CONTEXTUALIZED SCENE MODELING USING BOLTZMANN MACHINES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

İLKER BOZCAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JULY 2018

Approval of the thesis:

CONTEXTUALIZED SCENE MODELING USING BOLTZMANN MACHINES

submitted by **İLKER BOZCAN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Halit Oğuztüzün _____
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Sinan Kalkan _____
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Assist. Prof. Dr. Emre Akbaş
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Sinan Kalkan
Computer Engineering Department, METU _____

Assist. Prof. Dr. Esra Kadıoğlu Ürtiş
Computer Eng. Dept., TOBB Univ. of Economics and Technology _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: İLKER BOZCAN

Signature :

ABSTRACT

CONTEXTUALIZED SCENE MODELING USING BOLTZMANN MACHINES

Bozcan, İlker
M.S., Department of Computer Engineering
Supervisor : Assoc. Prof. Dr. Sinan Kalkan

July 2018, 72 pages

Scene modeling is very crucial for robots that need to perceive, reason about and manipulate the objects in their environments. In this thesis, we propose a variant of Boltzmann Machines (BMs) for contextualized scene modeling. Although many computational models have been proposed for the problem, ours is the first to bring together objects, relations, and affordances in a highly-capable generative model. For this end, we introduce a hybrid version of BMs where relations and affordances are introduced with shared, tri-way connections. We evaluate our method in comparison with several baselines on missing or out-of-context object detection, relation estimation, and affordance estimation tasks. Moreover, we also illustrate scene generation capabilities of the model.

Keywords: Scene Modeling, Context, Knowledge Bases, Boltzmann Machines, Deep Learning

ÖZ

BOLTZMANN MAKİNELERİ KULLANARAK BAĞLAMSALLAŞMIŞ SAHNE MODELLEMESİ

Bozcan, İlker
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Sinan Kalkan

Temmuz 2018, 72 sayfa

Sahne modellemesi, çevredeki nesneleri algılayan, işleten ve nesneler hakkında çıkarsama yapan robotlar için son derece önemlidir. Bu tezde, bağlamsallaşmış sahne modellemesi için farklı bir Boltzmann Makinesi (BM) öneriyoruz. Bu konuda birçok çalışma olmasına rağmen, bizimkisi nesneleri, ilişkileri ve sağlarlıkları bir araya getiren, son derece yetenekli üretken bir modeldir. Bu amaçla, ilişkilerin ve sağlarlıkların paylaşıldığı, üçlü bağlantılar içeren melez bir BM sunduk. Üstelik, ilişki hesaplama ve modelleme çalışmaları için bir veri kümesi sunduk. Yöntemimizi, nesne bulmada, bağlam dışı nesne tespitinde, ilişki ve sağlarlık hesaplamada ölçü alınan birkaç modelle kıyasladık. Dahası, modelimizin üretkenliğini gösteren örnekler sunduk.

Anahtar Kelimeler: Sahne Modellemesi, Bağlam, Bilgi Tabanları, Boltzmann Makineleri, Derin Öğrenme

Dedicated to my graceful mother for her love and hearty smile from the Heaven.

ACKNOWLEDGMENTS

Firstly, I wish to express my gratitude to my thesis advisor Sinan Kalkan for his friendship, guidance, advice, criticism, encouragements and toleration throughout my education. He is the one who teaches me how to walk on the academia.

I would like to thank Irmak Dogan for her kind friendship. She helped me whenever I felt stuck about anything and made me feel not alone.

I would like to thank my lab friends Cemal Aker, Ezgi Ekiz, Osman Tursun, Negin Bagherzadi, Yunus Terzioğlu, Sera Büyükgöz and Fatih Can Kurnaz. They accompanied me throughout my last two years.

I would like to thank my friends from ODTÜ DKSK for funny days. Specially, I appreciate to meet Bekir Hekim and Akın Aytekin. Sport is now my living way instead of just a hobby thanks to them.

I would like to special thanks Gökay Yurdakul, Eren Diler, Yunus Emre Demir, Muhammet Karaçalık and Hatice Karaçalık for their sincere friendship more than ten years.

I would like to thank Zülfiye Arğın for her encouragements and accompanying. She is the one behind the curtain. Without her support, I would not find my way.

Last but not least, I would like to thank my family for being always with me. They taught me how to take a stand for my values and how to devote myself to someone.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ALGORITHMS

ALGORITHMS

LIST OF ABBREVIATIONS

BM	Boltzmann Machine
RBM	Restricted Boltzmann Machine
DBM	Deep Boltzmann Machine
HBM	Hybrid Boltzmann Machine
COSMO	A Contextualized Scene Model with Triway Boltzmann Machines
OWL	Web Ontology Language
KB	Knowledge Base
MLN	Markov Logic Network
MRF	Markov Random Field
RQL	Robot Query Language

CHAPTER 1

INTRODUCTION

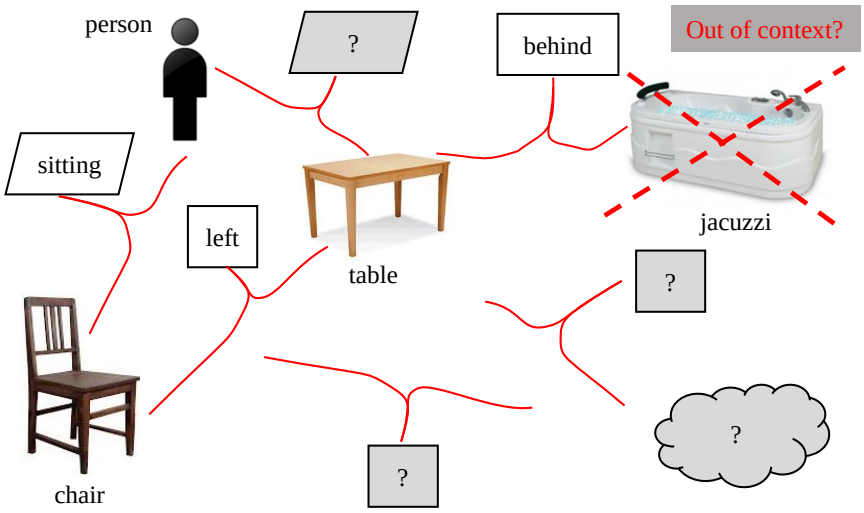
Having a model, i.e., a representation, of the environment is crucial for artificial and biological cognitive agents. A scene model is a representation that allows a robot to reason about the scene and what it contains in an efficient manner. For example, as shown in Figure 1.1(a), using a scene model, a robot can check (i) whether there is a certain object in the scene and if yes, where it is; (ii) whether an object is in the right-place in the scene; or (iii) whether there is something not expected or redundant in the scene.

A contextualized scene model, on the other hand, integrates the context of the scene in representing the environment and making inferences about what it contains. This is critical since it has been noted that context plays critical role in perception, reasoning, communication and action [4, 62]. Context helps these processes in resolving ambiguities, rectifying mispredictions, filtering irrelevant details, and adapting planning. These processes and problems are closely linked to a scene model, and therefore, scene models should contextualize what they represent.

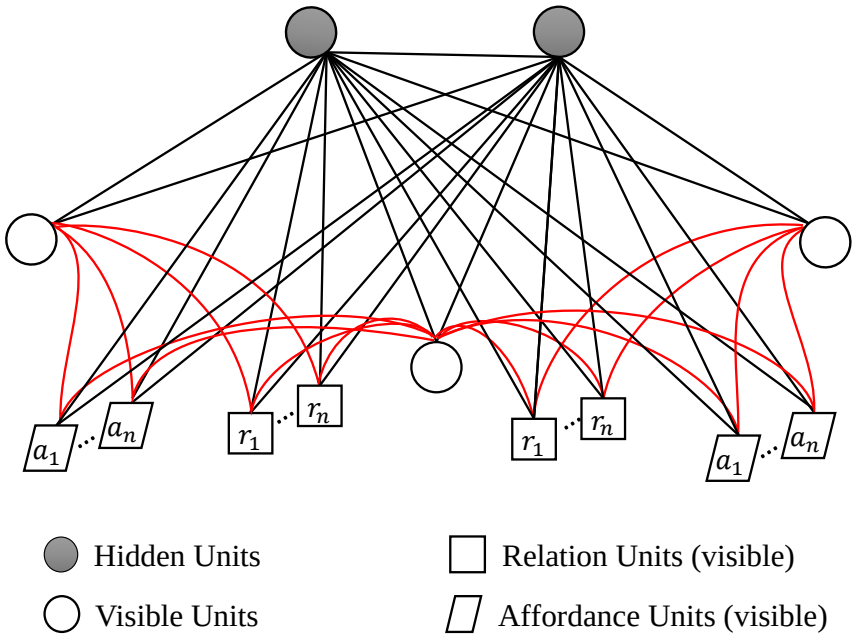
1.1 Problem Definition

In this thesis, we address the following problems:

- How can we model the environment in contextual manner? Even though the environment is modeled in different ways in the literature (by using graph chains, first order logic, Markov random fields etc.), our model is the first to use (and



(a)



(b)

Figure 1.1: (a) Example problems for which scene models help robots (given some incomplete or wrong observations from the environment). With our model, we can answer questions marked in gray. (b) An overview of COSMO, our hybrid tri-way Boltzmann Machine, where the tri-way edges are shown in red, as a contextualized scene model. [Best viewed in color]

adapt) Boltzmann Machines for scene modeling, which not only represents objects or relations between objects in the scene but also affordances of objects. Moreover, being generative, it is able to complete any missing information in the scene and make predictions given any information that may be available.

- What kind of problems in robotics can be solved by using a contextualized scene model? Robots should be aware of the context that they are in in order to operate autonomously. They can fill missing or irrelevant parts in the scene by using contextual information. Relations among objects can be varied according to the context and objects can afford different usabilities for different types of contexts.

- Can we use the contextual scene model to increase performance of object detectors? Object detectors can fail due to enlightenment conditions, object occlusion etc. Contextual scene models can correct misclassification of objects in the scene by using contextual information in it.

1.2 Contributions

The main contributions of the thesis are as follows:

- **Investigation of different types of knowledge bases for robotic tasks:** Extending robot knowledge by using external knowledge sources (bases) is an important issue in robotics. There are several knowledge bases. All of these methods rely on two main paradigms: logic-based or graph-based. Before developing a model that represent robotic knowledge, we review several robotic knowledge bases openly available in the literature.

  This part of the thesis has been published as a technical report [8].

- **Deep Boltzmann Machines for Scene Modeling:** To the best of our knowledge, ours is the first to use Deep Boltzmann Machines (DBM) [46] for scene modeling. With DBM, we introduce a generative scene model which incorporates objects, spatial relations and affordances. In order to be able to model

concepts like relations and affordances that require tri-way connections, we adapt and extend DBM by (i) combining together General BM [1] with higher-order BM [50], and (ii) introducing weight-sharing in order to have the same concepts of relations and affordances between different sets of variables.

This part of the thesis has been published as a paper [9].

The contributions presented in this thesis are disseminated in the following studies:

- İlker Bozcan and Sinan Kalkan. COSMO: Contextualized Scene Modeling with Boltzmann Machines. Robotics and Autonomous Systems (RAS) special issue on Semantic Policy and Action Representations for Autonomous Robots (SPAR), 2018. (Submitted)
- İlker Bozcan, Yağmur Oymak, İdil Zeynep Alemdar, and Sinan Kalkan. What is (missing or wrong) in the scene? A Hybrid Deep Boltzmann Machine For Contextualized Scene Modeling. International Conference on Robotics and Automation (ICRA), 2018.
- İlker Bozcan, Yağmur Oymak, İdil Zeynep Alemdar, and Sinan Kalkan. Sahnedeki (eksik ya da fazla) ne? Bağlamsallaşmış Sahne Modellemesi için Melez, Derin bir Boltzmann Makinesi. Türkiye Robot Bilim Konferansı (ToRK), 2018.
- İlker Bozcan and Sinan Kalkan. Combining Different Knowledge-bases into a Single Partially-grounded Robotic Knowledge-base. Technical Report No: METU-CENG-TR-2017-02, Department of Computer Engineering, Middle East Technical University, 2017.

The author has contributed to the following, which however are not presented in this thesis:

- Fethiye Irmak Doğan*, İlker Bozcan*, Mehmet Çelik and Sinan Kalkan. CINet: A Learning Based Approach to Incremental Context Modeling in Robots. In-

* Equal contribution

ternational Conference on Intelligent Robots and Systems (IROS), 2018. (Accepted)

1.3 Organization

In Section 2, the work related to our study is examined by focusing on different types of scene modeling, relation and affordance estimation studies. Moreover, we give background about Boltzmann Machines and variants.

In Section 3, we investigate and compare different types of robotic knowledge bases that are openly available in the literature. This part of thesis is used for determining necessities of proposed scene model. The content of this chapter has been published in [8].

In Section 4, our proposed contextual scene model based on Boltzmann Machines (COSMO) is presented. The structure of the model and the learning algorithm is described. The context of this chapter has been published in [9].

In Section 5, the experimental results of COSMO performing on several tasks including missing object estimation, out-of-context object detection, relation estimation and affordance estimation tasks are presented. We compare COSMO with baseline methods. We conduct experiments on real robot data and the NYU dataset.

In Section 6, the thesis is concluded by summarizing the proposed model. Moreover, the limitations of the models and future work are discussed.

CHAPTER 2

RELATED WORK AND BACKGROUND

In this chapter, we review related work on scene modeling, relation estimation and affordance prediction. In addition, we give background about Boltzmann Machines and their variants.

The content of this chapter has been published in [9] and [10].

2.1 Scene Modeling

Table 2.1: Comparison with existing studies on Scene Modeling.

Study	Main Method	Generative?	Relations?	Affordances?	Explicit Context?
[25]	DB processes	Y	N	N	N
[2]	MRF	Y	Y	N	N
[6]	scene graphs	N	N	Y	N
[12]	MRF	Y	N	Y	N
[35]	PL	N	Y	Y	N
[11]	MRF	Y	Y	Y	N
[44]	chain-graphs	N	Y	N	Y
[22]	PL	N	Y	N	Y
[30]	BN	N	Y	N	Y
[43, 60]	LDA v.	Y	Y	N	Y
[31]	MRF	Y	Y	N	Y
[14]	LDA	Y	N	Y	Y
[56]	ontology	N	Y	Y	Y
[49]	ontology	N	Y	Y	Y
COSMO	BM	Y	Y	Y	Y

Scene modeling is an important problem in Computer Vision and Robotics. During the last decade, especially probabilistic methods or probabilistic graphical models

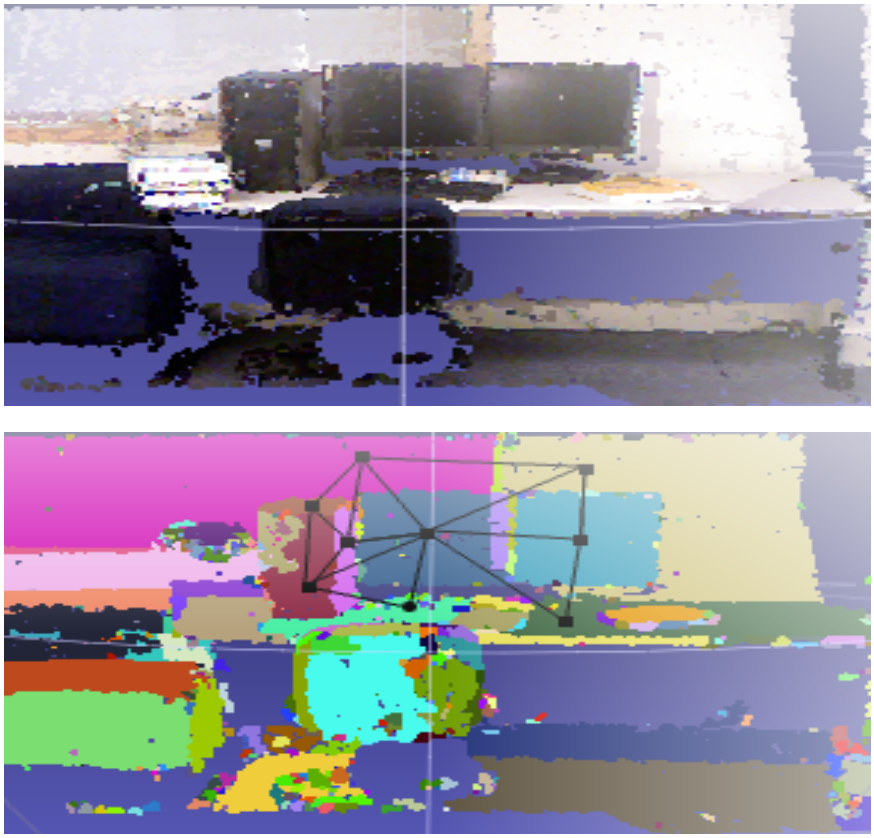


Figure 2.1: (Top) Point cloud of the scene. (Bottom) The segmentation of the point cloud. Black dots represents segments and spatially related segments are linked with black edges. (Figure source: [2])

such as Markov Random Fields (MRF) or Conditional Random Fields (CRF) [2, 12, 14, 31], Bayesian Networks (BN) [30, 51], Latent Dirichlet Allocation variants (LDA v.) [43, 60], Dirichlet and Beta (DB) processes [25], chain-graphs [44], predicate logic (PL) [22, 35], Scene Graphs [6], and ontologies [22, 49, 56] have been proposed for solving the problem.

