UNIVERSITÀ DEGLI STUDI
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE
ICT International Doctoral School

Semantic Image Interpretation

Integration of Numerical Data and Logical Knowledge for Cognitive Vision

Ivan Donadello

Advisor

Luciano Serafini
Fondazione Bruno Kessler

Committee

Prof. Dr. **Marco Cristani**
Università degli Studi di Verona

Prof. Dr. **Claudia d'Amato**
Università degli Studi di Bari

Prof. Dr. **Tillman Weyde**
City, University of London

Dr. **Daniele Porello**
Libera Università di Bolzano

January 2018

To my parents

Abstract

Semantic Image Interpretation (SII) is the process of generating a structured description of the content of an input image. This description is encoded as a labelled direct graph where nodes correspond to objects in the image and edges to semantic relations between objects. Such a detailed structure allows a more accurate searching and retrieval of images. In this thesis, we propose two well-founded methods for SII. Both methods exploit background knowledge, in the form of logical constraints of a knowledge base, about the domain of the images. The first method formalizes the SII as the extraction of a partial model of a knowledge base. Partial models are built with a clustering and reasoning algorithm that considers both low-level and semantic features of images. The second method uses the framework Logic Tensor Networks to build the labelled direct graph of an image. This framework is able to learn from data in presence of the logical constraints of the knowledge base. Therefore, the graph construction is performed by predicting the labels of the nodes and the relations according to the logical constraints and the features of the objects in the image. These methods improve the state-of-the-art by introducing two well-founded methodologies that integrate low-level and semantic features of images with logical knowledge. Indeed, other methods, do not deal with low-level features or use only statistical knowledge coming from training sets or corpora. Moreover, the second method overcomes the performance of the state-of-the-art on the standard task of visual relationship detection.

Keywords
[Knowledge Representation, Computer Vision, Machine Learning, Information Extraction]

Acknowledgments

First and foremost, I want to thank my Ph.D. advisor Luciano Serafini, who has shown me and helped me to appreciate and adopt the mindset, the rigour and the commitment required for conducting quality research. By constantly working with him, I had the opportunity to learn many of the skills expected from a researcher.

Another important person for my Ph.D. is Artur d'Avila Garcez, who hosted and supervised me in the Research Centre for Machine Learning at City, University of London, during my visiting. I improved my research skills and acquired more knowledge about Machine Learning, and, more specifically in learning with logical constraints. The interaction with Artur d'Avila Garcez brought very important (and published) results for my Ph.D.

Silvana Badaloni was my supervisor during the Master Thesis. I want to thank her for introducing me Luciano Serafini and for giving me the opportunity to spread my research as invited speaker at the University of Padova.

During my Ph.D., I had interactions with other people whose suggestions and interactions fruitfully impacted my work. Francesco Corcoglioniti and Marco Rospocher were my first guides to the topic of Semantic Web. The former taught me many tricks, best practices and technicalities on Semantic Web programming. The latter was determinant for understanding ontologies and their management. The Laboratory for Applied Ontology (headed by Nicola Guarino) was important for the discussions about the ontological research part of my Ph.D. The discussions with Andrea Passerini and Stefano Teso brought a significant improvement of my knowledge on Machine Learning and optimization with logical constraints. They gave me important suggestions on both theoretical and technical aspects. I thank Enver Sangineto and Gloria Zen for their nice (and deep) introduction to the world of object detection.

I acknowledge the funding of my Ph.D. given by Fondazione Bruno Kessler that hosted me in the DKM group for the whole duration of the Ph.D. program. I thank the reviewers Prof. Dr. Marco Cristani and Prof. Dr. Tillman Weyde for reviewing this thesis and providing useful comments. Their feedback improves the overall presentation and makes

the work stronger.

For their feedback, technical advices, and support in any form, I want to thank my colleagues and friends (in alphabetical order): Greta Adamo, Loris Bozzato, Gaetano Calabrese, Elena Cardillo, Alessandro Daniele, Riccardo De Masellis, Mario De Nisi, Chiara Di Francescomarino, Shahi Dost, Mauro Dragoni, Marco Fossati, Chiara Ghidini, Radim Nedbal, Giulio Petrucci, Williams Rizzi, Andrea Rubin, Emilio San Filippo and Roberto Tiella.

I want to thank my childhood friends Chiara and Riccardo for their constant presence even at a long distance. Last but not least, I warmly thank my family and Annalisa for their love, for supporting me during these years and, especially, for their patience at the time of writing this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The universe of digital data is huge, growing exponentially and digital images follow this trend as well. Thousands of pictures for every single user, or organization, are stored in social networks, web sites, repositories, hard drives, personal computers, smartphones and other devices. This big amount of pictures convey a huge quantity of information about, for example, the tourist preferences of a group of users (holiday pictures posted on social media), how a plot of land changes during the years (pictures taken from satellites or drones), the market trend for a commercial product (pictures containing diagrams and reports of an enterprise) and the movements of a person (or group) in a particular place at a given hour (pictures taken from video surveillance cameras). Given a single picture, a person is able to understand the contained information with a relative small amount of time. However, as the number of picture is huge, automatic tools that analyse and extract the information conveyed by pictures, in an effective and efficient manner, are necessary.

The content of an image can be analysed with different levels of granularity, where each level describes the image with, more (or less) details. In the *first level* the analysis returns a coarse description of the whole image without entering into details. For example, the image can be classified with some labels that describe the global scene depicted in the image, such as "countryside", "green", "person", "horse". These labels can be obtained by analysing metadata or descriptions associated to the images, or by performing the so-called *scene classification*. However, the labels are associated to the whole image without the possibility to link the labels to regions in the image. In a *second level* of image analysis this linking is performed. Here the analysis focuses on discovering (and localizing) objects (with their labels of classification) in the image. Moreover, it is possible to discover some attributes of the localized objects, such as, the color, the shape, the age (if the object is a person), etc. In this manner, it is possible to know what are the objects in the

scene and where they are. Examples of this kind of analysis are: (i) the *object detection* that localizes objects within bounding boxes (a rectangle around the object); (ii) the *semantic segmentation* that associates every pixel a set of labels describing the object which the pixel belongs to. At this level of analysis, for example, it is possible to have the labels "countryside" and "green" that describe the whole image plus two bounding boxes classified with "person" and "horse", respectively. However, to achieve a complete understanding of the scene, it is also necessary to discover some relations between the objects. In a *third level* of analysis of the image content, a detailed labelling of the objects in the scene, with their attributes, and the relations between them is performed. At this level, some background knowledge is of great help as it allows for reasoning about the visual objects in the scene. This reasoning improves the results of the analysis. For example, if the image analysis discovers that a person rides a horse then it is possible to infer that the contrary is false (the relation ride is antisymmetric). We can also infer that there are some animals in the image (horses are animals), that the person is on the horse, the horse is under the person and is carrying him/her. At this level, the analysis of the image content is very detailed and it is possible to organize the discovered objects, attributes and relations within a structure. For example, the objects (and attributes) can be nodes of a graph and the relations between objects can be the edges between the corresponding nodes. A *fourth level* of analysis performs some high-level reasoning on the image objects and relations to infer what is happening in the scene, that is, what are the main events and the participants. This level refines the previous level by extracting higher-level information from the structured description of the image content. For example, the graph returned by the third level could contain only the relation "on" between a person and a horse. Hence, the fourth level predicts the event "riding" as the most plausible event with the person and the horse as participants. This prediction can be encoded in the graph by adding, for example, a node labelled with "riding event" with two edges joining the person and horse nodes. These edges are labelled with the roles of the participants at the event, for example the subject and object of the event. The *Semantic Image Interpretation* (SII) [51, 71] is the task of extracting such a detailed structure of the image content.

The SII enables a set of applications on an image content that a coarse image description (such that the one of the first layer) does not allow. Here we list some examples:

Accurate Image Retrieval A structured description of an image content that links the nodes of the structure with regions of the image allows the retrieval of portions of images according to a given query. For example, in a forensic application, such a structured description can retrieve all the (bounding boxes of) people with a weapon

in their hand shot during a riot.

Complex Image Querying A structured image description can be easily converted into a RDF graph [41]. This allows us to perform structured queries on images by using Semantic Web languages, such as SPARQL [30]. These queries can have a certain complexity, for example they can be the join of two simpler queries: we want to retrieve all (and only) the images showing a person riding a horse *and* in the middle of some buildings.

Robot Interaction A robot moving in an environment can find many configurations of objects and these objects enable the robot to perform a set of actions on them. For example, a cup on a table can be grasped but without hitting the table.

Visual Question Answering Such a structured image description can be inspected in order to answer natural language questions about the image content. For example, a question could be "Is there anyone riding an animal?", with answer "Yes, a person" and the involved bounding boxes.

Image/Video Captioning A structured description of an image can be easily converted (with a language generator model) into a natural language caption. This can be applied also to videos as they are a sequence of images (the frames).

The aim of this thesis is to study new algorithms and techniques in order to give an important contribution to the SII problem. However, the semantic interpretation of images presents some aspects that make the problem challenging:

Relational Domain The domain of images is *relational*, that is, the data (the labelled bounding boxes) are linked together through relations. For example, the pair of bounding boxes containing a person and a horse can be related through many predicates: $(person, ride, horse)$, $(person, on, horse)$, $(horse, under, person)$ and $(horse, carry, person)$.

Hybrid Domain The objects of this domain can be described with both *semantic* and *numeric features*. The semantic features are the labels describing the types of objects, for example, "person" and "horse". The numeric features can be, for example, the bounding box coordinates or the visual features extracted with Computer Vision techniques. The same holds also for the relations between objects. A relation is described with a set of labels (for example, "ride" and "on") and with numeric features, such as, the intersection area between the bounding boxes or their geometric distance.

Background Knowledge This domain can be described with some *background knowledge* about the input image. When a SII system analyses a picture, some important knowledge can be taken into account. For example, the fact that usually horses are ridden by people and horses do not ride. Moreover, background knowledge can impose constraints (such as, implication or mutual negation) between predicates: if a person rides a horse then the person is on the horse and, automatically, it cannot be under the horse.

1.1 Contributions

In this thesis we present two well-founded methods, with relative evaluation, to solve the SII problem. The first method extracts the structured description of an image content by combining an *unsupervised* algorithm with standard logical reasoning on the background knowledge. The second method learns and predicts this structured description in a *supervised* manner. The background knowledge imposes logical constraints on such predictions. The contributions of these methods involve the following SII aspects:

C1 Integration of Numeric and Symbolic Features Both methods deal with the numeric features coming from a low-level analysis of images and the semantic features used to describe the image content.

C2 Dealing with Uncertainty The low-level analysis of images returns data with uncertainty. For example, an object detector could not be sure about the classification of some objects in the image. Both methods address the uncertainty of the data.

C3 Integration of Logical Background Knowledge Both methods exploit background knowledge that can be found, encoded as logical constraints, in knowledge bases. Logical knowledge is very powerful as it expresses, with a standard logical language, many information and constraints about the domain of the images.

The improvement with respect to the state-of-the-art is that both systems integrate the above contributions. Indeed, the works that mainly deal with relational domains and uncertainty (Machine Learning-based approaches) do not use background knowledge in form of logical constraints. The only background knowledge used is knowledge about the statistical dependencies between labels of objects and predicates. This statistical knowledge is less expressive than logical knowledge and, in some works, is tailored to a training set. On the other hand, the works that deal with logical knowledge (logical-

based approaches) hardly deal with the uncertainty coming from a low-level analysis of the image.

1.2 Structure of the Thesis

The remainder of the thesis is structured as follows:

Chapter 2 This chapter provides the definition of the problem.

Chapter 3 This chapter provides the state-of-the-art on SII ranging from works on object detection to works on the construction of the graph that describes the image content. The background notions necessary to understand the thesis are not explained here but at the beginning of every chapter.

Chapter 4 This chapter provides a theoretical framework for SII that is used for implementing the first SII algorithm of the thesis. This chapter represents Contributions C3.

Chapter 5 This chapter provides the first (unsupervised) algorithm that mixes low-level and semantic features for SII. The chapter also describes the datasets used for the evaluation, the exploited knowledge bases, and finally the evaluation. This chapter represents Contributions C1, C2, C3.

Chapter 6 This chapter presents Logic Tensor Networks. This is a new framework that learns from data in presence of logical constraints. This chapter represents Contributions C1, C2, C3.

Chapter 7 This chapter provides how the SII problem has been encoded with Logic Tensor Networks. It also discusses some technical aspects of Logic Tensor Networks that emerge on the application to SII. This chapter represents Contributions C1, C2, C3.

Chapter 8 This chapter provides the evaluation of Logic Tensor Networks to SII. The chapter describes in details the considered datasets (and their statistics) and knowledge bases, the performed experiments, the obtained results along with a comparison with methods of the state-of-the-art. This chapter represents Contributions C1, C2, C3.

Chapter 9 This chapter deepens chapter 3 by comparing in details the algorithms presented in the thesis with similar methods of the state-of-the-art.

Chapter 10 This chapter provides a discussion of the presented algorithms. It summarizes the results of the thesis, discusses limitations and possible research directions.

Appendix A This appendix describes a demo that implements a SII system according to the algorithms developed in the thesis. Some screenshots are provided to show the output of the system.

1.3 Publications

The core publications supporting the present thesis are listed below:

- Ivan Donadello, Luciano Serafini, and Artur S. d'Avila Garcez. Logic tensor networks for semantic image interpretation. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1596–1602. ijcai.org, 2017

- Luciano Serafini, Ivan Donadello, and Artur S. d'Avila Garcez. Learning and reasoning in logic tensor networks: theory and application to semantic image interpretation. In Ahmed Seffah, Birgit Penzenstadler, Carina Alves, and Xin Peng, editors, *Proceedings of the Symposium on Applied Computing, SAC 2017, Marrakech, Morocco, April 3-7, 2017*, pages 125–130. ACM, 2017

- Ivan Donadello and Luciano Serafini. Integration of numeric and symbolic information for semantic image interpretation. *Intelligenza Artificiale*, 10(1):33–47, 2016

- Ivan Donadello. Ontology based semantic image interpretation. In Elena Bellodi and Alessio Bonfietti, editors, *Proceedings of the Doctoral Consortium (DC) co-located with the 14th Conference of the Italian Association for Artificial Intelligence (AI*IA 2015), Ferrara, Italy, September 23-24, 2015.*, volume 1485 of *CEUR Workshop Proceedings*, pages 19–24. CEUR-WS.org, 2015

- Ivan Donadello and Luciano Serafini. Mixing low-level and semantic features for image interpretation - A framework and a simple case study. In Lourdes Agapito, Michael M. Bronstein, and Carsten Rother, editors, *Computer Vision - ECCV 2014 Workshops - Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part II*, volume 8926 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2014

1.4 Artefacts

The core artefacts supporting the present thesis are listed below (in ascending order of publications):

A1 The knowledge bases and the datasets for the experiments in [27]: `https://dkm.fbk.eu/technologies/knowpic`.

A2 The source code for the paper in [92]: `https://gitlab.fbk.eu/donadello/LTN_ACM_SAC17/`.

A3 The source code for the paper in [28]: `https://gitlab.fbk.eu/donadello/LTN_IJCAI17`.

A4 A demo that performs the semantic interpretation of an input image provided by the user. The demo first recognizes the objects in the image with an object detector and then classifies the semantic relationships between them.

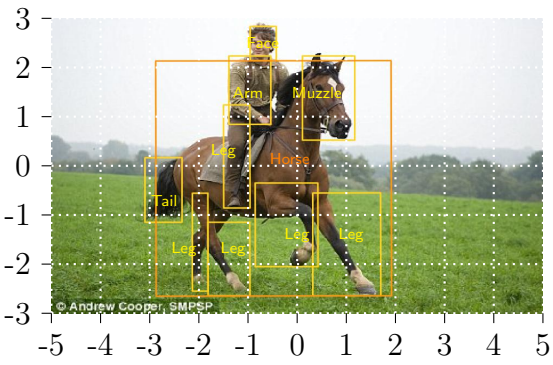
Chapter 2

The Problem

Semantic Image Interpretation (SII) is the task of generating a structured description of the content of images [51, 71, 55]. SII is much more than image classification [56], that is, the description of image content with a set of labels, SII aims at detecting the objects in images, their types, attributes and the relations between them. SII produces a semantically rich structure (for example, a logical model of a logical theory) that is both human understandable and processable by machines. Before defining the properties of such a structure we define what is the input of a SII system. As inputs we consider a background knowledge and a *semantically labelled picture*. To better explain our proposal, we use the simple running example of Figure 2.1.

The first input is the background (logical) knowledge. In recent years, it became clear that background knowledge about image context and content plays a key role in SII [103, 110]. Examples of useful background knowledge are: knowledge about objects qualities, for example, color, shape, relative size; knowledge about topological and spatial properties of objects, for example, the context where objects usually appear, the relative position, where an object is likely to be; relational knowledge, for example, the parts of complex objects or which objects can perform a certain action; taxonomical knowledge, that is, hierarchies of object types. As background knowledge we consider *knowledge bases* encoded with a logical language, for example First-Order Logic [8] or Description Logic [5]. In Figure 2.1, the background logical knowledge is represented with Description Logic and lists knowledge about the parts of objects, if some objects can ride or be ridden and a hierarchy of object types. The logical language of a knowledge base is characterized by a *signature* Σ that is a set containing non-logical symbols, such as, symbols for predicates, functions and constants. Knowledge bases are nowadays largely available in the form of RDF resources and OWL ontologies, with the spread of the Semantic Web and Linked

Open Data. Examples of available logical knowledge are WordNet [34], YAGO [66], ConceptNet [100], Cyc [59] and DBpedia [3].



Person	$\sqsubseteq$	$(= 1)$hasParts.Face
		$\sqcap\ (= 2)$hasParts.Leg
		$\sqcap (= 2)$hasParts.Arm
Horse	$\sqsubseteq$	$(= 1)$hasParts.Muzzle
		$\sqcap\ (= 4)$hasParts.Leg
		$\sqcap\ (= 1)$hasParts.Tail
Horse	$\sqsubseteq$	$\forall$hasPart.$(\neg$Face$)$
		$\sqcap\ \forall$hasPart.$(\neg$Arm$)$
Person	$\sqsubseteq$	$\forall$hasPart.$(\neg$Muzzle$)$
		$\sqcap\ \forall$hasPart.$(\neg$Tail$)$
Horse	$\sqsubseteq$	$\forall$ride.$\bot$
Person	$\sqsubseteq$	$\forall$ride^{-}.$\bot$
Leg	$\sqsubseteq$	Limb
Arm	$\sqsubseteq$	Limb

Figure 2.1: A semantically labelled picture and a Description Logic knowledge base that encodes the background knowledge.

The other input of a SII system is the picture to analyse. We assume that the picture has been preprocessed with a low-level analysis for having some proposals of the main objects in the image. This analysis is performed with the semantic segmentation [64] or with the object detection [36]. Semantic segmentation assigns every pixel in the image a set of labels with a weight: the labels describe the object that the pixel belong to (for example, a horse, a person or a head), the weights are the confidence scores of the semantic segmentation tool about the returned labels. Object detection (see Chapter 3), instead, detects objects in the image and represents them with bounding boxes: a rectangle around the detected object with a set of labels and weights. The labels describe the type of object in the bounding box and the weights are the confidence scores of the returned labels. Notice that, these labelled regions of pixels can be further processed by the SII system for a refinement. For example, the refinement could change the labels of some pixels or the coordinates of a bounding box. The other input of a SII is a *semantically labelled picture*:

Definition 1 (Semantically Labelled Picture)**.** *Let $S = \{s_1, \ldots, s_n\}$ be a set of segments (a segment is a set of contiguous pixels) returned by a low-level analysis of picture $\mathcal{P}$, and let Σ the signature of a logical language. A* semantically labelled picture *is a pair $\mathcal{P} = \langle S, L \rangle$, where L is a function that associates each segment $s \in S$ a set $L(s) \subseteq \Sigma \times (0, 1]$ of weighted labels $\langle l, w \rangle$.*

In this thesis we adopt the object detection as low-level technique of analysis of the input image because it is pretty forward to extract proposals for the objects. The labels associated to the bounding boxes are taken from the signature Σ used to specify the background knowledge. The semantically labelled picture of Figure 2.1 contains bounding boxes for the main objects in the image but some are missing. For example, there is no bounding box containing the whole person, neither for the grass, the sky and the trees in background. Notice that, if we search, at this stage, for a picture containing a person that is riding a horse, the picture will not be returned. The goal of SII is to build a semantic structure that contains also the presence of the non-recognized person and the fact that he is riding the recognized horse.

Given an input image, we define the structured description of the image content as a labelled direct graph with nodes and edges. A labelled node corresponds to segments of the input image containing an object. The labels of the node describe the corresponding object in the image: they can be attributes of the object (for example, "brown") or classification labels (for example, "horse"). A labelled direct edge starts from a subject node, ends in a object node and its labels (for example, "ride") describe a relation between the corresponding objects in the image. This graph is also called *scene graph* [55]. Moreover, nodes of the graph need to be aligned (that is, linked with) with the segments in the input image. That is, for every node of the graph, a SII system has to return the corresponding pixels in the image. This is necessary for the accurate image retrieval and visual question answering tasks mentioned in the previous chapter. We call this data structure *semantically interpreted picture*. Figure 2.2 shows the semantically interpreted picture of the running example.

It is worth to discuss which is the level of detail (or granularity) the SII has to account for. The input knowledge base can contain information ranging from general objects and their relations to very detailed properties of objects, such as the possible texture of clothes dressed by people. For example, a very detailed knowledge base could contain information that distinguishes pois texture from others. However, recognizing such low-level details of an image could affect the effectiveness of a SII system, as these details can be affected by noise (for example, deformation of the tiny circles) that falsify the statements in the knowledge base. Our choice is to apply the semantics at a proper level of detail. Therefore,

we left the detection of some objects and and their properties (for example, the shape, the material or the texture) to robust Computer Vision techniques, such as the object detection, that *learn* the semantics of some objects and their properties. Whereas the information contained in the knowledge base is used to discover relations between objects or to refine the output of the Computer Vision analysis. For example, the knowledge base could correct a detected pois texture on a horse.

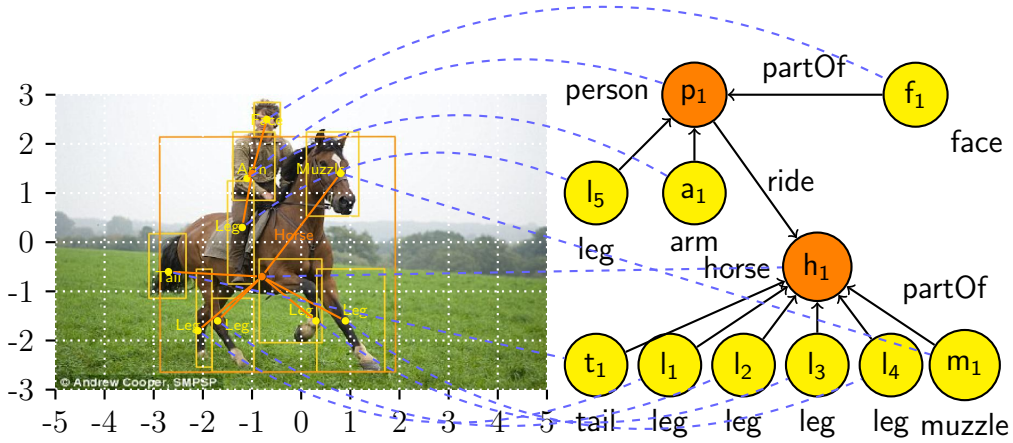


Figure 2.2: A semantically interpreted picture: the output of SII. The labels inside the nodes are identifiers.

The main challenge in developing a system for SII for building a semantically interpreted picture is bridging the so called *semantic gap* [51], which is the lack of a direct correspondence between the low-level features of an image and the high-level semantic concepts that a user adopts to interpret the picture. To address this problem the following research questions need to be posed:

1. what is an effective encoding for dealing with both low-level and semantic features of the image?

2. How can a SII system change (or discard) the proposals of an object detector if they disagree with other proposals?

3. How can a SII system infer the presence of some objects (or relations) with only few information coming from a low-level image analysis?

4. How can a SII system label (or discard) the nodes and edges of a semantically interpreted picture such that the labelling satisfies the information of a knowledge base?

Chapter 3

State of the Art

The previous chapter defines the Semantic Image Interpretation as the procedure of constructing a labelled graph, also called semantically interpreted picture or scene graph [55], that describes the semantic content of an input image. The labelled nodes represent objects in the image, the labelled edges represent semantic relations between objects. This review lists the main works that construct such a graph. These works are between two Artificial Intelligence communities: the Computer Vision and the Knowledge Representation community. These works can be divided into three groups:

Object Detection This group does not properly contain SII works but the main approaches necessary for extracting the basic components of a SII labelled graph: the nodes, that is, the objects in the image.

Visual Relationship Detection This group lists important works that build a labelled SII graph by predicting a set of visual relationships. A visual relationship is a relation between two objects in the image. In literature, a visual relationship is described as a triple $\langle subject, predicate, object \rangle$ where *subject* is the label of the node with an outgoing edge e, *predicate* is the label of the edge e and *object* is the label of the node receiving the edge e. If we refer to the running example of Figure 2.2, examples of visual relationships are: $\langle \mathsf{Person}, \mathsf{ride}, \mathsf{Horse} \rangle$, $\langle \mathsf{Leg}, \mathsf{partOf}, \mathsf{Person} \rangle$ and $\langle \mathsf{Muzzle}, \mathsf{partOf}, \mathsf{Horse} \rangle$. The main idea underlying this set of works is to consider two nodes in the scene graph, and classify them according to a set of relationships. This approach performs a *local choice* by classifying, each time, the relationships between only two objects in the input image.

Direct Graph Construction The works in this group construct the scene graph as a whole structure and not by classifying its parts separately. Here a prediction (of a

labelled node or edge) takes into account also the surrounding nodes, that is, the contextual information.

