:
2023

Vrije Universiteit Brussel
Faculteit Wetenschappen en
Bio-ingenieurswetenschappen
Departement Computerwetenschappen

Procedural Semantics for Human-like Language Understanding in Situated Environments

Proefschrift voorgelegd tot het behalen van de graad van doctor in de wetenschappen aan de Vrije Universiteit Brussel te verdedigen door

Lara VERHEYEN

Promotoren:
Prof. dr. Paul Van Eecke
Prof. dr. Katrien Beuls

Brussel, 12 december 2023

Printed by
Crazy Copy Center Productions
VUB Pleinlaan 2, 1050 Brussel
Tel : +32 2 629 33 44
crazycopy@vub.be
www.crazycopy.be

ISBN : 9789464443936
NUR CODE : 984
THEMA : UYQL

Jury Members

Chairman	Prof. dr. dr. Geraint Wiggins *Vrije Universiteit Brussel, Belgium*
Secretary	Prof. dr. Beat Signer *Vrije Universiteit Brussel, Belgium*
Internal member	Prof. dr. Stefanie Keulen *Vrije Universiteit Brussel, Belgium*
External members	Dr. Claire Bonial *U.S. Army Research Lab, United States of America*
	Dr. Robert Porzel *Universität Bremen, Germany*
Promotors	Prof. dr. Paul Van Eecke *Vrije Universiteit Brussel, Belgium*
	Prof. dr. Katrien Beuls *Université de Namur, Belgium*

Abstract

In recent years, data-driven approaches have become the predominant paradigm in the field of natural language processing. These approaches mostly rely on statistical patterns inferred from large collections of textual data. Since such systems achieve impressive results on a variety of NLP tasks and because they exhibit high levels of formal and syntactic correctness, it is often assumed that these systems understand language in a human-like way. While this might appear to be the case, these systems primarily focus on the form side of language, since they are mostly learned from textual data that is not grounded in the world. It can therefore be argued that they deal with language in a way that is fundamentally different than the way in which humans do, i.e. by constructing meaning through interactions with each other and their environment.

In this thesis, I investigate how aspects of human-like language understanding can be modelled by building systems, each focussing on different parts of human-like language understanding. This research resulted in three concrete contributions. A first contribution relates to the assumption that language cannot be separated from the environment in which it is used. Concretely, I present a system that is able to ground language in its environment and memory by introducing a procedural semantics that integrates these elements. This novel methodology achieves state-of-the-art results on two benchmark datasets for the task of visual dialogue. A second contribution consists in a model that starts from the idea that language is inherently connected to individual knowledge, since personal experiences shape how humans interpret language. This system integrates language with an agent's personal and dynamic knowledge system. Here, a proof-of-concept implementation demonstrates how agents can come to different interpretations of the same linguistic utterance through their individual knowledge. A third contribution starts from the assumption that language understanding becomes truly human-like when systems can reflect on their own language understanding and signal when they might fail to understand, for

instance due to a lack of knowledge. It concerns the development of a system that allows an agent to monitor its own process of language understanding. This allows an agent to estimate how well it has understood a given text and to identify and signal when it has misunderstood certain aspects. This monitoring system is applied on case studies from two different tasks: a visual dialogue task and a recipe understanding task. These systems illustrate how certain aspects of human-like language understanding can be computationally modelled and thereby provide a more human-like alternative to today's data-driven NLP systems.

Samenvatting

De laatste jaren zijn datagedreven methoden het centrale paradigma binnen het veld van natuurlijketaalverwerking geworden. Deze methoden steunen voornamelijk op statische patronen die ze hebben afgeleid uit grote collecties van tekstuele data. Aangezien deze systemen impressionante resultaten behalen op bepaalde natuurlijketaalverwerkingtaken en omdat ze een hoog niveau van formele en syntactische correctheid vertonen, wordt vaak aangenomen dat deze systemen taal begrijpen op een mensachtige manier. Hoewel dit het geval lijkt, vatten deze modellen voornamelijk enkel de vorm kant van taal, aangezien ze meestal enkel geleerd hebben van tekstuele data die niet verankerd zijn in de wereld. Het is daarom mogelijk om te beargumenteren dat deze modellen op een fundamenteel andere manier omgaan met taal dan hoe mensen ermee omgaan, die betekenis opbouwen door te interageren met elkaar en met hun omgeving.

In deze thesis onderzoek ik hoe bepaalde aspecten van mensachtig taalbegrip gemodelleerd kunnen worden door systemen te bouwen die elk op verschillende delen van mensachtig taalbegrip focussen. Dit onderzoek heeft geleid tot drie concrete contributies. Een eerste contributie gaat over de veronderstelling dat taal niet gescheiden kan worden van de omgeving waarin het gebruikt wordt. Ik introduceer namelijk een systeem dat taal kan verankeren in de omgeving en het geheugen door een procedurele semantiek voor te stellen die deze elementen integreert. Deze nieuwe methodologie behaalt *state-of-the-art* resultaten op twee datasets voor de taak van *visual dialogue*. Een tweede contributie bestaat uit een model dat begint van het idee dat taal inherent verbonden is met individuele kennis, aangezien persoonlijke belevingen de manier waarop mensen taal interpreteren, vormen. Dit systeem integreert taal met het persoonlijke en dynamische kennissysteem van een agent. Een *proof-of-concept* implementatie toont aan hoe verschillende agenten tot verschillende interpretaties van dezelfde talige uiting komen op basis van hun individuele

kennis. Een derde contributie start van het idee dat taalbegrip echt mensachtig wordt als systemen kunnen reflecteren over hun eigen taalbegrip en wanneer ze kunnen aangeven wanneer dit fout gaat, bijvoorbeeld omdat ze kennis missen. Concreet gaat het over de ontwikkeling van een systeem dat ervoor zorgt dat een agent zijn eigen proces van het begrijpen van taal kan monitoren. Dit zorgt ervoor dat een agent kan inschatten hoe goed hij een tekst heeft begrepen en dat hij kan identificeren en signaleren wanneer hij bepaalde aspecten niet begrepen heeft. Dit monitorsysteem is toegepast op gevalstudies uit twee verschillende taken, een *visual dialogue* taak en een taak met betrekking tot het begrijpen van een recept. Deze systemen tonen hoe bepaalde aspecten van mensachtig taalbegrip computationeel gemodelleerd kunnen worden en daardoor bieden ze een meer mensachtig alternatief dan de huidige datagedreven systemen voor natuurlijketaalverwerking.

Acknowledgements

This thesis would not have been possible without the help and support of a vast number of people who I would like to thank for everything they have done.

First and foremost, I would like to thank my two supervisors, Paul Van Eecke and Katrien Beuls for all their support, guidance and advice during these four years. Thank you, Paul and Katrien, for everything you taught me and for helping me become the researcher I am today, you created an environment for me to grow in. The way that you work together and the passion that you have for research is truly inspiring. Looking back, I am very happy that I answered that call from an unknown number in Brussels, expecting it to be a scam call. It turned out to be an amazing opportunity.

Next, I would like to thank my jury members, prof. dr. dr. Geraint Wiggins, prof. dr. Beat Signer, prof. dr. Stefanie Keulen, dr. Claire Bonial and dr. Robert Porzel for reading my thesis in such great detail and providing useful and insightful comments. Your suggestions and questions improved this thesis significantly.

I am very grateful for the opportunity to have worked with Luc Steels and Remi van Trijp. I truly enjoyed working together and I learned a lot from you both. I am looking forward to collaborating more in the future. Remi, thank you for hosting me during a research stay in Paris and for introducing me to all the remarkable people at Sony CSL, I am excited to rejoin you. Luc, your expertise has been invaluable at many occasions and it was a privilege to work with you.

It goes without saying that I have to thank my two closest colleagues, Jens Nevens and Jérôme Botoko Ekila. Jens, I know you had to share your office with me and had to listen to my monologues from early in the morning to the moment you needed to catch your train, thank you for pushing through. I cannot thank you enough for your guidance throughout my entire PhD journey. Thank you for always being there when I wanted to discuss my research and for patiently

answering the same questions over and over again. Also thanks to Jérôme. Your help has been instrumental in pushing my research forward. Indeed, thank you for all your contributions, for your useful comments on the text and for showing me the value of paying attention to detail. I promise I will apply this new-found knowledge when I refactor my code in the future. Thanks to the both of you, I truly enjoyed each and every one of our conversations and I know in my heart that both of you like listening to my fascinating stories.

I truly enjoyed my four years at the AI lab and I would like to thank everyone for creating such a nice environment. I will miss it. Thanks should go to Frederik Himpe and Brigitte Beyens for all their technical and administrative support. Many thanks to Jeroen Van Soest and Tom Willaert. Jeroen, your enthousiasm is contagious and encouraged me to stay positive at many occasions. Thank you Tom, for always being excited to discuss my research and for confronting me with all your critical questions.

Further, I had the opportunity to work with great people from other universities. Thank you, Jonas, Veronica, Liesbet and Alexane for all the interesting discussions and conversations. I am happy that I got to know every single one of you during our offsites and fun activities such as walking and playing sports. Especially, I would like to thank Liesbet for always being a listening ear during our car rides from and to Kessel.

And of course, special thanks to you, Jordi, I could not have undertaken this journey without your continuous support. Thank you for helping me in every way you can and for stimulating me to grasp every opportunity that comes ahead. Without your support, especially during these last months, this thesis would not have been possible.

Finally, I would like to thank my family for all the support that they have provided throughout this journey. A special thanks goes out to my parents, for teaching me to keep challenging myself and for encouraging me to step out of my comfort zone at times. I am sure these lessons were essential to get me to this point. Thank you, zus, for always answering my calls when I needed moral support, I know it felt like a full time job at times. Special thanks to my grandparents for always believing in me. Last but not least, thank you to all the rest of my family and friends, every single one of you has been an inspiration in your own way.

The research presented in this thesis has been funded by the Flemish Government under the 'Flanders AI research program'.

Contents

Chapter 1

Introduction

1.1 Towards human-like language understanding

During the last years, the field of natural language processing (NLP) adopted the use of data-driven, statistical approaches as its main method. This methodology entails extracting statistical patterns from enormous amounts of textual data and use those patterns to solve a wide variety of tasks. In particular generative pre-trained language models or so-called *large language models* are trained to generate text, thereby producing language that reaches an impressive level of fluency and human-likeness. These approaches achieve impressive results on a variety of tasks, for example summarisation, question-answering (see e.g. Zhao et al., 2023, for an overview). Rapid advancements in this field have resulted in

the release of new large-scale models every few months such as GPT-3 (Brown et al., 2020), PaLM (Chowdhery et al., 2022) and LLaMA (Touvron et al., 2023). Also outside of the academic world, large language models have accumulated an immense popularity. Especially ChatGPT, a chatbot introduced by OpenAI that relies on the GPT-3.5 model, has gained a lot of attention. ChatGPT has found its way into society and is adopted by the general public, who use it in a variety of ways, for example as a writing or coding assistant, search engine or chat companion. Moreover, current research investigates how these systems can be applied in fields such as healthcare (see e.g. Sallam, 2023) or education (see e.g. Kasneci et al., 2023), giving them an important role in society and trusting these systems to make crucial decisions that have an impact on human lives.

Given that data-driven, statistical models achieve impressive results on tasks and due to the high level of fluent language that they produce, it is often assumed that these systems are able to understand language in a human-like way. However, it can be argued that the way these systems learn language is fundamentally different from the way humans do (Bender and Koller, 2020). First, these systems are often only given enormous amounts of textual data as input, from which they are trained to extract statistical patterns, thus only capturing the form side of language. Humans, on the other hand, learn language through interactions in the world. For example, children acquire language by interacting with their caregivers in situated environments (Tomasello, 1995), talking and referring to the objects around them in the world. Human language is thus grounded in the world it is used in. Because these large neural models mainly focus on the form side of language and since they do not take the situational context into account, it has been argued that they will not be able to understand language in a human-like way (Bender and Koller, 2020). Second, these systems often overlook a crucial aspect of language: *meaning*. Meaning is inherently connected with language since the basic function of language is to transfer a meaning from a speaker to a listener by means of linguistic utterances (Grice, 1967). In order to make sense of what has been said, the hearer uses all available knowledge to reconstruct the meaning that the speaker wanted to convey. Not only linguistic knowledge can be used in this process, but it can be necessary to use, for example, vision or common-sense knowledge in order to make sense of what has been said. In short, humans are always trying to make sense of a situation and meaning is thus inherently connected with language. Although it may seem that pre-trained language systems capture some sort of semantic knowledge, the semantics are not sufficient to achieve truly intelligent systems, because meaning cannot be learned from textual data

alone, but requires grounding in the world (Bender and Koller, 2020). Therefore, intelligent systems need to be adept with mechanisms for grounding linguistic utterances in other knowledge sources such as knowledge or vision in order to achieve human-like language understanding. Another aspect in which it becomes visible that these large neural systems handle language in a different way than humans are the errors that these models produce. By analysing the output of the models, it becomes clear that the systems make mistakes that humans would never make (Mitchell, 2019). These systems make, for example, mistakes against common-sense knowledge (see e.g. Li et al., 2022; Ettinger, 2020) or are not capable of making pragmatic inferences (Ruis et al., 2023). Lastly, it is often believed within the data-driven NLP community, that the more data that the system is trained on, the better the system will perform. However, training on these large amounts of data has several repercussions, with limited interpretability being one of them. Not only is it hard to understand the model in itself, the data on which it is trained is so large that it becomes impossible to comprehend what is in the training data of the model (Bender et al., 2021; Kaddour et al., 2023), thereby making it a real challenge to control the input data of the model. In contrast, humans are able to learn language from a relatively small amount of data, needing significantly less data than what is required to train a large language model (Frank, 2023).

In short, there are several reasons to assume that large language models do not understand language in a human-like way. However, if these systems are to be used in society as systems that work together with humans for example to support education, or even as systems that make decisions in healthcare, it is crucial that human-like language capabilities are modelled. By building more human-like systems, humans will be able to understand the output and behaviour of the systems better. Moreover, if the systems are interpretable, the decisions of the models can be explained, thereby making it possible to rectify the potential mistakes and maybe even adjust the model, which is certainly necessary in high-risk settings.

In this thesis, I investigate how more human-like systems can be built that, in contrast to statistical, data-driven approaches, are able to understand language in a human-like way. In order to build such systems, three aspects, related to the problems with the systems described earlier, need to be considered, namely grounding, meaning and self-reflection. First of all, language systems cannot be seen independently from the world. They need to be integrated with other sources of knowledge that ground the language into the environment, for example the vision system or knowledge system (Steels, 2023). Secondly,

these systems need to be able to handle meaning, preferably by using explicit meaning representations. Taking inspiration from intelligent systems such as SHRDLU (Winograd, 1972) or LUNAR (Woods et al., 1972) and by building further on more recent systems such as the one introduced in Nevens et al. (2019a), I use procedural semantics as a meaning representation underlying linguistic observations. As introduced by Winograd (1972), Johnson-Laird (1977) and Woods et al. (1972) procedural semantics are meaning representations that are directly executable. These semantics can thus be executed computationally. In order to map linguistic utterances onto their meaning representations, I rely on computational construction grammar, which is an operationalisation of constructionist approaches to language (Fillmore, 1988; Goldberg, 1995, 2003; Croft, 2001). Specifically, I use Fluid Construction Grammar (Steels, 2011, 2017; van Trijp et al., 2022; Beuls and Van Eecke, 2023) to extract the meaning representations underlying the linguistic observations by designing grammars for this task. Once the meaning is retrieved, it can be executed, due to the procedural nature of procedural semantics. By using explicit meaning representations, the systems become more interpretable. Thirdly, human-like language understanding needs a way of reflecting on the understanding process to enable an agent to identify and signal failures in understanding. By modelling the ability to reflect on the understanding process, the systems are capable of indicating when they are not certain of a decision. This is crucial when working together with humans since humans need to be able to trust the output of the systems.

Concretely, this thesis introduces two case studies on modelling human-like language understanding systems using procedural semantics and one additional system that monitors the understanding process. Since the topic of human-like language understanding is quite broad, I limit the scope of this thesis to a selection of aspects of language understanding (discussed in 1.2). Specifically, I focus in the two case studies on (i) grounding linguistic utterances in the memory and vision of an agent and (ii) grounding utterances in the personal knowledge of an agent. The monitoring systems monitors the contribution of the knowledge sources that were consulted during the language understanding process. One of the two case studies introduced in this thesis focusses on building a system that achieves state-of-the-art results on benchmark tasks. The other system focusses on introducing a novel methodology by means of a proof-of-concept implementation, not competing on tasks, but laying the foundations of a model that can model crucial aspects of human-like language understanding. It is also necessary that the monitoring system is applicable on multiple tasks, therefore I apply this system on two widely different systems. All these systems are

integrated within the broader Babel framework (Loetzsch et al., 2008; Steels and Loetzsch, 2010; Nevens et al., 2019b), which provides a solid foundation for research and experiments in the field of language games.

Next, I will elaborate on the main objective of this thesis as well as the three concrete contributions.

1.2 Objective and contributions

The main aim of this thesis is to investigate how intelligent systems that perform human-like language understanding can be built. Since language understanding is a broad term, I focus on a selection of aspects of human-like language understanding as requirements for the systems. A first requirement relates to using meaning representations, as meaning is a central component to human-like language understanding. More specifically, I use procedural semantics, which has the advantage of being directly executable, making it an ideal choice of meaning representation in intelligent systems. Second, in contrast to data-driven approaches that are, for the most part, based on textual information, the systems in this thesis need to be able to ground language in the world as well as in their previously acquired knowledge. Concretely, I introduce two systems that can ground linguistic utterances in the situational environment and previously acquired knowledge using procedural semantics. These systems are discussed in Chapter 4 and 5 respectively. Moreover, I investigate how a system that models the human-like capability of reflecting on its own understanding process can be made. Humans are able to signal when they fail to understand and can ask for further explanation if necessary. This system is introduced in Chapter 6. A last requirement is the interpretability of the systems. All systems are preferably human-interpretable, thereby making the systems explainable. The research towards modelling these requirements in language understanding systems resulted in three concrete contributions.

1.2.1 Contribution: A system for grounding language in the world and memory of an agent

A first contribution of this thesis consists in a system that is able to ground language in the world and memory. This relates to the human-like aspect that linguistic utterances cannot be separated from the environment that they are not uttered in. To achieve this, a case study on challenging NLP benchmarks datasets for the task of visual dialogue (Das et al., 2017) was set up. This task

requires an agent to ground the questions both in the image that is provided as well as in the history of the dialogue. The goal is to build a system that has human-like language understanding capabilities while being able to compete with statistical, data-driven approaches on the visual dialogue challenge.

The visual dialogue task as introduced by Das et al. (2017) requires an agent to answer a series of questions about an image. The questions require the agent to be able to ground the question into the image. Moreover, since the task is to answer a series of questions, each question needs to be embedded into the larger context of the dialogue, which requires grounding the language in the memory of the agent. To solve the task, I introduce a procedural semantics that integrates language, the image and the memory. Moreover, the procedural semantics can be executed in a neuro-symbolic way, combining the strengths of both the neural and symbolic approaches. In order to retrieve the procedural semantics, I designed a grammar that can map between the linguistic utterances in the dataset (i.e. the *captions* and questions) to their meaning representations.

This contribution is discussed in detail in Chapter 4, and is supported by a web demonstration (Verheyen et al., 2022b), which can be found here: `https://ehai.ai.vub.ac.be/demos/visual-dialog/`. The research resulted in the following papers:

- Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2023). Neuro-symbolic procedural semantics for reasoning-intensive visual dialogue tasks. In Gal, K., Nowé, A., Nalepa, G. J., Fairstein, R., and Rădulescu, R., editors, *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI 2023)*, pages 2419–2426, Amsterdam, Netherlands. IOS Press
- Verheyen, L., Botoko Ekila, J., Nevens, J., Beuls, K., and Van Eecke, P. (Under review). Neuro-symbolic procedural semantics for explainable visual dialogue

1.2.2 Contribution: A system for grounding language in the personal dynamic memory of an agent

A second contribution of the thesis is a system that can ground language in the personal dynamic memory of an agent. As a result, different agents can interpret the same utterance differently, coming to different conclusions. Concretely, I introduce a system that is able to ground language into a knowledge and belief system, through a procedural semantics representation that is based on frames (Fillmore, 1976; Fillmore and Baker, 2001). This frame-based meaning represen-

tation can straightforwardly be integrated with the knowledge in the personal dynamic memory of an agent. The procedural nature of the representation comes from the fact that it is used by a reasoning engine to reason over the knowledge of the agent. The meaning representation underlying the linguistic utterances is retrieved by a designed grammar. During the interpretation of the language with regards to an agent's knowledge system, a narrative is constructed. This narrative is a personal view of the linguistic observation at hand and is rooted in the agent's knowledge system. By integrating language with a personal dynamic memory of an agent, the narratives that are constructed become truly individual and personal. In contrast to the previous contribution, the contribution of this system lies not in a system that can compete with state-of-the-art models on benchmark challenges. Instead, it lies in a proof-of-concept system that is able to combine crucial aspects of human-like language understanding, thereby laying the foundations of how such a system would be achieved.

I expand on this contribution in Chapter 5. The research resulted in the following paper:

- Van Eecke, P., Verheyen, L., Willaert, T., and Beuls, K. (2023a). The Candide model: How narratives emerge where observations meet beliefs. In Akoury, N., Clark, E., Iyyer, M., Chaturvedi, S., Brahman, F., and Chandu, K., editors, *Proceedings of the 5th Workshop on Narrative Understanding (WNU)*, pages 48–57. Association for Computational Linguistics

1.2.3 Contribution: A system for monitoring the understanding process of an agent

The last contribution lies in a monitoring system that tracks the understanding process of an agent. It relates to the human-like capability of reflecting on their own understanding process. When humans are trying to understand the situation at hand, they can identify when the understanding goes sideways and even identify where the understanding went wrong, sometimes even by identifying which type of knowledge was missing. During the language understanding process, different knowledge sources need to be consulted.

Concretely, I introduce a novel data structure, the Integrative Narrative Network (INN), which captures all information in the form of *narrative questions and answers* that come from the different knowledge sources that are consulted during the understanding process. The INN thus represents what the agent

understood during the understanding process. Through this network, the agent can identify possible knowledge gaps, which occurs when there are crucial narrative questions that are left unanswered. Moreover, the INNs at the end of the understanding process can be used to quantify how much an agent understood during a task.

Chapter 6 discusses this contribution in detail. It resulted in the following papers:

- Steels, L., Verheyen, L., and van Trijp, R. (2022a). An experiment in measuring understanding. In *Workshop on semantic techniques for narrative-based understanding: Workshop at IJCAI-ECAI 2022*, pages 36–42
- Steels, L., Verheyen, L., and van Trijp, R. (2022b). An experiment in measuring understanding. In Schlobach, S., Pérez-Ortiz, M., and Tielman, M., editors, *HHAI2022: Augmenting Human Intellect. Proceedings of the First International Conference on Hybrid Human-Artificial Intelligence*, pages 241–242. Frontiers in Artifical Intelligence and Applications
- Steels, L., Verheyen, L., and van Trijp, R. (Under Review). Integrative narrative networks. *Journal of Artificial Intelligence Research*

1.3 Structure of the thesis

The remainder of the thesis is structured as follows. Chapter 2 discusses the background on the topics of language understanding, procedural semantics and construction grammar. The technical foundations that underlie the thesis are introduced in Chapter 3. In Chapter 4, I introduced a neuro-symbolic procedural semantics to tackle a visual dialogue task (see contribution 1). Chapter 5 focusses on the second contribution that consists in a system that introduces a frame-based procedural semantics grounded in the knowledge system of an agent. The last contribution of a monitoring system through the use of Integrative Narrative Networks is discussed in Chapter 6. Lastly, the conclusions and future work are discussed in Chapter 7.

Chapter 2

Background

2.1 Introduction

In this chapter, I lay out the conceptual foundations underlying this thesis. The literature discussed here provides the background that is shared among the chapters. Each of the individual chapters (i.e. Chapter 4, 5 and 6) has its own background section, discussing the literature that is relevant for the research

presented in that chapter.

In Section 2.2, I discuss the literature on human-like language understanding. I start the discussion by briefly touching upon some well-known language understanding systems. Then, I discuss what human-like language understanding entails. I argue that it is the process of interpreting language in which meaning is built up by consulting relevant sources of knowledge. To operationalise this process, it is necessary to keep several crucial components in mind. First, truly human-like language understanding systems need to be able to represent meaning. Second, these intelligent systems need a way of accurately modelling their knowledge and context. Lastly, the systems need a way of parsing the linguistic utterances to their respective meaning representations. In Section 2.3, I discuss one of these meaning representations, namely procedural semantics, in more detail. Crucially, procedural semantics are meaning representations that can be executed computationally. Due to this direct executability, the meaning representations can be integrated in intelligent systems in a straightforward way, without first needing to transform the meaning representations into an executable format. In Section 2.4, I discuss construction grammar, which is a linguistic theory that starts from the assumption that all linguistic knowledge is captured in form-meaning mappings called constructions. Due to this focus on meaning, construction grammar, in contrast to more generative approaches to language that often exclude semantics, is well-suited to operationalise language processing in human-like language understanding systems. Both procedural semantics as well as construction grammar are two theoretical frameworks that exhibit well-suited properties to operationalise the language processing in truly intelligent systems. Therefore, I use both approaches as a theoretical foundation as well as the operationalisation (discussed in Chapter 3) of these approaches in the remainder of the thesis.

2.2 Human-like language understanding

Ever since the beginning of the field of AI, building systems that can communicate with humans has had a central position. In particular, the Turing test (Turing, 1950) is one of the first and arguably most important challenges introduced in the field. This test was proposed as a way to measure intelligence, and in the test a human evaluator needs to hold a conversation with both an AI system and another human.[1] The AI system is said to pass the Turing test if the human evaluator cannot tell the difference between its two interlocutors. Although

[1] Originally, the test was described to be about distinguishing a man and a woman.

this test is arguably one of the most well-know and impactful tests in AI, there has been a lot of debate on what this test actually measures. The question is whether the test measures the capacities of intelligence, involving language understanding, or the capability of mimicking language use (see e.g. Pinar Saygin et al., 2000, for an overview of the debate). Indeed, is it necessary to understand language to hold a conversation and pass for a human, or is it sufficient to be able to mimic language use based on the patterns derived from human language?

During these first years, many interesting systems for human-like language understanding were developed. Winograd (1972) introduced SHRDLU, a system that takes requests in natural language to perform actions in a blocks world and answer questions about the state of the world. Woods et al. (1972) introduced a system that could answer questions by retrieving information from a database on moon rocks. One of the first systems that was able to *chitchat* with humans was ELIZA (Weizenbaum, 1983). ELIZA took the role of a psychologist and was able to hold a conversation with a human. The system was remarkably impressive, and many believed that ELIZA has human capabilities, such as language understanding or emotion detection. This was, however, not the case. The system was cleverly designed and made use of mechanisms such as keyword detection as well as a script to respond to the user, which made the users feel like the system actually understood them. The tendency of humans to attribute human capacities to an AI system is called the ELIZA-effect.

Nowadays, large language models are often said to perform human-like language understanding. Transformer-based models (Vaswani et al., 2017) such as GPT (Radford et al., 2018), BERT (Devlin et al., 2018) and their descendants, are trained on incredibly large amounts of textual data and are able to generate language that resembles human language to an expressive extent. However, partly due to the ELIZA-effect, people now attribute human capacities such as understanding and even consciousness to these models (Mitchell and Krakauer, 2023), thereby anthropomorphising these systems and crediting them with capabilities than they don't necessarily have (Shanahan, 2022). It is argued by Bender and Koller (2020) that these systems are not capable of understanding in a truly human-like way, since they learn language in a fundamentally different way from the way humans do. These systems are trained on enormous amounts of textual data for the task of next-word prediction. After these large-scale models are trained, techniques such as fine-tuning or zero- or few-shot learning (Wei et al., 2022a; Brown et al., 2020) are applied to these models so that they can 'tackle' other types of tasks, such as question-answering or machine translation. More-

over, chain-of-thought prompting (Wei et al., 2022c) is used to elicit the model to reason. It is believed that these models have human reasoning capacities (Bisk et al., 2020b; Wei et al., 2022c; Cobbe et al., 2021). It is, however, important to keep in mind that the underlying task that these models were trained on is still next-word prediction. Moreover, by analysing the output of these models, it becomes clear that they lack crucial human-like aspects, for example common-sense knowledge (see e.g. Li et al., 2022; Ettinger, 2020), reasoning capabilities (see e.g. McKenna et al., 2023; Bender and Koller, 2020; Gendron et al., 2023), the ability to make implications (Ruis et al., 2023), consistency over factual knowledge (Elazar et al., 2021), logical deduction (Berglund et al., 2023) and grounding (Collins et al., 2022). There is, on the other hand, a large support for these systems, saying that they *are* capable of human-like language and claiming that these systems even shown 'emergent' abilities (i.e. abilities that show up in large models that were not attested for in smaller models) (Wei et al., 2022b), for example the ability to answer questions through prompting techniques. Lu et al. (2023) dispute these capabilities by stating that it is just in-context learning. Indeed, how would it be possible that a system that is trained to generate text suddenly have the capability of truly understanding? Of course, due to the many statistical patterns that are present in text and due to the syntactic patterns, that, to a large extent, resemble semantic patterns, it seems like these systems are able to understand language (Titus, forthcoming).

The question thus rises whether these systems perform human-like language understanding or whether they are merely mimicking language use. In order to answer this question, it is necessary to understand what human-like language understanding entails. In what follows, I will first discuss how we can come to a definition of human-like language understanding (Section 2.2.1). In Section 2.2.2, I will discuss how this process can be operationalised.

2.2.1 Towards a definition of human-like language understanding

Although language understanding comes naturally to humans, it is not hard to define. Several definitions of language understanding have been proposed throughout the years. Recently, based on their respective literature reviews, Blaha et al. (2022) and Hough and Gluck (2019) define understanding as follows:

> "Understanding is an ongoing cognitive activity of acquiring, integrating and expressing knowledge according to the task or situation at hand." (Blaha et al., 2022, p. 3)

> "The acquisition, organization, and appropriate use of knowledge to produce a response directed towards a goal, when that action is taken with awareness of its perceived purpose." (Hough and Gluck, 2019, p. 23).

From these definitions, it becomes clear that language understanding involves many aspects. I want to highlight a few of them.

First, language understanding is a process, and not only an end result. It is the process of interpreting language, while trying to make sense of the language at hand. That is, language is uttered with the purpose of conveying a particular intention to a listener in a specific context (Grice, 1967) and the listener is tasked to reconstruct this intention. Crucially, this is done based on the available context and the personal knowledge of a human, leading to a personal and individual interpretation.

Second, the 'appropriate use of knowledge' is needed to enable language understanding since language understanding is about the process of constructing meaning based on the relevant information, knowledge, common sense and beliefs of an individual. It is thus necessary to use these types of knowledge to come to an interpretation. During this language understanding process, a rich model of the utterance is constructed which integrates the information extracted from the observation itself, with information from the other sources that were consulted during the process (Steels, 2022b). Steels (2022a) argues that both 'fast' and 'slow' thinking (Kahneman, 2011) are required in the language understanding process. Fast thinking is a form of reactive intelligence. This type of thinking is a reaction to a stimulus, a response that is given without thinking. Slow thinking on the other hand is deliberative. During this type of thinking, a rich model of the situation is built up and from this model, inferences and conclusions can be made. Thus, the deliberative mode builds up a rich model of the observed utterance during language understanding. The reactive mode contributes to this process by providing information from cues that evoke a fast response, for example, perception (e.g. a fast recognition of a visual stimulus). It is thus necessary to combine more reactive techniques to AI (e.g. neural networks) together with more deliberative techniques (e.g. knowledge representation and reasoning) (Steels, 2022a). Neuro-symbolic techniques are well-suited for this type of integration (see e.g. Hitzler and Sarker, 2022, for an overview of the state-of-the-art).

Third, it is necessary to interpret the language with respect to the context, i.e. the situational context as well as the discourse context, it is uttered in. Language is

grounded in the world and is learned through the experiences and interactions of humans (Bisk et al., 2020a; Tomasello, 2003). Grounding the language in the world is thus crucial. For example, when someone calls out "*Duck!*", it depends on the situational context whether the speaker meant to call an animal or to warn someone to duck. Moreover, in addition to considering the situational context, discourse must also be considered (Winograd, 2001), for example to disambiguate utterances. Again, it is necessary to consult the necessary knowledge sources to interpret the context. In discourse, the ambiguity may lay in the use of pronouns. If a speaker says, "*He is fine now!*" after a conversation in which two persons were mentioned, with one of them being ill, the "*he*" will most likely refer back to the person that was previously mentioned to be ill.

Fourth, an aspect that is not captured explicitly in the definitions above is that the process of understanding is personal. It is based on an individual's own knowledge, beliefs and perception of the situation. Therefore, it is often the case that humans come to different interpretation, based on their own views on the situation. Crucially, this interpretation process involves narratives (Bruner, 1991). Narratives are based on the personal experiences of the individual and describe a specific situation or event and are used to make sense on the situation at hand. Recently, there has been a focus on formalising narratives based on the task and system at hand (Porzel, 2021). This enables the modelling of narratives in language understanding systems. However, due to the personal and individual nature of this understanding process, it is hard to evaluate. Because there is no ground truth, it is the consistency of the interpretation that needs to be taken into account.

To summarise, human-like language understanding thus involves the process of interpreting language using the appropriate knowledge sources based on both the situational and the discourse context, thereby building up meaning. Certainly, humans are always trying to make sense of the situation at hand in their own personal and individual way. This sense-making process often requires the consultation of several knowledge sources and flows in many directions. It can for example be the case that the interpretation of a situation requires common sense knowledge which in its turn requires knowledge from the visual context.

2.2.2 Operationalising human-like language understanding

Language understanding for humans is a complex process in which different types of information need to be consulted with information flowing in many

directions. To operationalise this process in intelligent systems, it becomes necessary to break it down into several sub-processes. As a first step, a language processing system parses the utterances to their meaning representation. Then, these meaning representations are executed or 'interpreted', thereby consulting knowledge sources when necessary. To operationalise these processes, three components are required. First, meaning needs to be represented in a way that it can be used and executed by intelligent systems. Second, it is necessary to model the knowledge of the system. Lastly, it is necessary to implement a language processing system that can parse linguistic utterances into their meaning.

Meaning representations

In language understanding systems, the semantics underlying the form side of language are often modelled through meaning representations. These meaning representations can broadly be divided in two categories: formal meaning representations and distributional meaning representations. Formal meaning representations focus on representing the logical meaning of sentences, while distributional approaches capture the meaning of words in terms of their distribution. I will briefly discuss well-known examples of both approaches.

Formal meaning representations capture the meaning of sentences in the form of logical statements. Well-known formal meaning representations include Abstract Meaning Representation (AMR; Banarescu et al., 2013), Discourse Representation Theory (DRT; Kamp and Reyle, 2013), frame semantics (Fillmore, 1976) and procedural semantics (Winograd, 1972; Woods, 1968; Woods et al., 1972). First, Abstract Meaning Representation (AMR; Banarescu et al., 2013) is a meaning representation that captures the semantics of English sentences in rooted, labelled, directed, acyclic graphs. It makes use of the PropBank framesets (Palmer et al., 2005) which include frames and their arguments. AMR does not capture tense, aspect and does not go beyond the sentence level. Dialogue-AMR (Bonial et al., 2019) is an extension of the standard AMR designed to facilitate human-robot dialogue. It captures the illocutionary force of language through speech acts and takes into account tense and aspect. Another meaning representation that goes beyond the sentence level is Discourse Representation Theory (DRT; Kamp and Reyle, 2013). DRT is an approach to semantics with a focus on representing the discourse structure. It is a logic representation representing a mental representation of the discourse. It starts from the discourse referents, which are the entities that are introduced in the discourse. These referents are subject to certain conditions, which represent the information

that has been given throughout the discourse. Throughout the sentences, discourse referents can be made equal, building up the information that is gained. Next, frame semantics (Fillmore, 1976; Fillmore and Baker, 2001) capture the meaning in terms of frames and slots. A frame can be seen as a schematic representation capturing a perspective on a situation. A frame is accompanied by its participants. Frames are 'evoked' by words, meaning that when a certain lexical item is heard, a frame is raised mentally. The idea of semantic frames led to the FrameNet project (Baker et al., 1998), in which they are trying to identify semantic frames together with an annotated corpus. Lastly, procedural semantics (Winograd, 1972; Woods, 1968; Woods et al., 1972) can be seen as an umbrella term for meaning representations that are directly executable. The meaning can take the form of the steps that are needed to find a solution to the utterance, for example in the systems introduced by Winograd (1972); Woods et al. (1972); Nevens et al. (2019a). Meaning representations can also consists of logical statements that can then be executed. Winograd (1975) discusses the procedural aspects of frame semantics, and in Chapter 5, I will introduce a frame-based meaning representation that can be executed through a logic inference system, which can thus be seen as a procedural semantics.

Distributional semantics starts from the assumption that the meaning of a word is based on the distribution it occurs in. "You shall know a word by the company it keeps." (Firth, 1957) is often cited in this context. These 'statistics-of-occurrence' (Titus, forthcoming) approaches follow the distributional hypothesis, namely that the meaning of a word comes from its occurrence. These meanings consists in the embeddings that are learned by training a model to predict the next word(s) based on large text corpora. Examples of such word embeddings include CBOW, Skip-gram and GloVe (Mikolov et al., 2013a,b; Pennington et al., 2014). Moreover, models such as GPT (Radford et al., 2018) or BERT (Devlin et al., 2018) can be attributed to this category. These models are able to take into account more context into the embedding, called 'contextual embeddings', thereby achieving impressive results on a variety of tasks (see Liu et al., 2020, for an overview). The learned embeddings are often believed to contain semantic information. It can be argued, however, that these embeddings cannot contain true semantic information and they only seem to capture meaning, since they are learned from textual data alone (Bender and Koller, 2020; Wu et al., 2021), the generation is not driven by semantic properties (Titus, forthcoming) and they are not linked to beliefs, perception and the world (Lake and Murphy, 2023).

In the remainder of the thesis, I use formal semantics and more specifically procedural semantics as the meaning representation underlying the can be

argued that distributional approaches appear to capture meaning since they only take the form side of language into account. Formal semantics start from a more human-like perspective on meaning. They capture the semantics in the form of logical statements that can be used for reasoning. Moreover, due to the directly executability of procedural semantics, it is highly adequate to use in intelligent systems. While other formal semantics are very successful in capturing and representing the meaning, they often need another step to operationalise them in the systems. By using procedural semantics, the need for such a step is removed. In Section 2.3, I will dive deeper into detail on this meaning representation. In Chapter 4 and 5, I introduce procedural semantics representations for the tasks of visual dialogue and narrative construction.

Representing knowledge and context

Beside meaning representations, it is necessary to model the knowledge of a system as well as the context. As Porzel (2010) argues, it is necessary to account for formal models of knowledge and context in order to build systems that have the same robustness of human languages, which includes the ability to resolve ambiguities, underspecification and noise. Knowledge is thus a crucial component of human-like language understanding systems since the systems need to be able to consult this knowledge during the interpretation process. I use 'knowledge' is a very broad sense, ranging from factual knowledge about the world, to common sense, beliefs, concepts and procedures that are able to ground the language into the world.

In general, a distinction can be made between semantic knowledge and episodic knowledge (Tulving, 1972). Semantic knowledge is about the commonly known facts, about the factual information. Episodic knowledge (Tulving, 1972) on the other hand is a form of 'remembering' the past experiences of an individual. This type of knowledge is about the events that a person has experienced, often containing perceptual and temporal information. One of the ways in which this information is represented is in terms of frames as introduced by Minsky (1974) and later developed by Fillmore (1976); Fillmore and Baker (2001) as a linguistic theory. Frames are schemas that contain slots that represent the frame's participants. Often, frames are the data structure that is used to represent the knowledge of a system. The goal of modelling semantic memories led to the development of large knowledge bases and ontologies that aim to capture on the one hand the information that is available on the web, such as Wikidata (Vrandečić and Krötzsch, 2014) and DBpedia (Lehmann et al., 2015) and on the other hand common sense knowledge, such as Cyc (Lenat, 1995).

However, evaluating these ontologies is a hard task, and often human experts are consulted (see e.g. Gómez-Pérez, 1999). Porzel and Malaka (2004) propose a way of quantitatively evaluating ontologies. The challenge is here that these knowledge bases need to be complete and correct in their representation of knowledge. The knowledge bases can then be consulted by query languages such as SPARQL and SQL to enable the use of this knowledge in the language understanding system. Crucially, the knowledge of humans is not fixed, as they learn over time. The memory of humans is dynamic (Schank, 1983), by adding new information over time as new experiences come in.

Furthermore, context cannot be separated from the process of language understanding, since language is grounded in the world as well as the discourse. It is thus necessary that the context is taken into account when dealing with language. Early language understanding systems such as SHRDLU (Winograd, 1972) model the context (i.e. a blocks world) in a symbolic way. The discourse is modelled through a list that keeps the entities that are mentioned. In later neural approaches, grounding is often done through the use of a part of the model that encaptures the information in the context. This is then integrated with the remainder of the model. For example, models that are introduced for the task of visual dialogue (Das et al., 2017), which involves answering a series of questions about visual input, use encoder-decoder based models to capture both the discourse as well as the image (see e.g. Das et al. (2017); Lu et al. (2017); Wu et al. (2018). Also the recent transformer-based systems are taking into account the context, for example by training on a dataset that contains both language as well as visual input, resulting in the so-called large vision-language models, for example, the well-known DALL-E system (Ramesh et al., 2022) (see e.g. Du et al., 2022, for an overview of other systems).

It is thus essential to accurately model both the knowledge that a system possesses as well as the context that the language is uttered in. In this thesis, modelling these two aspects and consulting them during the execution of the procedural semantics representation takes a central role. In Chapter 4, I model both the situational (i.e. an image) as the discourse context in a symbolic way. Both these representations can be consulted during the execution of the procedural semantics. In Chapter 5, I introduce the concept of a personal dynamic memory that stores all knowledge that an agent holds, including factual knowledge and beliefs, in the form of a frame-based representation. This is consulted through the logic inference that takes place during the execution of the meaning representation. In Chapter 6, I introduce the Integrative Narrative Networks, a data structure that captures the understanding process of the agent by integrat-

ing the contributions from the different knowledge sources that are needed to understand the language at hand.

Parsing

In order to build systems for language understanding, adequate parsing systems are needed. Systems for parsing language can be divided into two broad categories: (i) syntactic parsing systems, which mainly focus on retrieving the syntactic structure underlying a natural language utterance and (ii) semantic parsing systems, which focus on retrieving the semantic representation underlying a natural language utterance. Since meaning representation are a central component of this thesis, I focus on semantic parsing techniques. In particular, systems that are able to map natural language *queries* onto an *executable* meaning representation are of interest. In what follows, I will give an overview of parsing techniques, with a specific focus on systems that are able to retrieve a meaning representation underlying the natural language utterances.

Parsing language has known a long history with a variety of approaches. One of the most well-known approaches is generative grammar, pioneered by Chomsky (1956). This approach focuses on designing grammars that capture the innate grammatical structure of well-formed linguistic expressions. This grammatical structure is captured in terms of a hierarchical tree structure built through the application of rules that combine constituents. These rules allow to generate and parse only the *well-formed, grammatical* utterances of a language. These grammars are also called constituency grammars or phrase structure grammars. On a computational level, this process is operationalised through context-free grammars. Context-free grammars consist of (i) a lexicon that defines the words and symbols in the language and (ii) a set of rewrite rules that determine how symbols can be combined. For example, the lexical elements "*the*" (DETERMINER) and "*cat*" (NOUN) can be combined through the rules NOMINAL $\rightarrow$ NOUN and NP $\rightarrow$ DETERMINER NOMINAL. Although the main focus of context-free grammars is on syntactic parsing, variations on context-free grammars have been introduced throughout the years to map natural language onto their meaning representations (see e.g. Wong and Mooney, 2007; Huang et al., 2008).

Throughout the years, generative grammar has evolved into different approaches which can largely be divided into transformational and non-transformational generative approaches. Transformational approaches introduce the concept of transformations to go from the deep structure of a sentence (i.e. the phrase structure) to the surface structure which corresponds to the actual sentence.

Theories such as government and binding theory (Chomsky, 1993a) and minimalism theory (Chomsky, 1993b) belong to the transformational approaches. Again, the focus is on the syntactic parse of a sentence from which, if needed, semantic relations can be derived.

The non-transformational approaches such as Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag, 1994) and Sign-based Construction Grammar (SBCG; Boas and Sag, 2012) introduce a constraint-based formalism to analyse sentences. They use attribute-value matrices to describe the rules and lexical elements in the grammar and rely in unification to build up phrase structures. This approach is heavily lexicalised since it starts from the lexical elements in the grammar on which abstract rules can apply. Meaning is thus built up compositionally from the meaning of the lexical items. HPSG has been used as a semantic parsing engine to map natural languages onto their meaning representation (see e.g. McFetridge et al., 1996; Frank et al., 2007). The generative approach of Tree-Adjoining Grammar (TAG; Joshi and Schabes, 1997) starts from trees as the basic element in the syntactic structure. These trees can be combined through operations such as adjunction or substitution. Since the minimal element is a tree structure, TAG grammars are tree generation systems. Extensions of TAG have been used for semantic parsing (Lichte and Kallmeyer, 2017; Arps and Petitjean, 2018; Bladier et al., 2023). Further, Lexical-Functional Grammar (Bresnan, 1978, 1985) can be seen as a generative approach that analyses utterances on distinct levels. Mainly, a distinction is made between the level that captures the syntactic constituent structure of a sentence and a level that captures the grammatical functions.

Another influential approach based on phrase structure grammars is Combinatory Categorial Grammar (CCG; Steedman, 1987). This approach starts from the assignment of categories, which are either atomic or single-argument functions, to lexical elements. Through a set of minimal operators, such as application, composition and type-raising, sentences can be parsed. Simultaneously, a meaning representation in terms of lambda calculus can be built. In this way, Combinatory Categorial Grammar can be used as a semantic parsing engine that maps natural language utterances into logical forms, e.g. Zettlemoyer and Collins (2005, 2007); Artzi and Zettlemoyer (2011); Kwiatkowski et al. (2010); Krishnamurthy and Mitchell (2012); Berant et al. (2013); Cai and Yates (2013); Reddy et al. (2014); Pasupat and Liang (2015).

While the approaches discussed above play a prominent role in parsing natural language, they all primarily focus on the compositional aspects of language.

However, there are non-compositional aspects to language, for example in the case of idioms. Indeed, the meaning underlying idioms cannot be derived from the combination of the meaning of the lexical items. Construction grammar, as pioneered by Fillmore (1968, 1988) and Goldberg (1995), is a linguistic approach that starts from this non-compositionality of language. Construction grammar states that all linguistic knowledge is captured in form-meaning mappings called constructions. These constructions can contain information from all levels of linguistic analysis (e.g. phonology, morphology, syntax, pragmatics). Moreover, constructions can range from fully instantiated (e.g. a lexical item), partially instantiated (e.g. the X-LET-ALONE-Y-CXN analysed by Fillmore et al., 1988) to fully abstract constructions (e.g. an intransitive construction). Construction grammar thus takes as basic element the construction which maps between form and meaning. This analysis makes it straightforward to analyse idioms as constructions that map between the form of the idiom and its non-compositional meaning. For example, the PIECE-OF-CAKE-CXN is a mapping between the form *"a piece of cake"* and the meaning that something is easy. In this example, it is clear that the meaning cannot be derived from the lexical elements, which would result in a slice of some delicious, sweet dessert. Thus, construction grammar gives semantics a central role in linguistic analysis since it views meaning as an integral part of a construction. In contrast to most generative approaches, meaning does not need to be derived from the syntactic analysis, which makes construction grammar ideal for retrieving meaning representations underlying linguistic utterances. Fluid Construction Grammar (Steels, 2004; van Trijp et al., 2022; Beuls and Van Eecke, 2023), a computational implementation of construction grammar, has been used as a semantic parsing system to map between linguistic utterances and their meaning representation (see e.g. Nevens et al., 2019a; Doumen et al., 2023, respectively for the design and the learning of such grammars). For a more detailed discussion on construction grammar, see Section 2.4.

Besides grammar-based approaches, other techniques have been used to map natural language onto meaning representations. Indeed, the task of semantic parsing has also been tackled in a more probabilistic approach. Specifically neural approaches are commonly used to parse language to a meaning representation. For example, RNNs have been used to map questions onto meaning representations (Andreas et al., 2016a; Dong and Lapata, 2016; Zhong et al., 2017; Cheng et al., 2019). Recently, large language models have been used as a semantic parsing technique (see e.g. Drozdov et al., 2022; Shin and Van Durme, 2021; Gao et al., 2023; Chen et al., 2022; Cheng et al., 2023; Subramanian et al.,

2023).