In the study of Anand et al. [2], they segment the 3D point-cloud scenes, and predict the labels of segments. Each segment can have multiple classes that indicate attributes of segments such as object type, shape etc. MRF is used for scene modeling and each segment corresponds to node of the MRF graph (See Figure 2.1).

In the study of Celikkanat et al. [12], MRF is used for scene modeling. Each node in the MRF graph corresponds to one of three types of concepts including objects,

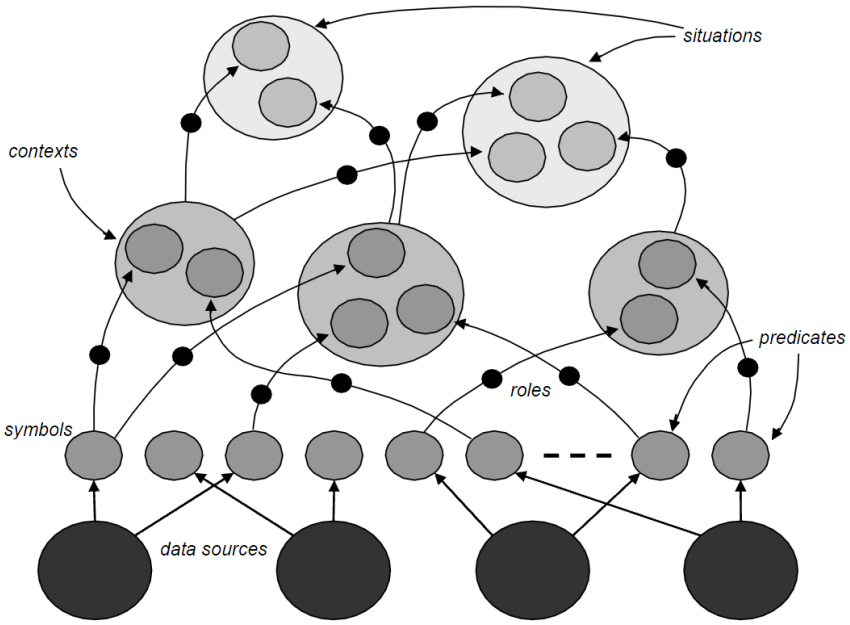


Figure 2.2: A schematic representation of the context assessment model proposed in [35] (Figure source: [35])

verbs and adjectives. Concept nodes can be connected to other nodes with weighted links. Weights indicate association between two nodes. In [14], context nodes are introduced. Concept nodes can be connected to context nodes with weighted edges. (See Figure 3.5)

In the study of Mastrogiovanni et al. [35], the Situation Definition Language (SDL) is introduced to representing and reasoning robotic knowledge in contextual hierarchy. Moreover, they introduce a predicate logic-based formalism to represent the predicates, contexts and situations. Hierarchical structure and proper level of abstraction allow to ease in reasoning (See Figure 2.2).

In the study of Bluementhal and Bruyninckx [6], a domain specific language (DSL) is proposed for scene graph based environment models. Proposed DSL is used for representing semantic knowledge about the scenes. The scene graph based world model consists of objects and relations among them. Relations are represented as actually directed acyclic graphs.

Lin et al. [31] represented as a scene in terms of objects and relations among them.

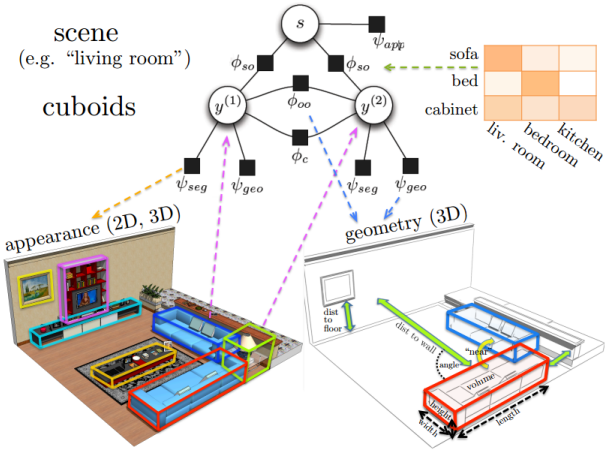


Figure 2.3: A demonstration of how CRF model can build on object cuboids in [31] (Figure source: [31])

Firstly, they rotate the point cloud of the indoor scene to canonical orientation in order to estimate the floor and wall planes. Then, they generate cuboids that encapsulate objects in the scene. By using Conditional Random Fields (CRF) for reasoning about the objects, their geometric properties and spatial relations (See Figure 2.3).

In the study of Li et al. [30], a context-aware framework based on Multi-Entity Bayesian Network (MEBN) is proposed. According to the authors, they prefer MEBN since it can provide reasoning under uncertainties. MEBN can represent repeated structures in BN. The proposed framework includes six logic based components: (a) Data processor that preprocesses sensor input data, (b) Semantic Mapper that parses the data coming from different data sources and fits them into proper format in order to keep in the ontology, (c) Ontology Model that keeps semantics of the data, (d) Rules Creator that enables operators to define knowledge processing rules for the ontology, (e) Context Reasoner that is able to deduce new knowledge according to information stored in the ontology and (f) Semantic Query that enables operators to query on the ontology to get values (See Figure 2.4).

In the study by Pronobis and Jensfelt [44], a layered structure is used to modeling objects and relations among them. They used chain-graph based conceptual map for probabilistic reasoning on the knowledge ontology (See Figure 2.5).

Among these studies, similar to ours, there are also models that explicitly integrate

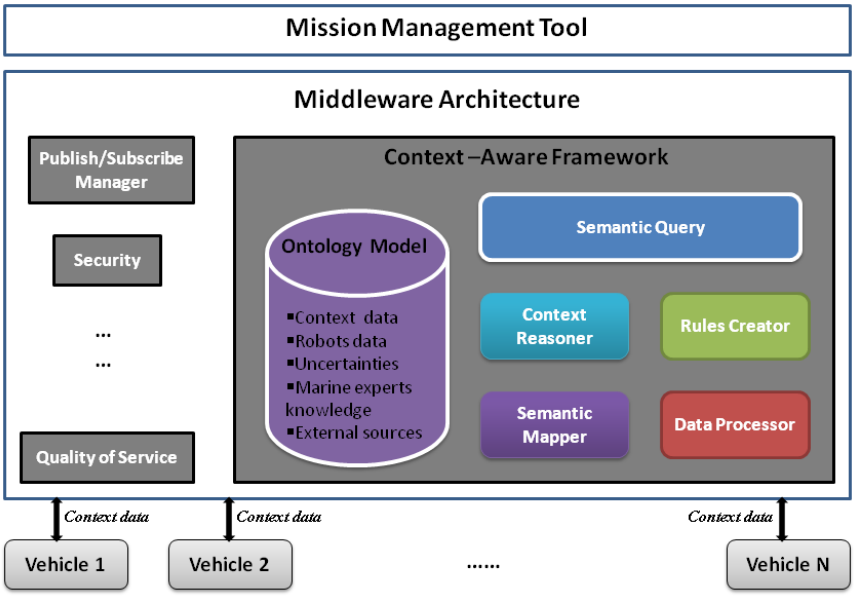


Figure 2.4: A demonstration of context-aware framework proposed in [30] (Figure source: [30])

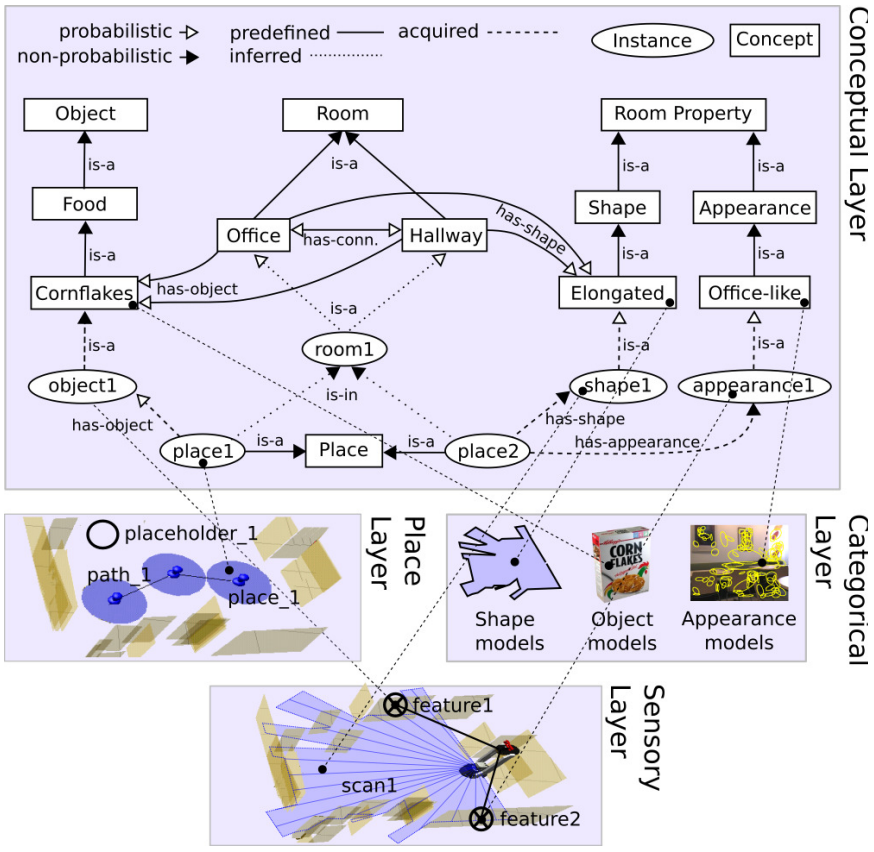


Figure 2.5: A demonstration of the layered structure proposed in [44]. The top layer represents conceptual layer. (Figure source: [44])

context into the model [14, 43, 60]. For example, Wang et al. [60] extend LDA to incorporate spatial positions between pixels in a local neighborhood in order to segment an image into semantically meaningful regions. Philbin et al. [43], on the other hand, include spatial arrangement between visual patches (i.e., words in LDA) to group similar images into a topic.

Among these, the work of Çelikkanat et al. [14] is the closest to ours. Çelikkanat et al. use object detections as visible variables and context as the latent variable in LDA. However, in their work, the main focus was on incremental learning of context nodes, and issues like spatial relations and generative abilities of the scene model were not considered.

2.2 Relation Estimation and Reasoning

Without loss of generality, we can broadly analyze relation estimation and reasoning studies in three main categories: The first category of methods use hand-crafted rules to determine whether a pre-determined set of spatial relations were present between objects in 2D or 3D, e.g., [55].

In the second category of methods, which use probabilistic graphical models such as Markov Random Fields [2, 11], Conditional Random Fields [31], Implicit Shape Models [36], and latent generative models [25], a probability distribution is modeled for relations between objects or entities. In these studies, Anand et al. [2] considered relations like "on-top" and "in-front" (and their symmetries); Celikkanat et al. [11], "left", "on", and "in-front" (and their symmetries); Lin et al. [31], "on-top", "close-to" relations; Meissner et al. [36], 6-DoF relations (rotation and translation) between objects; and, Joho et al. [25], an implicit model over local arrangements of objects is learned.

In the third category, relation estimation is formulated as a classification problem and solved using discriminative models, such as logistic regression [19], and deep learning [24]. The study by Guadarrama et al. [19] studied relations like "above", "behind", "close to", "inside of", "on", and "left" (and their symmetries), whereas

only two relations (“left”, “behind” - and their symmetries) are considered in [24].

In [48], a simple neural network is proposed for relational reasoning. They use plug-and-play structure to estimate relations among objects. In this work, objects mean activations of Convolutional Neural Network (CNN) feature maps, and they used Long-Short Term Memory (LSTM) network for question processing framework.

Existing efforts on modeling or estimating relations generally address the problem either for relations or relations and objects, and not consider related concepts such as affordances. Moreover, Boltzmann Machines have not been used for the problem in a scene modeling context.

2.3 Affordance Prediction

The concept of affordance, owing to Gibson [18], pertains to the actions that are provided by entities in the environment to the agents. With suitable formalisms for robotics studies [15], affordance-based models have been proposed for many important problems, such as manipulation [37], navigation [58], imitation learning [34], planning [59, 26], conceptualization [3, 26], – see [23, 63] for a review.

An important issue in affordance-based approaches is to be able to estimate the affordances of objects from visual input. For this end, support vector machines [28, 59], bayesian networks [38, 39], markov random fields [7, 12], and deep networks [16, 27, 41] have been widely used in the literature. However, affordance prediction is generally addressed independently from scene modeling tasks, and to the best of our knowledge, Boltzmann Machines have not been used for modeling affordances.

2.4 Boltzmann Machines

A Boltzmann Machine (BM) [1] is a stochastic, generative network. A BM can model the probability distribution of data, denoted by $\mathbf{v}$, with the help of hidden variables,

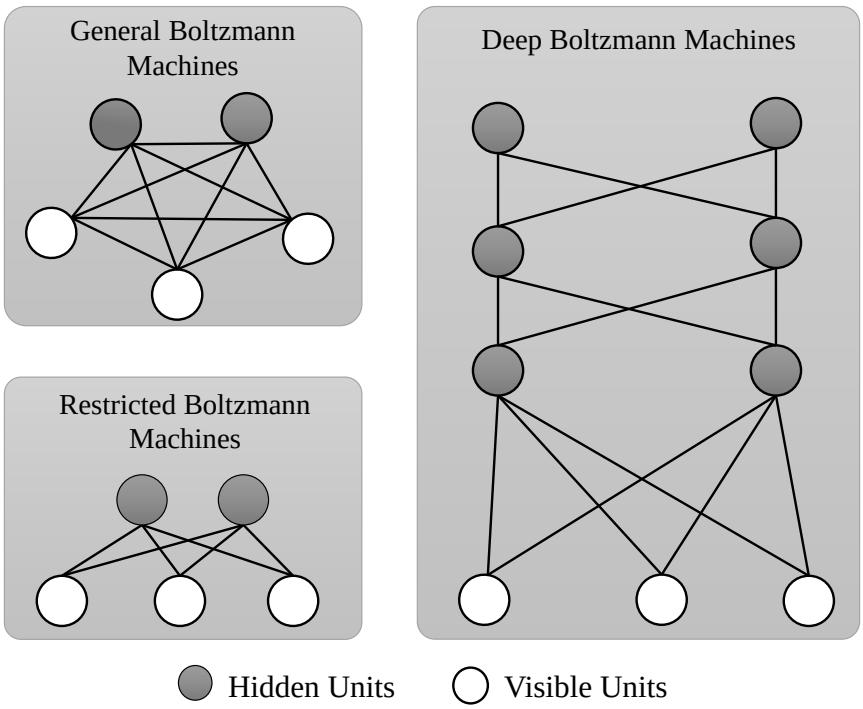


Figure 2.6: An illustration of different types of Boltzmann Machines (BM): General BM, Restricted BM and Deep BM. BM is stochastic network that is able to model probability distributions of high-dimensional data, and therefore, generate novel samples.

$\mathbf{h}$:

$$p(\mathbf{v}) = \sum_{\mathbf{h}} p(\mathbf{v}, \mathbf{h}). \tag{2.1}$$

In BMs, $\mathbf{v} = \{v_i\}_{i=1}^{V} \subset \{0,1\}^V$ is called the set of visible nodes, and $\mathbf{h} = \{h_i\}_{i=1}^{H} \subset \{0,1\}^H$ the hidden nodes. The visible nodes and the hidden nodes are connected to each other and how they are connected have led to different models – see Figure 2.6. In BMs, the connections are bi-directional; i.e., information can flow in both directions.

In a BM, one can talk about the compatibility, i.e., harmony, between two nodes connected by an edge. If, e.g., $n_i w_{ij} n_j$ is high for two nodes connected by an edge with weight w_{ij}, then nodes n_i and n_j are more compatible. However, generally, in BMs, the negative harmony, i.e., the energy of the network is used:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i<j} v_i w_{ij}^{vv} v_j - \sum_{i<j} h_i w_{ij}^{hh} h_j - \sum_{i<j} h_i w_{ij}^{hv} v_j, \tag{2.2}$$

where w^{vv}, w^{hh} and w^{hv} are the weights of the edges connecting visible-visible nodes, hidden-hidden nodes, and hidden-visible nodes respectively.

Being inspired from statistical mechanics, where systems with lower energies are favored more, BM associates the probability of being in a state (i.e., a configuration of $(\mathbf{v}, \mathbf{h})$) with the energy of the system as follows:

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h})), \tag{2.3}$$

where the normalizing term, also called the partition function, is defined as: $Z = \sum_{\mathbf{v'},\mathbf{h'}} E(\mathbf{v'}, \mathbf{h'})$. Notice that Z requires an integration over all possible states of the system, which is impractical to calculate in practice. Therefore, $p(\mathbf{v}, \mathbf{h})$ is iteratively learned by stochastically activating nodes in the network with probability based on the change in the energy of the system for an update:

$$p(n = 1) = \frac{1}{1 + e^{\Delta E_n / T}}, \tag{2.4}$$

where n is a visible or a hidden node; ΔE_n is the change in energy of the system if node n is turned on; and T is the temperature of the system, gradually decreased (annealed) to a low value. When T is high, the system can make radical updates that can even increase its energy; and when T is lowered, Equation 2.4 forces the network to make more deterministic updates, which lower the energy of the system.