3.1 Object Detection

Object detection provides the labelled nodes of a scene graph. That is, the main objects that can be found in a picture. The literature on object detection is huge and a detailed review is out of the scope of this presentation. For this reason, we list the most relevant works. A first processing of object detection is given by the extraction of the *object proposals*. These are a set of non classified bounding boxes that may potentially contain an object. There is a huge literature on object proposal methods and comprehensive surveys can be found in [49, 14]. Most of the object proposal works can be divided in two groups: (i) those based on grouping super-pixels (for example, Selective Search [106], Constrained Parametric Min-Cuts (CPMC) [13], Multiscale Combinatorial Grouping (MCG) [76]) and (ii) those based on sliding windows (for example, objectness in windows [1], EdgeBoxes [115]). Object proposal methods can be adopted as image preprocessing tools for object detectors. These take the input proposals and classify (or discard) them with some labels [46, 114, 104, 45, 101, 21, 58, 36, 37].

The real improvement of object detection performance is due to the use of Deep Learning techniques. These approaches are the state-of-the-art of object detection. OverFeat [93] gives one of the first improvements on object detection with deep learning. It is based on a multi-scale sliding window implemented with a Convolutional Neural Network (CNN) [56]. Regions with CNN features or R-CNN [37] uses object proposals computed with Selective Search. Here, a CNN extracts features for every proposal, these features are then processed by Support Vector Machines to classify each proposal. This method reached almost the 50% of improvement on the PASCAL VOC challenge [33] on object detection. A drawback of the method is the speed: for every proposal a CNN forward pass is performed for feature extraction. Fast R-CNN [36] overcomes this issue by first computing a convolutional feature map for the whole image. Then, it classifies each object proposal using a feature vector extracted from the feature map. The classification is performed with a set of fully connected layers that compute a softmax probability over the object classes. Faster R-CNN [83] is a further evolution of the previous works. It substitutes the use of object proposal algorithms with a Region Proposal Network (RPN) that computes the proposals starting from the feature map computed by the CNN. This proposals are then classified as in Fast R-CNN. YOLO [79] and SSD [63] have a simpler, but effective, architecture: a single CNN on the whole image detects and classifies

the bounding boxes. Other works are the so-called *part-based models*. They leverage the part-whole relation between whole objects and their parts to improve the object detection [17, 32, 35, 38, 32, 70]. We review some of them in the related work in Chapter 9.

3.2 Visual Relationship Detection

In [87] the notion of *visual phrase* is proposed as a prototype of visual relationship. The difference is that a visual phrase is associated to only a single bounding box that contains both subject and object. The work shows that the detection of visual phrases improves the detection of the single subjects and objects. The approach is based on training an object detector for every possible triple $\langle subject, predicate, object \rangle$. This suffers of scalability issues as the number of categories for the subjects/objects and the predicates grows. Closely related to the visual relationship detection is the visual semantic role labelling [42, 85, 108]. This task extracts from pictures a set of tuples such as: $\langle predicate, \{\langle role_1, label_1 \rangle, \ldots, \langle role_N, label_N \rangle\}\rangle$, where the roles represent some entities involved by the predicate such as agent, source, tool or place. A visual relationship is a tuple with only the subject and the object as roles. In [42] the authors propose a dataset and baselines based on object detection and regression of bounding boxes of roles involved in a predicate. The roles are limited to subject (performing the action), object and instrument/tool. A limitation is that the role of subject is assigned only to people.

Many works exploit deep learning techniques [88] for visual relationship detection. For example, in [24] the visual relationships are detected with a novel (and designed specifically) Deep Relational Network that exploits the statistical dependencies between relationships and the involved objects. In [61] the use of a deep reinforcement learning framework to detect visual relationships and attributes of objects is exploited. The authors of [60] propose a message passing algorithm to share information (and reasoning) among neural networks that encode the subject, the object and the predicate of a visual relationship. In [112] the visual relationship detection is related to the similarity of the subject and object in the same vector space. Indeed, the difference between the embedding vectors of the subject and the object gives information on the relationship between them.

The visual relationship detection can be improved by exploiting some background knowledge about the domain [78, 65, 6, 113, 73]. In [78] the background knowledge consists in logical constraints between the visual relationships (for example, implication or mutual exclusivity). The visual relationships are predicted with a Neural Network that maps, in the same embedding space, both visual information and constraints. In [65]

a visual relationship is predicted with a score that combines the background knowledge with the visual information. The former is a pre-trained word embedding (word2vec) [67] of the subject and object labels, such that similar triples are close in the embedding space. The latter consists in the visual features of the union of the subject and object bounding boxes (computed with a CNN). The work in [6] combines background knowledge with visual information in the same manner of [65]. The difference is that the considered knowledge is statistical information about the triples in the training set. For example, the likelihood that a wheel is part of a car. This knowledge is learnt with statistical link prediction methods [72]. In [109] the background knowledge (taken from the training set and Wikipedia documents) is encoded as a probability distribution of a relationship given the labels of the subject and the object. During the training phase, this knowledge drives the learning of a fully connected neural network that predicts visual relationships. Other approaches encode background knowledge with visual features in probabilistic graphical models. In [113, 73], visual features are combined with knowledge gathered from datasets, web resources or annotators, about object labels, properties (for example, shape, colour, size) and affordances, using a Markov Logic Network (MLN) [84]. This allows for querying of the MLN and thus to predict visual relationships in unseen images. Due to the specific knowledge-base schema adopted, the effectiveness of MLNs in this domain is evaluated only for Horn clauses, although the language of MLNs is more general.

3.3 Direct Graph Construction

Several works build a semantically interpreted picture by considering its labelled graph as a whole and not only its single components (the visual relationships). Indeed, the surrounding context can help the recognition of both objects and relations between them. For example, suppose we have bounding boxes for a horse, grass and a person (this last one with a low score). The presence of a horse and grass increases the likelihood to have a person that rides (or walks with) the horse that is on the grass. The first works that follow this intuition come from the Knowledge Representation community and they formalize the labelled graph of the SII as an interpretation of a logical language. Indeed, the scene graph is a set of logical statements in a logical language, for example, $\mathsf{Horse}(\mathsf{h}_1)$, $\mathsf{Person}(\mathsf{p}_1)$, $\mathsf{ride}(\mathsf{p}_1, \mathsf{h}_1)$. The scene graph is built with logical reasoning. The first work that formalizes the SII graph construction as a reasoning task in First-Order Logic (FOL) is in [82]. This approach assumes that the basic objects in the picture, along with their spatial relations, are already identified by some low-level image analysis (for example, from the object detection). Then, with logical reasoning on these basic facts, the complete

description of the image content is derived. However, such a complete description cannot be obtained for a picture. In [89] the picture is represented with the notion of *partial model* of a knowledge base. The generation of a semantically interpreted picture is performed with pure logical reasoning but low-level features of the image are not considered. In [71] the low-level image features are included in a Description Logic (DL) [5] knowledge base along with DL axioms. These axioms represent the connection between semantic types and low-level features (via data properties and concrete domains). For example, the standard dimension of a plate is formalized with a constraint on the data property size of the concept of plate. The graph construction is derived via deductive reasoning and a notion of *preference* between partial models is considered. However, writing axioms that map low-level features in concepts and relations could suffer of scalability (high engineering effort) as the number of concepts and relations grows and dealing with the noise coming from object detectors could be problematic. A different approach to the scene graph construction is based on abductive reasoning [74]. This technique infers the preferred partial model (explanation) starting from the observations coming from a low-level image processing (object and spatial relation detection). The preferred partial model of an image is the one that contains more evidence and less hypotheses. However, the method requires a set of DL rules for defining the abducible space, which need to be manually crafted. Other logic-based approaches use fuzzy DL to deal with the uncertainty coming from the object detection [50, 2]. These approaches limit themselves to spatial relations or to refine the labels of the detected objects. In [50] the authors proposed a fuzzy DL ontology of spatial relations and an algorithm for building scene graphs. The scene graph is constructed starting from some basic objects in the scene, then, through logical reasoning, semantic relations between objects (or new objects) are inferred. This method is extended in [2] where morphological reasoning [29] is applied to the objects discovered by the low-level image analysis.

Other methods for global SII graph construction are based on the minimization of an energy [19, 57, 17, 15] or a loss [31, 107] functions. For example, these methods start from a complete graph whose nodes and edges need to be labelled or discarded. These methods are mainly adopted by the Computer Vision community. In [19] the scene graph is built through energy minimization of a graphical model. Here the graph describes the spatial layout of an indoor scene. However, the relations between objects do not encode an explicit semantics of $\langle subject, predicate, object \rangle$ but rather they are spatial configuration of objects in an indoor space. In [57] the graph is encoded with a Conditional Random Field [54] and its best labelling (nodes are objects and attributes, edges relationships) is found through energy minimization. Potentials are defined by combining the object

detection score with geometric relations between objects and with text priors on the types of objects. Also in [17] the scene graph is encoded with a CRF but it is limited to the part-whole relation. Indeed, the nodes are whole objects and their parts, the edges are (implicitly) labelled with the part-whole relation. The aim is to leverage the par-whole relation to improve object detection performance. In [15] the energy function of the CRF combines visual information coming from the object detection with logical constraints of a DL knowledge base about the domain. In [31] the graph is predicted by finding the maximum spanning tree, according to a scoring function, from a complete weighted graph of the bounding boxes. The edges are labelled with spatial relationships defined with rules. The other relations (such as, driving, riding, playing, using, sitting, wearing) are predicted with a conditional probability on the subject, object and the spatial relation. However, this kind of prediction can introduce ambiguity: a person next to a dog can play with it or simply walking it. In addition, the approach is limited to five 2D-spatial relationships (for instance, relations such as behind or front-of are not considered) with effort for defining the rules. In [107] the score of the scene graph is maximized with an iterative message passing algorithm. For each iteration, the information about the nodes (labels and features) is passed for maximizing the likelihood of the relationships. The information about the relationships is passed for maximizing the likelihood of the nodes until a given number of iterations has reached. In this manner, the prediction of a single element (node or relation) takes into account the information of the other elements.

Chapter 4

Semantic Image Interpretation as a Ranking of Partial Models

In this chapter, a formal, and logic-based framework for Semantic Image Interpretation (SII) is described. The first observation is that an image is a partial view of the world. That is, the scene in a image can be cluttered and some objects cannot be visible or satisfy our background knowledge. For example, due to occlusions we can see only one leg of a person riding a horse. However, we are able to reason about this partial view, find an explanation to the lack of the other leg of the person and deduce that the partial view satisfies our background knowledge. With this intuition we formalize a semantically interpreted picture as a partial model of an input knowledge base, see Figure 2.2, that is, a logical interpretation whose completion satisfies the knowledge base. Moreover, many partial models exist for an input image, thus there is the need to find the one that best matches the image content. This is formalized with a cost function that assigns a cost to partial models. This function measures the (dis)agreement between the low-level features of our input image and the high-level semantic features contained in the partial model. Therefore, a semantically interpreted picture is a partial model that minimizes a cost function. As input, we assume to have a semantically labelled picture and a Description Logic knowledge base. The single segments of the input picture are labelled with the symbols in the signature Σ of the knowledge base, see Definition 1 and Figure 2.1.

The following section provides an overview of Description Logics. The next section defines our theoretical framework to SII, that is, the searching of a partial model that minimizes a cost function. All the explanations take into account the running example in Figure 2.1.

4.1 An Overview of Description Logics

Description Logics (DLs) [5] are one of the main formalisms for knowledge representation. DL languages have been widely used, from the middle of the '80s, for knowledge representation [5] and ontology engineering [39]. Moreover, they are an important underpinning for the OWL web ontology language as the World Wide Web Consortium (W3C) standardized.

DLs stemmed from early days of knowledge representation formalisms (late '70s, early '80s) such as *semantic networks* and *frame-based systems*. The former are graph-based formalisms for representing the meaning of sentences. The latter are data structures used to divide knowledge into substructures by representing "stereotyped situations". They can be considered as antecedents of object-oriented languages. Both are comprehensible and intuitively readable but they lack of a formal semantics. DLs were developed to overcome these limitations by providing (i) a clear semantics and (ii) inference techniques. The DLs *formal semantics* provides an unambiguous meaning to sentences of the DL language and allows a common grounding for humans and interoperability between humans and machines. Moreover, semantics makes possible to define a correct and complete *logical deduction* for inferring new information from facts of a knowledge base. This feature distinguishes DLs from other representation languages, such as the Unified Modelling Language (UML) [11] and the Entity-Relationship model language [16]. The process of computing inferences is called *reasoning*, and a computationally efficient reasoning algorithm is an important feature of a DL. This is one of the reasons about the existence of many description logics: the higher the expressivity of the language the higher the computational cost for reasoning. The best balance between them depends on the application. In the following we describe: (i) the syntax (with basic modelling notions), (ii) the semantics of a particular, and highly expressive, DL: $\mathcal{SROIQ}$ [48], and (iii) a brief overview of reasoning in DLs.

4.1.1 Syntax of Description Logics

All DLs provide the basic building blocks for modelling entity types and relationships between entities in a domain of interest. We start with the notion of *signature* $\Sigma = \Sigma_I \uplus \Sigma_C \uplus \Sigma_R$, or vocabulary, of a DL that is the set union of three finite and disjoint sets of symbols: the *individual names* Σ_I, the *concept names* Σ_C and the *role names* Σ_R. Individual names represent single individuals of our domain whereas concept names represent the types of the individuals, which extensively correspond to the set of individuals of such a type. Role names represent binary relationships between individuals.

In the standard translation of DL in FOL [12] individual names correspond to constants, concepts correspond to unary predicates and roles to binary predicates. If we want to model the scene in a picture, for example the one in Figure 2.1, we use two individual names, $\mathsf{john}, \mathsf{furia} \in \Sigma_I$, for representing the person and the horse, the concept names $\mathsf{Horse}, \mathsf{Person} \in \Sigma_C$ to represent their types and the role name $\mathsf{ride} \in \Sigma_R$ to represent the relation of a person riding a horse. In this presentation, we focus on the syntax of one of the most expressive DLs: $\mathcal{SROIQ}$ [48]. Its high expressiveness is determinant for reasoning in a domain such as image interpretation. Indeed, in SII, some roles can be transitive ($\mathcal{S}$) such as the role $\mathsf{right\ of}$. Some roles can include others ($\mathcal{R}$), for example the role on includes the role ride. There can be the presence of nominals ($\mathcal{O}$), for example a trained object detector could detect the presence of the individual Barack Obama. A role can be the inverse of another one ($\mathcal{I}$), for example the role $\mathsf{right\ of}$ is the inverse of $\mathsf{left\ of}$. Finally, concept and role names can be related through *qualified number restrictions* ($\mathcal{Q}$), for example we know a priori that horses have exactly four legs. A $\mathcal{SROIQ}$ *concept* is an expression defined by the following grammar:

$$\begin{aligned} C, D \;:=\; & A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \top \mid \bot \mid \exists R.C \mid \forall R.C \mid \\ & (\geq n)R.C \mid (\leq n)R.C \mid \exists R.Self \mid \{a_1, ..., a_n\} \end{aligned}$$

with $A \in \Sigma_C$, $R \in \Sigma_R$, and n is a non-negative integer. We assume that Σ_R is closed under inverse role, that is, if $R \in \Sigma_R$ then R^- (the inverse of R, see Table 4.1) is in Σ_R. In addition, Σ_R contains also the *universal role* U that always relates all pairs of individuals. We briefly describe the grammar:

- every concept name $A \in \Sigma_C$ is a concept expression;
- if C, D are concept expressions then also the *negation* $\neg C$ is a concept expression. For example, the concept Male can be defined as $\neg\mathsf{Female}$. The *disjunction* $C \sqcap D$ of two concepts is a concept expression. For example, the expression $\mathsf{Horse} \sqcap \mathsf{Brown}$ defines the concept of horses that have have brown fur. The *conjunction* $C \sqcup D$ of two concepts is a concept expression. For example, the expression $\mathsf{Horse} \sqcup \mathsf{Person}$ defines a concept (such as Animal in our picture domain) that includes people and horses.
- $\top$ (*top concept*) and $\bot$ (*bottom concept*) are concept expressions. They are syntactic sugars for the concept expressions $C \sqcup \neg C$ and $C \sqcap \neg C$, respectively.
- If $R \in \Sigma_R$ and C is a concept expression, then the *existential quantification* $\exists R.C$ is a concept expression. For example, the expression $\exists\mathsf{hasPart}.\mathsf{Tail}$ indicates all the

entities with a tail. The *universal quantification* $\forall R.C$ is also a concept expression. For example, the expression $\forall \mathsf{eat}.(\mathsf{Grass} \sqcup \mathsf{Vegetable})$ indicates all the entities that eat only grass or vegetables, such as horses.

- if $R \in \Sigma_R$, n is a non-negative integer and C is a concept expression, then $\exists R.Self$ (*self restriction*), $(\geq n)R.C$ (*at-least restriction*), $(\leq n)R.C$ (*at-most restriction*) and $(= n)R.C$ (*exact restriction*) are also concept expressions. The latter three are also known as qualified number restrictions, or cardinality constraints, and impose a constraint on the cardinality of the quantifier. For example, the expression $(= 4)\mathsf{hasPart.Leg}$ indicates all the entities with exactly four legs, that is the quadrupeds.
- $\{a_1, ..., a_n\}$ (*nominal concepts*) is a concept expression for every finite set $\{a_1, ..., a_n\} \subseteq \Sigma_I$ of individual names, for example, $\{\mathsf{john}, \mathsf{furia}\}$.

As DLs are formal languages it is possible to create statements called axioms. The axioms state information about our domain and are collected into three sets: *assertional axioms* (ABox $\mathcal{A}$), *terminological axioms* (TBox $\mathcal{T}$) and *relational axioms* (RBox $\mathcal{R}$). A DLs *knowledge base* $\mathcal{KB}$ is the union of these sets of axioms. ABox axioms encode factual knowledge about individuals, such as, the *(negated) concept assertions*:

$$\mathsf{Person}(\mathsf{john}), \neg\mathsf{Horse}(\mathsf{john}), \neg\mathsf{Person}(\mathsf{furia}), \mathsf{Horse}(\mathsf{furia}),$$

stating that the individual name john is an instance of the concept Person and not an instance of Horse. The individual name furia is an instance of Horse and not of Person. The *(negated) role assertions* describe relations between the individuals, such as, the individual john is riding the individual furia and not vice versa:

$$\mathsf{ride}(\mathsf{john}, \mathsf{furia}), \neg\mathsf{ride}(\mathsf{furia}, \mathsf{john}).$$

ABox axioms state also the equality or not between individual names. This is necessary due to the fact that in DL a single individual of the domain can be represented by many individual names. For example, the *individual inequality* assertion $\mathsf{julia} \not\approx \mathsf{john}$ states that Julia and John are different individuals, whereas the *individual equality* assertion $\mathsf{johnny} \approx \mathsf{john}$ states that Johnny and John are the same individual. On the other hand, the axioms in the TBox assert relationships between concepts, such as the *concept inclusion*. For example:

$$\mathsf{Horse} \sqsubseteq \mathsf{Animal}$$

states that all the individuals of type horse are also animals. The *concept equivalence* states that two concepts have the same individuals:

$$\mathsf{Human} \equiv \mathsf{Person}.$$

TBox axioms define concepts in terms of set theoretic combinations of other concepts, according to a specific DL syntax, see the grammar of $\mathcal{SROIQ}$ defined above. Finally, RBox axioms encode properties of roles such as the *role inclusion*. For example:

$$\mathsf{ride} \sqsubseteq \mathsf{on}$$

states that every pair of individuals related with the ride role are also related with on. The *role equivalence* axiom states that two roles have the same pairs of individuals, for example $\mathsf{above} \equiv \mathsf{over}$. Another RBox axiom is the *complex role inclusion* that relates roles through their composition and inclusion. For example, it is possible to concatenate the carry and hasPart roles to state that every time a subject carries an object then the subject carries also the parts of the object:

$$\mathsf{carry} \circ \mathsf{hasPart} \sqsubseteq \mathsf{carry}$$

Finally, the *role disjointness* axioms state that two different roles cannot share pairs of individuals. For example, it is not possible that John is on and below the horse at the same time: $Disjoint(\mathsf{on}, \mathsf{below})$. The defined axioms can be resumed in the following table:

ABox axioms		**TBox axioms**	
$C(a)$	concept assertions	$C \sqsubseteq D$	concept inclusion
$\neg C(a)$	negated concept assertions	$C \equiv D$	concept equivalence
$R(a, b)$	role assertions		
$\neg R(a, b)$	negated role assertions		
$a \approx b$	individual equality		
$a \not\approx b$	individual inequality		

RBox axioms	
$R \sqsubseteq S$	role inclusion
$R \equiv S$	role equivalence
$R \circ S \sqsubseteq Q$	complex role inclusion
$Disjoint(R, S)$	role disjointness

4.1.2 Semantics of Description Logics

The syntax of a DL describes only the correct syntactic statements that can be expressed in that DL but it does not state anything about the *meaning* of these statements. The *semantics* of DLs provides a formal meaning for DLs concept expressions and axioms. The central notion of DLs semantics is the definition of *interpretation*, that allows us to compute the set theoretic meaning of the complex concept expressions and the truth value

(true or false) of an axiom. An interpretation $\mathcal{I}$ of Σ is a pair $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set, called the *interpretation domain*, that can be conceived as the collection of individuals, or things, that exist in the "world" that $\mathcal{I}$ represents. The function $\cdot^{\mathcal{I}}$, called *interpretation function*, grounds the symbols in the signature Σ (individuals, concepts and roles) in the elements of $\Delta^{\mathcal{I}}$ according to the following rules:

- individual names are interpreted as elements of the domain, formally $\cdot^{\mathcal{I}} : \Sigma_I \to \Delta^{\mathcal{I}}$;
- concept names are interpreted as subsets of the domain, formally $\cdot^{\mathcal{I}} : \Sigma_C \to 2^{\Delta^{\mathcal{I}}}$;
- role names are interpreted as binary relations, formally $\cdot^{\mathcal{I}} : \Sigma_R \to 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}}$;

An interpretation is a *complete* abstract description of the state of the world in terms of existing objects (that is, the elements of $\Delta^{\mathcal{I}}$), object types (that is, the interpretations via $\cdot^{\mathcal{I}}$ of the symbols in Σ_C) and relations between objects (that is, the interpretations of the symbols in Σ_R). By now, the defined interpretation determines only the semantics of the symbols in the signature Σ. However, to compute the truth value of a DLs axiom it is necessary to extend the interpretation function $\cdot^{\mathcal{I}}$ also to complex ($\mathcal{SROIQ}$) concepts and roles. This extension is rather intuitive, for example the interpretation of $\mathsf{Horse} \sqcap \mathsf{Brown}$ (see above) is the intersection of the interpretation of Horse and Brown, see Table 4.1. We can now define the truth value (true or false) of an axiom according to an interpretation. An axiom ϕ is *satisfied* by $\mathcal{I}$ (or is true under $\mathcal{I}$), in symbols $\mathcal{I} \models \phi$, if the corresponding condition in Table 4.2 is met. If $\mathcal{I}$ satisfies all the axioms in a knowledge base $\mathcal{KB}$, in symbols $\mathcal{I} \models \mathcal{KB}$, we say that $\mathcal{I}$ is a *model* for $\mathcal{KB}$. We say that $\mathcal{KB}$ is *consistent* (or *satisfiable*) if it is satisfied by at least one model. An axiom ϕ is a *logical consequence* of $\mathcal{KB}$ (or $\mathcal{KB}$ entails ϕ, in symbols $\mathcal{KB} \models \phi$) if ϕ is true in every model of $\mathcal{KB}$. The axioms of the knowledge base are constraints on the states of the world. For instance, the axiom $\mathsf{Horse} \sqsubseteq (= 4)\mathsf{hasPart.Leg}$ states that every horse has exactly four legs. This implies that worlds where horses have only three or five legs are impossible.

4.1.3 Reasoning in Description Logics

Semantics provides a meaning to a $\mathcal{KB}$ symbols and defines what a logical consequence between axioms is, but it does not state how to compute (if possible) the entailments. This computation is called *reasoning* and one great advantage of DLs is that many reasoning tasks are decidable. We describe some important reasoning tasks:

Knowledge Base Satisfiability A knowledge base $\mathcal{KB}$ is satisfiable if there exists an interpretation $\mathcal{I}$ that satisfies all the axioms in $\mathcal{KB}$, in symbols $\mathcal{I} \models \mathcal{KB}$. This task

	Syntax	**Semantics**
Individuals:		
Individual name	a	$a^{\mathcal{I}}$
Concepts:		
Atomic concept	A	$A^{\mathcal{I}}$
Intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Complement	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Top concept	$\top$	$\Delta^{\mathcal{I}}$
Bottom concept	$\bot$	$\emptyset$
Existential restriction	$\exists R.C$	$\{d \in \Delta^{\mathcal{I}} \mid \text{for some } \langle d, d' \rangle \in R^{\mathcal{I}}, d' \in C^{\mathcal{I}}\}$
Universal restriction	$\forall R.C$	$\{d \in \Delta^{\mathcal{I}} \mid \text{for all } \langle d, d' \rangle \in R^{\mathcal{I}}, d' \in C^{\mathcal{I}}\}$
At-least restriction	$(\geq n)R.C$	$\{d \in \Delta^{\mathcal{I}} \mid \#\{d' \in C^{\mathcal{I}} \mid \langle d, d' \rangle \in R^{\mathcal{I}}\} \geq n\}$
At-most restriction	$(\leq n)R.C$	$\{d \in \Delta^{\mathcal{I}} \mid \#\{d' \in C^{\mathcal{I}} \mid \langle d, d' \rangle \in R^{\mathcal{I}}\} \leq n\}$
Exact restriction	$(= n)R.C$	$\{d \in \Delta^{\mathcal{I}} \mid \#\{d' \in C^{\mathcal{I}} \mid \langle d, d' \rangle \in R^{\mathcal{I}}\} = n\}$
Local reflexivity	$\exists R.Self$	$\{d \in \Delta^{\mathcal{I}} \mid \langle d, d \rangle \in R^{\mathcal{I}}\}$
Nominal	$\{a_1 \dots a_n\}$	$\{a^{\mathcal{I}} \dots a_n^{\mathcal{I}}\}$
Roles:		
Atomic role	R	$R^{\mathcal{I}}$
Inverse role	R^-	$\{\langle a, b \rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \langle b, a \rangle \in R^{\mathcal{I}}\}$
Universal role	U	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Table 4.1: Syntax and semantics of $\mathcal{SROIQ}$ constructors with $a, b \in \Sigma_I$, $A \in \Sigma_C$ is a concept name, C and D are concept expressions, $R \in \Sigma_R$ and $\#S$ is the cardinality of the set S.