Grammar-based approaches generally have the advantage of being accurate and interpretable. Neural approaches can be more broadly used and can achieve impressive results, but require more training data. In general, using a parser that maps language to a meaning representation has the advantage of making the system more interpretable since the grammar that is used can be observed and interpreted. In terms of the grammar-based approaches, construction grammar is a good fit, since it uses the notion of a construction (i.e. a mapping between form and meaning containing information from all levels of linguistic analysis) as its most fundamental unit. Therefore, construction grammar is useful for handling the non-compositionality of language. For these reasons, I will adopt this approach and specifically its computational implementation in terms of Fluid Construction Grammar as a parsing engine throughout the thesis.

In this thesis, I will use computational construction grammar as a way of parsing from linguistic utterances to their meaning representations. In Section 2.4, I will discuss construction grammar in more detail. In Chapter 3, I will describe the technical foundations of Fluid Construction Grammar, which is the computational construction grammar implementation that I will use in the remainder of the thesis.

2.2.3 Discussion

Human-like language understanding is thus a process in which the appropriate knowledge sources are consulted in order to build up a rich model of the language at hand. This is a process that comes natural to humans since humans are always trying to make sense of the situation. While current data-driven, statistical natural language understanding systems are often believed to understand language, there are some aspects of human language understanding that are missing mostly due to how these systems are implemented. First of all, these models are not able to capture meaning in a way that is similar to humans. The models start from modelling meaning by looking at occurrences and thereby only taking into account the form side of language (Bender and Koller, 2020; Lake and Murphy, 2023). Moreover, the systems are not truly grounded in the situational or discourse context (Bender and Koller, 2020). It is often believed that these system will eventually achieve to handle context and meaning in a human-like way by providing these models with more data, however, it is argued by, amongst others, Bender and Koller (2020) that it will not be possible to capture the situational context and grounding the systems just by providing

more textual data. Winograd (2001, p. 403) already addressed this by stating that "Context is not just more text."

Furthermore, the evaluation of these types of techniques considers for the most part only the result of the model (i.e. the generated language). This can be mainly attributed to the focus on benchmarking in NLP (Raji et al., 2021). Many benchmarks have been introduced, and the personal nature of the language understanding process is hard to cast in an annotation scheme. Therefore, the benchmarks often focus on the result and not the process itself. Furthermore, the focus on benchmarking brings the problem of drifting away from the research on building systems that can truly understanding to building systems that perform well on a specific task (Bowman and Dahl, 2021). Focusing on the results also takes away from the aspect that language understanding is an individual experience to a human. Language is grounded in the personal knowledge of a human, which can even change over time as new experiences and knowledge come in. Thus, the evaluation of language understanding models should take this into account.

To operationalise truly human-like language understanding, it becomes necessary to accurately model the semantics underlying linguistic observations through meaning representations. I discussed commonly used meaning representations, including procedural semantics. In Section 2.3, I will discuss this meaning representation in more detail. Next, accurately modelling context as well as knowledge is needed since both these sources of knowledge need to be consulted during the understanding process. Moreover, due to the central position of meaning representations, it is necessary to explore parsing techniques that can parse from the language to the meaning representations. In this context, I discussed grammar-based and neural parsing techniques. In Section 2.4, I will discuss computational construction grammar as a framework for mapping between linguistic observations and their meaning representations.

2.3 Procedural semantics

In this section, I discuss procedural semantics as a meaning representation that is adequate for modelling the semantics underlying linguistic utterances in truly intelligent systems.

Procedural semantics was introduced by Woods (1968), Woods et al. (1972) and Winograd (1972). Both authors introduced procedural semantics as part of language processing systems, namely SHRDLU and LUNAR. In these question-

answering systems, it was needed to parse natural language utterances to a logical representation that could be understood by a computer. This adheres to the need for a representation as explained by Winograd (1972, p. 28): "In practical terms, we need a transducer that can work with a syntactic analyzer, and produce data which is acceptable to a logical deductive system." Specifically, by parsing a question into an executable meaning representation, an answer can then be computed in a straightforward manner. The application in question-answering systems immediately shows the benefit of the approach, namely that the meaning can be executed and an answer can be computed. Thus, procedural semantics consists in programs that can be executed computationally. More strictly speaking, the semantics consists in the procedures that need to be executed. These procedures are called primitives or primitive operations, and represent the smallest procedure that can be executed. These primitives are then combined into a meaning network. Later, Johnson-Laird (1977) described the term procedural semantics through a compile-and-execute metaphor. Specifically, he describes that language can be compiled into a program that can then be executed, in a similar way as computer programs are first compiled and then executed. The difficulty is the complicated nature of natural languages as opposed to programming languages and adequate parsing systems are thus required (Johnson-Laird, 1977).

In the remainder of this thesis, I will follow the compile-and-execute metaphor of Johnson-Laird (1977) and use the definition of procedural semantics as meaning representations that can be executed algorithmically.

2.3.1 Systems operationalised through procedural semantics

SHRDLU introduced by Winograd (1972) is a system for language understanding that answers questions and commands in a dialogue with a human user. The system is situated in a virtual blocks world and has a 'robotic arm' that can execute these commands. Crucially, SHRDLU uses a procedural semantic representation to easily integrate language with knowledge about the world and language. Concretely, a grammar parses the utterances to a semantic representation, which is given to a planner system that processes the representation and deduces facts and knowledge. Moreover, SHRDLU is a dialogue system and is thus capable of handling non-trivial co-references. Once a pronoun is encountered, the system uses a heuristic to find the referent either in the same sentence or in the previous sentences, assigning a value to the possible referent(s).

The three first sentences from the example dialogue introduced in Winograd

(1971, p. 35) are shown below:

> "pick up a red block."
> "OK."
> "grasp the pyramid."
> "I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN."
> "find a block which is taller than the one you are holding and put it into the box."
> "BY "IT", I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING."

Even though the system is quite brittle (the questions and utterances need to be well-chosen in order for the system to be able to handle them), it revolutionised the field of AI by introducing the concept of using procedures as an underlying meaning. Due to the procedural semantics, the integration of knowledge and language becomes more feasible. That is to say, the procedures that the parser provides can be executed immediately since the knowledge of the system also consists in procedures.

Woods (1968) introduced a system that is able to answer questions concerning information stored in a database of airline information. The parser of the system translated the language into a meaning representation consisting of primitives. The primitives are the functions, predicates and commands with which all information in the database can be retrieved. The question-answering system can answer questions such as "*Does American have a flight which goes from Boston to Chicago?*" and "*What American Airlines flights arrive in Chicago from Boston before 1:00 p.m.?*". This set of primitives is easily expendable by implementing new operations. Furthermore, the LUNAR system introduced by Woods et al. (1972) is a question-answering system that can answer questions about chemical data concerning moon rocks and soil composition of the moon. This system was developed to provide scientists with an easy interface to the database through natural language. An example sentence is "*Give me all analyses of S10046*". This system also uses a parser to translate the natural language utterances to a meaning representation that is able to query the database.

From these foundational systems such as SHRDLU and LUNAR, different approaches to meaning representations that can be executed algorithmically have been explored. In current research, four classifications of procedural semantics systems can be distinguished. A first type of meaning representations consists in representations that contain the steps or primitives that need to be executed, following the approach introduced by Winograd (1972). First, the neural module

network approach introduced by Andreas et al. (2016b) and further developed by Johnson et al. (2017b) and Kottur et al. (2018) captures the meaning in a program from which a neural model predicts an answer. These networks consist of primitive operations representing the steps that need to be executed to come to an answer. Then, the whole network is given to a neural network that is trained over these networks to predict answers. Furthermore, neuro-symbolic approaches to visual question answering (Yi et al., 2018; Mao et al., 2019), visual dialogue (Abdessaied et al., 2022) and video question answering (Yi et al., 2020) make use of programs consisting in primitive operations. In contrast to the neural module network approach, the primitives in these networks are implemented in a symbolic way. The primitives operate over a symbolic representation of the world that is retrieved in a neural way. Similarly, the purely symbolic approach introduced by Nevens et al. (2019a) introduces a set of symbolic primitives that operate over the symbolic representation of images. Other approaches have explored the use of semantics that can query databases, in a similar way as Woods (1968); Woods et al. (1972). These approaches include parsing natural language to SQL (Zhong et al., 2017), FunQL (Cheng et al., 2019) or SPARQL (Yahya et al., 2012). Each of these are languages that can operate over a database and it starts from the hypotheses that the meaning of questions coincide with the query to retrieve the information from the database. A third type of meaning representation are based on logic representation which can then be executed using a deductive system, often in the form of lambda calculus (Zettlemoyer and Collins, 2005; Kwiatkowski et al., 2010; Krishnamurthy and Mitchell, 2012; Berant et al., 2013; Cai and Yates, 2013; Reddy et al., 2014; Pasupat and Liang, 2015). However, it is often necessary to first transform these meaning representations into forms over which the system can reason. Thereby, they lack the advantage of the direct executability that the other representations possess. There are, however, systems that use a logic representation that can immediately be given to a deductive system (see e.g. Krishnamurthy and Kollar, 2013). A last kind of meaning representations consists in representing the utterance using code, which can then be executed directly. Gao et al. (2023) introduces Program-Aided Language models (PAL) and Chen et al. (2022) Program-of-Thoughts (PoT) prompting paradigm, both of these approaches use a large language model to generate code that is then given to an interpreter. Following these paradigms, Subramanian et al. (2023) introduces a system that uses a large language model to generate Python code to solve visual question answering questions and the model introduced by Cheng et al. (2023) generates SQL or Python code to access databases based on language models. Gupta and Kembhavi (2023) shows that the approach of prompting a large language model to generate a program can

be applied on multiple tasks. Moreover, robotic systems use a similar approach of using language models to generate code (Liang et al., 2023). These systems rely on large pre-trained language models to generate code on the basis of a few example prompts.

2.3.2 Operationalising procedural semantics

The operationalisation of procedural semantics can be done in a straightforward manner by implementing the primitive operations that a meaning representation can contain. This implementation can be done in several ways. The primitive operations can be queries to a database. In this case, they can be implemented through neural networks or purely symbolically as operations over a representation of the world. Another way of operationalising these procedural semantics is by giving them to a logical deductive system. Many of these systems thus use their own implementation framework. There exists, however, an open-source framework for operationalising procedural semantics: Incremental Recruitment Language (IRL; Van den Broeck, 2008; Spranger et al., 2012). IRL consists in a framework for representing, interpreting and composing procedural semantic networks. The primitive operations are represented through predicates with arguments. Interpreting the network then consists in executing each of the primitives and finding bindings for the variables. There is no prerequisite of primitives that need to be used and the implementation is also completely free. The primitive operations can, for example, be implemented through a call to a neural network or through logic operations such as counting or filtering a set. Moreover, the system is completely open-ended. This framework is used to interpret the semantic representation in a variety of experiments, mostly on the evolution of language (see e.g. Bleys (2016) for language games on the emergence of colour lexicons, Spranger (2016) for experiments on the evolution and emergence on spatial constructions, Pauw and Hilferty (2012) for the emergence of quantifiers). In these experiments, IRL is used to compose meaning networks and to interpret them. In Nevens et al. (2019a) IRL is used as a framework for interpreting the procedural semantic representations underlying the natural language questions in a visual question answering task.

2.3.3 Discussion

Procedural semantics is an ideal type of meaning representations to use in language understanding systems. Following the compile-and-execute metaphor, this type of semantics captures the meaning of a linguistic utterance in a repre-

sentation that is directly executable. Moreover, some systems, such as SHRDLU (Winograd, 1972) or the neural module networks (Andreas et al., 2016b), describe the procedures that are needed to interpret the sentence, thereby showing the truly procedural aspect of procedural semantics. These minimal operations that can be executed are called primitive operations or primitives.

In this chapter, I discussed some systems using procedural semantics. By executing the meaning representation, it becomes possible to ground the language in the situational and discourse context. For example, the SHRDLU system consists in primitive operations that can access the representation of the world as well as primitives that can access the representation of the discourse. Moreover, I discussed Incremental Recruitment Language (IRL) as a framework for operationalising procedural semantics. It consists in a toolkit for representing, interpreting and composing procedural semantics. In Chapter 3, I will discuss IRL in more detail and in Chapter 4 I use IRL as a way of representing and executing the neuro-symbolic procedural semantics.

2.4 Construction grammar

Construction grammar originally started as an alternative approach for the more generative approaches to language, as introduced by Chomsky (Chomsky, 1956) (see Section 2.2.2). Traditionally, in generative approaches to language, language is modelled through a lexicon and a set of grammar rules. The lexicon and the grammar combined are able to cover the 'core' of the language. Utterances that cannot be covered, for example, idiomatic expressions, are viewed as the 'periphery'. Construction grammar as introduced by Fillmore (1988, 1968); Goldberg (1995) on the other hand starts from the principle that these idiomatic expressions are interesting phenomena worth investigating, starting with the 'let alone' construction analysed by Fillmore et al. (1988). Nowadays, construction grammar contains many different 'flavours', such that it is now often referred to as constructionist approaches (Goldberg, 2003). Even though all these constructionist approaches have their own research foci and frameworks, they all agree on the principle that constructions form the basis of linguistic analysis. These constructions are form-meaning mappings and constitute a relation between some sort of form and some sort of semantic information. Moreover, constructionist approaches aim to model all linguistic usage, removing the distinction between the core and periphery of language.

Many theories on Construction Grammar (CxG) exist, each with a different

'flavour'. Following van Trijp (2024); Ungerer and Hartmann (2023), I briefly introduce the main constructionist approaches: Berkeley Construction Grammar, Cognitive Construction Grammar, Sign-based Construction Grammar and Fluid Construction Grammar. For an in-depth discussion on the constructionist approaches to language, see amongst others van Trijp (2024) and Ungerer and Hartmann (2023).

Berkeley Construction Grammar (Fillmore et al., 1988; Kay and Fillmore, 1999) focusses on the grammatical description of linguistic phenomena. This approach forms the foundation of current constructionist approaches. Cognitive Construction Grammar (Goldberg, 1995, 2006, 2019) puts psychological plausibility central. It is a usage-based approach and also focusses on language learning. Cognitive Construction Grammar want to formalise as less as possible and use informal notations to describe constructions. Methodologically, they focus on corpus-based methods as well as psychological experiments. Radical Construction grammar (Croft, 2001) is a more typological-oriented approach, focusing on describing linguistic knowledge in the form of language-specific and construction-specific constructions. Moreover, these constructions are related to each other. As Cognitive Construction Grammar, this approach make use of less formal descriptions. Sign-based Construction Grammar (Sag, 2012) finds its roots in Head-driven Phrase Structure Grammar (Pollard and Sag, 1994). The approach is highly formalised and focusses on detailed descriptions of constructions, describing them in the form of attribute-value matrices. Due to its high precision and formalisation, it can be used to investigate the preciseness of constructions. It can, however, be argued that this approach focusses more on form than on meaning. Although Fluid Construction Grammar is sometimes categorised as a flavour of construction grammar, it is not a linguistic theory in itself, but rather a technical operationalisation of the basic tenets underlying construction grammar. Using this framework, it is possible to operationalise construction grammars. This way, linguistic theories (i) can be tested on internal consistency and precision, (ii) they can be validated on corpus data, and (iii) these insights can be used to develop language technologies (van Trijp et al., 2022).

Beuls and Van Eecke (2024) provide an in-depth discussion on the similarities between AI and construction grammar. Due to the development of both frameworks during the same period in time, as well as the influence that they had on each other, construction grammar is an ideal framework to operationalise human-like language understanding. Concretely, construction grammar starts from many of the principles that are required in human-like language under-

standing systems. These similarities are discussed by Beuls and Van Eecke (2024) and can be summarised in the light of the requirements of human-like language understanding systems as follows: Construction grammar sees communication as the basic function of language. Communication is the process in which information is transferred from a speaker to a listener. This listener can, in their turn, communicate information back to the original speaker. Language is thus a bi-directional process and it entails both comprehension (i.e. the process of going from form to meaning) as well as production (i.e. the process of going from meaning to form). The bi-directionality is modelled in construction grammar through the use of the same construction inventory in both processes. Construction grammar starts from the principle that meaning and form are inherently connected, which becomes clear through the basic tenet that all linguistic knowledge should be captured through form-meaning mappings called constructions. Certainly, truly intelligent systems require to handle meaning since building up meaning is inherently connected to the process of language understanding. Moreover, construction grammar adheres to the principle that language is grounded in the world. Language serves a communicative purpose, and since language is uttered in the world, it is necessary to ground it in this same world. In order to do so, knowledge about the world, both in terms of factual knowledge as well as common-sense knowledge is required. Lastly, language is acquired, not innate and language can change over time. Therefore, it is necessary to keep in mind that language understanding systems need to be able to be adaptive over time.

2.4.1 Basic tenets

Even though many flavours to construction grammar exist, constructionist approaches to language adhere, for the most part, to a few principles that are laid out by amongst others (Fillmore, 1988; Goldberg, 1995; Kay and Fillmore, 1999; Croft, 2001). van Trijp et al. (2022) and Beuls and Van Eecke (2023) summarise the basic tenets of construction grammar as follows:

- Constructions are form-meaning mappings. All linguistic knowledge of a language user is captured in these constructions. These constructions constitute the basic elements of both production and comprehension.

- All constructions are situated on a lexicon-grammar continuum, meaning that there exists no distinction between the lexicon or grammar rules as is often the case in traditional approaches to grammar. Constructions can of course contain more lexical information, such as a the BREAK-A-LEG-CXN or

can be more grammatical such as the RESULTATIVE-CXN. Constructions can also be partially instantiated, for example the X-TAKE-Y-FOR-GRANTED-CXN. All constructions are thus situated somewhere on this continuum.

- Constructions can capture information from all levels of linguistic analysis. They can contain information from phonetics, phonology, morphology, syntax, semantics and pragmatics. Constructions can also be multi-modal, for example in the case of gesture constructions. Typically, the form side of a construction includes information from phonetics, phonology, morpho-syntax, while the meaning side contains semantic or pragmatic information.

- Language is a dynamic system, there is no fixed set of constructions. Constructions can change over time and become more or less entrenched. Constructions are acquired and represent the linguistic knowledge of an individual language user.

2.4.2 Computational construction grammar

Computational construction grammar aims to operationalise the basic tenets of construction grammar. There are a few approaches that aim to computationally model construction grammar, amongst which Embodied Construction Grammar and Fluid Construction Grammar are probably the most well-known. Embodied Construction Grammar (ECG; Bergen and Chang, 2005; Feldman et al., 2009) is a computational operationalisation of constructional parsing, focusing on embodiment and the neural foundations of linguistic analysis. It concentrates on comprehension, by providing a processing engine for parsing through constructions. It is, however, not able to model language production.

Fluid Construction Grammar (FCG; Beuls and Van Eecke, 2023; van Trijp et al., 2022; Steels, 2011, 2017) is a framework for representing, processing and learning construction grammars. It starts from the basic tenets described above, thereby providing an ideal framework to operationalise any of the constructionist approaches (van Trijp et al., 2022). It is not a linguistic theory in itself, but can be used to verify and test the linguistic analyses. It operationalises the constructional approaches by providing a representation for constructions, consisting in feature-value pairs. The features that are used are completely free and can contain features from all levels of linguistic analyses. Moreover, the same set of constructions can be used in both comprehension and formulation, making language a truly bi-directional process. FCG also provides support for learning grammars through the mechanisms of intention reading and pattern

finding. Throughout the years, FCG has been used in many facets, for example in grammar learning experiments (van Trijp and Steels, 2012; Beuls and Steels, 2013; Van Eecke, 2018; Nevens et al., 2022; Doumen et al., 2023), as a grammar that models linguistic analyses (see e.g. van Trijp (2017) for a grammar on English, Van Eecke (2017) for a grammar covering the Dutch verb phrase, Beuls (2017) for a model of Spanish verbs, van Trijp (2011) for argument structure constructions, Beuls (2011) for Hungarian verbal agreement, Marques and Beuls (2016) for evaluation strategies of the designed grammars), as well as in language technology applications (Willaert et al., 2020, 2021, 2022; Beuls et al., 2021; Nevens et al., 2019a; Verheyen et al., 2022b, 2023). Due to the possibility of both comprehension and production processes and due to the many applications and analyses that were possible through FCG, it can be argued that FCG is the most advanced computational construction grammar approach that nowadays exists.

2.4.3 Discussion

Construction grammar and in particular computational construction grammar is a useful framework to implement language processing in systems for human-like language understanding since it relies on many of the same principles that are required in truly intelligent systems. This can be traced back to the similar attitude towards language that both fields of Artificial Intelligence and construction grammar exhibit as well as to the continued influence that these two fields have had on each other (Beuls and Van Eecke, 2024). Construction grammar starts from the principle that all linguistic knowledge is captured in form-meaning mappings, thereby taking into account meaning representations which are required in truly intelligent systems. Moreover, they start from the principle that language is grounded in the world, which makes it necessary to model the knowledge of the world so that the systems can interpret it.

Computational construction grammar aims to operationalise the basic tenets underlying construction grammar. Specifically, Fluid Construction Grammar provides a framework for representing, processing and learning computational construction grammars, thereby provides an ideal basis for building human-like language understanding systems. In Chapter 3, I will discuss Fluid Construction Grammar in more detail.

2.5 Conclusion

In this chapter, I have reviewed the literature on language understanding, procedural semantics and construction grammar, thereby laying the conceptual foundations for the remainder of the thesis.

Language understanding can be seen as a process in which knowledge is used to make sense of the situation at hand. During this process, a rich model is built up. To operationalise language understanding in intelligent systems, it is necessary to represent the meaning underlying linguistic utterances, as well as to represent the context (i.e. both the situational context to ground the language in the environment as the discourse context to ground the language in the conversation). Moreover, adequate parsing techniques are required to map the utterances to their respective meaning representations.

In Section 2.3, I introduced procedural semantics as a meaning representation that can be executed computationally, following the compile-and-execute metaphor introduced by Johnson-Laird (1977). Natural language can be parsed into a meaning representation that is then executed in some way. Procedural semantics was introduced by pioneers such as Winograd (1972); Woods (1968); Woods et al. (1972). Later development led to systems that use query languages such as SQL or SPARQL or programming languages such as Python as the basis for the underlying meaning representations. Other approaches include neural module networks, neuro-symbolic approaches or purely symbolic approaches. I briefly discussed Incremental Recruitment Language as a framework for operationalising procedural semantics. Throughout this thesis, meaning representations that can be executed algorithmically have a central position. In Chapter 4, I introduce a neuro-symbolic approach that uses procedural semantics operationalised through the IRL framework, I will introduce the IRL framework in Chapter 3. Chapter 5 introduces a novel frame-based procedural semantics representation that can be executed by a Prolog reasoning engine.

I have outlined the basic principles of construction grammar and discussed how they can be operationalised computationally. Furthermore, I discussed why construction grammar and, more specifically, computational construction grammar is a suitable framework to use in systems that understand language in a human-like way. Moreover, I have also briefly introduced FCG as a framework for computational construction grammar. In the next chapter, I will elaborate on the technical aspects of FCG since it will be used as the language processing backbone of the human-like language understanding systems that I introduce in this thesis. Concretely, in Chapter 4 and 5, I introduce hand-written grammars

to map between linguistic utterances and their meaning representation.

Chapter 3

Technical Foundations

3.1 Introduction

This chapter outlines the technical foundations that underlie this thesis. Specifically, I will discuss two components of the Babel toolkit (Loetzsch et al., 2008; Steels and Loetzsch, 2010; Nevens et al., 2019b) that are used in the systems introduced in the next chapters: Incremental Recruitment Language and Fluid Construction Grammar. First, I will describe Incremental Recruitment Language

(IRL; Van den Broeck, 2008; Spranger et al., 2012), a framework for representing, interpreting and composing procedural semantic representations. In Chapter 4 and 6, the IRL framework is used to execute procedural semantic representations. Second, I will discuss the framework of Fluid Construction Grammar (FCG; Steels, 2011, 2017; van Trijp et al., 2022; Beuls and Van Eecke, 2023), a language processing engine for representing, processing and learning construction grammars by operationalising the ideas of construction grammar. In Chapter 4, 5 and 6, I describe how I used FCG as a language processing engine to map utterances to their respective meaning representations. In this chapter, I will give a brief introduction to IRL and FCG and I will focus on the parts that are necessary to understand the remainder of the thesis. The discussions of IRL and FCG are based on Nevens (2022) and Van Eecke (2018) respectively and I refer the reader to those sources for a more detailed discussion on the two systems.

Both the IRL and FCG frameworks are part of the Babel toolkit (Loetzsch et al., 2008; Steels and Loetzsch, 2010; Nevens et al., 2019b). Babel is designed to operationalise language game experiments, in which agents develop and learn their own communicative systems from the ground up (see e.g. Beuls and Steels, 2013; Beuls and Höfer, 2011). It consists of an experiment framework, which provides a framework to set up language game experiments in which FCG can be used as a language processing engine and IRL as a framework to execute semantic networks. Furthermore, Babel consists of a robot interface to conduct experiments in the real world using physical embodiments (Nevens et al., 2019b). Babel can be found at `https://emergent-languages.org/`. Babel is currently in active development and as well as using Babel as a toolkit to operationalise the experiments, I made several contributions, which I highlighted in the respective chapters. All experiments conducted in this thesis are integrated in the Babel framework.

Recently, the FCG editor, a user-friendly standalone environment for FCG, was released (van Trijp et al., 2022). Originally, the FCG editor was designed to be an environment for FCG, but the FCG editor now includes both FCG and IRL. This way, users have one freely accessible environment for processing language using FCG and executing procedural semantics with IRL. The FCG editor is available at: `https://www.fcg-net.org/download/`.

3.2 Incremental Recruitment Language

Incremental Recruitment Language (IRL) is a framework for representing, executing and composing procedural semantic networks (Van den Broeck, 2008; Spranger et al., 2012).

IRL operationalises the ideas underlying procedural semantics (Woods et al., 1972; Winograd, 1972), which states that the meanings of utterances are represented as programs that are directly executable. IRL was designed in the context of the language game paradigm, which investigates how communicative systems can emerge in populations of agents (Steels, 1995, 2012; Nevens et al., 2019b). In these language game experiments, IRL serves as the bridge between the language and the agent's sensorimotor experience, by representing the meaning of the utterances in procedural semantic networks and by executing these meaning networks in the environment of the agent. Furthermore, IRL supports the composition of semantic networks, which occurs in a language game when a speaker is conceptualising what it wants to express or during the intention reading process during learning of construction grammars. In what follows, I will focus on the representation and the execution processes, I refer the reader interested in IRL's composition process to Nevens (2022).

IRL represent the meaning as a semantic network composed of primitive operations. The primitive operations represent the basic cognitive capacities of an agent (e.g. segmenting a scene into objects, filtering an object on a colour), but it is completely up to the user to choose the primitive operations. These primitives are represented as predicates consisting of a primitive name and a list of arguments that are variables. The predicates in the network are linked through the variables.

The execution of an IRL program, (i.e. the primitive application process), is a problem-solving process that looks for bindings for each of the variables in the network. Executing an IRL program thus means executing each of the primitives in the network so that a binding for each of the variables is found. This set of bindings needs to be consistent and complete. The implementation of the IRL primitives is completely up to the user. The primitives can, for example, be operationalised symbolically, or subsymbolically, by calling neural networks or other web services.

Figure 3.1 shows an example of a procedural semantics network, which will be used as an example throughout this chapter. The network in this figure could be the underlying meaning of sentences as "*How many cubes are there in image*

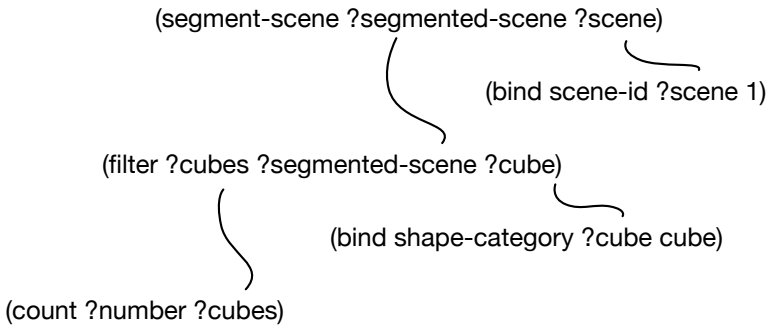


Figure 3.1: A meaning representation in terms of procedural semantics which consists of three primitive operations (i.e. segment-scene, filter and count) and two bind statements.

1?" or "*What number of cubes are present in image 1?*". To answer these types of questions, the following steps need to be executed: first, the scene needs to be segmented into the objects, then, these objects need to be filtered based on the shape cube, resulting in a set of cubes, then the elements in this set need to be counted. The result of this counting operation is the answer to the question. These steps are represented in the meaning network as the primitives (SEGMENT-SCENE ?SEGMENTED-SCENE ?SCENE), (FILTER ?CUBES ?SEGMENTED-SCENE ?CUBE) and (COUNT ?NUMBER ?CUBES). It also has two bind statements: (BIND SCENE-ID ?SCENE 1) and (BIND SHAPE-CATEGORY ?CUBE CUBE). In the first bind statement the scene-index 1 is bound to the ?SCENE variable and the second bind statement binds the concept of CUBE to the ?CUBE variable.

3.2.1 Bind operators and semantic entities

The bind operators in IRL are special operators that add semantic entities to an IRL network by binding semantic entities to variables. Semantic entities represent (i) the concepts or (ii) the representations of the environment of the agent. The concepts include categories, prototypes, relations that an agent stores in its conceptual inventory. These concepts are grounded in the environment by the agent's sensorimotor experiences. IRL does not impose a specific grounding process. For example, it is possible to learn the concepts through a discriminative-based approach (Nevens et al., 2020) and use these learned concepts as bindings. (BIND COLOUR-CATEGORY ?COLOUR BLUE) is an example of a binding where a concept is bound, namely the concept of BLUE is bound to the variable ?COLOUR. Next to concepts, the bind operators can also be used to bind representations (e.g., a world model) or partial representations of the

environment (e.g. a world model manipulated by the primitive operations) to a variable. For example, in the beginning of the execution of the network shown in Figure 3.1 a world model of the scene is bound to the variable ?SEGMENTED-SCENE. Later, the world model is filtered on the concept of cubes and this entity is bound to the ?CUBES variable. The semantic entities are typed.

Bindings in an IRL network are added through executing the network or by *bind statements*. A *bind statement* is a binding that is explicitly stated in the semantic network. In Figure 3.1 two bind statements can be found: (BIND SCENE-ID ?SCENE 1) and (BIND SHAPE-CATEGORY ?CUBE CUBE)

3.2.2 Primitives

The primitive operations represent the basic cognitive capacities of an agent. This set of operators can be seen as the *instruction set* of the brain, namely all operations that can be executed by this agent. For example, the basic operation of counting can be represented by a COUNT primitive. It is completely up to the user to choose and design the primitive operators, since IRL does not require any certain primitive operation to be present in the program. The implementation is also completely up to the user. Depending on what is needed, a primitive can have a symbolic or subsymbolic implementation. For example, in chapter 4, I introduce a neuro-symbolic procedural semantics for the task of visual dialogue where a combination of symbolic and subsymbolic primitives is designed.

A primitive consists of a primitive name followed by one or more arguments. Each of these arguments is typed and is represented by a variable that is either bound or unbound. For example, in (FILTER ?CUBES ?SEGMENTED-SCENE ?CUBE), the primitive name is FILTER, and the arguments are the variables ?CUBES, ?SEGMENTED-SCENE and ?CUBES. Depending on which of the arguments are bound or unbound, the primitive will be executed differently. These different executions are the different *cases* of a primitive, which make the evaluation of the IRL program multidirectional. For example, if the ?CUBE and ?SEGMENTED-SCENE variables in the FILTER primitive are bound, then the execution of this primitive will lead to a binding for ?CUBES containing the set of cubes from the segmented scene. If all arguments of a primitive are bound, then the primitive typically performs a consistency check to make sure that all bindings are consistent with each other.

Figure 3.2 shows the multidirectionality of the FILTER-BY-COLOR primitive. Four different cases are shown. Figure 3.2a shows the case that is mostly used during execution of the primitive in a semantic network during interpretation. The

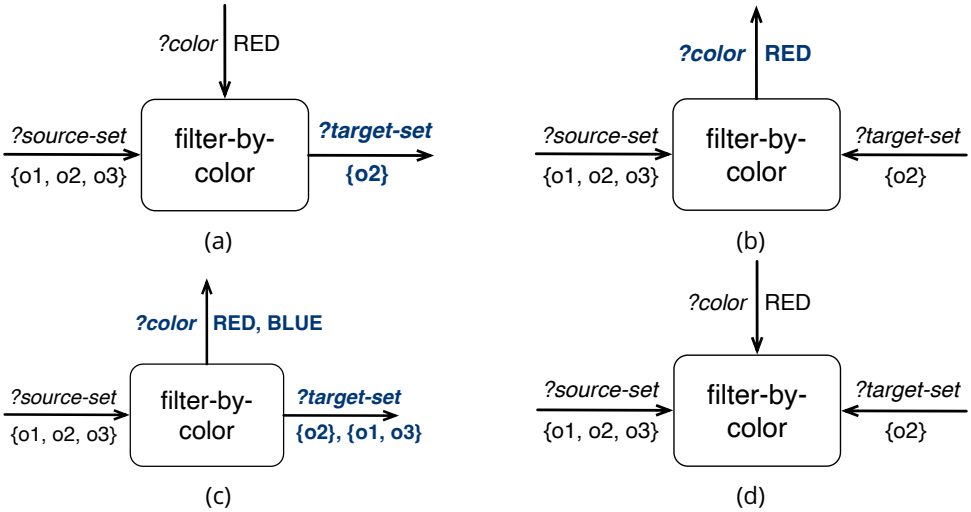


Figure 3.2: An example of the multidirectionality of the primitive operation (FILTER-BY-COLOR ?SOURCE-SET ?TARGET-SET ?COLOR). The direction of the arrows indicate whether the arguments are bound (incoming arrows) at the beginning of execution or whether the execution binds values to arguments (outgoing arrows). Figure adapted from Nevens (2022)

given source set with objects is filtered on a certain color leading to a target set containing objects with that color. In this case, the source set and the color on which to filter on are bound (i.e. o1, o2, o3 is bound to ?SOURCE-SET and RED is bound to ?COLOR) and the execution of the primitive leads to a binding for the target set (i.e. o2 binds to ?TARGET-SET). The second case starts from bindings for ?SOURCE-SET and ?TARGET-SET and binds the color RED to ?COLOR, meaning that the set of objects bound to ?TARGET-SET are of the color red. This case is typically used during learning in language game experiments. The third case is relevant during the composition process in experiments, in which the agent gets a set of objects as input to the filter operation and composes different combinations of objects based on the concepts that can be used for filtering. Lastly, Figure 3.2d shows the case of the FILTER primitive in which all the arguments are bound. In this case, the primitive performs a consistency check, making sure that the values of the bindings are consistent with each other.

3.2.3 Primitive application process

The execution in IRL is a problem-solving process, which is characterised by Russell and Norvig (2009) as the process in which operators contribute to the

state representation until all the goal tests are satisfied and a solution is found. The process starts from an initial state representation. In IRL, the initial state consists in the set of bindings at the beginning of processing. Then, the primitive operations that are available in the primitive inventory are checked whether they can apply, and if a primitive can apply, the set of bindings is updated. This process goes on until the set of bindings is consistent and complete and a solution is thus found. The state representation is the set of bindings at a given point in time and the initial state contains the bind statements that were in the network at the beginning of processing. The operators are the primitives, which can apply if the preconditions are satisfied. The preconditions are the bound and unbound variables in the different cases of the primitive. Thus, IRL checks whether the bindings that a primitive requires to be bound (specified in the different *cases* of that primitive) are indeed bound in the set of bindings in the state representation. If a case of a primitive requires certain variables to be bound, that primitive can apply if the state representation holds those bindings. If the preconditions are met, IRL tries to apply that primitive, which can either succeed or fail. If the application succeeds, IRL returns either new bindings or it returns no bindings (in case of a consistency check). These *postconditions* are the new bindings set. The application of a primitive fails if the variables that were bound are inconsistent with each other. If this occurs, IRL backtracks over the other nodes in the search tree. The goal tests are satisfied when all variables in the network are bound (i.e. the network is consistent) and all primitives are executed (i.e. the execution is complete).

Figure 3.3 shows the execution of the semantic network shown in Figure 3.1. The execution starts with an initial node with bindings for the ?SCENE and ?CUBE variable (the two bind statements in the IRL network). The primitive inventory consists of three primitive operations: (FILTER ?TARGET ?SOURCE ?CONCEPT), (COUNT-SET ?NUMBER ?SOURCE) and (SEGMENT-SCENE ?SEGMENTED-SCENE ?SCENE). Starting from the initial node, three nodes are created, and IRL checks which nodes can be executed based on the current bindings and the different cases of the primitives. The FILTER and COUNT-SET primitives cannot be executed (indicated by the blue colour of the node), since every case of the FILTER primitive expects a binding for ?SOURCE, and every case of the COUNT-SET primitive also expects a binding for the ?SOURCE variable. These expectations are modelled by the different cases of the primitive (see Figure 3.2 for the FILTER primitive). It would thus in theory be possible to add a case in which the ?SOURCE is unbound. In practice, however, this is nearly impossible. How would it be feasible to account for all possible sources based on a certain colour or target set?

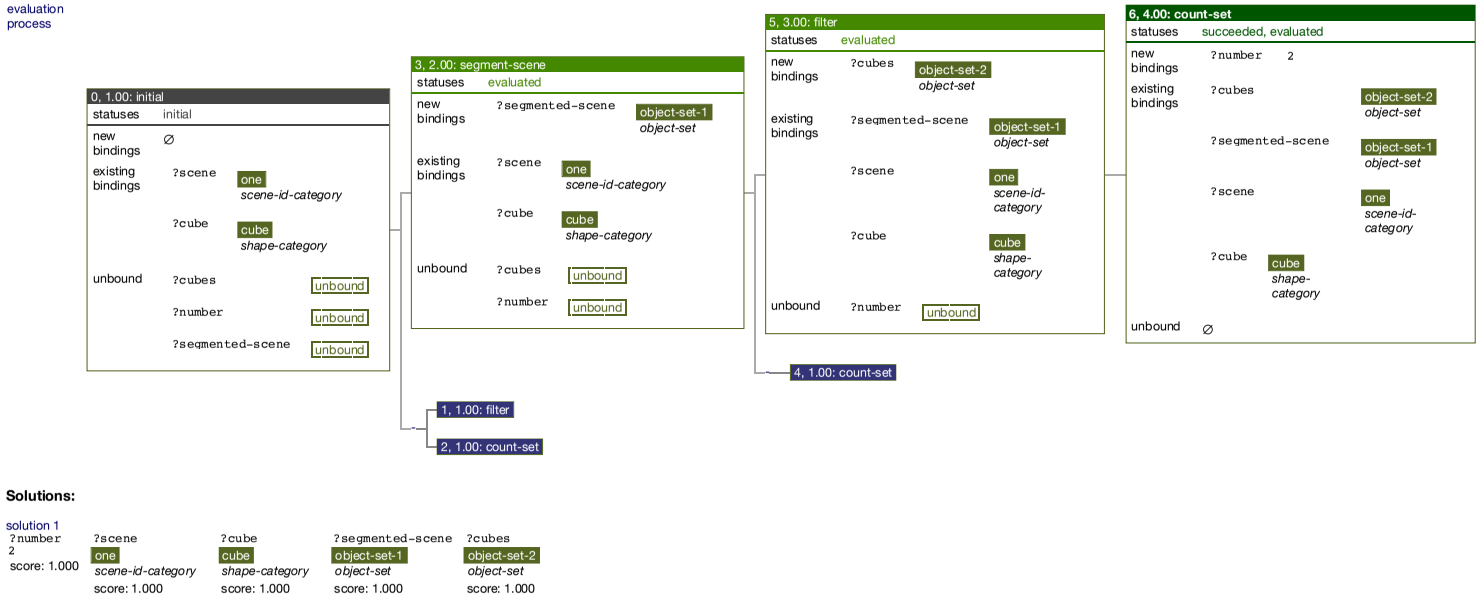


Figure 3.3: The search process of IRL when evaluating the semantic network consisting of the primitive operations SEGMENT-SCENE, FILTER and COUNT. The solution node is indicated in dark green. The nodes on the path leading to this solution are expanded. Every node represents the evaluation of one primitive operation in which new bindings are found. At the bottom of the figure, the solutions in terms of the bindings that were found during execution can be found.

Thus, the FILTER and COUNT-SET primitive cannot apply at the first node. The SEGMENT-SCENE primitive, however, can be applied which results in a binding for the ?SEGMENTED-SCENE variable, namely OBJECT-SET-1. Now, due to this binding, the FILTER primitive can be executed. It binds OBJECT-SET-2 to ?CUBES. Lastly, the COUNT-SET primitive finds a binding 2 for the ?NUMBER variable. In this last step, a solution is found since the set of bindings is consistent and complete. The dark green colour of the node indicates that it is a solution node.

3.2.4 Discussion

In this section, I gave a short introduction on Incremental Recruitment Language (IRL), a language that is designed to represent, execute and compose semantic networks. IRL is an operationalisation of procedural semantics and an IRL program is thus an executable program consisting of primitive operations. Each of the primitive operations can then be implemented in its own way. The execution in IRL is a problem-solving process in which bindings for each of the variables in the primitive operations is found. The choice of and the implementation of the primitive operations is up to the user. In chapter 4, I introduce a set of neuro-symbolic primitives to solve the task of visual dialogue.

Although I introduced IRL as a language to represent, execute and compose semantic networks, I mainly focussed on the representations and execution of IRL networks, since these are the components of IRL that are used in the remainder of this thesis. I refer the reader that is interested in learning more about composing semantic networks to chapter 2.4.5 in Nevens (2022).

3.3 Fluid Construction Grammar

Fluid Construction grammar (FCG) (Steels and De Beule, 2006; Steels, 2011, 2017; van Trijp et al., 2022; Beuls and Van Eecke, 2023) is an operationalisation of the ideas of constructionist approaches to language, which I introduced in Chapter 2. FCG provides a framework to study the emergence, evolution, acquisition and processing of language and it can be used to build intelligent systems that have human-like language systems (Beuls and Van Eecke, 2023). Specifically, Fluid Construction Grammar is a framework for representing, processing and learning construction grammars.

The notion of communication as a bidirectional system is central to FCG, mainly due to its roots in the language game paradigm, which models how communica-

tion systems in agents can emerge. Communication is the process in which a speaker aims to convey a meaning or intention to a listener through linguistic utterances. The listener in their turn tries to reconstruct the meaning from the utterance. Crucially, communication is a bidirectional process, since a person can both be a speaker and a hearer. It is thus important that both these processes can be modelled through one system. FCG facilitates this by providing the opportunity to both comprehend an utterance (i.e. mapping the form to a meaning) and formulate a meaning (i.e. mapping a meaning to a form) using the same set of constructions.

FCG is a approach to language following the linguistic theory of construction grammar, and aims to operationalise the basic tenets of construction grammar as laid out in Chapter 2.4. According to constructionist approaches to language, constructions (i.e. form-meaning mappings) are the basic units of language and capture all linguistic knowledge. FCG provides a data structure for constructions, which serves as a skeleton. The information that can be added to the constructions is completely up to the grammar designer and can contain information from all levels of linguistic analysis. A construction can thus contain features that are phonetic, morpho-syntactic, semantic or pragmatic. Typically, the form features contain phonetical or morpho-syntactical information and the meaning features contain semantic or pragmatic information. Moreover, FCG follows the tenet that states that all constructions are on a lexicon-grammar continuum, meaning that there is no difference between a construction that captures a word or a grammatical rule. FCG uses the same data structure for all types of constructions, making no distinction between lexical or grammatical constructions. The set of constructions is not fixed, constructions can be added over time for example during the process of learning of constructions (Nevens et al., 2022; Doumen et al., 2023).

FCG does not impose any restrictions on the representation of the form and the meaning. Typically, a string is given as form. However, form features can consists of multimodal information, which can be useful when analysing sign languages (van Trijp, 2015). The choice of meaning representation is up to the user, with procedural semantics (Winograd, 1972; Woods et al., 1972) and Abstract Meaning Representation (Banarescu et al., 2013) as commonly used meaning representations. In particular, procedural semantics is used as a meaning representation for the fully operational grammar introduced in Chapter 4.

I will use the utterance "*What size is the red cube?*" as an example throughout this

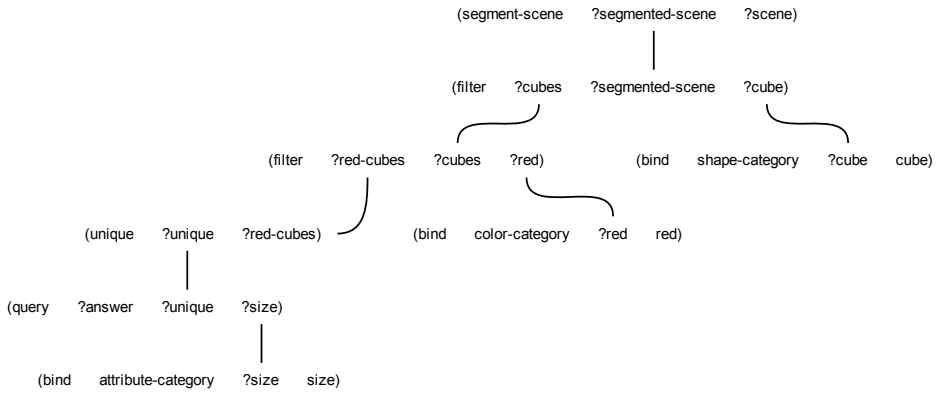


Figure 3.4: A procedural semantics representation underlying the utterance "*What size is the red cube?*". The meaning consists of the primitive SEGMENT-SCENE, followed by two FILTER operations, a UNIQUE and a QUERY primitive.

chapter. The meaning representation of this sentence is in terms of procedural semantics and consists of five primitives and three bind statements. Figure 3.4 shows the meaning network underlying the question.

The basic building blocks of Fluid Construction Grammar are discussed in the next section. First, I will expand on constructions, which are form-meaning mappings, followed by transient structures, which capture the linguistic information at a certain moment during language processing. Lastly, the construction application process, the interaction between constructions and transient structures, will be addressed.

3.3.1 Constructions

Construction grammar starts from the idea that the constructions are the basic building block of language. Each construction is a form-meaning mapping. FCG provides a data structure for constructions consisting of feature-value pairs. The choice of the feature-value pairs is completely up to the grammar designer, FCG has no requirements for certain features to be part of a construction. The constructions are stored in the construction inventory.

A construction contains pre- and postconditions, where the preconditions are the features that are required for a construction to apply and the postconditions are the features that are added after a construction applies. An example construction is shown in Figure 3.5. A typical construction consists of a conditional part and a contributing part. The units in the conditional part are divided in

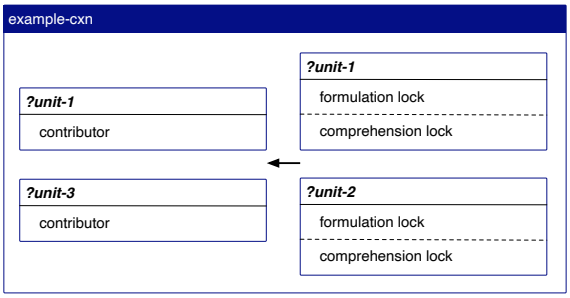


Figure 3.5: A schematic representation of a construction. Image adapted from Van Eecke (2018, p. 41).

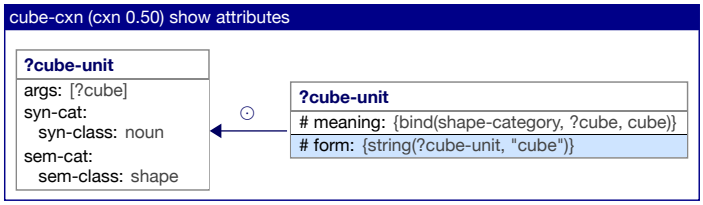
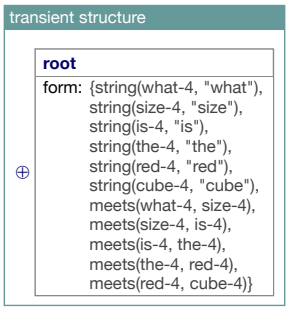


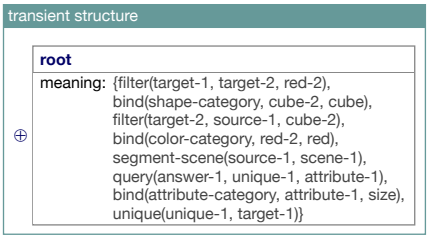
Figure 3.6: An example of a construction in FCG. The CUBE-CXN maps between the form "*cube*" and the meaning (BIND SHAPE-CATEGORY ?CUBE CUBE) and consists of a conditional part (right of the arrow) and a contributing part (left of the arrow).

preconditions for formulation (found in the formulation lock) and preconditions for comprehension (found in the comprehension lock). The features in the contributing part are the features that are added when a construction applies.

Figure 3.6 shows an example of an FCG construction, namely the CUBE-CXN. The CUBE-CXN is a lexical construction that looks for the string "*cube*" in comprehension and the meaning (BIND SHAPE-CATEGORY ?CUBE CUBE) in formulation. If the features in the lock are found, the features from the other lock are added, together with the features in the contributing part. In the example construction, the features in the contributing part are ARGS, SYN-CLASS and SEM-CLASS with as values ?CUBE, NOUN and SHAPE respectively. These features are chosen for illustrative purposes and other features can be used depending on the need. The features that are added can then be used as information for other constructions to apply on. For example, the NOMINAL-CXN can have as precondition that the SYN-CLASS needs to be a noun.