2.4.1 Training a BM

Training a BM means that its weights are updated to model $p(\mathbf{v})$ as accurately as possible. Let us use $p^+(\mathbf{v})$ to denote the true probability of the data, and $p^-(\mathbf{v})$, the probability estimated by the model. Then, a BM is trained in order to minimize the dissimilarity, e.g., the Kullback-Leibler divergence, between $p^+(\mathbf{v})$ and $p^-(\mathbf{v})$. Taking gradient of the divergence with respect to a weight, w_{ij}, gives us the rate at which we should update it:

$$w_{ij} \leftarrow w_{ij} - \alpha(p^+_{ij} - p^-_{ij}), \tag{2.5}$$

where p^+_{ij} is the expected joint activation of nodes s_i and s_j when samples from the data are clamped on the visible units and the state of the network is updated accord-

ingly (called the positive phase); p_{ij}^{-} is the expected joint activation of nodes s_i and s_j when the network is randomly initialized and the state of the network is updated accordingly (called the negative phase); and α is a learning rate.

For training BMs, maximum Likelihood based methods are used [1, 40, 46]. However, since the partition function, Z, is intractable, directly computing p_{ij}^{+} and p_{ij}^{-} is not possible for general BMs. Therefore, Monte Carlo Markov Chain methods such as Gibbs sampling or Variational Inference methods such as mean field approaches are used to approximate p_{ij}^{+} and p_{ij}^{-}. Despite these methods, learning is still impractical owing to the connections within hidden and visible nodes, and potentially high number of hidden nodes.

2.4.2 Restricted Boltzmann Machines (RBM)

Since training is rather slow and limiting in BM, its restricted version (Restricted Boltzmann Machines) with only connections between hidden and visible nodes have been proposed [47] (see Figure 2.6). With these restrictions, the hidden units are conditionally independent given the visible units that makes learning practical. Overall energy formula in RBMs is updated as follows:

$$E = -\frac{1}{2}\sum_{i=1}^{|v|}\sum_{j=1}^{|h|} v_i h_j W_{ij}$$

Contrastive Divergence (CD) algorithm is used for training RBM. CD relies on approximation of gradient signal that indicated in 2.5. In this algorithm, the Markov chain is initialized with the input sample, then hidden units are sampled (the positive phase). Before finishing convergence of the Markov chain, hidden units are initialized with the values that recently sampled and the negative phase is conducted. Even if the CD cannot give exact values of weight's gradients, it can give information about the directions of gradients.

2.4.3 Higher Order Boltzmann Machines

Some problems require the edges to combine more than two nodes at once, which have led to the Higher-order Boltzmann Machines (HBM) [50]. With HBM, one can introduce edges of any order to link multiple nodes together (See Figure 2.7). Overall energy formula for HBM with tri-way edges is updated as follows:

$$E = -\frac{1}{2}\sum_{i=1}^{|s|}\sum_{j=1}^{|s|}\sum_{k=1}^{|s|} s_i s_j s_k W_{ijk}$$

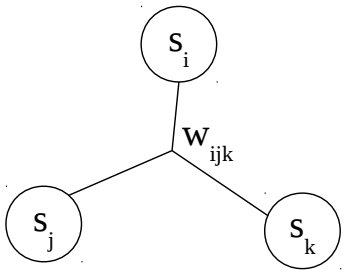


Figure 2.7: An illustration of Higher Order Boltzmann Machines. An edge connects more than two nodes.

2.4.4 Deep Boltzmann Machines

In a Deep Boltzmann Machine (DBM) [46], hidden nodes are stacked in layered structure. By using hidden layers, high level representations can be built from unlabeled data [46]. Unlike Deep Belief Nets, internal hidden layers get top-down and bottom-up signals. The overall energy formula for DBM is updated as follows:

$$E = -\frac{1}{2}\sum_{i=1}^{|v|}\sum_{j=1}^{|h^1|} v_i h_j^1 W_{ij}^1 - \frac{1}{2}\sum_{i=1}^{|h^1|}\sum_{j=1}^{|h^2|} h_i^1 h_j^2 W_{ij}^2 - \frac{1}{2}\sum_{i=1}^{|h^2|}\sum_{j=1}^{|h^3|} h_i^2 h_j^3 W_{ij}^3$$

2.5 Summary

Main points of related work can be described as follows:

- There is no effort on using Boltzmann Machines or its variants for contextualized scene models.
- Existence of objects in the scene, relations and affordances among them contribute to context.
- Structure of our model allows variety type of relations (spatial, temporal etc.) and concepts (physical objects, affordances, temporal entities etc.) to be used in model.
- Generative model can complete missing information about a scene according to context.

CHAPTER 3

COMBINING DIFFERENT KNOWLEDGE-BASES INTO A SINGLE PARTIALLY-GROUNDED ROBOTIC KNOWLEDGE-BASE

In this chapter, we investigate different types of robotic knowledge bases to determine necessities of contextual scene models.

The content of this chapter has been published as a technical report [8].

3.1 Knowledge-bases

Extending robot knowledge by using external knowledge sources (bases) is an important issue in robotics. There are several knowledge bases. All of these methods rely on two main paradigms: logic-based or graph-based. These knowledge representation methods have some advantages and disadvantages, and it is best if they are combined together.

3.1.1 KnowRob

KnowRob is the robot knowledge ontology proposed by Tenorth and Beetz [56] – see also Figure 3.1. In KnowRob, Knowledge is represented as formal logical statements using predefined templates. Widely used robotic concepts are ordered in hierarchical manner by using Web Ontology Language (OWL) syntax. There are variety of types of concepts which are related to robotics in the KnowRob ontology. For example:

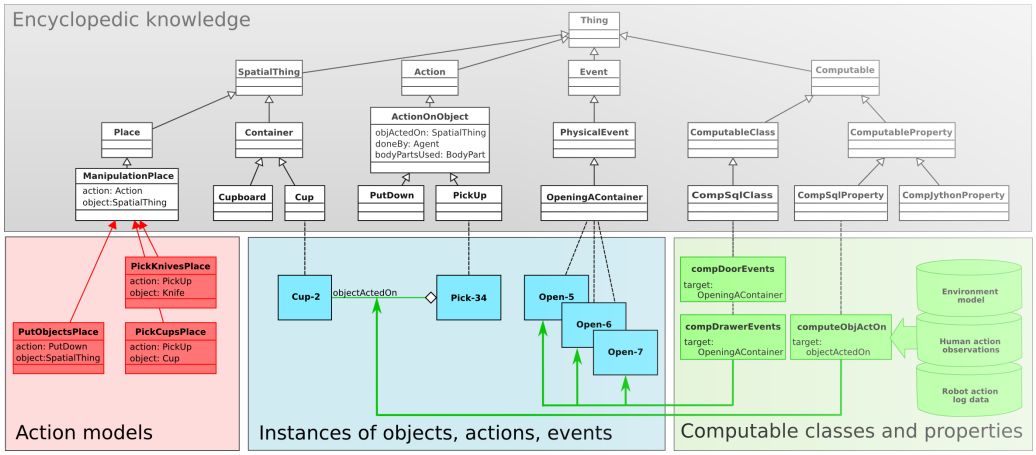


Figure 3.1: An overview of the KnowRob knowledge-base. (Figure source: [56])

- **Agent-Generic concepts** such as robot types (PR2, B21...), person.
- **Information Bearing concepts** such as semantic map of environment, floor plan.
- **Mathematical or Computational concepts** such as matrix, vectors, different types of algorithms.
- **Object type concepts** such as color, shape or intrinsic state (device is on/off etc.) of objects.
- **Spatial concepts** such as objects, object shapes, obstacles.
- **Temporal concepts** such as qualitative time of day (morning, afternoon).

A concept may have one or several parent concepts. Each concept may have several properties. For example, spatial object concepts may have spatial relations with other objects and event concepts may have `hasSubEvent` relation with some event concepts.

KnowRob allows making assertions/insertions using an OWL language:

```
owl_assert(manipActions, onProperty, actionType),
owl_assert(manipActions, hasValue, 'ActionOnObject'),
owl_assert(manipActions, type, 'restrictions'),
owl_assert(manipPosModel, type, 'ActionModel').
```

and make queries in the knowledge-base via the same language, e.g.:

```
owl_query(?OVEN, properPhysicalPartTypes, ?KNOB),
owl_query(?OVEN, type, 'Oven'),
owl_query(?KNOB, causes-Underspecified, ?HEATING),
owl_query(?HEATING, postEvents, ?BOILING),
owl_query(?BOILING, type, 'Boiling').
```

KnowRob has the advantage of compatibility with Open Cyc*, a comprehensive ontology including everyday common sense knowledge.

3.1.2 RoboBrain

RoboBrain is a knowledge engine introduced by Cornell and Stanford University in 2015 [49] – see Figure 3.2.

In contrast to KnowRob which represents knowledge in logical statements, RoboBrain represents knowledge as a directed graph. Each node represent concepts and edges represents relations between concepts. There can be several edges between two concepts. Knowledge graph can be extended by external sources if edges from the edge set of RoboBrain is used. Robot Query Language (RQL) is used for reasoning in the graph.

RQL can be used as interface between the agent and knowledge graph. The agent can query possible affordances of perceived object , get trajectory to apply desired affordance or get parameters of given trajectories etc. For instance, consider query q as "*affordances n:= fetch({name:n})→ ['HasAffordance'] → (v{src:'Affordance'})*". q returns possible object affordances for each object which is intereseted in the environment. Now, consider query q_2 as *trajectories a:= fetch({handle : a}) → ['HasParameters'] → (v{src: 'Affordance', type: 'Trajectory' })* . q_2 returns motion trajectories for each affordances.

* http://www.opencyc.org/

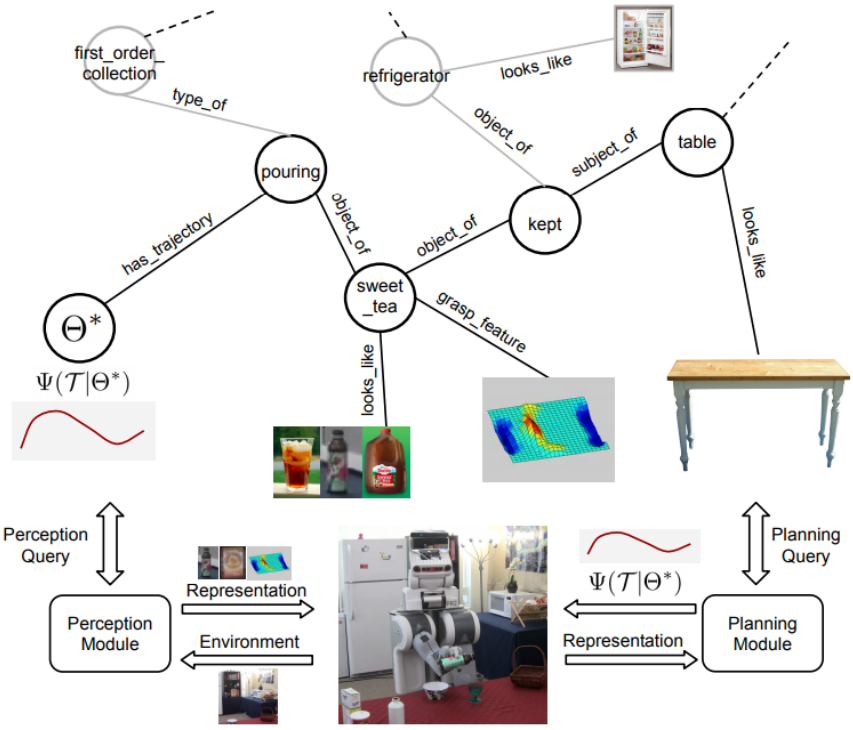


Figure 3.2: An illustration of the RoboBrain knowledge-base. (Figure source: [49])

3.1.3 Knowledge Base (KB) using Markov Logic Network (MLN)

A probabilistic graphical model for representing information between object-action concepts was introduced by Yuke Zhu, Alireza Fathi and Li Fei-Fei from CS Department of Stanford University [64] – see Figure 3.3. In this work, knowledge is represented as a Markov Logic Network, composed of Markov Random Field and First Order Logic.

In this model, as in RoboBrain, nodes correspond to concepts and edges correspond to relations. There can be only one undirected edge between two nodes unlike in RoboBrain.

Concepts have four categories: instances (objects), categories, affordances and attributes (weight, size or visual attributes).

There are two main sources used to populate the knowledge graph: web and dataset. Attribute concepts are collected from Amazon, eBay. Categorical concepts are col-

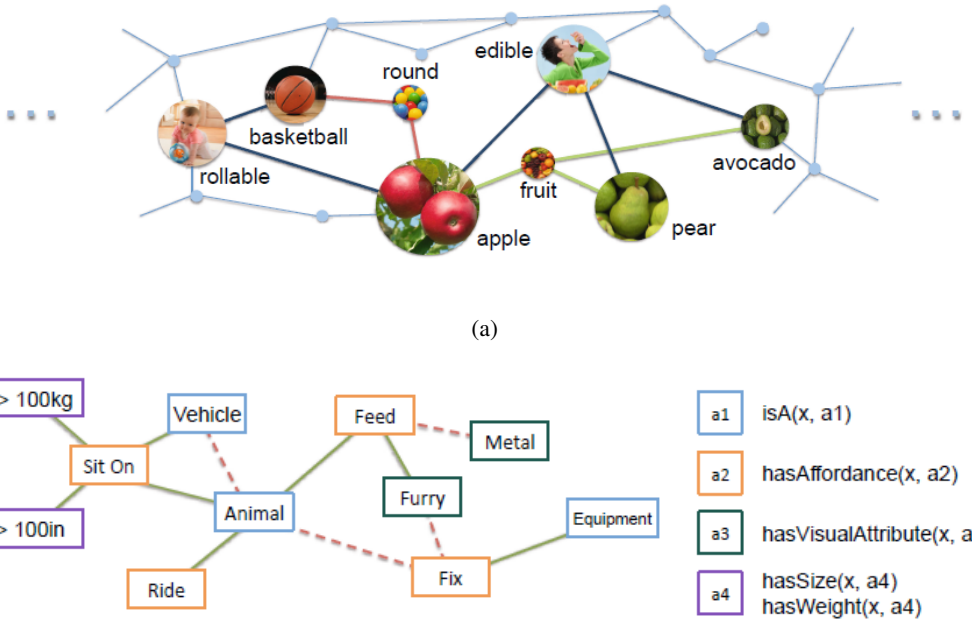


Figure 3.3: An illustration of the probabilistic graphical knowledge-base. (Figure source: [64])

lected from FreeBase, and WordNet. Affordance concepts are collected from Google Ngram or manually labeled. The dataset is used for creating visual attributes of objects, infer the affordances of objects.

Using RoboBrain, affordance prediction of a novel object, estimating human pose for an affordance and question answering by using knowledge graph is applicable.

3.1.4 ConceptNet

ConceptNet [54] is a free semantic network originated from Open Mind Common Sense project [52] – see Figure 3.4 for a snapshot. It includes several kinds of relations between concepts that might be helpful for robotic research. There are denser connection between concepts than projects mentioned above.

ConceptNet 5 is the most up-to-date version. It contains more than 2 million concepts and approximately 28 million total edges. ConceptNet provides weights for each edge determined by volunteer participants and knowledge sources on the web. Edges are

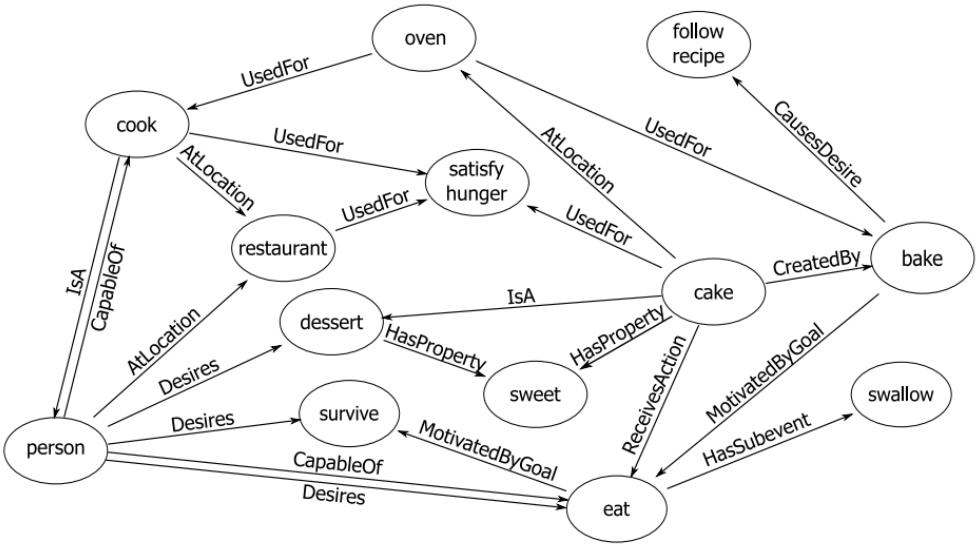


Figure 3.4: A portion of a cluster with related concepts from ConceptNet (Figure source: [54])

helpful while determining semantic similarity between concepts.