	Syntax	**Semantics**
ABox axioms:		
Concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
Negated concept assertion	$\neg C(a)$	$a^{\mathcal{I}} \notin C^{\mathcal{I}}$
Role assertion	$R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$
Negated role assertion	$\neg R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \notin R^{\mathcal{I}}$
Individual equality	$a \approx b$	$a^{\mathcal{I}} = b^{\mathcal{I}}$
Individual inequality	$a \not\approx b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$
TBox axioms:		
Concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
Concept equivalence	$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
RBox axioms:		
Role inclusion	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
Role equivalence	$R \equiv S$	$R^{\mathcal{I}} = S^{\mathcal{I}}$
Complex role inclusion	$R_1 \circ R_2 \sqsubseteq S$	$R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}} \subseteq S$
Role disjointness	$Disjoint(R, S)$	$R^{\mathcal{I}} \cap S^{\mathcal{I}} = \emptyset$

Table 4.2: Syntax and semantics of $\mathcal{SROIQ}$ axioms with $a, b \in \Sigma_I$, C and D are concept expressions and $R \in \Sigma_R$.

checks the existence (yes or not) of a model for $\mathcal{KB}$. Checking knowledge bases consistency is very important as they model information of the real world that cannot be contradictory. Indeed, in most of the cases, contradictory axioms have no benefits and could be very dangerous, for example in applications for managing power plant systems. This reasoning task will be used in our first SII algorithm for checking the existence of a partial model describing a picture. Indeed, we want to discard interpretations that contradict the knowledge base axioms. For example, interpretations where horses have more than four legs.

Axiom Entailment A DL knowledge base $\mathcal{KB}$ entails an axiom ϕ if ϕ is true in every model of $\mathcal{KB}$. The reasoning task here is to check if $\mathcal{KB} \models \phi$ or not. This task allows a user to "logically query" a knowledge base by checking if a new input axiom is true or not.

Concept Satisfiability A concept $C \in \Sigma_C$ is *satisfiable*, with respect to $\mathcal{KB}$, if there exists a model $\mathcal{I}$ of $\mathcal{KB}$ that maps C to a nonempty set: $C^{\mathcal{I}} \neq \emptyset$. Concept satisfiability can be reduced to axiom entailment by checking whether $\mathcal{KB} \models C \sqsubseteq \bot$, and thus it is a decision problem with yes or not as answer.

All these tasks are decidable in DLs and there exist sound and complete decision procedures to compute the reasoning. These procedures are implemented in optimized tools called *reasoners* that are freely available, such as FaCT++ [105], HermiT [94], Pellet [96] and RacerPro [43]. The reasoning paradigm underlying these tools is the tableau method [5] that tries to construct models of a given knowledge base. If this succeeds, the knowledge base is satisfiable, if the construction necessarily fails then there is unsatisfiability.

4.2 Semantic Image Interpretation as a Partial Models Ranking

In this section we provide a formal (logic-based) framework for defining a semantically interpreted picture. The inputs of our framework are a semantically labelled picture $\mathcal{P}$, that can be computed with an object detector (for example Fast R-CNN [36]) and a DL knowledge base $\mathcal{KB}$ with signature Σ. The symbols in Σ are used as labels for the object detector.

We start from the assumption that an image is a partial view of the world. For example, in Figure 2.2 only one leg of the person is visible due to occlusions. Therefore, a formal representation of the content of an image should be a partial view of a model of

$\mathcal{KB}$. This view can be considered as an interpretation of the language of $\mathcal{KB}$, but it does not necessarily satisfy all the axioms of $\mathcal{KB}$. Indeed, the claim that a person has two legs is not satisfied in the picture but it is satisfied in the real world, supposing to be in a normal situation. Thus, if we formalize the world as a model of our knowledge base $\mathcal{KB}$, we formalize the picture with the notion of *partial model*[1] $\mathcal{I}_p$ of $\mathcal{KB}$, see Figure 4.1.

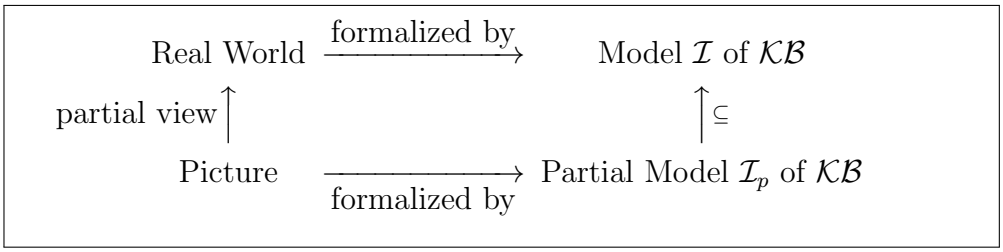


Figure 4.1: The world is formalized with a model of $\mathcal{KB}$ and the partial view of the world contained in the picture is formalized with a partial model of $\mathcal{KB}$.

Definition 2 (Extension of an Interpretation)**.** *Let $\mathcal{I}$ and $\mathcal{I}'$ be two interpretations of the signatures Σ and Σ' respectively, with $\Sigma \subseteq \Sigma'$; $\mathcal{I}'$ is an* extension of $\mathcal{I}$*, or equivalently $\mathcal{I}'$* extends $\mathcal{I}$*, in symbols $\mathcal{I}' \supseteq \mathcal{I}$,if $\Delta^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}'}$, $a^{\mathcal{I}} = a^{\mathcal{I}'}$, $C^{\mathcal{I}} = C^{\mathcal{I}'} \cap \Delta^{\mathcal{I}}$ $R^{\mathcal{I}} = R^{\mathcal{I}'} \cap \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, for all $a \in \Sigma_I$, $C \in \Sigma_C$ and $R \in \Sigma_R$.*

Definition 3 (Partial Model)**.** *Let $\mathcal{KB}$ a knowledge base, $\mathcal{I}_p$ is a* partial model *for $\mathcal{KB}$, in symbols $\mathcal{I}_p \models_p \mathcal{KB}$, if there is a model $\mathcal{I}$ of $\mathcal{KB}$ ($\mathcal{I} \models \mathcal{KB}$) that extends $\mathcal{I}_p$.*

Following the intuition about partial models we define the semantic image interpretation as the computation of a partial model $\mathcal{I}_p = \langle \Delta^{\mathcal{I}_p}, \cdot^{\mathcal{I}_p} \rangle$ of $\mathcal{KB}$. Thus, the construction of a structured representation of the semantic content of an image consists in a method for creating the individuals (the nodes) of $\Delta^{\mathcal{I}_p}$, assigning them a type and linking together (the edges) according to $\cdot^{\mathcal{I}_p}$. Having this graph describing the image content is not enough. We need also the information about the object detection. For example, in an information retrieval system it could be also necessary to return the single bounding boxes. So, we need a link between the elements of our partial model and their corresponding bounding boxes. This consideration leads to the following formal definition of a semantically interpreted picture.

Definition 4 (Semantically Interpreted Picture)**.** *Given a knowledge base $\mathcal{KB}$ with signature Σ and a semantically labelled picture $\mathcal{P} = \langle S, L \rangle$, a* semantically interpreted picture *is a triple $\mathbb{S} = (\mathcal{P}, \mathcal{I}_p, \mathcal{G})$ where:*

[1] Our definition slightly differs from the one of [89].

- $\mathcal{I}_p = \langle \Delta^{\mathcal{I}_p}, \cdot^{\mathcal{I}_p} \rangle$ *is a partial model of* $\mathcal{KB}$*;*
- $\mathcal{G} \subseteq \Delta^{\mathcal{I}_p} \times S$ *is a left-total*[2] *relation called* grounding relation.

The grounding *of every* $d \in \Delta^{\mathcal{I}_p}$*, denoted by* $\mathcal{G}(d)$*, is the set* $\{s \in S \mid \langle d, s \rangle \in \mathcal{G}\}$.

Figure 2.2 shows a semantically labelled picture that describes our running example. The partial model contains a person riding a horse with four legs, a muzzle and a tail, whereas the person has a visible leg, a visible arm and a face. The grounding of the parts and the horse are the corresponding initial bounding boxes, whereas the grounding of the person is the union of the bounding boxes associated to its parts. Note that the above definition can be stated also for the more expressive First-Order Logic. We focus on the $\mathcal{SROIQ}$ DL [48] due to its high expressivity (see above) and its decidability. Indeed, the computation of a partial model $\mathcal{I}_p$ requires the use of a reasoning tool for checking if $\mathcal{I}_p \models_p \mathcal{KB}$ (knowledge base satisfiability), see Section 4.1.3.

The picture content can be described by many partial models. Figure 4.2 shows the original partial model of our running example (above) and three partial models (below). That is, these partial models satisfy the constraints of $\mathcal{KB}$ in our running example (Figure 2.1) but do not encode the real content of the picture, they contain some errors or miss some information. Indeed, partial model A contains almost all the original nodes corresponding to objects but it misses many relations between objects. Partial model B contains correct relations but it misses many nodes and some nodes have wrong labels (the nodes a_1, f_1 and l_5). Finally, partial model C contains all the nodes, many correct relations but with additional errors (there is, for example, a partOf relation between a Leg and an Arm). However, Definition 4 does not provide any criterion to select the partial model that better describes the content of a picture. We need a criterion to decide whether a partial model is a good explanation of the picture content. A criterion for the selection of a partial model could be the comparison with an original partial model of the picture, also known as *ground truth* GT. A method of comparison is a function that returns a score $\mathcal{S}$ (or cost $\mathscr{L}$) of similarity (or dissimilarity) between a partial model and the ground truth. This function enables us to rank the partial models and to return the best one, that is, the one that best matches the image content. Such a function can take into account the number of correct labelled triples $\langle \textit{subject node} - \textit{relation} - \textit{object node} \rangle$ in the partial model graph. For instance, the partial model A contains the triples $\langle \mathsf{person\ p_1} - \mathsf{ride} - \mathsf{horse\ h_1} \rangle$ and $\langle \mathsf{leg\ l_5} - \mathsf{partOf} - \mathsf{person\ p_1} \rangle$. Examples of ranking functions consider the fraction

[2]Every logical individual is associated to at least one bounding box. A bounding box can have no connection with a logical individual, this allows the framework to handle, and possibly discard, false positive bounding boxes of the semantically labelled picture.

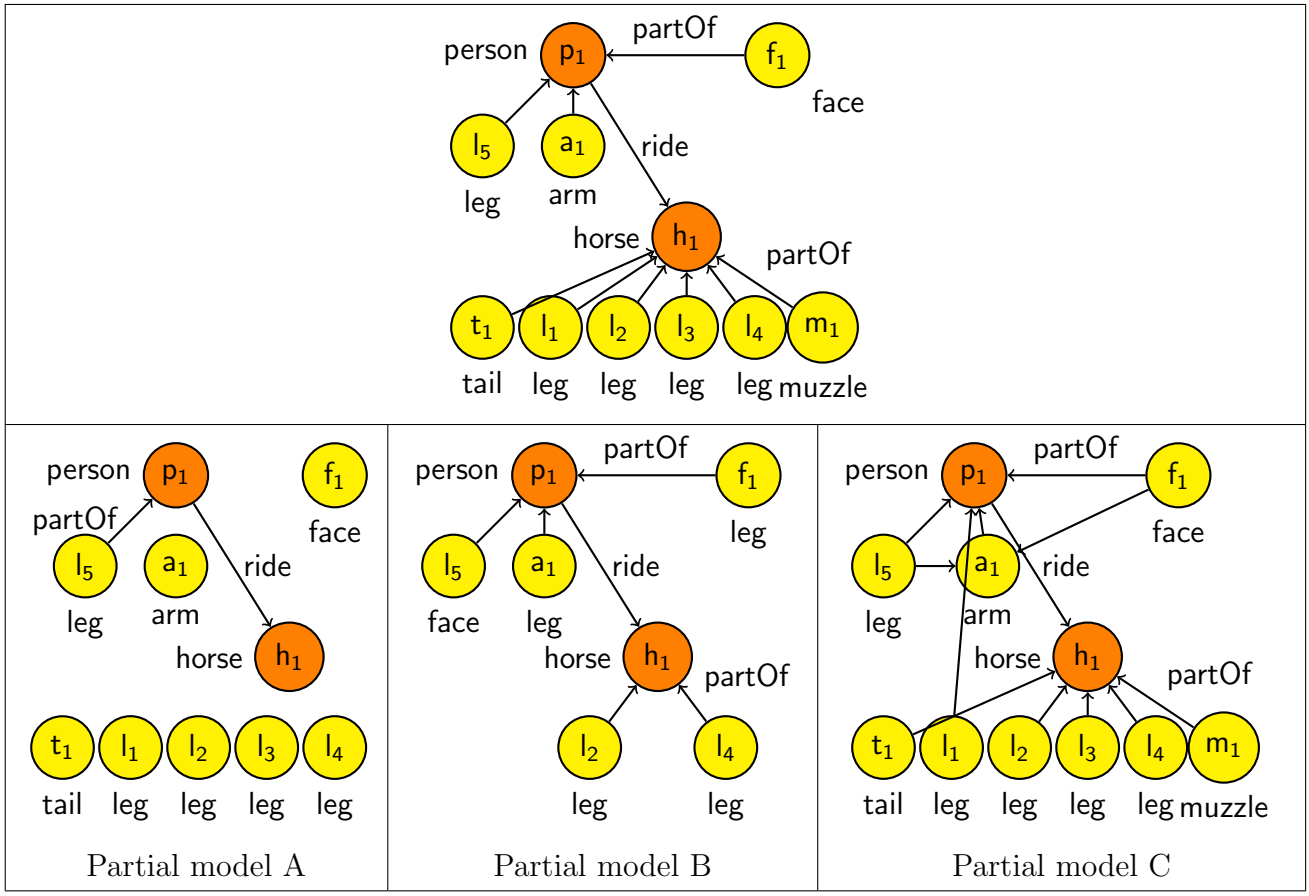


Figure 4.2: The original partial model of Figure 2.2 (above) and three partial models of the same input picture (below). For presentation purposes the edges without labels are implicitly labelled with partOf.

of correct triples in a partial model according to the ground truth GT or the fraction of retrieved triples of the GT. These functions are also known as *precision* and *recall*:

$$\mathcal{S}_{prec} = \frac{|\text{partial model correct triples}|}{|\text{partial model triples}|}, \quad \mathcal{S}_{rec} = \frac{|\text{partial model correct triples}|}{|\text{correct GT triples}|}.$$

These functions consider different aspects of the partial models and thus return different rankings. If we consider $\mathcal{S}_{prec}$, the ranking, with scores, is A (1.0), C (0.75), B (0.5). On the other hand, the ranking with $\mathcal{S}_{rec}$ is C (0.9), B (0.3), A (0.2). The difference in the ranking position of partial model A is due to the fact that this partial model has only few but correct triples. We introduce a loss function $\mathscr{L}_{\mathcal{KB}}$ that measures the "distance" between the partial model and the image content in order to perform a ranking of partial models and choosing the best one. The *most plausible partial model* $\mathcal{I}_p^*$ is the partial

model that minimizes $\mathscr{L}_{\mathcal{KB}}$:

$$(\mathcal{I}_p^*, \mathcal{G}^*) = \underset{\substack{\mathcal{I}_p \models_p \mathcal{KB} \\ \mathcal{G} \subseteq \Delta^{\mathcal{I}_p} \times S}}{\operatorname{argmin}} \mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G}). \tag{4.1}$$

$\mathscr{L}_{\mathcal{KB}}$ measures the (dis)agreement between low-level image features of $\mathcal{P}$ and high-level semantic features contained in $\mathcal{I}_p$, with respect to the low-level/high-level mapping $\mathcal{G}$. The higher $\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G})$ the less the agreement between $\mathcal{I}_p$ and $\mathcal{P}$. For instance, if the element $d \in \Delta^{\mathcal{I}_p}$ of $\mathcal{I}_p$ is grounded to the segment s ($\mathcal{G}(d) = \{s\}$) then $\mathscr{L}_{\mathcal{KB}}$ is lower when $\mathcal{I}_p$ assigns to d the types that correspond to the labels of s with higher weights. Similarly, $\mathscr{L}_{\mathcal{KB}}$ penalizes the partial models that satisfy $R(d, d')$ when the low-level features of $\mathcal{G}(d)$ and $\mathcal{G}(d')$ are in disagreement with the relation R. For example, $\mathscr{L}_{\mathcal{KB}}$ penalizes the models that satisfy $\mathsf{close}(d, d')$ when the relative distance between $\mathcal{G}(d)$ and $\mathcal{G}(d')$ is high. As can be seen from the above examples, the definition of $\mathscr{L}_{\mathcal{KB}}$ heavily depends on the semantics expressed by $\mathcal{KB}$ and on the picture content. Thus, a partial model is a good explanation of the picture content if it minimizes $\mathscr{L}_{\mathcal{KB}}$. It is worth to notice that the number of partial models can be huge according to the complexity of the image, that is, the number of segments of the input labelled picture. Therefore, it is infeasible to explore the whole search space of partial models in order to minimize $\mathscr{L}_{\mathcal{KB}}$. There is the need of algorithms that find the pair $(\mathcal{I}_p^*, \mathcal{G}^*)$ in scalable manner. For example, in the next chapter, the proposed algorithm has polynomial time complexity according to the input image.

Definition 5 (Semantic Image Interpretation Problem)**.** *Given a knowledge base $\mathcal{KB}$, a semantically labelled picture $\mathcal{P}$ and a loss function $\mathscr{L}_{\mathcal{KB}}$, the* semantic image interpretation problem *is finding a partial model $\mathcal{I}_p$ and a grounding $\mathcal{G}$ that minimize $\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G})$.*

However, finding a partial model that minimizes $\mathscr{L}$ cannot be performed by using the ground truth because an original partial model is not always available. Indeed, the task of SII is to predict partial models from new images. Therefore, we need an a priori definition of the loss function that is independent from a ground truth. In the next chapter we define the loss function as a combination of clustering metrics.

Chapter 5

Ranking Partial Models with Clustering

In this chapter, a clustering technique for predicting the best partial model that describes an image content is developed. The idea is to start from already detected objects in the image and cluster them according to some relation. This intuition holds for many relations such as the part-whole relation or the relations that describe events. For example, for the part-whole relation whole objects and their parts need to be grouped together. Regarding the events, the objects participating at the event need to be clustered. Moreover, also the type of the whole object/event has to be discovered. The great advantage of clustering techniques relies in their unsupervision: there is no need of a ground truth of partial models as training set along with a training phase.

The following section provides an overview of clustering analysis as background for the first SII algorithm. Then, the loss function is defined as the clustering balance between the intra and inter-cluster similarity. These similarities are distance measures that consider low-level and semantic features of objects in the image. The next section describes an innovative agglomerative clustering algorithm that searches for a partial model that optimizes the loss function. Then, an experimental evaluation tests the quality of the approach. The algorithm has been evaluated on two datasets on the task of part-whole detection between composite objects and their parts. The results confirm the quality of the method. Finally, the clustering algorithm is discussed along with its advantages and limitations. All the explanations take into account the running example of Figure 2.1.

5.1 An Overview of Cluster Analysis

Cluster analysis [102] (or simply *clustering*) divides input data (called data set) into groups (called *clusters*) that are understandable by humans. This means that the clusters should capture the natural structure of the data, that is, the elements on the same cluster share similar features. Cluster analysis is a useful tool for data analysis, information extraction and as preprocessing technique for other types of analysis, such as data summarization or data compression. Cluster analysis has been applied in a wide variety of fields: psychology (for example, identification of different types of depression), biology (for example, the automatic creation of taxonomies or the understanding of gene functions), business analysis (for example, the segmentation of customers for marketing activities), pattern recognition (used, for example, in understanding the Earth's climate), information retrieval (for example, for grouping the results of a search engine query), machine learning, and data mining.

Clustering is the problem of grouping the elements of the data set into groups (clusters) so that the elements within a group are similar each other (intra-cluster similarity) and different from the elements in the other clusters (inter-cluster similarity). The greater the similarity within a cluster and the greater the difference between clusters, the better, or more separate, the clustering. A collection of clusters is commonly referred as a clustering. The are several types of clusterings, the main distinction is between *partitional* and *hierarchical clustering*. In partitional clustering the clusters are a partition of the data set: every object is assigned to one cluster and the clusters are pairwise disjoint. Hierarchical clustering is a generalisation of partitional clustering where clusters can be recursively clustered in higher-level clusters.

We briefly describe the following two clustering techniques as a background for understanding our initial proposals to the Semantic Image Interpretation problem: the *agglomerative hierarchical clustering* [102] and the *Self-Organizing Maps clustering* [53]. The former technique is suitable for exploring the huge search space of clusterings by repeatedly aggregating clusters and evaluating a loss function on the result. The latter has been chosen as a sort of baseline for comparison due to its ability of automatically estimate the number of clusters, see next sections.

5.1.1 Agglomerative Hierarchical Clustering

The *agglomerative hierarchical clustering* [102] technique starts with N elements as individual clusters and, iteratively, merges the closest pair of clusters. This requires an $N * N$ similarity (or distance) matrix between the data set elements, see Algorithm 1. A hier-

Algorithm 1: Basic Agglomerative Hierarchical Clustering Algorithm.

Input: A data set and a similarity matrix

1 Assign each item to a cluster.
2 **repeat**
3 Find the most similar pair of clusters in the similarity matrix and merge them together.
4 Update the similarity matrix to reflect the proximity between the new cluster and the original clusters.
5 **until** Only one cluster remains.

archical clustering can be graphically displayed as a tree-like diagram called *dendrogram*, see Figure 5.1, which displays both the cluster-subcluster relationships and the order in which the clusters were merged. A crucial operation of Algorithm 1 is the updating of

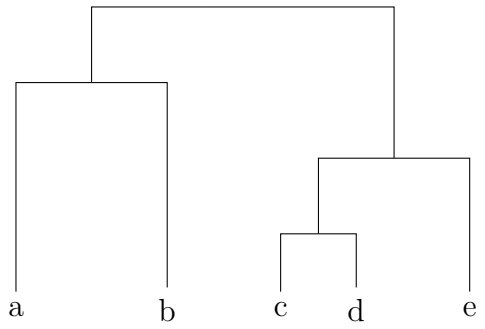


Figure 5.1: A hierarchical clustering of five elements shown as a dendrogram.

the similarity matrix between the new cluster and the original ones. This boils down to define a similarity between clusters, or groups of elements. Examples of common cluster similarities are the MIN, MAX and Group Average functions. The MIN function defines the similarity between two cluster as the similarity between the two closest points in the clusters. The MAX function is the opposite: it considers the similarity between the two farthest points in the clusters. Finally, the group average function defines the cluster similarity as the average pairwise similarity of all pairs of points with the first component in one cluster and the second component in the other cluster.

An important issue of hierarchical clustering regards the number of output clusters (or configuration). For example, the whole dendogram could be of little interest and a cut of the tree at a certain height could be more significant. If we consider the clustering of Figure 5.1 we can have the following clusterings (with a different number of clusters) $\{\{a, b\}, \{\{c, d\}, e\}\}$ and $\{\{a\}, \{b\}, \{\{c, d\}, e\}\}$ according to the particular tree height that we cut the dendogram. This problem could be addressed by considering internal properties

of the clusters such as the intra-cluster error sum Λ and the inter-cluster error sum Γ. We briefly describe the method proposed in [52] as a background of our clustering algorithm for SII described in Section 5.2.

Let $\mathcal{D} = \{\mathbf{e}_1\}_{i=1}^{N}$ a dataset of N elements where each element is a feature vector in a m dimensional space: $\mathbf{e}_i = \langle e_{i1} \dots e_{im} \rangle$. A cluster C_j is simply a set of n_j elements: $C_j = \{\mathbf{e}_1^{(j)} \dots \mathbf{e}_{n_j}^{(j)}\}$. The *centroid* $\mathbf{e}_0^{(j)}$ of a cluster C_j is the mean vector of the cluster elements:

$$\mathbf{e}_0^j = \frac{1}{n_j} \sum_{i=1}^{n_j} \mathbf{e}_i^{(j)}.$$

Recalling the above-mentioned definition, a clustering groups elements into clusters such that elements in the same cluster have maximum (intra-cluster) similarity and elements in different clusters have minimum (inter-cluster) similarity. According to [52], the intra-cluster similarity can be defined as the intra-cluster error sum Λ, that is the squared error of the Euclidean distance between the elements and the centroid of a cluster:

$$\Lambda = \sum_{j=1}^{k} \sum_{i=1}^{n_j} ||\mathbf{e}_i^{(j)} - \mathbf{e}_0^{(j)}||_2^2 \tag{5.1}$$

where k is the total number of clusters. On the other hand, the inter-cluster similarity can be defined as the inter-cluster error sum Γ, that is the squared error of the Euclidean distance between the centroids $\{\mathbf{e}_0^{(j)}\}_{j=1}^{k}$ of the k clusters and the global centroid $\mathbf{e}_0$ of the dataset:

$$\Gamma = \sum_{j=1}^{k} ||\mathbf{e}_i^{(j)} - \mathbf{e}_0||_2^2 \tag{5.2}$$

where

$$\mathbf{e}_0 = \frac{1}{N} \sum_{i=1}^{N} \mathbf{e}_1.$$

Given a clustering χ, the *clustering balance* is a weighted sum of Λ and Γ:

$$\mathcal{E}(\chi) = \alpha\Lambda + (1 - \alpha)\Gamma. \tag{5.3}$$

According to [52], Λ is always nondecreasing and Γ is always nonincreasing and the optimal clustering configuration is the one that minimizes the clustering balance $\mathcal{E}$. A clustering configuration is also useful as a proposal input for partitional clustering algorithms (such as K-means). In Section 5.2, we develop an agglomerative clustering algorithm that minimizes the clustering balance and Λ, Γ encode both geometric and semantic information of the bounding boxes of an input semantically labelled picture.