(a) Initial transient structure during comprehension of the sentence "*What size is the red cube?*".



(b) Initial transient structure during formulation of the meaning representation consisting of two FILTER primitives, a SEGMENT-SCENE primitive, a QUERY primitive, a UNIQUE primitive and two bind statements.

Figure 3.7: Initial transient structure during comprehension of the sentence "*What size is the red cube?*" (left) and formulation of the meaning underlying the sentence (right).

3.3.2 Transient structures

The transient structure is the data structure in FCG that holds the information that is added during the construction application process. It is the structure to which the postconditions are added and the preconditions of the constructions are checked against. As constructions, the transient structure consists of units with feature-value pairs.

The language processing starts with de-rendering the input into the *initial transient structure*. The de-rendering process depends on the direction of processing. In comprehension, the utterance that was given as input is first tokenised and each word is given a unique identifier. In formulation, the de-rendering process takes the meaning representation as a set of predicates and adds it to the initial transient structure. An example of the initial transient structure, which is the result of the de-rendering process when comprehending the sentence "*What size is the red cube?*" can be found in figure 3.7a. This initial transient structure consists of string and meets predicates stored in the *root*.

The initial transient structure serves as a starting point for the construction application process. During this process, the transient structure is used to check whether constructions can apply. The transient structure is updated as new information from the postconditions of applied constructions comes in. The updated transient structure is then the basis for checking the preconditions of other constructions.

3.3.3 Construction application process

The construction application process in FCG is a problem solving process (Bleys et al., 2011; Van Eecke, 2018; Van Eecke and Beuls, 2017; Steels and Van Eecke, 2018). A problem solving process is characterized by (i) operators, (ii) state representations (including the initial state) and (iii) goal tests (Russell and Norvig, 2009). The process starts from an initial state. The operators - if they can apply - change the state representation, and each time a state is changed, a goal test checks whether the current state satisfies the goal test. The process of applying operators goes on until the goal test is satisfied and a solution is found. For a detailed comparison between the construction application process and the 8-queens problem, an iconic example of a problem solving process in AI, I refer the reader to Van Eecke (2018); Steels and Van Eecke (2018) and Nevens (2022).

During the construction application process, the operators are the *constructions* and the state representations are the *transient structures*. In the comprehension process, the goal is to find a meaning representation that corresponds to the input form. In the formulation process, a form needs to be found for a given meaning representation. The transient structures capture the information during processing and the constructions operate on these transient structures. An example of a construction application process during comprehension of the question "*What size is the red cube?*" is shown in Figure 3.8. In this figure, the green nodes are the steps in the application process. The name of the construction that applied is shown in the node. The dark green colour of the last node indicates that a solution has been found. In each of the nodes, the transient structure that results from the application of the construction is shown.

Constructions consist of preconditions and postconditions. In order for a construction to apply, the preconditions need to be satisfied. In the comprehension process, the preconditions are found in the comprehension lock of the construction and if they *match* with the transient structure, the features in the formulation lock and the postconditions in the contributing part are *merged* into the transient structure. Both the *matching* and *merging* operations in FCG are unification algorithms (Steels and De Beule, 2006; De Beule, 2012). After matching and merging, the transient structure is updated and the goal tests are checked. In FCG, the hash operator is a special operator which checks whether the feature is in the *root*, if the feature is found, it is taken out of the root.

The goal test checks whether a node is a solution. The commonly used goal tests for comprehension are: (i) no-applicable-cxns, which checks whether there are no more constructions that can apply, (ii) no-strings-in-root, which checks

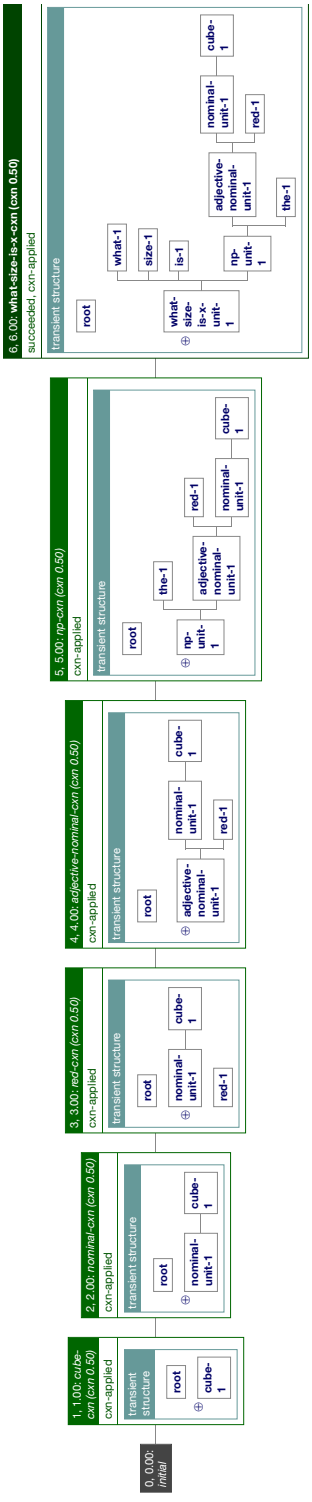


Figure 3.8: The construction application process during comprehension of the sentence *"What size is the red cube?"*. In total, six constructions applied. The solution node is indicated with a dark green colour.

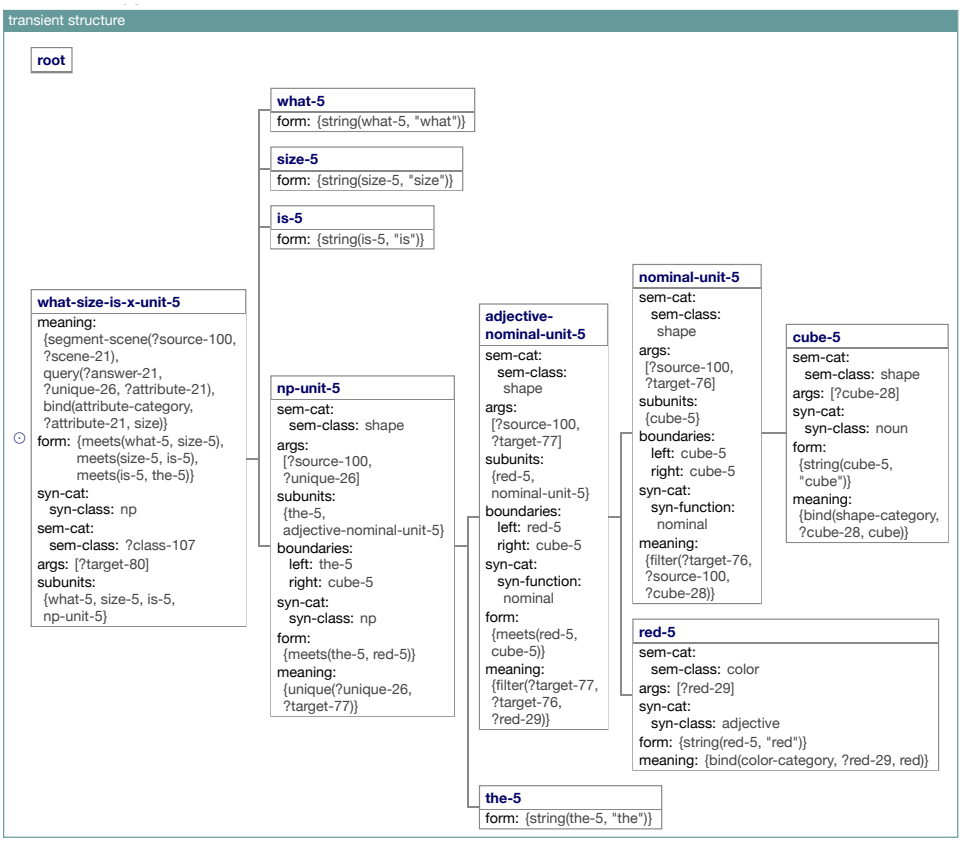


Figure 3.9: The transient structure at the end of comprehending the sentence "*What size is the red cube?*".

whether there are no more strings in the root, (iii) connected-semantic-network, which checks whether the semantic network is connected and (iv) connected-structure, which checks whether the transient structure is connected. For formulation, the goal tests typically are: (i) no-applicable-cxns, similar to the goal test in comprehension, (ii) no-more-meaning-in-root, which checks whether there are no more meaning predicates in the root, (iii) connected-structure, which checks whether the structure is connected.

3.3.4 Learning constructions

Up until now, this chapter mainly addressed the representation of constructions and transient structure and the construction application process. However, I did not yet discuss how the constructions are obtained. In FCG, there are two ways to acquire grammars: they can either be designed (i.e. written by a grammar designer) or they can be learned. In what follows, I discuss the learning of construction grammars through the mechanisms of *intention reading* and *pattern finding* as introduced by Nevens (2022), Nevens et al. (2022) and Doumen et al. (2023).

Recent major breakthroughs by Nevens (2022), Nevens et al. (2022) and Doumen et al. (2023) have introduced a set of generally applicable learning operators in FCG. Of course, these insights build further on a long tradition, starting with the learning of constructions for specific linguistic phenomena, for example a grammar for Russian aspect (Gerasymova and Spranger, 2010, 2012), English spatial expressions (Spranger and Steels, 2015; Spranger, 2015, 2017) and Hungarian agreement (Beuls et al., 2010). Later experiments have introduced more general learning operators applied to the English noun phrase (Van Eecke, 2018). These experiments have led to the development of the more generally applicable learning operators of Nevens (2022), Nevens et al. (2022) and Doumen et al. (2023) that enable the learning of grammars through situated communicative interactions.

The learning of construction grammars in FCG takes inspiration from child language acquisition which, according to Tomasello (2003, 2009), relies on two cognitive mechanisms: intention reading and pattern finding. Intention reading refers to the process of reconstructing the intended meaning of the observed utterance. Pattern finding is finding syntactico-semantic generalisations over form-meaning pairings. Intention reading and pattern finding are both operationalised in the Babel framework. On a technical level, the pattern finding operators are implemented in the meta-layer architecture of FCG (Van Eecke

and Beuls, 2017) and intention reading uses the composer in IRL (Van den Broeck, 2008). During the process of pattern finding, a network of grammatical categories that links the categories of the constructions emerges.

Currently, there are two main approaches to learn construction grammars in FCG. First, it is possible to learn from semantically annotated corpora using the pattern finding mechanism (Doumen et al., 2023). Second, a grammar can be learned through situated interactions where the meaning is not observed, but where a hypothesis of the meaning is created through intention reading, which the pattern finding mechanism then uses to make generalisations (Nevens et al., 2022). I will discuss both the pattern finding mechanism and the intention reading processes.

Learning constructions through pattern finding The goal of pattern finding is to learn syntactico-semantic generalisations (i.e. constructions) from semantically annotated corpora. Learning through pattern finding is integrated in the meta-layer architecture of FCG (Van Eecke, 2018; Van Eecke and Beuls, 2017). The meta-layer consists of a set of diagnostics and repairs (Beuls et al., 2012) that are able to construct constructions by finding the similarities and differences between previously learned constructions and the observation consisting of both a form and (the gold standard or reconstructed) meaning. The process starts when a diagnostic is triggered in the meta-layer, indicating that a linguistic observation cannot be comprehended. Then, a repair enables the learning of a holophrase construction, which consists of a mapping between an entire linguistic observation and its meaning. Other repairs are able to detect differences and similarities between the observation and previously learned constructions using anti-unification algorithms, in order to find generalisations and differences from which constructions can be learned. Figure 3.10 gives an example of the pattern finding process. The observation is the utterance "*How many spheres are there?*" and its meaning representation. The set of previously learned constructions contains, among others, the HOW-MANY-CUBES-ARE-THERE-CXN. However, with these constructions it is not possible to comprehend the utterance, so the meta-layer becomes active and the repair is triggered. Then, the repair anti-unifies the utterance and the form of the construction and finds the differences (i.e. "*cubes*" and "*spheres*") and the similarities (i.e. "*how many ?x are there?*"). Then, on the meaning side, the meaning of the observation is compared against the meaning of the construction and a generalisation and differences are found. From this generalisation, an item-based construction HOW-MANY-X-ARE-THERE-CXN can be made, which contains a slot that can be filled by the constructions

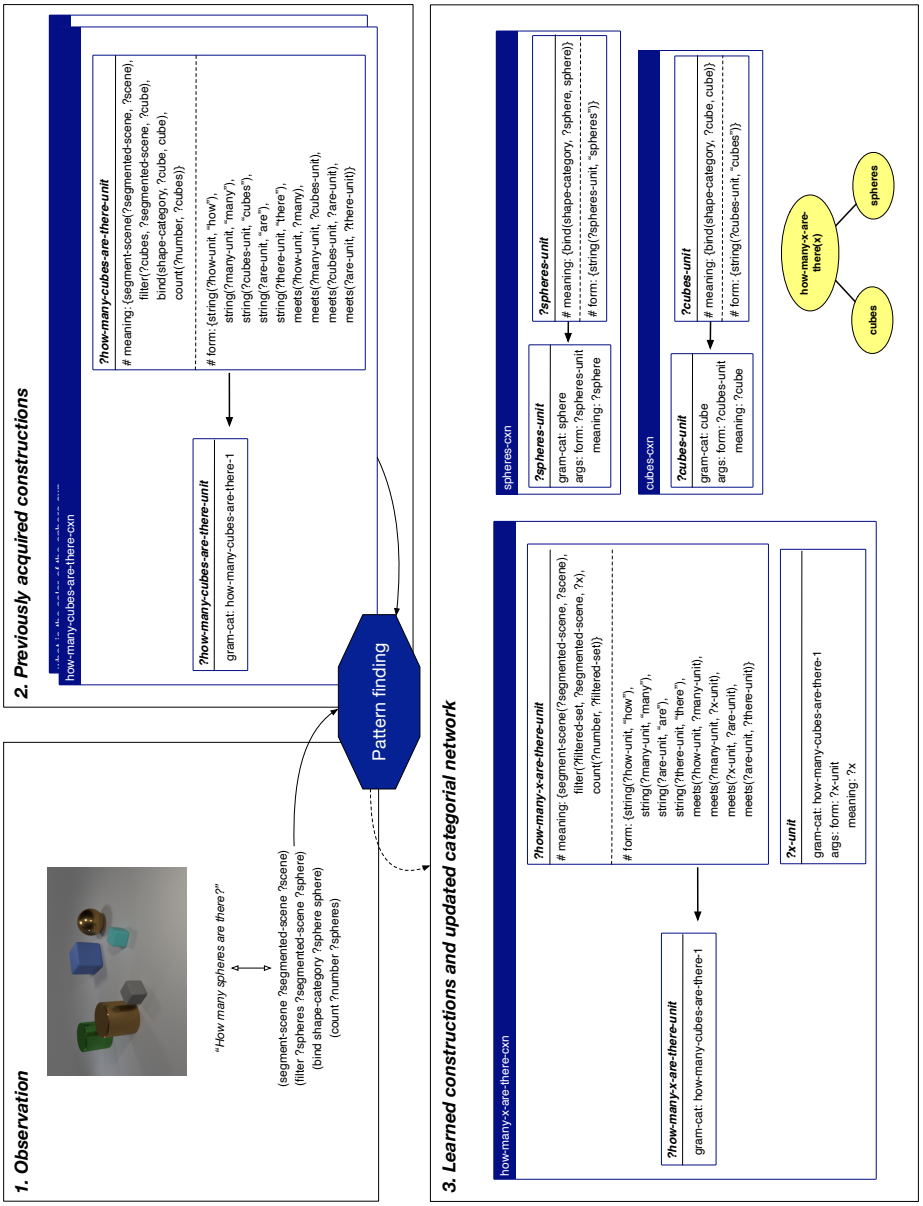


Figure 3.10: A schematic representation of the pattern finding process. Figure adapted from Beuls and Van Eecke (2023).

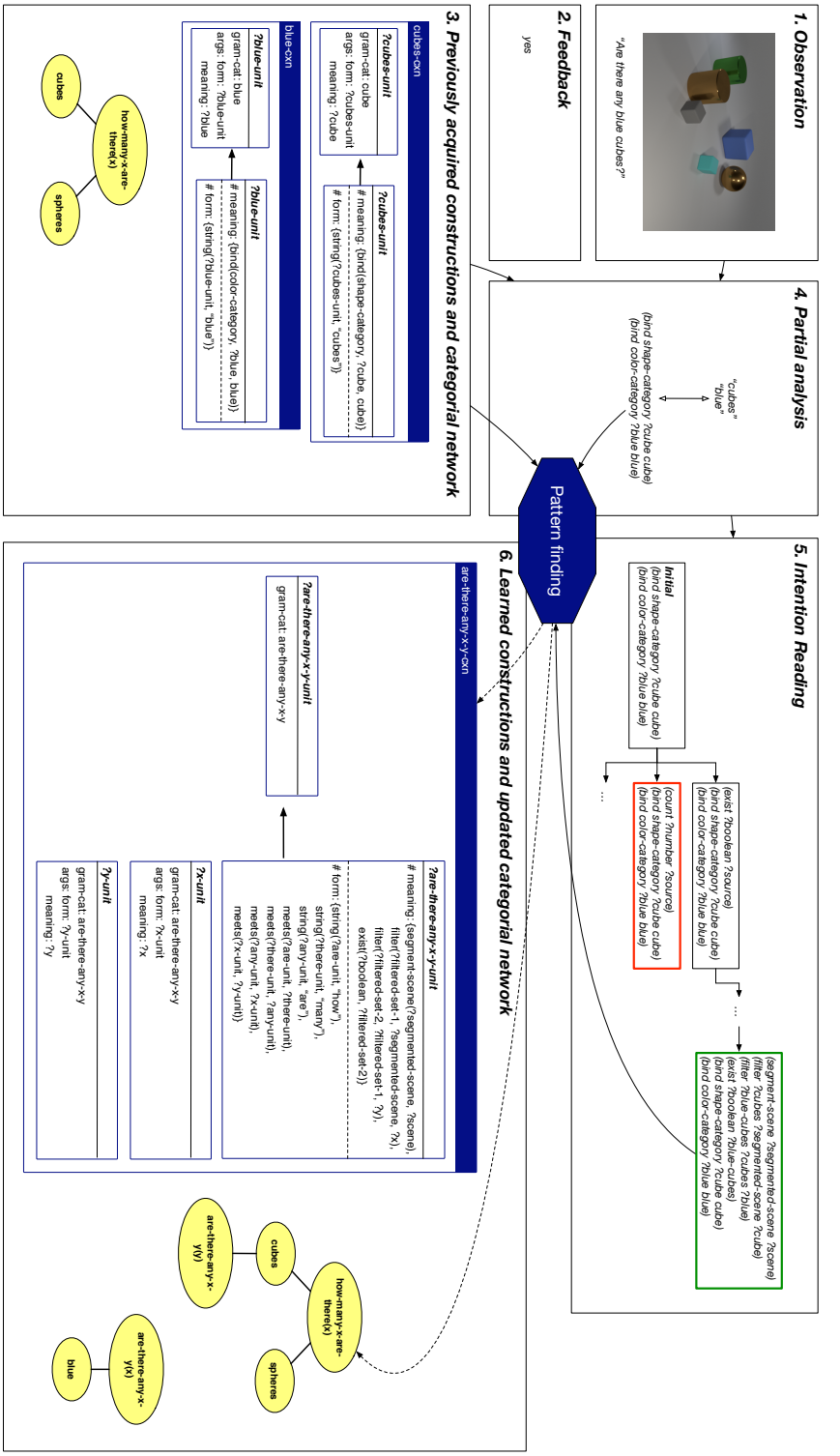


Figure 3.11: A schematic representation of the intention reading and pattern finding process. Figure adapted from Beuls and Van Eecke (2023).

that are learned from the differences on the form and the meaning side (i.e. the CUBES-CXN and the SPHERES-CXN). Crucially, these three constructions are linked to each other in the categorial network (Van Eecke, 2018, Ch. 4). For a detailed discussion on learning construction grammars through the pattern finding mechanism, I refer the reader to Doumen et al. (2023).

Learning constructions through intention reading and pattern finding A second way of learning construction grammars removes the need of starting from semantically annotated corpora. Due to the operationalisation of the processes of intention reading and pattern finding, grammars can be learned from the ground up, without observing the underlying meaning representation. In this case, the intention reading mechanism is integrated with the pattern finding operators. In these experiments, the meaning is not directly observed, but only a form of feedback (i.e. the answer to the question) is provided, from which a hypothesis of the meaning underlying the observed utterance is created. Specifically, intention reading is the process that is responsible for reconstructing the meaning representation and pattern finding then finds generalisations over the form-meaning pairs. As in the pattern finding experiment, the meta-layer is triggered if no solution is found when comprehending the observed utterance. When there is no success, a form of feedback is provided, which is the answer to the observed question in the case of the experiments in Nevens et al. (2022). The intention reading mechanism uses the feedback to reconstruct a meaning representation that leads to the answer. Then, pattern finding will use the composed meaning representation to generalise over the utterance and meaning. In some cases, the comprehension process in FCG already provides a partial analysis from constructions that could apply. Figure 3.11 shows the interplay between the intention reading and pattern finding processes. From the observation "*Are there any blue cubes?*", the feedback YES and the previously acquired constructions, the comprehension process in FCG provides a partial analysis. Two constructions (i.e. the CUBES-CXN and the BLUE-CXN) applied and added two bind statements to the meaning. Intention reading starts from this partial analysis and composes a meaning representation that leads to the answer YES. Then, pattern finding will create an item-based construction namely the ARE-THERE-ANY-X-Y-CXN that has two slots which could be filled with the existing BLUE-CXN and SPHERES-CXN. The link between the x slot in the ARE-THERE-ANY-X-Y-CXN and the BLUE-CXN and the link between the 'y' slot in the ARE-THERE-ANY-X-Y-CXN and the SPHERES-CXN are added in the categorial network, resulting in a network in which categories emerge. I refer the interested reader to (Nevens et al., 2022; Nevens, 2022) for a detailed overview on the intention reading and pattern finding mechanisms in

FCG.

3.3.5 Discussion

In this section, I discussed Fluid Construction Grammar as a framework for representing, processing and learning computational construction grammars. I focussed on the parts that are fundamental to understand the remainder of the thesis. Therefore, many features, such as the meta-layer (Van Eecke and Beuls, 2017), footprints, expansion operators, the categorial network (Van Eecke, 2018), construction sets, hashing or scoring constructions or neural heuristics (Van Eecke et al., 2022) could not be discussed in detail. For a more elaborate explanation of Fluid Construction Grammar, I refer the reader to the chapters on FCG in Van Eecke (2018) and Nevens (2022).

Furthermore, I discussed how computational construction grammar can be learned. Although it is a possibility to learn grammars, the grammars in this thesis (see Chapter 4, 5) are written by hand. The reason is that the main focus of the thesis is on designing adequate meaning representations and finding mechanisms to execute those meaning representations. Therefore, learning the grammars falls outside the scope of this thesis.

3.4 Conclusion

This chapter introduced the technical foundations underlying the remaining chapters of this thesis. Specifically, I introduced Incremental Recruitment Language (IRL) as a system for representing, processing and composing procedural semantic networks and Fluid Construction Grammar (FCG) as a framework for representing, processing and learning computational construction grammars. The processing of procedural semantics and construction grammars in IRL and FCG respectively, is operationalised through a search process. Both systems are part of the larger Babel framework and can thus seamlessly work together. FCG can be used as a way of mapping linguistic utterances onto a meaning representation that can then be executed by IRL, which is how the order of execution in the systems introduced in the next chapters (specifically Chapter 4 and 5). This process can also be reversed, so that IRL can be used to compose meaning representations that can then be produced through the production process in FCG. I want to note that this two-step process is a necessary simplification of the way the language understanding process of humans. Indeed, human-like language understanding is a process in which meaning is built up through inter-

preting the language at hand while consulting the necessary knowledge sources. In Chapter 4, I introduce an architecture that first maps linguistic utterances to a meaning representation by means of a hand-written grammar, then the meaning representation is executed by IRL. Again, in Chapter 5, I use a designed grammar to map linguistic utterances to meaning representations.

Chapter 4

Case study: Neuro-Symbolic Procedural Semantics for Visual Dialogue

4.1 Introduction

This chapter is based on Verheyen et al. (2023) and Verheyen et al., (Under Review). The web demonstration discussed in Verheyen et al. (2022b) accompanies this chapter and can be found at: `https://ehai.ai.vub.ac.be/demos/visual-dialog/`. I was the main contributor to the research presented here.

In this chapter, I introduce a system that models the human-like capacity of grounding language into the environment and the discourse. The methodology presented here is designed to tackle the task of visual dialogue, in which an agent needs to hold a meaningful and coherent conversation with a human interlocutor discussing an image. To solve this task, it is thus needed to ground the linguistic utterances in both the situational context (i.e. the image) and the discourse context. Specifically, I validate the introduced methodology on two datasets: CLEVR-Dialog (Kottur et al., 2019) and MNIST Dialog (Seo et al., 2017). The system consists in a hand-written computational construction grammar operationalised through the Fluid Construction Grammar framework that can map questions and statements from the two datasets to a procedural semantics representation. This meaning representation can be executed in a neuro-symbolic way using the Incremental Recruitment Language framework. Further, I introduce a novel data structure, i.e. the conversation memory, which represents the information conveyed in the dialogue in an incremental and explicit way. Moreover, the system is interpretable by design, which I illustrate through a number of examples.

In what follows, I first give an introduction to the task of visual dialogue and the proposed methodology (Section 4.2). Then, I discuss the background and related work (Section 4.3), situating the contribution of this work with respect to earlier work on visual dialogue (Section 4.3.1) and procedural semantics (Section 4.3.2). In Section 4.4, the novel methodology for solving visual dialogue tasks using a neuro-symbolic procedural semantic representation that integrates with a conversation memory is discussed in detail. Section 4.5 presents two experiments in which the method is applied to the MNIST Dialog and CLEVR-Dialog benchmark challenges. Finally, Section 4.6 reflects on the results, contributions and impact of the work.

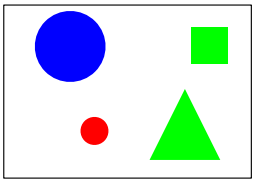
4.2 A novel methodology for the task of visual dialogue

Visual dialogue refers to the task in which an artificial agent and a human hold a meaningful and coherent conversation that is grounded in visual input (Das et al., 2017). Typically, an agent needs to answer a sequence of questions about a given image, where the questions can only be understood in relation to previous question-answer pairs. In many respects, the task of visual dialogue is similar to the task of visual question answering (Antol et al., 2015), with the additional difficulty that the question-answer pairs are not independent from each other.

A schematic depiction of a typical visual dialogue task is shown in Figure 4.1. In this example, an agent is presented with the image on the left, and needs to answer the sequence of questions Q_1 to Q_4 on the right. The four question-answer pairs constitute a coherent dialogue, in which Q_1 ("*Are there any triangles?*") can be answered based on the image alone, but in which Q_2 to Q_4 ("*How many?*", "*Is there an object to its left?*", "*What is its colour?*") can only be answered based on the combination of the image and the previous question-answer pairs.

In this chapter, I introduce the use of neuro-symbolic procedural semantic representations for solving visual dialogue tasks. This method builds further on earlier work in the area of visual question answering, in which procedural semantic representations, as pioneered by amongst others Winograd (1972), Woods et al. (1972) and Johnson-Laird (1977), have already been successfully used for representing the meaning of questions in the form of executable queries (Andreas et al., 2016a; Johnson et al., 2017b; Nevens et al., 2019a). Such procedural semantic representations capture the logical structure underlying a question, and can be executed on a given image to compute an answer.

An example of a procedural semantic representation for the question "*Are there more squares than circles?*", asked about the image in Figure 4.1, is shown in



Q_1: Are there any triangles?
A_1: **Yes**

Q_2: How many?
A_2: **One**

Q_3: Is there an object to its left?
A_3: **Yes**

Q_4: What is its colour?
A_4: **Red**

Figure 4.1: Schematic representation of a typical visual dialogue task, in which an artificial agent needs to answer a sequence of follow-up questions about an image.

Figure 4.2. The query is composed of six operations that need to be performed by an artificial agent in order to retrieve the answer to the question. First of all, the SEGMENT-SCENE operation segments the image that it received as input (bound to the variable ?SCENE) and binds the set of foreground objects to the ?SEGMENTED-SCENE variable. Then, two FILTER operations take this set of objects as input and bind the set of squares and the set of circles to the variables ?SQUARES and ?CIRCLES respectively. Then, the set of squares and the set of circles are counted by COUNT operations and the cardinality of each set is computed. Finally, the GREATER-THAN operation checks whether the cardinality of the first set is larger than the cardinality of the second set. The result of this last operation (in this case NO) is at the same time the answer to the question as a whole.

The operations, which are also called *primitive operations* or *primitives*, correspond to atomic actions that an artificial agent can perform. Depending on the techniques used for implementing these operations, procedural semantic representations can be *subsymbolic*, cf. the neural module networks used by Andreas et al. (2016a), or *symbolic*, cf. the set operations used by Nevens et al. (2019a). Here, I combine the strengths of both subsymbolic and symbolic operations through the introduction of *neuro-symbolic procedural semantic representations*, cf. the approach used by Manhaeve et al. (2021).

When moving from visual question answering to visual dialogue, the two-step process of first mapping a question to its logical structure and then executing the corresponding query on an image becomes more challenging. As individual questions are no longer independent from each other, they no longer map onto queries that are directly executable on an image alone. For example, in the question "*What is its colour?*", the possessive anaphoric pronoun "its" refers to an object that was introduced by an earlier question-answer pair, and which must be retrieved in order to be able to answer the question. As opposed to visual question answering systems, visual dialogue systems thus need to be able to keep track of the information that has been conveyed during earlier dialogue turns, as well as to use this information for answering questions in later turns.

In order to overcome this challenge, I introduce the use of a *conversation memory* as a data structure that explicitly and incrementally stores the information that is expressed in the subsequent turns of a dialogue. Additionally, I present a procedural semantic representation for visual dialogue tasks, which is able to query both visual input and the conversation memory. Due to its neuro-symbolic nature, this semantic representation can exploit the strengths of both

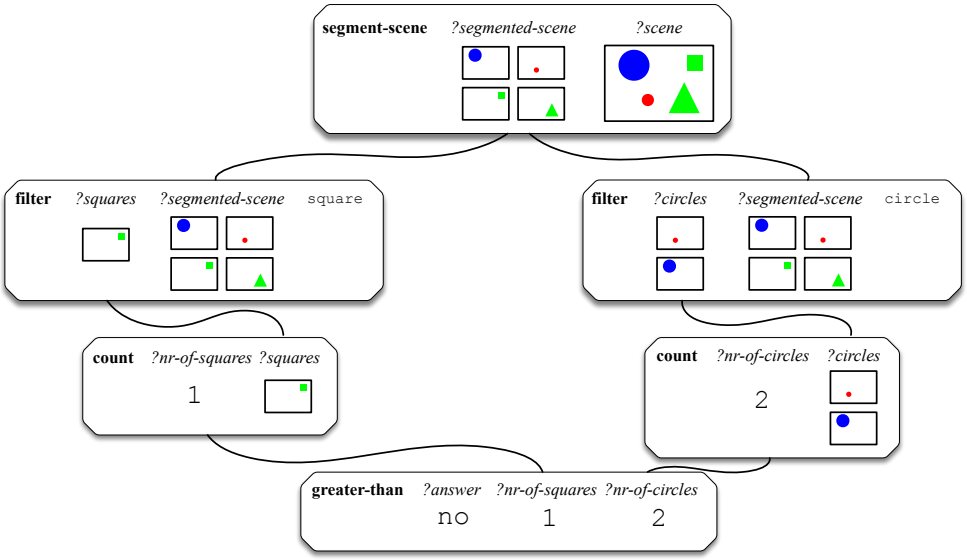


Figure 4.2: Example of a procedural semantic representation for the question "*Are there more squares than circles?*", executed on the image in Figure 4.1. The answer to the question given this image is NO.

subsymbolic systems for interacting with perceptual data, in this case the image, and of symbolic systems for reasoning based on previously acquired knowledge, in this case by retrieving structured information from the conversation memory.

The evaluation of the novel methodology on the standard MNIST Dialog benchmark (Seo et al., 2017) and the more challenging CLEVR-Dialog benchmark (Kottur et al., 2019) shows that through the introduction of a conversation memory and the design of a compatible neuro-symbolic procedural semantic representation, I have been able to transfer the success of using procedural semantics in the field of visual question answering to the much more challenging field of visual dialogue. Presenting a methodology that tackles visual dialogue tasks by reasoning over both structured (memory) and unstructured (image) data, contributes to the growing body of research in artificial intelligence that tackles tasks that involve both low-level perception and high-level reasoning using a combination of neural and symbolic techniques (Manhaeve et al., 2021; Evans et al., 2021; Badreddine et al., 2022). It thereby bears the promise of leading to the development of artificial agents with more explainable, consistent and human-like cognitive capacities.

4.3 Background and related work

This section sketches the background and prior work that forms the backbone of the research. In particular, I focus on the state of the art in the fields of visual dialogue (4.3.1) and procedural semantics (4.3.2).

4.3.1 Visual dialogue

Agents holding coherent conversations with humans about the scenes they observe has been a central topic in the field of artificial intelligence since its inception in the 1950s, with SHRDLU (Winograd, 1972) and Shakey (Nilsson, 1984) being the most notable early systems developed. More recently, also the machine learning community has become increasingly interested in the topic of artificial agents holding coherent conversations with humans about visual content. This has led to the establishment of the standardised task of *visual dialogue* as introduced by Das et al. (2017), and subsequently to a number of dedicated datasets and benchmark challenges, including VisDial (Das et al., 2017), MNIST Dialog (Seo et al., 2017) and CLEVR-Dialog (Kottur et al., 2019). The task of visual dialogue can be seen as an extension of the task of visual question answering (Antol et al., 2015). While both tasks involve answering questions about images, the questions in a visual dialogue task are organised in a coherent conversation and can involve reference to entities introduced by earlier question-answer pairs. The additional challenge faced by visual dialogue systems amounts thus to taking into account earlier dialogue turns when answering later questions.

The state of the art in visual dialogue is dominated by attention-based neural network approaches, which mainly differ in how they deal with co-references between question-answer pairs. In general, these approaches use an encoder-decoder architecture (Sutskever et al., 2014), which learns to attend to those regions of the image and/or previous question-answer pairs that are most relevant to answering a given question. Das et al. (2017) introduce encoders based on late fusion, hierarchical encoding (Serban et al., 2017) and memory networks (Bordes et al., 2017). These encoders encode the question, textual history and image, and identify those parts of the textual history that are most relevant to answering the question. A discriminative decoder can then be used to rank candidate answers, or a generative decoder can be used to produce an answer. Lu et al. (2017) present history-conditioned image attentive encoders which do not only encode the question, textual history and entire image, but also attend over specific regions in the image that played a role in the dialogue history.

Jain et al. (2018) integrate answer options as early input to the model, as to maximally exploit their informativeness. Building further on this approach, Guo et al. (2019) introduce a two-stage process, in which the candidate answers are first scored by a co-attention network. This ranking is then passed as input to a second co-attention network during a so-called synergistic stage. Wu et al. (2018) propose a co-attention encoder which jointly reasons over the input image, the question and the previous question-answer pairs. This encoder is in turn part of a larger architecture for adversarial learning, which learns to approximate human answers using a reinforcement learning-based discriminator. Schwartz et al. (2019) extend this work by presenting a more general co-attention-based model that can include any number of input modalities. Kang et al. (2019) propose the use of dual attention networks for resolving visual co-references. Linguistic co-references are resolved by a first attention module, and the corresponding entities are then grounded in the image by a second attention module. Gan et al. (2019) introduce a recurrent dual attention network that performs multi-step reasoning, integrating visual and textual reasoning in an iterative process. Niu et al. (2019) introduce an algorithm that recursively traverses earlier question-answer pairs based on co-references, in order to retrieve visual attentions for the relevant entities. Zheng et al. (2019) propose a graph neural network approach to visual dialogue, where the nodes are dialogue turns and the edges represent co-reference links between these turns. Answering a question amounts then to inferring unknown node values. Yang et al. (2019) present a history-aware co-attention network that is robust against imperfect history input. Their learning approach, called history-advantage sequence training, is inspired by actor-critic methods in reinforcement learning in the sense that it includes an adversarial critic which intentionally introduces wrong answers with the goal of improving robustness. Zhang et al. (2019) propose a weighted likelihood estimation method for training generative decoders, with the goal of making them less biased towards frequent answers such as "*I don't know*". Wang et al. (2020) integrate pre-trained BERT language models into a transformer-based encoder. Li and Moens (2021) extend this approach by integrating soft linguistic constraints, encoding preference for specific part-of-speech tags and closeness between pronouns and their antecedents.

A next line of research focusses on more explicitly keeping track of the entities that were evoked in earlier dialogue turns, both visually and textually, and on resolving co-references and ambiguities with respect to these entities. Starting from the observation that the proportion of follow-up questions with non-trivial co-references in existing visual dialogue datasets, in particular VisDial, is limited

(Massiceti et al., 2018; Agarwal et al., 2020), Seo et al. (2017) introduce the MNIST Dialog dataset with the specific purpose of evaluating to what extent visual dialogue models are actually capable of reasoning about previously introduced discourse entities. MNIST Dialog is characterised by a large proportion of interdependent questions that are highly ambiguous with respect to synthetically generated scenes, unless co-references are adequately resolved. As the scenes and questions are bias-free, the questions cannot be answered without reasoning about both the scene and dialogue history. In the same paper, the authors introduce a model that explicitly represents the dialogue history as a combination of previous question-answer pairs and their associated attentions. From this associative memory, the model is able to retrieve the relevant attention for a given question. Building further on this work, Kottur et al. (2018) also represent the dialogue history in the form of an associative memory, but the keys are here more fine-grained entity-level descriptions instead of question-answer pairs. The authors introduce a neural module network architecture (Andreas et al., 2016b) in which the meaning representation includes two dedicated modules (REFER and EXCLUDE) for interacting with the associative memory. Cho and Kim (2021) extend Kottur et al. (2018)'s model with a separate treatment of personal and impersonal pronouns. Kottur et al. (2019) introduce the CLEVR-Dialog dataset for studying and benchmarking multi-turn reasoning in visual dialogue. This dataset was developed as a more challenging alternative to MNIST Dialog, where questions cannot only depend on the previous question-answer pair, but also on any combination of earlier question-answer pairs. Shah et al. (2020) introduce three extensions of memory, attention and composition (MAC) networks (Hudson and Manning, 2018) that deal with the conversational nature of visual dialogue tasks. A first extension consists in passing information across dialogue turns by initialising the memory state of the first MAC-cell of each turn with the value of the memory state of the last MAC-cell of the previous turn. A second extension concerns a context-aware attention mechanism that implements a transformer-like self-attention mechanism on the previous control states. A final extension consists in appending the entire dialogue history to the current question. They report that all three techniques lead to important improvements with respect to the state of the art. Finally, the neuro-symbolic approach of Abdessaied et al. (2022) keeps track of the mentioned entities by using a dynamic knowledge base which can be queried by their fetch operation.

4.3.2 Procedural semantics

Procedural semantic representations, as pioneered by Woods (1968), Winograd (1972) and Johnson-Laird (1977) capture the meaning of linguistic expressions in the form of programs that can be executed algorithmically. The use of procedural semantics is of particular interest to conversational agents, especially when these agents need to be able to truly understand linguistic expressions as uttered by a human, for example in the case of instructions to be carried out in the world or questions to be answered in natural language. The procedural semantics paradigm was indeed the result of a number of ambitious research projects in this direction in the 1960s and 1970s. The SHRDLU system (Winograd, 1972) was able to hold coherent conversations with a human about a blocks world. It could move blocks as instructed by the human, reason about actions and affordances, and answer questions about both actions and the state of the world. SHRUDLU's rule-based grammar and reasoning system were not only able to understand and produce English utterances, but could also ask for clarifications when the system was unable to disambiguate input utterances. The LUNAR system (Woods et al., 1972) enabled lunar geologists to query chemical analysis data on lunar rock and soil composition using natural language, without having to learn a formal query language or the structure of NASA's databases. English utterances were analysed by an augmented transition network (ATN)-based parser and then mapped onto queries that could be executed on the databases. Steedman and Johnson-Laird (1978) took this approach further, by introducing semantic transition networks (STNs). As compared to ATNs, STNs are able to directly build up a semantic representation, instead of needing to pass through an intermediate syntactic structure. Since then, this pioneering work has given rise to a broad spectrum of procedural semantics-based question answering systems. While the coverage and applicability of these systems have drastically improved over time, the conversational aspects that were once the hallmark of SHRDLU, have gradually moved away from the focus of attention.

Over the last decades, procedural semantic representations have been extensively used in systems for querying databases using natural language, in combination with a variety of grammar formalisms. Warren and Pereira (1982) introduce the use of an extension of definite clause grammars (Pereira and Warren, 1980), called extraposition grammars, to parse natural language questions into logic-based executable queries. Zelle and Mooney (1996) introduce an inductive logic programming approach to learn definite clause grammars and Kanazawa (2007) uses definite clause grammars to parse natural language questions into efficient datalog queries. A large body of work embraces combinatory

categorial grammar (CCG; Steedman, 1987) as a semantic parsing engine that maps natural language utterances onto logical forms expressed in the lambda calculus (Zettlemoyer and Collins, 2005; Kwiatkowski et al., 2010; Krishnamurthy and Mitchell, 2012; Berant et al., 2013; Cai and Yates, 2013; Reddy et al., 2014; Pasupat and Liang, 2015). Other work adopts Head-Driven Phrase Structure Grammar (HPSG) (McFetridge et al., 1996; Frank et al., 2007), computational construction grammar (Nevens et al., 2019a), dependency parsing (Andreas et al., 2016b) or variations on context-free grammars (Wong and Mooney, 2007; Huang et al., 2008). Apart from grammar-based approaches, also neural approaches have been used to map questions onto executable queries, in particular using recurrent neural networks such as LSTMs (Andreas et al., 2016a; Dong and Lapata, 2016; Zhong et al., 2017; Cheng et al., 2019).

When it comes to the properties of the procedural semantic representations themselves, three different approaches can be distinguished. A first class of models represent the meaning of utterances as queries expressed in a database querying language, such as SQL (Zhong et al., 2017), FunQL (Cheng et al., 2019) or SPARQL (Yahya et al., 2012). The main advantage of this approach is that the expressiveness of the semantic representation coincides with the expressiveness of the query language, and that the semantic representations can be directly executed on a database. The main disadvantages of this approach are that only questions can be represented straightforwardly and that the structure of the queries is often far removed from the way in which information is represented in natural language. A second class of models represent the meaning of questions using logical forms, often defined in terms of variations on the lambda calculus (see e.g. the work cited above in the context of CCG). Such representations are more expressive, can represent more sentence types, and more closely mirror the compositional nature of linguistic utterances. However, an additional step is needed to transform the logical forms to executable queries. The third class of models use formalisms that were especially designed to represent the meaning of natural language utterances using procedural semantic representations. These formalisms typically provide a way to define so-called *primitive operations*, which correspond to functions or predicates that can be implemented computationally. These primitive operations can be compositionally combined into larger programs, often called *semantic networks*, through shared input and output arguments. These programs can then be evaluated by executing the individual primitive operations while propagating the appropriate arguments from one operation to the other. Examples of models of this class include meaning representations expressed in Incremental Recruitment Language (IRL;

Van den Broeck, 2008; Spranger et al., 2012), as used for example by Pauw and Hilferty (2016) and Nevens et al. (2019a), or the meaning representations used by Andreas et al. (2016a), Johnson et al. (2017a) and Andreas et al. (2020). While the primitive operations used in these special-purpose procedural semantics languages need to be implemented or learnt, this approach has the advantage that the languages are open-ended and directly executable. Moreover, this means that the procedural semantic languages can be tailored towards the task at hand, and that the primitive operations and their combination can be designed to better reflect the compositional nature of natural language utterances.

Primitive operations in procedural semantics can be operationalised symbolically or subsymbolically. Subsymbolic primitives perform operations over numeric representations such as scalars, vectors or tensors. They usually deal with the categorisation of sensor values observed in the world, often extracted from images. Symbolic primitives on the other hand perform operations over meaningful symbols, typically implementing higher-level reasoning processes. Neuro-symbolic procedural semantic systems allow to combine symbolic and subsymbolic primitives in semantic networks. In these networks, subsymbolic primitives typically deal with perception tasks, while symbolic primitives typically deal with reasoning tasks. Procedural semantic representations of all three types have been proposed. Neural module networks have been introduced by Andreas et al. (2016b) as an operationalisation of fully subsymbolic procedural semantic representations applied to visual question answering tasks. Kottur et al. (2018) extend this approach to visual dialogue by adding primitive operations that perform multi-turn co-reference resolution. Yi et al. (2018), Mao et al. (2019), Abdessaied et al. (2022) and Nevens et al. (2019a) present a symbolic approach where the procedural semantic representations are not executed on the image directly, but on a scene graph representation that is generated first. Finally, Manhaeve et al. (2018, 2021) propose a neuro-symbolic procedural semantic engine which integrates neural predicates in probabilistic logic programs and Badreddine et al. (2022) present a framework aimed at representing fully differentiable logic representations.

4.4 Methodology

The novel approach to visual dialogue operationalises two main ideas. First, the history of a dialogue is represented explicitly, incrementally and in a structured way. I refer to the data structure holding this information by the term *conversation memory*. Second, the meaning of linguistic utterances is repre-

sented using a *neuro-symbolic procedural semantic representation* that combines subsymbolic and symbolic primitive operations. The conversation memory and neuro-symbolic procedural semantic representation are presented in Section 4.4.1 and 4.4.2 respectively. The full methodology is as follows. Each utterance of a dialogue goes through the same process. First, the utterance is mapped onto a procedural semantics representation using computational construction grammar (see 4.4.4). Then, this procedural semantics is executed in a neuro-symbolic way (see Section 4.4.2). The primitive operations that perform co-reference resolution or require reasoning are executed symbolically, the primitive operations that perform perception are executed subsymbolically, relying on a set of neural modules (see Section 4.4.3). In case of a question, the execution leads to an answer. In case of a caption, the procedural semantics is just executed. In both cases, the conversation memory is updated based on the procedural semantic representation (see Section4.4.5).

4.4.1 Conversation memory

The conversation memory captures all information about the dialogue history that can be relevant for interpreting later dialogue turns. It represents this information in an explicit, human-interpretable way, and is incrementally extended after each dialogue turn. Per turn, the conversation memory stores:

- a timestamp capturing the turn number.
- the utterance observed during the turn.
- the sentence type of this utterance, indicating for example the question type for questions.
- the reply that was produced, if applicable.
- the topic of the conversation from an information structure point of view.
- a symbolic representation of the set of all entities evoked during the dialogue up to this turn, including all their properties that were mentioned.
- for each entity, a pointer to an attention over the image that highlights its grounding in the input.

As an example, Figure 4.3 shows the state of the conversation memory after processing the dialogue introduced in Figure 4.1. For now, I only briefly introduce the conversation memory data structure. Section 4.4.5 will then describe in

more detail the information captured in the conversation memory, how this information is extracted from the dialogue, and how it is added. The conversation memory in the figure holds information about four subsequent dialogue turns. In the first turn, the question "*Are there any triangles?*" of type QUESTION-EXIST is observed and the answer "*Yes*" is returned. The topic of the conversation at this point is the entity OBJECT-1. Both the grounding of entity OBJECT-1 in the input image and its mentioned shape property are stored in the conversation memory. In the second turn, the question "*How many?*" of type QUESTION-COUNT is asked about the current topic of the conversation and the answer "*One*" is returned. The topic of the conversation does not change and no additional information is added. In the third turn, the question "*Is there an object to its left?*" of type QUESTION-EXIST is processed and the answer "*Yes*" is returned. A new entity OBJECT-2 is added to the conversation memory with as only information its grounding in the input image. The topic of the conversation now shifts to entity OBJECT-2. Finally, at the fourth turn, the question "*What is its colour*?" is processed. The topic of the conversation, namely OBJECT-2, is inferred from the previous turn and the answer "*Red*" is returned. The colour property of OBJECT-2 is added to the representation of this entity in the conversation memory.

In general, the conversation memory should store after each dialogue turn all discourse information that might be relevant for interpreting later dialogue turns. The information that is included in the implementation of the conversation memory reflects the information that is relevant in the visual dialogue tasks that are tackled in Section 4.5. I do not claim in any way that this information is sufficient to model everyday conversations between human interlocutors, which fall outside the scope of these benchmark challenges. Further research in pragmatics is needed in order to construct more accurate models of the role that discourse information plays in human conversation.

4.4.2 Neuro-symbolic procedural semantics

In tandem with the conversation memory, I introduce a neuro-symbolic procedural semantic representation that is designed to represent the meaning of utterances in their discourse context. The set of primitive operations that is part of the semantic representation is an extension of the set of predicates used in the annotation of the CLEVR VQA dataset (Johnson et al., 2017a). On the one hand, this extension was made for the procedural semantic representation to be applicable to a larger number of datasets, and on the other hand to be able to deal with the conversational aspects of dialogue through the consultation of information stored in the conversation memory.