ConceptNet is also multilingual, it contains 10 core languages which is supported by ConceptNet 5. They are English, French, Italian, German, Spanish, Russian, Portuguese, Japanese, Dutch, Chinese. Furthermore, there are 68 languages that has vocabulary with more than 10.000 words.

There can be variety of types of edges in a ConceptNet graph. Each edge between two concepts indicates a relation between them. For instance, **relatedTo** is the most general relation between two concepts. It indicates positive relation between concepts. **isA** relation indicates subtype-supertype relation between two concepts. Full list of relations can be found in [54].

3.1.5 Probabilistic Concept Web

The probabilistic concept web [13] is a probabilistic graphical representation of robot knowledge based on Markov Random Fields. As in graph representation of knowledge sources, concepts are nodes of the graph and edges are relations between nodes – see Figure 3.5 for a snapshot.

Each concept is represented by their prototypes that is extracted from training set by using proposed method in [13]. Edge weights between concepts are determined by co-occurrences of concepts in robot-object interaction.

There are three types of **grounded concepts** according to robot sensor space in the probabilistic concept web:

- Noun concepts: {ball,box,cup,cylinder,plate,tool}.
- Adjective concepts: {edgy,round,noisy,silent,tall,short,hard,soft,thin,thick}
- Verb concepts: {drop, grasp, moveForward, moveBackward, moveLeft, moveRight, pushBackward, pushForward, pushLeft, pushRight, shake, throw}

In original work [13], three scenarios are used to test the concept web. Firstly, iCub perceives the object visually and guesses about its properties. Using Concept Web convergence, properties (adjective concepts such as hard, round etc.) of objects are refined and which behaviors (verb concepts shake, drop etc.) are applicable is determined.

Secondly, iCub is said to apply some behavior to given object. In this scenario, object concepts and verb concepts are activated. After convergence in the MRF graph, hidden concepts that indicate object properties are activated. Haptic and audio related properties of objects are determined without touching the object using by MRF graph.

In last scenario, iCub is said to apply some behavior to multiple object on the environment. Using Concept Web, iCub decides which object is the best one to apply desired behavior.

3.1.6 A Comparison of the Knowledge-bases

As stated in Table 3.1, Concept Web, RoboBrain and KnowRob are robotic projects. They try to represent knowledge in order for robots to manipulate objects better. KB using MLN is a knowledge representation model for extracting knowledge from im-

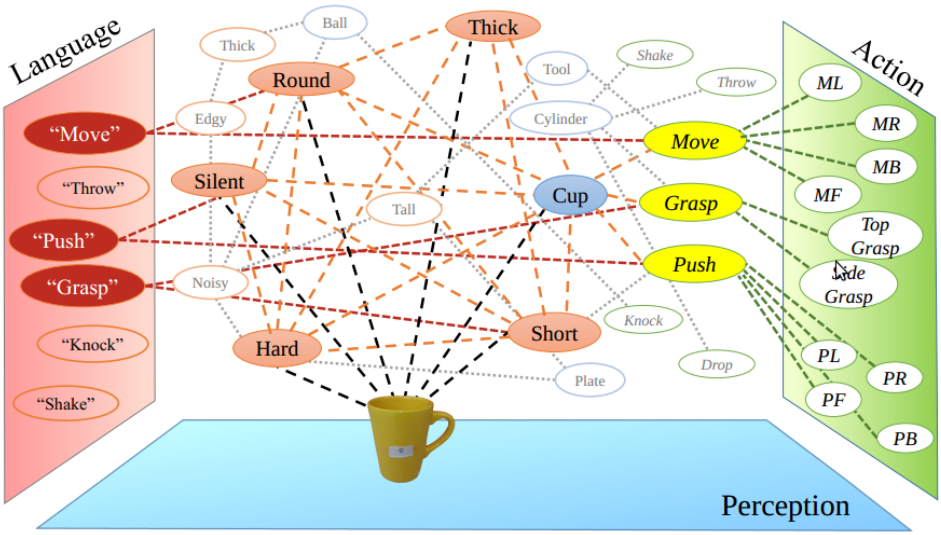


Figure 3.5: Schematic presentation of Scenario 1. iCub is presented with a cup, allowed to interact with it freely, and expected to predict the type and properties of the object, as well as what kind of behaviors can be applied on this object. The converged concept web is depicted. The action space and verb concepts are contoured with green, whereas blue and orange colors represent the noun and adjective categories for the object, respectively. The gray and smaller fonts show inactive concepts in the web, while bigger fonts and colored nodes represent activated concepts. There are other concepts and connections that are not shown for clarity. ML: Move-Left, MR: Move-Right, MB: Move-Backward, MF: Move-Forward, PL: Push-Left, PR: Push-Right, PB: Push-Backward, PF: Push-Forward. (Figure source: [13])

ages. ConceptNet is knowledge representation for general purpose in AI and Natural Language Processing.

Only in concept web, knowledge in the graph is fully grounded in robot's sensor level. Therefore, each concept has semantic according to robots sensor stimuli. In RoboBrain and KnowRob, concepts are also grounded but they include concepts that are extracted from web without grounding. Other two works are not concerned with grounding of concepts.

Concept Web includes concepts that are created by using haptic, audio and visual

Table 3.1: Comparison of knowledge representation models

	Probabilistic Concept Web [13]	**KB using MLN** [64]	**ConceptNet** [54]	**RoboBrain** [49]	**KnowRob** [56]
Field	Robotic	Image	AI	Robotic	Robotic
Grounded	Yes	No	No	Yes (Partially)	Yes (Partially)
Perception	Visual Haptic Audio	Image Weight&Size	No	Grasping features 3D clouds images, videos etc.	3D clouds
Representation	Graph (MRF)	Graph (MLN)	Directed graph	Directed graph	Formal statements
Knowledge Acquisition	Robot	Web, Dataset	Volunteers, Web	Web/Robot	Web (Cyc) Hand-coded Robot
Interface	None	Queries	Web/API	Queries (RQL)	Queries

sensors. Therefore, concept semantic has created according to the richest diversity of sensor data. ConceptNet does not include physical semantic for concepts. Each concept may interpreted by its relations with other concepts instead of sensor data. Other works, concept grounding is made using by visual data including images, 3D points, videos etc. KB using MLN is a web as a source for extracting size and weight of objects also.

Concept Web and KB using MLN are based on Markov Random Field Theory (MRF). In MRF, relations between nodes have weight represented as conditional probability of activating each other. This theory allows probabilistic inference of knowledge instead of deterministic ones. In ConceptNet, knowledge is represented by directed graph and edges have weight. However, unlike in Concept Web and KB using MLN, weights are determined hand-coded by using web source and volunteers. KnowRob represents knowledge as formal statements using predefined templates.

Knowledge acquisition is made by robot in Concept Web, RoboBrain and Knowrob. However, RoboBrain and KnowRob use web as knowledge source also. KB using MLN uses Stanford 40 Actions dataset[61] and web. ConceptNet knowledge is crawled by using web and participating volunteers.

There is no interface between knowledge representation model and agent in Concept Web. KB using MLN, RoboBrain and KnowRob use their own query languages as

interface and ConceptNet has Python API for accessing knowledge graph.

3.2 Merging Concept Web, ConceptNet and KnowRob

We transfered concept web [13] concepts into KnowRob ontology. Affordances for objects in KnowRob are not defined. Therefore, by using concept web, we assign affordances for common noun concepts. In addition, we assign probability of having some property given in adjectives to each noun concept.

Using ConceptNet 5, we include several super concept and subconcept of given concepts. We can extract information about where an object can store and which behaviors are applicable for this object. Attributes of *ball, box* and *cup* concepts in Concept Web are enriched by using ConceptNet 5.

Final properties of merged concepts are here, only some examples of common concepts in three knowledge representation models are shown below. Source of the knowledge is provided in parentheses:

- **ball**
  - is subconcept of *PortableObjects* (KnowRob)
  - may have spatial properties with objects such as being near of some object etc. (KnowRob)
  - has affordance of move, push, shake, throw (Concept Web)
  - has probability of some adjective properties: soft, noisy, short, thick, round (Concept Web)
  - can be stored in a toybox (ConceptNet)
  - have subtypes such as soccer ball, beach ball (ConceptNet)
  - is used for throwing, bouncing, playing a game (ConceptNet)
- **box**
  - is subconcept of *SpatialThingTypeByShape* (KnowRob)

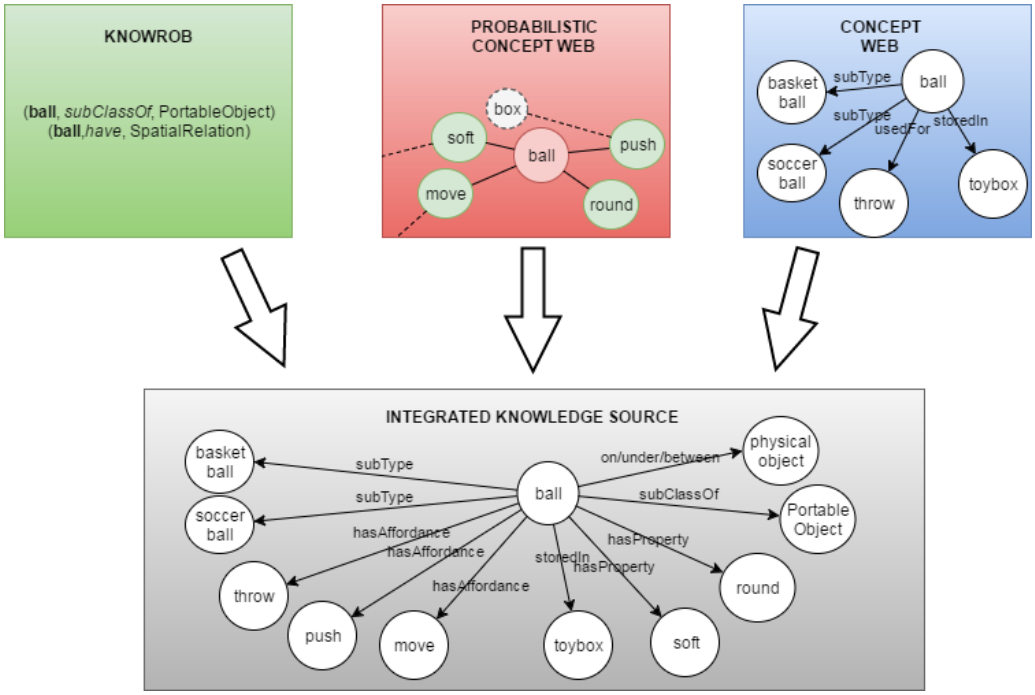


Figure 3.6: In this figure, integration of three knowledge sources is shown. As an example, attributes of "ball" concepts in different knowledge sources are merged and new "ball" concept is created in new knowledge source.

  - may have spatial properties with objects such as being near of some object etc. (KnowRob)
  - has affordance of drop, grasp, move, push, shake, throw (Concept Web)
  - has probability of some adjective properties: soft, noisy, short, thick, edgy (Concept Web)
  - can be stored in a any garage, ballpark (ConceptNet)
  - have subtypes such as ballot bax (ConceptNet)
  - is used for storing something in (ConceptNet)

- **cup**
  - is subconcept of *DrinkingVessel* (KnowRob)
  - may have spatial properties with objects such as being near of some object etc. (KnowRob)
  - is stored in some cup board (KnowRob)

  - has subconcept *mug* (KnowRob)
  - has affordance of grasp, move, push (Concept Web)
  - has probability of some adjective properties: hard, silent, short, thick, round (Concept Web)
  - can be stored in a table, shelf (ConceptNet)
  - have subtypes such as champagne cup (ConceptNet)
  - is used for drinking (ConceptNet)

- **cylinder**
  - is subconcept of *SpatialThingTypeByShape* (KnowRob)
  - may have spatial properties with objects such as being near of some object etc. (KnowRob)
  - has affordance of drop, grasp, move, push, shake, throw (Concept Web)
  - has probability of some adjective properties: hard, silent, tall, thin, round (Concept Web)

- **grasp**
  - is subconcept of *HoldingWithOneHand* (KnowRob)
  - has subconcept *PowerGrasp, IntermediateGrasp, PrecisionGrasp* (KnowRob)
  - is grounded on iCUB robot (Concept Web)

- **push**
  - is subconcept of *TransportationEvent* (KnowRob)
  - might be directional such as push left/right/forward/backward (Concept Web)
  - is grounded on iCUB robot (Concept Web)

CHAPTER 4

COSMO: A CONTEXTUALIZED SCENE MODEL WITH TRIWAY BM

In this chapter, we propose a novel model in order to modeling environment in contextual manner. Our model is based on Boltzmann Machines by extending them with triway edges within visible nodes. We name the model as "COSMO" which stands for "A Contextualized Scene Model With Triway BM".

The context of this chapter has been published in "International Conference on Robotics and Automation 2018" (ICRA 2018) [10] and is under revision for the Robotics and Autonomous Systems (RAS) journal special issue on Semantic Policy and Action Representations for Autonomous Robots (SPAR) [9].

4.1 The Model: COSMO

We extend and adapt DBM for contextualized scene modeling task. As shown in Figure 1.1, our model consists of visible (input) layer and hidden layer(s) corresponding to contextual representation of the scene.

We define a scene ($\mathbf{s} \in \mathcal{S}$) to be the tuple of an object vector ($\mathbf{o}$) describing objects currently visible to the robot (see Figure 4.1), the vector of the spatial relations ($\mathbf{r}$) between the objects, and the vector of affordances ($\mathbf{a}$). A visible node corresponds to an object, a relation or an affordance, and is set active (value 1) if corresponding object, affordance or relation exists in the scene (in this sense, $\mathbf{v} = (\mathbf{o}, \mathbf{r}, \mathbf{a})$). The hidden nodes ($\mathbf{h}$) then represent latent joint configurations of the visible nodes; i.e., they correspond to contextual information eminent in the scene.

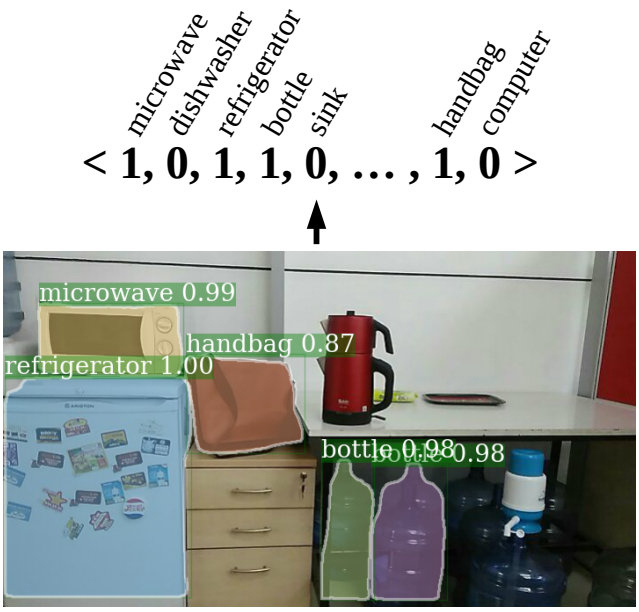


Figure 4.1: Object vector $\mathbf{o}$ describes existence of objects in the scene $\mathbf{s}$. (Figure source: [13])

Relation and affordance nodes link two object nodes with single *tri-way* edges, and visible nodes are fully connected to hidden nodes ($\mathbf{h}$). The overall energy of the hybrid BM then is updated as follows:

$$
\begin{aligned}
E(\mathbf{o}, \mathbf{h}, \mathbf{r}, \mathbf{a}) \quad = \quad & -\sum_{i<j} h_i w_{ij}^{ho} o_j \\
& -\sum_{i,j,k} w_{ijk}^{r} r_{ijk} o_j o_k - \sum_{i,j,k,l} w_{ij}^{rh} r_{ijk} h_l \\
& -\sum_{i,j,k} w_{ijk}^{a} a_{ijk} o_j o_k - \sum_{i,j,k,l} w_{ij}^{ah} a_{ijk} h_l,
\end{aligned}
\tag{4.1}
$$

where the new terms compared to the energy definition in Equation 2.2 are highlighted in red. r_{ijk} denotes spatial relation node with type i, between object nodes o_j and o_k. a_{ijk} is an affordance relation with type i, between objects nodes o_j and o_k. w_{ijk}^{r} is the weight of the tri-way edge connecting object nodes o_j, o_k and spatial relation node (visible) r_i; and, similarly, w_{ijk}^{a} is the weight of the tri-way edge connecting object nodes o_j, o_k and affordance node (visible) a_i.

4.1.1 Training and Inference

In order to make training faster, we dropped the connections between the hidden neurons and took gradient of the divergence $(KL(p^+(\mathbf{o},\mathbf{r},\mathbf{a}) \parallel p^-(\mathbf{o},\mathbf{r},\mathbf{a})))$ with respect to each type of weight as in Equation 2.5.