5.1.2 Self-Organizing Map Clustering

Self-Organizing Maps (SOMs) [53] are a sort of artificial neural networks trained with unsupervised data to produce a low-dimensional (usually two-dimensional) and discrete representation (also called feature map) of the input data. They are a useful method for visualizing high-dimensional data in a two dimensional space.

A SOM consists of a (usually) two-dimensional hexagonal or rectangular grid (or map) M. Each node v of the grid is associated with a vector of weights $\mathbf{w}_v$ of the same dimension of the input data $\mathbf{e} \in \mathcal{D} = \{\mathbf{e}_i\}_{i=1}^N$. The underlying idea of SOM is to adapt the grid to the input data, that is, parts of the map respond similarly to certain input patterns. When an input data $\mathbf{e}$ is fed to the map the SOM algorithm finds the node with the closest (smallest Euclidean distance) weight vector to $\mathbf{e}$ (competitive learning). This node is called best matching unit (BMU). The weights of the BMU and of its neighbours in the SOM are adjusted towards the input vector. The magnitude of the change decreases with the time and with the distance (within the lattice) from the BMU. This process is repeated for a big number of cycles. Algorithm 2 describes in details the learning process. Here $\alpha(t)$ is a

Algorithm 2: Self-Organizing Map Algorithm.

Input: A data set $\mathcal{D}$
Output: A feature map M

```
Let M be a 2-dimensional grid of nodes.
Randomize the node weight vectors in M.
while t < max_number_iterations do
    Let e ∈ D an input vector.
    bmu ← argmin_{v∈M} ||e − w_v||_2^2        // bmu stands for best matching unit
    foreach neighbour node v of bmu do
        w_v = w_v + θ(bmu, v, t) · α(t) · (e − w_v)        // also the bmu is updated
return M
```

monotonically decreasing learning coefficient. $\theta(bmu, v, t)$ is the neighbourhood function which gives the distance between the BMU and the node v at step t. Common choices for θ are the step or the Gaussian function. The function θ decreases monotonically according to the learning step. At the beginning, the neighbourhood is broad so the weights updating takes place on the global scale. At the end, the neighbourhoods are just few nodes so the weight updating is performed locally.

The returned feature map M has to be interpreted to return a clustering. During the training step the weight vectors of the neighbours of the BMU are moved in the same direction, thus similar input patterns excite adjacent nodes. Therefore, similar patterns

are mapped together and and dissimilar ones apart. This makes a SOM a semantic map that can be visualized with a unified distance matrix (U-matrix) [23]. A U-matrix reports the mean of the Euclidean distances between a single node and its neighbours in grayscale values. Thus, light colors depict closely spaced weight vectors of the nodes and darker colors indicate separated node weight vectors. This representation can be processed with simple image processing techniques to easily identify the clusters.

We use this clustering technique in the very first experiments of our SII models [26, 25], and in Section 5.4 we compare this technique with an algorithm based on agglomerative clustering, see Section 5.3.

5.2 The Loss Function as Clustering Optimization

In the following, we concentrate on the part-whole relation and we define a loss function $\mathscr{L}_{\mathcal{KB}}$ that is used to recognise the presence of complex objects starting from their parts. As input, we consider a semantically labelled picture and a DL knowledge base whose symbols are used as labels in the input picture. The choice of this particular task is firstly motivated by the fact that, given the novelty of the approach, we want to test it on a simplified, though real, scenario. Second, the part-whole relation has a special status with respect to other relations, as it can be used to represent, via reification, a large class of relations [40]. In our running example, the relationship between the person and the horse can be reified in an object of type "riding event" whose parts are the person and the horse. Third, the advantage of exploiting parts for detecting a whole object is that parts have a low intra-class variability, their configuration provides useful information about the whole object and they better deal with deformations and pose variations [70]. Fourth, there is a well-established dataset, the PASCAL-Part-dataset [17]. The problem of recognising complex objects once simpler objects (parts) have been detected can be seen as a clustering problem and we specify the loss function in terms of a clustering optimisation function.

Recognising the presence of complex objects and their parts, starting from a known set of atomic objects, can be seen as a hierarchical clustering problem with the additional task of typing the intermediate nodes. More precisely: the *clustering solution associated to a semantically interpreted picture* $\mathbb{S} = (\mathcal{P}, \mathcal{I}_p, \mathcal{G})$ is equal to $\mathcal{C} = \{C_d \mid d \in \Delta^{\mathcal{I}_p}\}$ where each $C_d = \{s \in \mathcal{G}(d') \mid d' \in \Delta^{\mathcal{I}_p}, \langle d, d' \rangle \in \mathsf{hasPart}^{\mathcal{I}_p}\}$. If we assume that the $\mathsf{hasPart}$ relation is inverse functional, transitive and irreflexive[1], the clustering $\mathcal{C}$ is guaranteed to be hierarchical. However, other relations do not respect these assumptions and other

[1]The standard ontological axioms of classical mereology assume the reflexivity of the part-whole relation. Here we use a restricted version of this relation by considering that nothing can be part of itself.

type of clustering techniques need to be used. For example, if a relation is not functional then its members can belong to more than one cluster and overlapping clustering [4] has to be used. As stated in Section 5.1, clustering algorithms are based on a distance measure between input elements. We propose a distance measure $\delta(d, d')$ that combines the Euclidean distance $\delta_{\mathcal{G}}(d, d')$ between the centroids of $\mathcal{G}(d)$ and $\mathcal{G}(d')$ called *grounding distance*, and a *semantic quasidistance*[2] $\delta_{\mathcal{KB}}(d, d')$ between the types of d and d' in $\mathcal{I}_p$. As grounding distance we use the L_2^2 norm on the centroids of the segments[3] $\delta_{\mathcal{G}}(d, d') = ||cent(\mathcal{G}(d)) - cent(\mathcal{G}(d'))||_2^2$ (the centroids are scaled to the interval $[0, 1]$). The use of the centroid of $\mathcal{G}(d)$ as a numeric feature is enough to show the effectiveness of our approach. Indeed, the combination of the centroid with the L_2^2 norm tends to group objects close in the space. This assumption is based on the Law of Proximity of Gestalt Psychology [97] that states that parts of the same object are usually close. However, other relations are more difficult to recognize, for example the proximity between a person and a horse is not sufficient to distinguish between the ride and the next-to relation. Moreover, other relations are difficult to recognize using the proximity as the subject and the object could be distant in the space, for example the look-at, above, flying-kite and jumping-on relations. In these cases, the approach can be generalised by considering other features like shape, texture, color, inverse distance, angle with a given axis, etc. For the semantic quasidistance we specialize the Hirst and St-Onge measure (HSO) defined in [47]. Here concepts have a big distance if (1) the knowledge base path (number of arcs) between them is high and (2) this path has a large number of changes of directions. If a path is composed only by upward (or downward) ISA arcs then it has no changes of directions. Whereas, if a path is composed by composing upward, downward ISA arcs with a partOf arc then the path has changes of directions. With this idea in mind our semantic quasidistance $\delta_{\mathcal{KB}}(d, d')$ assigns a small value to concepts constrained with the hasPart relation. Whereas $\delta_{\mathcal{KB}}(d, d')$ assigns a larger value to pairs of concepts with no hasPart constraint between them or with a negative hasPart constraint. We define $\delta_{\mathcal{KB}}(d, d')$ as

$$\delta_{\mathcal{KB}}(d, d') = \min \left\{ par_{\mathcal{KB}}(C, C') \;\middle|\; \begin{array}{l} \langle C, w \rangle \in L(\mathcal{G}(d)), \\ \langle C', w' \rangle \in L(\mathcal{G}(d')), \\ \text{for no } D, D' \in \Sigma_C, \\ D \sqsubseteq C, D' \sqsubseteq C' \end{array} \right\} \tag{5.4}$$

[2]Semantic distance is not required to be symmetric.

[3]Note that, the method is general and can be applied to semantically labelled picture obtained by both semantic segmentation and object detection. For this reason we use the general term segments.

with

$$par_{\mathcal{KB}}(C, C') = \begin{cases} 1 & \text{if } \mathcal{KB} \models C \sqsubseteq \exists \mathsf{hasPart}.C' \\ \infty & \text{if } \mathcal{KB} \models C \sqsubseteq \neg\exists \mathsf{hasPart}.C' \\ \gamma & \text{otherwise} \end{cases}$$

where the parameter γ, according to the definition of HSO, can be defined as: $\gamma = pathLength + k * changesDirections$, with k a constant. Examples of semantic quasidistance between the elements of the partial model of Figure 2.2 are the following:

$$\begin{aligned} \delta_{\mathcal{KB}}(\mathsf{h}_1, \mathsf{t}_1) &= par_{\mathcal{KB}}(\mathsf{Horse}, \mathsf{Tail}) = 1 \\ \delta_{\mathcal{KB}}(\mathsf{h}_1, \mathsf{f}_1) &= par_{\mathcal{KB}}(\mathsf{Horse}, \mathsf{Face}) = \infty \\ \delta_{\mathcal{KB}}(\mathsf{h}_1, \mathsf{p}_1) &= par_{\mathcal{KB}}(\mathsf{Horse}, \mathsf{Person}) = \gamma. \end{aligned}$$

Following Section 5.1.1 and [52], we define the intra-cluster error sum Λ as:

$$\Lambda = \sum_{\langle p,d\rangle \in \mathsf{hasPart}^{\mathcal{I}_p}} \left(\beta \delta_{\mathcal{G}}(p, d) + (1 - \beta) \frac{\delta_{\mathcal{KB}}(p, d)}{w(d, \mathcal{G}, \mathcal{I}_p)} \right) \tag{5.5}$$

with $w(d, \mathcal{G}, \mathcal{I}_p) = w_l$, if l is the most specific concept that $\mathcal{I}_p$ assigns to d and w_l is the weight associated to the label l in $\mathcal{G}(d)$. Otherwise $w(d, \mathcal{G}, \mathcal{I}_p)$ is undefined. The parameter $\beta \in [0, 1]$ mixes the geometric and the semantic contribution. The inter-cluster error sum Γ is defined as:

$$\Gamma = \sum_{\langle p,p'\rangle \in (\exists \mathsf{hasPart}.\top)^{\mathcal{I}_p}} \delta_{\mathcal{G}}(p, p'). \tag{5.6}$$

Finally, we define the loss function $\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G})$ as the clustering balance (with parameter $\alpha \in [0, 1]$) between Λ and Γ:

$$\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G}) = \alpha \cdot \Lambda + (1 - \alpha) \cdot \Gamma. \tag{5.7}$$

5.3 A Clustering and DL Reasoning Algorithm for Minimizing the Loss Function

Minimizing Equation (5.7) analytically is not possible as $\mathscr{L}_{\mathcal{KB}}$ is not expressed in an analytical form. We therefore developed the PartWholeClusteringAlgorithm (in short PWCA, Algorithm 3) that preforms a greedy search on the lattice of clusterings of the set of segments $\{s_1, \ldots, s_n\}$ of $\mathcal{P}$. For each clustering $\mathcal{C}$, PWCA generates a partial model-grounding pair $(\mathcal{I}_p, \mathcal{G})$ that minimizes $\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}_p, \mathcal{G})$. PWCA starts from the initial

clustering $\mathcal{C}_0 = \{\{s_1\}, \ldots, \{s_n\}\}$. At each iteration PWCA (i) considers a clustering $\mathcal{C}$; (ii) generates all super-clusterings of $\mathcal{C}$ by merging any pair of clusters in $\mathcal{C}$ (agglomerative step performed by AGGLOMERATIVEMATRIX($\mathcal{C}$)) and (iii) selects the super-clustering $\mathcal{C}'$ whose associated partial model-grounding pair (computed by BESTMOD) minimizes the loss function. PWCA terminates when $|\mathcal{C}| = 1$, that is, no super-clustering can be generated form $\mathcal{C}$, or when there is no partial model that can be associated to the super-clusterings of $\mathcal{C}$. PWCA returns the partial model-grounding pair with minimal loss among those that have been generated.

Algorithm 3: PARTWHOLECLUSTERINGALGORITHM($\mathcal{P}, \mathcal{KB}$)

Input: A labelled picture $\mathcal{P} = \langle S, L\rangle$ and a knowledge base $\mathcal{KB}$
Output: A partial model $\mathcal{I}_p$ and its grounding $\mathcal{G}$ on $\mathcal{P}$

$\mathcal{C}_0 \leftarrow \{\{s_1\} \ldots \{s_{|S|}\}\}$ from $\mathcal{P}$
$(\mathcal{I}_p^*, \mathcal{G}^*) \leftarrow$ BESTMOD($\mathcal{P}, \mathcal{C}_0, \mathcal{KB}$)
$F \leftarrow \{\mathcal{C}_0\}$
while $F \neq \emptyset$ **do**
  $\mathcal{C} \leftarrow$ REMOVEELEMENT(F)
  $A \leftarrow$ AGGLOMERATIVEMATRIX($\mathcal{C}$)
  if $A \neq \emptyset$ **then**
    $\mathcal{C}' \leftarrow \text{argmin}_{\mathcal{C} \in A} \mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \text{BESTMOD}(\mathcal{P}, \mathcal{C}, \mathcal{KB}))$
    $F \leftarrow \{C'\}$
    $(\mathcal{I}'_p, \mathcal{G}') \leftarrow$ BESTMOD($\mathcal{P}, \mathcal{C}', \mathcal{KB}$)
    if $\mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}'_p, \mathcal{G}') < \mathscr{L}_{\mathcal{KB}}(\mathcal{P}, \mathcal{I}^*_p, \mathcal{G}^*)$ **then**
      $(\mathcal{I}^*, \mathcal{G}^*) \leftarrow (\mathcal{I}', \mathcal{G}')$
return $(\mathcal{I}_p^*, \mathcal{G}^*)$

BESTMOD generates a partial model-grounding pair $(\mathcal{I}_p, \mathcal{G})$ from a clustering $\mathcal{C} = \{C_1 \ldots C_n\}$ as follows: The domain of $\mathcal{I}_p$ is $\Delta^{\mathcal{I}_p} = \{d_s \mid s \in C_j \text{ and } C_j \in \mathcal{C}\} \cup \{p_j \mid C_j \in \mathcal{C}\}$; it contains the element d_s for each segment s of $\mathcal{P}$, and an additional element p_j that corresponds to the parent (the composite object) of the cluster C_j. The grounding $\mathcal{G}(d_s)$ is $\{s\}$, whereas the grounding of p_j is given by $\mathcal{G}(p_j) = \{\bigcup_{s \in C_j} s\}$. At this step, BESTMOD selects one concept for every $d \in \Delta^{\mathcal{I}_p}$ according to the weighted multiple labels associated to the segments of $\mathcal{P}$. The function ca_j, called the *concept assigning function* for $\mathcal{C}_i$, is defined as:

$$ca_j : C_j \cup \mathcal{G}(p_j) \rightarrow \Sigma_C$$

such that for all $s \in C_j$, $\langle ca_j(s), w\rangle \in L(s)$. Let $CA(C_j)$ be the set of all concept assignment functions for C_j. The best concept assignment function for cluster $C_j \in \mathcal{C}$ is

the one such that:

$$ca_j^* = \underset{ca_j \in CA(C_j)}{\operatorname{argmin}} \sum_{s \in C_j} \frac{par_{\mathcal{KB}}(ca_j(p_j), ca_j(s))}{wl(s, ca_j(s))} \tag{5.8}$$

where the function $wl(s,l) = w$, if $\exists l : \langle l, w\rangle \in L(s)$. Otherwise $wl(s,l)$ is undefined. For every $C_j \in \mathcal{C}$ the algorithm computes Equation (5.8) obtaining the best concept assignment ca_j^* for C_j. Then PWCA constructs the interpretation function for every element of $l \in \Sigma_C$:

$$l^{\mathcal{I}_p} = \{d \in \Delta^{\mathcal{I}_p} \mid ca_j^*(d) = l \text{ for some } C_j \in \mathcal{C}\}$$

and $\mathsf{hasPart}^{\mathcal{I}_p} = \{\langle p_j, d\rangle : \mathcal{G}(d) \subseteq C_j\}$. In this manner, PWCA handles possible detection errors where a segment is classified according to the label with the highest weight. The step of introducing a new logical individual, with type given by Equation (5.8), that "explains" a cluster according to $\mathcal{KB}$ can be seen as an abduction operation. As a final step BestMod checks if $\mathcal{I}_p$ is a partial model of $\mathcal{KB}$. This is done by checking the consistency of $\mathcal{KB}$ extended with the ABox $\mathcal{A}$ that represents $\mathcal{I}_p$[4]. Thus, we check if $\mathcal{I}_p \models_p \mathcal{KB} \cup \mathcal{A}$ using a standard DL reasoner, see Section 4.1.3.

AgglomerativeMatrix generates all possible clusterings that can be obtained by joining any pair of clusters in $\mathcal{C} = \{C_1, \ldots, C_n\}$. The result is represented in a matrix A whose elements are $a_{kh} = (\mathcal{C} \setminus \{C_k, C_h\}) \cup \{C_k \cup C_h\}$. For every clustering in A the BestMod procedure generates the corresponding partial model and grounding. The algorithm performs a greedy choice, it selects the clustering whose partial model and grounding minimize the loss function, see the combination of argmin with BestMod, line number 8. During these operations of clusterings generation and local minimum selection the partial model $\mathcal{I}_p^*$ and grounding $\mathcal{G}^*$ that minimize the loss are stored (lines 10, 11, 12) to be returned at the end, line 13. The algorithm terminates if there are no clusterings to process, that is, PWCA reaches the bottom of the lattice or all the clusterings generate no partial models.

Regarding the computational complexity of PWCA, the algorithm starts from $\mathcal{C}_0$, with $|S|$ clusters, and passes through $O(|S|)$ levels of the lattice to reach the bottom. A level contains clusterings with n clusters. Therefore, for every level the algorithm performs $O(n^2)$ operations due to the super-clusterings generation and the greedy choice. Thus, the whole algorithm visits $O(|S|^3)$ nodes of the lattice.

In our running example, the algorithm starts with the clustering (for simplicity we use individuals instead of bounding boxes) $\mathcal{C}_0 = \{\{\mathsf{l}_1\}, \{\mathsf{l}_2\}, \{\mathsf{l}_3\}, \{\mathsf{l}_4\}, \{\mathsf{l}_5\}, \{\mathsf{a}_1\}, \{\mathsf{m}_1\}, \{\mathsf{t}_1\}, \{\mathsf{f}_1\}\}$.

[4]The interpretation $\mathcal{I}_p$ is represented with an ABox as follows: if $d \in l^{\mathcal{I}_p}$, with $l \in \Sigma_C$, then $l(\mathsf{d})$ is in the ABox, if $(p,d) \in \mathsf{hasPart}^{\mathcal{I}_p}$, with $\mathsf{hasPart} \in \Sigma_R$, then $\mathsf{hasPart}(\mathsf{p},\mathsf{d})$ is in the ABox.

For presentation purposes, we assume that only the parts in the running example are detected and whole objects (the person and the horse) are missing. BESTMOD$(\mathcal{P}, \mathcal{C}_0, \mathcal{KB})$ computes the partial model $\mathcal{I}_p^*$ and the grounding $\mathcal{G}^*$ corresponding to $\mathcal{C}_0$ as follows:

- $\Delta^{\mathcal{I}_p}$ is defined as a set of 9 individuals corresponding to simple objects, $\mathsf{l}_1, \ldots, \mathsf{f}_9$, one for every cluster in $\mathcal{C}_0$;
- $\Delta^{\mathcal{I}_p}$ is extended with 9 new individuals corresponding to composite objects, $\mathsf{p}_1, \ldots, \mathsf{p}_9$, one for every cluster in $\mathcal{C}_0$;
- then the part-whole assertions are added (for example, $\mathsf{hasPart}(\mathsf{p}_1, \mathsf{l}_1)$, $\mathsf{hasPart}(\mathsf{p}_7, \mathsf{m}_1)$, and $\mathsf{hasPart}(\mathsf{p}_9, \mathsf{f}_1)$);
- then BESTMOD assigns the best concept types to all individuals in $\Delta^{\mathcal{I}_p}$ according to Equation (5.8) (for example, $\mathsf{Leg}(\mathsf{l}_1)$, $\mathsf{Arm}(\mathsf{a}_1)$,..., $\mathsf{Muzzle}(\mathsf{m}_1)$, $\mathsf{Horse}(\mathsf{p}_1) \sqcup \mathsf{Person}(\mathsf{p}_1)$, $\mathsf{Horse}(\mathsf{p}_7)$, and $\mathsf{Person}(\mathsf{p}_9)$);
- Finally BESTMOD checks the consistency of $\mathcal{I}_p^*$ (represented with an ABox) with a DL reasoner.

Then PWCA enters in the while loop and the best child clustering of $\mathcal{C}_0$, selected by the argmin, is $\mathcal{C}' = \{\{\mathsf{l}_1\},\{\mathsf{l}_2\},\{\mathsf{l}_3\},\{\mathsf{l}_5\},\{\mathsf{a}_1\},\{\mathsf{m}_1\},\{\mathsf{t}_1\},\{\mathsf{f}_1, \mathsf{l}_4\}\}$. The BESTMOD procedure generates a new partial model (in a new ABox) from $\mathcal{C}'$. According to the steps above, every element in every cluster is first introduced as a logical individual in the ABox. Then, there is the introduction of individuals corresponding to the parents, the introduction of the part-whole assertions and the assignment of the concept types (Equation (5.8)). For example, the cluster $\{\mathsf{f}_1, \mathsf{l}_4\}$ has parent p'_1 of type Person. At the end of the while loop the algorithm outputs the partial model corresponding to the clustering $\{\{\mathsf{a}_1, \mathsf{l}_1\}, \{\mathsf{f}_1, \mathsf{l}_4, \mathsf{l}_5\}, \{\mathsf{m}_1, \mathsf{t}_1, \mathsf{l}_2, \mathsf{l}_3\}\}$. In this partial model the first cluster is represented with a logical individual of type Person with two $\mathsf{hasPart}$ relations with individuals of type Arm and Tail. The second cluster is still represented with a person with the $\mathsf{hasPart}$ relation with three individual of type Face, Leg and Leg. The last cluster is represented with an individual of type Horse having as parts logical individuals of type Muzzle, Tail, Leg and Leg. Notice that this clustering partially matches with the ground truth of Figure 2.2. Indeed the types of the parents are computed correctly, whereas a horse leg is assigned to a person.

5.4 Experimental Evaluation: Object Classification from Parts

To evaluate the quality of PWCA we focus on the classification of complex objects from their parts. We evaluate two aspects of this task: (i) the ability of PWCA to group together parts belonging to the same complex object and (ii) given the clustered parts, the ability of PWCA to assign the correct label to the composite object. In the following we present the input datasets of semantically labelled pictures and the knowledge base, the evaluation criteria and the results.

5.4.1 The PASCAL-Part and PartOf Datasets

To evaluate our approach we need a ground truth dataset of images annotated with bounding boxes containing an object. Each bounding box is correctly annotated with a label describing its object type: a part or a whole object. Moreover, pairs of bounding boxes are annotated with the part-of relation. One such a dataset is the PASCAL-Part-dataset [17] containing 10103 annotated images. Labels are divided into three main groups: animals, vehicles and indoor objects, with their corresponding parts and "part-of" label. Whole objects inside the same group can share parts. Whole objects of different groups do not share any part. However, the labels for parts are very specific, for example, "left lower leg" and "right hand". Since we are not interested in such a fine-grained distinction, we merged, without loss of generality, the bounding boxes of the images that refer to the same part in a unique bounding box. For example, two bounding boxes labelled with "left lower leg" and "left front leg" of the same leg have been merged in a bounding box labelled with "leg". In this way, we limit our experiments to a dataset with 20 labels for whole objects and 39 labels for parts. In addition, we remove from the dataset any bounding box with height or width smaller than 6 pixels. Then we split the dataset into PASCAL-Part training set (7687 images) and PASCAL-Part test set (2416 images) such that they contain the 80% and 20% of the objects of every class respectively. We perform the evaluation on the test set.

The images of the PASCAL-Part test set contain few whole objects (an average of 2.10 whole objects per image) thus in many cases there is few variability between classes of whole objects. Since we want to test our algorithm on images that contain more composite objects and parts, we develop a new dataset called PartOf dataset. Using LabelMe [86] we annotate 203 pictures with simple objects, whole objects and their part-of relations. The labels of the PartOf dataset include classes of buildings,

trees, people, street vehicles, and their parts. Table 5.1 summarises the main differences and figures of these datasets. The PartOf dataset contains more complex objects and

	PASCAL-Part test set	PartOf dataset
Pictures	2416	203
Average of simple objects per picture	12.89	13.28
Average of complex objects per picture	2.10	9.84
Average of parts per complex object	5.69	2.04

Table 5.1: The main figure of the PASCAL-Part test set and the PartOf dataset.

parts per picture than PASCAL-Part. The presence of more complex objects in our dataset is more challenging for PWCA than the PASCAL-Part test set. The datasets of the experiments and the knowledge bases (see next section) are available at `https://dkm.fbk.eu/technologies/knowpic`.