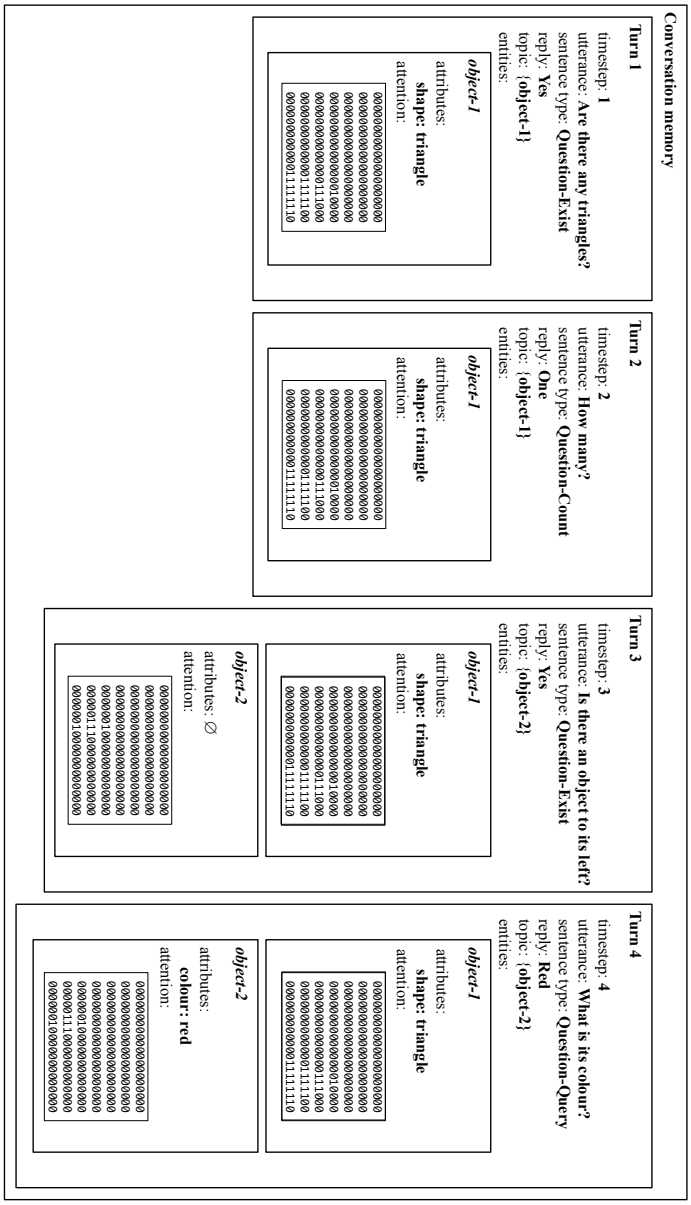


Figure 4.3: Schematic representation of the conversation memory after the fourth turn of the dialogue sketched in Figure 4.1. The conversation memory is incrementally updated after each dialogue turn as new information becomes available.

The neuro-symbolic procedural semantic representation combines subsymbolic primitives that implement operations over unstructured data, in particular input images or attentions, with symbolic primitives that implement operations over structured data, in particular information contained in the conversation memory. Primitives that can operate on both structured and unstructured input have both a symbolic and a subsymbolic implementation. At runtime, the adequate implementation is then chosen based on the type of the input arguments.

The neuro-symbolic procedural semantic representation makes use of 16 primitive operations, which can combine to represent the meaning of statements and questions about objects in an image. The statements and questions can be about the existence and number of objects in the image, their attributes and the spatial relationships between the objects. The primitive operations are defined and implemented as described below. A schematic representation of the internal architecture of each primitive operation is also provided in Figure 4.4 and an overview of the different primitive operations as categorised by their symbolic or subsymbolic nature is shown in Table 4.1.

- The SEGMENT-SCENE(?SEGMENTED-SCENE, ?SCENE) primitive operation binds a segmentation of the input image bound to ?SCENE to the ?SEGMENTED-SCENE variable, i.e. a set of attentions in which each attention highlights one of the objects in the image. This primitive operation is implemented subsymbolically as a Mask R-CNN-based neural network that performs instance segmentation (He et al., 2017). The SEGMENT-SCENE primitive is used in the representation of the meaning of each statement or question about an image. For example, it serves as a starting point for computing an answer to the question "*Are there any green cylinders?*".

- The FILTER(?TARGET-SET, ?SOURCE-SET, ?SCENE, ?CATEGORY) primitive operation binds ?TARGET-SET to the set of all instances of ?CATEGORY present in ?SOURCE-SET. ?CATEGORY needs to be bound to a conceptual category to filter by, such as GREEN, CUBE or LARGE. The filter operation is implemented both symbolically and subsymbolically. The symbolic implementation is used to filter entities from the conversation memory by binding the set of all entities from the ?SOURCE-SET set that have ?CATEGORY among their symbolic attributes to ?TARGET-SET. The subsymbolic implementation classifies each attention in ?SOURCE-SET according to whether it fits best the class ?CATEGORY in ?SCENE or a different class of the same attribute category. The set of attentions that are predicted to belong to class ?CATEGORY are bound to ?TARGET-SET. This classification process is implemented on top of the

shared inventory of neural modules discussed later in this section. The subsymbolic implementation of the filter primitive is for example used to compute the set of green objects when processing the utterance "*Are there any green objects?*". The symbolic implementation is for example used to compute the set of green objects when processing the utterance "*How many cubes are there among the aforementioned green objects?*".

- The RELATE(?TARGET-SET, ?SOURCE-OBJECT, ?SEGMENTED-SCENE, ?SCENE, ?SPATIAL-RELATION) primitive operation binds ?TARGET-SET to the set of all attentions in ?SEGMENTED-SCENE for which ?SPATIAL-RELATION holds with respect to ?SOURCE-OBJECT. For example, if ?SPATIAL-RELATION is bound to RIGHT, ?TARGET-SET will be bound to the set of all attentions over objects that are located to the right of ?SOURCE-OBJECT. This primitive operation is implemented on top of the shared inventory of neural modules discussed later in this section. It classifies each attention in ?SEGMENTED-SCENE according to whether it is ?SPATIAL-RELATION with respect to ?SOURCE-OBJECT in ?SCENE. The primitive is used for example to compute the set of objects located to the right of a green sphere when processing the utterance "*How many objects are to the right of the green sphere?*".

- The EXTREME-RELATE(?TARGET-OBJECT, ?SOURCE-SET, ?SCENE, ?SPATIAL-DIRECTION) primitive operation binds ?TARGET-OBJECT to the attention in ?SOURCE-SET over the object that is located most towards the spatial direction described by ?SPATIAL-DIRECTION. For example, if ?SPATIAL-DIRECTION is bound to RIGHT, ?TARGET-OBJECT will be bound to the attention over the rightmost object present in ?SOURCE-SET. This primitive operation is implemented on top of the shared inventory of neural modules discussed later in this section. The primitive is used for example to compute the rightmost object when processing the utterance "*What is the colour of the rightmost object?*".

- The IMMEDIATE-RELATE(?TARGET-OBJECT, ?SOURCE-OBJECT, ?SEGMENTED-SCENE, ?SCENE, ?SPATIAL-RELATION) primitive operation binds ?TARGET-OBJECT to the attention in ?SEGMENTED-SCENE over the object in ?SCENE that is located most closely to ?SOURCE-OBJECT according to ?SPATIAL-RELATION. For example, if ?SPATIAL-RELATION is bound to RIGHT, ?TARGET-OBJECT will be bound to the attention over the object in ?SCENE that is located most closely to the right of ?SOURCE-OBJECT. This primitive operation is implemented on top of the shared inventory of neural modules discussed later in this section. The primitive is used for example to compute the object that is located most closely to the right of the green sphere in the utterance "*What is the shape*

of the object right of the green sphere?".

- The UNIQUE(?TARGET-OBJECT, ?SOURCE-SET) primitive operation checks whether the set bound to ?SOURCE-SET contains only one attention. If this is the case, it binds ?TARGET-OBJECT to this attention. If ?SOURCE-SET is empty, the primitive signals failure. If ?SOURCE-SET contains more than one attention, it triggers a search process with as many branches as there are attentions in ?SOURCE-SET. Each attention in ?SOURCE-SET is bound to ?TARGET-OBJECT in exactly one branch with the average confidence score of the attention accumulated over any previous primitives taken as the heuristic value of the branch. The UNIQUE primitive is implemented through symbolic set operations. It is for example used for processing utterances that contain articles, such as "*What is the material of the green sphere?*" or "*There is a green object left of a red object.*".

- The QUERY(?TARGET-CATEGORY, ?SOURCE-OBJECT, ?SCENE, ?ATTRIBUTE-CATEGORY) primitive operation queries the ?ATTRIBUTE-CATEGORY of ?SOURCE-OBJECT and binds the resulting value to ?TARGET-CATEGORY. ?ATTRIBUTE-CATEGORY needs to be bound to the name of an attribute category, such as SHAPE, COLOUR or SIZE. The resulting values are conceptual categories such as BLOCK, RED or LARGE. This primitive operation is implemented on top of the shared inventory of neural modules discussed later in this section. Based on ?ATTRIBUTE-CATEGORY (e.g. size), a subset of binary classifiers associated to this ?ATTRIBUTE-CATEGORY is selected (e.g. large, small). The category associated to the binary classifier yielding the highest confidence score (for a positive result) is bound to ?TARGET-CATEGORY. The query primitive is used to query properties of objects, for example the material of the green sphere in the utterance "*What is the material of the green sphere?*".

- The COUNT(?TARGET-NUMBER, ?SOURCE-SET) primitive operation binds the cardinality of ?SOURCE-SET to ?TARGET-NUMBER. This primitive operation is implemented through a symbolic set operation. An example utterance that requires the COUNT primitive is the question "*How many spheres are there?*".

- The EXIST(?TARGET-BOOLEAN, ?SOURCE-SET) primitive operation checks whether the set bound to ?SOURCE-SET contains at least one element. If so, ?TARGET-BOOLEAN is bound to YES, otherwise to NO. This primitive operation is implemented through symbolic set operations. An example of an utterance requiring the EXIST primitive is the question "*Are there any spheres?*".

- The MORE-THAN-ONE(?TARGET-BOOLEAN, ?SOURCE-SET) primitive operation checks whether the set bound to ?SOURCE-SET contains multiple elements (i.e. at least two). If so, ?TARGET-BOOLEAN is bound to YES, otherwise to NO. This primitive operation is implemented through symbolic set operations. An example of an utterance that requires the MORE-THAN-ONE primitive is the statement "*There are multiple spheres in the image.*".

- The EXIST-OR-COUNT(?TARGET, ?SOURCE-SET, ?CONVERSATION-MEMORY) primitive operation calls either the EXIST primitive operation or the COUNT primitive operation on '*?source-set*' and binds the result to '*?target*'. Whether the exist or count operation is called, depends on the sentence type of the previous turn in '*?conversation-memory*'. This primitive operation is implemented through symbolic operations on the conversation memory and through calls to other primitive operations. For example, if a question '*and to its right*?' follows a count-type question such as '*How many objects are there to the left of the green cube?*', the count primitive will be used to count the objects to the right of the green cube. If the same question follows an exist-type question such as '*Are there any objects to the left of the green cube?*', the exist primitive will be called to determine whether there are any objects to the right of that green cube.

- The GET-TOPIC(?TARGET-TOPIC, ?CONVERSATION-MEMORY) primitive operation binds ?TARGET-TOPIC to the current topic of the conversation as stored in ?CONVERSATION-MEMORY, i.e. the set of objects that is the topic of the conversation after processing the previous turn. This primitive operation is implemented symbolically. It is used to resolve anaphora in questions such as "*and its colour*?", following questions such as "*What is the shape of the small object left of the green cube?*", which shifted the topic to the small object left of the green cube.

- The GET-PREVIOUS-TOPIC(?TARGET-TOPIC, ?CONVERSATION-MEMORY) primitive operation binds ?TARGET-TOPIC to the previous topic of the conversation, i.e. the set of objects that was the topic before processing the last turn. This primitive operation is implemented symbolically. It is used to resolve anaphora in questions such as "*and to its left?*" following questions such as "*Are there any objects to its right?*", which follow themselves questions such as "*Is there a green cube?*". In this case, the question "*and to its left?*" refers to the green cube and not to the objects to the right of the green cube.

- The GET-ATTRIBUTE-CATEGORY(?TARGET-CATEGORY, ?CONVERSATION-MEMORY) primitive operation binds '?TARGET-CATEGORY' to the attribute category that

was queried most recently in the conversation. This primitive operation is implemented symbolically and is used to resolve anaphora in utterances such as "*and that of the green sphere?*" following utterances such as "*What is the material of the grey cylinder?*".

- The FIND-IN-SCENE(?TARGET-OBJECT-SET-SCENE, ?SOURCE-OBJECT-SET-SCENE, ?SOURCE-OBJECT-SET-MEMORY) primitive operation relates one or more objects from the conversation memory with their counterparts in the input image. Concretely, this operation takes as input a set of entities stored in the conversation memory, bound to ?SOURCE-OBJECT-SET-MEMORY, and the attentions bound to ?SOURCE-OBJECT-SET-SCENE. It then finds the attentions of the entities from the conversation memory in the scene and binds this set to ?TARGET-OBJECT-SET-SCENE. This primitive is implemented symbolically as a straightforward lookup function. The FIND-IN-SCENE primitive is used to resolve anaphora in utterances such as "*What is its material?*" following utterances such as "*Is there a green cube?*". Here, the FIND-IN-SCENE primitive relates the representation of the green cube as retrieved from the conversation memory with the green cube as observed in the image.

- The SET-DIFFERENCE(?TARGET-OBJECT-SET-SCENE, ?SOURCE-OBJECT-SET-SCENE, ?SOURCE-OBJECT-SET-MEMORY) primitive operation binds ?TARGET-OBJECT-SET-SCENE to the subset of ?SOURCE-OBJECT-SET-SCENE that contains all attentions over objects that are not part of ?SOURCE-OBJECT-SET-MEMORY. It does this by first using the find-in-scene primitive to retrieve the attentions over the objects in ?SOURCE-OBJECT-SET-MEMORY and then subtracting these from ?SOURCE-OBJECT-SET-SCENE. This primitive is implemented through symbolic functions. The SET-DIFFERENCE primitive is used to process utterances that explicitly refer to objects that were not previously mentioned, for example in the utterance "*Are there other objects sharing its colour?*". Here, the word "*other*" refers to the set of objects in the scene that do not appear in the conversation memory.

The subsymbolic primitive operations that query attributes of objects (QUERY), that filter objects based on their attributes (FILTER), and that spatially relate objects to each other (RELATE, EXTREME-RELATE and IMMEDIATE-RELATE) are all implemented on top of a shared inventory of neural modules. These modules are implemented as binary classifiers that are trained to predict whether a specific conceptual categorisation holds for a given object or set of objects in a scene. They should be interpreted as atomic distinctions that underlie the conceptual reasoning of an agent operationalised through a variety of primitive operations.

Table 4.1: Overview of primitive operations categorised by their symbolic or subsymbolic implementation.

symbolic	*subsymbolic*
FILTER	FILTER
UNIQUE	SEGMENT-SCENE
COUNT	RELATE
EXIST	EXTREME-RELATE
MORE-THAN-ONE	IMMEDIATE-RELATE
EXIST-OR-COUNT	QUERY
GET-TOPIC	
GET-PREVIOUS-TOPIC	
GET-ATTRIBUTE-CATEGORY	
FIND-IN-SCENE	
SET-DIFFERENCE	

Table 4.2: Overview of the shared inventory of neural modules on top of which the subsymbolic primitive operations are built. All modules are implemented as binary classifiers adopting the SqueezeNet architecture (Iandola et al., 2016).

Shared inventory of neural modules			
colour-blue?	relate-behind?	extreme-relate-right?	style-stroke?
colour-red?	relate-left?	extreme-relate-front?	style-flat?
colour-brown?	relate-right?	extreme-relate-middle?	number-0?
colour-green?	relate-front?	size-small?	number-1?
colour-cyan?	immediate-relate-behind?	size-large?	number-2?
colour-grey?	immediate-relate-left?	bgcolour-white?	number-3?
colour-purple?	immediate-relate-right?	bgcolour-cyan?	number-4?
colour-yellow?	immediate-relate-front?	bgcolour-salmon?	number-5?
colour-violet?	immediate-relate-above?	bgcolour-silver?	number-6?
shape-cube?	immediate-relate-below?	bgcolour-yellow?	number-7?
shape-cylinder?	extreme-relate-behind?	material-metal?	number-8?
shape-sphere?	extreme-relate-left?	material-rubber?	number-9?

Using a shared inventory of highly-specialised neural modules across different primitive operations, as opposed to training a dedicated neural module for each subsymbolic primitive operation, has two main advantages. First, it enhances the consistency of the overall reasoning process, as the different reasoning steps make use of the same conceptual representations and inferences (see Section 4.6). Second, it facilitates the addition of new primitive operations as they can maximally reuse cognitive capacities that have previously been acquired. All binary classifiers are convolutional neural networks that adopt the SqueezeNet architecture (Iandola et al., 2016). An overview of the neural modules is shown in Table 4.2 and full details on their implementation and evaluation are provided in the following Section.

4.4.3 Neural modules

This section provides full details on the architecture and training regime of the neural modules underlying the subsymbolic primitive operations. The neural modules perform either instance segmentation (those used by the SEGMENT-SCENE operation) or binary classification (those used by the QUERY, FILTER, RELATE, EXTREME-RELATE and IMMEDIATE-RELATE operations).

Modules performing instance segmentation

The instance segmentation module is implemented through a Mask R-CNN model (He et al., 2017). This module takes as input an image and returns a set of as many visual attentions as there are objects in the scene, with each attention highlighting one of the objects. The instance segmentation module is implemented using the Detectron2 framework (Wu et al., 2019). It consists in a model for instance segmentation pretrained on the COCO images dataset (Lin et al., 2014), which is then finetuned on the CLEVR mini dataset (Yi et al., 2018). For finetuning, a batch size of 8, a learning rate of 0.00025 and 10,000 iterations are used.

While the same architecture could also be used for processing the MNIST Dialog images, instance segmentation is not really an issue when it comes to this dataset. As all images consist of the same 4x4 grid layout, it is straightforwardly dividable into 16 visual attentions. As a consequence, the instance segmentation module is not needed.

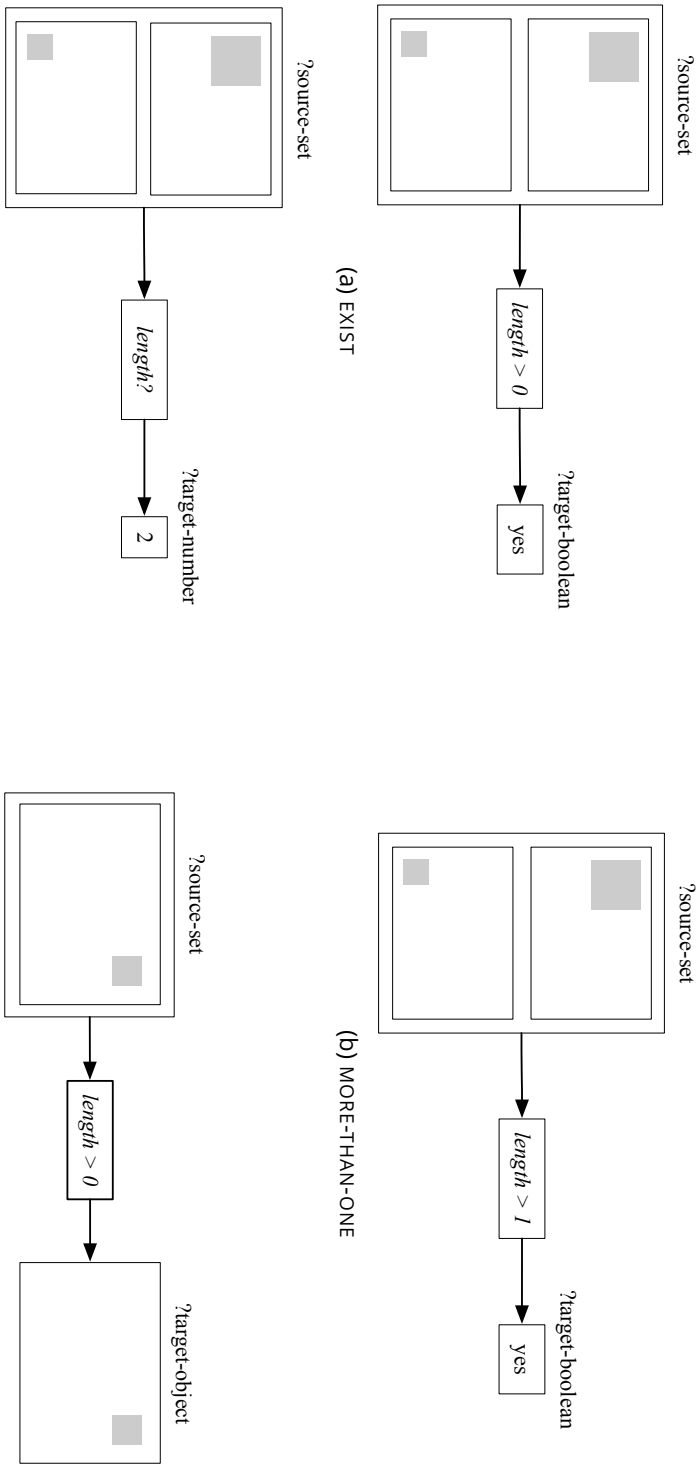
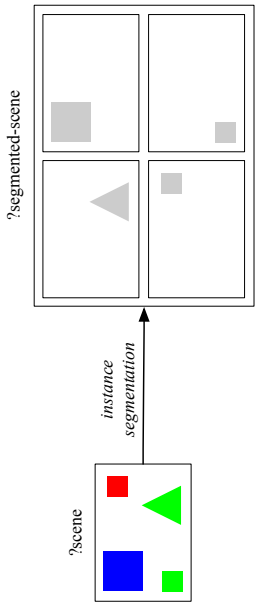
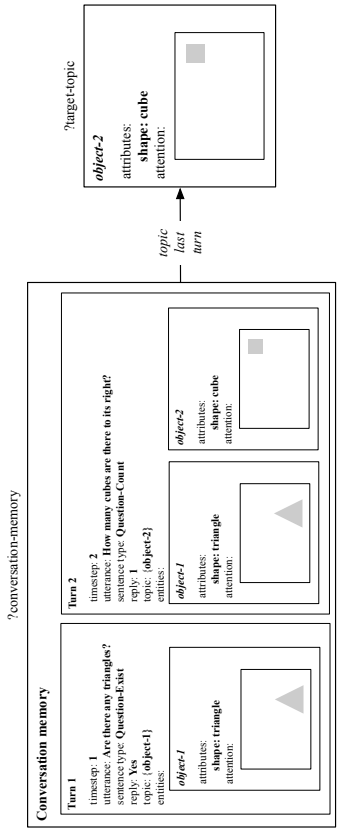


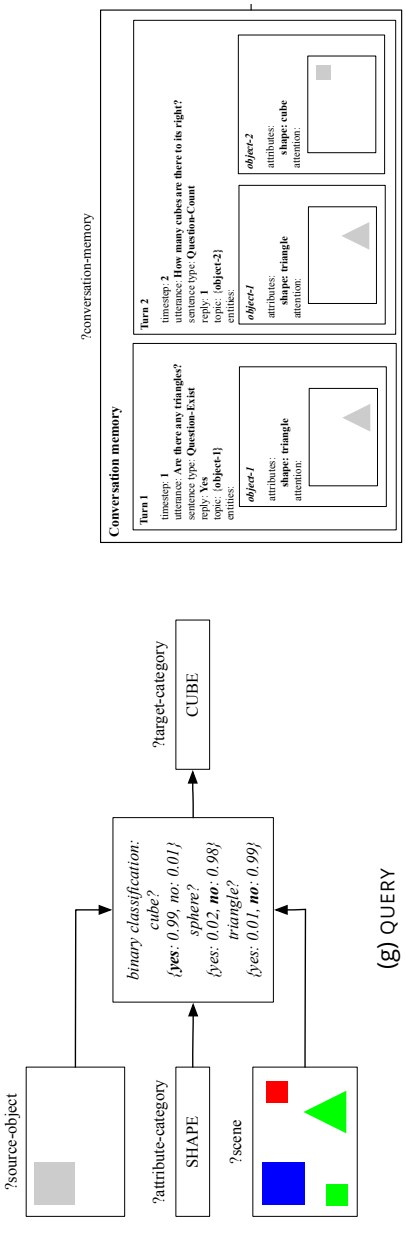
Figure 4.4: Schematic representation of the implementation of the primitive operations.



(e) SEGMENT-SCENE

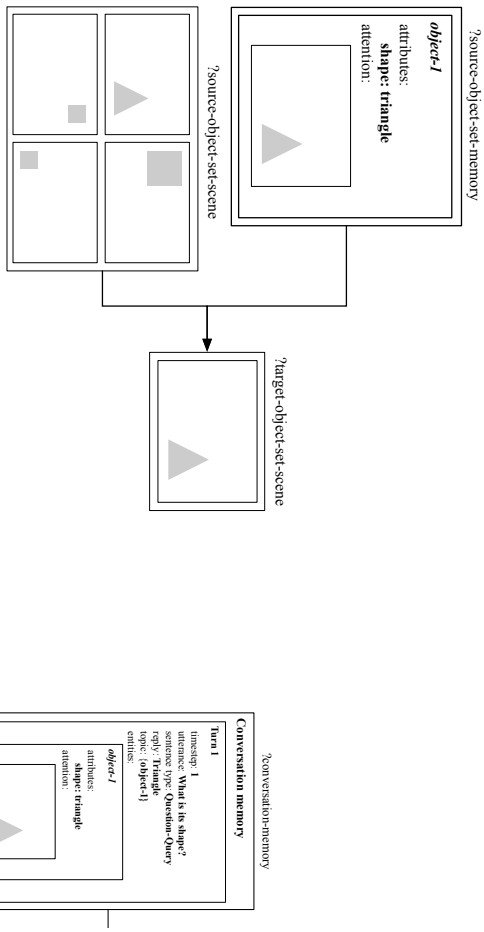


(f) GET-TOPIC

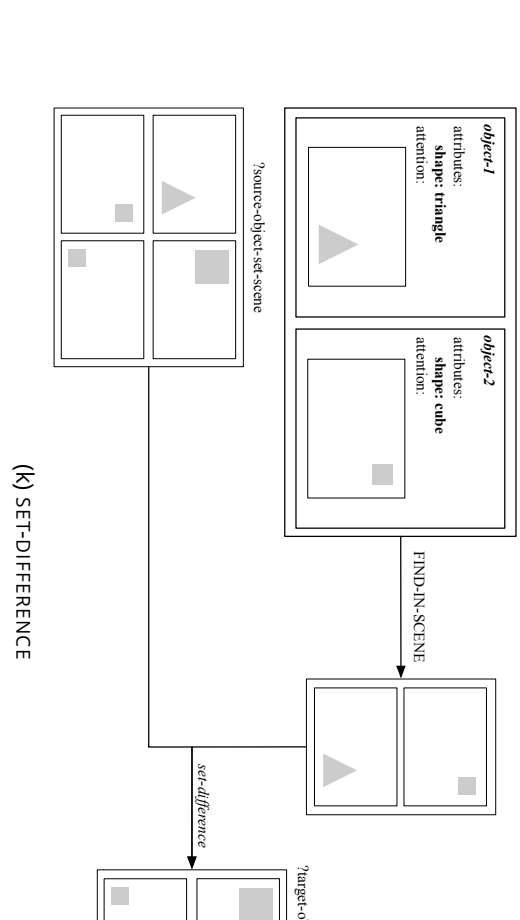


(g) QUERY

(h) GET-PREVIOUS-TOPIC



(i) FIND-IN-SCENE



(j) GET-ATTRIBUTE-CATEGORY

(k) SET-DIFFERENCE

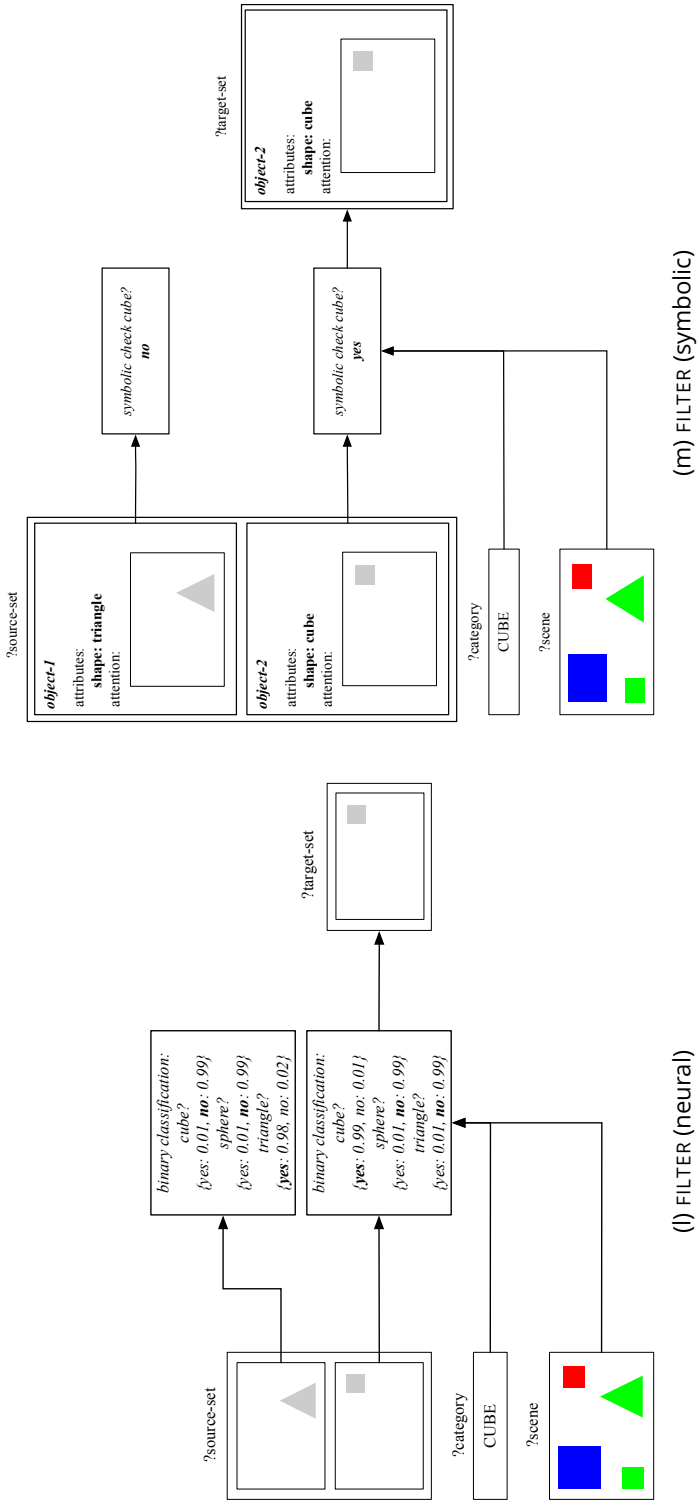
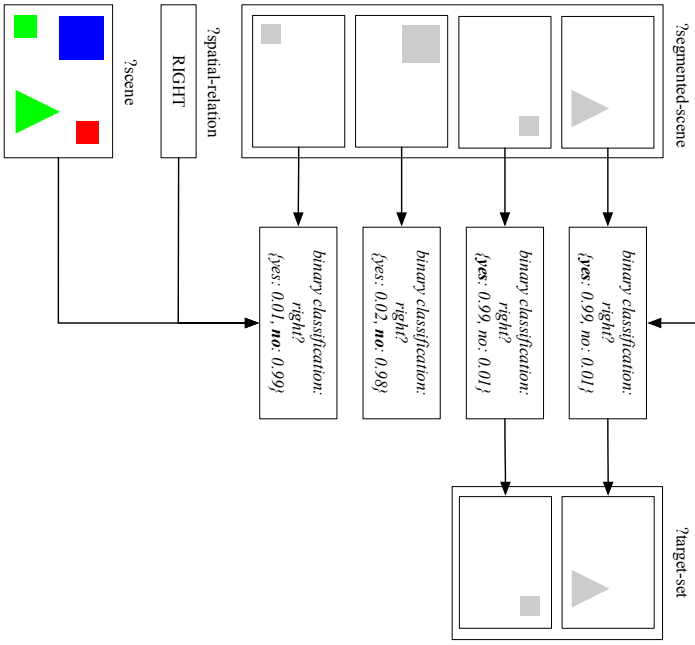
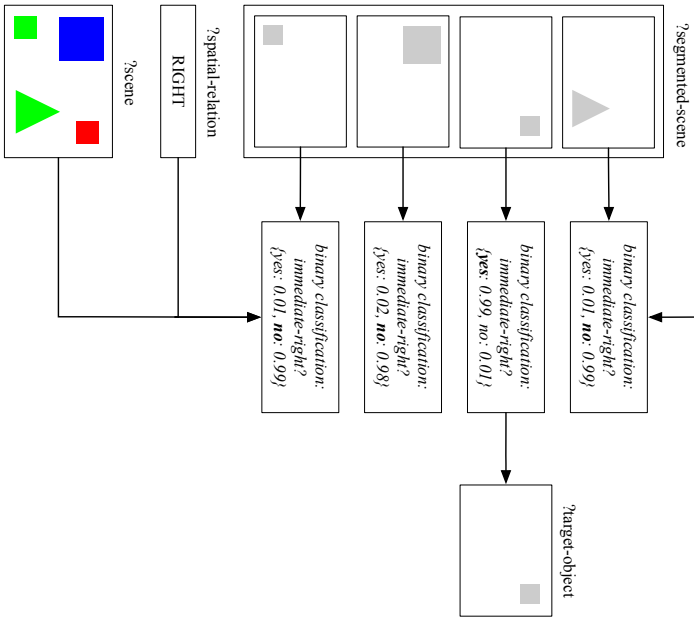


Figure 4.4: Schematic representation of the implementation of the primitive operations.

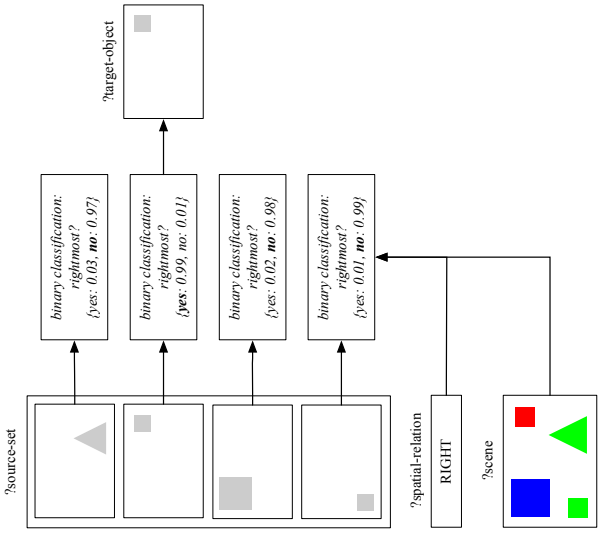


(n) RELATE

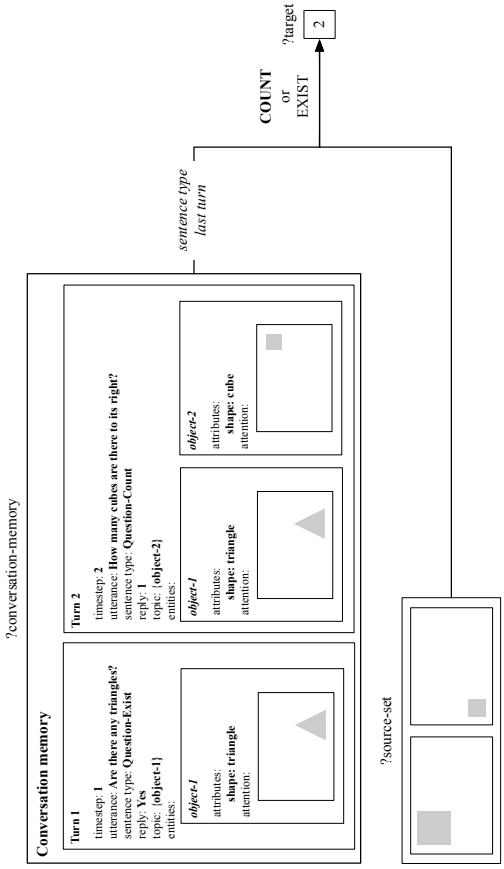


(o) IMMEDIATE-RELATE

Figure 4.4: Schematic representation of the implementation of the primitive operations.



(q) EXTREME-RELATE



(p) EXIST-OR-COUNT

Figure 4.4: Schematic representation of the implementation of the primitive operations.

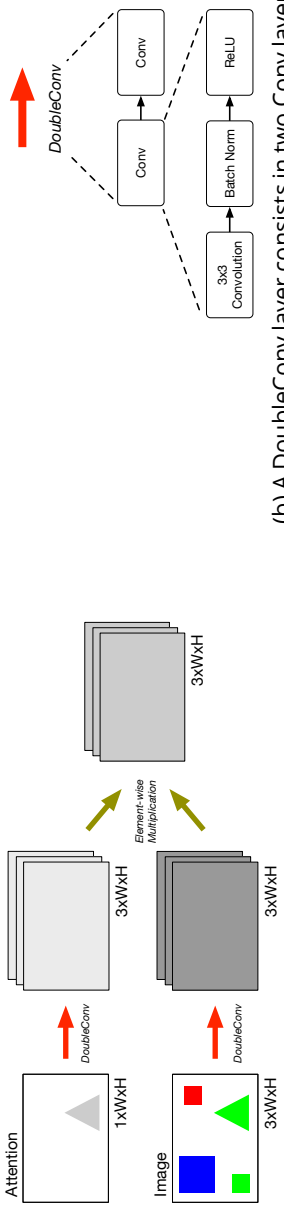
Modules performing binary classification

The binary classification modules are trained to predict whether a specific conceptual categorisation holds for a given object or set of objects in a scene. The binary classifiers are each implemented by a convolutional neural network (CNN) adopting the SqueezeNet architecture (Iandola et al., 2016) in its 1.1 version[1]. As the SqueezeNet architecture expects a single RGB image as input, the modules first combine their different inputs into a single tensor using a pre-encoding layer, as illustrated in Figure 4.5a for modules with two inputs (i.e. the colour, material, style, size, shape, number, bgcolour and extreme-relate modules) and in Figure 4.5c for modules with three inputs (i.e. the relate and immediate-relate modules). Both pre-encoding layers make use of a *DoubleConv* operation, which is shown in Figure 4.5b. A DoubleConv operation consists in two *Conv* operations, which each consist in a convolutional layer with kernel size 3 and padding 1, followed by a batch normalisation operation (Ioffe and Szegedy, 2015) and a rectified linear unit (ReLU). In general, the first Conv operation in this DoubleConv operation changes the number of channels while the second Conv operation keeps the number of channels the same.

When pre-encoding two inputs, each of the inputs first passes through a DoubleConv layer. If the input is an attention, the first Conv in this DoubleConv layer increases the number of channels from 1 tot 3. If the input is an image, the number of channels does not change. Then, the resulting feature maps are combined using element-wise multiplication, which results in a $3xWxH$ tensor. When pre-encoding three inputs, the image and the two attentions each pass through a distinct DoubleConv layer, which increases the number of channels from 1 to 3 for the attentions and keeps the number of channels the same for the image. Each of these DoubleConv operations is followed by a max pooling operation, which halves the width and height dimensions of the feature maps. Then, the resulting feature maps of the image are concatenated with the resulting feature maps of the first attention. The concatenation then passes through another DoubleConv layer, which doubles the number of channels from 6 to 12. The same steps apply for the image and the second attention. The two resulting tensors are then concatenated and passed a final time through a DoubleConv layer, which reduces the number of channels to three, resulting in a $3xW_{/2}xH_{/2}$ tensor.

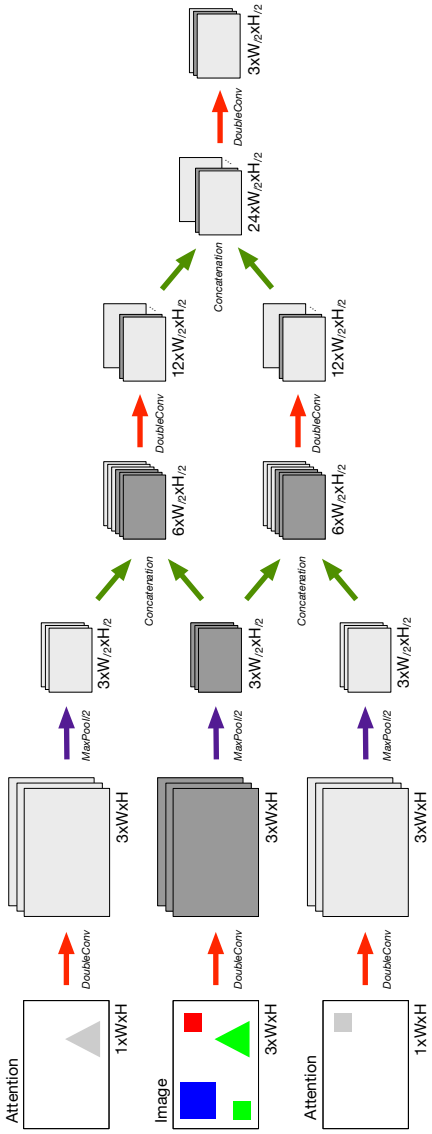
The result of the pre-encoding layer is then passed to the SqueezeNet architecture, followed by a LogsSoftMax layer. The output is binary and consists of

[1] `https://github.com/forresti/SqueezeNet/tree/master/SqueezeNet_v1.1`



(a) The architecture of the pre-encoding layer with two inputs.

(b) A DoubleConv layer consists in two Conv layers, each consisting in a 3x3 convolution, followed by a batch norm and a ReLU.



(c) The architecture of the pre-encoding layer with three inputs.

Figure 4.5: Schematic overview of the pre-encoding operations.

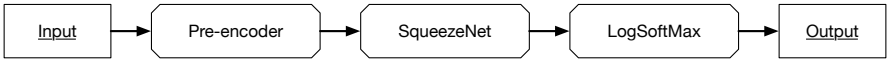


Figure 4.6: The architecture of the modules. The input goes through a pre-encoder, then through the SqueezeNet architecture, followed by a LogSoftMax layer.

predictions for the labels 'yes' and 'no'. The overall architecture of the modules is shown in Figure 4.6.

In order to train and evaluate the neural modules, one dataset per module per benchmark is generated. For CLEVR-Dialog, I have used images 0-59,999 of the original training set as training data and images 60,000-69,999 of the original training set as validation data. Then, the instance segmentation module described above was used to find all instances of objects in these images. The Euclidean distance between the coordinates specified in the metadata and the coordinates of the predicted instances is then computed in order to link the object instances to their symbolic description in the metadata of the dataset. The result is a dataset in which each predicted instance (in the form of a visual attention) is accompanied by a symbolic description of its attributes. Based on this dataset, one dataset per module was then generated, which consists of instances annotated with a label 'yes' or 'no'. The correct label is found via the symbolic meta-data. A total of 28 datasets were generated, one for each CLEVR-Dialog module.

For MNIST Dialog, the dataset generation process was similar. The 30,000 images from the training set are used as training data and the 10,000 images from the validation set as validation data. First, all instances in the images are found by dividing the image into 16 attentions. Then, these instances are linked to their symbolic description using the index provided in the meta-data. This resulted in a dataset with all instances paired with their symbolic description. Based on this dataset, one dataset per module was generated, with consists in all the instances accompanied by a label that indicates whether the instance has the attribute or not. A total of 26 datasets were generated, one for each of the MNIST Dialog modules.

The hyperparameters used for training the CLEVR-Dialog modules are the following: batch size 128, learning rate 0.0001, and negative log likelihood (NLL) as the loss function. For the MNIST Dialog modules, a batch size of 256, a learning rate of 0.0001, and negative log-likelihood (NLL) as the loss function were used. In total, 28 modules for CLEVR-Dialog and 26 modules for MNIST Dialog were

Table 4.3: Overview of the loss and accuracy of the CLEVR-Dialog and MNIST Dialog modules.

CLEVR-Dialog Modules	Loss	Acc.	MNIST Dialog Modules	Loss	Acc.
colour-blue?	0.0011	99.99	colour-blue?	0.0020	99.97
colour-brown?	0.0033	99.95	colour-red?	0.0072	99.93
colour-cyan?	0.0019	99.98	colour-green?	0.00	100.0
colour-grey?	0.0039	99.97	colour-violet?	0.0013	99.98
colour-green?	0.0027	99.97	colour-brown?	0.0060	99.81
colour-purple?	0.0011	99.98	style-stroke?	0.00	100.00
colour-red?	0.0038	99.96	style-flat?	0.00	100.00
colour-yellow?	0.0061	99.95	bgcolour-white?	0.00	100.00
shape-cube?	0.0089	99.88	bgcolour-cyan?	0.00	100.00
shape-cylinder?	0.0093	99.80	bgcolour-salmon?	0.00	100.00
shape-sphere?	0.0072	99.90	bgcolour-yellow?	0.00	100.00
size-small?	0.0079	99.92	bgcolour-silver?	0.00	100.00
size-large?	0.0101	99.91	number-0?	0.0130	99.95
material-metal?	0.0089	99.90	number-1?	0.0049	99.85
material-rubber?	0.0084	99.91	number-2?	0.0040	99.95
relate-behind?	0.0050	99.91	number-3?	0.0009	99.98
relate-left?	0.0014	99.97	number-4?	0.0080	99.90
relate-right?	0.0021	99.97	number-5?	0.0101	99.66
relate-front?	0.0034	99.92	number-6?	0.0019	99.92
immediate-relate-behind?	0.0091	99.82	number-7?	0.0075	99.88
immediate-relate-left?	0.0037	99.89	number-8?	0.0016	99.97
immediate-relate-right?	0.0034	99.91	number-9?	0.0037	99.94
immediate-relate-front?	0.0068	99.84	immediate-relate-above?	0.00	100.0
extreme-relate-behind?	0.0114	99.78	immediate-relate-below?	0.00	100.0
extreme-relate-left?	0.0029	99.93	immediate-relate-left?	0.00	100.0
extreme-relate-right?	0.0047	99.95	immediate-relate-right?	0.00	100.0
extreme-relate-front?	0.0009	99.98			
extreme-relate-middle?	0.0835	97.59			

trained. An overview of the loss and the accuracy on the validation set is given in table 4.3.

For training the modules, the HPC infrastructure provided by the Vlaams Supercomputer Center (VSC) with modern CPU (Intel Xeon) and GPU (Nvidia Tesla P100, Nvidia A100, Nvidia Volta V100) platforms was used.

4.4.4 Visual dialogue grammar

In order to map the utterances from the datasets to their meaning representation, a computational construction grammar-based approach is used. This approach is responsible for mapping the utterances onto their meaning representation composed of the primitive operations described in the previous section. The grammar is based on the grammar described in Nevens et al. (2019a), extending it with constructions that are able to handle co-references. For example, co-references can either be signalled through a lexical item such as

"*it*", "*aforementioned*" or "*previous*". In other cases, the co-reference is indicated with "*the*". This leads to constructions such as the IT-CXN, which is a mapping between the form "*it*" and the meaning GET-TOPIC, UNIQUE and FIND-IN-CONTEXT and the THE-ANAPHORIC-NP-CXN maps between the form "*the*" followed by a noun phrase and the meaning consisting of the primitives GET-TOPIC, UNIQUE and FIND-IN-CONTEXT. Another extension that needed to be made is handling the declarative sentences. Indeed, the dialogues contain captions that are statements about the objects in the image. In contrast to the questions, the meaning of these captions does not contain an open variable that leads to the answer. The meaning representations of captions are straightforwardly modelled through declarative constructions that add a bind statement to the meaning network. For example, the meaning of the caption "*There is a red object.*" contains a YES that is bound to the EXIST primitive. The grammar is operationalised through the Fluid Construction Grammar framework introduced in Chapter 3. The grammar covers all the utterances from the CLEVR-Dialog dataset and the MNIST Dialog dataset. It consists in total of 268 constructions that are either morphological, lexical or grammatical. Morphological and lexical constructions cover the lexical items such as the colours, sizes, materials, spatial relationships, shapes etc. The other constructions capture the more grammatical structures such as nominals, noun phrases, prepositional phrases, interrogative and declarative structures etc. The grammar is validated through executing the resulting meaning representation and checking whether the computed answer is correct (see Section 4.5.1 and 4.5.2), thereby achieving 100% accuracy in the symbolic evaluation.

As discussed in Chapter 3, language processing in FCG is a search process in which the constructions operate over the transient structure. This process starts from the initial transient structure. Then, the preconditions of the constructions in the grammar are checked against the transient structure. If the preconditions of a certain construction are met, then this construction applies, resulting in a new transient structure. Then, the goal tests are checked, if no solution is found, the search process continues.