Let's denote G as a measure of the divergence $(KL(p^+(\mathbf{o},\mathbf{r},\mathbf{a}) \parallel p^-(\mathbf{o},\mathbf{r},\mathbf{a})))$:

$$G(\mathbf{o},\mathbf{r},\mathbf{a}) = KL(p^+(\mathbf{o},\mathbf{r},\mathbf{a}) \parallel p^-(\mathbf{o},\mathbf{r},\mathbf{a})) = \sum_{\alpha} p^+(\mathbf{o}_\alpha,\mathbf{r}_\alpha,\mathbf{a}_\alpha) \ln \frac{p^+(\mathbf{o}_\alpha,\mathbf{r}_\alpha,\mathbf{a}_\alpha)}{p^-(\mathbf{o}_\alpha,\mathbf{r}_\alpha,\mathbf{a}_\alpha)}, \tag{4.2}$$

where $p^+(\mathbf{o}_\alpha,\mathbf{r}_\alpha,\mathbf{a}_\alpha)$ is the probability of the visible nodes being in state α when the input is clamped to the visible units, and $p^-(\mathbf{o}_\alpha,\mathbf{r}_\alpha,\mathbf{a}_\alpha)$ is the corresponding probability when network runs freely. Note that state a is concatenation of states of object, relations and affordances nodes. For learning, we need to minimize G. Since G depends on weights we can take partial derivatives of G with respect to the all types of weights.

According to the new energy definition (Equation 4.2) and connections, probabilities of being active for visible and hidden units are given by:

$$p(o_i = 1 \mid \mathbf{o},\mathbf{h},\mathbf{r},\mathbf{a}) = \sigma\Big(\sum_j h_j w^{ho}_{ij} + \sum_{j,k} w^{r}_{ijk} r_{ijk} o_j + \sum_{j,k} w^{a}_{ijk} a_{ijk} o_j\Big), \tag{4.3}$$

$$p(h_l = 1 \mid \mathbf{o},\mathbf{r},\mathbf{a}) = \sigma\Big(\sum_i o_i w^{ho}_{il} + \sum_{i,j,k} r_{ijk} w^{rh}_{ij} + \sum_{i,j,k} a_{ijk} w^{ah}_{ij}\Big), \tag{4.4}$$

$$p(r_{ijk} = 1 \mid \mathbf{o},\mathbf{h}) = \sigma\Big(w^{r}_{ijk} o_j o_k + \sum_l w^{rh}_{ij} h_l\Big), \tag{4.5}$$

$$p(a_{ijk} = 1 \mid \mathbf{o},\mathbf{h}) = \sigma\Big(w^{a}_{ijk} o_j o_k + \sum_l w^{ah}_{ij} h_l\Big). \tag{4.6}$$

For training COSMO, in the positive phase, as usual, we clamp the visible units with the objects, the relations and the affordances between the objects and calculate p^+ for any edge in the network.

In the negative phase, firstly, object units are sampled with a two-step Gibbs sampling by using activation of hidden units only. In this way, initially, the model sees the environment as bag of objects by not considering relations and affordances. Then,

relation and affordance nodes are sampled by using hidden nodes (context) and recently sampled object nodes. We calculate p^- for any edge in the network with these two steps.

The overall algorithm is summarized in Algorithm 1.

At the end of the negative phase, input scene ($\mathbf{s}$) is re-sampled, and $\mathbf{s}'$ denotes new scene including recently sampled objects, relations and affordances during negative phase.

Algorithm 1 Training COSMO.

Input: Training data, $\mathcal{S} = \{\mathbf{s}^i\}_i$; learning rate, α; number of epochs, m.
Output: Learned weights, $\mathbf{w}$.

for m epochs **do**
  for $\mathbf{s} \in \mathcal{S}$ **do**
    / Positive Phase */*
    $\mathbf{o}^{(0)} \leftarrow \mathbf{s}^{\alpha}_{\mathbf{o}}$, $\mathbf{r}^{(0)} \leftarrow \mathbf{s}^{\alpha}_{\mathbf{r}}$, $\mathbf{a}^{(0)} \leftarrow \mathbf{s}^{\alpha}_{\mathbf{a}}$,
    $\mathbf{h}^{(0)} \leftarrow p(\mathbf{h} \mid \mathbf{o}^{(0)}, \mathbf{r}^{(0)}, \mathbf{a}^{(0)})$
    Calculate p^+ for each edge.

    / Negative Phase */*
    Sample $\hat{\mathbf{h}}^{(0)}$ using Eqn. 4.4.
    $\mathbf{o}^{(1)} \leftarrow 0$, $\mathbf{r}^{(1)} \leftarrow 0$, $\mathbf{a}^{(1)} \leftarrow 0$
    $\mathbf{o}^{(1)} \leftarrow p(\mathbf{o} \mid \mathbf{o}^{(1)}, \hat{\mathbf{h}}^{(0)}, \mathbf{r}^{(1)}, \mathbf{a}^{(1)})$
    $\mathbf{r}^{(1)} \leftarrow p(\mathbf{r} \mid \mathbf{o}^{(1)}, \hat{\mathbf{h}}^{(0)}, \mathbf{a}^{(1)})$
    $\mathbf{a}^{(1)} \leftarrow p(\mathbf{a} \mid \mathbf{o}^{(1)}, \hat{\mathbf{h}}^{(0)}, \mathbf{r}^{(1)})$
    Sample $\hat{\mathbf{o}}^{(1)}, \hat{\mathbf{r}}^{(1)}, \hat{\mathbf{a}}^{(1)}$ using Eqn. 4.3, 4.5, 4.6.
    $\mathbf{h}^{(1)} \leftarrow p(\mathbf{h} \mid \hat{\mathbf{o}}^{(1)}, \hat{\mathbf{r}}^{(1)}, \hat{\mathbf{a}}^{(1)})$
    Calculate p^- for each edge.

    Update weights using Eqn. 2.5.

Since our dataset has small number of samples and input vectors are too sparse, pre-

cise inferences are crucial. Therefore, we prefer Gibbs sampling [17] that is a Monte Carlo Markov Chain (MCMC) method to approximate true data distribution instead of variational inference since MCMC methods can provide precise inference but variational inference methods cannot guarantee that [5].

4.1.2 Derivatives of Weights

Derivatives of weights can be calculated by taking partial derivative of G with respect to the particular types of weights.

For example, let us calculate $\partial G/\partial w_{ij}^{ho}$:

$$\frac{\partial G}{\partial w_{ij}^{ho}} = -\sum_{\alpha} \frac{p^{+}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha)}{p^{-}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha)} \frac{\partial p^{-}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha)}{\partial w_{ij}^{ho}}. \tag{4.7}$$

In Eq. 4.7, we do not take partial derivative of $p^{+}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha)$ with respect the weight since the input is clamped to the visible units in this probability. Therefore, it does not depend on weights.

Model expectations during negative phase can be defined as:

$$\begin{aligned} p^{-}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) &= \sum_{\beta} p^{-}(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha, \mathbf{h}_\beta), \\ &= \frac{\sum_\beta e^{-E_{\alpha\beta}}}{\sum_{\lambda\mu} e^{-E_{\lambda\mu}}}, \end{aligned} \tag{4.8}$$

where we assume $T = 0$.

Recall the energy term (Equation 4.2), when visible units are in state α and hidden states are in state β:

$$\begin{aligned} E(\mathbf{o}_\alpha, \mathbf{h}_\beta, \mathbf{r}_\alpha, \mathbf{a}_\alpha) &= -\sum_{i<j} h_i^\beta w_{ij}^{ho} o_j^\alpha \\ &\quad -\sum_{i,j,k} w_{ijk}^{r} r_{ijk}^\alpha o_j^\alpha o_k^\alpha - \sum_{i,j,k,l} w_{ij}^{rh} r_{ijk}^\alpha h_l^\beta \\ &\quad -\sum_{i,j,k} w_{ijk}^{a} a_{ijk}^\alpha o_j^\alpha o_k^\alpha - \sum_{i,j,k,l} w_{ij}^{ah} a_{ijk}^\alpha h_l^\beta, \end{aligned} \tag{4.9}$$

where $\mathbf{o}_\alpha$, $\mathbf{r}_\alpha$ and $\mathbf{a}_\alpha$ are vectors of visible units (objects, relations and affordances respectively) where the visible units are in state α. o_i^α, r_{ijk}^α and a_{ijk}^α are object, relations

and affordance units with specific indexes when the visible units are in state α. $\mathbf{h}_\beta$ is vector of hidden units when they are in state β. h_i^β is a hidden unit with a specific index when it is in state β.

We can calculate partial derivative of the energy term $E(\mathbf{o}_\alpha, \mathbf{h}_\beta, \mathbf{r}_\alpha, \mathbf{a}_\alpha)$ with respect to hiddens to objects weights ∂w_{ij}^{ho}.

$$\frac{\partial e^{-E_{\alpha\beta}}}{\partial w_{ij}^{ho}} = e^{-E_{\alpha\beta}} h_i^\beta o_j^\alpha. \tag{4.10}$$

By using Equation 4.8, recall:

$$\begin{aligned}
\frac{\partial p^-(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha)}{\partial w_{ij}^{ho}} &= \frac{\partial \frac{\sum_\beta e^{-E_{\alpha\beta}}}{\sum_{\lambda\mu} e^{-E_{\lambda\mu}}}}{\partial w_{ij}^{ho}}, \\
&= \frac{\sum_\beta e^{-E_{\alpha\beta}} h_i^\beta o_j^\alpha}{\sum_{\lambda\mu} e^{-E_{\lambda\mu}}} - \frac{\sum_\beta e^{-E_{\alpha\beta}} \sum_{\lambda\mu} e^{-E_{\lambda\mu}} h_i^\lambda o_j^\mu}{(\sum_{\lambda\mu} e^{-E_{\lambda\mu}})^2}, \\
&= \sum_\beta p^-(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha, \mathbf{h}_\beta) h_i^\beta o_j^\alpha, \\
&\quad -p^-(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) \sum_{\lambda\mu} p^-(\mathbf{o}_\lambda, \mathbf{r}_\lambda, \mathbf{a}_\lambda, \mathbf{h}_\mu) h_i^\mu o_j^\lambda.
\end{aligned} \tag{4.11}$$

We can rewrite positive and negative phase probabilities:

$$\begin{aligned}
p^+(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha, \mathbf{h}_\beta) &= p^+(\mathbf{h}_\beta | \mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) p^+(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha), \\
p^-(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha, \mathbf{h}_\beta) &= p^-(\mathbf{h}_\beta | \mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) p^-(\mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha),
\end{aligned}$$

and, recall that probability of hidden states for given visible states are same when network converges (i.e. hidden states are same when the input clamped to the visible nodes or when it runs freely.). Hence:

$$p^+(\mathbf{h}_\beta | \mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) = p^-(\mathbf{h}_\beta | \mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) \tag{4.12}$$

and, note that when inputs are clamped to the visible nodes, summation of probabilities of visible units being active for all visible configuration equals to 1:

$$\sum_\alpha p^+(\mathbf{h}_\beta | \mathbf{o}_\alpha, \mathbf{r}_\alpha, \mathbf{a}_\alpha) = 1. \tag{4.13}$$

By using Eq. 4.7, 4.11, 4.12 and 4.13, we can conclude that:

$$\frac{\partial G}{\partial w_{ij}^{ho}} = < h_i o_j >^+ - < h_i o_j >^-. \tag{4.14}$$

where $< h_i o_j >^+$ is the average probability of both units h_i and o_j being active when inputs are clamped to the visible nodes and $< h_i o_j >^-$ is corresponding probability when the network runs freely.

CHAPTER 5

EXPERIMENTS AND RESULTS

In this section, we evaluate COSMO on several scene modeling and robotics problems and compare the model against several baselines and alternative methods whenever possible.

The context of this chapter has been published in "International Conference on Robotics and Automation 2018" (ICRA 2018) [10] and has been submitted to the Robotics and Autonomous Systems (RAS) journal special issue on Semantic Policy and Action Representations for Autonomous Robots (SPAR) which is still under evaluation [9].

5.1 The Dataset

For our experiments, we formed a dataset composed of $6,976$ scenes, half of which is sampled from the *Visual Genome* (VG) dataset [29] and the other half from the SUN-RGBD dataset [53]. We used samples from both datasets since (i) the VG dataset has spatial relationships but these do not include relations useful for robots, such as left and right, and (ii) the VG dataset mostly includes outdoor datasets, which we compensate using the SUN-RGBD dataset, which is composed of indoor scenes only. Therefore, we included equal number of samples from both the VG and the SUN-RGBD datasets. However, the SUN-RGBD dataset did not have spatial relations labeled, therefore, we did manual labeling for the SUN-RGBD dataset.

Our dataset consists of 90 objects that commonly exist in scenes, including human-like (man, woman, boy etc.), physical objects (cup, bottle, jacket etc.), part of build-

Figure 5.1: Example scenes from the merged dataset used for the experiments.

ings (door, window etc.).

Our dataset is composed of the following eight spatial relations: left, right, front, behind, on-top, under, above, below. These spatial relations are annotated in the VG dataset already. However, we extended the original SUN-RGBD dataset by manually annotating these eight spatial relations. Moreover, we included verb-relations in the VG dataset as affordances into the dataset. The set of affordances include eat-ability, push-ability, play-ability, wear-ability, sit-ability, hold-ability, carry-ability, ride-ability, push-ability, use-ability.

Let us use $\mathcal{S} = \{\mathbf{s}_1, ..., \mathbf{s}_{6,976}\}$, where $\mathbf{s}_i$ denotes i^{th} sample, to denote the dataset. $\mathbf{s}_i$ has a vector form that represents the presence of objects, relations and affordances among them in the scene. Active (observed) variables are set to value 1, or to value 0 otherwise. Opposite spatial relations (e.g., left and right) can be represented as single relations in BMs since if object o_1 is to the left of object o_2, then we can state that object o_2 is to the right of object o_1. As a result, each sample is represented by a binary vector that has length $113,490$ $(90 + 14 \times 90 \times 90)$.

5.2 Compared Models

We compare COSMO with General Boltzmann Machine (GBM), Restricted Boltzmann Machine (RBM) and Relation Network (RN) [48] for several scene reasoning tasks that are crucial for various robotic scenarios. For GBM and RBM models, we used the same number of hidden nodes as in COSMO – see Section 5.4.

5.2.1 General Boltzmann Machine (GBM)

GBMs are unrestricted in terms of connectivity or the hierarchy in the network (either among the hidden or the visible nodes). However, this may make learning impractical, especially when hidden nodes are connected to each other. We allow connections within visible nodes to incorporate interactions between objects as required for scene modeling. In this structure, visible nodes consist of object, relation and affordance nodes as in COSMO. Unlike COSMO, GBM uses two-way edges for relation and affordance nodes, instead of tri-way edges. Similar to COSMO, all visible nodes are fully connected to the hidden nodes but connections within hidden nodes are not allowed. Therefore, the only difference between COSMO and the GBM model is how relation and affordance nodes are connected to objects. To make it comparable with COSMO, we used the same number of layers and hidden neurons in GBM as in COSMO.

5.2.2 Restricted Boltzmann Machine (RBM)

Different from GBM, an RBM only allows connections between visible and hidden nodes. To make it comparable with COSMO, we used the same number of layers and hidden neurons in RBM as in COSMO.

5.2.3 Relation Network (RN)

RNs [48] are simple neural networks to address problems related to relational reasoning. We modified RNs as shown in Figure 5.2 to make them compatible for our experiments: The input vectors are embedded with a Multi-Layer-Perceptron (MLP) and the activations of MLP are used as object pairs for another MLP, called the g network. In the original model, object pairs are concatenated with an embedding of a query text; however, we omit this since we assume that the model has one type of question for each scenario. For example, for the spatial relation estimation task, only object and affordance vectors are used as input and spatial relations are predicted.

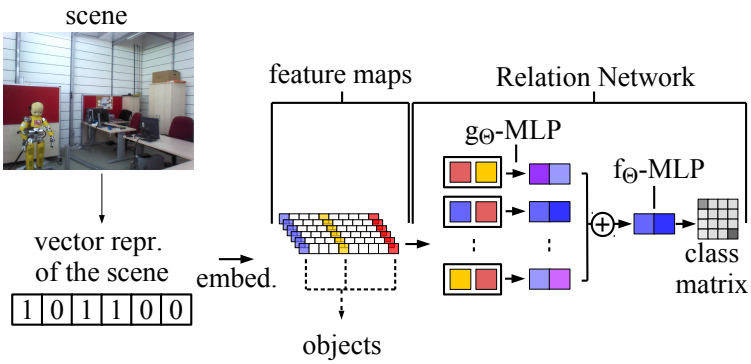


Figure 5.2: The Relational network (RN) architecture: A scene is represented by binary vector that indicates existence of object, spatial relations and affordances among them. The input vector is embedded using Multi-Layer-Perceptron (MLP). Activations of the MLP are used as feature maps to produce a set of objects for RN. Objects are illustrated as blue, yellow and red. Object pairs are fed into the g network whose output is fed into the f network to compute the relations. [Best viewed in color]

In this case, the model is trying to answer the question "what are relations among all objects in the scene?". Training RN to answer a specific question "what is the relation between object a and object b" requires additional training samples, including question-answer pairs. These are crucial drawbacks of RN when compared to COSMO. Being generative, COSMO have more flexibility on what can be queried with the scene modeled.

Our implementation of the RN method closely follows the original study. However, we had to adjust the architecture to fit to our data sizes. The embedding MLP network is composed of 2 layers (with 128 and 128 neurons respectively) with ReLU non-linearities. The g network is a MLP with 2 layers (with 256 and 256 neurons respectively) with sigmoid non-linearities. The prediction network, f, then is a MLP with 2 layers (with 64 and 64 neurons respectively) with sigmoid non-linearities.