5.4.2 The PASCAL-Part and PartOf Knowledge Bases

We also manually build two knowledge bases about the meronymy of the classes of objects in the datasets. Such knowledge bases are also called meronomies. That is, hierarchies that deal with the part-whole relation instead of the classical subsumption between classes. In the specific, a meronomy states the parts for a set of classes through existential (and possibly cardinality) constraints. Figure 5.2 shows extracts of the knowledge bases for the PASCAL-Part and PartOf datasets, respectively. In a large scale setting a meronomy can be automatically extracted from Semantic Web resources like WordNet [34] or Yago [66]. However, the cardinality constraints should be manually added or extracted from semantic networks such as ConceptNet [100]. This procedure requires human supervision as ConceptNet is not complete and is affected by noise: for example, some mappings to WordNet and many cardinality constraints are missing or misspelled. For this reason, in the experiments we also investigate the impact of cardinality constraints and check if they are really necessary. Table 5.3 summarises the main quantitative differences of these knowledge bases (note that for PASCAL-Part $\mathcal{KB}$ there are two additional classes with respect to the dataset: Animal and Vehicle). The PartOf knowledge base contains less classes for whole objects but more parts than the PASCAL-Part knowledge base. Furthermore, PartOf $\mathcal{KB}$ contains many more part-whole axioms than the PASCAL-Part one. This means that every whole object is related with more parts. Moreover, there are many classes that are neither whole objects nor parts. This makes possible to test if

PASCAL-Part $\mathcal{KB}$			**PartOf $\mathcal{KB}$**		
			Person	$\sqsubseteq$	$(=1)$hasParts.Mouth
Dog	$\sqsubseteq$	$(=1)$hasParts.Muzzle			$\sqcap$ $(=2)$hasParts.Leg
		$\sqcap$ $(=1)$hasParts.Nose			$\sqcap(=1)$hasParts.Nose
		$\sqcap(=1)$hasParts.Tail			$\sqcap$ $(=1)$hasParts.Head
		$\sqcap$ $(=2)$hasParts.Ear			$\sqcap(=1)$hasParts.Torso
		$\sqcap(=1)$hasParts.Torso			$\sqcap$ $(=2)$hasParts.Arm
		$\sqcap$ $(=2)$hasParts.Eye			$\sqcap(=2)$hasParts.Eye
		$\sqcap(=1)$hasParts.Neck			$\sqcap$ $\exists$hasParts.Hair
		$\sqcap$ $(=1)$hasParts.Head	Tree	$\sqsubseteq$	$(=1)$hasParts.Trunk
		$\sqcap$ $(=4)$hasParts.Leg			$\sqcap$ $\exists$hasParts.Foliage
PottedPlant	$\sqsubseteq$	$(=1)$hasParts.Pot	Motorbike	$\sqsubseteq$	$(=1)$hasPart.Bodywork
		$\sqcap$ $(=1)$hasParts.Plant			$\sqcap$ $(=1)$hasParts.LicensePlate
Car	$\sqsubseteq$	$(=1)$hasPart.Bodywork			$\sqcap$ $(=2)$hasParts.Mirror
		$\sqcap$ $(=1)$hasParts.LicensePlate			$\sqcap$ $(=2)$hasParts.Wheel
		$\sqcap$ $(=2)$hasParts.Mirror			$\sqcap$ $(=2)$hasParts.Tyre
		$\sqcap$ $(=4)$hasParts.Wheel			$\sqcap$ $(=1)$hasParts.Handlebar
		$\sqcap$ $\exists$hasParts.Door			$\sqcap$ $\exists$hasParts.Taillight
		$\sqcap$ $\exists$hasParts.Headlight			$\sqcap$ $\exists$hasParts.Headlight
		$\sqcap$ $\exists$hasParts.Window			$\sqcap$ $\exists$hasParts.Saddle

Table 5.2: An excerpt of the PASCAL-Part and the PartOf knowledge bases.

	PASCAL-Part $\mathcal{KB}$	**PartOf $\mathcal{KB}$**
Axioms in $\mathcal{KB}$	85	229
Classes	61	127
Classes for complex objects	22	10
Classes for parts	39	48
Classes for other objects	0	69

Table 5.3: The main figure of the PASCAL-Part and the PartOf knowledge bases.

PWCA is able to deal with objects not related with constraints.

5.4.3 Evaluation Criteria and Results

To compute the performance of PWCA with respect to a gold standard dataset $\mathcal{D}$, we need to define a distance measure between the output of PWCA and the annotations on the elements of $\mathcal{D}$. We suppose that both, the output of PWCA, and the annotations on $\mathcal{D}$, are represented with ABoxes. We, therefore, need to define a distance between ABoxes. Let $\mathcal{A}_{\mathcal{P}}^{PWCA}$ be the ABox that represents the output of PWCA on $\mathcal{P}$ and let $\mathcal{A}_{\mathcal{P}}^{\mathcal{D}}$ the ABox associated to the picture $\mathcal{P}$ in the dataset $\mathcal{D}$. We define the following two measures.

Grouping (GRP) measures how good is PWCA at grouping parts of the same composite object.

$$prec_{\text{GRP}} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{P} \in \mathcal{D}} \frac{|\mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{PWCA}) \cap \mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}{|\mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{PWCA})|}$$

$$rec_{\text{GRP}} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{P} \in \mathcal{D}} \frac{|\mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{PWCA}) \cap \mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}{|\mathsf{sibl}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}$$

where for any ABox $\mathcal{A}$, $\mathsf{sibl}(\mathcal{A}) = \{\langle d, d' \rangle \mid \exists d'' : \{\mathsf{hasPart}(d'', d), \mathsf{hasPart}(d'', d')\} \subseteq \mathcal{A}\}$.

Complex-object prediction (COP) measures how good is PWCA at predicting the type of a composite object.

$$prec_{\text{COP}} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{P} \in \mathcal{D}} \frac{|\mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{PWCA}) \cap \mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}{|\mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{PWCA})|}$$

$$rec_{\text{COP}} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{P} \in \mathcal{D}} \frac{|\mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{PWCA}) \cap \mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}{|\mathsf{ptype}(\mathcal{A}_{\mathcal{P}}^{\mathcal{D}})|}$$

where for any ABox $\mathcal{A}$, $\mathsf{ptype}(\mathcal{A}) = \{\langle d, \mathsf{C} \rangle \mid \exists d' : \{\mathsf{hasPart}(d', d), \mathsf{C}(d')\} \subseteq \mathcal{A}\}$. Intuitively, $\mathsf{ptype}(\mathcal{A})$ is the set of pairs $\langle d, \mathsf{C} \rangle$ such that, according to $\mathcal{A}$, d is a part of an element of type C. For both measures the F1 is defined as usual. We run PWCA on the images of both PASCAL-Part and PartOf datasets where we remove the information about the part-of relation and the complex objects. Then we measure the distance of the output produced by PWCA with the full annotations in the datasets. As experimental setting we used the Pellet DL reasoner [96] in the BestMod procedure. We run the experiments on an Intel Xeon E5-1660 v3 3.00 GHZ, 16 core, 32 GB DDR4.

To evaluate the impact of semantic information in the problem of the recognition of the part-whole relation in images, we run different configurations of PWCA:

1. PWCA where the parameters α and β have been optimized ($\text{PWCA}_{\alpha,\beta}$);

2. PWCA with $\beta = 1.0$ and α optimized, that is, no semantic information is taken into account ($\text{PWCA}_{\alpha,\beta=1}$);

3. PWCA with optimal parameters α and β where the knowledge base has been extended with a set of cardinality constraints that restrict the number of parts of the same class for each complex object ($\text{PWCA}_{\alpha,\beta+\text{CA}}$).

4. Our previous version of PWCA [26, 25] (PWCA_{SOM}, see Algorithm 4) which proposes to solve the same task with a non-parametric clustering algorithm based on Self-Organizing Maps, see Section 5.1.2. The algorithm first encodes the segments

Algorithm 4: PARTWHOLECLUSTERINGALGORITHMSOM($\mathcal{P}, \mathcal{KB}$)

Input: A labelled picture $\mathcal{P} = \langle S, L \rangle$ and a knowledge base $\mathcal{KB}$
Output: A partial model $\mathcal{I}_p$ and its grounding $\mathcal{G}$ on $\mathcal{P}$

```
E ← ENCODE(P)
for l ← 1 to max_number_iterations do
    C ← SOM(E)
    (I_p^*, G^*) ← BESTMODSOM(P, C, KB)
    if I_p^* ⊨_p KB then
        return (I_p^*, G^*)
return (I_p^*, G^*)
```

of $\mathcal{P}$ into a data set $\mathcal{E}$ of feature vectors. For each segment in $\mathcal{P}$, its feature vector contains the centroid coordinates and the semantic distances, Equation (5.4), between the segment label and the label of the other segments. Here we consider the label with the highest weight. Now, the algorithm performs the clustering with the SOM algorithm and then the BESTMODSOM procedure creates an interpretation $\mathcal{I}_p^*$ and a grounding $\mathcal{G}^*$. The difference with the above BESTMOD procedure is that here we predict only the label of the complex object without considering the weights of the parts. In addition, BESTMODSOM returns only a logical interpretation, so the algorithm iterates the SOM clustering and the BESTMODSOM procedure until a partial model is found or a maximum number of iterations has reached. Therefore, PWCA_{SOM} does not guarantee that images are interpreted as partial models.

Table 5.4 shows the details of the evaluation:

1. The performance of $\text{PWCA}_{\alpha,\beta}$ on the PASCAL-PART test set and PARTOF dataset respectively (first and fourth line of the table) are encouraging. For both datasets we

	$prec_{\text{GRP}}$	rec_{GRP}	$F1_{\text{GRP}}$	$prec_{\text{COP}}$	rec_{COP}	$F1_{\text{COP}}$
$\text{PWCA}_{\alpha,\beta}$ (PASCAL-PART)	0.78	0.78	**0.74**	0.99	0.87	**0.92**
$\text{PWCA}_{\alpha,\beta=1}$ (PASCAL-PART)	0.75	0.55	0.59	0.97	0.83	0.89
$\text{PWCA}_{\alpha,\beta+\text{CA}}$ (PASCAL-PART)	0.79	0.66	0.7	0.99	0.87	**0.92**
$\text{PWCA}_{\alpha,\beta}$ (PARTOF)	0.66	0.92	**0.73**	0.91	0.81	**0.85**
$\text{PWCA}_{\alpha,\beta=1}$ (PARTOF)	0.32	0.68	0.37	0.89	0.75	0.8
$\text{PWCA}_{\alpha,\beta+\text{CA}}$ (PARTOF)	0.68	0.79	0.71	0.91	0.81	**0.85**
PWCA_{SOM} (PARTOF)	0.61	0.89	0.67	0.73	0.75	0.74

Table 5.4: Results of the partial models evaluation on the PASCAL-PART test set and on the PARTOF dataset. The first line for both datasets contains the results for $\alpha = 0.5, \beta = 0.3$, the second line the baseline with $\beta = 1.0$, the third line contains the results for $\alpha = 0.5, \beta = 0.3$ with all cardinality axioms in the knowledge bases. In bold the best F1 measures for **GRP** and **COP** for both datasets.

use the parameters $\alpha = 0.5$, $\beta = 0.3$, that maximize the performance of a portion of the PARTOF training set. The choice of reusing the parameters that maximize the "hardest" dataset well generalizes to a standard dataset. Indeed we obtain better results for the PASCAL-PART test set with respect to the PARTOF dataset.

2. To show the impact of background knowledge on the performance of PWCA, we compare PWCA with a **knowledge-blind baseline** ($\text{PWCA}_{\alpha,\beta=1}$). This is obtained by setting $\beta = 1.0$ in (5.5) with the effect of cancelling out the impact of the semantic quasidistance in the loss function. Comparing this baseline with the results of other settings, we can see that semantic features significantly improve the F1 measure on both datasets.

3. Most of the Semantic Web resources about meronomy (WORDNET [34]) have no cardinality axioms, such as $\mathsf{wholeObject} \sqsubseteq (= n)\mathsf{hasPart.part}$. To check if they are really necessary, we test how adding **cardinality axioms** to the knowledge base affects the performance of PWCA. Thus, we run PWCA with the knowledge base containing all cardinality axioms ($\text{PWCA}_{\alpha,\beta+\text{CA}}$). In both datasets we used the optimal parameters $\alpha = 0.5$, $\beta = 0.3$. Adding all cardinality axioms to the knowledge bases slightly improves the precision in the **GRP** measure at the price of a dramatic decrease of the recall. Whereas, cardinality axioms do not affect the **COP**. We can conclude that PWCA works better with simple part-whole knowledge bases with no cardinality axioms.

4. We compare PWCA with **our previous work** PWCA$_{SOM}$ [26]. We test PWCA$_{SOM}$ on PartOf dataset and we can see that PWCA outperforms our previous algorithm. As PWCA$_{SOM}$ does not guarantee that images are always interpreted in partial models, in our implementation the 8% of the cases generate interpretations which are not models. PWCA, instead, always returns partial models.

5.5 Discussion

Our first proposal to Semantic Image Interpretation is a well-founded and general framework that interprets an image as a partial model that minimizes a loss function. The construction of the partial model is guided by an agglomerative clustering algorithm, PWCA, that integrates semantic information with low-level numeric features and performs DLs reasoning. We evaluate the quality of the method on the task of part-whole detection between whole objects and their parts. We test the algorithm on two datasets, the results confirm that the integration of low-level features and semantic information improves the methods that rely only on numeric features. The advantages of PWCA are:

Unsupervision PWCA is completely unsupervised and can be easily adapted to specific domains by providing an appropriate part-whole knowledge base, for example WordNet [34]. Usually, providing a part-whole knowledge base is less expensive then producing a training set of images. More expressive knowledge bases can be automatically obtained from Web resources (such as, corpora, Web documents, RDF repositories) with the use of ontology learning techniques, see, for example, [20, 75].

Dealing with noise PWCA deals with multiple and noisy labels on parts. Given a segment labelled with many labels, PWCA selects the best label by taking into account also the labels assigned to the sibling segments. Thus, PWCA can discard labels with the highest weight in favour of labels with lower weight when these optimize a global labelling of the elements of the cluster.

Extension to events The heuristic of grouping simple objects according to their geometric and semantic proximity could be also applied to the relation participate-in an event. In this case, the simple objects are the participants at the event represented in the image, whereas the composite objects are the events themselves.

On the other hand, PWCA suffers of some limitations:

False positives bounding boxes The ability of PWCA to retrieve false negatives of the object detectors (the inference of a composite object from its parts) is counterbalanced by the lack of discarding false positives. For example, a wrong bounding box with label "eye" in the middle of a meadow is not recognized as false positive and thus is not discarded.

Heuristic Many types of events do not satisfy the heuristic of grouping objects close both in the geometric and in the semantic space. For example, in events such as look-at, above, flying-kite, jumping-on the participants can be distant in the space.

Handcrafted loss function The defined loss function partly depends on the considered relation. In this chapter, it has been defined for the part-whole relation but other relations, such as ride or next-to, require a different loss definition with additional features. In a large scale setting with many relations this can be a scalability issue.

Computational cost PWCA is computationally demanding: given an input labelled picture with $|S|$ bounding boxes, PWCA takes $O(|S|^3)$ operations to compute a semantically interpreted picture.

Therefore, in the next chapter we use a totally different (supervised), and more flexible, approach able to overcome these issues.

Chapter 6

Logic Tensor Networks

In the previous sections we developed a SII technique based on partial models. This models satisfy the logical constraints of a knowledge base and are built with a totally unsupervised algorithm. The advantage of being independent from a training set is counterbalanced by the hard generalization to many binary relations. To overcome this issue, we develop an alternative *supervised* method based on Neural Networks [9]. In the last years, Neural Networks have grown popularity due to their achievements in many real-life scenarios. Indeed, their great advantage is the ability to learn (that is, to progressively improve the performance on) tasks by considering examples, without a rule-based specification of the task. This allows machines, for example, to perform classical cognitive tasks, such as the classification of images, or natural language texts or speeches, according to some categories. Other application examples are more conceptual and can involve some reasoning, such as the machine translation or the link prediction on knowledge graphs. However, the application of Neural Networks to SII is not straightforward as SII presents some peculiarities that need to be considered. First, SII deals with *relational domains*, that is, domains with objects (in the image) that are connected each other through semantic relations. Second, data are uncertain. Indeed, the scene is cluttered so the objects can overlap each other and cannot be totally visible. Finally, a scene in a picture satisfies the *logical constraints* of the background knowledge of the domain. Indeed, we a priori know that people ride horses and not vice-versa. This available knowledge is of great help to semantically interpret pictures. Therefore, we need a technique able to learn from relational (uncertain) data in presence of logical constraints.

For these reasons, we apply Logic Tensor Networks (LTNs) [91] to SII. LTNs is a framework that integrates learning from numerical data and logical reasoning. The learning is based on Neural Tensor Networks (NTNs) [98] and the integration with reasoning is

performed with predicate Fuzzy Logic [44]. With this formulation, it is possible to satisfy the SII requirements, that is, learning relational data in presence of logical constraints and to reason with data affected by uncertainty.

In the following sections we provide an overview of Fuzzy Logic and NTNs as backgrounds for LTNs. Then, we describe in details the LTNs framework. All the explanations take into account the running example in Figure 2.1.

6.1 An Overview of Fuzzy Logic

Fuzzy Logic was introduced in 1965 by Lotfi Zadeh in order to model imprecise and vague propositions, such as: young man, small horse or enough food. Fuzzy Logic is a form of many-valued logic[1] that generalizes classical boolean logic by taking the truth values of formulas in the interval $[0, 1]$ instead of the set $\{0, 1\}$. In this way, Fuzzy Logic encodes a notion of *different levels of truth* where the extreme values $0, 1$ correspond to the absolute truth and falseness, respectively. The success of Fuzzy Logic is due to its ability to reason about facts that are inherently approximate and this is frequent in real-life applications. Common applications of Fuzzy Logic range from control theory to artificial intelligence in general regarding, for examples, domains such as medicine, economics and agriculture. In the following presentation the main considered references are [44, 22].

6.1.1 Syntax and Semantics of Fuzzy Logic

We first present the syntax and the semantics of a *propositional* Fuzzy Logic language.

Definition 6 (Propositional Fuzzy Logic Language)**.** *A* propositional fuzzy logic language *consists of the* propositional variables $p, q, r, \ldots$, *the connectives* $\rightarrow$ *(implication),* $\wedge$ *(conjunction),* $\vee$ *(disjunction), and the propositional constants* $\top, \bot$*.*

Here, a proposition is a sentence that can be assigned a truth value. The sentences "the grass is green", "the man on the horse is young" are propositions whereas the sentence "hurry up, ride the horse!" is not. This language is the same propositional language of classical logic [8]. The difference is in its semantics as we show in this section. Formulas are defined as usual: a propositional variable is a formula; if ϕ, ψ are formulas then $\phi \rightarrow \psi$, $\phi \wedge \psi$, $\phi \vee \psi$, are formulas. The connective $\neg$ is defined as: $\neg\phi$ is $\phi \rightarrow \bot$. In classical logic, the truth value of a formula is taken from the set $\{0, 1\}$ whereas in Fuzzy Logic the

[1]Many-valued logics are non classical logics that deal with more than two truth values. In this sense, Fuzzy Logic was already studied since the 1920s, as infinite-valued logic by Łukasiewicz and Tarski.

set is the interval $[0, 1]$ with its natural ordering $\leq$. This means that if $y \leq x$ then x is a stronger or equal truth than y. The truth value 0 means the total falsity and the value 1 total truth, as in classical logic. The generalization of classical logic resides in the fact that all the intermediate values are a continuum of truth values. The logical connectives (for example, conjunction, disjunction, negation and implication) have, as classical logic, functional semantics: the truth value of a complex formula is determined by the truth value of its components. The difference with respect to classical logic is that connectives are interpreted in a function from $[0, 1]^2$ to $[0, 1]$.

The conjunction connective in Fuzzy Logic is interpreted by a binary operation on truth values, called *t-norm*, which satisfies a minimal set of properties to capture the intuitive meaning of the conjunction[2].

Definition 7 (T-norm)**.** *A* t-norm *is a function T from $[0, 1]^2$ to $[0, 1]$ satisfying the following conditions:*

Commutative $T(x, y) = T(y, x)$*;*

Associative $T(x, T(y, z)) = T((x, y), z)$*;*

Non-decreasing *if $x \leq y$ then $T(z, x) \leq T(z, y)$;*

Zero and One $T(0, x) = 0$ *and* $T(1, x) = x$*.*

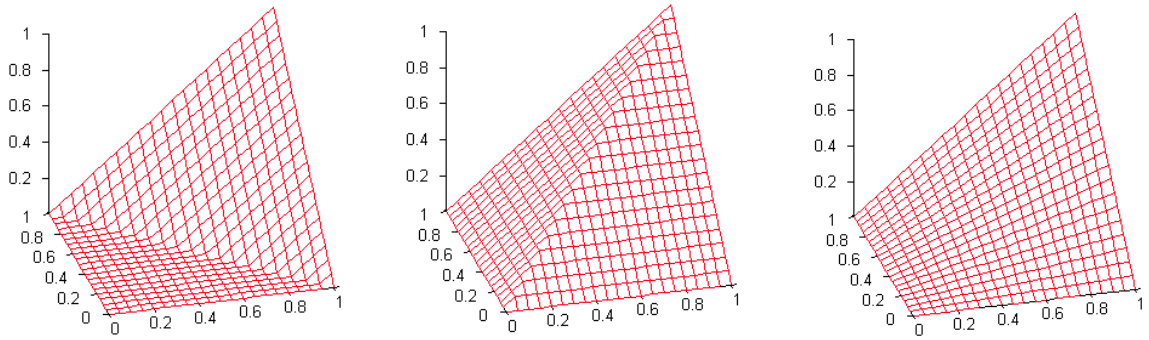
A t-norm is continuous if the function T is continuous in the usual sense. Figure 6.1 shows examples of the most important continuous t-norms.

The implication connective $\rightarrow$ is interpreted in a function $\Rightarrow$, called *residuum*, that satisfies the *residuation condition*. That is, let T be a t-norm and x, y be the truth values of the propositional formulas ϕ, ψ, respectively. Let $x \Rightarrow y$ be the truth value of $\phi \rightarrow \psi$, then for every $z \in [0, 1]$, holds:

$$T(x, z) \leq y \text{ iff } z \leq (x \Rightarrow y).$$

The intuition here is that the more $\phi \rightarrow \psi$ is true, the less additional information is carried by ϕ with respect to ψ. Therefore, the semantics of the implication can be intended as the maximum truth value to be "added" to x to obtain y. The function $\Rightarrow$ is called the *the residuum* of the t-norm T.

[2]In classical logic, the conjunction and the disjunction are *idempotent*, that is, $\phi \wedge \phi = \phi$ and $\phi \vee \phi = \phi$. In general, t-norms and conorms are not idempotent. Thus, to satisfy the idempotence, it is common to also introduce the *weak conjunction* $\wedge_w$ and the *weak disjunction* $\vee_w$. We exclude these connectives from the presentation as LTNs do not deal with them. Further details can be found in [22].



(a) Łukasiewicz t-norm: $T(x, y) = \max(0, x + y - 1)$

(b) Gödel t-norm: $T(x, y) = \min(x, y)$

(c) Product t-norm: $T(x, y) = x \cdot y$

Figure 6.1: Examples of the main Fuzzy Logic t-norms.

Definition 8 (Residuum of a t-norm)**.** *The* residuum of a t-norm T *is a function* $\Rightarrow$ *from* $[0, 1]^2$ *to* $[0, 1]$ *defined as:*

$$x \Rightarrow y = \max(\{z | T(x, z) \leq y\})$$

Some basic properties of the residuum are:

1. if $x \leq y$ then $x \Rightarrow y = 1$;
2. $(1 \Rightarrow x) = x$;
3. $(x \Rightarrow 1) = 1$;
4. if $x \leq y$ then $x = T(y, y \Rightarrow x)$;

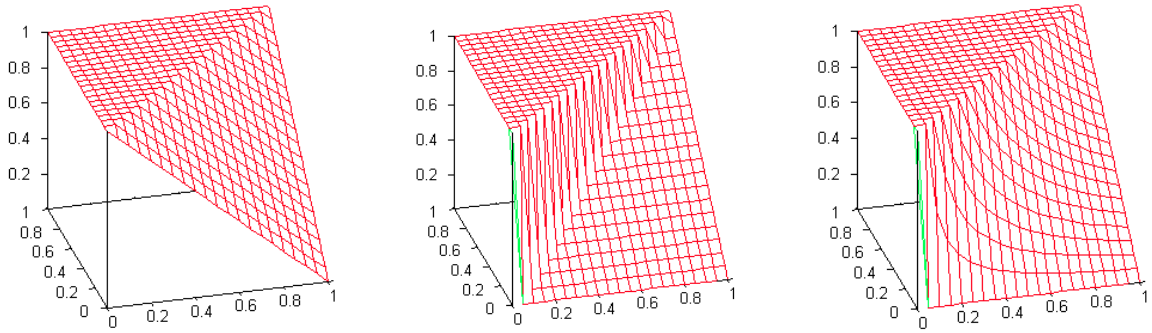
Figure 6.2 shows the residua for the three main introduced t-norms.

In classical logic the negation of a proposition $\neg\phi$ is defined as $\phi \rightarrow 0$. Fuzzy Logic proceeds in a similar manner:

Definition 9 (Precomplement)**.** *Given a t-norm* T *and its residuum* $\Rightarrow$*, the precomplement operator* $\sim$ *is defined as:*

$$\sim x = x \Rightarrow 0.$$

The precomplements of the mentioned t-norms are:



(a) Łukasiewicz residuum: $x \Rightarrow y = 1 - x + y$

(b) Gödel residuum: $x \Rightarrow y = y$

(c) Product residuum: $x \Rightarrow y = y/x$

Figure 6.2: Examples of the main Fuzzy Logic residua. If $x \leq y$ then $x \Rightarrow y = 1$.

$$\text{Łukasiewicz:} \quad \sim x = 1 - x$$

$$\text{Gödel:} \quad \sim x = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Product:} \quad \sim x = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

The logical disjunction is represented with the notion of *t-conorm* (also *s-norm*) S, a function that is dual to the t-norm.