Next, I demonstrate the grammar in comprehension using an example utterance of the CLEVR-Dialog dataset: "*What material is the aforementioned red object?*". The meaning representation that underlies this question is shown in Figure 4.7.

An example of the application of a construction

Comprehending the sentence "*What material is the aforementioned red object?*" consists in building a meaning representation for this sentence. The search

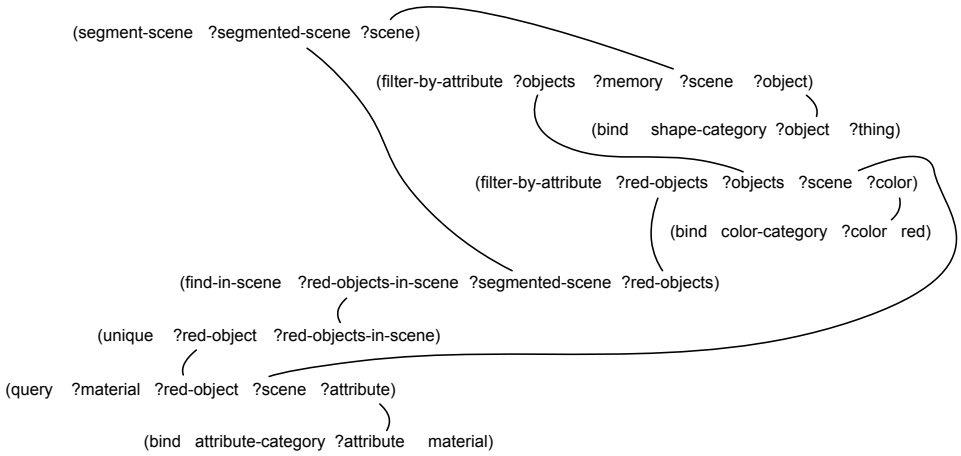


Figure 4.7: Meaning network of the question 'What material is the aforementioned red object?', consisting of a SEGMENT-SCENE primitive, followed by two FILTER-BY-ATTRIBUTE operations, followed by a FIND-IN-SCENE, a UNIQUE operation and lastly a QUERY operation.

process starts with obtaining the initial transient structure through the de-rendering process. The standard FCG de-rendering during comprehension splits the utterance into tokens with an identifier. The initial transient structure after de-rendering the utterance "*What material is the aforementioned red object?*" is shown in Figure 4.8. Next, the pre-conditions of the constructions are checked against this transient structure. If the pre-conditions of a construction match, the post-conditions are merged into the transient structure. The pre-conditions during comprehension of the MATERIAL-CXN check whether there is a string predicate (STRING ?MATERIAL-UNIT "MATERIAL") in the root of the transient structure. In this case, the string predicate can be found indicated in blue). The pre-conditions are thus satisfied and the postconditions (in this case, the meaning, the ARGS, SEM-CAT and SYN-CAT features, are merged into the transient structure (indicated in green). Figure 4.8 shows the initial transient structure and the transient structure, which is the result of the application of the MATERIAL-CXN. First, the features in the comprehension lock of the conditional part are matched against the initial transient structure. Then, the features of the formulation lock and the contributing part are merged into the ?MATERIAL-UNIT, resulting in the new transient structure shown in figure 4.8. This process of matching and merging continues until a solution is found.

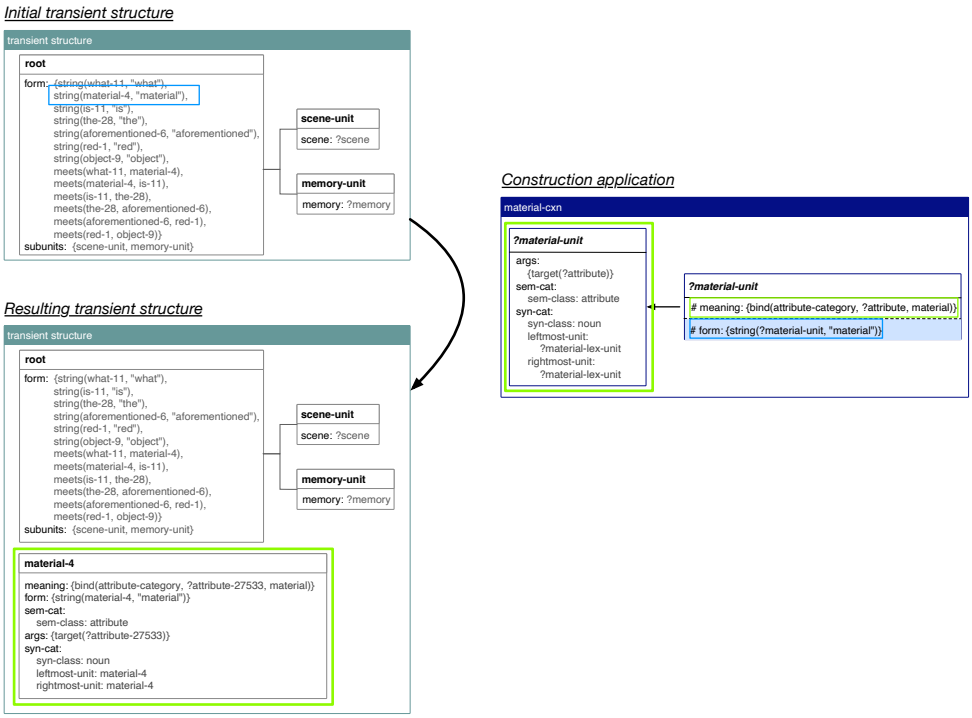


Figure 4.8: Construction application of the MATERIAL-CXN. Matching operations are indicated with a blue box, merging operations in green.

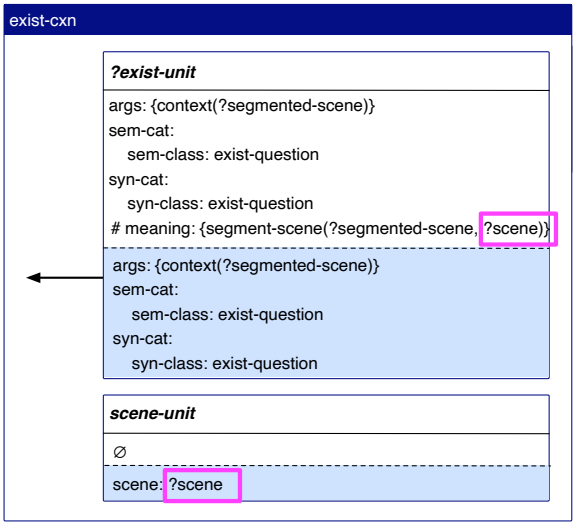


Figure 4.9: Schematic representation of the exist-cxn construction, in which the '?scene' variable is taken from the root (highlighted in pink).

Accessing the scene and memory from the root

In order to execute the meaning network, the current scene and memory should be included in the meaning network. The primitive operations SEGMENT-SCENE, FILTER, RELATE, EXTREME-RELATE, IMMEDIATE-RELATE take a scene-pathname that points to the current scene as one of their arguments. The GET-TOPIC, GET-PREVIOUS-TOPIC and in some cases the FILTER operation take as input the current memory. In these cases, constructions take variables directly from the root. For example, in the exist-construction, the meaning contains the primitive operation SEGMENT-SCENE(?SEGMENTED-SCENE, ?SCENE). The variable ?SCENE is found in the scene-unit in the root of the transient structure.

After the language processing, when the meaning network is passed to IRL, two bind statements are added to the meaning network. A bind statement BIND(PATHNAME-ENTITY, ?SCENE, PATHNAME-ENTITY-1) is added, where pathname-entity-1, which refers to the pathname of the image that is currently under consideration, is bound to the ?SCENE variable. Also a bind statement BIND(WORLD-MODEL, ?MEMORY, WORLD-MODEL-1) is added, where WORLD-MODEL-1 is the conversation memory at that point in the dialogue, which is then bound to the ?MEMORY variable. This way, the primitive operations have access to the current image and memory.

4.4.5 Extending the conversation memory

The conversation memory is extended with new information after each dialogue turn. Concretely, after each turn, a new turn representation is created for the current timestep (see the four boxes in Figure 4.3). The timestep, utterance and reply slots of the turn representation are straightforwardly filled based on the available information. The sentence type is inferred from the final primitive operation executed during the evaluation of the semantic network for the current utterance. The topic corresponds to the set of objects that was bound to the input argument of the same primitive operation call. Finally, entities are added or updated based on the properties of the objects that were mentioned during the current turn.

For example, in the first turn of Figure 4.3, the question "*Are there any triangles?*" is asked and the response is "*Yes*". It can be inferred that the question is of type QUESTION-EXIST based on the fact that the semantic network representing the meaning of the question ends with the EXIST primitive (the semantic network is not shown in the figure). The topic corresponds to the set containing the only triangle that was present in the input image, and which served as the input set to be processed by the EXIST primitive. A representation of this object is added to the list of entities with its mentioned TRIANGLE property and an attention that grounds the object in the image. In the third turn, the question "*Is there an object to its left?*" of type QUESTION-EXIST is asked and the answer "*Yes*" is returned. The topic now shifts to the set containing the only object that was to the left of the previous topic, as this was the input to the EXIST primitive. No information apart from its grounding in the world is added to the entity representation, as no additional information was mentioned. In the final turn, the question "*What is its colour?*" of type QUESTION-QUERY is asked and the answer *Red* is given. The property COLOUR: RED is added to the representation of the topic entity. The topic does not shift, as it was again the same object that was the input to the final QUERY primitive.

4.5 Experiments

I now validate the novel methodology using two standard benchmark challenges in the field of visual dialogue, in particular MNIST Dialog (Seo et al., 2017) and CLEVR-Dialog (Kottur et al., 2019). Both benchmarks were explicitly designed to be bias-free and to include a large proportion of non-trivial co-references across dialogue turns. Due to these two characteristics, answering the questions in

Q_1: Are there brown digits?
A_1: **Five**

Q_2: Are there digits in a yellow background among them?
A_2: **One**

Q_3: What is the number of the digit above it?
A_3: **1**

Q_4: What is the background color of the 1?
A_4: **White**

Q_5: What is the color of the digit below it?
A_5: **Brown**

Q_6: Are there more brown digits?
A_6: **Four**

Q_7: Are there 0's among them?
A_7: **One**

Q_8: What is the style of it?
A_8: **Stroke**

Q_9: What is the color of the digit below it?
A_9: **Red**

Q_{10}: What is the number of the digit above it?
A_{10}: **0**

Figure 4.10: Example dialogue from the MNIST Dialog dataset.

the datasets cannot be done based on any statistical properties of the scenes, questions and answers, but requires actual reasoning about both the visual content and the discourse context.

4.5.1 MNIST Dialog

Data

The MNIST Dialog dataset consists of 50,000 images, which are each accompanied by three dialogues. Each dialogue is in turn composed of 10 question-answer pairs. Each image consists in a synthetically generated 4x4 grid of hand-drawn digits with four randomly sampled attributes: colour (RED, GREEN, BLUE, PURPLE or BROWN), background colour (CYAN, YELLOW, WHITE, SILVER or SALMON), number (from 0 to 9) and style (FLAT or STROKE). A symbolic description of the scene is also provided as meta-data, but is not part of the actual benchmark. The questions and answers are automatically generated. The questions can either query attributes of a single digit (e.g. "*What is the colour of the digit below it?*") or count digits based on one or more of their attributes ("*Are there brown digits?*"[2]). They can also include references to the spatial relations between the digits. The answers always take the form of a single word. An example dialogue from the MNIST Dialog dataset is shown in Figure 4.10. Seo et al. (2017) estimate that 94% of the questions involve co-references, either in the form of pronouns or in the form of definite noun phrases.

Operationalisation and experimental set-up

There are three main challenges involved in the operationalisation of the proposed methodology for the MNIST Dialog benchmark. First of all, it is necessary

[2]Somewhat counterintuitively, the answer to this question is a number and not a boolean value.

to find a way to map the MNIST questions to semantic networks that are composed of the primitive operations that are introduced in Section 4.4.2. This is a highly non-trivial task, as the MNIST dataset does not come with any semantic annotation of the questions. Second, the neural modules that are used by these primitive operations need to be trained on the MNIST dataset images. Finally, it is necessary to be able to evaluate the process of mapping from questions to semantic networks, the execution of these networks, and the neural modules themselves independently from each other.

In order to operationalise the process of mapping from the MNIST questions to their semantic representations, a computational construction grammar-based approach (Van Eecke and Beuls, 2018; Beuls et al., 2021; van Trijp et al., 2022) was adopted. Concretely, I extended the computational construction grammar developed by Nevens et al. (2019a) for the CLEVR VQA dataset (Johnson et al., 2017a) so that it is able to handle constructions involving co-referential expressions. The meaning predicates contributed by these additional constructions are expressed in terms of the primitive operations defined above. Other approaches for mapping from natural language utterances to semantic networks, such as LSTM-based techniques, have also been proposed in the literature (see Section 4.3), but require a gold standard annotation of the semantic networks in the dataset. The grammar that is used in this approach is discussed in Section 4.4.4. The execution of the semantic networks is modelled using the Incremental Recruitment Language (IRL) framework (Van den Broeck, 2008; Spranger et al., 2012; Nevens et al., 2019b), a procedural semantics implementation.

In order to verify the aptness of the semantic representations resulting from the language processing process, I have in a first phase made symbolic implementations of the primitive operations that work on the noise-free metadata that describe the images rather than on the images themselves. By doing this, it was possible to verify whether the predicted semantic networks would in theory always lead to the correct answer given a question and a scene. Indeed, the semantic networks achieved a 100% accuracy when applied to the metadata of the images. This proves on the one hand that the primitive operations presented in Section 4.4.2 are indeed sufficient to represent the meaning of the questions in the dataset, and on the other hand that the grammar covers the dataset completely. It is obviously the temporary noise-free condition of the synthetic dataset that makes the 100% figure possible.

The neural modules underlying the primitives described in Section 4.4.2 were then trained on the training section of the MNIST dataset and their accuracy was

evaluated on the validation set. All individual primitive operations achieved an accuracy of over 99.80% on the image data. The details of the training process and the evaluation results of the individual neural primitives were described in 4.4.3.

An operational example of the methodology as applied to a question and scene from the MNIST Dialog dataset is shown in Figure 4.11. The figure shows the execution of the semantic network corresponding to the question "*What is its colour?*". This question is asked as the second turn in a dialogue, following the question-answer pair "*How many 3's are there? One.*". The semantic representation is composed of five primitive operations: segmenting the image (SEGMENT-SCENE), retrieving the topic of the conversation from the conversation memory (GET-TOPIC), retrieving the topic in the scene (FIND-IN-SCENE), checking whether the retrieved topic corresponds to a single object (UNIQUE) and querying the colour of this object (QUERY). When it comes to the execution of this network, the GET-TOPIC primitive extracts the topic from the last turn of the conversation memory and binds the retrieved topic to the variable ?TARGET-TOPIC. The SEGMENT-SCENE primitive binds a segmentation of the entire scene to the ?SEGMENTED-SCENE variable. The FIND-IN-SCENE primitive uses the bindings of ?TARGET-TOPIC and ?SEGMENTED-SCENE to compute the topic of the previous turn in the current scene. The resulting attention, in this case highlighting a single cell in the second row on the third column, is bound to the variable ?TOPIC-IN-SCENE. The UNIQUE primitive operation checks whether there is a single attention in the set bound to ?TOPIC-IN-SCENE and binds the attention to the variable ?TARGET-OBJECT. Finally, the QUERY primitive queries the colour attribute of the target object and binds the answer GREEN to the *?answer* variable. In terms of the classification of primitives introduced in Section 4.4.2, the SEGMENT-SCENE and QUERY operations have a subsymbolic implementation, whereas the UNIQUE, GET-TOPIC and FIND-IN-SCENE operations have a symbolic implementation. It is the FIND-IN-SCENE operation that bridges between the symbolic and subsymbolic domains.

When it comes to evaluating the performance of the overall system on the test portion of the MNIST Dialog benchmark dataset, two different experimental set-ups are included. First of all, in the '*standard*' setting, the accuracy of the answers provided by the execution of the semantic networks that result from language processing is evaluated. In the '*guessing*' setting, the system is allowed to make an educated guess when the execution of a semantic network fails and therefore does not lead to an answer. The educated guess is made based on the question type as identified by the grammar and the distribution of answers per

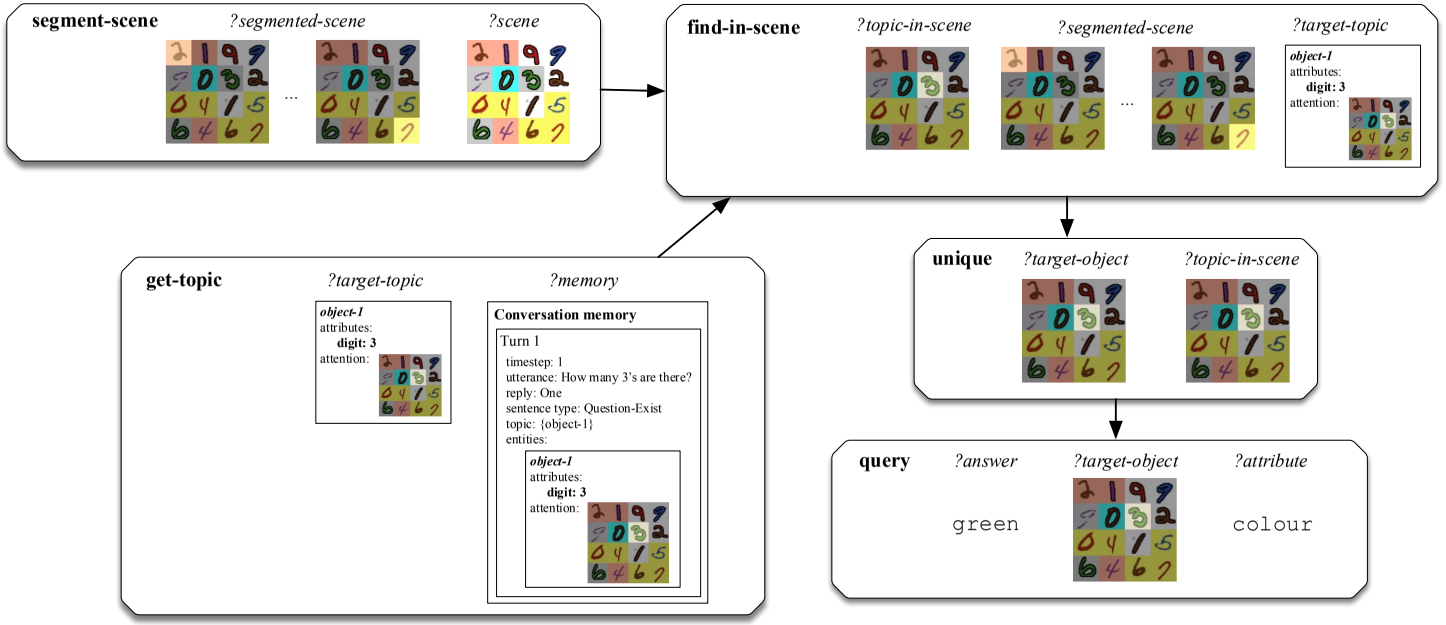


Figure 4.11: Schematic representation of the execution of the semantic representation for the utterance '*What is its colour?*' following the utterance '*How many 3's are there?*' on a scene from the MNIST Dialog dataset.

question type in the training set. For example, if the question "*What is the colour of the 6?*" is asked and the conversation memory does not contain a reference to any 6s, for example due to a previous classification error, the execution of the semantic network fails and a guess is made based on the distribution of colours as answers in the training data. The '*guessing*' setting is provided in order to be able to straightforwardly compare the results of the methodology to neural approaches which always provide an answer even if its probability is low. The experimental results obtained on the MNIST Dialog dataset are provided in Table 4.4 and will be discussed in Section 4.6.

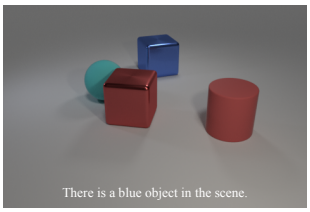
4.5.2 CLEVR-Dialog

Data

The CLEVR-Dialog dataset consists of 85,000 images, which are each accompanied by five dialogues. Each dialogue starts with a caption that makes a statement about the contents of the image (e.g. "*There is a grey object right of a large object*"). The caption is then followed by 10 question-answer pairs. The images depict synthetically generated scenes consisting of 3D geometrical objects with randomly sampled attributes: shape (CUBE, SPHERE or CYLINDER), size (SMALL or LARGE), colour (GREEN, RED, GREY, BLUE, BROWN, YELLOW, PURPLE or CYAN) and material (RUBBER or METAL). The questions involve querying an attribute of an object in the scene (e.g. "*What shape is it?*"), counting objects based on one or more of their attributes (e.g. "*How many green spheres are there?*"), and querying whether a set of objects satisfies a given description (e.g. "*Are there any green spheres?*"). The questions can involve reference to different kinds of spatial relations between objects (e.g. "*the left block*" and "*the block left of the green cylinder*"). In contrast to MNIST Dialog questions, anaphora in CLEVR-Dialog questions can refer to entities mentioned in any of the previous dialogue turns. Moreover, resolving history-dependent questions can require taking into account the entire dialogue history, as is for example the case in questions such as "*How many other objects are present in the image?*". An example dialogue from the CLEVR-Dialog dataset is shown in Figure 4.12.

Operationalisation and Experimental Set-up

The challenges involved in operationalising the introduced methodology for the CLEVR-Dialog benchmark are the same as those discussed above in the context of the MNIST Dialog benchmark: (i) mapping the CLEVR-Dialog questions to semantic networks that are composed of the primitive operations introduced



Q_1: If there is an object left of it, what size is it?
A_1: **Large**

Q_2: How about material?
A_2: **Metal**

Q_3: What about shape?
A_3: **Cube**

Q_4: And that of the above blue object?
A_4: **Cube**

Q_5: How many objects does that metallic object have to its behind?
A_5: **Two**

Q_6: What number of other objects share similar shape with the aforementioned big object?
A_6: **One**

Q_7: If there is an object to the right of the earlier big object, what material is it?
A_7: **Metal**

Q_8: Does it have objects to behind itself in the scene?
A_8: **No**

Q_9: What color is the above large object?
A_9: **Red**

Q_{10}: What number of other objects in the picture are of same shape with the previous blue object?
A_{10}: **One**

Figure 4.12: Example dialogue from the CLEVR-Dialog dataset.

in Section 4.4.2, (ii) training the neural modules underlying these operations on the CLEVR-Dialog images, and (iii) evaluating the accuracy of the language processing system and the neural modules.

In order to map from utterances to procedural semantic networks, the exact same construction grammar as the one used for the MNIST Dialog benchmark is used. In order to verify the aptness of the programs and language processing system, a temporary symbolic implementation of the primitives was created to evaluate the programs that resulted from language processing on the noise-free metadata that describe the images in the dataset. An accuracy of 99.99% was achieved. After an exhaustive error analysis, I could conclude that the non-perfect accuracy was due to scenes that contained an even number of objects and in which a question relied on reference to the object 'in the middle'[3]. As the dataset was constructed in such a way that these questions are impossible to answer reliably, even for a human, I concluded that the primitives are sufficient to solve the task of CLEVR-Dialog, and that the grammar achieves maximum coverage on the CLEVR-Dialog questions.

The neural modules underlying the primitive operations described in Section 4.4.2 were trained on the training portion of the CLEVR-Dialog dataset and their accuracy was evaluated on a held-out validation set of 10,000 images. All modules except the 'extreme-relate-middle?' module achieved an accuracy of over 99.7%. The lower accuracy of this module (97.59%) is probably due to the previously described problem in which a question can refer to the 'middle' object in a scene with an even number of objects. The details of the training process and the evaluation results of each individual module were described in 4.4.3.

An operational example of the execution of a semantic network underlying a question from the CLEVR-Dialog dataset on an image is shown in Figure 4.13. In

[3]This information was communicated to and acknowledged by the creators of the dataset.

this example, the same question as in Figure 4.11 is asked, namely "*What is its colour?*". However, in this case the question follows the caption "*There is a large sphere.*". Also, the question is now asked about a 3D rendered image rather than about a 2D 4x4 grid. The grammar maps the question to the same underlying procedural semantic program consisting of five primitive operations. However, the implementations of these primitives now make use of the neural modules trained on the CLEVR-Dialog images. The primitive operations are executed and the answer "*cyan*" is returned.

In order to evaluate the performance of the overall system on the test portion of the CLEVR-Dialog benchmark dataset, the two same experimental set-ups as for the MNIST Dialog dataset are used. In particular, I provide the '*standard*' and '*guessing*' settings. The experimental results obtained on the CLEVR-Dialog dataset are provided in Table 4.4 and will be discussed in Section 4.6.

4.6 Results and Discussion

An overview of the evaluation results of the system on the MNIST Dialog and CLEVR-Dialog benchmark datasets is shown at the bottom of Table 4.4. In the best-performing experimental setting, i.e. the '*guessing*' setting, where the system makes an educated guess when the execution of a semantic network fails, this system achieves a question-level accuracy of 99.8% on the MNIST Dialog benchmark and of 99.2% on the more challenging CLEVR-Dialog benchmark. In the '*standard*' setting, i.e. without guessing, it achieves a question-level accuracy of 99.8% and 99.0% respectively. The table also reports on the system's performance on the standard CLEVR VQA benchmark, with a question-level accuracy of 99.7%. CLEVR VQA is not a visual dialogue benchmark, but has been included for reference as it has been a very popular benchmark in the literature.

The table also compares the results against previous approaches, namely the encoder-decoder-based approaches presented by Das et al. (2017) and Seo et al. (2017), the neural module networks-based approaches by Andreas et al. (2016b), Johnson et al. (2017b), Hu et al. (2017), Mascharka et al. (2018) and Kottur et al. (2019), the MAC network-based approaches by Hudson and Manning (2018) and Shah et al. (2020), and the neuro-symbolic scene-graph-based approach by Yi et al. (2018) and Abdessaied et al. (2022). The introduced system outperforms the state-of-art on MNIST Dialog, and obtains near-state-of-the-art performance

[4]The evaluation of the model of the CLEVR dataset is reported by Mao et al. (2019).
[5]The evaluation of the model on the MNIST Dialog dataset is reported by Kottur et al. (2018) and of CLEVR-Dialog by Kottur et al. (2019).

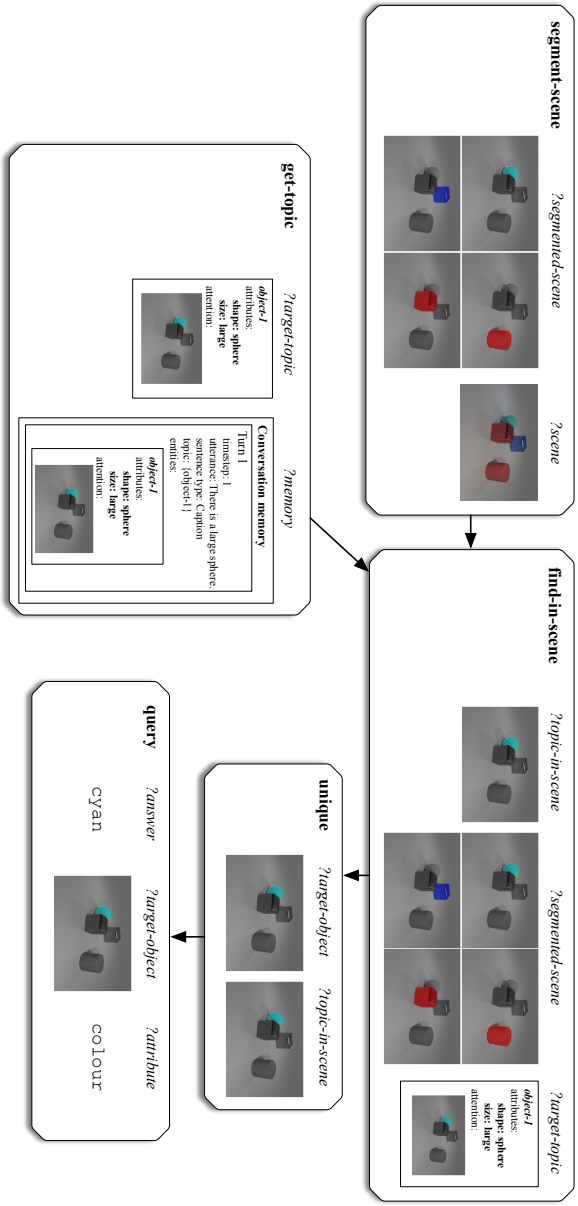


Figure 4.13: Schematic representation of the execution of the semantic representation for the utterance *"What is its colour?"* following the caption *'There is a large sphere.'* on a scene from the CLEVR-Dialog dataset.

Table 4.4: Overview of results for MNIST Dialog, CLEVR-Dialog and CLEVR VQA

	MNIST Dialog	CLEVR-Dialog	CLEVR VQA
Encoder-decoder approaches			
LF (Das et al., 2017)	45.1	55.9	/
HRE (Das et al., 2017)	49.1	63.3	/
MN (Das et al., 2017)	48.5	59.6	/
AMEM (Seo et al., 2017)	96.4	/	/
Neural module networks approaches			
NMN[4](Andreas et al., 2016b)	/	/	72.1
IEP (Johnson et al., 2017b)	/	/	96.9
N2NMN[5](Hu et al., 2017)	23.8	56.6	83.7
TbD (Mascharka et al., 2018)	/	/	99.1
corefNMN (Kottur et al., 2019)	99.3	68.0	/
MAC network approaches			
MAC (Hudson and Manning, 2018)	/	/	98.9
MAC-CQ-CAA-MTM (Shah et al., 2020)	/	98.3	/
Neuro-symbolic approaches			
NS-VQA (Yi et al., 2018)	/	/	**99.8**
NSVD (Abdessaied et al., 2022)	/	**99.7**	/
Ours - Neuro-symbolic procedural semantics			
standard	**99.8**	99.0	99.7
guessing	**99.8**	99.2	99.7

on CLEVR-Dialog and CLEVR VQA. While other approaches that tackle both visual dialogue benchmark challenges typically perform much better on the easier MNIST Dialog benchmark as compared to more challenging CLEVR-Dialog benchmark, our novel approach obtains consistently good results across both datasets.

While the reported benchmark accuracies are definitely important to validate the methodology in comparison to existing approaches, the more prominent contribution of the methodology that I present lies in four main characteristics that distinguish it from the state of the art in visual dialogue. First of all, the methodology is explainable in human-interpretable terms. Input utterances are mapped onto procedural semantic representations, which correspond to logic programs. These programs, which reveal the logical structure underlying an input utterance, are composed of human-interpretable primitive operations, such as COUNT, QUERY and FILTER. This means that the result of the initial language processing step can be inspected and understood by the user. The conversation memory of the system also stores information about the history of a dialogue in a structured and human-interpretable way, thereby being fully transparent about what is remembered by the system. The input and output of each primitive operation can be traced and interpreted, as they consist in either meaningful symbols (human-interpretable categories) or visual attentions over images. Given that these visual attentions are the input and output of human-interpretable operations, humans are able to judge whether an attention corresponds to what is expected or not. As the symbolically implemented primitives can be traced on a meaningful level, the only aspect of the system where the interpretability of the computation is limited is situated in the subsymbolic primitives that deal with perception on the lowest level. By pushing the neuro-symbolic boundary so far down, it is ensured that any reasoning capabilities that exceed the perception of basic categories is explainable in human-interpretable terms.

A related advantage of this approach is that it avoids inconsistencies in reasoning by implementing its subsymbolic primitive operations on top of a shared inventory of highly-specialised neural modules. Keeping consistency across reasoning operations is a highly desirable property of intelligent systems, which at the same time leads to a more human-like behaviour. For example, it is obvious that the human capabilities of recognising objects and counting objects rely on the same conceptual distinctions. This is reflected in the system by implementing the COUNT primitive in terms of computing the cardinality of a set of objects returned by a FILTER operation, which is itself implemented based on the same set of binary classifiers as the QUERY operation. The answer to the question "*How*

many red blocks are there?" is as a consequence guaranteed to be consistent with the answers to the question "*What is the colour of the block?*" asked for each block in the scene.

A third asset of the approach is that it can effectively monitor its own performance. This has become a topic of high interest in the AI community, since deep neural networks often provide confidence scores of poor quality, especially when it comes to out-of-distribution data (Nguyen et al., 2015; Goodfellow et al., 2015). Concretely, in this case, the system knows that it has not been able to answer a question based on sound logic reasoning if the execution of a semantic network fails. While it can still make an educated guess in such cases, the system then indicates that the result should be interpreted with extra care. In fact, the execution of a semantic network fails in 65.7% of the CLEVR-Dialog errors (i.e. errors in the 'standard' setting) and in 41.7% of the MNIST Dialog errors (in the 'standard' setting as well). The remaining 34.3% and 58.3% of errors respectively remain undetected by the system. This amounts to only 0.3% of the questions in CLEVR-Dialog and 0.1% of the questions in MNIST Dialog.

A final advantage resides in the modularity of the approach. New primitive operations can be added to the system in order to accommodate new tasks or to model new cognitive capabilities acquired by an artificial agent. These new primitives can add to both the logical and perceptive reasoning capabilities of the agent. Where appropriate, they can reuse neural modules used by existing primitives without needing to retrain them. Neural modules can also dynamically be added, but these might affect the performance of other modules and therefore require retraining some of them. For example, adding a binary classifier for a new colour will likely affect the performance of existing binary classifiers for other colours, as these were trained in the absence of the new colour category.

Figure 4.14 and 4.15 illustrate the interpretability of the introduced approach by providing two examples of questions from the CLEVR-Dialog dataset that were wrongly answered. Concretely, these examples show how the system supports the tracking of the source of errors by providing insight into the logical structure underlying the question, and into the input and output of the different primitive operations that were performed. Figure 4.14 shows the execution of the semantic network underlying the utterance '*How many brown objects are there?*' on a given CLEVR scene. The question has been analysed into three primitive operations: segmenting the scene (SEGMENT-SCENE), filtering the segmented scene for the colour brown (FILTER) and counting the number of the

resulting set of brown objects (COUNT). The result of the counting operation, which is at the same time returned as the answer to the question, is TWO. However, this answer does not match the gold standard answer from the dataset, which is ONE. In fact, when scrutinising the execution trace of the semantic network on the scene, it becomes clear that the filter operation has retrieved two brown objects. After a visual inspection of the attentions, the human observer can see that the leftmost object in the scene was wrongly classified as being brown and the source of the error has been found. If we would now query the colour of the leftmost object in the scene, the system is also guaranteed to answer BROWN, as the FILTER and QUERY primitives internally rely on the same neural classifiers. Thus, while the answer to the question is wrong, it is logically consistent with the overall perception and reasoning skills of the system.

Figure 4.15 traces back the source of the erroneous answer THREE to the question "*How many other objects are there?*". The question is analysed into four primitive operations: segmenting the scene (SEGMENT-SCENE), filtering the conversation memory for objects (FILTER), taking the set difference between the objects in the segmented scene and those retrieved from the conversation memory, and counting the resulting set (COUNT). In this case, the conversation memory spans two turns in which only a single object has been mentioned. Indeed, the scene contains three objects apart from the one that has been mentioned already. All aspects of the construction and execution of the semantic network seem to be flawless, but the answer THREE does not match the gold standard answer TWO. This tells us that the problem does not occur while processing the current dialogue turn, but that it must stem from an error in processing a previous dialogue turn that had as a consequence that a second mentioned object was not recognised and therefore does not appear in the conversation memory. The user can then continue analysing the previous turns to retrieve the original source of the problem.

4.7 Conclusion

This chapter has introduced a novel methodology to solve visual dialogue tasks, based on the use of neuro-symbolic procedural semantic representations. Concretely, this methodology encompasses (i) the use of a conversation memory as a data structure that explicitly and incrementally represents the information that is expressed during the subsequent turns of a dialogue, and (ii) the representation of natural language expressions as neuro-symbolic semantic networks that are grounded in both visual input and the conversation memory. These

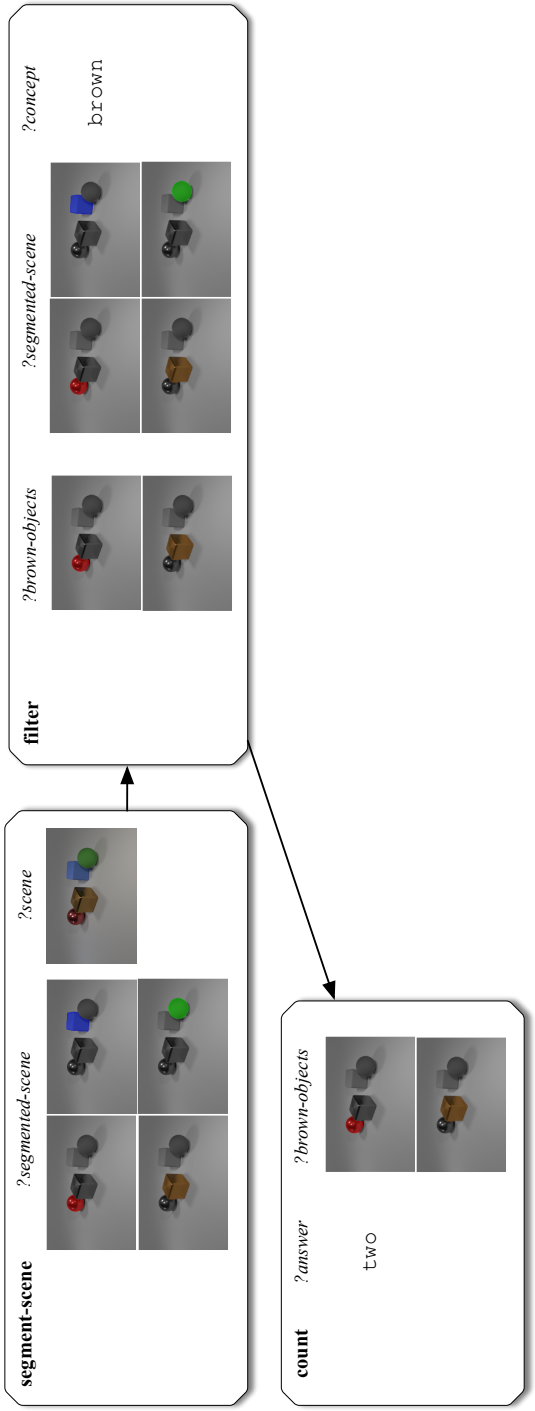


Figure 4.14: Schematic representation of the execution of the semantic network underlying the utterance *"How many brown objects are there?"* on a scene from the CLEVR-Dialog dataset, illustrating the transparency of the approach. The filter operation wrongly recognises the leftmost object to be brown. As a consequence, two brown objects are counted instead of one.

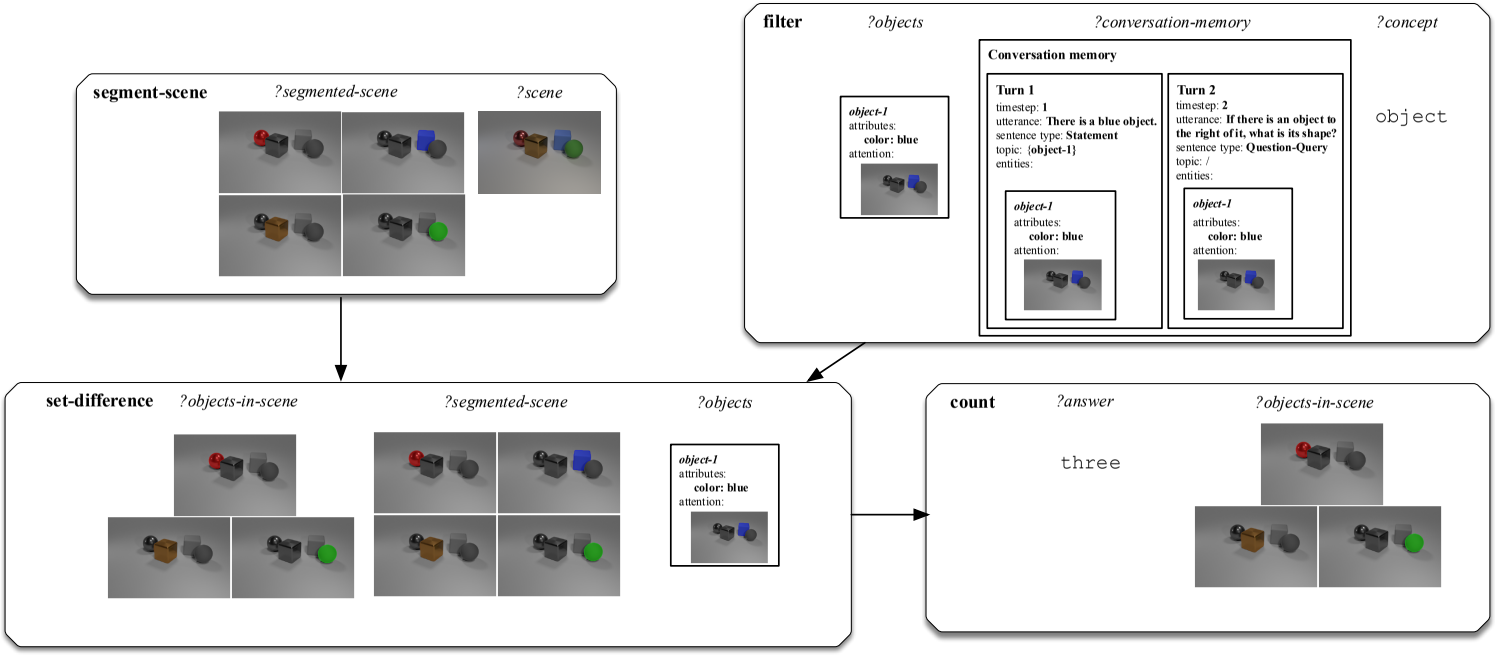


Figure 4.15: Schematic representation of the execution of the semantic network underlying the utterance "*How many other objects are there?*" on a scene from the CLEVR-Dialog dataset, illustrating the transparency of the approach. The figure shows that the semantic network and its execution are flawless. As a consequence, the erroneous answer THREE must be due to an error in the conversation memory introduced in a previous dialogue turn.

networks are composed of a combination of subsymbolic primitive operations that model the perceptual capacities of an agent and symbolic primitive operations that model its reasoning capabilities. The evaluation of the methodology on the MNIST Dialog and CLEVR-Dialog benchmarks shows that the system achieves competitive results with a question-level accuracy of 99.8% and 99.2% respectively.

The methodology presents four main advantages with respect to the state of the art in visual dialogue, which is dominated by attention-based neural network approaches. First of all, the methodology is to a great extent explainable in human-interpretable terms. The semantic networks that represent the meaning of natural language utterances are composed of human-interpretable primitive operations, their input and output arguments are either meaningful symbols or interpretable visual attentions, and the conversation memory represents information conveyed in earlier dialogue turns using a transparent symbolic data structure. This enables the human observer to verify whether an answer returned by the system is indeed the result of sound logic reasoning, as well as to trace back the exact source of any perception or reasoning errors that might occur. Second, the methodology avoids potential reasoning inconsistencies by implementing the primitive operations on top of a shared inventory of highly-specialised neural modules. This ensures at least that the results of different primitive operations are guaranteed to be consistent with each other, whether the neural modules have made correct predictions or not. Third, the system can effectively monitor its own performance, as errors that result from language processing or from the execution of individual primitive operations lead in many cases to an automatically detectable failure in the execution of a semantic network. Finally, the modularity of the approach ensures that new primitive operations can be dynamically added in order to accommodate new tasks or in order to model new cognitive capacities acquired by an agent. These new primitive operations can thereby build further on existing primitive operations or neural modules where appropriate.

Finally, the research contributes to the growing body of research in artificial intelligence that tackles tasks that involve both low-level perception and high-level reasoning using a combination of neural and symbolic techniques. Neural techniques are used to deal with low-level perception tasks and thereby give rise to meaningful symbols that can then be used as a basis for higher-level reasoning operations. It thereby bears the promise of leading to the development of artificial agents with more explainable, consistent and human-like cognitive capacities.

Chapter 5

Case study: Procedural Semantics for Frame-based Narrative Construction

5.1 Introduction

This chapter is based on Van Eecke et al. (2023a). I contributed to the implementation of the Candide model and mapping out the conceptual foundations.

In the previous chapter I introduced a procedural semantics that is adequate for reasoning over visual input and discourse. I showed that state-of-the-art

results on two visual dialogue tasks can be achieved using this representation. Next, I will show how procedural semantics can be used for reasoning with regard to previously acquired knowledge. In contrast to the previous chapter, I will not focus on tackling benchmark tasks, but I will introduce the conceptual foundations that underlie this idea as well as discuss a proof-of-concept implementation. Concretely, I will present the *Candide* model, a model for human-like, narrative-based language understanding. The model starts from the idea that narratives emerge when individuals are confronted with an utterance. They will interpret this utterance by reasoning over their personal knowledge. This interpretation process is personal since each agent has its own personal knowledge base. Therefore, agents can come to different conclusions when confronted with the same utterance. The chain of reasoning operations that the agent goes through to draw a conclusion, is what I will call the *narrative* of an agent. This narrative is interpretable, making the agent able to explain its reasoning process. Similar to humans, this methodology makes agents able to (i) reason over personal knowledge, thereby constructing a narrative and (ii) explain this reasoning process. The research presented here is a step towards building systems for language understanding that are personal, interpretable and human-like.

In this chapter, I will first focus on the background. Next, I will discuss the overall architecture of the Candide model and its technical operationalisation. Lastly, I will discuss the contribution of the model and the avenues of future research.

5.2 Narrative-based language understanding

Recently, neural machine learning techniques have taken over the field of NLP, achieving impressive results on several tasks such as machine translation, speech recognition, text summarisation, semantic role labelling and sentiment analysis. These models are fit to exploit statistical properties in large amounts of texts by capturing co-occurrences of characters, words and sentence. By only using textual input to train these models and not taking into account meaning or grounding in the world, these models fail at truly understanding (Bender and Koller, 2020). Moreover, this research focusses on building models that perform well on benchmark datasets, instead of building models that truly understand (Bowman and Dahl, 2021), which results in research that focusses on performance instead of modelling understanding.

One of the main limitations of current NLP systems is the lack of capability to model human-like, narrative-based language understanding. Crucially, narrative-

based language understanding uses personal knowledge and beliefs to interpret a linguistic observation. Narratives are thus rooted in the experiences of an individual and different individuals can have different interpretations of the same linguistic observation Steels (2022a) reflecting the nature of human language and cognition. This capacity of divergent interpretations is hard to capture in current NLP systems, since there is not one 'ground truth' interpretation. Moreover, the narratives that emerge are not captured in text, but are construed through an interpretation process over the observation and the personal knowledge and beliefs of an individual agent. It is thus hard to cast this narrative-based language understanding in the annotation schemes required for the machine learning paradigm. Modelling this capacity is a crucial challenge for the computational linguistics community.

The primary objective of this research is to introduce a novel approach to narrative-based language understanding that starts from the idea that narratives emerge through the process of interpreting novel observations with respect to previously acquired knowledge and beliefs. Concretely, I present a computational model of this interpretation process. The model integrates three main components: (i) a personal dynamic memory that holds a frame-based representation of the knowledge and beliefs of an individual agent, (ii) a construction grammar that maps between linguistic observations and a frame-based representation of their meaning, and (iii) a reasoning engine that performs logic inference over the information stored in the personal dynamic memory.

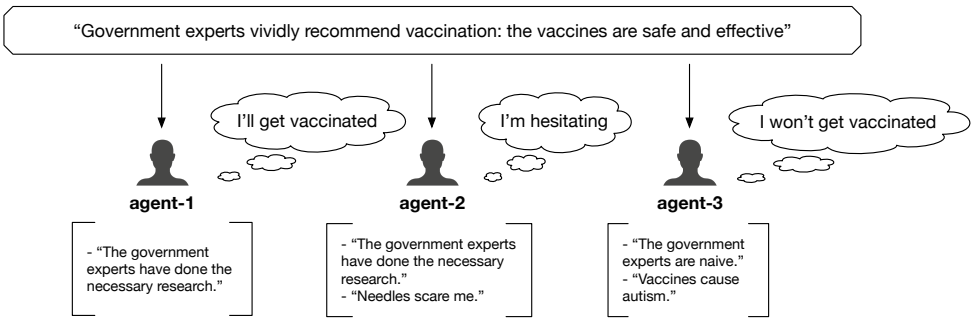


Figure 5.1: Informal sketch of the Candide model. Narrative-based language understanding is conceived as the interpretation process of a linguistic observation with respect to an agent's individual belief system. Three agents observe the same observation that the government experts recommend vaccination. Based on their personal belief system, each agent draws their own personal conclusion and construes their own narrative.