We trained the RN model using Adam optimizer with default parameters (and learning rate of 0.001), 32 sized-batches and early stopping.

5.3 Network Training Performance

The dataset (composed of $6,976$ scenes) is split into three randomly: 60% for training, 30% for testing and 10% for validation. This split is used for training and testing all methods. For evaluating the training performance, we calculated an error on the difference between the clamped visible units and reconstructed visible states that are sampled in the negative phase:

$$E_{train} = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{S}} \sum_{i} \left(p(s_i^+) - p(s_i^-) \right)^2, \tag{5.1}$$

where the cumulative sum is normalized with the total number of samples ($|\mathcal{S}|$).

Figure 5.3 plots the error separately for the objects ($\mathbf{o}$), the spatial relations ($\mathbf{r}$) and affordances ($\mathbf{a}$). From the figure, we observe that the error is consistently decreasing for all types of visible units for both the training data and the validation data, suggesting that the networks is learning to represent the probability over objects, the spatial relations and the affordances very well.

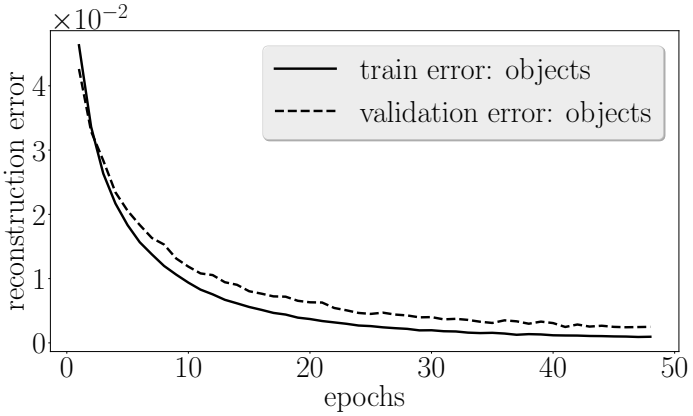
However, we observe in Figure 5.3(b) that the network learns affordances faster than objects and relations. This difference is owing to the fact that the set of possible affordances in a scene is much sparser than objects and relations, making the network quickly learn to estimate 0 (zero) for affordances, leading to a sudden decrease in the loss.

Table 5.1: Average running time in seconds (time) for one epoch and total number of parameters for different models.

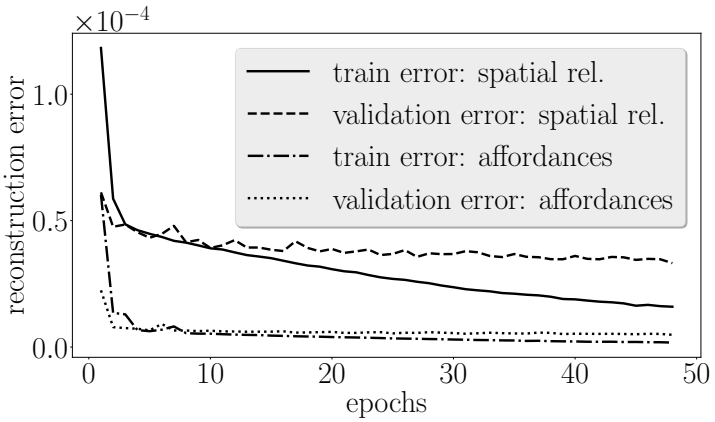
	COSMO	GBM	RBM	RN
Time (seconds)	181.06	208.22	179.06	102.30
# of params.	$13,802,600$	$13,871,200$	$13,734,000$	$12,463,104$

5.4 Analyzing the Hyper-parameters

We evaluated the effect of various hyper-parameters on COSMO's training performance (Eqn. 5.1). For all the analyses performed in this section, we looked at the error on the validation data (see Section 5.1).



(a)

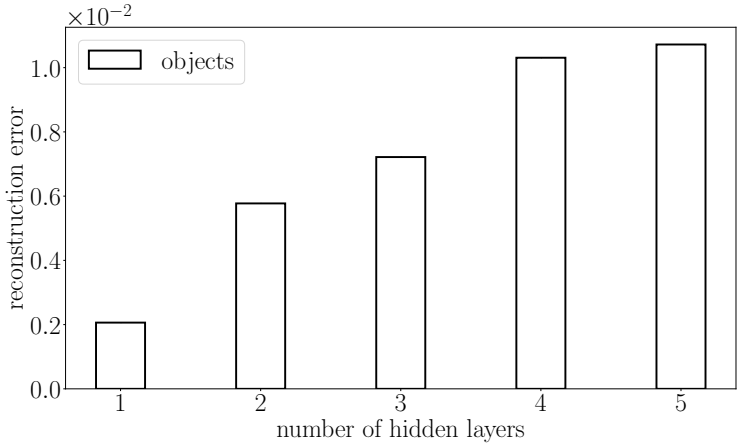


(b)

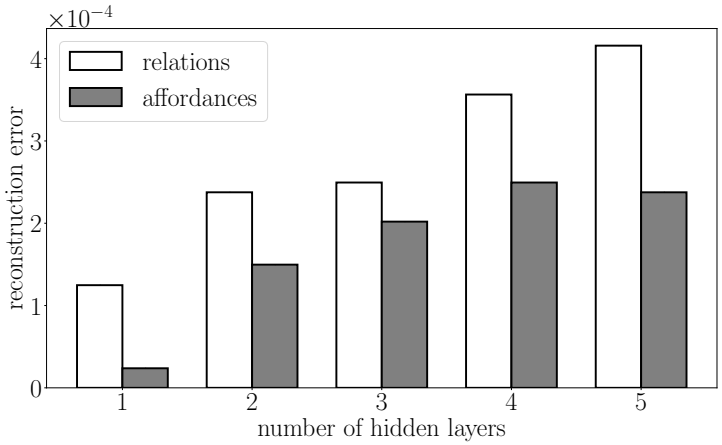
Figure 5.3: Reconstruction error vs. epochs plot during COSMO training for (a) objects and (b) spatial relations and affordances.

First, we analyzed the effect of the number of hidden layers. For this end, we tested models with 1, 2, 3, 4 and 5 hidden layers. As shown in Figure 5.4, the model has the lowest reconstruction error (for all types of visible nodes) with only one hidden layer, and the error rises when number of hidden layers increases. This increase might be because the dataset might be falling insufficient for the increase in the number of parameters when the number of layers increases.

Secondly, we analyzed the effect of the number of hidden neurons in a hidden layer. We tested 50, 100, 200, 400, 800 and 2000 hidden neurons. As shown in Figure



(a) objects

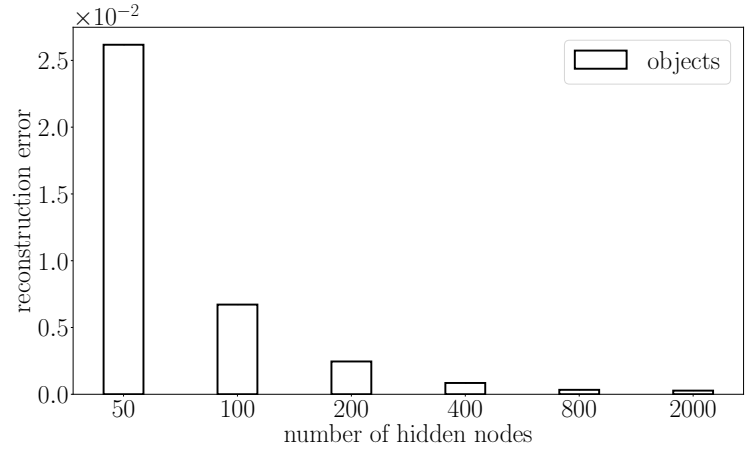


(b) relations and affordances

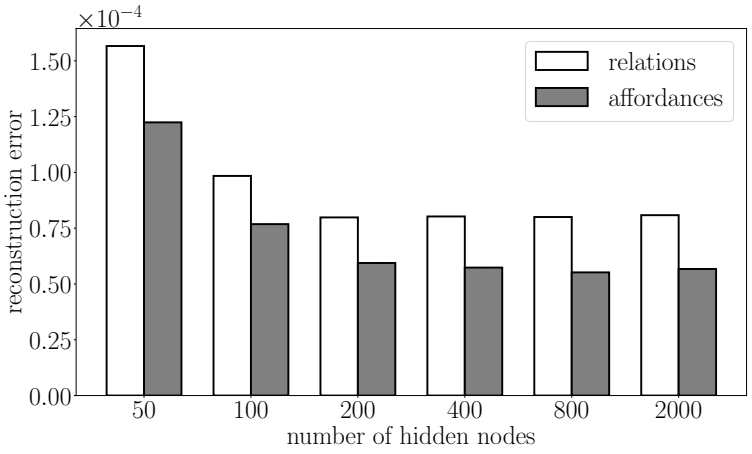
Figure 5.4: Reconstruction errors (after 30 epochs) for different numbers of hidden layers. (a) Reconstruction error for object nodes. (b) Reconstruction error for relation and affordance nodes.

5.5, the reconstruction error decreases when number of hidden neurons increases, as expected.

Lastly, we analyzed the effect of different annealing schedules. We tried the following annealing schedules (selected from [42]), namely, exponential multiplicative cooling (emc, Eq. 5.2), linear multiplicative cooling (li-mc, Eq. 5.3) and logarithmic



(a) objects



(b) relations and affordances

Figure 5.5: Reconstruction errors (after 30 epochs) for different number of hidden nodes. (a) Reconstruction error for object nodes. (b) Reconstruction error for relation nodes and affordance nodes.

multiplicative cooling (log-mc, Eq. 5.4), with initial temperature (T_0) set to 4.0:

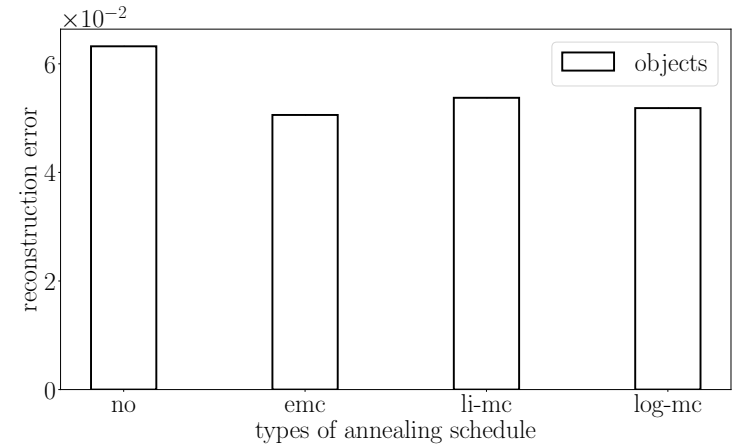
$$T_i = T_0 \cdot a^i, \qquad (0.8 \leqslant a \leqslant 0.9) \tag{5.2}$$

$$T_i = \frac{T_0}{1 + a \times i}, \qquad (a > 0) \tag{5.3}$$

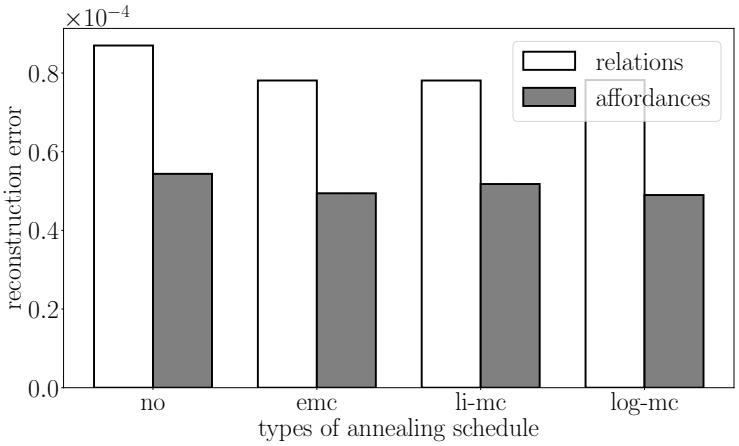
$$T_i = \frac{T_0}{1 + a \log(1 + i)}, \qquad (a > 1) \tag{5.4}$$

As shown in Figure 5.6, although annealing schedules decreases reconstruction errors all types of nodes, differences between them are not significant.

In summary, our analysis suggests that COSMO with one hidden layer, with 400 hidden nodes (although, as shown in Figure 5.5, 800 or more hidden nodes provide better performance, the performance gain is insignificant compared to the computational overload) and emc annealing performs best and therefore, in the rest of the paper, we adopted these settings for COSMO. For RBM and GBM, we used the same number of hidden nodes as COSMO.



(a) objects



(b) relations and affordances

Figure 5.6: Reconstruction errors (after 30 epochs) for different annealing schedules with initial temperature 4.0. (a) Reconstruction error for object nodes. (b) Reconstruction error for relation and affordance nodes.

5.5 Comparison Measures

For evaluating the performance of the methods, we use precision, recall and F-measure which are defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{5.5}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5.6}$$

$$\text{F-measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{5.7}$$

where TP, FP and FN stand for the number of true positives, false positives and false negatives, respectively. Definitions of TP, FP and FN are task-dependent, and therefore, they are defined for each task separately.

5.6 Task 1: Spatial Relation Estimation

Being generative, COSMO can estimate relations in the scene given the objects in the scene. Contextual information that arises from active objects, regardless of spatial relation and affordance nodes, allows the model to determine which spatial relations should be active according to the context.

For testing, initially, the model sees the environment in a "bag of objects" sense by clamping only objects to the visible nodes and relation nodes are set to zero. Next, the hidden nodes (i.e., context) are sampled using object nodes only. Then, the spatial relation nodes are sampled from objects, affordances and the context. This procedure is summarized in Algorithm 2.

For this task, we define True Positive (TP) as the number of spatial relation nodes which are active in both the input scene ($\mathbf{s}$) and the reconstructed scene ($\mathbf{s}'$); True Negative (TN) as the number of spatial relation nodes which are both in-active in $\mathbf{s}$ and $\mathbf{s}'$; False Positive (FP) as the number of spatial relation nodes which are inactive in $\mathbf{s}$ but active in $\mathbf{s}'$; False Negative (FN) as the number of spatial relation nodes which

Algorithm 2 Algorithm used for the relation estimation task (Task 1).

Input: A scene $\mathbf{s}$; the number of Gibbs steps, k.
Output: Relation node activations, $\mathbf{r}$.

$\mathbf{r} \leftarrow 0$. ▷ Set relation node activations to 0.
for k sampling steps **do**
    $\mathbf{o} \leftarrow \mathbf{s_o}$, $\mathbf{a} \leftarrow \mathbf{s_a}$ ▷ Clamp objects and affordances.
    Sample hidden nodes **h** using Eq. 4.4.
    Sample relation nodes **r** using Eq. 4.5.

are active in $\mathbf{s}$ and in-active in $\mathbf{s}'$. These are defined formally as follows:

$$TP = \left|\{x : x \in G_r^+ \wedge x \in M_r^+\}\right|, \tag{5.8}$$

$$TN = \left|\{x : x \in G_r^- \wedge x \in M_r^-\}\right|, \tag{5.9}$$

$$FP = \left|\{x : x \in G_r^- \wedge x \in M_r^+\}\right|, \tag{5.10}$$

$$FN = \left|\{x : x \in G_r^+ \wedge x \in M_r^-\}\right|, \tag{5.11}$$

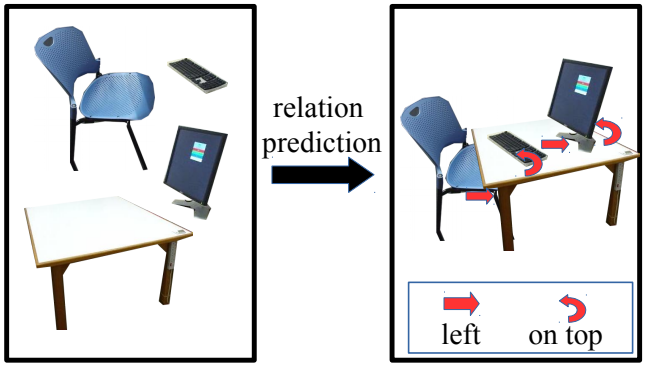
where, x is a relation node; G_r^+ and G_r^- are respectively sets of active and passive relation nodes in the sample; and M_r^+ and M_r^- are sets of active and passive relation nodes respectively at the end of model's reconstruction.

Table 5.2 lists the performance of COSMO for this task and compares it against RBM, GBM, and RN. In comparison to the other models, we see that COSMO provides the best performance.

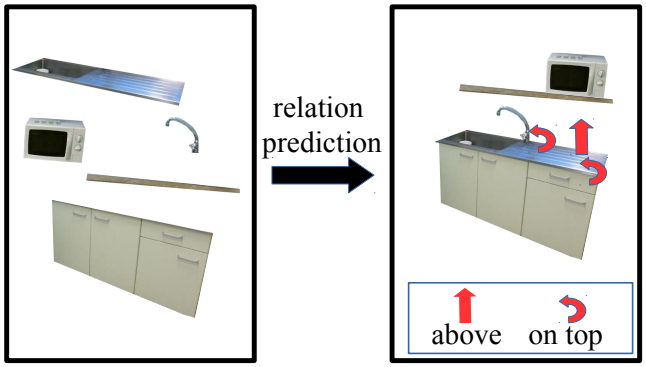
Moreover, we provide some visual examples in Figure 5.7, where we see that our model can discover spatial relations between objects, i.e., how to roughly place a set of objects together.