Definition 10. *Given a t-norm T, the function $S(x, y)$ is a* t-conorm *if:*

$$S(x, y) = 1 - T(1 - x, 1 - y).$$

This generalizes De Morgan's Laws. A t-conorm satisfies the following properties:

Commutativity $S(x, y) = S(y, x)$;

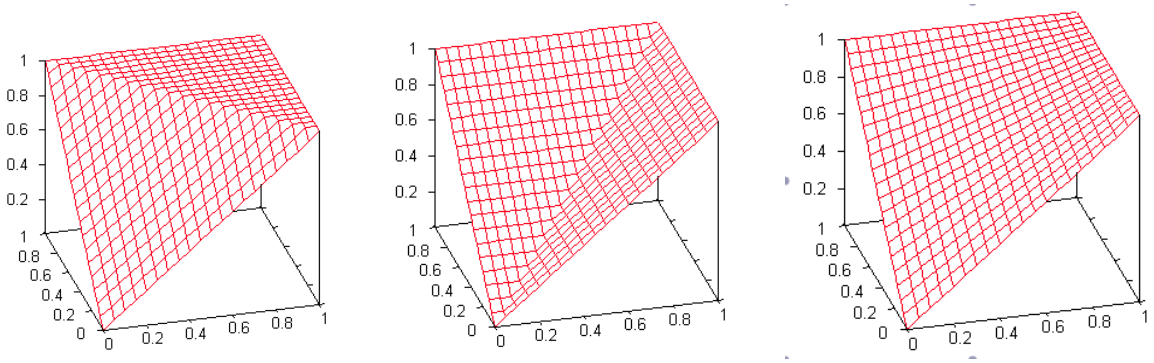
Associativity $S(x, S(y, z)) = S(S(x, y), z)$;

Non-decreasing if $x \leq y$ then $S(z, x) \leq S(z, y)$;

Zero and One $S(0, x) = x$ and $S(1, x) = 1$.

Figure 6.3 shows the t-conorms of the mentioned t-norms.

Now we explicit the semantics of propositional Fuzzy Logic formulas through t-norms. An *evaluation* of propositional variables is a function e that assigns each propositional



(a) Łukasiewicz t-conorm: $S(x, y) = \min(1, x + y)$

(b) Gödel t-conorm: $S(x, y) = \max(x, y)$

(c) Product t-conorm: $S(x, y) = x + y - xy$

Figure 6.3: Examples of the main Fuzzy Logic t-conorms.

variable p its truth value $e(p) \in [0, 1]$. The truth value of the formulas is given by:

$$\begin{aligned} e(\top) &= 1 \\ e(\bot) &= 0 \\ e(\phi \to \phi) &= e(\phi) \Rightarrow e(\phi) \\ e(\phi \wedge \psi) &= T(e(\phi), e(\psi)) \\ e(\phi \vee \psi) &= S(e(\phi), e(\psi)) \\ e(\neg\phi) &= \sim e(\phi). \end{aligned} \tag{6.1}$$

6.1.2 Predicate Fuzzy Logic

Propositional logic suffers of some expressivity limitations. For example, sentences like "all horses are animals", "there is a person who is riding a horse" or "every natural number is either even or odd" are very difficult to express (they assume to know a priori all the individuals of a "universe") or even impossible with a finite formula. First-Order Logic (or predicate logic) FOL [8] is introduced to overcome these limitations. In Fuzzy Logic the same operation is performed by introducing a *predicate language:*

Definition 11 (Predicate language)**.** *A* predicate language $\mathcal{PL}$ *consists of a non-empty set* $\mathcal{P}$ *of* predicate symbols*, a set* $\mathcal{F}$ *of* function symbols *and a set* $\mathcal{C}$ *of* constant symbols*. Predicate symbols are denoted with letters* P*,* Q*,* $R, \ldots$*, function symbols with* f*,* g*,* $h, \ldots$ *and constant symbols with* c*,* d*,* $e \ldots$ *A function/predicate symbol* s *is associated with a positive natural number* $\alpha(s)$ *denoting its arity. If* $\alpha(s) = n$ *we say that* s *is* n*-ary.*

Moreover, the logical symbols *are* variables $x, y, \ldots$, *connectives* $\rightarrow, \wedge, \vee$, truth constants $\top, \bot$ *and* quantifiers $\forall, \exists$.

The connective $\neg$ is defined as in propositional logic. *Terms* are variables or constants. If $t_1, \ldots, t_n$ are terms and f is a n-ary symbol function, then $f(t_1, \ldots, t_n)$ is a term. If $t_1, \ldots, t_n$ are terms and P is a n-ary predicate symbol, then $P(t_1, \ldots, t_n)$ is an *atomic formula*. An atomic formula is a formula. If ϕ, ψ are formulas, then $\top$, $\bot$, $\phi \rightarrow \psi$, $\phi \wedge \psi$, $\phi \vee \psi$ and $\neg\phi$ are formulas. If ϕ is a formula and x a variable, then $\forall x\phi$ and $\exists x\phi$ are formulas. This language is the same predicate language of FOL [8]. The difference is in its semantics as we show in what follows. The logical formulas of $\mathcal{PL}$ allow us to express relational information and general knowledge about a domain in a uniform manner. In the SII domain, $\mathcal{PL}$ formulas can express (i) simple facts, such as that the bounding box b contains a horse, written $\mathsf{Horse}(b)$, or the object in bounding box b_1 is riding the object in bounding box b_2, written $\mathsf{ride}(b_1, b_2)$; (ii) complex facts, such as that b contains an object that is both a horse and a mammal, written $\mathsf{Horse}(b) \wedge \mathsf{Mammal}(b)$; (iii) common knowledge, such as that the relation ride is asymmetric, written $\forall xy(\mathsf{ride}(x, y) \rightarrow \neg\mathsf{ride}(y, x))$, or that horses are ridden by people, written $\forall xy(\mathsf{Horse}(x) \wedge \mathsf{ride}(y, x) \rightarrow \mathsf{Person}(y))$. Now we explicit the semantics of predicate Fuzzy Logic formulas through t-norms.

Definition 12 (Structure)**.** *A* structure $\mathbf{M}$ *for a predicate language* $\mathcal{PL}$ *has the form:* $\langle M, (P_{\mathbf{M}})_{P\in\mathcal{P}}, (F_{\mathbf{M}})_{F\in\mathcal{F}}, (C_{\mathbf{M}})_{C\in\mathcal{C}}\rangle$ *where* M *is a non-empty* domain*; for each* n*-ary predicate symbol* $P \in \mathcal{P}$*,* $P_{\mathbf{M}} : M^n \rightarrow [0, 1]$ *is a function that maps* n*-tuples of domain elements in truth values; for each* n*-ary function symbol* $F \in \mathcal{F}$*,* $F_{\mathbf{M}} : M^n \rightarrow M$ *is a function that maps* n*-tuples of domain elements in domain elements; for each constant symbol* $C \in \mathcal{C}$*,* $C_{\mathbf{M}} : C \rightarrow M$ *is a function that maps constants to domain elements.*

Let $\mathbf{M}$ be a structure for $\mathcal{PL}$, an *evaluation* of the variables is a mapping v which assigns each variable x an element $v(x) \in M$. With the notation $v[x/a]$ we denote the evaluation that coincides with v on all the variables but x, which is assigned to a. Now, we have all the formalism to define the semantics of terms and formulas of predicate Fuzzy Logic by stating their truth values. Let $\mathbf{M}$ be a structure for $\mathcal{PL}$, v an evaluation, T a t-norm with $\Rightarrow$ the associated residuum and its dual s-norm S, the *truth values* of terms

and formulas is:

$$\begin{aligned}
\|x\|_{\mathbf{M},v} &= v(x);\\
\|\top\|_{\mathbf{M},v} &= 1;\\
\|\bot\|_{\mathbf{M},v} &= 0;\\
\|P(t_1,\ldots,t_n)\|_{\mathbf{M},v} &= P_{\mathbf{M}}(\|t_1\|_{\mathbf{M},v},\ldots,\|t_n\|_{\mathbf{M},v});\\
\|f(t_1,\ldots,t_n)\|_{\mathbf{M},v} &= F_{\mathbf{M}}(\|t_1\|_{\mathbf{M},v},\ldots,\|t_n\|_{\mathbf{M},v});\\
\|\phi\rightarrow\psi\|_{\mathbf{M},v} &= \|\phi\|_{\mathbf{M},v}\Rightarrow\|\psi\|_{\mathbf{M},v};\\
\|\phi\wedge\psi\|_{\mathbf{M},v} &= T(\|\phi\|_{\mathbf{M},v},\|\psi\|_{\mathbf{M},v});\\
\|\phi\vee\psi\|_{\mathbf{M},v} &= S(\|\phi\|_{\mathbf{M},v},\|\psi\|_{\mathbf{M},v});\\
\|\forall x\phi\|_{\mathbf{M},v} &= \inf\{\|\phi\|_{\mathbf{M},v[x/a]}|a\in M\};\\
\|\exists x\phi\|_{\mathbf{M},v} &= \sup\{\|\phi\|_{\mathbf{M},v[x/a]}|a\in M\}.
\end{aligned} \tag{6.2}$$

If the infimum or supremum do not exist, the truth value of the quantified formula is undefined.

6.2 An Overview of Neural Tensor Networks

Neural Tensor Networks (NTNs) [98] are Neural Networks [9] introduced for learning in relational domains [77]. These domains present the data as a *knowledge graph* consisting of labelled nodes (the entities or objects of the domain) connected by labelled edges (semantic relations between objects). For example, a dataset of semantically interpreted pictures is a relational domain. The main goals in relational domains are: (i) the predictions of missing labels in nodes or edges (also known as knowledge base completion); (ii) the prediction of properties of nodes (for example, features of objects); (iii) the clustering of nodes based on their connective patterns [72]. The prediction of a missing edge requires to predict the existence of an unseen triple $\langle subject, relation, object\rangle$. This prediction requires some reasoning. For example, the never seen triple $\langle \mathsf{Person}, \mathsf{drive}, \mathsf{Truck}\rangle$ can be inferred by the seen triple $\langle \mathsf{Person}, \mathsf{drive}, \mathsf{Car}\rangle$ by reasoning on the similarity between Car and Truck. Indeed, both have some wheels, a bodywork, an engine, they run and can be parked on streets. NTNs are effective models able to perform this kind of reasoning [98]. Therefore, the goal of NTNs is to learn a model that exploits common sense reasoning in order to predict missing edges from training knowledge graphs.

Let $\mathcal{E}$ be a set of entities (or objects) of a relational domain and each entity $e\in\mathcal{E}$ is

represented as a d-vector $\mathbf{e} \in \mathbb{R}^d$ of features[3]. Let $\mathcal{R}$ be a set of relations of the domain, a NTNs is a function $g : \mathcal{E} \times \mathcal{R} \times \mathcal{E} :\rightarrow \mathbb{R}$ that returns a score about the likelihood of the triple $\langle e_1, R, e_2 \rangle$, with $e_1, e_2 \in \mathcal{E}$, $R \in \mathcal{R}$. That is, the score is high if e_1 is in relation R with e_2, low otherwise. The function g has the following form:

$$g(\langle e_1, R, e_2 \rangle) = u_R^\intercal f \left(\mathbf{e}_1^\intercal W_R^{[1:k]} \mathbf{e}_2 + V_R \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix} + b_R \right), \tag{6.3}$$

where $f = \tanh$ is a standard non-linearity applied element-wise. The element $W_R^{[1:k]}$ is a tensor that is used to perform the bilinear tensor product $\mathbf{e}_1^\intercal W_R^{[1:k]} \mathbf{e}_2$ resulting in a vector $h \in \mathbb{R}^k$. Each entry of h is computed by one slice $i = 1 \dots k$ of the tensor W_R: $h_i = \mathbf{e}_1^\intercal W_R^{[i]} \mathbf{e}_2$. The other components of the formula are the standard parameters of neural networks: $V_R \in \mathbb{R}^{k \times 2d}$, $b_R \in \mathbb{R}^k$ and $u_R \in \mathbb{R}^k$. In particular, the vector parameter u_R linearly combines the features returned by the non-linearity function f for returning a score. Figure 6.4 shows a graphical representation of the NTN equation. The advantage of

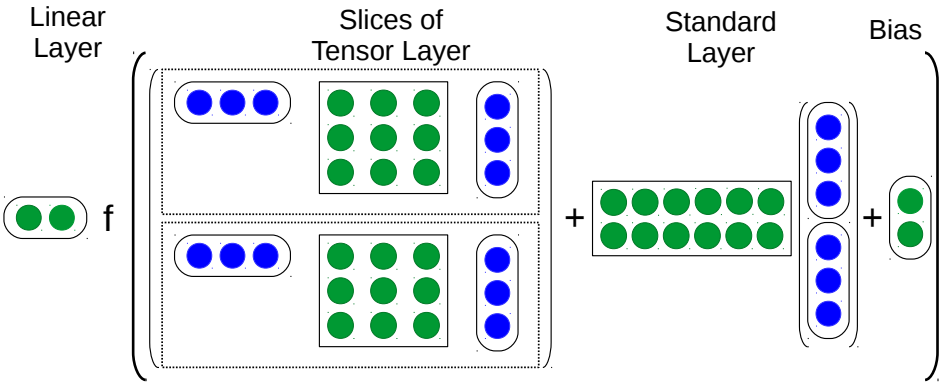


Figure 6.4: Visualization of Neural Tensor Network. The first (from left) linear layer is the parameter vector u, the dashed boxes are the slices of the tensor W ($k = 2$) and contain the bilinear product $\mathbf{e}_1^\intercal W^{[1:k]} \mathbf{e}_2$ (the input entities $\mathbf{e}_1, \mathbf{e}_2$ are blue vectors with $d = 3$). Then the standard layer $V \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix}$ and the bias b follow. The network parameters are represented with green color.

this network, with respect to previous models of Neural Networks for relational domains, is that it learns the joint interaction of the entity vectors with the tensor of parameters $W_R^{[1:k]}$. The several k slices of the tensor map the pair of entities $\langle \mathbf{e}_1, \mathbf{e}_2 \rangle$ in a k-dimensional latent space for the relation R that is able to capture different instantiations of R [98]. For example, the partOf relation can occur between animals and their parts or between vehicles

[3]These features can be a learnt embedding of e or manually defined.

and their parts. Moreover, the non-linearity given by the tanh function increases the NTN expressive power. This leads to good performance in the knowledge base completion task [98]. NTN is a generalization of the bilinear model $g(\langle e_1, R, e_2\rangle) = \mathbf{e}_1^\intercal W_R \mathbf{e}_2$ for relational data [72], which is obtained setting $k = 1$, $V_R = 0$, $b_R = 0$ and $f =$ identity.

6.3 Logic Tensor Networks

Logic Tensor Networks (LTNs) [91] adopt the same syntax of a First-Order predicate language $\mathcal{PL}$ (with the symbols in the signature $\Sigma = \langle \mathcal{P}, \mathcal{F}, \mathcal{C}\rangle$, see Section 6.1) to express relational information and a priori knowledge about a domain. LTNs semantics deviates from the standard *abstract* semantics of Predicate Fuzzy Logic towards a *concrete* semantics. Indeed, the interpretation domain is a subset of $\mathbb{R}^n$, that is, constant symbols and closed terms are associated with a n-dimensional tuple of real numbers. The intuition is that this n-vector encodes n numerical features of an object. These features can be manually engineered. In the SII domain, the engineered features of a bounding box can be the score confidence of an object detector for several classes of objects, the bounding box coordinates, the area, the colors, etc. Functional symbols are interpreted as real-valued functions and predicates are interpreted as functions on real vectors to the interval $[0, 1]$. The intuition is that the interpretation of a formula represents the degree of truth for that formula: an higher value means a higher degree of truth. To emphasize the fact that in LTNs the interpretation of $\mathcal{PL}$ formulas is in a "real" world the term *(semantic) grounding*, denoted by $\mathcal{G}$, is used instead of the standard term *interpretation*[4]. The idea underlying the notion of grounding is that a grounding has to capture the latent correlation between the features of objects and their categorical/relational properties. In any case, the grounding is synonym of interpretation in the predicate Fuzzy Logic.

Definition 13. *An* n-grounding *(with* $n \in \mathbb{N}$, $n > 0$*), or simply* grounding, $\mathcal{G}$ *for a First-Order Language* $\mathcal{PL}$ *is a function from the signature of* $\mathcal{PL}$ *that satisfies the conditions:*

1. $\mathcal{G}(c) \in \mathbb{R}^n$ *for every constant symbol* $c \in \mathcal{C}$*;*
2. $\mathcal{G}(f) \in \mathbb{R}^{n \cdot \alpha(f)} \longrightarrow \mathbb{R}^n$ *for every functional symbol* $f \in \mathcal{F}$*;*
3. $\mathcal{G}(P) \in \mathbb{R}^{n \cdot \alpha(P)} \longrightarrow [0, 1]$ *for every predicate symbol* $P \in \mathcal{P}$*.*

[4] In logic literature, the term "grounding" is the replacing of the variables of a term/formula with constants, or terms not containing other variables.

Given a grounding $\mathcal{G}$ and let $term(\mathcal{PL}) = \{t_1, t_2, t_3, \ldots\}$ be the set of closed terms of $\mathcal{PL}$ (that is, the terms that do not contain any variable), the semantics of closed terms and atomic formulas is inductively defined as follows:

$$\begin{aligned} \mathcal{G}(f(t_1, \ldots, t_m)) &= \mathcal{G}(f)(\mathcal{G}(t_1), \ldots, \mathcal{G}(t_m)) \\ \mathcal{G}(P(t_1, \ldots, t_m)) &= \mathcal{G}(P)(\mathcal{G}(t_1), \ldots, \mathcal{G}(t_m)). \end{aligned} \tag{6.4}$$

The semantics for non-atomic formulas is defined according to t-norms functions used in propositional and predicate Fuzzy Logic, see Equation (6.1). If we take, for example, the Łukasiewicz t-norm, we have:

$$\begin{aligned} \mathcal{G}(\phi \to \psi) &= \min(1, 1 - \mathcal{G}(\phi) + \mathcal{G}(\psi)) \\ \mathcal{G}(\phi \wedge \psi) &= \max(0, \mathcal{G}(\phi) + \mathcal{G}(\psi) - 1) \\ \mathcal{G}(\phi \vee \psi) &= \min(1, \mathcal{G}(\phi) + \mathcal{G}(\psi)) \\ \mathcal{G}(\neg\phi) &= 1 - \mathcal{G}(\phi). \end{aligned} \tag{6.5}$$

LTNs provide a semantics also for universal and existential quantifiers that differs from the semantics of classical Fuzzy Logic. Indeed, the interpretation for the universal quantifier (Section 6.1.2) leads to define $\mathcal{G}(\forall x \phi(x))$ for LTNs as:

$$\mathcal{G}(\forall x \phi(x)) = \inf\{\mathcal{G}(\phi(t)) | t \in term(\mathcal{PL})\}. \tag{6.6}$$

This definition is not in the spirit of LTNs as it does not tolerate exceptions. Indeed, the presence of an exception to a universally-quantified formula (for example, $\forall xy(\mathsf{Horse}(x) \wedge \mathsf{ride}(y, x) \to \mathsf{Person}(y))$ such as a dog riding a horse in a circus, would falsify the whole formula. LTNs handle these outliers giving a higher truth-value to the formula $\forall x \phi(x)$ if many examples satisfy $\phi(x)$. Thus, the interpretation of a universal quantified constraints $\forall x \phi(x)$ is that *normally* $\phi(x)$ holds. This aspect fulfils the requirements of a SII systems. Indeed, in a visual scene (due to occlusions, perspective of the view or unexpected situations) the logical constraints of common knowledge are not always respected. Formally, the semantics for the universal quantifier is defined in terms of *mean*-operator:

Definition 14. *Let* $mean_p(x_1, \ldots, x_d) = \left(\frac{1}{d} \sum_{i=1}^{d} x_i^p\right)^{\frac{1}{p}}$, *with* $p \in \mathbb{Z}$, $d \in \mathbb{N}$, *the grounding for* $\forall x \phi(x)$ *is*

$$\mathcal{G}(\forall x \phi(x)) = \lim_{d \to |term(\mathcal{PL})|} mean_p(\mathcal{G}(\phi(t_1)), \mathcal{G}(\phi(t_2)), \ldots, \mathcal{G}(\phi(t_d))). \tag{6.7}$$

This definition states that the grounding of a universally quantified formula $\forall x\phi(x)$ is the mean of the d groundings of the quantifier-free formula $\phi(x)$, with d that tends to all the terms of the predicate language $\mathcal{PL}$. The limit operator is used as the set of terms $term(\mathcal{PL})$ is infinite due to the recursive application of function symbols to simpler terms. Examples of popular mean operators can be obtained by changing the parameter p in the definition, for example, $p = -1$ gives the harmonic mean, $p = 1$ the arithmetic mean and $p = 2$ the geometric mean.

The semantics of the existential quantifier is usually determined by the semantics of the universal quantifier by exploiting the equivalence between $\exists$ and $\neg\forall\neg$. However, this approach presents a drawback in LTNs: if we consider, for instance, the Lukasiewicz t-norm and the arithmetic mean, the semantics of $\forall$ becomes $\mathcal{G}(\forall x\phi(x)) = \mathcal{G}(\exists x\phi(x))$. Therefore, in LTNs existential quantification is interpreted via Skolemization: every formula of the form $\forall x_1, \ldots, x_n(\ldots \exists y\phi(x_1, \ldots, x_n, y))$ is rewritten as

$$\forall x_1, \ldots, x_n(\ldots \phi(x_1, \ldots, x_n, f(x_1, \ldots, x_n)))$$

by introducing the so-called *Skolem function* $f(x_1, \ldots, x_n)$, that is a new n-ary function symbol. In this way, the Skolem functions allow the removal of existential quantifiers from the logical formulas of LTNs. For example, if we consider the formula $\forall x(\mathsf{Person}(x) \rightarrow \exists y(\mathsf{partOf}(y, x) \wedge \mathsf{Leg}(y))$ the existential quantification of variable y is translated in the Skolem function $\mathsf{legOf}(x)$ applied to x (which is the universally quantified variable) and the quantifier $\exists y$ occurs in the scope of its quantification. The final formula is

$$\forall x(\mathsf{Person}(x) \rightarrow \mathsf{partOf}(\mathsf{legOf}(x), x) \wedge \mathsf{Leg}(\mathsf{legOf}(x)).$$

A function suitable for a grounding should preserve some form of *regularity*. Let $b \in \mathcal{C}$ a constant that refers to, for example, a bounding box that contains a horse. Let $\mathbf{v}$ the feature vector returned by the grounding of b, that is, $\mathcal{G}(b) = \mathbf{v}$, then it holds that $\mathcal{G}(\mathsf{Horse})(\mathbf{v}) \approx 1$. Moreover, for every bounding box with feature vector $\mathbf{v}'$ similar to $\mathbf{v}$, $\mathcal{G}(\mathsf{Horse})(\mathbf{v}') \approx 1$ holds. In the following we show some functions for groundings.

6.3.1 Rule-Based Grounding

In the above the syntax and semantics of LTNs have been introduced without stating how to compute the groundings for functions and atomic formulas. These groundings can be explicitly defined in terms of real functions (rule-based groundings), learnt from examples or a combination of both approaches. For example, the grounding of some predicates can be computed with a rule-based method, whereas it can be learnt for other predicates. In the following we provide some examples of rule-based groundings.

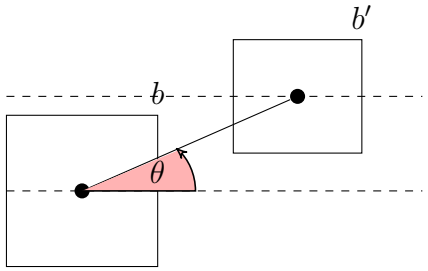


Figure 6.5: The angle between two bounding boxes.

Regarding the **constant** symbols of bounding boxes, we denote with $score(C, b)$ the classification score of an object detector (for example, Fast R-CNN [36]) for the constant symbol b for the class $C \in \mathcal{P}_1$. The bounding box coordinates for b are denoted with $\langle x_0(b), y_0(b), x_1(b), y_1(b)\rangle$, where $\langle x_0(b), y_0(b)\rangle$ is the top-left corner of b and $\langle x_1(b), y_1(b)\rangle$ is the bottom-right corner of b. The grounding for a bounding box constant $b \in \mathcal{C}$ can be defined as the feature vector $\mathbf{v}_b$:

$$\langle score(C_1, b), \ldots, score(C_{|\mathcal{P}_1|}, b), x_0(b), y_0(b), x_1(b), y_1(b)\rangle.$$

A simple grounding for **unary predicates** in $\mathcal{P}$ could totally rely on the score returned by an external classifier of objects. In our running example, we suppose to have an object detector trained on the classes of the example. The object detector returns a score $score(C, b)$ regarding the possibility that bounding box b contain an object of class C. Let $b \in \mathcal{C}$ a constant symbol that refers to a bounding box in the picture, the grounding for the predicate $C \in \mathcal{P}$ is:

$$\mathcal{G}(C(b)) = \sigma(score(C, b))$$

with σ the sigmoid function that maps the score of the object detector in the interval $[0, 1]$. A rule-based grounding can be defined also for **binary predicates**. Each predicate needs its own rule and we present examples for spatial relations and for the partOf predicates. The choice of the spatial relations is due to the fact that some works [68, 10] define the spatial relations as *membership functions* [111]. A membership function returns the degree of membership of an element to a set. In Fuzzy Logic, a membership function represents the truth value of an atomic formula. Let $b, b' \in \mathcal{C}$ be two bounding box constants, and θ be the angle made by the line passing through the centroids of b and b' and the x-axis, see Figure 6.5. The groundings for the spatial relations right of, below, above, left of between

b, b' are the membership functions proposed in [68]:

$$\begin{aligned}
\mathcal{G}(\mathsf{right\ of}(b,b')) &= \begin{cases} \cos^2(\theta) & -\pi/2 \leq \theta \leq \pi/2 \\ 0 & \text{otherwise} \end{cases} \\
\mathcal{G}(\mathsf{below}(b,b')) &= \begin{cases} \sin^2(\theta) & 0 \leq \theta \leq \pi \\ 0 & \text{otherwise} \end{cases} \\
\mathcal{G}(\mathsf{above}(b,b')) &= \begin{cases} \sin^2(\theta) & -\pi \leq \theta \leq 0 \\ 0 & \text{otherwise} \end{cases} \\
\mathcal{G}(\mathsf{left\ of}(b,b')) &= \begin{cases} \cos^2(\theta) & -\pi \leq \theta \leq -\pi/2, \pi/2 \leq \theta \leq \pi \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Regarding the binary predicate partOf, a possible grounding is based on the naïve hypothesis that the more a bounding box b is contained in a bounding box b', the higher the possibility that b is part of b'. Thus, $\mathcal{G}(\mathsf{partOf}(b,b'))$ can be defined as the *inclusion ratio* $ir(b,b')$ of bounding box b into bounding box b', that is, how much b is included into b':

$$\mathcal{G}(\mathsf{partOf}(b,b')) = ir(b,b') = \frac{intersec(b,b')}{area(b)}, \tag{6.8}$$

where $intersec(b,b')$ is the area of the intersection of b with b' and $area(b)$ is the area of bounding box b. This rule-based grounding for partOf can be refined in a more accurate manner by taking into account also the *type compatibility* between parts and whole objects, that is, by multiplying the inclusion ratio with a factor w_{ij}:

$$\mathcal{G}(\mathsf{partOf}(b,b')) = \begin{cases} 1 & \text{if } ir(b,b') \cdot \max_{i,j=1}^{|\mathcal{P}_1|}(w_{ij} \cdot v_i \cdot v'_j) \geq th_{ir} \\ 0 & \text{otherwise} \end{cases} \tag{6.9}$$

with th_{ir} a threshold (for example $th_{ir} > 0.5$), and $w_{ij} = 1$ if C_i is part of C_j, and 0 otherwise.