Figure 5.1 shows the conceptual ideas of the Candide model. In this figure,

three agents observe the same utterance "*Government experts vividly recommend vaccination: the vaccines are safe and effective.*" When they are asked whether or not they will get vaccinated, they interpret the utterance with regard to their own personal knowledge and beliefs and each agent comes to a different conclusion. The first agent draws the conclusion that they will get vaccinated, based on their belief that the government experts have done the necessary research. The second agent is hesitating, since they believe that the experts have done their research, but they are also scared of needles. The last agent refuses to get vaccinated. They believe that the experts are naive and that vaccines can cause diseases. Each of the agents construes their own personal narrative that justifies their conclusion. These narratives are personal and the example clearly shows that different agents construe different narratives and come to different conclusions, even when confronted with the same observation.

Personal, dynamic and interpretable models of narrative-based language understanding are of great interest to the fields of computational linguistics and artificial intelligence alike. To the field of computational linguistics, they contribute a perspective that emphasises the individual and contextualised nature of linguistic communication, which contrasts with the static and perspective-agnostic models that dominate the field of NLP today. In the field of artificial intelligence, they respond to the growing interest in the development of artificial agents that combine human-like language understanding with interpretable, value-aware and ethics-guided reasoning (see e.g. Steels, 2020; Montes and Sierra, 2022; Abbo and Belpaeme, 2023).

The model highlights properties that are crucial to narrative-based language understanding. First of all, a model of analysis can only be adequate if it captures the personal nature of narratives. Whether or not a conclusion is justified does not depend on its truth or falsehood from an external perspective, but only on whether it is supported by the beliefs held by an agent. Second, narratives are not captured as such in linguistic artefacts. While authors convey messages that are grounded in their belief systems, these messages do not encode the belief systems themselves. The intended meaning underlying a message needs to be reconstructed inferentially based on the belief system of the receiver Grice (1967); Sperber and Wilson (1986). Finally, it is essential that the interpretation process that is modelled is transparent and human-interpretable. The goal is not merely to draw conclusions given linguistic input, but to reveal the background knowledge, beliefs and reasoning processes that underlie the conclusions that are drawn.

In the next section, I will discuss the overall architecture of the model, followed by its technical operationalisation and a number of illustrative examples. Section 5.5 reflects on the contributions of the model.

5.3 The Candide model

The Candide model relies on the idea that narratives are construed through the personal interpretation process over a personal and dynamic memory of an agent. The agents maps the utterance to a meaning representation and then executes the meaning by reasoning over it and its knowledge and beliefs. Each of the following elements is crucial to the model:

The **Personal Dynamic Memory** of an agent is a data structure that stores the knowledge and beliefs of an agent in a logic representation that supports automated reasoning. The knowledge and beliefs are stored in terms of frames and roles. This information can either be a 'fact' (e.g., I am scared of needles.) or a 'rule' (e.g., I only trust the government if the government experts have done the necessary research.). The memory is personal since the knowledge and beliefs are individual. The knowledge and beliefs are not fixed and can be expanded when the agent is encountered with novel observations and experiences. The PDM is thus conceived of as a dynamic entity to which new knowledge and beliefs can be added at any point in time. Reasoning over the PDM is non-monotonic, as updated beliefs can alter conclusions.

The **Belief System** of an agent at a given point in time equals all information that is stored in the agent's PDM at that moment in time. Each entry in the PDM carries a confidence score, which reflects the degree of certainty of the agent with respect to that entry. However, there exists no formal or conceptual distinction between entries based on their epistemological status, avoiding the need to distinguish between 'knowledge', 'facts', 'opinions' and 'beliefs' for example.

A **Conclusion** is a piece of information that logically follows from a reasoning operation over the belief system of an agent. A typical example would be the answer to a question.

Language Comprehension is the process of mapping an utterance to a logic representation of its meaning. The meaning representation is in procedural semantics and consists of predicates in a frame-based representation. Crucially, the predicates that are used in the meaning representation are the same format

as the knowledge and beliefs in the PDM. While language comprehension is primarily concerned with retrieving the information captured in the linguistic input, rather than its integration with respect to the personal dynamic memory, it is heavily intertwined with other aspects of the interpretation process as well. Indeed, the linguistic knowledge needed to support language comprehension is personal and dynamic, and thereby unavoidably constitutes a first layer of individual interpretation.

Reasoning is the engine that performs logic operations that are conducted over the knowledge and beliefs in the PDM of an agent. It is operationalised using the Prolog engine.

A **Narrative** is defined by the chain of reasoning operations to prove the conclusion based on the knowledge and beliefs in the PDM of an agent. This chain of operations is explicit and thus interpretable. In the example of the vaccinations, the narrative of the agents are the arguments that they made to come to the conclusion whether or not they would get vaccinated. For example, agent-1 constructs the narrative that they will get vaccinated since they trust the government experts. Crucially, the narrative that an agent builds is personal, since it is routed on its personal knowledge.

Interpretation is the process of interpreting an utterance and consists of all aspects described above. It comprises both language comprehension and reasoning over the PDM of an agent to draw a conclusion. The interpretation process thus comprises all steps that an agent goes through starting from hearing the observation to drawing a conclusion.

This model for narrative-based language understanding is named after Voltaire's "*Candide ou l'optimisme*" (Voltaire, 1759). It is inspired by one of the main themes of the novel, namely that a character's belief system and history of past experiences shape the way in which they interpret the world in which they live. As such, different characters in the novel represent different philosophical positions and thereby construe different narratives to explain the same situations and events. The main protagonist, Candide, starts out as a young, naive 'blank slate'. Through conversations with the Leibnizian optimist Pangloss and the fatalistic pessimist Martin, and as a result of long travels that make him experience the hardships of the world, Candide gradually develops his own belief system in light of which he ever more wisely interprets the situations and events he witnesses.

Following the main theme of the novel, the aim is not to model a single 'true' interpretation of an observation, but to show that different beliefs can lead to

different interpretations. Moreover, the belief system of an agent is considered to be dynamic, with the interpretations and conclusions of an agent shifting as more experience and knowledge are gathered. In order to formalise these high-level ideas, I introduce the following operational definitions:

5.4 Technical operationalisation

Next, I will discuss how each of the different parts discussed above is operationalised. I will use a first proof-of-concept implementation for this. Two agents encounter the statement "*Sam sent a postcard to Robin*" and then are both asked the same question "*What did Robin receive from Sam?*", to which they both answer differently. The first agent will answer "*a postcard*" while the second agent will answer "*nothing*". These different conclusions are reached because some of the beliefs of the agents differ. Using this example, I will explain how the PDM, the language processing and the reasoning engine are operationalised.

5.4.1 Language comprehension

The language comprehension component is responsible for mapping between linguistic input, in particular utterances, paragraphs and texts, and a formal representation of their underlying meaning. The language comprehension component is operationalised using the Fluid Construction Grammar framework (FCG – `https://fcg-net.org`; Steels, 2004; van Trijp et al., 2022; Beuls and Van Eecke, 2023).

The choice for FCG as the backbone of the language comprehension component of the model is motivated by four main reasons. First of all, in line with its theoretical grounding in usage-based construction grammar, FCG offers a uniform way to represent and process linguistic phenomena, whether or not they can be analysed compositionally (Beuls and Van Eecke, 2023). Second, FCG is compatible with a wide variety of meaning representations (van Trijp et al., 2022), including the frame-semantic representation that will be used to represent the knowledge and beliefs captured in the personal dynamic memory of the agents. Third, FCG's symbolic learning operators are especially designed to facilitate the one-shot learning of constructions given new linguistic observations, thereby maximally reflecting the personal and dynamic nature of an agent's linguistic capacities (Van Eecke, 2018; Nevens et al., 2022; Doumen et al., 2023). Finally, the symbolic data structures and unification-based processing algorithms employed by FCG ensure that the representation of an agent's linguistic knowledge, as

well as its language comprehension, production and learning processes, are transparent and human-interpretable (Van Eecke and Beuls, 2017).

The semantic representation that is chosen captures the meaning underlying linguistic expressions in the form of semantic frames (Fillmore, 1976; Fillmore and Baker, 2001). As such, the meaning of the utterance "*Sam sent Robin a postcard*" could be represented through a SENDING frame, with "*Sam*", "*Robin*" and "*a postcard*" respectively taking up the roles of SENDER, RECIPIENT and THEME. In terms of data structures, I represent instances of semantic frames through two types of predicates: entities and roles. Entity predicates are used to represent referents, i.e. objects, people, events and situations that can be referred to. In this example, *Sam*, *Robin*, *the postcard*, *the sending event* and *the transfer situation* serve as entities. Role predicates are used to represent relations between entities. Each role predicate expresses a relation between a frame role (e.g. SENDER), the frame to which that role is associated (SENDING), the entity that is taking up the role (*Sam*), the entity that represents the frame instance (*the sending event*) and the entity that represents the situation about which the frame is expressed (*the transfer situation*). There exists a subtle yet important distinction between frame instances and situations. A situation is defined in terms of an agent's world model, while a frame instance assumes a linguistically expressed perspective on a situation. In this example, the transfer situation is linguistically expressed as a sending event, while the same situation could also have been expressed as a receiving event (e.g. "*Robin received a postcard from Sam*"). Note that both the frame instance and the situation are reified as entities and can thus be referred to. The entity and role predicates follow the FrameNet conventions (`https://framenet.icsi.berkeley.edu`) and are represented in standard Prolog syntax (ISO/IEC 13211), as exemplified in Listing 5.1. The predicates take the form of ROLE(ROLE, FRAME, ENTITY, FRAMEINSTANCE, SITUATION). The predicate name is ROLE, the first and second arguments are the name of the role and the frame in which the entity playing the role participates respectively. The third argument is the instance of the entity that plays the role in the frame. The last arguments are the entity that represents the instance of the event and the entity that represents the instance of the situation. These last two refer to the instantiation of a specific event or frame.

5.4.2 Personal dynamic memory

The Personal Dynamic Memory stores the knowledge and beliefs of an agent. There is no distinction between knowledge or beliefs, every piece of information of an agent is stored in the PDM. The knowledge and beliefs are stored as

```
% Entity predicates

entity(sam).
entity(robin).
entity(postcard).
entity(sending_event).
entity(transfer_situation).

% Role predicates

role(sender,sending,sam,sending_event,transfer_situation).
role(recipient,sending,robin,sending_event,transfer_situation).
role(theme,sending,postcard,sending_event,transfer_situation).
```

Code fragment 5.1: Frame-semantic representation underlying the utterance "*Sam sent Robin a postcard*" as a combination of entity and role predicates expressed in standard Prolog syntax.

Prolog facts and rules and express relations between entities, roles, frames and situations. Crucially, the information stored in the PDM takes the same form as the meaning provided by the language processing, making it possible to dynamically merge the beliefs provided by the language into the PDM.

For the purposes of this section, I will assume that the agents observe the utterance "*Sam sent Robin a postcard*", comprehend it into the frame-based semantic representation shown in Listing 5.1, and add this representation to their personal dynamic memory. It will be assumed that the agents already hold a number of previously acquired beliefs, in particular about the relation between the semantic frames of SENDING and RECEIVING. As such, they believe that the DONOR role in an instance of the RECEIVING frame, cast over a particular situation, is taken up by the same entity that takes up the SENDER role in an instance of the SENDING frame cast over the same situation. However, this alignment only holds under the condition that the postal services are operational. In other terms, each sending event corresponds to a receiving event if the postal services are operational, and the sender of the sending event corresponds to the donor of the receiving event. At the same time, the agents believe that a similar alignment can be made for the other roles of the SENDING and RECEIVING frames. Moreover, they believe that the postal services are operational if no general strike is taking place. A formal encoding of these beliefs is shown in Listing 5.2. The Listing shows six beliefs regarding the relations between the entities playing roles in the RECEIVING and SENDING frame. Following Prolog syntax, variables are written with a capital letter. In each of the rules, the same variable is used

```
% Belief about the operationality of the mail

mail_operational :- not(general_strike).

% Beliefs about the relation between the sending
% frame and the receiving frame

role(donor,receiving,Entity,_,Situation) :-
    role(sender,sending,Entity,_,Situation),
    !, mail_operational.

role(recipient,receiving,Entity ,_,Situation) :-
    role(recipient,sending,Entity,_,Situation),
    !, mail_operational.

role(theme,receiving,Entity,_,Situation) :-
    role(theme,sending,Entity,_,Situation),
    !, mail_operational.

role(sender,sending,Entity,_,Situation) :-
    role(donor,receiving,Entity,_,Situation),
    !, mail_operational.

role(recipient,sending,Entity,_,Situation) :-
    role(recipient,receiving,Entity,_,Situation),
    !, mail_operational.

role(theme,sending,Entity,_,Situation) :-
    role(theme,receiving,Entity,_,Situation),
    !, mail_operational.
```

Code fragment 5.2: The beliefs of the example agents concerning the operationality of the mail and the conditional alignment between the SENDING and RECEIVING frames.

for `Entity` and `Situation`, meaning that it is the same entity participating in the role of the different related frames and that the frames relate to the same situation. The underscore is used to denote variables which are not further specified. In the rules, the underscore is used to indicate that the frame instance is underspecified. Indeed, the frame instances of the RECEIVING and SENDING frame are different, the situation, however, is the same, which is indicated by the variable `Situation`.

While the agents hold the same beliefs about the relation between the SENDING and RECEIVING frames, as well as the conditions under which the postal services are operational, they hold different beliefs about the current state of social

```
% Belief about the state of social unrest

general_strike :- false.
```

Code fragment 5.3: Agent 1's belief that there is no general strike.

```
% Belief about the state of social unrest

general_strike :- true.
```

Code fragment 5.4: Agent 2's belief that there is a general strike.

```
% Query

?- role(theme,receiving,What,Event,Situation),
   role(recipient,receiving,robin,Event,Situation),
   role(donor,receiving,sam,Event,Situation).

% Answer by Agent 1:

What = postcard,
Situation = transfer_situation.

% Answer by Agent 2:

false.
```

Code fragment 5.5: Frame-semantic representation underlying the question "*What did Robin receive from Sam?*" with two different answers as computed by the Prolog engine based on the PDMs of Agent 1 and Agent 2.

unrest. As such, Agent 1 believes that there is no general strike, while Agent 2 believes that a general strike is going on at the moment. These beliefs are formally encoded in Listing 5.3 and 5.4 respectively.

I define the PDM of Agent 1 to be the combination of the facts and rules specified in Listings 5.1, 5.2 and 5.3, and the PDM of Agent 2 to consist of the facts and rules specified in Listings 5.1, 5.2 and 5.4. The proof-of-concept implementation does not address the issue of modelling the confidence of an agent with respect to its individual beliefs. The most straightforward way to operationalise this in the current proof of concept would be to use probabilistic logic programming, e.g. through ProbLog (De Raedt et al., 2007).

The model does not make any assumptions about the origin of the beliefs captured in the personal dynamic memory of an agent. Beliefs can result from the language comprehension process, from abductive reasoning processes, or could even by designed by a knowledge engineer.

5.4.3 Reasoning and narrative construction

As the beliefs stored in the personal dynamic memory of an agent and the meaning of natural language utterances as comprehended by an agent are both represented as a collection of Prolog facts and rules, logical reasoning can naturally be operationalised through SLD-resolution-based inference. This means that agents can be asked to prove logic formulae that correspond to natural language questions. The conclusion of the proof then constitutes the answer to the question, while the proof itself corresponds to the narrative that explains the reasoning behind it.

Suppose that the two example agents are asked to answer the question "*What did Robin receive from Sam?*". The agents first use their grammar to comprehend this question into its frame-semantic representation, as shown at the top of Listing 5.5. The interrogative nature of the question is reflected by the presence of variables in the semantic representation, denoted by symbols starting with a capital letter. In this case, the interest is in the entity taking up the role of THEME in the receiving event, represented by the variable `What`. The agents are then asked to find a proof for the meaning representation of the question, given the beliefs stored in their respective personal dynamic memories.

Agent 1 reasons that the `transfer_situation` that was previously described (see Listing 5.1) can be viewed as an instance of the RECEIVING frame, given the facts (i) that there is no general strike, (ii) that the mail service is therefore operational,

and (iii) that the `transfer_sitation` is already believed to be an instance of the SENDING frame in which `robin` takes up the role of RECIPIENT and `sam` the role of SENDER. The agent concludes that this reasoning process is (only) valid under the condition that the variables `What` and `Situation` are bound to the values `postcard` and `transfer_situation` respectively. In other terms, Agent 1 comes to the conclusion that Robin received the postcard that was sent to them by Sam.

Agent 2 on the other hand reasons that it knows of no situation that could be viewed as a receiving event in which `sam` and `robin` take up the roles of DONOR and RECIPIENT respectively. Although this agent holds the same beliefs as Agent 1 when it comes to the link between the `sending` and `receiving` frames, Agent 2's belief that a general strike is going on leads to the belief that the postal services are dysfunctional, which in turn leads to the belief that the sending event cast over `transfer_situation` does not correspond to any receiving event. In other terms, Agent 2 beliefs that, while a postcard was sent by Sam to Robin, it was never received at Robin's end because of a general strike that paralysed the postal services.

Figures 5.2 and 5.3 show a schematic overview of the different steps involved in the respective reasoning processes of Agent 1 and Agent 2 when asked to answer the question "*What did Robin receive from Sam?*". The meaning representation of the question is shown in the yellow boxes at the top of the figures and corresponds to a Prolog query. The facts and rules that can be used to prove the query are those stored in the personal dynamic memories of the agents and correspond to those presented in Listings 5.1, 5.2 and 5.3 (Agent 1) and Listings 5.1, 5.2 and 5.4 (Agent 2).

The conjunction of three clauses that constitutes the query can indeed be proven by Agent 1 through a chain of subproofs that establish the link between there not being a general strike, the operationality of the postal services and the alignment of the SENDING and RECEIVING frames. The solid arrows denote the subproofs that were used to prove the top-level query. The labels on the arrows denote the variable bindings that resulted from the subproofs. While the set of bindings that result from proving the top-level query can be considered the conclusion of the reasoning process, it is the chain of subproofs that constitutes the narrative of the agent with respect to this conclusion. The same query cannot be proven by Agent 2, where the proof already fails at the first conjunct. in fact, Agent 2 fails to prove the alignment between instances of the RECEIVING and SENDING frames, as its belief that a general strike is going on leads to a failure to prove

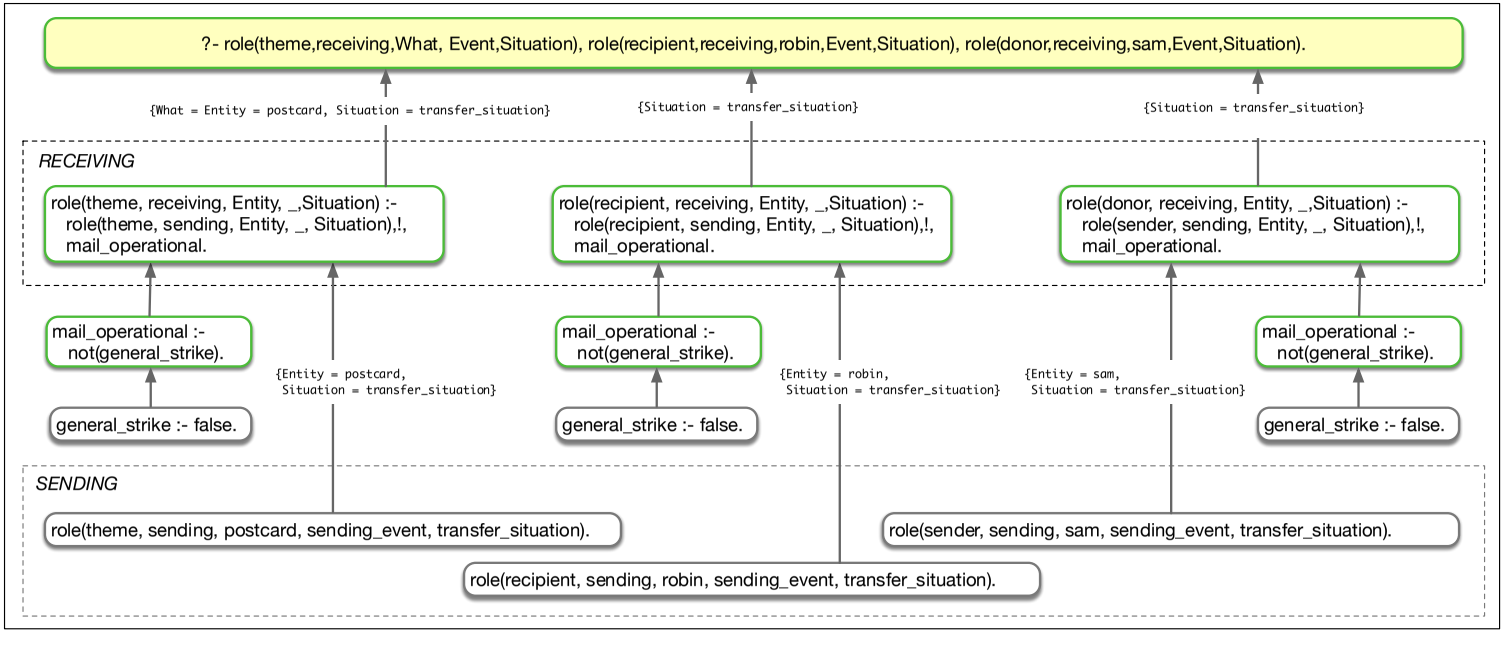


Figure 5.2: Narrative constructed by Agent 1 for responding to the question "*What did Robin receive from Sam?*" based on the frame-semantic information captured in its PDM (cf. Listings 5.1, 5.2 and 5.3).

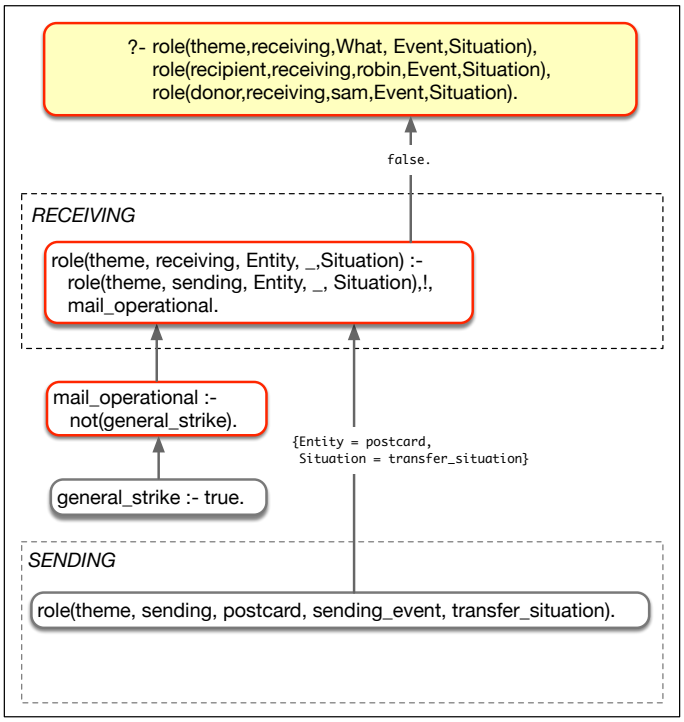


Figure 5.3: Narrative constructed by Agent 2 for responding to the question "*What did Robin receive from Sam?*" based on the frame-semantic information captured in its PDM (cf. Listings 5.1, 5.2 and 5.4).

that the postal services are operational, which is a precondition for the link between the two frames to be established. Note that when a conclusion cannot be proven, the narrative needs to be constructed abductively. Indeed, it consists here in finding a minimal explanation for why a conclusion does not follow from a collection of facts and rules.

5.5 Discussion

In this chapter, I introduced the Candide model as a computational architecture for modelling human-like, narrative-based language understanding. As such, I have presented an approach that radically breaks with today's mainstream natural language processing paradigm. Rather than modelling the co-occurrence of characters and words in enormous amounts of textual data, this approach focusses on the logic reasoning processes that may justify different interpretations of the same linguistic observations. While this forces us to take an enormous leap back, it bears the promise of contributing a perspective that emphasises the individual and contextualised nature of linguistic communication to the fields of computational linguistics and artificial intelligence.

I have defined narratives to be chains of reasoning operations that underlie the conclusions drawn by an individual based on their belief system. This belief system is personal and dynamic in nature, as it is continuously being shaped by new linguistic and non-linguistic experiences. Narratives are thus not captured in texts as such, but need to be construed through a personal interpretation process. A narrative thereby reflects the perspective of an individual on the world, as the process of narrative construction necessarily takes one's entire belief system into account.

The construction of a narrative is a means rather than an end. While the end is to reach a conclusion, for example to answer a question, to resolve a co-reference, or to make sense of a novel observation or experience, the means to reach that end is to construe a narrative that is consistent with one's belief system. In this view, the construction of a narrative is not a task in itself, but serves the purpose of solving an external task through human-interpretable reasoning processes. As narratives highly depend on external tasks and individual belief systems, they are hard to annotate in linguistic resources, since whether a narrative is justified or not only depends on whether it is consistent with the input that is observed in combination with the beliefs held by an individual. Narrative-based language understanding therefore largely coincides with the use of explainable

methods for solving a variety of NLP tasks, including question answering, text summarisation and sentiment analysis, with the difference that the focus in evaluation shifts from the task accuracy to the soundness of the reasoning processes involved.

The Candide model operationalises this vision through a combination of frame-based constructional language processing and logic reasoning. As such, the belief system of an agent is represented as a collection of facts and rules that support automated reasoning through logic inference. The Fluid Construction Grammar-based language comprehension component is used to map between natural language utterances and a frame-based representation of their meaning. This semantic representation makes use of the same format as the one used to represent the agent's belief system, facilitating the straightforward integration of new beliefs into the agent's personal dynamic memory. The Prolog-based reasoning component can be leveraged to solve external tasks by proving logic formulae based on the facts and rules stored in the agent's personal dynamic memory. It is during this process of logic inference that narratives emerge as logical explanations that justify the conclusions drawn by an agent. I have illustrated the proof-of-concept implementation of the Candide model by means of a didactic example that shows how two agents who hold slightly different beliefs interpret the same linguistic observation differently, as they construe different narratives that lead to substantially different conclusions.

While I have laid the conceptual foundations of a novel approach to narrative-based language understanding, I left the issue of operationalising the approach on a larger scale unaddressed. An agent could start out as a blank slate, with an empty belief system and grammar. Through experience, an agent would then gradually build up linguistic and non-linguistic beliefs in a constructivist manner through the processes of intention reading and pattern finding. These processes have abundantly been attested in children (see e.g. Pine and Lieven, 1997; Tomasello, 2003) and have more recently been operationalised at scale in artificial agents through abductive reasoning processes (see e.g. Nevens et al., 2022; Doumen et al., 2023; Beuls and Van Eecke, 2023). These preliminary results could be considered to be modest yet promising steps towards the moon-shot of building personal, dynamic and human-interpretable models of narrative-based language understanding.

5.6 Conclusion

This chapter introduced a procedural semantics for frame-based narrative construction. Concretely, I introduced the Candide model, which relies on three foundations. First, a personal dynamic memory that stores all knowledge and beliefs of an agent using a frame-based procedural semantics. Second, a language processing engine that maps between linguistic observations and their frame-based meaning representations. Third, a reasoning engine that reasons over the meaning representation and the knowledge in the personal dynamic memory to come to a conclusion. During this process, narratives emerge. Narratives are defined as the chain of reasoning operations that were needed to come to a conclusion.

The methodology presented here models the human-like capacity of grounding language in knowledge. Moreover, it models a personal interpretation process. Indeed, different humans can come to different interpretations based on their own personal interpretation of the situation. Again, procedural semantics play a crucial role. By using the same frame-based representation for both the knowledge and beliefs of the agent as well as the meaning underlying the utterances, reasoning over the meaning based on the knowledge becomes straightforward. The narratives that emerge from this process are interpretable, since they consist in the chain of reasoning steps that were needed to come to a conclusion, which is explicit and thus interpretable.

Chapter 6

Monitoring the Understanding Process using Integrative Narrative Networks

6.1 Introduction

Part of the research reported on in this chapter has been published in Steels et al. (2022a) and Steels et al. (2022b) and other parts are currently under review at the Journal of Artificial Intelligence Research. It concerns joint work with Luc Steels and Remi van Trijp. The implementation of the experiments was my work.

In the previous chapters, I illustrated how procedural semantics can be used as a methodology to tackle language understanding tasks in a human-like way. Since human-like language understanding is grounded in an environment, the systems introduced in the previous chapters integrate language understanding with vision, memory and knowledge. Another essential aspect of human-like language understanding is that humans can monitor their own understanding process and can identify knowledge gaps when they occur. In this chapter, I will introduce a way of monitoring the understanding process of an agent so that the agent can recognise and signal its own lack of understanding. In order to do so, I introduce the Integrative Narrative Network (INN), a data structure that combines narrative questions (i.e. the expectations that arise during the understanding process) and answers to these questions. By looking at the interaction between the questions and answers in the INN, the agent gets insight in its own understanding process. This way, it can identify its lack of understanding, showing a human-like capacity. I will illustrate the use of the INN with experiments on two different language understanding tasks: a visual dialogue task and a recipe execution task.

In the remainder of the chapter, I first tackle the question of how understanding can be monitored (see Section 6.2), by looking at the definition of language understanding as well as the terms narrative questions and answers. In Section 6.3, I introduce the data structure of Integrative Narrative Networks (INN), that capture the understanding process of an agent in terms of narrative questions and answers. In sections 6.4 and 6.5 I apply the INN as a monitoring system on the tasks of visual dialogue and a recipe execution task respectively. Section 6.6 discusses the contributions and concludes this chapter.

6.2 How to monitor understanding?

Understanding is a concept that has interested various fields of research including, among others, philosophy, psychology, linguistics and artificial intelligence. As a result many definitions of understanding have been proposed (see Chapter 2 for an overview). There is, however, not a clear cut definition of what understanding includes. Nevertheless, it is argued that it relies on both reactive and deliberative intelligence Steels (2023). Reactive intelligence is 'fast' thinking, providing an immediate response (Kahneman, 2011). Deliberative thinking on the other hand is a slow process (Kahneman, 2011), it concerns building a rich model of the situation at hand based on language, knowledge, vision, mental simulation, inference etc. Coming up with this rich model that integrates information from all sorts of knowledge is called understanding Steels (2023).

Blaha et al. (2022) emphasises that understanding is a process. It is an *"ongoing cognitive activity of acquiring, integrating and expressing knowledge according to the task or situation at hand"* (Blaha et al., 2022). For example, understanding a recipe is not only about the outcome of the recipe, i.e. the dish. It is also about finding out how this dish is made and what the different steps are to achieve the end result. Understanding a recipe includes understanding the language in the recipe and it is also about imagining how one would cook the recipe, either in mental simulation or in a real kitchen environment. Furthermore, it involves being able to recover when something goes wrong, for example when a certain ingredient is not in the kitchen or when the cooking process does not go the way as planned. Understanding thus involves going through the process of solving a task.

Hough and Gluck (2019) came to a set of eight common features of understanding among different research fields by reviewing the literature concerning this topic. One of these features is metacognitive monitoring which *"allows identification of knowledge gaps or faulty processing based on detection of discrepancies between knowledge or expectations and the environment."* (Hough and Gluck, 2019, p. 23). Monitoring metacognition (see Cox (2005) for an overview) concerns observing and analysing cognitive processes or in this case, the process of understanding. Monitoring this process is necessary to detect gaps which can then be resolved. For example, when reading a story and a certain word is unknown, the reader can look up the word in a dictionary to learn the meaning of the word. Thus, monitoring and in some cases regulating the understanding process is needed to come to full understanding of the situation.

Following Steels et al. (2022a), I define understanding as the dynamic process in

which different sources of knowledge generate and answer questions. Indeed, when we are trying to understand, certain questions and expectations arise. For example, when reading a 'whodunit' the question 'who is the killer?' immediately pops up and the reader expects that this question will be answered at the end. When a knife is mentioned in the novel, the reader asks him or herself how the knife is related to the story, maybe even expecting that it is the murder weapon. Crucially, it is the knowledge that the reader has about 'whodunits' that provides these questions. Other sources of knowledge, such as language or vision, can provide and answer further questions. Answering these questions plays a crucial role in the degree of understanding that is reached. In the 'whodunit', full understanding is only reached when the murderer is found, but other questions, such as 'what is the motive of the murderer?' may also need to be answered to come to understanding. The state in which the reader feels that all key questions are satisfactorily answered is called narrative closure (Carroll, 2007). This is when the understanding of the situation is reached. A recipe is understood when the main question of how to prepare the dish is answered. However, this is only possible when the sub-questions that arise during the intermediate steps in the recipe are solved.

The dynamic between questions popping up and the need to answer them is continually going on during understanding. Monitoring this process and keeping track how many questions are unanswered, gives an indication how well the understanding process is going. In fact, if key questions are not solved, narrative closure cannot be reached. Moreover, it is possible to steer the process, since the questions that are not answered signify that there is a knowledge gap, which then may get solved by other processes.

In order to keep track of the narrative questions and answers that arise during the process, a novel data structure is introduced: the Integrative Narrative Network (INN). The INN acts as a kind of blackboard to which different knowledge sources can write information. Moreover, I introduce a novel way of measuring the process of understanding by using meta-level monitors to monitor the dynamics in the INNs. The more questions remain unsolved during a task, the less understanding there is. This allows for a quantification of the understanding by measuring the amount of solved questions.

In what follows, I will discuss the Integrative Narrative Networks as a data structure that allows to monitor the understanding process. I will illustrate how the INNs are built up during this process, how they can be monitored and how this information can be used to analyse the understanding by applying the INNs

on two tasks: a visual dialogue task and a recipe execution task.

6.3 Defining Integrative Narrative Networks

An Integrative Narrative Network is a network that integrates all questions and answers that arise during the understanding process of an agent. During this process, many different knowledge sources are consulted, which either pose or answer questions. In Section 6.3.3, I will discuss in detail how the different knowledge sources of language, discourse modelling, perception, reasoning, mental simulation and ontology contribute to the INN. I focus on these knowledge sources since these are the ones that are consulted in the experiments that I will later present. It is however not the case that the knowledge sources that can contribute to the INN are limited to these six.

During the execution of a task, the relevant knowledge sources are monitored using the monitor system in the Babel architecture 3. This system provides a way to define monitors that become active when certain triggers such as new questions or answers, are detected by the knowledge sources. The monitors then collect relevant information by observing the state of understanding at that point, including which knowledge source was responsible and update the INN with this information. So, when for example the language introduces a questions, a node representing this question is added to the INN. When this question is later answered, a node and a link between this answer and the question nodes are added.

6.3.1 Formal definition

In definition 6.3.1, I introduce a formal definition for the Integrative Narrative Networks. The components of INNs are entities, constants, questions, primitives and frames. The questions are represented by variables. Answers can either be other questions, entities or constants. Frames are responsible for introducing questions from the ontology, while primitives are the source of questions from mental simulation. These components can be linked in various ways. For example, the arguments of a primitive are linked to that primitive and the questions that are evoked by a frame are connected to that frame. Questions can be answered by other questions, thus providing a link between two variables. Executing a primitive leads to bindings, which are entities or constants, linked to the arguments of the primitives.

Definition 6.3.1. An *Integrative Narrative Network* (INN) consists of 5 types of elements and links between those elements:

- E is the set of *entities*,
- C is the set of *constants*,
- Q is the set of *questions*,
- P is the set of *primitives*,
- F is the set of *frames*,
- $M \subseteq P \times Q$, called the *meaning network*, is the set of relations between the primitive operations and their arguments,
- $N \subseteq F \times Q$ is the set of relations between the frames and the questions that they raise,
- $O \subseteq Q \times Q$ is the set of relations between questions and questions,
- $R \subseteq Q \times E$ is the set of relations between questions and the entities that answer these questions,
- $S \subseteq Q \times C$ is the set of relations between questions and the constants that answer these questions.

6.3.2 Visualisation

The visual representations of the nodes and edges in the INN reveals the type of element or relation that they represent. The questions are all represented by diamonds. Their colour indicates whether the question is open (green) or answered (red). Blue nodes are used to represent the answers. The nodes are hexagons when they are entities and squares if they are constants. The primitive operations are represented as purple triangles. The colour of the links between the nodes indicates the source of the questions or answers. The link between a primitive operation and its arguments is purple, since the primitive operation raises the question. This question (green node) is then linked through a green link to its answer. The questions that are raised by a frame are linked to the frame via a blue link, indicating that it is the frame that is responsible for asking the questions.

Figure 6.1 shows an example of an INN after execution the utterance "*A sphere is present in the scene.*" followed by "*What is the shape of the aforementioned red cube?*". The execution is performed using the methodology introduced in Chapter 4. Notice how this is a suboptimal dialogue, since there is no "*aforementioned red cube*". Thus, the dialogue cannot be understood completely and questions remain unanswered. The 10 primitive operations of the meaning network underlying the utterance are indicated by purple triangles. The questions are indicated

by either green (solved questions) or red diamonds (unanswered questions). These red diamonds are the questions that remain unanswered due to the impossibility of finding the referent of "*the aforementioned red cube*". Answers to questions are entities represented as blue hexagons. In this example, the entities are not raising other questions, therefore, there are no links starting from the entity nodes.

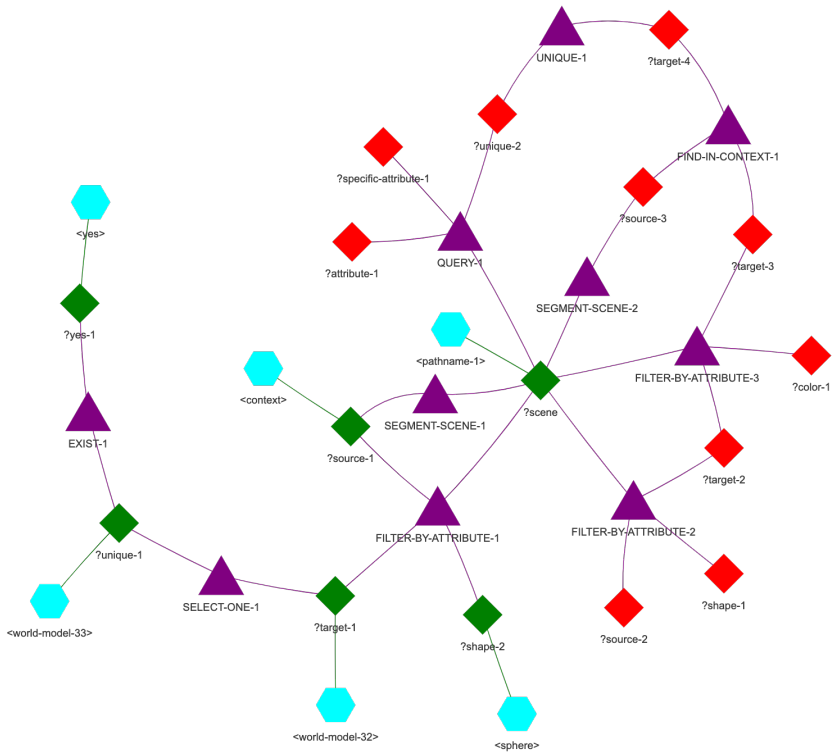


Figure 6.1: An example of an Integrative Narrative Network as part of the visual dialogue about geometric objects. Many questions remain unanswered.

6.3.3 Knowledge sources

Next, I will briefly discuss several possible knowledge sources and how each of them contributes to the INN by either raising or answering questions.

Language A first knowledge source is language, which can raise and answer questions. Language processing is responsible for mapping utterances onto a procedural semantics representation. This means that the meaning consists

of primitive operations, connected through shared arguments. Each of these arguments are variables and are thus narrative questions that are introduced by the language processing. In the experiments, Fluid Construction Grammar (FCG) (see Chapter 3) is used to operationalise language processing. Construction application can either add primitive operations and/or link primitive operations through their arguments. Each new variable that gets added to the meaning is a new narrative question. When two variables are linked to each other, this is considered one narrative answer. Indeed, by linking two variables, we know that these questions are in fact the same, and thus one of those questions is already solved. In the integrative narrative network, a link between the two variables is added.

Discourse model Another source of information is the discourse model. I discuss two different ways of how a discourse model can contribute to the INN. Depending on the implementation of the discourse model, the knowledge source can either (i) pose and answer questions or (ii) only answer questions. When a referent is introduced in the discourse, it can be seen as a question that needs to be solved. Each time the discourse model solves a co-reference, a narrative answer is added to the network. For example, the ingredients in a recipe are entities that will be referred to later on. The introduction of these entities are thus narrative questions and can be seen as the question 'what do you need to do with this ingredient?'. Later, when the recipe refers to "*the butter*", the discourse model solves the co-reference by linking "*the butter*" to the butter from the ingredients and thus solving a narrative question.

Qualitative simulation The qualitative or mental simulation is a knowledge source that is responsible for executing the meaning representations. Since the meaning representation is in procedural semantics, each primitive operations is an action that can be taken in the world. In the experiments, Incremental Recruitment Language (IRL) (see Chapter 3) is used for mental simulation. IRL thus executes the meaning representation provided by the language knowledge source by finding bindings for each of the variables in the semantic network. Each of these bindings, either an entity or a constant, is then a narrative answer provided by the qualitative simulation.

Perception Another knowledge source is perception. This knowledge source is responsible for answering questions related to the visual input that is given. For example, the execution of the primitive operation QUERY answers the question

?WHICH-ATTRIBUTE by looking at the input object bound to ?SOURCE and the image bound to ?SCENE and querying the attribute category bound to ?ATTRIBUTE-CATEGORY.

Reasoning The reasoning knowledge source consists of the primitive operations that perform reasoning operations. This knowledge source is responsible for answering questions that can only be answered using logic reasoning operations. For example, the primitive operation COUNT counts the elements in the input and binds the result to ?NUMBER, thereby answering this question.

Ontology A last knowledge source that I will discuss here is the ontology. The ontology defines the frames that exist in the world as well as the slots of these frames. Entities are instances of these frames. When an entity is introduced during the understanding process, the slots associated with the frame of which the entity is an instance, become narrative questions. When a default or another knowledge source provides a value for a slot, a question gets answered. For example, when a BOWL entity is encountered, the questions of ?COVERED and ?CONTENTS are posed. Immediately, the default values of NOT-COVERED and NO-CONTENTS will be added as answers. During mental simulation, entities are introduced or values of entities are changed. When an entity is introduced, all slots associated with the frame are raised as questions. When the value of a slot changes or when the value of an unbound slot is set, only that slot is considered a question; since the other slots were already answered before.

6.3.4 Quantifying understanding using INNs

It is highly interesting to investigate the contribution of different knowledge sources in the understanding process. For example, one text may rely a lot on common sense knowledge and inference whereas another may rely heavily on an image next to the text. Identifying contributions is not only a function of the text, it also says something about the effectiveness of different knowledge sources. For example, an understanding system could be weak in language processing so that it has to rely more on other knowledge sources such as common sense inference, ontologies and world models. This is like a person who is less familiar with a foreign language but nevertheless tries to make sense of what is being said through the context and background knowledge. Moreover, by being able to pinpoint where the understanding fails, the system can steer its understanding process, either by consulting other knowledge sources, jumping to a meta-level to solve a problem or asking a human for feedback. For example,

in human-machine dialogues, the understanding system can ask targeted help based on which questions in the INN are open at some point in time and could not be answered through other knowledge sources (Thomason et al., 2020).

The information from the INNs makes it possible to find out the contributions of the different knowledge sources and quantify these contributions. This is done via the meta-level monitors in the Babel system that monitor the information of the nodes in the INN. Furthermore, it is possible to identify how well the task is understood by analysing how many questions remain at a certain point in processing. Narrative closure is reached when there are no remaining open questions in the INN.

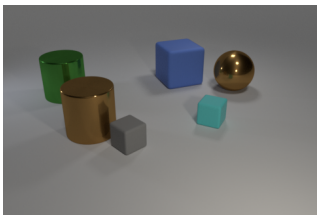
Next, I will apply the methodology of the INNs on two different tasks. I will illustrate in detail how the relevant knowledge sources build up the INN during these tasks. Further I will discuss the dynamics between the questions and answers of the sources. Lastly, I will show how this information can be used to quantify the understanding process in the task.

6.4 Monitoring understanding in a visual dialogue task

The first experiment I will present is on the task of visual dialogue. This task involves modelling an agent that can hold a meaningful and coherent conversation about visual input (Das et al., 2017). More specifically, an agent needs to answer a series of questions about an image. In Chapter 4, I introduced a novel methodology using neuro-symbolic procedural semantics to solve the visual dialogue task. Solving this task requires a variety of knowledge sources: language processing for comprehending the utterances, perception for finding the necessary information in the image that accompanies the dialogue, a discourse model for solving the co-references in the dialogues and reasoning operations operationalised through primitive operations. In the understanding process of the visual dialogue task, narrative questions are mainly invoked by language processing, while perception, discourse and reasoning typically provide narrative answers.

6.4.1 Contributions from different knowledge sources

To illustrate how the different knowledge sources contribute to building the INN, I use the example dialogue shown in Figure 6.2. Figure 6.3 shows the INN after



Caption: A sphere is present in the scene.
If there is an object in front of it, what color is it? - Green
And material? - Metal
How about the aforementioned round object? - Metal
What is the shape of the above green object? - Cylinder
Does the above round object have objects to its behind? - Yes
What number of other objects in the picture share similar material with the above cylinder? - 2
Any other objects? - Yes
If there is an object in front of that cylinder, what color is it? - Cyan
What is the size of the earlier cylinder? - Large
What is the count of objects to its front in the picture? - 3

Figure 6.2: An example dialogue from the CLEVR-Dialog dataset accompanied by an image.

the execution of this dialogue using the methodology described in 4.

Language processing The constructional language processing contributes to the INN by raising and solving questions. The constructions that apply can either introduce variables or link variables. For example, when comprehending the utterance "*A sphere is present in the scene*", the X-IS-PRESENT-CXN shown in Figure 6.4a adds the primitive (EXIST ?YES ?SOURCE) to the meaning, thereby raising two questions ?YES and ?SOURCE. Furthermore, the construction adds the bind statement (BIND BOOLEAN-CATEGORY ?YES YES) to the meaning. Notice that the ?YES variable is the same, so this construction immediately answers the question ?YES that it raised before by connecting the variable to the bind statement. Of course the question ?SOURCE remains. Figure 6.4b shows the INN after the construction applies. There is a primitive operation EXIST indicated by a purple triangle, two diamond shaped nodes, indicating questions and a constant, which is a square.

Perception, reasoning and discourse The knowledge sources of discourse and perception are consulted as part of the execution in IRL through primitives that consult the discourse model, trigger perception or perform reasoning operations. In Figure 6.5a the primitive GET-LAST-TOPIC consults the discourse model to find the topic of the previous turn. The primitives SEGMENT-SCENE, FIND-IN-CONTEXT and QUERY are perception operations and respectively (i) segment the scene into the set of objects, (ii) find the previous topic in the scene and (iii) query the object for a shape. The UNIQUE primitive performs a logical operation to check whether there is only one object in its input. These different knowledge sources thus contribute 5 answers, with 1 answer provided by the discourse model (the answer WORLD-MODEL-3 for the question ?OBJECT-149), 3 answers provided by perception (the answer CONTEXT for ?CONTEXT-538 and ?OBJECT-SET-113, and SPHERE for ?SPECIFIC-ATTRIBUTE-227) and 1 answer provided by reasoning

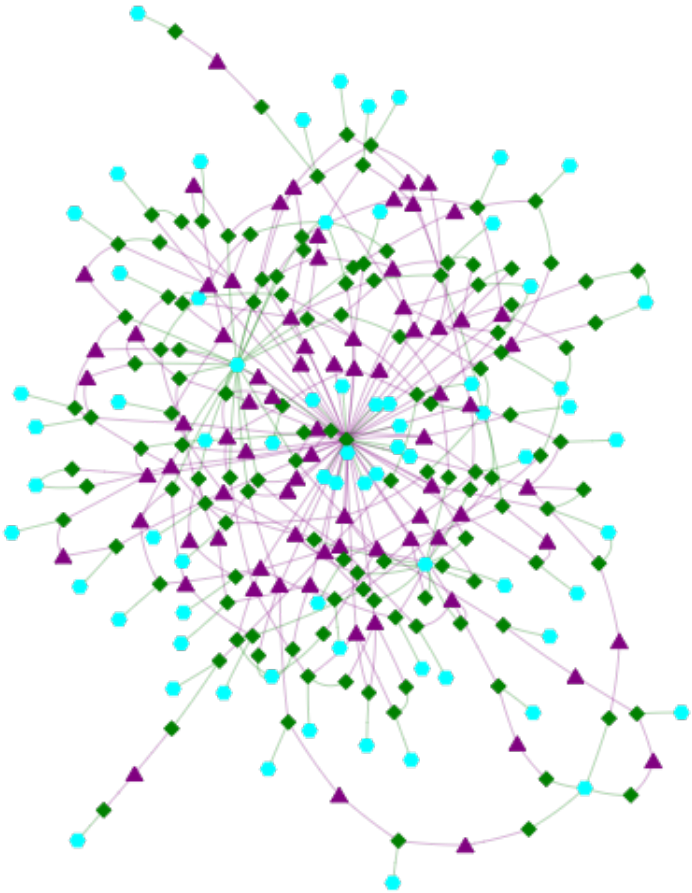
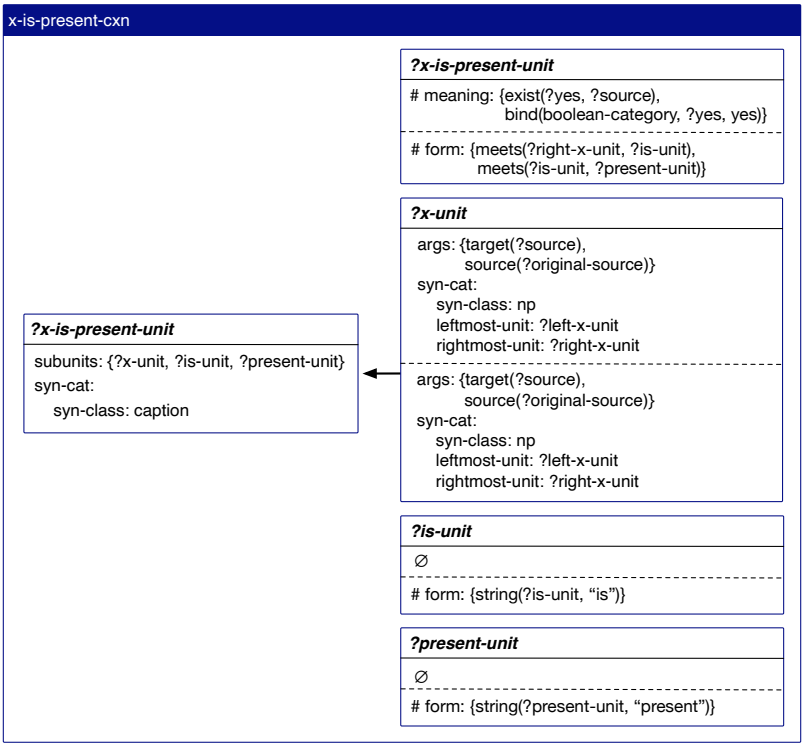
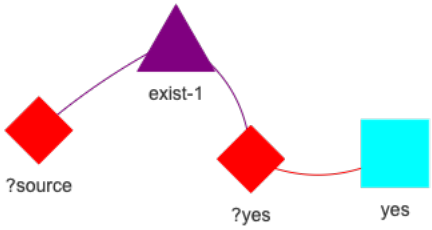


Figure 6.3: The Integrative Narrative Network after executing the dialogue shown in Figure 6.2.