Table 5.2: Task 1 (Spatial Relation Estimation) performances.

	Precision	**Recall**	**F1-measure**
COSMO	0.1511	**0.3112**	**0.2034**
GBM	**0.1559**	0.1125	0.1307
RBM	0.0043	0.0132	0.0066
RN	0.0166	0.0132	0.0147



(a)



(b)

Figure 5.7: Some example relations estimated by COSMO for given sets of objects (Task 1). Only a subset of the relations are shown for the sake of visibility.

In some cases, naturally, the "bag of objects" approach may not provide enough contextual information in order model to predict ground truth spatial relationships in the test set. For instance, consider a scene consisting of *plate, table, cabinet* objects. In a kitchen with eating context, the *plate* can be *on* the *table*, whereas, in a kitchen without eating context, *plate* is likely to be *in* the *cabinet*. These cases can reduce testing accuracy for the estimated relation between objects like *plate*. However, given such examples during training, COSMO is able to capture the probability of all these cases and therefore handle scene modeling tasks in such settings accordingly.

5.7 Task 2: What is missing in the scene?

In this task, COSMO predicts missing objects in the scene according to the current context. The model is provided "partially observed scenes" where some of the objects are removed randomly for testing.

Firstly, observed objects, spatial relations and affordances are clamped to the visible units, then the model is relaxed to find hidden node activations (i.e. the context of the scene). Finally, by using visible (scene description) and hidden (context) node activations, the network tries to find the missing objects in the scene as outlined in Algorithm 3.

For this task, we define TP as the number of object nodes that are activated correctly according to ground truth sample; FP as the number of object nodes that the model activates but it should be deactivated according to ground truth; TN as the number of object nodes that are deactivated correctly according to ground truth and FN as number of objects that the model deactivates yet should be activated according to ground truth. We can formally define these as follows:

$$\begin{aligned} TP &= \left|\{x : x \in G_o^+ \wedge x \in M_o^+\}\right|, \\ TN &= \left|\{x : x \in G_o^- \wedge x \in M_o^-\}\right|, \\ FP &= \left|\{x : x \in G_o^- \wedge x \in M_o^+\}\right|, \\ FN &= \left|\{x : x \in G_o^+ \wedge x \in M_o^-\}\right|, \end{aligned} \tag{5.12}$$

where x is an object node; G_o^+ and G_o^- are the sets of active and passive object nodes respectively in ground truth sample; and M_o^+ and M_o^- are sets of active and passive object nodes respectively at the end of model's reconstruction.

As shown in Table 5.3, our model performs better than RBM, GBM and RN. See also Figure 5.8, which shows some visual examples for most likely objects found for a target position in the scene.

Algorithm 3 The algorithm for finding missing objects (Task 2).

1: **Input:** A scene, $\mathbf{s}$; the number of Gibbs steps, k.
2: **Output:** Initially in-active object nodes in $\mathbf{s}$, $\mathbf{o}'$.
3:
4: **for** k sampling steps **do**
5: $\mathbf{o} \leftarrow \mathbf{s_o}$; $\mathbf{r} \leftarrow \mathbf{s_r}$; $\mathbf{a} \leftarrow \mathbf{s_a}$ ▷ Clamp input scene.
6: Sample hidden nodes $\mathbf{h}$ using Eq. 4.4.
7: Sample *in-active* object nodes $\mathbf{o}'$ using Eq. 4.3.

Table 5.3: Task 2 (finding missing objects) performances.

	Precision	**Recall**	**F1-measure**
COSMO	**0.9387**	**0.0527**	**0.0998**
GBM	0.8260	0.0415	0.0790
RBM	0.7250	0.0301	0.0578
RN	0.8000	0.0212	0.0414

5.8 Task 3: What is extra in the scene?

In this task, COSMO predicts objects that are out of context in the scene. For this purpose, objects are randomly selected and added to the original scene for testing.

Firstly, observed objects, spatial relations and affordances are clamped to the visible units, then the model is relaxed to find hidden node activations (i.e. the context of the scene). Finally, by using visible (scene description) and hidden (context) node activations, the network tries to remove objects that are out of context in the scene as outlined in Algorithm 4.

For this task, we use the TP, TN, FP and FN as defined in Equation 5.12.

As shown in Table 5.4, our model performs better than RBM, GBM and RN for finding extra objects in the scene. See also Figure 5.9, which shows some visual examples for finding the object that is out of context in the scene.

(a)

(b)

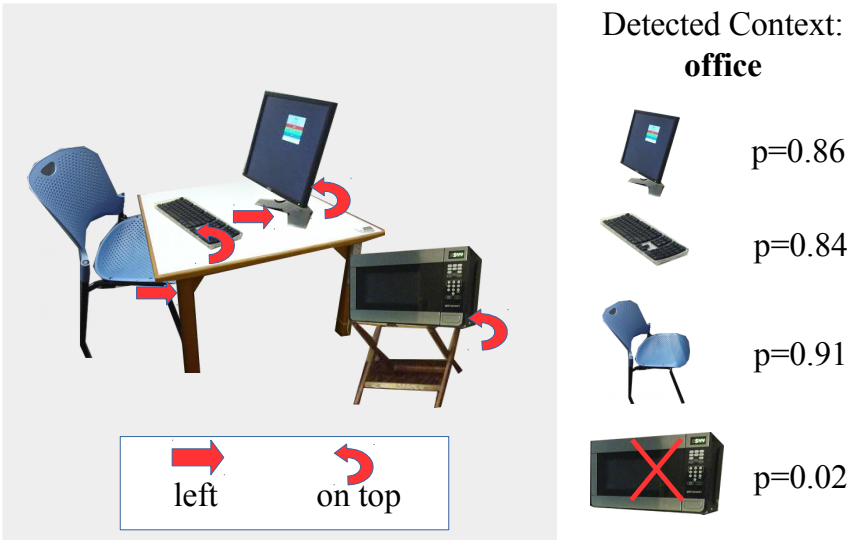
Figure 5.8: Some examples illustrating the performance of COSMO on finding a missing object in a scene (Task 2).

5.9 Task 4: Affordance Prediction

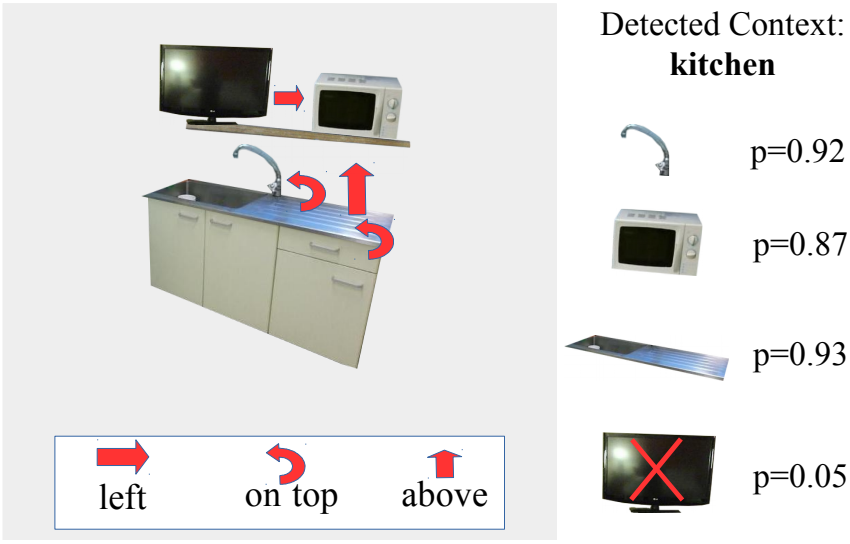
Affordances of objects may differ for different subjects in different contexts [57]. Therefore, agents should be aware of the context that they are in in order to reason about the affordances of objects. We show that COSMO can allow agents to deter-

Table 5.4: Task 3 (finding extra objects) performances.

	Precision	**Recall**	**F1-measure**
COSMO	**0.9183**	**0.0482**	**0.0917**
GBM	0.8113	0.0479	0.0865
RBM	0.7826	0.0382	0.0729
RN	0.7368	0.0297	0.0572



(a)



(b)

Figure 5.9: Some examples illustrating the performance of COSMO on finding the out-of-context object in a scene (Task 3).

Algorithm 4 The algorithm for finding the out-of-context object (Task 3).

Input: A scene, $\mathbf{s}$; the number of Gibbs steps, k.
Output: Initially active object nodes in $\mathbf{s}$, $\mathbf{o}'$.

for k sampling steps **do**
    $\mathbf{o} \leftarrow \mathbf{s_o}$; $\mathbf{r} \leftarrow \mathbf{s_r}$; $\mathbf{a} \leftarrow \mathbf{s_a}$ ▷ Clamp input scene.
    Sample hidden nodes $\mathbf{h}$ using Eq. 4.4.
    Sample *active* object nodes $\mathbf{o}'$ using Eq. 4.3.

mine affordances of objects using the current context.

For this task, firstly, objects and relations are clamped to the visible nodes, and the hidden nodes (i.e. context) are sampled. Then, affordance nodes are sampled using hidden nodes (context), objects and relations (current scene), as illustrated in Algorithm 5.

Algorithm 5 Algorithm for the affordance prediction task (Task 4).

Input: A scene, $\mathbf{s}$; the number of Gibbs steps, k.
Output: Affordance node activations, $\mathbf{a}$.

$\mathbf{a} \leftarrow 0$. ▷ Set affordance nodes to 0.
for k sampling steps **do**
    $\mathbf{o} \leftarrow \mathbf{s_o}$, $\mathbf{r} \leftarrow \mathbf{s_r}$ ▷ Clamp objects and relations.
    Sample hidden nodes $\mathbf{h}$ using Eq. 4.4.
    Sample affordance nodes $\mathbf{a}$ using Eq. 4.6.

For this task, we define TP as the number of affordance nodes that are activated correctly according to the ground truth sample; FP as the number of affordance nodes that the model activates but should be deactivated according to the ground truth; TN as the number of affordance nodes that are deactivated correctly according to ground truth, and FN as the number of affordance nodes that the model deactivates yet should

be activated according to the ground truth. We defined them formally as follows:

$$
\begin{aligned}
TP &= \left|\{x : x \in G_a^+ \wedge x \in M_a^+\}\right|, \\
TN &= \left|\{x : x \in G_a^- \wedge x \in M_a^-\}\right|, \\
FP &= \left|\{x : x \in G_a^- \wedge x \in M_a^+\}\right|, \\
FN &= \left|\{x : x \in G_a^+ \wedge x \in M_a^-\}\right|,
\end{aligned}
\tag{5.13}
$$

where, x is an affordance node; G_a^+ and G_a^- are sets of active and passive affordance nodes respectively in the ground truth sample; and M_a^+ and M_a^- are the sets of active and passive affordance nodes respectively at the end of model's reconstruction.

As shown in Table 5.5, our model performs better than RBM, GBM and RN. See also Figure 5.10(a), which shows some visual examples for affordance prediction for different objects.

Table 5.5: Task 4 (affordance prediction) performances.

	Precision	**Recall**	**F1-measure**
COSMO	**0.2039**	**0.3129**	**0.2469**
GBM	0.1372	0.1068	0.1201
RBM	0.0769	0.0076	0.0138
RN	0.0125	0.0091	0.0105

5.10 Task 5: Objects affording an action

Being generative, COSMO allows reasoning about object affordances in various ways. In this task, we evaluate the methods on finding objects that afford a certain action. For this end, some of the object nodes, which are the object part of an affordance-triplet, and corresponding affordance nodes are deactivated. Then, the model samples hidden nodes using the partially observed scene. In the reconstruction phase, the model samples deactivated objects and affordance nodes that includes these objects using context and observed scene. This is formalized in Algorithm 6.

For this task, we use same formal definitions of TP, FP, TN and FN in Equation 5.13. However, in this task, G_a^+, G_a^-, M_a^+ and M_a^- include affordance nodes that correspond to a specific *action* and *subject* instead of all affordance nodes.

Table 5.6 lists the performance of the methods for finding the objects affording a certain action. We see a significant difference between the performance of COSMO and those of GBM, RBM and RN. See also Figure 5.10(b), which shows some visual examples for predicting the object that affording specific action.

Algorithm 6 The algorithm for finding objects that afford a given action (Task 5).

Input: A scene, $\mathbf{s}$; an action, *act*; the subject of action, *subj*; the number of Gibbs steps, k.
Output: Active affordance nodes, $\mathbf{a}_{i_a i_s i_o}$.
i_a = index of *act* in affordance vocabulary.
i_o = index of *subj* in object vocabulary.
for k sampling steps **do**
    $\mathbf{o} \leftarrow \mathbf{s_o}$ ▷ Clamp objects to the visible nodes.
    $\mathbf{r} \leftarrow \mathbf{s_r}$ ▷ Clamp relations to the visible nodes.
    $\mathbf{a} \leftarrow \mathbf{s_a}$ ▷ Clamp affordances to the visible nodes.
    Sample hidden nodes **h** using Eq. 4.4.
    Sample affordance node $\mathbf{a}_{i_a i_s i_o}$ using Eq. 4.3.

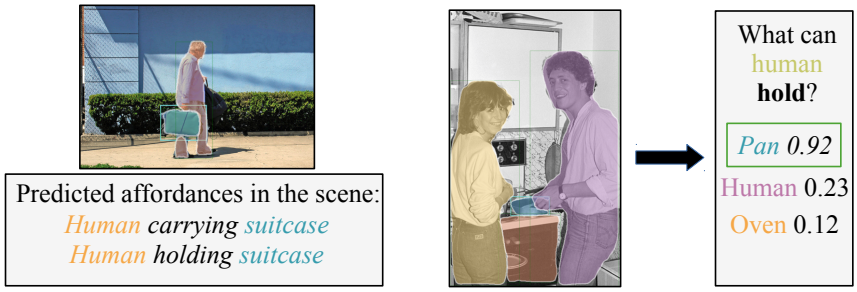
Table 5.6: Task 5 (finding objects affording a given action) performances.

	Precision	**Recall**	**F1-measure**
COSMO	**0.3170**	**0.4482**	**0.3714**
GBM	0.2537	0.0739	0.1144
RBM	0.0740	0.0869	0.0800
RN	0.0157	0.0689	0.0256

5.11 Task 6: Who is the actor for this task?

Robots should also be able to reason about the possible actors (subjects) of a given action or a task. Context plays a critical role here since it can modulate the candidate actors for a given action.

In this task, we evaluate performances on finding proper subjects (actors) for a certain action with a specific object. For this end, some of the object nodes, which are the subject part of an affordance-triplet, and affordance nodes are deactivated. Then, the



(a) (b)

Figure 5.10: Some examples illustrating (a) the performance of COSMO on affordance prediction (Task 4) and (b) finding objects that affording specific action (Task 5).

model samples hidden nodes using the partially observed scene. In the reconstruction phase, the model samples deactivated objects and affordance nodes that have proper subject for given action by using context and observed scene. This is formalized in Algorithm 7.

For this task, we use same formal definitions of TP, FP, TN and FN in Equation 5.13. However, in this task, G_a^+, G_a^-, M_a^+ and M_a^- include affordance nodes that correspond to a specific *action* and *object* instead of all affordance nodes.

In Table 5.7, the performances of the methods are listed. We see that GBM performs better in terms of precision whereas COSMO yields a much better recall performance, leading to an overall better performance in terms of the F-measure.

Table 5.7: Task 6 (What is the actor of the affordance?) performances.

	Precision	**Recall**	**F1-measure**
COSMO	0.3055	**0.4782**	**0.3728**
GBM	**0.3333**	0.0689	0.1142
RBM	0.0539	0.0586	0.0561
RN	0.0312	0.1739	0.0529

Algorithm 7 The algorithm for finding the subject for a given action (Task 6).

Input: A scene, $\mathbf{s}$; an action, *act*; the object of action, *obj*; the number of Gibbs steps, k.
Output: Active affordance nodes, $\mathbf{a}_{i_a i_s i_o}$.
i_a = index of *act* in affordance vocabulary.
i_o = index of *obj* in object vocabulary.
for k sampling steps **do**
    $\mathbf{o} \leftarrow \mathbf{s_o}$ ▷ Clamp objects to the visible nodes.
    $\mathbf{r} \leftarrow \mathbf{s_r}$ ▷ Clamp relations to the visible nodes.
    $\mathbf{a} \leftarrow \mathbf{s_a}$ ▷ Clamp affordances to the visible nodes.
    Sample hidden nodes **h** using Eq. 4.4.
    Sample affordance node $\mathbf{a}_{i_a i_s i_o}$ using Eq. 4.3.

5.12 Task 7: Improving Object Detection

In this task, we test whether we can use COSMO to rectify wrong detections and find missing detections made by object detectors. For this purpose, we used three state-of-the-art three object detection networks (namely, RetinaNet [32], Faster R-CNN [45], and Mask R-CNN [20]) with the ResNet-101-FPN [21] backbone model trained on the COCO dataset [33].

For this task, we first run the deep object detector on the input image. Then, we provide the detected objects to COSMO, and relax the network to see how COSMO updates the object nodes. We calculate average precision over 100 randomly selected images and compare the performance of the deep detectors before and after applying COSMO.