However, explicitly defining a grounding could present some drawbacks. First, these groundings are heuristics that need to be handcrafted and, in a large scale setting, this could represent a scalability issue. Second, these groundings can be inaccurate. For example, the classification score of the bounding box detector may be wrong; due to the perspective of the picture an object behind another one could have the bounding box coordinates that satisfy the grounding for above; a bounding box could be included into another without being in part-of relation, etc. For these reasons, in the next section we present a method for learning groundings from data.

6.3.2 Tensor-Network-Based Grounding

Learning the grounding from data needs the use of parametric functions whose parameters are tweaked in a training process that tries to fit the training data. The LTNs approach could be applied to any family of parametric functions [91] such as, linear or polynomial functions, Gaussian mixtures, neural networks. However, LTNs focus on functions based on linear transformations and on NTNs, see Section 6.2.

The grounding for **function symbols** is based on linear transformations from $\mathbb{R}^{mn}$ to $\mathbb{R}^n$. Let $b_1, \ldots, b_m$ be some constant symbols in $\mathcal{C}$ with feature vectors $\mathbf{v}_i = \mathcal{G}(b_i) \in \mathbb{R}^n$, with $i = 1 \ldots m$, and $\mathbf{v} = \langle \mathbf{v}_1; \ldots; \mathbf{v}_m \rangle$ is a mn-ary vector given by the vertical stacking of each feature vector $\mathbf{v}_i$. If $f(b_1, \ldots, b_m)$ is a m-ary function then:

$$\mathcal{G}(f)(\mathbf{v}) = M_f \mathbf{v} + N_f. \tag{6.10}$$

The parameters of $\mathcal{G}(f)$ are the $n \times mn$ real matrix M_f and the n-vector N_f. Regarding the **predicate symbols**, the grounding $\mathcal{G}(P)$ of an m-ary predicate $P(b_1, \ldots, b_m)$ is a generalization of the neural tensor network of Section 6.2 as a function from $\mathbb{R}^{mn}$ to $[0, 1]$:

$$\mathcal{G}(P)(\mathbf{v}) = \sigma\left(u_P^\intercal \tanh\left(\mathbf{v}^\intercal W_P^{[1:k]} \mathbf{v} + V_P \mathbf{v} + b_P\right)\right) \tag{6.11}$$

with σ the sigmoid function. The LTNs version of the mentioned tensor network focus on the generalization to m-ary predicates instead of only binary predicates. Indeed, the input vector $\mathbf{v}$ is the concatenation of all the m groundings $\mathbf{v}_1, \ldots, \mathbf{v}_m$ of the input elements $b_1, \ldots, b_m$ of P. Moreover, the original formulation of the tensor network returns a score regarding how much the two input elements are related with the predicate P. LTNs, instead, use the sigmoid σ to return the truth value of the atomic formula $P(b_1, \ldots, b_m)$. The parameters for P are: $u_P \in \mathbb{R}^k$, a 3-D tensor $W_P^{[1:k]} \in \mathbb{R}^{k \times mn \times mn}$, $V_P \in \mathbb{R}^{k \times mn}$ and $b_P \in \mathbb{R}^k$. The parameter u_P computes a linear combination of the quadratic features returned by the tensor product. With Equations (6.10) and (6.11) the grounding of a complex LTNs formula can be computed by first computing the groundings of the closed terms and the atomic formulas contained in the complex formula. Then, these groundings are combined using a specific t-norm, see Equation 6.5.

6.3.3 Learning as Best Satisfiability

In the previous section, we define the logic on which LTNs is found along with its syntax and semantics. The central notion of LTNs is the grounding $\mathcal{G}$ for constants, functions and predicates. The grounding for constants is defined as a feature vector associated to

the constant of the domain. The grounding for functions and predicates is a parametric function. Unfortunately, an a priori definition of the grounding is not possible in general and it should be learned from data. This learning is performed by optimizing the truth values of the formulas in a LTNs knowledge base. In the following the introduce the notion of *grounded theory* (a LTNs knowledge base) and see how the learning of a grounding is performed. We start with the notion of *partial grounding* $\hat{\mathcal{G}}$, that is a grounding defined on a subset of the signature of $\mathcal{PL}$. A grounding $\mathcal{G}$ for $\mathcal{PL}$ is a *completion* of $\hat{\mathcal{G}}$ (in symbols $\hat{\mathcal{G}} \subseteq \mathcal{G}$) if $\mathcal{G}$ coincides with $\hat{\mathcal{G}}$ on the symbols where $\hat{\mathcal{G}}$ is defined.

Definition 15. *A* grounded theory GT *is a pair* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ *with* $\mathcal{K}$ *a set of closed formulas and* $\hat{\mathcal{G}}$ *a partial grounding.*

Here follows the definition of *satisfiability* of a grounded theory in LTNs.

Definition 16. *A grounding* $\mathcal{G}$ *satisfies a grounded theory* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ *if* $\hat{\mathcal{G}} \subseteq \mathcal{G}$ *and* $\mathcal{G}(\phi) = 1$*, for all* $\phi \in \mathcal{K}$*. A grounded theory* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ *is* satisfiable *if there exists a grounding* $\mathcal{G}$ *that satisfies* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$*.*

According to the above definition, the satisfiability of $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ can be obtained by searching for a grounding $\mathcal{G}$ that extends $\hat{\mathcal{G}}$ such that *every* formula in $\mathcal{K}$ has value 1. When a grounded theory is not satisfiable a user can be interested in a degree of satisfaction of the GT. This is defined as follows with the notion of *best satisfiability.*

Definition 17. *Let* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ *be a grounded theory, the* best satisfiability problem *amounts at searching an extension* $\mathcal{G}^*$ *of* $\hat{\mathcal{G}}$ *in* $\mathbb{G}$ *(the set of all possible groundings) that maximizes the truth value of the conjunction of the formulas in* $\mathcal{K}$*:*

$$\mathcal{G}^* = \underset{\hat{\mathcal{G}} \subseteq \mathcal{G} \in \mathbb{G}}{\operatorname{argmax}} \mathcal{G}\left(\bigwedge_{\phi \in \mathcal{K}} \phi\right). \tag{6.12}$$

The best satisfiability problem can be seen as an optimization problem on the set of parameters to be learned. Let $\Theta = \{M_f, N_f \mid f \in \mathcal{F}\} \cup \{W_P, V_P, b_P, u_P \mid P \in \mathcal{P}\}$ be the set of parameters. Let $\mathcal{G}(\cdot|\Theta)$ denote the grounding obtained by setting the parameters of the parametric functions to Θ. Thus, the best satisfiability problem aims at finding the best set of parameters Θ such that:

$$\Theta^* = \operatorname{argmax}_\Theta \mathcal{G}\left(\bigwedge_{\phi \in \mathcal{K}} \phi \middle| \Theta\right) - \lambda \|\Theta\|_2^2 \tag{6.13}$$

with $\lambda \|\Theta\|_2^2$ a smoothing factor of the parameters to avoid data over fitting. The grounding of a grounded theory is given by the considered t-norm applied to all the formulas in $\mathcal{K}$. Given the associativity of t-norms, the grounding computation (that is, one step of the optimization of Equation (6.13)) is linear with respect to the formulas in $\mathcal{K}$.

6.3.4 From Knowledge Bases to Tensor Networks

In the following, we graphically show how to encode a set of formulas in $\mathcal{K}$ in Logic Tensor Networks. The idea is: for every formula in $\mathcal{K}$ we build a LTN and then aggregate all the networks with an *and* operator according to Equation (6.12). Let us consider $\mathcal{K}$ containing only the formula $drive(x, y) \rightarrow Vehicle(y)$, with input elements $\mathbf{v}, \mathbf{u}$. The computation of the grounding of this formula is given by the Logic Tensor Network in Figure 6.6. In this tensor network the parameters to be learned are $W_d, V_d, b_d, u_d, W_V, V_V, b_V, u_V$ (d stands for *drive* and V stands for $Vehicle$). The learning is performed through maximization of the degree of satisfiability (the truth value) of $drive(x, y) \rightarrow Vehicle(y)$, see previous section. However, a knowledge base $\mathcal{K}$ may contain many formulas: $\mathcal{K} = \{\phi_1, \ldots \phi_q\}$,

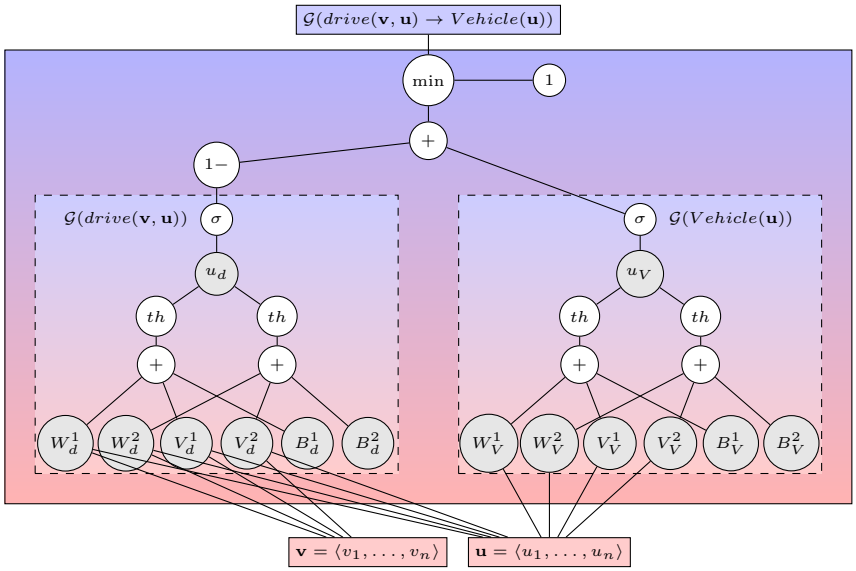


Figure 6.6: Logic Tensor Network for the formula $drive(x, y) \rightarrow Vechicle(y)$, with $\mathcal{G}(x) = \mathbf{v}$, $\mathcal{G}(y) = \mathbf{u}$ and $k = 2$. The considered t-norm is the Łukasiewicz. Therefore, $\mathcal{G}(drive(\mathbf{v}, \mathbf{u}) \rightarrow Vehicle(\mathbf{u})) = \min(1, 1 - \mathcal{G}(drive(\mathbf{v}, \mathbf{u})) + \mathcal{G}(Vehicle(\mathbf{u})))$.

and the grounding $\mathcal{G}(\phi)$ is computed for every formula $\phi \in \mathcal{K}$ obtaining a set of LTNs. Notice that the blocks (the dashed rectangles in Figure 6.6) of the networks correspond to predicates in $\mathcal{PL}$ that can appear in many formulas. Therefore, some blocks can be linked with blocks of other formulas, thus composing a very complex network. Finally, the outputs of the networks, that is, the groundings $\mathcal{G}(\phi)$, with $\phi \in \mathcal{K}$, are connected with some operators that implement the $\bigwedge$ operator (according to the chosen t-norm) that returns the grounding $\mathcal{G}(\mathcal{K})$ of the whole knowledge base $\mathcal{K}$, see Equation (6.12).

Chapter 7

Semantic Image Interpretation with Logic Tensor Networks

The goal of this chapter is to introduce how the SII problem can be solved by using LTNs. Suppose you have a dataset that is useful for learning and evaluating the approach. Such a dataset contains relational data, that is, objects and pair of objects labelled with a set of labels. This dataset consists of pictures annotated with bounding boxes around the objects. The labels (also known as *semantic classes* or *object types*) for the bounding boxes describe the physical objects contained in the bounding boxes. For example, if a bounding box b_1 contains a person its labels are Person and Animal. Whereas the labels between pairs of bounding boxes describe the semantic relations (also called *visual relationships*) between the physical objects in the bounding boxes. For example, let us suppose to have a second bounding box b_2 under b_1 and labelled with Horse and Animal, the pair $\langle b_1, b_2 \rangle$ is labelled with the relations ride and on. This representation can be encoded with a knowledge graph, see Section 6.2. This kind of datasets (and in general relational data) can be easily formalized with LTNs with a set of atomic formulas: $\{\mathsf{Person}(b_1), \mathsf{Animal}(b_1), \mathsf{Horse}(b_2), \mathsf{Animal}(b_2), \mathsf{ride}(b_1, b_2), \mathsf{on}(b_1, b_2)\}$.

However, the information in the dataset is *incomplete*, that is, many bounding boxes have no label annotations, or there are missing relations between bounding boxes or even some pictures have no annotations at all. Therefore, the LTNs goal is to exploit the information encoded in the dataset to complete the missing information. This task is also called knowledge base completion, see the previous chapter. LTNs are developed to encode and solve this task with the help of logical constraints. Indeed, the available SII data can be encoded with a grounded theory $\mathcal{T}_{\text{SII}} = \langle \mathcal{K}_{\text{SII}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$, where $\mathcal{K}_{\text{SII}}$ is a LTNs knowledge base and $\hat{\mathcal{G}}_{\text{SII}}$ is a partial grounding. The knowledge base encodes, with LTNs

formulas, the bounding box annotations in the dataset and some background knowledge about the domain. The task is to complete the partial knowledge of the SII dataset by finding a grounding $\mathcal{G}^*_{\text{SII}}$, that extends $\hat{\mathcal{G}}_{\text{SII}}$, such that:

$$\begin{aligned} \mathcal{G}^*_{\text{SII}}(C(b)) &\mapsto [0,1] \\ \mathcal{G}^*_{\text{SII}}(R(b_1,b_2)) &\mapsto [0,1] \end{aligned}$$

for every unary (C) and binary (R) predicate symbol and for every (pair of) bounding box in the dataset[1]. The grounding $\mathcal{G}^*_{\text{SII}}$ is found by maximizing the satisfiability of $\mathcal{T}_{\text{SII}}$, see Equation (6.12).

The chapter follows with the explanation of the single components of the grounded theory $\mathcal{T}_{\text{SII}} = \langle \mathcal{K}_{\text{SII}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$: Section 7.1 explains the LTNs knowledge base $\mathcal{K}_{\text{SII}}$, whereas Section 7.2 explains the computation of the grounding $\hat{\mathcal{G}}_{\text{SII}}$. Section 7.3 presents some considerations about the whole optimization process. Finally, Section 7.4 shows the modules of the Python package that implements LTNs for SII.

7.1 The Knowledge Base $\mathcal{K}_{\text{SII}}$

Let *Pics* be a SII dataset and $B(p)$ be the set of bounding boxes in picture $p \in Pics$. Each bounding box (and pairs of bounding boxes) is annotated with a set of labels, as described above. Let $\Sigma_{\text{SII}} = \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ be a $\mathcal{PL}$ signature where $\mathcal{P} = \{\mathcal{P}_1\} \cup \{\mathcal{P}_2\}$ is the set of predicates. $\mathcal{P}_1$ is the set of unary predicates that are the object types used to label the bounding boxes, for example, $\mathcal{P}_1 = \{\mathsf{Horse}, \mathsf{Person}, \mathsf{Elephant}, \mathsf{Leg}, \mathsf{Pizza}, \mathsf{Shirt}, \dots\}$. The set $\mathcal{P}_2$ contains binary predicates used to label pairs of bounding boxes, for example, $\mathcal{P}_2 = \{\mathsf{partOf}, \mathsf{ride}, \mathsf{on}, \mathsf{under}, \mathsf{wear}, \mathsf{eat}, \dots\}$. The set $\mathcal{F}$ is left empty as the SII task of partial knowledge completion does not require function symbols. Let $\mathcal{C} = \bigcup_{p \in Pics} B(p)$ be the set of constants for all the bounding boxes in the dataset *Pics*. The knowledge base $\mathcal{K}_{\text{SII}} = \mathcal{POS} \cup \mathcal{NEG} \cup \mathcal{BK}$ is a set that contains three kind of information: the positive facts $\mathcal{POS}$, the negative facts $\mathcal{NEG}$ and the background knowledge $\mathcal{BK}$.

Positive Facts This set of information regards the labels of the annotations of the (pairs of) bounding boxes. These facts are the positive examples for learning the predicates. The positive examples for a semantic class C are the atomic formulas $C(b)$, for every bounding box b labelled with class $C \in \mathcal{P}_1$ in the pictures of the dataset. The positive examples for a relation R are the atomic formulas $R(b_1, b_2)$, for every pair of bounding

[1]For SII we do not use function symbols.

boxes $\langle b_1, b_2 \rangle$ labelled with the binary relation $R \in \mathcal{P}_2$ in the pictures of the dataset. We denote with ϕ^+ the set containing all the positive examples (that is, the positive atomic formulas) for the predicate $\phi \in \mathcal{P}$. The set $\mathcal{POS}$ contains all the positive examples of the dataset for all the predicates:

$$\mathcal{POS} = \bigcup_{\phi \in \mathcal{P}} \phi^+.$$

Negative Facts This set of facts contains the negative examples necessary for learning the predicates. SII datasets usually contain only partial and positive information without negative labels for objects and relations. Indeed, the labels indicate only the content of a bounding box or the relations between them. However, the missing information in the dataset cannot be considered negative information as the *Close World Assumption* (CWA) [81] does. CWA states that every unknown fact (that is, a missing label on a bounding box or on an edge) is false and thus can be regarded as a negative example. This assumption is too strong because the missing information does not necessary imply falsity. Therefore, the negative examples are computed by adopting a relaxed version, called R-CWA, of the CWA. Regarding the negative examples for a semantic class C, R-CWA completes the annotations of the single bounding boxes according to the IsA constraints of a considered knowledge base. For example, the constraint $\forall x(\mathsf{Horse}(x) \rightarrow \mathsf{Animal}(x))$ leads to complete the annotation of the bounding boxes containing horses with the extra label "animal". Then, the negative examples are the atomic formulas $\neg C(b)$, for every bounding box b not labelled with the class $C \in \mathcal{P}_1$. The negative examples for a relation R are the atomic formulas $\neg R(b_1, b_2)$, for every pair of bounding boxes $\langle b_1, b_2 \rangle$ belonging to the same picture and not labelled with *any* binary relation. R-CWA avoids to consider, as negative examples for a relation R, bounding boxes annotated with another relation. Consider, for example, the case where b_1 and b_2 are annotated only with the ride relation: with the CWA the pair b_1, b_2 is a wrong negative example for the on relation whereas our relaxed version avoids this issue. We denote with ϕ^- the set containing all the negative examples (that is, the negated atomic formulas) for the predicate $\phi \in \mathcal{P}$. The set $\mathcal{NEG}$ contains all the negative examples of the dataset for all the predicates:

$$\mathcal{NEG} = \bigcup_{\phi \in \mathcal{P}} \phi^-.$$

Background Knowledge The above sources of information add to $\mathcal{K}_{\text{SII}}$ only assertional (factual) information about the labels of the (pairs of) bounding boxes in *Pics*. However, the object classes and the relations can be related through logical constraints. Indeed, a

relation can imply another one, for example, "a person rides a horse" implies "a person is on a horse", or "a person is on the left of a horse" implies "a person is next to a horse". Other examples of constraints state that some object classes cannot be the subject/object of some relations, for example "horses cannot ride" and "a person cannot be worn". These constraints express some background (or prior) knowledge $\mathcal{BK}$ about the domain of the pictures in our dataset. In LTNs, the background knowledge can be formalized as a set of $\mathcal{PL}$ formulas. Here follow some categories of logical constraints for SII with examples:

IsA these constraints encode containment between semantic classes or binary relations. For example: $\forall x(\mathsf{Horse}(x) \rightarrow \mathsf{Animal}(x))$ and $\forall xy(\mathsf{ride}(x,y) \rightarrow \mathsf{on}(x,y))$.

Mutual Exclusivity These constraints encode mutual exclusivity between classes or relations. For example: $\forall x(\mathsf{Animal}(x) \rightarrow \neg\mathsf{Car}(x))$ and $\forall xy(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{wear}(x,y))$.

Inverse These constraints regard the binary relations and state that one binary relation is the inverse of another one. That is, if R is the inverse relation of R' and if $R(b_1,b_2)$ holds then also $R'(b_2,b_1)$ holds. For example: $\forall xy(\mathsf{on}(x,y) \rightarrow \mathsf{under}(y,x))$ and $\forall xy(\mathsf{left\ of}(x,y) \rightarrow \mathsf{right\ of}(y,x))$.

Symmetry These constraints regard the binary relations and encode the symmetric (or antisymmetric) property of binary relations. That is, if the pair $\langle b_1,b_2\rangle$ is related with R then also the inverse pair $\langle b_2,b_1\rangle$ is (or not). For example: $\forall xy(\mathsf{near}(x,y) \rightarrow \mathsf{near}(y,x))$ and $\forall xy(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{ride}(y,x))$.

Reflexivity These constraints regard the binary relations and encode the reflexivity property of binary relations. That is, if a relation R always holds for a single bounding box or not: $R(b,b)$. For example: $\forall x(\mathsf{near}(x,x))$ and $\forall x(\neg\mathsf{wear}(x,x))$.

Domain and Range These constraints join classes and binary relations and encode the types of subjects (domain) and objects (range) that are involved in a binary relation. For example: $\forall xy(\mathsf{Person}(x) \wedge \mathsf{ride}(x,y) \rightarrow \mathsf{Horse}(y) \vee \mathsf{Motorcycle}(y) \vee \mathsf{Bicycle}(y))$ and $\forall xy(\mathsf{Pizza}(y) \wedge \mathsf{eat}(x,y) \rightarrow \mathsf{Person}(x))$.

Negative Domain and Range These constraints join semantic classes and binary relations and discard some classes for the domain/range of the binary relations. For example: $\forall xy(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x))$ and $\forall xy(\mathsf{drive}(x,y) \rightarrow \neg\mathsf{Pizza}(y))$.

These logical constraints can be manually defined according to the domain and the signature Σ_{SII} or retrieved by on-line linguistic resources such as WordNet [34], FrameNet [7] and VerbNet [90]. For example, the IsA constraints about classes can be retrieved

in WordNet, whereas all the mentioned resources provide the range and domain of the binary relations through the so-called *frames*. A frame is a data structure that describes the semantic roles (such as the domain and the range with their semantic classes) of each binary relation.

Therefore, our knowledge base $\mathcal{K}_{\text{SII}} = \mathcal{POS} \cup \mathcal{NEG} \cup \mathcal{BK}$, contains positive and negative examples and general prior knowledge about the SII domain. All this information is necessary for training a LTNs model and it is expressed with a uniform representation through LTNs formulas.

7.2 The Grounding $\hat{\mathcal{G}}_{\text{SII}}$

As stated in Section 6.3, the LTNs grounding is based on numeric features of the objects of the domain, that is, bounding boxes for SII. The bounding boxes, along with their features, can be provided by a SII dataset or extracted from images with the use of a trained object detector, such as Fast R-CNN [36]. The features can be divided into two groups: the *semantic features* and the *geometric features*. The first group of features represents the fact that a bounding box b could contain an object whose semantic class is $C \in \mathcal{P}_1$. This set of features is denoted with $\{score(C, b)\}_{C \in \mathcal{P}_1}$, where $score(C, b)$ is the classification score of the object detector for b according to the class $C \in \mathcal{P}_1$. The second group of features describes geometric properties of bounding boxes:

- $\langle x_0(b), y_0(b), x_1(b), y_1(b) \rangle$: these features are the coordinates of the bounding box b. The pair $\langle x_0(b), y_0(b) \rangle$ is the top-left corner of b whereas $\langle x_1(b), y_1(b) \rangle$ is the bottom-right corner of b.

- $area(b)$: this feature is the area of b.

The grounding for a pair of bounding boxes can be defined by concatenating the groundings of the single bounding boxes as the LTNs framework states, see Section 6.3. However, when dealing with n-tuples of objects, adding some extra features regarding the combination of these n objects improves the performance of the LTNs model. In the SII domain the extra features involve geometrical joint properties of both bounding boxes:

- $intersec(b_1, b_2)$: this feature is the area of the intersection between bounding boxes b_1, b_2.

- $euclid\_dist(b_1, b_2)$: this feature is the Euclidean distance between the centroids of bounding boxes b_1, b_2. The centroid of a bounding box is computed by using the bounding box coordinates.

- $sin(b_1, b_2)$: this feature is the sine of the angle between the centroid of b_1 and the centroid of b_2.
- $cos(b_1, b_2)$: this feature is the cosine of the angle between the centroid of b_1 and the centroid of b_2.