(the answer WORLD-MODEL-4 for ?UNIQUE-474). The bindings of CONVERSATION-MEMORY to ?MEMORY, PATHNAME-ENTITY-2 to ?SCENE and SHAPE to ?ATTRIBUTE-225 were already solved by language processing. The Integrative Narrative Network in Figure 6.5b shows the network after execution. The five primitives are instantiated and visualised by purple triangles. The arguments that were posed as questions are visualized by green diamonds. Each of the questions is answered by an entity, represented with a blue square since these entities are not further associated with a frame here.

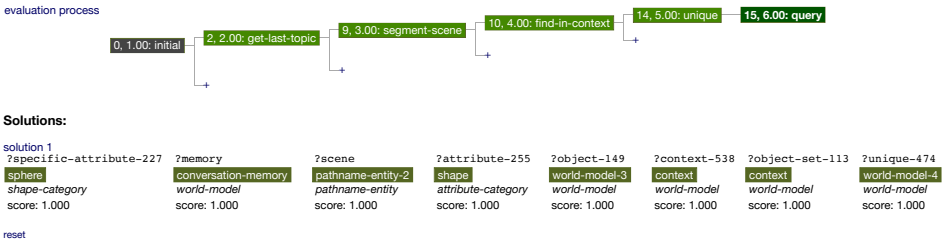


(a) A schematic representation of the construction named X-IS-PRESENT-CXN.

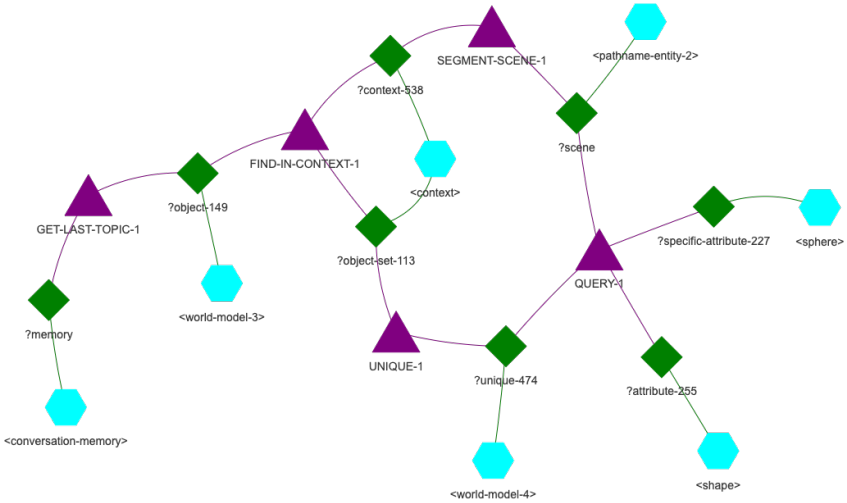


(b) The INN after the X-IS-PRESENT-CXN has applied.

Figure 6.4: Contributions from language to the INN



(a) The execution of the IRL network as part of the visual dialogue example shown in Figure 6.2. The primitive GET-LAST-TOPIC consults the discourse model, the primitives FIND-IN-CONTEXT, SEGMENT-SCENE, QUERY consult the perception knowledge source and the UNIQUE primitive performs a reasoning operation.



(b) The INN after the execution of the IRL network. The primitives are indicated by purple triangles, their arguments by green nodes. The questions raised by the primitives are solved by consulting the discourse model, performing perception, and reasoning. The answers are represented as blue hexagons or squares.

Figure 6.5: Contributions from discourse, perception and reasoning to the INN

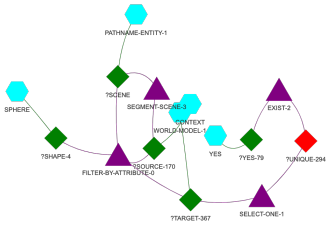
6.4.2 Building the INN

Figure 6.6 shows the INN at four different steps of the execution of the dialogue shown in Figure 6.2. The first network is a snapshot during the execution of the semantic representation underlying the caption "*A sphere is present in the scene.*", which was obtained after language processing. Four primitive operations, represented by purple triangles, are in the INN, and the SELECT-ONE primitive is currently being executed. This primitive is connected to one red question, which is the only question that remains before successfully understanding the caption. Next, the questions from the dialogue are each mapped onto a meaning representation by language processing and then executed with IRL. The second figure is a snapshot of the execution of the meaning representation underlying "*How about the aforementioned round object?*". Again, this is moments before fully understanding this question, since only two nodes remain unanswered. The third figure shows the INN right after the execution of the two last primitives, all narrative questions have been answered, which is necessary in order to answer the question from the task, before going further in the dialogue. The last figure is the INN later in the dialogue, again during IRL. The INN has grown significantly and a few questions remain unanswered. At the end of the dialogue, the INN will look like the one shown in Figure 6.3.

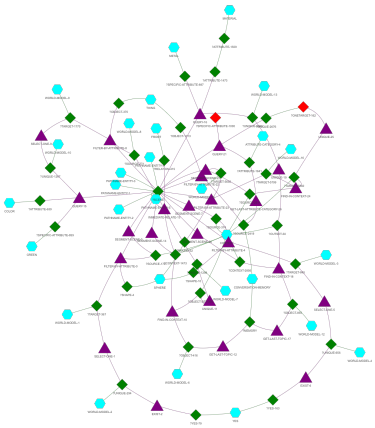
6.4.3 Quantifying the contributions of knowledge sources

It is possible to monitor the number of questions and answers that arise during the understanding process. For example, Figure 6.7 shows how much each of the knowledge sources contributed during the execution of the dialogue shown in Figure 6.2. The black line shows the total number of questions that were introduced. The stacked areas are the different knowledge sources that were used during the understanding process: language, perception, inference and the discourse model. Each time a question from the dialogue is executed, the stacked area and the black line come together. Thus, after solving a question from the dialogue, all narrative questions that were raised were answered. At the end, narrative closure is reached, since all questions in the INN are answered.

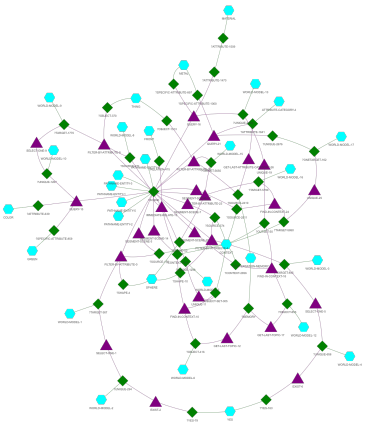
Language is the largest contributor of answers when executing the dialogue. Perception, inference and the discourse model are almost equally important, with inference (reasoning) being the knowledge source that answers slightly more questions.



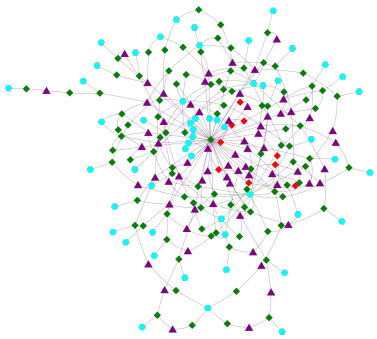
(a) Snapshot of the INN during the execution of the semantic network underlying the caption.



(b) Snapshot of the INN during the execution of the semantic network underlying the third question in the dialogue: "*How about the aforementioned round object?*".



(c) Snapshot of the INN just after the execution of the semantic network underlying the third question in the dialogue: "*How about the aforementioned round object?*".



(d) Snapshot of the INN at the end of execution of the entire dialogue.

Figure 6.6: The INN at four different stages of the execution of the visual dialogue.

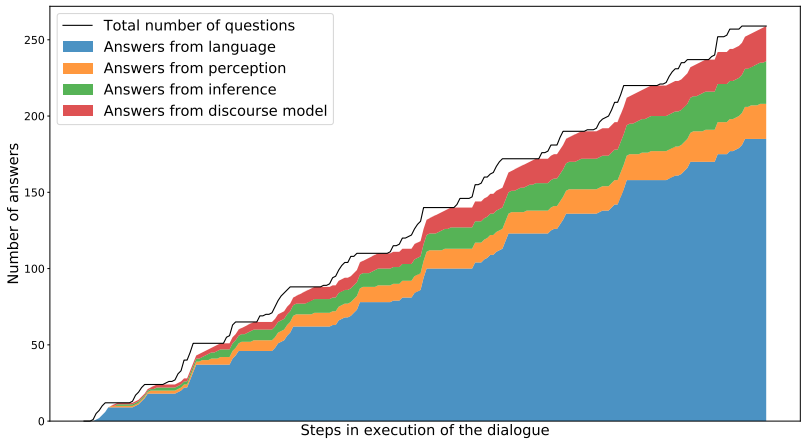


Figure 6.7: The contribution of the different knowledge sources when executing the visual dialogue shown in Figure 6.2. Notice that at certain steps, the stacked bars and the black line meet, corresponding to a single question in the dialogue, meaning that all narrative questions raised so far are answered.

6.5 Monitoring understanding in a recipe execution task

The recipe execution task as defined by Nevens et al. (2023); De Haes (2023) consists of the task of mapping between recipes and a procedural semantics representation that can be executed in a simulated kitchen environment. The procedural semantics representation consists of a set of primitive operations. I will take as an example the almond crescent cookies recipe from this dataset, for which a computational construction grammar was developed to map between the recipe and the semantic representation. A solution for the execution of the semantic representation is provided by the benchmark. Concretely, understanding the recipe goes as follows. The first instruction is taken and mapped onto its meaning representation using a construction grammar operationalised through FCG. Then the resulting meaning is executed in mental simulation using IRL. This process continues until each of the utterances in the recipe is executed. Mapping between the language and the meaning is not trivial, since the language in recipes is often fragmented and underspecified. For example, an instruction could be "*mix thoroughly*", in which it is not explicitly stated what needs to be mixed. Further, recipes contain several co-references, such as "*the butter*" or "*the*

mixture". To handle these challenges, the grammar contains constructions that can consult the discourse model and the ontology during comprehension.

The web demonstration https://ehai.ai.vub.ac.be/demos/recipe-understanding/ shows the execution of the almond crescent cookies recipe.

Recipe for almond crescent cookies:
Ingredients: 226 grams butter, room temperature. 116 grams sugar. 4 grams vanilla extract, 4 grams almond extract, 340 grams flour, 112 grams almond flour, and 29 grams powdered sugar
Instructions:
1. Beat the butter and the sugar together until light and fluffy.
2. Add the vanilla and almond extracts and mix.
3. Add the flour and the almond flour.
4. Mix thoroughly.
5. Take generous tablespoons of the dough and roll it into a small ball, about an inch in diameter, and then shape it into a crescent shape.
6. Place onto a parchment paper lined baking sheet.
7. Bake at 175 degrees Celsius for 15 - 20 minutes.
8. Dust with powdered sugar.

During the execution of the recipe, several knowledge sources are consulted. The language, discourse model and ontology raise and answer questions. The mental simulation (i.e., the execution of the semantic representation in a simulated environment), only provides answers. Figure 6.8 shows the INN after executing the almond crescent cookies recipe. Now I will discuss in detail how the different knowledge sources contributed to building the INN.

6.5.1 Contributions from different knowledge sources

Language processing Consider comprehending the utterance "*226 grams butter, room temperature.*" While comprehending this instruction, several constructions apply, including the QUANTITY-UNIT-INGREDIENT-CXN (shown in Figure 6.9a). The QUANTITY-UNIT-INGREDIENT-CXN looks for units with a numeral for the quantity, a noun for the unit and a noun for the ingredient. It also accesses the kitchen-state, which is the symbolic representation of the kitchen in which the recipe is executed. When these units are found and the construction applies, the instantiated primitive (FETCH-AND-PROPORTION-1 ?INGREDIENT-OUT ?KITCHEN-STATE-OUT ?KITCHEN-STATE-IN ?TARGET-CONTAINER ?INGREDIENT-IN ?QUANTITY ?UNIT) is added to the INN, thereby introducing 7 narrative questions. The construction also solves 4 four questions by linking the variables ?INGREDIENT-IN, ?QUANTITY,

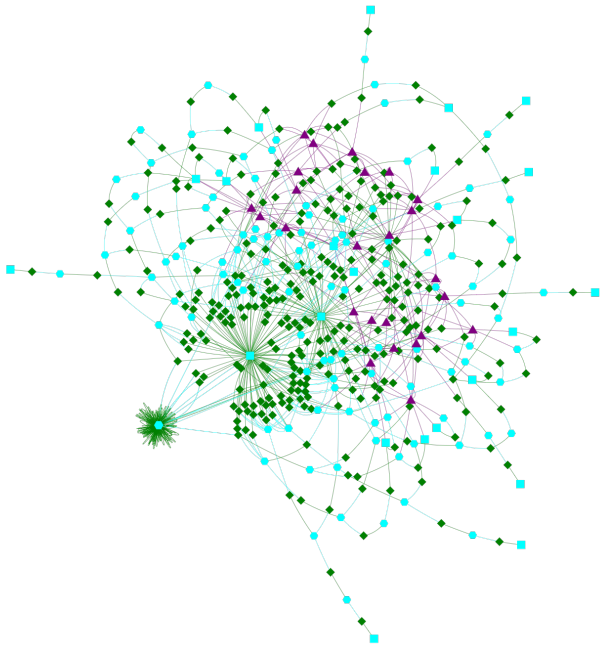
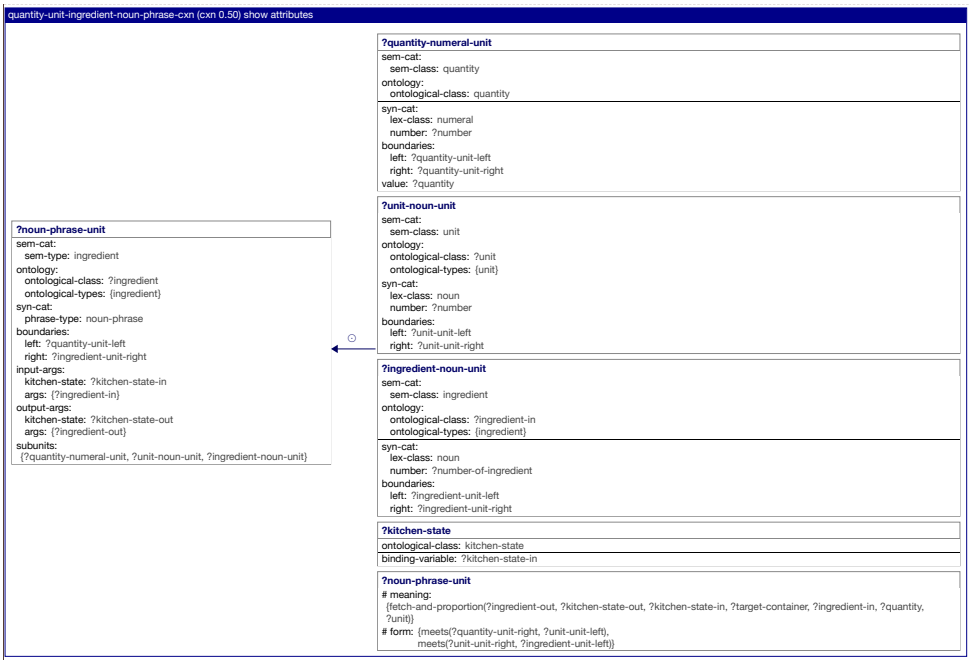


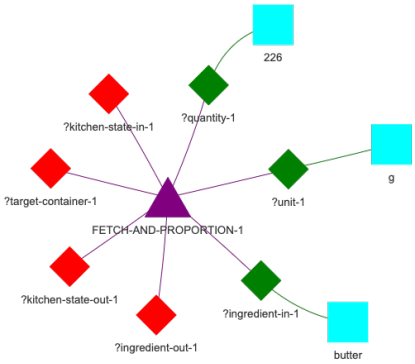
Figure 6.8: The INN after executing the almond crescent cookies recipe.

?UNIT and ?KITCHEN-STATE-IN to variables from the linguistic units. The INN after application of the QUANTITY-UNIT-INGREDIENT-CXN is shown in Figure 6.9b). The primitive FETCH-AND-PROPORTION-1 is represented as a triangle. Its arguments, the 7 questions, are represented as red diamonds. 3 of these arguments were linked to constants that were introduced earlier. Since these questions are thus solved, the nodes are green. The other 4 red nodes remain open questions at the moment. They can become green later on, either by linking them to other questions, or by binding them in mental simulation.

Qualitative mental simulation Qualitative mental simulation is implemented through the IRL system. Executing the semantic representation thus involves finding consistent and complete bindings for the primitives in the meaning representation. Executing a primitive operation either creates a new entity, changes the properties of an entity or introduces a constant. Each binding is an answer to a question. For example, the network in Figure 6.10a consists of two primitives FETCH-AND-PROPORTION and BRING-UP-TO-TEMPERATURE. The FETCH-AND-PROPORTION finds a certain ingredient of a certain quantity and unit in the kitchen state, proportions this ingredient and returns a new kitchen state with the proportioned ingredient. The execution started from the initial bindings



(a) During comprehension, the QUANTITY-UNIT-INGREDIENT-CXN looks for a kitchen-state, quantity, unit and ingredient. It introduces the primitive FETCH-AND-PROPORTION together with its arguments.

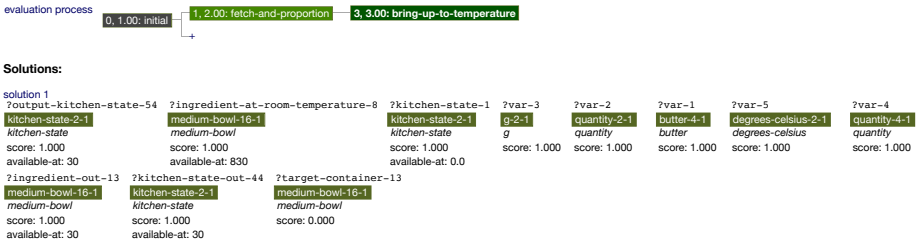


(b) The Integrative Narrative Network that is the result of the QUANTITY-UNIT-INGREDIENT-CXN. It consists of the primitive FETCH-AND-PROPORTION-1 (indicated by the triangle) and its arguments ?UNIT-1, ?QUANTITY-1, ?INGREDIENT-IN-1, ?INGREDIENT-OUT-1, ?TARGET-CONTAINER-1, ?KITCHEN-STATE-IN-1, ?KITCHEN-STATE-OUT-1, some indicated by red diamonds, meaning that the questions have not yet been resolved, some with green diamonds, meaning that they have been solved.

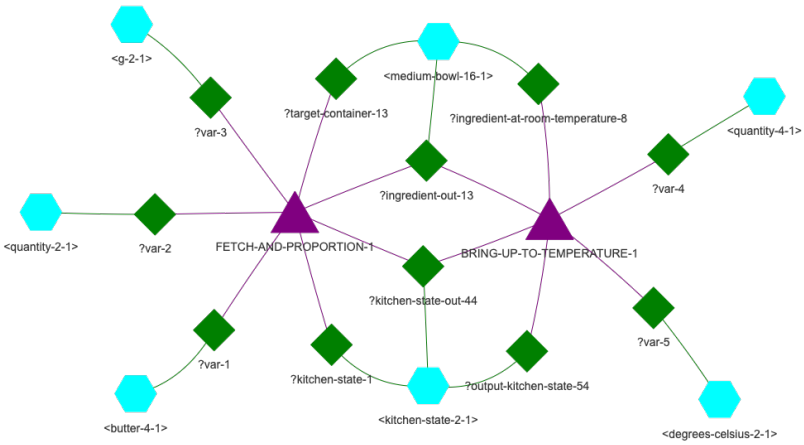
Figure 6.9: Contributions from language processing to the INN

of BUTTER-4-1 to ?VAR-1, QUANTITY-2-1 to ?VAR-2, G-2-1 to ?VAR-3 and KITCHEN-STATE-2-1 to ?KITCHEN-STATE-1 provided in the language processing and bound the entity MEDIUM-BOWL-16-1 to ?INGREDIENT-OUT-13, MEDIUM-BOWL-16-1 to ?TARGET-CONTAINER-13 and KITCHEN-STATE-2-1 to ?KITCHEN-STATE-OUT-44, thereby providing 3 answers. Then, the BRING-UP-TO-TEMPERATURE primitive makes sure that this ingredient in the new kitchen state is brought up to room temperature and then returns the ingredient and another kitchen state. The primitive binds the entity MEDIUM-BOWL-16-1 to ?INGREDIENT-AT-ROOM-TEMPERATURE-8 and the entity KITCHEN-STATE-2-1 to ?OUTPUT-KITCHEN-STATE-54, thereby providing two answers. The bindings QUANTITY-4-1 to ?VAR-4 and DEGREES-CELSIUS-2-1 to ?VAR-5 were already provided during the language processing. The bindings MEDIUM-BOWL-16-1 to ?INGREDIENT-OUT-13 and KITCHEN-STATE-2-1 to ?KITCHEN-STATE-OUT-44 were made during the execution of the previous primitive. In short, executing these two primitive operations answered 5 questions. The INN in Figure 6.10b shows the two primitives in the form of purple triangles, linked to their arguments represented as diamonds. All the arguments are bound to an entity, thus indicating that all questions are answered by the simulation.

Discourse model In the recipe execution task, the discourse model is responsible for raising questions and answering questions. It is implemented through a list of accessible entities, which are the entities that are accessible or under consideration at that moment in processing. These entities result from the mental simulation. For example, if a certain ingredient is mentioned in a recipe, this ingredient becomes accessible at that moment, meaning that instructions later on can refer back to it. Concretely, the bindings that results from mental simulation are added to the list of accessible entities. Each of these entities is a question that is raised by the discourse model. For example, executing the meaning representation underlying the instruction "*116 grams sugar*" yields a bowl with the proportioned sugar in it. Then, the question of 'what to do with this bowl?' is raised by putting the entity on the list of accessible entities. The information on the list consists of the binding variable of the entity, its ontological class and types and the slots of the entity. This list is then added to the transient structure (see 6.11a), so that constructions, such as the THE-X-CXN shown in Figure 6.11b, can access it during comprehension. These types of constructions match with the units in the accessible entities. For example, if the utterance contains "*the sugar*", it refers to the sugar that was mentioned before. So, the construction needs to find this sugar in the list of accessible entities by comparing the ontological class that rises from the word 'sugar' with the ontological classes of the elements in the list of accessible entities. Concretely, the ?X-UNIT-IN-WORLD-UNIT



(a) Qualitative mental simulation solving the IRL network.



(b) The Integrative Narrative Network after the qualitative mental simulation. It consists of the primitives FETCH-AND-PROPORTION-1 and BRING-UP-TO-TEMPERATURE-1. These primitives pose different questions (the arguments of the primitives), which are all solved during the mental simulation (indicated by the blue nodes).

Figure 6.10: Contributions from qualitative mental simulation to the INN

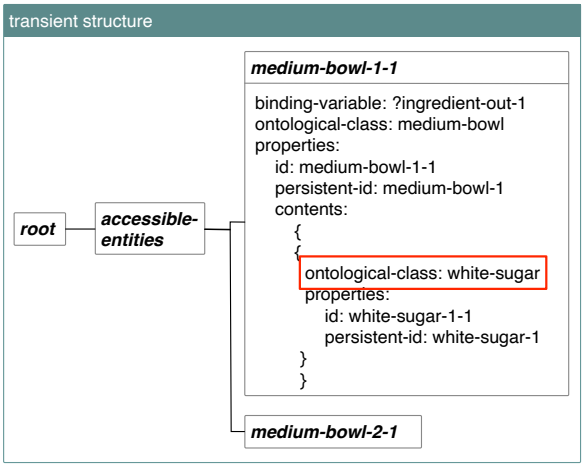
in the THE-X-CXN looks for a unit with a similar ontological-class as the ?X-UNIT-IN-UTTERANCE-UNIT, thereby ensuring that the ontological class of the entity in the accessible entities (bound to the ?ONTOLOGICAL-CLASS-WORLD variable) and the ontological class of the utterance (bound to the ?ONTOLOGICAL-CLASS-UTTERANCE variable) is the most similar.[1] Since the entity that is raised as a question by the discourse model was already in the INN, the node that refers to the binding variable belonging to the entity becomes green. Once the question is solved, a link from the binding variable of the entity to the answer in the INN is added. Then, the red node becomes green again.

Ontology The ontology defines the slots for an entity based on the frame with which it is defined. Using an example from recipe understanding, consider the entity BUTTER-1 shown in Figure 6.12a. This entity is a frame-instance of the BUTTER frame, and has the slots 'melted', 'beaten', 'mixed', 'keep-refrigerated'. Each of these slots has a default value given in the ontology, either 't' or 'nil'. Thus, by introducing this entity, four questions are posed and immediately answered. During qualitative mental simulation, entities are created and added to the INN (see Figure 6.12b). The entity itself is visualised by a blue hexagon node, connected with blue lines to the four questions raised by the frame associated with this entity. The names of the questions for each of the slots are a combination of the name of the slot and the entity the slot belongs to: ?MELTED-BUTTER-1, ?BEATEN-BUTTER-1, ?MIXED-BUTTER-1, and ?KEEP-REFRIGERATED-BUTTER-1. The questions are visualised as diamonds. They are green since they have been answered with default values ('t' or 'nil'). The answers 't' and 'nil' are visualised with blue squares because there is no frame associated with these values. These default values can be overwritten later.

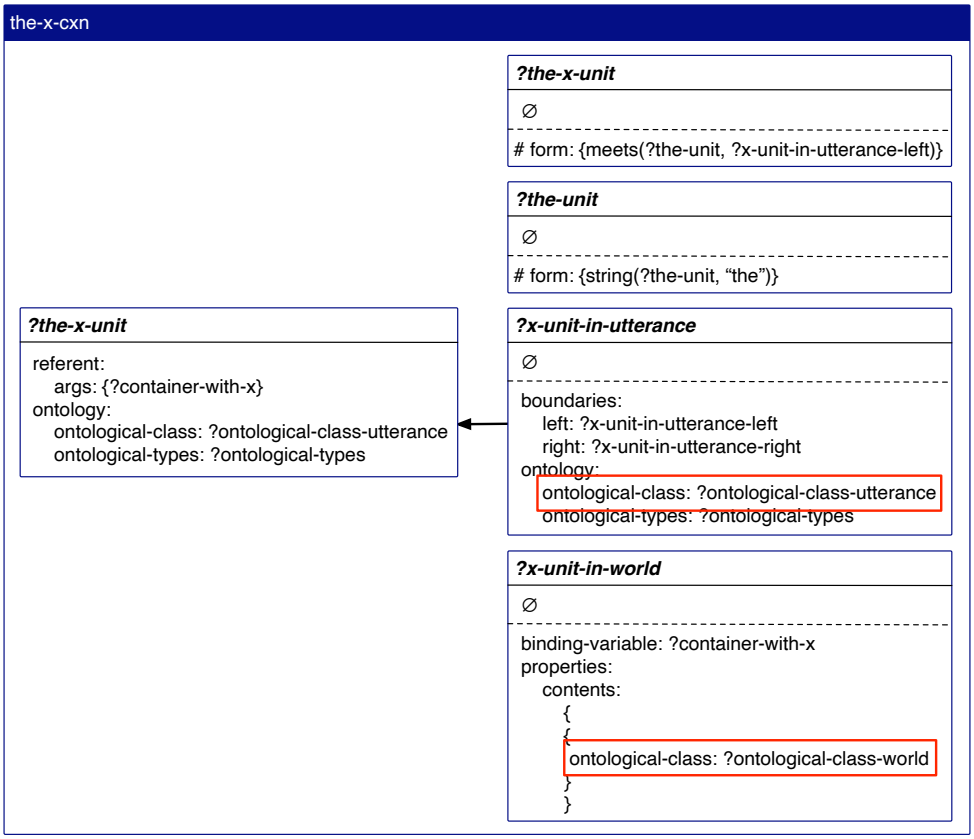
6.5.2 Building the INN

Next, I will show how the different knowledge sources interact to build up the Integrative Narrative Network. Figure 6.13 shows the INN at four different stages during the execution of a small recipe:
226 grams butter, room temperature. 116 grams sugar.
Beat the butter and the sugar together until light and fluffy.
The first figure shows the INN during the mental simulation of the semantic representation underlying the instruction "*226 grams butter, room temperature*".

[1] The THE-X-CXN uses an *expansion operator* that allows for procedural attachment (see Van Eecke (2018, p. 44), to compare the ontological classes bound to the variables ?ONTOLOGICAL-CLASS-WORLD and ?ONTOLOGICAL-CLASS-UTTERANCE.

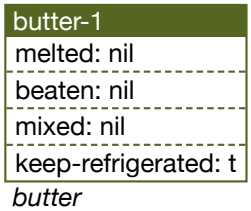


(a) An example of the transient structure with two accessible entities MEDIUM-BOWL-1-1 and MEDIUM-BOWL-2-1. Some properties of the medium bowl were left out due to illustrative purposes.

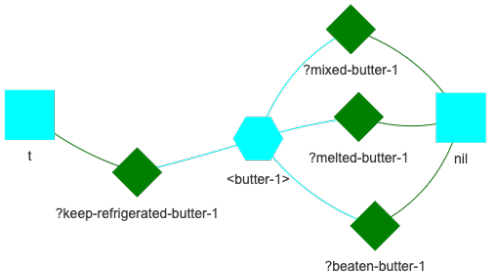


(b) The THE-X-CXN that solves co-references by accessing the accessible entities with the X-UNIT-IN-WORLD-UNIT.

Figure 6.11: Contributions from qualitative mental simulation to the INN



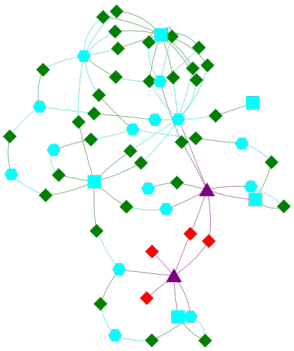
(a) The BUTTER-1 entity, which is an instance of the BUTTER frame with the slots 'melted', 'beaten', 'mixed' and 'keep-refrigerated'.



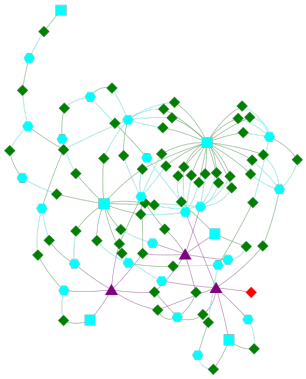
(b) The Integrative Narrative Network of the BUTTER-1 entity. Four questions are posed (?MELTED-BUTTER-1, ?BEATEN-BUTTER-1, ?MIXED-BUTTER-1, ?KEEP-REFRIGERATED-BUTTER-1) and are answered (so far) by their default values.

Figure 6.12: Contributions from the ontology to the INN

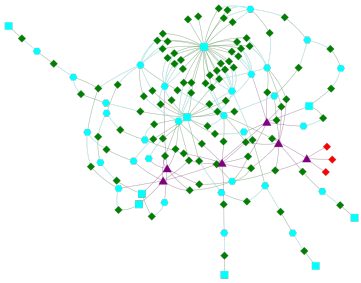
The semantic representation consists of two primitive operations (FETCH-AND-PROPORTION and BRING-UP-TO-TEMPERATURE), represented in the network as purple triangles. The arguments of the primitives are connected via purple links. Once a primitive is executed, the entities that are bound to the arguments appear in the INN, with a green link from the argument to the entity. The questions that the frame associated with an entity raises are visualised by diamonds connected to the entity with a blue line. These questions can be answered immediately by a constant value (a blue square) or an entity (a blue hexagon), which can raise even more questions. Then, the second instruction "*116 grams sugar*" is mapped onto its semantic representation, consisting of one primitive operation: FETCH-AND-PROPORTION. The INN in Figure 6.13b now consists of three primitive operations visualised by purple triangles. This figure is a snapshot during the mental simulation of the FETCH-AND-PROPORTION primitive. One diamond connected to a purple triangle is still red, namely the argument of the FETCH-AND-PROPORTION primitive that will be bound once the primitive is executed. The third figure is again a snapshot during the execution of a primitive, now the BEAT primitive. Three questions still need to be solved, indicated by the red diamonds. The last figure shows the INN at the end of the execution of the recipe, all questions are solved, and all red diamonds have become green. Narrative closure is reached.



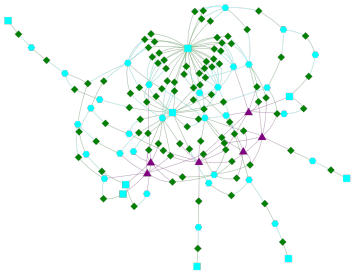
(a) The INN after comprehending the first instruction "*226 grams butter, room temperature.*" and mentally simulating the primitive operations FETCH-AND-PROPORTION and BRING-UP-TO-TEMPERATURE .



(b) The INN after comprehending the second instruction "*116 grams sugar.*" and mentally simulating the FETCH action.



(c) The INN with three remaining questions (indicated by red nodes) during the execution of the BEAT primitive in the semantic network underlying "*Beat the butter and the sugar together until light and fluffy.*"



(d) The INN at the end of executing a small recipe, all questions have been solved.

Figure 6.13: The INN at four different stages of the recipe execution.

6.5.3 Quantifying the contributions of knowledge sources

Next to collecting the questions and answers into the INN, it is also possible to keep track how many questions and answers are provided by each knowledge source. Figure 6.14 shows an example of a stack plot showing the contribution of the different knowledge sources in providing answers during the execution of the almond crescent cookies recipe. The black line shows the number of questions that were raised during the execution. The stacked areas show the number of answers from each knowledge source that was used during the execution. At the end, the stacked area meets the black line, meaning that all questions have been answered and narrative closure has been reached. The stacked areas follow the black line very closely, since most of the questions are all almost immediately answered. In the middle of the execution the gap between the questions and answers is the largest. This is after the execution of the ingredients, and these ingredients are all questions that were raised by the discourse model. They are answered later when the instructions refer back to the ingredients list.

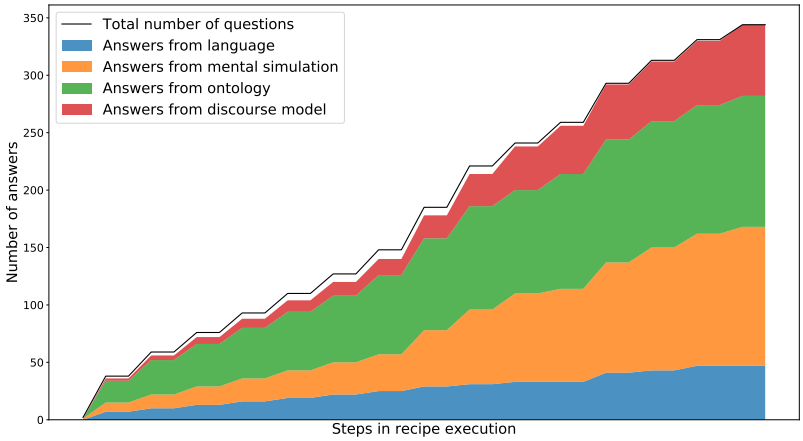


Figure 6.14: The contribution of the language, mental simulation, ontology and discourse model during the execution of the almond cookies recipe.

In this experiment, the contributions of the mental simulation and the ontology are larger than the contributions from the discourse model and the language. Indeed, the ontology provides a large number of answers by filling in slots with default values. The mental simulation is also responsible for a large number of

answers, by filling and changing values of the entities. The language provides a smaller number of answers by linking the questions in the meaning networks. The discourse model is also a small contributor in terms of number of answers. It is important to note however that the importance of a knowledge source cannot be judged solely on the total number of answers it provides. The discourse model is in terms of total number of answers the smallest contributor, but the answers that it does provide are crucial to the execution of the recipe. Specifically, it is essential that the co-references in the recipe are solved, so that all ingredients and instructions are processed correctly. Not solving a question that was raised by a frame, could, in contrast, be less crucial to the understanding process.

6.6 Conclusion

In this chapter I introduced a way of monitoring the understanding process by introducing a novel data structure: Integrative Narrative Networks (INN). The INN captures the narrative questions and answers that are raised during the understanding process. The INNs are thus rich models of the situation at hand that integrates information from different sources of knowledge. I applied the methodology on two different task, showing the applicability of the methodology. I explained in detail how the INNs are built up by the knowledge sources and how the understanding process of the two tasks can be quantified and analysed. As shown, INNs can be used to study and understand the understanding processes. I focussed on the following aspects of the INNs:

- I discussed how the INNs can be used to identify when something is fully understood and thus whether narrative closure is reached. By counting the number of questions that are unsolved at the end of processing, it can be analysed whether closure is achieved. Ideally, this number is zero. If not, the questions that are unsolved can be identified, showing where the understanding went wrong. This demonstrates the insights that the INNs provide in the understanding process and the interpretability of the INNs themselves.

- I discussed how the contributions of each of the knowledge sources towards understanding can be quantified, making it possible to identify which knowledge sources were crucial towards understanding. Furthermore, it can be useful to find out which knowledge sources are not performing adequately, by analysing which knowledge source is responsible for the

most unanswered questions.

- I mentioned how the INNs could be used when understanding fails. Indeed, their structure, dynamics and quantitative characterisations can be used to steer the understanding process itself. In particular it can guide the agent to actions that accelerate or improve understanding by focusing on certain questions and activating or giving more computational resources to knowledge sources that can help to resolve them.

The work I presented is a step towards building systems that can track and evaluate their own understanding process. Future research can investigate how the methodology can be used to steer the understanding process more actively, either by allowing the agent to ask for feedback or by using meta-level operators so that the system can resolve the questions itself. The INNs allow for the system to monitor its own understanding process, thereby showing a human-like, interpretable capacity that is needed during language understanding.

Chapter 7

Conclusions

7.1 Introduction

In this thesis, I investigated how systems that perform human-like language understanding can be built, focussing on certain aspects of human-like language understanding, namely grounding, meaning and self-reflection. Nowadays, the main paradigm in the field of NLP lies in statistical, data-driven approaches. Following Bender and Koller (2020), I discussed how these data-driven systems are not designed to achieve true human-like language understanding, primarily due to their focus on the form side of language by learning from large amounts of textual data. Therefore, they do not take into account meaning or grounding in the environment, which are both inherently connected to human-like language

understanding. In order to investigate how human-like language understanding systems can be built, I limited the scope of the broad field of human-like language understanding to three requirements which the systems introduced in this thesis should meet. First, the systems are required to have adequate mechanisms for representing and processing meaning. Meaning is a crucial aspect of language since communication is the process of transferring a (non-observable) meaning from a listener to a speaker by conveying linguistic utterances. To enable language understanding systems to handle meaning, it is necessary to find and design adequate meaning representations that capture the semantics underlying linguistic utterances. In this thesis, procedural semantics is chosen as meaning representation. It is a highly useful meaning representation to operationalise the semantics in intelligent systems since the meaning representation is directly executable, removing the need for an extra step that translates the meaning representation in something that the intelligent system can use. Second, the systems needed to be able to ground linguistic utterances in the environment. Language is not an independent system that can be disconnected from the environment that it is used in. Crucially, other sources of knowledge such as vision systems or knowledge bases need to be consulted during the language understanding process. Third, the capability of humans to monitor their understanding process needed to be modelled. This provides the possibility for an agent to reflect on its own language understanding process. This can be useful, for example, in situations where humans work together with intelligent agents since it is crucial that systems can signal their lack of understanding and maybe even ask humans for feedback.

Concretely, in this thesis I introduced three systems (discussed in detail in Section 7.2) that operationalise these requirements. The systems that are introduced in this thesis are interpretable by design, which is an important aspect of intelligent systems. When these systems are used in our society, it is key that humans can interpret their decisions. A goal of the thesis was to show that systems that are perform more human-like language understanding are capable of competing with more data-driven approaches on benchmark datasets to show the possibility of scaling such systems. This was achieved with the introduction of the neuro-symbolic procedural semantics that achieved competitive results on two benchmarks datasets for the task of visual dialogue. This thesis also laid out the foundations of human-like language understanding systems. This resulted in a proof-of-concept implementation of these foundational ideas in the form of the Candide model.

7.2 Achievements

This thesis provides insights on how intelligent systems that perform human-like language understanding can be built. In order to do so, I introduced three systems that model aspects of human-like language understanding. Each of these systems provide insights on both the conceptual as methodological level, resulting in the concrete achievements that are discussed below.

7.2.1 A neuro-symbolic procedural semantics for visual dialogue tasks

A first achievement consists in the introduction of a neuro-symbolic procedural semantics that is able to ground linguistic utterances in the image and dialogue history. This system models the human-like capacity of being able to (i) integrate linguistic utterances with the environment in which they are used and (ii) integrate linguistic utterances with the history of the conversation. This novel methodology is validated on two benchmark datasets for the task of visual dialogue.

Concretely, the achievement is the introduction of a novel methodology, which consists of (i) a neuro-symbolic procedural semantics that integrates symbolic and subsymbolic primitive operations and (ii) a conversation memory that stores the history of the dialogue in an incremental and explicit way. Symbolic operations perform reasoning operations and extract knowledge from the conversation memory. The subsymbolic primitives on the other hand are related to perception and rely on a shared set of neural modules. Each of these modules is a neural network specialised in a specific task (e.g. recognising a specific colour or shape, image segmentation ...). The combination makes it possible to exploit the strengths of both approaches and contributes to the growing literature that shows that neuro-symbolic approaches are well-suited to solve tasks that involve both structured and unstructured data. Moreover, the system is flexible and easily expandable, due to the possibility of adding new primitive operations. Concretely, when the dataset is extended with, for example, new colours, it is only necessary to train modules for those colours. These modules can then be added to the set of modules, but there is no need to retrain the whole system. Due to the shared set of neural modules underlying the primitive operations, the reasoning becomes consistent. For example, the primitive operations that filter or query for a certain colour both rely on this colour-recognising module, so both primitives will be consistent in their recognition of that colour. Both the

modularity and the consistency of the systems are two major advantages of the introduced approach.

This model achieves competitive results on two benchmark datasets in the field of NLP, thereby showing that systems that model human-like aspects can indeed compete with more data-driven approaches. Moreover, I showed that this methodology is interpretable by design. I demonstrated the interpretability by interpreting the result of the execution process of the procedural semantics underlying incorrectly answered questions. By interpreting the reasoning process of the system, it possible to identify which operation led to the wrong answer. This gives then the opportunity to reimplement, retrain or adjust these primitive operations, to avoid that the mistake happens again. In short, the answers of the systems can be easily explained and interpreted by users.

7.2.2 A frame-based procedural semantics for narrative construction

A second achievement consists in the introduction of the Candide model, a model for narrative construction. It consists of a frame-based procedural semantics that enables an agent to construct narratives by interpreting an utterance in the light of its personal dynamic memory (PDM). This system aims to model the human-like capability of interpreting a situation in a personal and dynamic way by grounding linguistic utterances in personal knowledge. This can lead to different interpretations of the same observation across the population. It can even be the case that the interpretation of the same individual changes over time due to new information that the individual has acquired.

Methodologically, a language understanding system that relies on a novel frame-based procedural semantics was introduced. The interpretation process starts by using the agent's grammar to comprehend a linguistic observation resulting in a frame-based procedural semantics representation. Next, this meaning representation is executed by reasoning over the personal dynamic memory of an agent using a Prolog-based inference engine. Each aspect of the interpretation process is thus individual to the agent and can result in the construction of different narratives across a population of agents. Crucially, the meaning that is provided by the language processing is represented in the same way as the knowledge and beliefs stored in the personal dynamic memory of an agent. The frame-based meaning representation of both the meaning and the knowledge of an agent allows for the seamless integration of the language processing and the personal dynamic memory. This way, the result of the language processing

can be easily added to the personal dynamic memory and can then be used as knowledge to interpret subsequent utterances. Moreover, the reasoning process becomes straightforward since the meaning representation corresponds to the query for the inference engine.

Conceptually, the research resulted in a definition of narratives as the chain of reasoning operators that led to a conclusion. This facilitates the interpretability of the narratives that are constructed by an agent given that the reasoning process can be traced back, making it possible to identify which beliefs led to an interpretation. This way, the human working together with the system can understand why the system made a certain decision. Moreover, the system could be used to identify which pieces of knowledge led to a certain narrative or to identify what beliefs are missing to interpret a situation in a certain way.

7.2.3 A monitoring system using Integrative Narrative Networks

A last achievement discussed in this thesis is the introduction of the integrative narrative networks (INN), which allow agents to monitor their own understanding process. The INNs model the human-like capability of reflecting on their own understanding. Humans are able to signal when something is not clear or well understood, and they can even, in some cases, identify where the misunderstanding occurred. Moreover, humans rely on multiple sources of knowledge during language understanding. The INNs reflect these capacities by combining narrative questions and answers that are raised by different sources of knowledge in one network. The narrative questions can be seen as the expectations that a certain knowledge source raises, which can then be answered later on by information provided by either the same knowledge source or others. Full understanding or narrative closure is reached when all questions are solved and remaining questions can be signalled, indicating the knowledge gaps of an agent. The INNs make it possible to gain insights in the understanding process of an agent. For example, they allow the detection of the knowledge sources that played a crucial role during understanding or to indicate the knowledge sources that are not optimal. Moreover, quantifying the understanding process through the INNs becomes useful when systems are working together with humans since the systems can then indicate how certain they are of a given answer. Then, humans could provide the necessary feedback to integrate missing knowledge, thereby filling the knowledge gap.

Methodologically, the INN constitutes a data structure that gathers information

from the different knowledge sources that are required to understand an utterance. This is achieved through a monitoring system that detects changes in each of the different knowledge sources and adds them in the form of narratives questions or answers to the INNs. Lastly, the understanding process can be quantified by measuring the amount of answered and unanswered questions in the INN. I illustrated the use of the INNs by monitoring intelligent systems tackling one of dialogues from the visual dialogue task introduced in Chapter 4 and a recipe from a recipe understanding challenge. This way, I showed that this system can be applied on different tasks. Moreover, the integration of the monitoring system in the Babel architecture allows for future systems to easily make use of the INNs. Specifically, it enables the tracing and quantification of the understanding processes of intelligent agents operationalised through this framework.

7.3 Challenges and avenues for future research

While the systems in this thesis show that crucial aspects of human-like language understanding can indeed be modelled, several challenges remain, which for the most part relate to scalability of the approaches introduced in the thesis. This can be achieved in different areas.

First, the computational construction grammars introduced in Chapter 4 and 5 are written by hand. However, as discussed in Chapter 3, Nevens et al. (2022) and Doumen et al. (2023) provide initial results on how computational construction grammars can be learned. The learning operators that they developed are integrated in the Fluid Construction Grammar framework and are based on the two cognitive processes of intention reading and pattern finding (Tomasello, 2003, 2009). Intention reading is the process of reconstructing a meaning representation underlying linguistic utterances based on the provided feedback and pattern finding is responsible for finding generalisations over form-meaning pairs. While pattern finding allows to learn computational construction grammars from semantically annotated corpora (Doumen et al., 2023), the integration with intention reading enables the learning of grammars through communicative interactions in the world Nevens et al. (2022). In future work, these mechanisms could be extended to handle the visual dialogue datasets discussed in Chapter 4, thereby lifting the limitation of needing to write the grammar by hand. The challenge lies extending the learning operators that were developed for single-turn utterances to multi-turn dialogues. A first step would be to start from the semantically annotated datasets and investigate how the learning operators of

pattern finding need to be extended to handle dialogue settings. Specifically, the challenge lies in extending the operators to allow the learning of generalisations that take into account dialogue-specific information both on the form side and the meaning side. A second step would be to add the integration with intention reading, thereby removing the need for semantically annotated datasets. This way, the agent can learn a grammar from communicative interactions alone. In particular, this requires to integrate the introduced conversation memory in the intention reading process.