As shown in Table 5.8, COSMO significantly improves the detection performance of the deep networks. Looking at the visual example provided in Figure 5.11, we observe that COSMO can correct the mistakes made by the object detectors, and suggest objects that were missed by the detectors.

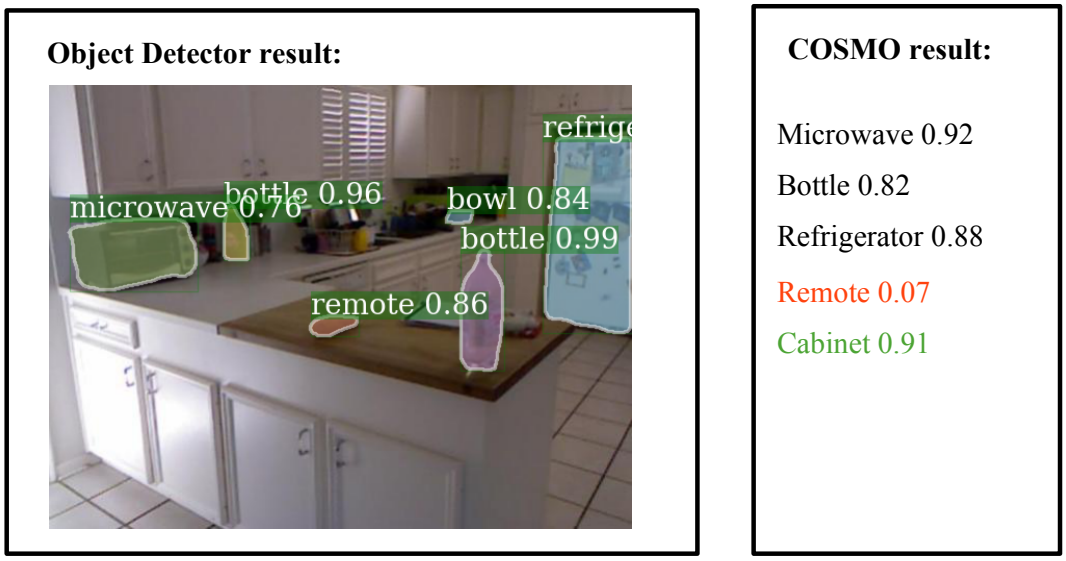


Figure 5.11: An example showing that COSMO improves the result of an object detector. COSMO is provided by the objects that are labeled by the object detector and updates the object nodes. After this step, COSMO assigns low probability to "remote" object (i.e., it determines "remote" as out of context). Then, it assigns high probability to "cabinet" object that should exist in the scene according to the context (i.e., it determines "cabinet" is missing in the scene). Other objects are omitted for the sake of visibility.

5.13 Task 8: Random scene generation

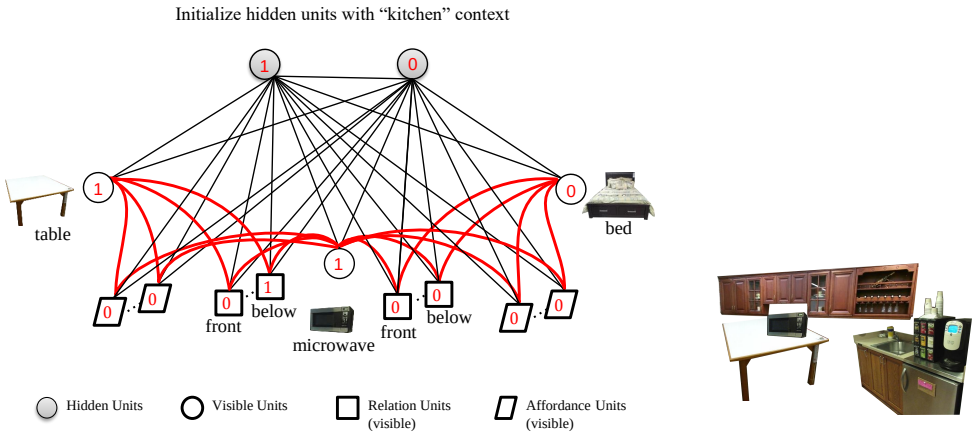
In this task, we demonstrate how we can use another generative ability of COSMO: we can select a hidden node (or more of them, leaving the other hidden neurons randomly initialized or set to zero), and sample visible nodes (including relations and affordances) that describe a scene. Figure 5.12 shows a visual example.

5.14 Task 9: Experiments on a Real Robot

In this task, we evaluate COSMO on Nao and illustrate how the tasks 1-8 conducted in this section can be useful for a robot. For this purpose, Nao uses Mask R-CNN

Table 5.8: Task 7: Improving object detections with COSMO. The average precision for different object detectors with and without COSMO are listed.

	w/o COSMO	**w COSMO**
RetinaNet [32]	0.4964	0.6966
Faster R-CNN [45]	0.4388	0.6752
Mask R-CNN [20]	0.4273	0.6648



(a) (b)

Figure 5.12: An example illustrating scene generation capability of COSMO (Task 8). (a) When a context (hidden node) is activated, (b) active nodes in the sampled visible nodes define a scene for the context. In (b), the "selected' objects are placed in the scene based on the predicted spatial relations.

to detect objects in the scene, and COSMO is initialized with these detections (only object nodes are clamped with the detected objects, the other visible nodes (relations and affordances) are estimated after sampling the hidden nodes). See Figure 5.13 for a snapshot.

Once COSMO is relaxed, Nao can reason about objects, relations, affordances, missing objects or out-of-context objects in the scene. An interactive experiment has been conducted with Nao where Nao answers questions about the scene using the active nodes in COSMO. See the accompanying video (also provided at `https://bozcani.github.io/COSMO`) for the experiments.

Figure 5.13: A snapshot of an experiment performed with Nao (Task 9). Nao uses Mask R-CNN to detect objects in the scene, and COSMO is initialized with these detections. Once this has been performed, Nao can reason about relations, affordances, missing objects or out-of-context objects in the scene. See the accompanying video (also provided at `https://bozcani.github.io/COSMO`) for the experiments.

CHAPTER 6

CONCLUSION AND DISCUSSION

In this thesis, we merged different types of robotic knowledge bases and proposed a contextual scene model that is useful for several robotic tasks.

Before proposing a proper mathematical model for scene modeling, we analyzed different types of robotic knowledge bases. For this end, we investigated and compared KnowRob [56], RoboBrain [49], KB using MLN [64], ConceptNet [54], ConceptWeb [12]. By using Web Ontology Language (OWL), we merged some of the concepts of ConceptWeb, ConceptNet and KnowRob, and we decided to represent objects, spatial relations and affordances in our scene model.

Moreover, we proposed a novel method (COSMO) for contextualized scene modeling. For this purpose, we extended Boltzmann Machines (BMs) to include spatial relations and affordances via tri-way edges in the model since we concluded that a relation or an affordance can be represented as triplet that contains *subject*, *relation* and *object* in the first part of this thesis. For integrating spatial relations and affordances into the model, we introduced shared nodes into BMs, allowing the concept of relations and affordances to be shared among different objects pairs. We evaluated and compared our model on several tasks on a real dataset and a real robot platform.

The experimented tasks included spatial relation estimation, finding missing objects in the scene, finding out-of-context objects, random scene generation, affordance prediction, finding actors and objects for particular verb. We showed that our model is the best compared to our baseline models (RBM, GBM, RN). Moreover, we tested whether we can use COSMO to increase the performance of object detectors. For

this end, we used annotations of objects that are detected by different types of object detectors. COSMO works as contextual pipeline over perception pipeline (i.e. object detectors). We showed that COSMO significantly improves object detection performances.

Moreover, we run COSMO on NAO Humanoid Robot. For this task, we use object detectors that detect objects in the images that are captured from NAO. By using COSMO, NAO can reason about objects and relations and affordances among them.

6.1 Limitations and Future Work

The COSMO has a limitation of assuming fixed-length object vocabulary. Therefore, novel objects that have not yet been encountered cannot be adapted to the model. However, in real life, types of objects can vary according to the different contexts. To overcome this problem, input layer would be designed in incremental manner.

The second limitation is related to the scalability. The model considers all possible object pairs for given relations or affordances. Therefore, the number of relations and affordances nodes increases exponentially with the increase in the number of objects. In the future work, relation and affordance embedding can be used and adapted to the COSMO.

The model represents the environment in terms of existence of objects. It does not detect each object if multiple instances of an object are contained in a scene. Even if we do not handle multiple objects explicitly, they can be inferred by activations of relations. For example, if the model find relations of "the lamp is left of the bed" and "the lamp is right of the bed", it means there are at least two lamps in the scene.

In our work, spatial relations are represented as qualitative abstractions (left, right, behind etc.) from metric data. This might be inadequate for tasks requiring precise location detection of objects.

Lastly, the number of object nodes is rather few compared to numbers of affordance and relation nodes. Therefore, effect of object nodes to hidden activations can be

diminished by relation and affordance nodes. To overcome this problem, additional weights can be added to links between object and hidden nodes in order to balance effect of relations, affordances and objects to hidden activations.

REFERENCES

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

[2] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. Contextually guided semantic labeling and search for three-dimensional point clouds. *The International Journal of Robotics Research*, 32(1):19–34, 2013.

[3] I. Atıl, N. Dag, S. Kalkan, and E. Sahin. Affordances and emergence of concepts. In *10th International Conference on Epigenetic Robotics*, 2010.

[4] L. W. Barsalou. Simulation, situated conceptualization, and prediction. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521):1281–1289, 2009.

[5] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, (just-accepted), 2017.

[6] S. Blumenthal and H. Bruyninckx. Towards a domain specific language for a scene graph based robotic world model. *arXiv preprint arXiv:1408.0200*, 2014.

[7] A. Boularias, O. Kroemer, and J. Peters. Learning robot grasping from 3-d images with markov random fields. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1548–1553. IEEE, 2011.

[8] I. Bozcan and S. Kalkan. Combining different knowledge-bases into a single partially-grounded robotic knowledge-base. 2017.

[9] I. Bozcan and S. Kalkan. Cosmo: Contextualized scene modeling with boltzmann machines. *arXiv preprint arXiv:1807.00511*, 2018.

[10] I. Bozcan, Y. Oymak, İ. Z. Alemdar, and S. Kalkan. What is (missing or wrong) in the scene? a hybrid deep boltzmann machine for contextualized scene modeling. *Accepted for IEEE International Conference on Robotics and Automation (ICRA) 2018; arXiv preprint arXiv:1710.05664*, 2018.

[11] H. Çelikkanat, E. Şahin, and S. Kalkan. Integrating spatial concepts into a probabilistic concept web. In *IEEE International Conference on Advanced Robotics (ICAR)*, 2015.

[12] H. Çelikkanat, G. Orhan, and S. Kalkan. A probabilistic concept web on a humanoid robot. *IEEE Transactions on Autonomous Mental Development*, 7(2):92–106, 2015.

[13] H. Celikkanat, G. Orhan, and S. Kalkan. A probabilistic concept web on a humanoid robot. *IEEE Transactions on Autonomous Mental Development*, 7(2):92–106, 2015.

[14] H. Celikkanat, G. Orhan, N. Pugeault, F. Guerin, E. Şahin, and S. Kalkan. Learning context on a humanoid robot using incremental latent dirichlet allocation. *IEEE Transactions on Cognitive and Developmental Systems*, 8(1):42–59, 2016.

[15] E. Şahin, M. Çakmak, M. R. Doğar, E. Uğur, and G. Üçoluk. To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior*, 15(4):447–472, 2007.

[16] T.-T. Do, A. Nguyen, I. Reid, D. G. Caldwell, and N. G. Tsagarakis. Affordancenet: An end-to-end deep learning approach for object affordance detection. *arXiv preprint arXiv:1709.07326*, 2017.

[17] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, (6):721–741, 1984.

[18] J. J. Gibson. *The Ecologial Approach to visual perception*. Lawrence Erlbaum Associates, 1986.

[19] S. Guadarrama, L. Riano, D. Golland, D. Gouhring, Y. Jia, D. Klein, P. Abbeel, and T. Darrell. Grounding spatial relations for human-robot interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988. IEEE, 2017.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[22] W. Hwang, J. Park, H. Suh, H. Kim, and I. H. Suh. Ontology-based framework of robot context modeling and reasoning for object recognition. In *Int. Conf. on Fuzzy Systems and Knowledge Discovery*, 2006.

[23] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor. Affordances in psychology, neuroscience and robotics: a survey. *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[24] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *arXiv preprint arXiv:1612.06890*, 2016.

[25] D. Joho, G. D. Tipaldi, N. Engelhard, C. Stachniss, and W. Burgard. Nonparametric bayesian models for unsupervised scene analysis and reconstruction. *Robotics*, page 161, 2013.

[26] S. Kalkan, N. Dag, O. Yürüten, A. M. Borghi, and E. Şahin. Verb concepts from affordances. *Interaction Studies*, 15(1):1–37, 2014.

[27] M. Kokic, J. A. Stork, J. A. Haustein, and D. Kragic. Affordance detection for task-specific grasping using deep learning. In *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 91–98. IEEE, 2017.

[28] H. S. Koppula, R. Gupta, and A. Saxena. Learning human activities and object affordances from rgb-d videos. *The International Journal of Robotics Research*, 32(8):951–970, 2013.

[29] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[30] X. Li, J.-F. Martínez, G. Rubio, and D. Gómez. Context reasoning in underwater robots using mebn. *arXiv preprint arXiv:1706.07204*, 2017.

[31] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1417–1424, 2013.

[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

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

[34] M. Lopes, F. S. Melo, and L. Montesano. Affordance-based imitation learning in robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1015–1021. IEEE, 2007.

[35] F. Mastrogiovanni, A. Scalmato, A. Sgorbissa, and R. Zaccaria. Robots and intelligent environments: Knowledge representation and distributed context assessment. *Automatika*, 52(3):256–268, 2011.

[36] P. Meissner, R. Reckling, R. Jakel, S. R. Schmidt-Rohr, and R. Dillmann. Recognizing scenes with hierarchical implicit shape models based on spatial object relations for programming by demonstration. In *IEEE International Conference on Advanced Robotics (ICAR)*, 2013.

[37] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt. Learning relational affordance models for robots in multi-object manipulation tasks. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4373–4378. IEEE, 2012.

[38] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Modeling affordances using bayesian networks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4102–4107, 2007.

[39] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Learning object affordances: From sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[40] R. M. Neal. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992.

[41] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis. Detecting object affordances with convolutional neural networks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2765–2770. IEEE, 2016.

[42] Y. Nourani and B. Andresen. A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical and General*, 31(41):8373, 1998.

[43] J. Philbin, J. Sivic, and A. Zisserman. Geometric LDA: A generative model for particular object discovery. In *British Machine Vision Conference (BMVC)*, 2008.

[44] A. Pronobis and P. Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[45] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015.

[46] R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *Artificial Intelligence and Statistics*, pages 448–455, 2009.

[47] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[48] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4974–4983, 2017.

[49] A. Saxena, A. Jain, O. Sener, A. Jami, D. K. Misra, and H. S. Koppula. Robobrain: Large-scale knowledge engine for robots. *arXiv preprint arXiv:1412.0691*, 2014.

[50] T. J. Sejnowski. Higher-order boltzmann machines. In *AIP Conference Proceedings*, volume 151, pages 398–403. AIP, 1986.

[51] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(11):1778–1792, 2005.

[52] P. Singh, T. Lin, E. T. Mueller, G. Lim, T. Perkins, and W. L. Zhu. Open mind common sense: Knowledge acquisition from the general public. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 1223–1237. Springer, 2002.

[53] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, volume 5, page 6, 2015.

[54] R. Speer and C. Havasi. Conceptnet 5: A large semantic network for relational knowledge. In *The People's Web Meets NLP*, pages 161–176. Springer, 2013.

[55] E. Stopp, K.-P. Gapp, G. Herzog, T. Laengle, and T. C. Lueth. Utilizing spatial relations for natural language access to an autonomous mobile robot. In *Annual Conference on Artificial Intelligence*, volume 861, page 39. Springer Science & Business Media, 1994.

[56] M. Tenorth and M. Beetz. Knowrob-knowledge processing for autonomous personal robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4261–4266. IEEE, 2009.

[57] M. Tucker and R. Ellis. On the relations between seen objects and components of potential actions. *Journal of Experimental Psychology: Human perception and performance*, 24(3):830, 1998.

[58] E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin. The learning and use of traversability affordance using range images on a mobile robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1721–1726. IEEE, 2007.

[59] K. F. Uyanik, Y. Calskan, A. K. Bozcuoglu, O. Yuruten, S. Kalkan, and E. Sahin. Learning social affordances and using them for planning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35, 2013.

[60] X. Wang and E. Grimson. Spatial latent dirichlet allocation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1577–1584, 2008.

[61] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1331–1338. IEEE, 2011.

[62] W. Yeh and L. W. Barsalou. The situated nature of concepts. *The American journal of psychology*, pages 349–384, 2006.

[63] P. Zech, S. Haller, S. R. Lakani, B. Ridge, E. Ugur, and J. Piater. Computational models of affordance in robotics: a taxonomy and systematic classification. *Adaptive Behavior*, 25(5):235–271, 2017.

[64] Y. Zhu, A. Fathi, and L. Fei-Fei. Reasoning about object affordances in a knowledge base representation. In *European conference on computer vision*, pages 408–424. Springer, 2014.