Second, an interesting avenue for scaling the Candide model introduced in Chapter 5 consists in endowing the agents with operators that allow them to learn their knowledge and beliefs as well as their grammar, thereby lifting the limitation of predefining these components. Again, this can be achieved by extending the developed mechanisms of intention reading and pattern finding. However, the challenge here is to allow for the build-up of knowledge and beliefs. In such an experiment the grammar and knowledge would be learned simultaneously as agents interact with the world. The agents start without any knowledge, extracting frames from the observations that they encounter and integrating them with their personal dynamic memory. The pattern finding mechanism could be used to find generalisations and learn mappings between the utterances and the frame-based meaning representations. Intention reading would be needed to construct the frame-based meaning representations. This way, the agents can build up their beliefs and linguistic knowledge through experience.

Another possibility for scalability relates to the neuro-symbolic procedural semantics approach introduced in Chapter 4. The methodology is applied on two visual dialogue tasks, showing that the methodology can achieve state-of-the-art results on multiple datasets. However, these datasets are diagnostic and are specifically designed to test the ability of models to solve tasks without using short cuts. The datasets are therefore specifically designed without biases that occur in the real world. The images in the datasets are synthetically generated and the questions are template-based. Although the performance of the methodology on these diagnostic datasets shows that it is adequate for solving this task, the question now remains how the methodology would scale to more realistic settings. For example, how does the methodology need to be extended to work on real-world images or natural language questions. In order to test this, other datasets that contain either real-world images such as GQA-VD (Zhang et al., 2022) or crowd-sourced natural language questions such as VisDial (Das et al., 2017) could be explored. Of course, several extensions to the model need

to be made. First, relating to the perception, the set of neural modules need to be extended to contain modules that can perceive the objects and attributes in these datasets. Secondly, relating to the reasoning and specifically reasoning over the dialogues, the conversation memory as well as the primitives need to be extended to work with other types of co-references. Lastly, going to larger and more diverse natural language datasets, the grammar will need be learned through intention reading and pattern finding.

Finally, both in the case of the neuro-symbolic approach to visual dialogue and the frame-based approach to narrative construction, the challenge is to define the set of primitive operations or frames that is needed to further scale the systems. Further research and experiments are needed to identify the minimal set of primitive operations to solve tasks. This set could then be used as the starting point for the intention reading process, to learn to compose and build meaning networks. As a first step towards defining the minimal set of frames, the FrameNet dataset (`https://framenet.icsi.berkeley.edu`) could be explored. In this project, researchers are attempting to describe all frames that occur in language. This is a highly non-trivial task. However, by analysing this set, it could be possible to find the minimal set of frames that is required to learn the frames over time. In any case, FrameNet gives an indication of which frames occur in languages, and the Candide model should be able to integrate or learn these frames. Other datasets that are semantically annotated such as the dataset introduced in Remijnse et al. (2022) should also be investigated. This dataset consists of utterances annotated with entities, roles and frames following the FrameNet convention. Moreover, the entities that play a certain role in a frame are grounded in the WikiData knowledge graph. A first experiment of the Candide model could be applied on this dataset with the aim of validating the methodology on a larger scale dataset. Intention reading and pattern finding experiments could then be set up to investigate how the beliefs in the form of frames are acquired.

In short, the main challenge in scaling up the systems in this thesis largely lies in enabling the agent to learn both the computational construction grammars as well as their knowledge and beliefs. Future research can explore the potential of the mechanisms of intention reading and pattern finding for this task.

7.4 Final remarks

In this thesis, I have introduced three systems that each model aspects of human-like language understanding such as grounding, meaning and reflection. Concretely, in Chapter 4, I have introduced a procedural semantic representation that is adequate to ground language in the environment and discourse. The neuro-symbolic execution of this representation leads to state-of-the-art results on the task of visual dialogue. Furthermore, in Chapter 5, I have introduced a frame-based procedural semantics that is able to ground language in the knowledge and beliefs of an agent. Here, I have demonstrated through examples how different agents can come to different interpretations. Finally, in Chapter 6, I have presented a monitoring system that can be used to quantify an agent's language understanding process, enabling the agent to reflect on its own language understanding. Each of these systems contributes in its own way to the growing literature on building more human-like language understanding systems. I discussed how future work needs to focus on integrating and scaling these systems. Together, the systems introduced in this thesis are a small but important step towards the ultimate goal of modelling systems capable of true human-like language understanding.

Bibliography

Abbo, G. A. and Belpaeme, T. (2023). Users' perspectives on value awareness in social robots. In Cakmak, M. and Leite, I., editors, *HRI2023, the 18th ACM/IEEE International Conference on Human-Robot Interaction*, pages 1–5, New York, NY, USA. Association for Computing Machinery.

Abdessaied, A., Bâce, M., and Bulling, A. (2022). Neuro-symbolic visual dialog. In Calzolari, N., Huang, C., Kim, H., Pustejovsky, J., Wanner, L., Choi, K., Ryu, P., Chen, H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 192–217. International Committee on Computational Linguistics.

Agarwal, S., Bui, T., Lee, J.-Y., Konstas, I., and Rieser, V. (2020). History for visual dialog: Do we really need it? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8182–8197. Association for Computational Linguistics.

Andreas, J., Bufe, J., Burkett, D., Chen, C., Clausman, J., Crawford, J., Crim, K., DeLoach, J., Dorner, L., Eisner, J., Fang, H., Guo, A., Hall, D., Hayes, K., Hill, K., Ho, D., Iwaszuk, W., Jha, S., Klein, D., Krishnamurthy, J., Lanman, T., Liang, P., Lin, C. H., Lintsbakh, I., McGovern, A., Nisnevich, A., Pauls, A., Petters, D., Read, B., Roth, D., Roy, S., Rusak, J., Short, B., Slomin, D., Snyder, B., Striplin, S., Su, Y., Tellman, Z., Thomson, S., Vorobev, A., Witoszko, I., Wolfe, J., Wray, A., Zhang, Y., and Zotov, A. (2020). Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016a). Learning to compose neural networks for question answering. In Knight, K., Nenkova, A., and Rambow, O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1545–1554. Association for Computational Linguistics.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016b). Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48. IEEE Computer Society.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. (2015). VQA: Visual question answering. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433. IEEE Computer Society.

Arps, D. and Petitjean, S. (2018). A parser for ltag and frame semantics. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Artzi, Y. and Zettlemoyer, L. (2011). Bootstrapping semantic parsers from conversations. In Barzilay, R. and Johnson, M., editors, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 421–432. Association for Computational Linguistics.

Badreddine, S., d'Avila Garcez, A., Serafini, L., and Spranger, M. (2022). Logic tensor networks. *Artificial Intelligence*, 303:103649.

Baker, C., Fillmore, C., and Lowe, J. (1998). The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT 2021)*, pages 610–623, New York, NY, USA. Association for Computing Machinery.

Bender, E. M. and Koller, A. (2020). Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198. Association for Computational Linguistics.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Bergen, B. and Chang, N. (2005). Embodied Construction Grammar in simulation-based language understanding. In Fried, M. and Östman, J.-O., editors, *Construction Grammars: Cognitive Grounding and Theoretical Extensions*, pages 147–190. John Benjamins, Amsterdam, Netherlands.

Berglund, L., Tong, M., Kaufmann, M., Balesni, M., Stickland, A. C., Korbak, T., and Evans, O. (2023). The reversal curse: Llms trained on "a is b" fail to learn "b is a". *arXiv preprint arXiv:2309.12288*.

Beuls, K. (2011). Construction sets and unmarked forms: A case study for Hungarian verbal agreement. In Steels, L., editor, *Design Patterns in Fluid Construction Grammar*, pages 237–264. John Benjamins, Amsterdam, Netherlands.

Beuls, K. (2017). An open-ended computational construction grammar for Spanish verb conjugation. *Constructions and Frames*, 9(2):278–301.

Beuls, K., Gerasymova, K., and van Trijp, R. (2010). Situated learning through the use of language games. In *Proceedings of the 19th Annual Machine Learning Conference of Belgium and The Netherlands (BeNeLearn)*, pages 1–6.

Beuls, K. and Höfer, S. (2011). Simulating the emergence of grammatical agreement in multi-agent language games. In Walsh, T., editor, *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pages 61–66, Palo Alto, CA, USA. AAAI Press.

Beuls, K. and Steels, L. (2013). Agent-based models of strategies for the emergence and evolution of grammatical agreement. *PLOS ONE*, 8(3):e58960.

Beuls, K. and Van Eecke, P. (2023). Fluid Construction Grammar: State of the art and future outlook. In Bonial, C. and Tayyar Madabushi, H., editors, *Proceedings of the First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)*, pages 41–50.

Beuls, K. and Van Eecke, P. (2024). Construction grammar and artificial intelligence. In Fried, M. and Nikiforidou, K., editors, *The Cambridge Handbook of Construction Grammar*. Cambridge University Press, Cambridge, United Kingdom. Forthcoming.

Beuls, K., Van Eecke, P., and Cangalovic, V. S. (2021). A computational construction grammar approach to semantic frame extraction. *Linguistics Vanguard*, 7(1):20180015.

Beuls, K., van Trijp, R., and Wellens, P. (2012). Diagnostics and repairs in Fluid Construction Grammar. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 215–234. Springer, New York, NY, USA.

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N., and Turian, J. (2020a). Experience grounds language. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735. Association for Computational Linguistics.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. (2020b). Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Bladier, T., Kallmeyer, L., and Evang, K. (2023). Data-driven frame-semantic parsing with tree wrapping grammar. In *Proceedings of the 15th International Conference on Computational Semantics*, pages 223–232.

Blaha, L. M., Abrams, M., Bibyk, S. A., Bonial, C., Hartzler, B. M., Hsu, C. D., Khemlani, S., King, J., St Amant, R., Trafton, J. G., and Wong, R. (2022). Understanding is a process. *Frontiers in Systems Neuroscience*, 16:1–18.

Bleys, J. (2016). *Language strategies for the domain of colour*. Language Science Press, Berlin, Germany.

Bleys, J., Stadler, K., and De Beule, J. (2011). Search in linguistic processing. In Steels, L., editor, *Design Patterns in Fluid Construction Grammar*, pages 149–179. John Benjamins, Amsterdam, Netherlands.

Boas, H. C. and Sag, I. A. (2012). *Sign-Based Construction Grammar*. CSLI Publications/Center for the Study of Language and Information, Stanford, CA, USA.

Bonial, C., Donatelli, L., Lukin, S. M., Tratz, S., Artstein, R., Traum, D., and Voss, C. R. (2019). Augmenting abstract meaning representation for human-robot dialogue. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 199–210. Association for Computational Linguistics.

Bordes, A., Boureau, Y., and Weston, J. (2017). Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations (ICLR 2017)*, pages 1–15.

Bowman, S. R. and Dahl, G. E. (2021). What will it take to fix benchmarking in natural language understanding? In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2021)*, pages 4843–4855. Association for Computational Linguistics.

Bresnan, J. (1978). *A Realistic Transformational Grammar*. Cambridge, Mass.: MIT Press.

Bresnan, J. (1985). *The Mental Representation of Grammatical Relations*. MIT Press series on cognitive theory and mental representation. MIT Press.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H.-T., editors, *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 1877–1901, Red Hook, NY, USA. Curran Associates Inc.

Bruner, J. (1991). The narrative construction of reality. *Critical Inquiry*, 18(1):1–21.

Cai, Q. and Yates, A. (2013). Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433. Association for Computational Linguistics.

Carroll, N. (2007). Narrative closure. *Philosophical Studies.*, 135:1–15.

Chen, W., Ma, X., Wang, X., and Cohen, W. W. (2022). Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Cheng, J., Reddy, S., Saraswat, V., and Lapata, M. (2019). Learning an executable neural semantic parser. *Computational Linguistics*, 45(1):59–94.

Cheng, Z., Xie, T., Shi, P., Li, C., Nadkarni, R., Hu, Y., Xiong, C., Radev, D., Ostendorf, M., Zettlemoyer, L., Smith, N. A., and Yu, T. (2023). Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations*.

Cho, Y. and Kim, I. (2021). NMN-VD: A neural module network for visual dialog. *Sensors*, 21(3):931.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124.

Chomsky, N. (1993a). *Lectures on Government and Binding*. De Gruyter Mouton, Berlin, New York.

Chomsky, N. (1993b). A minimalist program for linguistic theory. *The view from Building 20: Essays in linguistics in honor of Sylvain Bromberger*.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2022). Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Collins, K. M., Wong, C., Feng, J., Wei, M., and Tenenbaum, J. (2022). Structured, flexible, and robust: benchmarking and improving large language models towards more human-like behavior in out-of-distribution reasoning tasks. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 44.

Cox, M. T. (2005). Metacognition in computation: A selected research review. *Artificial Intelligence*, 169(2):104–141.

Croft, W. (2001). *Radical construction grammar: Syntactic theory in typological perspective*. Oxford University Press, Oxford, United Kingdom.

Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J. M. F., Parikh, D., and Batra, D. (2017). Visual dialog. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1080–1089. IEEE Computer Society.

De Beule, J. (2012). A formal deconstruction of Fluid Construction Grammar. In Steels, L., editor, *Computational Issues in Fluid Construction Grammar*, volume 7249 of *Lecture Notes in Computer Science*, pages 215–238. Springer, Berlin, Germany.

De Haes, R. (2023). A benchmark for recipe understanding in autonomous agents. Master's thesis, Vrije Universiteit Brussel, Brussels, Belgium.

De Raedt, L., Kimmig, A., and Toivonen, H. (2007). Problog: A probabilistic prolog and its application in link discovery. In Veloso, M. M., editor, *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 2468–2473, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186. Association for Computational Linguistics.

Dong, L. and Lapata, M. (2016). Language to logical form with neural attention. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43. Association for Computational Linguistics.

Doumen, J., Beuls, K., and Van Eecke, P. (2023). Modelling language acquisition through syntactico-semantic pattern finding. In Vlachos, A. and Augenstein, I., editors, *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1317–1327. Association for Computational Linguistics.

Drozdov, A., Schärli, N., Akyürek, E., Scales, N., Song, X., Chen, X., Bousquet, O., and Zhou, D. (2022). Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*.

Du, Y., Liu, Z., Li, J., and Zhao, W. X. (2022). A survey of vision-language pre-trained models. In De Raedt, L., editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-ECAI 2022)*, pages 5436–5443. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Elazar, Y., Kassner, N., Ravfogel, S., Ravichander, A., Hovy, E., Schütze, H., and Goldberg, Y. (2021). Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Ettinger, A. (2020). What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Evans, R., Bošnjak, M., Buesing, L., Ellis, K., Pfau, D., Kohli, P., and Sergot, M. (2021). Making sense of raw input. *Artificial Intelligence*, 299:103521.

Feldman, J., Dodge, E., and Bryant, J. (2009). Embodied construction grammar. In Heine, B. and Narrog, H., editors, *The Oxford Handbook of Linguistic Analysis*, pages 121–146. Oxford University Press, Oxford, United Kingdom.

Fillmore, C. (1968). The case for case. In Bach, E. W. and Harms, R. T., editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart & Winston, New York, NY, USA.

Fillmore, C., Kay, P., and O'connor, M. (1988). Regularity and idiomaticity in grammatical constructions: The case of let alone. *Language*, 64(3):501–538.

Fillmore, C. J. (1976). Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280, pages 20–32.

Fillmore, C. J. (1988). The mechanisms of "construction grammar". In *Annual Meeting of the Berkeley Linguistics Society*, volume 14, pages 35–55.

Fillmore, C. J. and Baker, C. F. (2001). Frame semantics for text understanding. In *Proceedings of WordNet and Other Lexical Resources Workshop, NAACL*, volume 6.

Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in Linguistic Analysis*.

Frank, A., Krieger, H.-U., Xu, F., Uszkoreit, H., Crysmann, B., Jörg, B., and Schäfer, U. (2007). Question answering from structured knowledge sources. *Journal of Applied Logic*, 5(1):20–48.

Frank, M. C. (2023). Bridging the data gap between children and large language models. *PsyArXiv preprint*.

Gan, Z., Cheng, Y., Kholy, A. E., Li, L., Liu, J., and Gao, J. (2019). Multi-step reasoning via recurrent dual attention for visual dialog. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6463–6474. Association for Computational Linguistics.

Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., and Neubig, G. (2023). PAL: Program-aided language models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *International Conference on Machine Learning (ICML 2023)*, pages 10764–10799. PMLR.

Gendron, G., Bao, Q., Witbrock, M., and Dobbie, G. (2023). Large language models are not abstract reasoners. *arXiv preprint arXiv:2305.19555*.

Gerasymova, K. and Spranger, M. (2010). Acquisition of grammar in autonomous artificial systems. In Coelho, H., Studer, R., and Woolridge, M., editors, *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI-2010)*, pages 923–928.

Gerasymova, K. and Spranger, M. (2012). An experiment in temporal language learning. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 237–254. Springer, New York, NY, USA.

Goldberg, A. E. (1995). *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago, IL, USA.

Goldberg, A. E. (2003). Constructions: A new theoretical approach to language. *Trends in Cognitive Sciences*, 7(5):219–224.

Goldberg, A. E. (2006). *Constructions at work: The nature of generalization in language*. Oxford University Press, Oxford, United Kingdom.

Goldberg, A. E. (2019). *Explain me this*. Princeton University Press, Princeton, NJ, USA.

Gómez-Pérez, A. (1999). Evaluation of taxonomic knowledge in ontologies and knowledge bases.

Goodfellow, I., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations (ICLR 2015)*, pages 1–15.

Grice, P. (1967). Logic and conversation. In Grice, P., editor, *Studies in the Way of Words*, pages 41–58. Harvard University Press, Cambridge, MA, USA.

Guo, D., Xu, C., and Tao, D. (2019). Image-question-answer synergistic network for visual dialog. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10434–10443. IEEE Computer Society.

Gupta, T. and Kembhavi, A. (2023). Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask r-cnn. In Cucchiara, R., Matsushita, Y., Sebe, N., and Soatto, S., editors, *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969. IEEE Computer Society.

Hitzler, P. and Sarker, M. K. (2022). *Neuro-symbolic artificial intelligence: The state of the art*. Frontiers in Artificial Intelligence and Applications Vol 342. IOS Press, Amsterdam, Netherlands.

Hough, A. R. and Gluck, K. A. (2019). The understanding problem in cognitive science. *Advances in Cognitive Systems*, 8:13–32.

Hu, R., Andreas, J., Rohrbach, M., Darrell, T., and Saenko, K. (2017). Learning to reason: End-to-end module networks for visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 804–813. IEEE Computer Society.

Huang, B.-B., Zhang, G., and Sheu, P. C. Y. (2008). A natural language database interface based on a probabilistic context free grammar. In *Proceedings of the IEEE International workshop on Semantic Computing and Systems*, pages 155–162. IEEE.

Hudson, D. A. and Manning, C. D. (2018). Compositional attention networks for machine reasoning. In *6th International Conference on Learning Representations (ICLR 2018)*, pages 1–20.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and $<$0.5mb model size. *arXiv preprint arXiv:1602.07360*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456, Cambridge, MA, USA. PMLR.

Jain, U., Lazebnik, S., and Schwing, A. (2018). Two can play this game: Visual dialog with discriminative question generation and answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5754–5763. IEEE Computer Society.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. (2017a). CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910. IEEE Computer Society.

Johnson, J., Hariharan, B., van der Maaten, L., Hoffman, J., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. (2017b). Inferring and executing programs for visual reasoning. In Cucchiara, R., Matsushita, Y., Sebe, N., and Soatto, S., editors, *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2989–2998. IEEE Computer Society.

Johnson-Laird, P. N. (1977). Procedural semantics. *Cognition*, 5(3):189–214.

Joshi, A. K. and Schabes, Y. (1997). Tree-adjoining grammars. In *Handbook of Formal Languages: Volume 3 Beyond Words*, pages 69–123. Springer.

Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., and McHardy, R. (2023). Challenges and applications of large language models. *arXiv preprint:arXiv:2307.10169v1*.

Kahneman, D. (2011). *Thinking, fast and slow*. Farrar, Straus and Giroux, New York, NY, USA.

Kamp, H. and Reyle, U. (2013). *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, volume 42. Springer Science & Business Media.

Kanazawa, M. (2007). Parsing and generation as datalog queries. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 176–183. Association for Computational Linguistics.

Kang, G.-C., Lim, J., and Zhang, B.-T. (2019). Dual attention networks for visual reference resolution in visual dialog. 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 2024–2033. Association for Computational Linguistics.

Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., Krusche, S., Kutyniok, G., Michaeli, T., Nerdel, C., Pfeffer, J., Poquet, O., Sailer, M., Schmidt, A., Seidel, T., Stadler, M., Weller, J., Kuhn, J., and Kasneci, G. (2023). Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103:102274.

Kay, P. and Fillmore, C. (1999). Grammatical constructions and linguistic generalizations: The what's X doing Y? construction. *Language*, 75(1):1–33.

Kottur, S., Moura, J. M. F., Parikh, D., Batra, D., and Rohrbach, M. (2018). Visual coreference resolution in visual dialog using neural module networks. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *European Conference on Computer Vision (ECCV 2018)*, pages 153–169. Springer.

Kottur, S., Moura, J. M. F., Parikh, D., Batra, D., and Rohrbach, M. (2019). Clevr-dialog: A diagnostic dataset for multi-round reasoning in visual dialog. 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 582–595. Association for Computational Linguistics.

Krishnamurthy, J. and Kollar, T. (2013). Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics*, 1:193–206.

Krishnamurthy, J. and Mitchell, T. (2012). Weakly supervised training of semantic parsers. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 754–765.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic CCG grammars from logical form with higher-order unification.

In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1223–1233.

Lake, B. M. and Murphy, G. L. (2023). Word meaning in minds and machines. *Psychological review*, 130(2):401.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al. (2015). Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Lenat, D. B. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):33–38.

Li, M. and Moens, M.-F. (2021). Modeling coreference relations in visual dialog. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3306–3318. Association for Computational Linguistics.

Li, X. L., Kuncoro, A., Hoffmann, J., de Masson d'Autume, C., Blunsom, P., and Nematzadeh, A. (2022). A systematic investigation of commonsense knowledge in large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11838–11855. Association for Computational Linguistics.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. (2023). Code as policies: Language model programs for embodied control. In O'Malley, M. K., editor, *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE.

Lichte, T. and Kallmeyer, L. (2017). Tree-Adjoining Grammar: A tree-based constructionist grammar framework for natural language understanding. In *The 2017 AAAI Spring Symposium Series*, pages 205–212. AAAI Press.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.

Liu, Q., Kusner, M. J., and Blunsom, P. (2020). A survey on contextual embeddings. *arXiv preprint arXiv:2003.07278*.

Loetzsch, M., Wellens, P., De Beule, J., Bleys, J., and van Trijp, R. (2008). The Babel2 manual. Technical Report 01-08, AI-Memo.

Lu, J., Kannan, A., Yang, J., Parikh, D., and Batra, D. (2017). Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model. In Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 313–323, Red Hook, NY, USA. Curran Associates Inc.

Lu, S., Bigoulaeva, I., Sachdeva, R., Madabushi, H. T., and Gurevych, I. (2023). Are emergent abilities in large language models just in-context learning? *arXiv preprint arXiv:2309.01809*.

Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., and De Raedt, L. (2018). DeepProbLog: Neural probabilistic logic programming. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 3753–3760, Red Hook, NY, USA. Curran Associates Inc.

Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., and De Raedt, L. (2021). Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504.

Mao, J., Gan, C., Kohli, P., Tenenbaum, J. B., and Wu, J. (2019). The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *7th International Conference on Learning Representations (ICLR 2019)*.

Marques, T. and Beuls, K. (2016). Evaluation strategies for computational construction grammars. In Matsumoto, Y. and Prasad, R., editors, *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1137–1146. International Committee on Computational Linguistics.

Mascharka, D., Tran, P., Soklaski, R., and Majumdar, A. (2018). Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4942–4950. IEEE Computer Society.

Massiceti, D., Dokania, P., Siddharth, N., and Torr, P. (2018). Visual dialogue without vision or dialogue. In *Critiquing and Correcting Trends in Machine Learning Workshop: NeurIPS 2018*.

McFetridge, P., Popowich, F., and Fass, D. (1996). An analysis of compounds in HPSG (Head-driven Phrase Structure Grammar) for database queries. *Data & Knowledge Engineering*, 20(2):195–209.

McKenna, N., Li, T., Cheng, L., Hosseini, M. J., Johnson, M., and Steedman, M. (2023). Sources of hallucination by large language models on inference tasks. *arXiv preprint arXiv:2305.14552*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In *ICLR*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Burges, C. J., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, pages 1–9, Red Hook, NY, USA. Curran Associates Inc.

Minsky, M. (1974). A framework for representing knowledge. Technical report, Technical Report 306, MIT AI Laboratory, Cambridge, MA, USA.

Mitchell, M. (2019). Artificial intelligence hits the barrier of meaning. *Information*, 10:51.

Mitchell, M. and Krakauer, D. C. (2023). The debate over understanding in AI's large language models. *Proceedings of the National Academy of Sciences*, 120(13):e2215907120.

Montes, N. and Sierra, C. (2022). Synthesis and properties of optimally value-aligned normative systems. *Journal of Artificial Intelligence Research*, 74:1739–1774.

Nevens, J. (2022). *Representing and learning linguistic structures on the conceptual, morphosyntactic, and semantic level*. PhD thesis, Vrije Universiteit Brussel, Brussels: VUB Press.

Nevens, J., De Haes, R., Beuls, K., and Van Eecke, P. (2023). A benchmark for recipe understanding in artificial agents. In *33rd Meeting of Computational Linguistics in The Netherlands (CLIN 33)*.

Nevens, J., Doumen, J., Van Eecke, P., and Beuls, K. (2022). Language acquisition through intention reading and pattern finding. In Calzolari, N. and Huang, C.-R., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 15–25. International Committee on Computational Linguistics.

Nevens, J., Van Eecke, P., and Beuls, K. (2019a). Computational construction grammar for visual question answering. *Linguistics Vanguard*, 5(1):20180070.

Nevens, J., Van Eecke, P., and Beuls, K. (2019b). A practical guide to studying emergent communication through grounded language games. In *AISB 2019 Symposium on Language Learning for Artificial Agents*, pages 1–8. AISB.

Nevens, J., Van Eecke, P., and Beuls, K. (2020). From continuous observations to symbolic concepts: A discrimination-based strategy for grounded concept learning. *Frontiers in Robotics and AI*, 7(84).

Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 427–436. IEEE Computer Society.

Nilsson, N. (1984). Shakey the robot. Technical report, Technical Note No. 323, Artificial Intelligence Center, SRI International, Menlo Park, CA, USA.

Niu, Y., Zhang, H., Zhang, M., Zhang, J., Lu, Z., and Wen, J.-R. (2019). Recursive visual attention in visual dialog. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6672–6681. IEEE Computer Society.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Pasupat, P. and Liang, P. (2015). Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480. Association for Computational Linguistics.

Pauw, S. and Hilferty, J. (2012). The emergence of quantifiers. In Steels, L., editor, *Experiments in Cultural Language Evolution*, volume 3, pages 277–304. John Benjamins, Amsterdam, Netherlands.

Pauw, S. and Hilferty, J. (2016). Embodied cognitive semantics for quantification. *Belgian Journal of Linguistics*, 30(1):251–264.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global vectors for word representation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Pereira, F. C. N. and Warren, D. H. D. (1980). Definite clause grammars for language analysis—a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13(3):231–278.

Pinar Saygin, A., Cicekli, I., and Akman, V. (2000). Turing test: 50 years later. *Minds and machines*, 10(4):463–518.

Pine, J. M. and Lieven, E. V. M. (1997). Slot and frame patterns and the development of the determiner category. *Applied Psycholinguistics*, 18(2):123–138.

Pollard, C. and Sag, I. A. (1994). *Head-driven Phrase Structure Grammar*. University of Chicago Press, Chicago, IL, USA.

Porzel, R. (2010). *Contextual computing: models and applications*. Cognitive Technologies. Springer, Heidelberg, Germany.

Porzel, R. (2021). On formalizing narratives. In Righetti, G., De Giorgis, S., Hedblom, M. M., and Kutz, O., editors, *Proceedings of the Joint Ontology Workshops 2021*, pages 1–8, Aachen, Germany. CEUR.

Porzel, R. and Malaka, R. (2004). A task-based approach for ontology evaluation. In Buitelaar, P., Handschuh, S., and Magini, B., editors, *ECAI Workshop on Ontology Learning and Population*, pages 1–6. CEUR.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training. https://openai.com/research/language-unsupervised.

Raji, D., Denton, E., Bender, E. M., Hanna, A., and Paullada, A. (2021). Ai and the everything in the whole wide world benchmark. In Vanschoren, J. and Yeung, S., editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, pages 1–17, Red Hook, NY, USA. Curran Associates Inc.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Reddy, S., Lapata, M., and Steedman, M. (2014). Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

Remijnse, L., Vossen, P., Fokkens, A., and Titarsolej, S. (2022). Introducing Frege to Fillmore: A FrameNet dataset that captures both sense and reference. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 39–50. European Language Resources Association.

Ruis, L. E., Khan, A., Biderman, S., Hooker, S., Rocktäschel, T., and Grefenstette, E. (2023). Large language models are not zero-shot communicators. *arXiv preprint arXiv:2210.14986v1*.

Russell, S. and Norvig, P. (2009). *Artificial intelligence: A modern approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition.

Sag, I. A. (2012). Sign-based construction grammar: An informal synopsis. In Boas, H. C. and Sag, I. A., editors, *Sign-based construction grammar*, pages 69–202. CSLI Publications/Center for the Study of Language and Information, Stanford, CA, USA.

Sallam, M. (2023). ChatGPT utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. *Healthcare*, 11(6):1–20.

Schank, R. C. (1983). *Dynamic memory: A theory of reminding and learning in computers and people*. Cambridge University Press.

Schwartz, I., Yu, S., Hazan, T., and Schwing, A. G. (2019). Factor graph attention. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2039–2048. IEEE Computer Society.

Seo, P. H., Lehrmann, A., Han, B., and Sigal, L. (2017). Visual reference resolution using attention memory for visual dialog. In Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 3722–3732, Red Hook, NY, USA. Curran Associates Inc.

Serban, I., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. In Singh, S. and Markovitch, S., editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3295–3301. AAAI Press.

Shah, M. A., Mehri, S., and Srinivasan, T. (2020). Reasoning over history: Context aware visual dialog. In *Proceedings of the First International Workshop on Natural Language Processing Beyond Text*, pages 75–83. Association for Computational Linguistics.

Shanahan, M. (2022). Talking about large language models. *arXiv preprint arXiv:2212.03551*.

Shin, R. and Van Durme, B. (2021). Few-shot semantic parsing with language models trained on code. *arXiv preprint arXiv:2112.08696*.

Sperber, D. and Wilson, D. (1986). *Relevance: Communication and cognition*. Harvard University Press, Cambridge, MA, USA.

Spranger, M. (2015). Incremental grounded language learning in robot-robot interactions: Examples from spatial language. In *2015 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pages 196–201.

Spranger, M. (2016). *The evolution of grounded spatial language*. Language Science Press, Berlin, Germany.

Spranger, M. (2017). Usage-based grounded construction learning: A computational model. In *The 2017 AAAI Spring Symposium Series*, pages 245–250. AAAI Press.

Spranger, M., Pauw, S., Loetzsch, M., and Steels, L. (2012). Open-ended procedural semantics. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 153–172. Springer, New York, NY, USA.

Spranger, M. and Steels, L. (2015). Co-acquisition of syntax and semantics: an investigation in spatial language. In Yang, Q. and Wooldridge, M., editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 1909–1915, Palo Alto, CA, USA. AAAI Press.

Steedman, M. (1987). Combinatory grammars and parasitic gaps. *Natural Language & Linguistic Theory*, 5(3):403–439.

Steedman, M. J. and Johnson-Laird, P. N. (1978). A programmatic theory of linguistic performance. In Campbell, R. N. and Smith, P. T., editors, *Recent Advances in the Psychology of Language: Formal and Experimental Approaches*, pages 171–192, New York, NY, USA. Springer.

Steels, L. (1995). A self-organizing spatial vocabulary. *Artificial Life*, 2(3):319–332.

Steels, L. (2004). Constructivist development of grounded construction grammar. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 9–16.

Steels, L., editor (2011). *Design patterns in Fluid Construction Grammar*. John Benjamins, Amsterdam, Netherlands.

Steels, L., editor (2012). *Experiments in cultural language evolution*. John Benjamins, Amsterdam, Netherlands.

Steels, L. (2017). Basics of Fluid Construction Grammar. *Constructions and Frames*, 9(2):178–225.

Steels, L. (2020). Personal dynamic memories are necessary to deal with meaning and understanding in human-centric AI. In *Proceedings of the First International Workshop on New Foundations for Human-Centered Artificial Intelligence (NeHuAI@ECAI)*, pages 11–16.

Steels, L. (2022a). Towards meaningful human-centric AI. In Steels, L., editor, *Foundations for meaning and understanding in human-centric AI*, pages 5–28. Venice International University, Venice, Italy.

Steels, L. (2023). Conceptual foundations for human-centric AI. In Chetouani, M., Dignum, V., Lukowicz, P., and Sierra, C., editors, *Human-Centered Artificial Intelligence (ACAI 2021)*, pages 8–35, Cham, Switzerland. Springer.

Steels, L. and De Beule, J. (2006). Unify and merge in Fluid Construction Grammar. In *International Workshop on Emergence and Evolution of Linguistic Communication (EELC 2006)*, pages 197–223.

Steels, L. and Loetzsch, M. (2010). Babel: A tool for running experiments on the evolution of language. In Nolfi, S. and Mirolli, M., editors, *Evolution of Communication and Language in Embodied Agents*, pages 307–313. Springer, Berlin, Germany.

Steels, L. and Van Eecke, P. (2018). Insight grammar learning using pro-and anti-unification. *Transactions on Human-Robot interaction*.

Steels, L. and Verheyen, L. (2022). Quantifying the contribution of different knowledge sources in narrative-based text understanding. In *Benelux Conference on Artificial Intelligence and Belgian Dutch Conference on Machine Learning (BNAIC/BeNeLearn)*.

Steels, L., Verheyen, L., and van Trijp, R. (2022a). An experiment in measuring understanding. In *Workshop on semantic techniques for narrative-based understanding: Workshop at IJCAI-ECAI 2022*, pages 36–42.

Steels, L., Verheyen, L., and van Trijp, R. (2022b). An experiment in measuring understanding. In Schlobach, S., Pérez-Ortiz, M., and Tielman, M., editors, *HHAI2022: Augmenting Human Intellect. Proceedings of the First International Conference on Hybrid Human-Artificial Intelligence*, pages 241–242. Frontiers in Artifical Intelligence and Applications.

Steels, L., Verheyen, L., and van Trijp, R. (Under Review). Integrative narrative networks. *Journal of Artificial Intelligence Research*.

Steels, L. e. (2022b). *Foundations for Meaning and Understanding in Human-centric AI*. Venice International University, Venice, Italy.

Subramanian, S., Narasimhan, M., Khangaonkar, K., Yang, K., Nagrani, A., Schmid, C., Zeng, A., Darrell, T., and Klein, D. (2023). Modular visual question answering via code generation. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 747–761. Association for Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 3104–3112, Red Hook, NY, USA. Curran Associates Inc.

Thomason, J., Padmakumar, A., Sinapov, J., Walker, N., Jiang, Y., Yedidsion, H., Hart, J., Stone, P., and Mooney, R. J. (2020). Jointly improving parsing and perception for natural language commands through human-robot dialog. *Journal of Artificial Intelligence Research*, 67:327–374.

Titus, L. M. (Forthcoming). Does ChatGPT have semantic understanding? a problem with the statistics-of-occurence strategy. *Cognitive Systems Research*.

Tomasello, M. (1995). Joint attention as social cognition. In Moore, C. and Dunham, P. J., editors, *Joint attention: Its origins and role in development*, chapter 6, pages 103–130. Psychology Press.

Tomasello, M. (2003). *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press, Harvard, MA, USA.

Tomasello, M. (2009). The usage-based theory of language acquisition. In *The Cambridge handbook of child language*, pages 69–87. Cambridge University Press, Cambridge, United Kingdom.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Tulving, E. (1972). Episodic and semantic memory. In Tulving, E. and Donaldson, W., editors, *Organization of memory*. Academic Press.

Turing, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, LIX(236):433–460.

Ungerer, T. and Hartmann, S. (2023). *Constructionist Approaches: Past, Present, Future*. Elements in Construction Grammar. Cambridge University Press, Cambridge, United Kingdom.

Van den Broeck, W. (2008). Constraint based compositional semantics. In Smith, A. D. M., Smith, K., and Ferrer i Cancho, R., editors, *Proceedings of the 7th International Conference on the Evolution of Language (EVOLANG7)*, pages 338–345. World Scientific.

Van Eecke, P. (2017). Robust processing of the dutch verb phrase. *Constructions and Frames*, 9(2):226–250.

Van Eecke, P. (2018). *Generalisation and specialisation operators for computational construction grammar and their application in evolutionary linguistics Research*. PhD thesis, Vrije Universiteit Brussel, Brussels: VUB Press.

Van Eecke, P. and Beuls, K. (2017). Meta-layer problem solving for computational construction grammar. In *The 2017 AAAI Spring Symposium Series*, pages 258–265. AAAI Press.

Van Eecke, P. and Beuls, K. (2018). Exploring the creative potential of computational construction grammar. *Zeitschrift für Anglistik und Amerikanistik*, 66(3):341–355.

Van Eecke, P., Nevens, J., and Beuls, K. (2022). Neural heuristics for scaling constructional language processing. *Journal of Language Modelling*, 10(2):287–314.

Van Eecke, P., Verheyen, L., Willaert, T., and Beuls, K. (2023a). The Candide model: How narratives emerge where observations meet beliefs. In Akoury, N., Clark, E., Iyyer, M., Chaturvedi, S., Brahman, F., and Chandu, K., editors, *Proceedings of the 5th Workshop on Narrative Understanding (WNU)*, pages 48–57. Association for Computational Linguistics.

Van Eecke, P., Verheyen, L., Willaert, T., and Beuls, K. (2023b). The Candide model: How narratives emerge where observations meet beliefs. In *33rd Meeting of Computational Linguistics in The Netherlands (CLIN 33)*.

van Trijp, R. (2011). A design pattern for argument structure constructions. In Steels, L., editor, *Design Patterns in Fluid Construction Grammar*, pages 115–145. John Benjamins, Amsterdam, Netherlands.

van Trijp, R. (2015). Towards bidirectional processing models of sign language: A constructional approach in fluid construction grammar. In Airenti, G., Bara, B. G., and Sandini, G., editors, *Proceedings of the EuroAsianPacific Joint Conference on Cognitive Science*, pages 668–673. University of Torino.

van Trijp, R. (2017). A computational construction grammar for English. In *The 2017 AAAI Spring Symposium Series*, pages 266–273. AAAI Press.

van Trijp, R. (2024). Different constructional approaches in practice: A comparative study. In Fried, M. and Nikiforidou, K., editors, *The Cambridge Handbook of Construction Grammar*. Cambridge University Press, Cambridge, United Kingdom. Forthcoming.

van Trijp, R., Beuls, K., and Van Eecke, P. (2022). The FCG Editor: An innovative environment for engineering computational construction grammars. *PLOS ONE*, 17(6):e0269708.

van Trijp, R. and Steels, L. (2012). Multilevel alignment maintains language systematicity. *Advances in Complex Systems*, 15(3–4):1250039.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Verheyen, L. (2023). The Candide model: A model for human-like, narrative-based language understanding. In *10th European Starting AI Researchers' Symposium (STAIRS)*.

Verheyen, L., Botoko Ekila, J., Nevens, J., Beuls, K., and Van Eecke, P. (Under review). Neuro-symbolic procedural semantics for explainable visual dialogue.

Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2022a). Hybrid procedural semantics for visual dialogue. In *32nd Meeting of Computational Linguistics in The Netherlands (CLIN 32)*.

Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2022b). Hybrid procedural semantics for visual dialogue: An interactive web demonstration. In *Workshop on semantic techniques for narrative-based understanding: Workshop at IJCAI-ECAI 2022*, pages 48–52.

Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2023). Neuro-symbolic procedural semantics for reasoning-intensive visual dialogue

tasks. In Gal, K., Nowé, A., Nalepa, G. J., Fairstein, R., and Rădulescu, R., editors, *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI 2023)*, pages 2419–2426, Amsterdam, Netherlands. IOS Press.

Verheyen, L., Van Eecke, P., and Beuls, K. (2021). Computational construction grammar and procedural semantics for visual dialogue. In *11th International Conference on Construction Grammar*.

Verheyen, L., Van Eecke, P., and Beuls, K. (2022c). Computational construction grammar and procedural semantics for visual dialog. In *Abstract from Linguists' Day - Taaldag - Journée Linguistique LSB*.

Voltaire (1759). *Candide ou l'Optimisme*. Gabriel Cramer, Genève, Switzerland.

Vrandečić, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Wang, Y., Joty, S., Lyu, M. R., King, I., Xiong, C., and Hoi, S. C. H. (2020). VD-BERT: A unified vision and dialog transformer with BERT. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3325–3338. Association for Computational Linguistics.

Warren, D. H. D. and Pereira, F. C. N. (1982). An efficient easily adaptable system for interpreting natural language queries. *American Journal of Computational Linguistics*, 8(3-4):110–122.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022a). Finetuned language models are zero-shot learners. In *10th International Conference on Learning Representations (ICLR 2022)*, pages 1–46. OpenReview.net.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022b). Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2022c). Chain-of-thought prompting elicits reasoning in large language models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pages 24824–24837. Curran Associates, Inc.

Weizenbaum, J. (1983). ELIZA - A computer program for the study of natural language communication between man and machine (reprint). *Commun. ACM*, 26(1):23–28.

Willaert, T., Banisch, S., Van Eecke, P., and Beuls, K. (2022). Tracking causal relations in the news: Data, tools, and models for the analysis of argumentative statements in online media. *Digital Scholarship in the Humanities*, 37(4). fqab107.

Willaert, T., Van Eecke, P., Beuls, K., and Steels, L. (2020). Building social media observatories for monitoring online opinion dynamics. *Social Media + Society*, 6(2).

Willaert, T., Van Eecke, P., Van Soest, J., and Beuls, K. (2021). An opinion facilitator for online news media. *Frontiers in Big Data*, 4:1–10.

Winograd, T. (1971). *Procedures as a representation for data in a computer program for understanding natural language*. PhD thesis.

Winograd, T. (1972). Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

Winograd, T. (1975). Frame representations and the declarative/procedural controversy. In *Representation and understanding*, pages 185–210. Elsevier.

Winograd, T. (2001). Architectures for context. *Human–Computer Interaction*, 16(2-4):401–419.

Wong, Y. W. and Mooney, R. (2007). Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 960–967. Association for Computational Linguistics.

Woods, W. A. (1968). Procedural semantics for a question-answering machine. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, pages 457–471, New York, NY, USA.

Woods, W. A., Kaplan, R. M., and Webber, B. L. (1972). The lunar sciences natural language information system: Final report. Technical report, BBN Report.

Wu, Q., Wang, P., Shen, C., Reid, I., and van den Hengel, A. (2018). Are you talking to me?: Reasoned visual dialog generation through adversarial learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6106–6115. IEEE Computer Society.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. (2019). Detectron2. `https://github.com/facebookresearch/detectron2`.

Wu, Z., Peng, H., and Smith, N. A. (2021). Infusing finetuning with semantic dependencies. *Transactions of the Association for Computational Linguistics*, 9:226–242.

Yahya, M., Berberich, K., Elbassuoni, S., Ramanath, M., Tresp, V., and Weikum, G. (2012). Natural language questions for the web of data. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 379–390.

Yang, T., Zha, Z.-J., and Zhang, H. (2019). Making history matter: History-advantage sequence training for visual dialog. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2561–2569. IEEE Computer Society.

Yi, K., Gan, C., Li, Y., Kohli, P., Wu, J., Torralba, A., and Tenenbaum, J. B. (2020). CLEVRER: Collision events for video representation and reasoning. In Song, D., Cho, K., and White, M., editors, *8th International Conference on Learning Representations (ICLR 2020)*, pages 1–19.

Yi, K., Wu, J., Gan, C., Torralba, A., Kohli, P., and Tenenbaum, J. (2018). Neural-symbolic VQA: Disentangling reasoning from vision and language understanding. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 1031–1042, Red Hook, NY, USA. Curran Associates Inc.

Zelle, J. M. and Mooney, R. J. (1996). Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, pages 1050–1055. AAAI Press.

Zettlemoyer, L. and Collins, M. (2005). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In Bacchus, F. and Jaakkola, T., editors, *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 658–666. AUAI Press.

Zettlemoyer, L. and Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687. Association for Computational Linguistics.

Zhang, H., Ghosh, S., Heck, L., Walsh, S., Zhang, J., Zhang, J., and Kuo, C.-C. J. (2019). Generative visual dialogue system via weighted likelihood estimation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pages 1025–1031. AAAI Press.

Zhang, Y., Jiang, M., and Zhao, Q. (2022). New datasets and models for contextual reasoning in visual dialog. In Avidan, S., Brostow, G., Farinella, G. M., and Hasnner, T., editors, *European Conference on Computer Vision (ECCV 2022)*, pages 434–451. Springer.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.-Y., and Wen, J.-R. (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zheng, Z., Wang, W., Qi, S., and Zhu, S.-C. (2019). Reasoning visual dialogs with structural and partial observations. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6669–6678. IEEE Computer Society.

Zhong, V., Xiong, C., and Socher, R. (2017). Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Appendix A

List of Publications

The peer-reviewed papers and extended abstracts that I contributed to during my PhD are listed below. A full list of my publications, presentations and posters can be found at: `https://researchportal.vub.be/en/persons/lara-verheyen`.

Peer-reviewed papers:

- Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2023). Neuro-symbolic procedural semantics for reasoning-intensive visual dialogue tasks. In Gal, K., Nowé, A., Nalepa, G. J., Fairstein, R., and Rădulescu, R., editors, *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI 2023)*, pages 2419–2426, Amsterdam, Netherlands. IOS Press

- Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2022b). Hybrid procedural semantics for visual dialogue: An interactive web demonstration. In *Workshop on semantic techniques for narrative-based understanding: Workshop at IJCAI-ECAI 2022*, pages 48–52

- Van Eecke, P., Verheyen, L., Willaert, T., and Beuls, K. (2023a). The Candide model: How narratives emerge where observations meet beliefs. In Akoury, N., Clark, E., Iyyer, M., Chaturvedi, S., Brahman, F., and Chandu, K., editors, *Proceedings of the 5th Workshop on Narrative Understanding (WNU)*, pages 48–57. Association for Computational Linguistics

- Steels, L., Verheyen, L., and van Trijp, R. (2022a). An experiment in measuring understanding. In *Workshop on semantic techniques for narrative-based understanding: Workshop at IJCAI-ECAI 2022*, pages 36–42

- Steels, L., Verheyen, L., and van Trijp, R. (2022b). An experiment in measuring understanding. In Schlobach, S., Pérez-Ortiz, M., and Tielman, M., editors, *HHAI2022: Augmenting Human Intellect. Proceedings of the First International Conference on Hybrid Human-Artificial Intelligence*, pages 241–242. Frontiers in Artifical Intelligence and Applications

Peer-reviewed (extended) abstracts:

- Verheyen, L. (2023). The Candide model: A model for human-like, narrative-based language understanding. In *10th European Starting AI Researchers' Symposium (STAIRS)*
- Van Eecke, P., Verheyen, L., Willaert, T., and Beuls, K. (2023b). The Candide model: How narratives emerge where observations meet beliefs. In *33rd Meeting of Computational Linguistics in The Netherlands (CLIN 33)*
- Verheyen, L., Botoko Ekila, J., Nevens, J., Van Eecke, P., and Beuls, K. (2022a). Hybrid procedural semantics for visual dialogue. In *32nd Meeting of Computational Linguistics in The Netherlands (CLIN 32)*
- Steels, L. and Verheyen, L. (2022). Quantifying the contribution of different knowledge sources in narrative-based text understanding. In *Benelux Conference on Artificial Intelligence and Belgian Dutch Conference on Machine Learning (BNAIC/BeNeLearn)*
- Verheyen, L., Van Eecke, P., and Beuls, K. (2021). Computational construction grammar and procedural semantics for visual dialogue. In *11th International Conference on Construction Grammar*
- Verheyen, L., Van Eecke, P., and Beuls, K. (2022c). Computational construction grammar and procedural semantics for visual dialog. In *Abstract from Linguists' Day - Taaldag - Journée Linguistique LSB*