Except for the semantic features and the bounding box coordinates, the computation of all other functions is performed by using the bounding box coordinates. All the features are normalized in the interval $[-1, 1]$. In addition, sine and cosine are defined according to the θ angle between b_1 and b_2 computed in a counter-clockwise manner, see Figure 6.5. Regarding the **constants**, the grounding for a bounding box constant $b \in \mathcal{C}$ can be defined as a feature vector composed of semantic and geometric features. More formally, for each bounding box constant $b \in \mathcal{C}$, the grounding $\hat{\mathcal{G}}_{\text{SII}}(b) = \mathbf{v}_b \in \mathbb{R}^{|\mathcal{P}_1|+4}$ is:

$$\mathbf{v}_b = \langle score(C_1, b), \ldots, score(C_{|\mathcal{P}_1|}, b), x_0(b), y_0(b), x_1(b), y_1(b) \rangle \tag{7.1}$$

with the last four elements the coordinates of the top-left and bottom-right corners of b defined above. Another definition for the grounding of constants is its *one-hot encoding*: the semantic features take the value of 1 in the position of the class with the highest detection score. All other positions are zero and the geometric features remain unchanged:

$$\mathbf{v}_b^i = \begin{cases} 1 & \text{if } i = \text{argmax}_{1 \leq l \leq |\mathcal{P}_1|}\, score(C_l, b) \\ 0 & \text{otherwise} \end{cases} \tag{7.2}$$

with $\mathbf{v}_b^i$ the i-th entry in vector $\mathbf{v}_b$. The grounding for a pair of bounding boxes $\langle b_1, b_2 \rangle$ is the concatenation of the groundings of the single bounding boxes $\langle \mathbf{v}_{b_1} : \mathbf{v}_{b_2} \rangle$ with some extra features as discussed above:

$$\mathbf{v}_{b_1,b_2} = \langle \mathbf{v}_{b_1} : \mathbf{v}_{b_2}, ir(b_1, b_2), ir(b_2, b_1), \frac{area(b_1)}{area(b_2)}, \frac{area(b_2)}{area(b_1)}, euclid\_dist(b_1, b_2), sin(b_1, b_2), cos(b_1, b_2) \rangle \tag{7.3}$$

with $ir(b_1, b_2) = intersec(b_1, b_2)/area(b_1)$ is the inclusion ratio defined in Section 6.3.1. Regarding the **unary predicates** in $\mathcal{P}_1$, a simple grounding can be defined by adopting a one-vs-all multi-classifier approach. Given a bounding box constant b, along with its feature vector $\mathbf{v}_b = \langle v_1, \ldots, v_{|\mathcal{P}_1|+4} \rangle$, and a predicate symbol $C_i \in \mathcal{P}_1$, the grounding for C_i is:

$$\hat{\mathcal{G}}_{\text{SII}}(C_i)(\mathbf{v}_b) = \begin{cases} 1 & \text{if } i = \text{argmax}_{1 \leq l \leq |\mathcal{P}_1|}\, \mathbf{v}_b^l \\ 0 & \text{otherwise.} \end{cases} \tag{7.4}$$

A slightly different definition of the above grounding is to substitute the value 1 in Equation 7.4 with $score(C_i, b)$ if $i = \text{argmax}_{1 \leq l \leq |\mathcal{P}_1|} \mathbf{v}_b^l$ and left zero otherwise. This rule-based grounding can be a good solution if the performance of the object detection are good. Otherwise, it is possible to learn the groundings from bounding box examples according to Equation (6.11). Regarding the **binary predicates** in $\mathcal{P}_2$, a rule-based grounding can be difficult to define as it requires a different analysis for each relation (scalability issue) and it could be inaccurate. As explained in Section 6.3.3, some groundings should be learned from data by maximizing the truth values of the logic formulas in the grounded theory. However, due to the flexibility of LTNs, this does not prevent us from using both groundings: a rule-based for unary predicates and a learnt one for relations.

The grounding of the **logical constraints** in $\mathcal{BK}$ is determined by (i) instantiating the constraints in $\mathcal{BK}$ with all the bounding box constants $b \in \mathcal{C}$. Here, it is worth to notice that every constraint has to be instantiated with only bounding box constants belonging to the same picture. Constraints on bounding boxes on different images showing different scenes have no sense for building a semantically interpreted picture. (ii) Computing the groundings of the atomic formulas for every instantiated constraint; (iii) combining the groundings of the atomic formulas according to the LTNs semantics (that is, the chosen t-norm) for every instantiated constraints; (iv) aggregating the groundings of every instantiated constraint with a mean operator, according to the LTNs semantics for universal quantifiers, see Section 6.3. Here follows an example of grounding computation for the logical constraint $\psi = \forall xy(\mathsf{ride}(x, y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x))$, that is, horses and cats do not ride. Let $\mathcal{C} = \{b_1, b_2\}$ be the set of constants, (i) the instantiations of the formula ψ according to $\mathcal{C}$ is:

$$\begin{aligned}
&\mathsf{ride}(b_1, b_1) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_1)\\
&\mathsf{ride}(b_1, b_2) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_1)\\
&\mathsf{ride}(b_2, b_1) \rightarrow \neg\mathsf{Horse}(b_2) \wedge \neg\mathsf{Cat}(b_2)\\
&\mathsf{ride}(b_2, b_2) \rightarrow \neg\mathsf{Horse}(b_2) \wedge \neg\mathsf{Cat}(b_2)
\end{aligned}$$

Then, (ii) the computation of the groundings of the atomic formulas of each instantiation is performed:

	$\hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x, y))$	$\hat{\mathcal{G}}_{\text{SII}}(\mathsf{Horse}(x))$	$\hat{\mathcal{G}}_{\text{SII}}(\mathsf{Cat}(x))$
$\mathsf{ride}(b_1, b_1) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_1)$	0.16	0.11	0.29
$\mathsf{ride}(b_1, b_2) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_1)$	0.76	0.11	0.29
$\mathsf{ride}(b_2, b_1) \rightarrow \neg\mathsf{Horse}(b_2) \wedge \neg\mathsf{Cat}(b_2)$	0.13	0.85	0.17
$\mathsf{ride}(b_2, b_2) \rightarrow \neg\mathsf{Horse}(b_2) \wedge \neg\mathsf{Cat}(b_2)$	0.39	0.85	0.17

The next step (iii) is the aggregation of these results to obtain the grounding of the formula $\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x)$. This is done, according to the LTNs semantics, by considering a t-norm. If we take, for example, the Łukasiewicz t-norm[2], we have that $\hat{\mathcal{G}}_{\text{SII}}(\psi) = 1$ if $\hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x,y)) \leq \max(0, 1 - \hat{\mathcal{G}}_{\text{SII}}(\mathsf{Horse}(x)) + 1 - \hat{\mathcal{G}}_{\text{SII}}(\mathsf{Cat}(x)) - 1)$ (first line of the following table), and $\hat{\mathcal{G}}_{\text{SII}}(\psi) = 1 - \hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x,y)) + \max(0, 1 - \hat{\mathcal{G}}_{\text{SII}}(\mathsf{Horse}(x)) + 1 - \hat{\mathcal{G}}_{\text{SII}}(\mathsf{Cat}(x)) - 1)$ otherwise:

	$\hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x))$
$\mathsf{ride}(b_1,b_1) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_2)$	1
$\mathsf{ride}(b_1,b_2) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_2)$	$(1 - 0.76) + \max(0, 1 - 0.11 + 1 - 0.29 - 1) = 0.84$
$\mathsf{ride}(b_1,b_3) \rightarrow \neg\mathsf{Horse}(b_1) \wedge \neg\mathsf{Cat}(b_2)$	$(1 - 0.13) + \max(0, 1 - 0.85 + 1 - 0.17 - 1) = 0.87$
$\mathsf{ride}(b_2,b_2) \rightarrow \neg\mathsf{Horse}(b_2) \wedge \neg\mathsf{Cat}(b_2)$	$(1 - 0.39) + \max(0, 1 - 0.85 + 1 - 0.17 - 1) = 0.61$

Finally, (iv) the grounding of the universal quantifier is computed with a mean operator. According to Equation (6.7), if we use, for example, the harmonic mean as mean operator ($p = 1$) we obtain: $\hat{\mathcal{G}}_{\text{SII}}(\forall xy(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x))) = mean_p(\hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x)) | \langle x,y\rangle \in T)$ with $T = \{\langle b_1,b_1\rangle, \langle b_1,b_2\rangle, \langle b_2,b_1\rangle, \langle b_2,b_2\rangle\}$. Therefore,

$$mean_{p=1}(\hat{\mathcal{G}}_{\text{SII}}(\mathsf{ride}(x,y) \rightarrow \neg\mathsf{Horse}(x) \wedge \neg\mathsf{Cat}(x)) | \langle x,y\rangle \in T) = \left(\frac{1^{-1} + 0.84^{-1} + 0.87^{-1} + 0.61^{-1}}{4}\right)^{-1} = 0.80.$$

This simple example gives an intuition of the computation of the grounding for logical constraints. For more complex constraints, we compute the parse tree of every instantiated constraint, all the groundings of the atomic formulas are computed and the results are combined in a bottom-up fashion to return the grounding of the whole instantiated constraint. The results are then combined according to the grounding of the universal quantifier. The logical connectives are computed according to the chosen t-norm, residuum and s-norm. For further details see Section 6.3.4.

7.3 The Optimization of the Grounded Theory $\mathcal{T}_{\text{SII}}$

Once the grounded theory $\mathcal{T}_{\text{SII}}$ is defined, its satisfiability needs to be maximized according to Equation (6.13), in order to learn the parameters Θ^* of LTNs:

$$\Theta^* = \text{argmax}_\Theta \, \hat{\mathcal{G}}_{\text{SII}} \left(\bigwedge_{\phi \in \mathcal{K}_{\text{SII}}} \phi \middle| \, \Theta \right) - \lambda \|\Theta\|_2^2.$$

[2]We recall that, with this t-norm, $\hat{\mathcal{G}}_{\text{SII}}(\neg\phi) = 1 - \hat{\mathcal{G}}_{\text{SII}}(\phi)$, $\hat{\mathcal{G}}_{\text{SII}}(\phi \wedge \chi) = \max(0, \hat{\mathcal{G}}_{\text{SII}}(\phi) + \hat{\mathcal{G}}_{\text{SII}}(\chi) - 1)$ and $\hat{\mathcal{G}}_{\text{SII}}(\phi \rightarrow \chi) = 1 - \hat{\mathcal{G}}_{\text{SII}}(\phi) + \hat{\mathcal{G}}_{\text{SII}}(\chi)$ if $\hat{\mathcal{G}}_{\text{SII}}(\phi) > \hat{\mathcal{G}}_{\text{SII}}(\chi)$, 1 otherwise.

The above Equation maximizes the grounding of the conjunctions of the formulas in the knowledge base $\mathcal{K}_{\text{SII}}$. The grounding of the conjunctions is the t-norm of the single groundings. However, when working with optimization according to data, the use of the main t-norms could present some numeric pitfalls that need to be avoided:

Łukasiewicz t-norm According to this t-norm, the satisfiability of a knowledge base is given by the formula: $\hat{\mathcal{G}}_{\text{SII}}(\bigwedge_{\phi\in\mathcal{K}_{\text{SII}}}\phi) = \max\{0, \sum_{\phi\in\mathcal{K}_{\text{SII}}}\hat{\mathcal{G}}_{\text{SII}}(\phi) - |\mathcal{K}_{\text{SII}}| + 1\}$. Thus, the higher the number of formulas the higher their grounding should be to have a satisfiability value bigger than zero. However, as we are optimizing the groundings of the formulas, even a small number of formulas in $\mathcal{K}_{\text{SII}}$ with a low grounding value can lead the knowledge base satisfiability to zero. We call this issue *zero satisfiability* problem.

Gödel t-norm This t-norm considers the satisfiability of the whole knowledge base as the minimum of the groundings of all its formulas: $\hat{\mathcal{G}}_{\text{SII}}(\bigwedge_{\phi\in\mathcal{K}_{\text{SII}}}\phi) = \min\{\hat{\mathcal{G}}_{\text{SII}}(\phi)|\phi \in \mathcal{K}_{\text{SII}}\}$. The problem is that the optimization process could get stuck in a *local optimum*. Indeed, a single (or a group of) axiom could be too difficult to learn for LTNs, and thus the satisfiability of the knowledge base is the grounding of this difficult axiom. The optimizer tries to increase this value but without any improvement (this particular predicate is too difficult to learn) and thus leaving out the other predicates from the optimization.

Product t-norm This t-norm considers the satisfiability of the whole knowledge base as the product of the groundings of all its formulas: $\hat{\mathcal{G}}_{\text{SII}}(\bigwedge_{\phi\in\mathcal{K}_{\text{SII}}}\phi) = \prod_{\phi\in\mathcal{K}_{\text{SII}}}\hat{\mathcal{G}}_{\text{SII}}(\phi)$. As a knowledge base can have hundreds of formulas, the product of hundreds of groundings (values in $[0, 1]$) can result in a very small number and thus incurring in *underflow* problems.

To avoid these issues, we provide another definition of knowledge base satisfiability, more in the spirit of optimization with data. The knowledge base satisfiability is a mean of the groundings of the single formulas. This idea is similar to the grounding of the universal quantifier. Indeed, a mean operator returns a global satisfiability of the knowledge base without the side effect of a single predicate that determines the satisfiability. Moreover, it avoids the other numeric problems of the t-norms. Thus, we adopt the following optimization task:

$$\Theta^* = \text{argmax}_\Theta \, mean_p\left(\hat{\mathcal{G}}_{\text{SII}}(\phi|\Theta)|\phi \in \mathcal{K}_{\text{SII}}\right) - \lambda\|\Omega\|_2^2. \tag{7.5}$$

As stated in Chapter 6, some well-known means are obtained by setting up the parameter p with an integer number. For example, if we set up $p \to -\infty$ we obtain the minimum

of the groundings and thus the Gödel t-norm. If we set $p = -1$ or $p \to 0$, we obtain the harmonic and the geometric mean, respectively, These choices are reasonable for avoiding the mentioned t-norm issues. On the other hand, with a higher value for p we obtain means that are more influenced by the higher grounding values, that is, by the predicates "easy" to learn. These means return a too optimistic value of the satisfiability and this wrongly avoids the need of optimization.

The instantiation of the logical constraints with all the constant symbols in $\mathcal{C}$ (belonging to the same image) is not tractable due to the combinatorial explosion of the grounded constraints. Therefore, we perform a uniform random sampling of the grounded constraints such that every constraint has the same number of instantiations.

Another numeric issue related to t-norms regards the aggregation of groundings of the atomic formulas in the constraints according to the semantics of the logical connectives. In particular, we focus on the grounding of the negation $\hat{\mathcal{G}}_{\text{SII}}(\neg\phi)$ for the atomic formula ϕ. As LTNs have a functional semantics, the value $\hat{\mathcal{G}}_{\text{SII}}(\neg\phi)$ depends from the value of $\hat{\mathcal{G}}_{\text{SII}}(\phi)$ according to a selected t-norm. If we consider, for example, the Gödel and the Product t-norms we obtain the value of 0 for $\hat{\mathcal{G}}_{\text{SII}}(\neg\phi)$ in the majority of the cases, as it is rather unlikely to predict an atomic formula with grounding exactly 1, see the precomplement definition in Chapter 6. The Łukasiewicz t-norm, instead, avoids this problem by defining $\hat{\mathcal{G}}_{\text{SII}}(\neg\phi)$ as $1 - \hat{\mathcal{G}}_{\text{SII}}(\phi)$.

7.4 The Implementation of Logic Tensor Networks

LTNs have been implemented as a Google TensorFlow^{TM} library whose root folder is `LTN`. This library contains the source code and the resources (datasets and knowledge bases) necessary to implement the main functions of a Machine Learning system and its evaluation. The library can be divided into these software modules:

LTNs core This module of the library is the pure implementation of LTNs and it is totally independent from the domain, the dataset and the evaluation. It contains the single file `LTN/logictensornetworks.py` that takes as input a LTNs knowledge base $\mathcal{K}_{\text{SII}}$ and builds the relative tensor network as shown in Section 6.3.4. This is performed by defining the computational graph of the groundings for the predicates, the functions, the logical constraints and the optimization of the knowledge base satisfiability, see Equation (7.5). The computational graph is defined by using the TensorFlow^{TM} methods. This file is the starting point for applying LTNs to other domains with different datasets and tasks.

Dataset and knowledge base parsing A SII dataset and knowledge base need to be parsed in order to be converted into a LTNs knowledge base $\mathcal{K}_{\text{SII}}$. The dataset is split in the folders `LTN/data/train` and `LTN/data/test`, the first one is used for training and the second one for the evaluation. The knowledge base is stored in the folder `LTN/data/ontology` into several files according to the semantic classes, the relationships and the types of logical constraints. Both dataset and knowledge base are stored in `csv` format. After the parsing, the feature of the (pairs of) bounding boxes are extracted for computing the grounding of constant symbols, see Section 7.2. Parsing and features extraction are implemented in a single Python script, usually named with the name of the dataset.

Training After the process of parsing the dataset/knowledge base and the extraction of features, the LTNs knowledge base $\mathcal{K}_{\text{SII}}$ (containing LTNs formulas for examples and constraints) is instantiated and the training process starts in order to compute $\hat{\mathcal{G}}_{\text{SII}}$. A training step is performed for a maximum number of epochs in order to minimize the loss function. The data (examples and instantiations of the constraints) are passed in batches and randomly shuffled according to a fixed number of iterations. This is implemented in a Python script called `LTN/train.py`. In addition, in this file the hyperparameters of $\mathcal{K}_{\text{SII}}$, such as the means for universal quantification and loss function (Equation (7.5)), the chosen t-norm, the hyperparameters for the training, such as number of training epochs, type of loss optimizer, smoothing factor λ, the number k of tensor layers, are set. Once the training is finished the parameters of the grounded theory $\mathcal{T}_{\text{SII}}$ are saved in a folder called `LTN/models`.

Testing The LTNs grounded theory is evaluated according to some tasks (for example the detection of relations between bounding boxes) on the test set, see next Chapter. This is implemented in the Python script `LTN/evaluate.py` and the results are saved in the `LTN/results` folder. In some cases, the test set provides also some code for the evaluation, thus the script `LTN/evaluate.py` provides an interface from LTNs to the test set code.

Chapter 8

Experimental Results with Logic Tensor Networks

The previous chapters show all the theoretical framework for LTNs and how SII has been encoded with a LTNs grounded theory $\mathcal{T}_{\text{SII}}$. This Chapter describes a set of experiments that allow us to evaluate LTNs on the following aspects of a semantically interpreted picture construction.

Performance on the SII Problem The construction of a graph that describes the content of a picture implies two main tasks. First, the detection and the classification of bounding boxes with some labels (that is, the unary predicates) that describe objects, and then the classification of visual relationships (that is, the binary predicates) between two bounding boxes. Notice that these tasks are the completion of the partial information in a SII dataset. These tasks are tested with two LTNs models: a first one trained only with positive and negative examples of the unary and binary predicates. This model allows us to evaluate the effectiveness of LTNs in a classical Machine Learning setting. A second one trained with examples and logical constraints of a background knowledge. This is an important feature of LTNs as the semantic classes of the bounding boxes are not independent from the relations between them. This connection is expressed with logical constraints of a background knowledge. Therefore, it is important to test if such a background knowledge improves the performance of a classical Machine Learning setting, see our works [92, 28].

Robustness to Noisy Labels The present and the following tasks are more related with issues of the datasets for training Machine Learning models than to strictly

SII tasks. However, solving this issues affecting the datasets makes a trained SII system more robust. Indeed, as training sets for Neural Networks become larger, some problems related to human annotation become acute [80]. Two main problems are (i) the quality of the labels (subjective labelling) and (ii) the completeness of the dataset, that is, the quantity of missing label. The first task regards the quality of the labels. Indeed, annotators may not agree on the semantic class label for a bounding box or on the visual relationship for a pair of bounding boxes. Therefore, it is possible to have a consistent amount of noisy labels and it is very important to have a SII system robust to this noise. This experiment aims at testing the robustness of LTNs to noisy training labels for both semantic classes and visual relationships according to the SII tasks defined above.

Zero-Shot Learning This experiment regards the so-called *zero-shot learning* [99] and tests the ability of LTNs to deal with unseen visual relationships according to the SII tasks defined above. This experiment concerns the second mentioned problem affecting datasets: the missing labels in a dataset. Indeed, it is quite hard for annotators to label every instance in a dataset. For example, in a SII dataset, some semantic classes can have no training examples (bounding boxes), thus, they cannot be detected. Another SII example regards the visual relationships: some instances of relationships (for example, the relation stand on between two bounding boxes labelled with Elephant and Street, respectively) can have no annotations. However, never seen instances can be predicted by *similarity* with already seen examples. For instance, it is possible to deduce that elephants stand on streets by a certain similarity between elephants and horses (supposed that annotations between horses and streets are given).

The experiments for these tasks are conducted on two SII datasets: the PASCAL-Part-dataset [17] and the Visual Relationship Dataset (VRD) [65]. Each dataset is used to test different tasks. We specify the tasks for each dataset in Table 8.1.

Dataset	**Performance**	**Robustness**	**Zero-Shot Learning**
PASCAL-Part-dataset	Section 8.1.2	Section 8.1.3	
Visual Relationship Dataset	Section 8.2.3		Section 8.2.4

Table 8.1: The SII tasks for every considered dataset.

This chapter follows with two sections, each one for a dataset. Section 8.1 presents the results obtained on the PASCAL-Part-dataset whereas Section 8.2 introduces the

VRD dataset along with the results obtained on it.

8.1 Objects Classification and Part-of Detection

The first experiment is conducted on the PASCAL-PART-dataset [17] presented in Section 5.4.1. The performance of LTNs are tested on the following tasks[1]

Object Classification Given a set of bounding boxes, provided by the PASCAL-PART-dataset, the task is to assign each bounding box with an object type. The set $\mathcal{P}_1 \in \Sigma_{SII}$ contains the 60 semantic classes of the PASCAL-PART-dataset.

Part-of Detection Given a set of pairs of bounding boxes (in this case all the pairs belonging to the same picture), provided by the PASCAL-PART-dataset, the task is to predict if they are in part-of relation or not. That is, if the object contained in the first bounding box is part of the object contained in the second one. In this case, $\mathcal{P}_2 \in \Sigma_{SII}$ contains only the partOf predicate. This is a particular instance of the visual relationship detection task where the prediction is performed only on a single binary predicate. With multiple relations the general task is described in Section 8.2 and Figure 8.3.

Notice that these two tasks are not independent due to the constraints that join the whole objects with their parts. In addition, these tasks do not require the use of an object detector for finding bounding boxes as they are totally provided by the dataset. Therefore, the predictions do not depend on the object detector performance. In this manner, it is possible to focus only on the ability of LTNs to predict unary and binary predicates. The performance of LTNs on both tasks have been evaluated with two LTNs models: a first one trained with only positive and negative examples ($\mathcal{T}_{\text{expl}}$) and a second one trained with examples and constraints ($\mathcal{T}_{\text{prior}}$). This evaluation has been performed by splitting the PASCAL-PART-dataset in two parts: the training set for training the LTNs models and the test set for performing the evaluation, as in a classical Machine Learning setting. Details regarding this splitting are reported in Section 5.4.1.

[1]The code for the experiments with the PASCAL-PART-dataset can be downloaded here `https://gitlab.fbk.eu/donadello/LTN_ACM_SAC17/` [92] and here `https://gitlab.fbk.eu/donadello/LTN_IJCAI17` [28].

8.1.1 The PASCAL-Part Background Knowledge

In this evaluation, the logical background knowledge $\mathcal{BK}$ adopted here is a FOL extended version of the one in Section 5.4.2 used for testing the unsupervised method. This new background knowledge is not a simple meronomy of the semantic classes. Indeed, it also contains constraints about the mutual exclusivity between classes, properties of the partOf relation, domain and range constraints. Here follow some examples of logical constraints in $\mathcal{BK}$ grouped by categories:

Mutual Exclusivity These constraints encode the mutual exclusivity between all the semantic classes of the PASCAL-PART-dataset. Some examples are: $\forall x(\mathsf{Bicycle}(x) \rightarrow \neg\mathsf{Motorcycle}(x))$ and $\forall x(\mathsf{Saddle}(x) \rightarrow \neg\mathsf{Wheel}(x))$.

Anti-symmetry This constraint is the antisymmetric property of the partOf relation: $\forall xy(\mathsf{partOf}(x,y) \rightarrow \neg\mathsf{partOf}(y,x))$.

Anti-reflexivity This constraint is the anti-reflexivity property of partOf relation[2]: $\forall x(\neg\mathsf{partOf}(x,x))$.

Domain and Range These constraints include the meronomy of the semantic classes of the dataset (the background knowledge of Section 5.4.2). For example, this constraint states the parts of a motorcycle: $\forall xy(\mathsf{Motorcycle}(y) \wedge \mathsf{partOf}(x,y) \rightarrow \mathsf{Handlebar}(x) \vee \mathsf{Headlight}(x) \vee \mathsf{Saddle}(x) \vee \mathsf{Wheel}(x))$. In addition, for every semantic class in $\mathcal{P}_1$ denoting a part object, these constraints list the classes of whole objects containing the given part. For example, this constraint states the whole objects having a saddle: $\forall xy(\mathsf{Saddle}(x) \wedge \mathsf{partOf}(x,y) \rightarrow \mathsf{Motorcycle}(y) \vee \mathsf{Bicycle}(y))$.

The combination of mutual exclusivity with domain and range constraints makes $\mathcal{BK}$ "closed" under the partOf relation. Indeed, for every whole object, the background knowledge states *all and only* its part objects. On the other hand, for every part object, the background knowledge states *all and only* its whole objects.

8.1.2 Performance With and Without Constraints

In order to test the performance of LTNs on the above-mentioned SII tasks, we define a grounded theory $\mathcal{T}_{\text{expl}} = \langle \mathcal{K}_{\text{expl}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$[3], where $\mathcal{K}_{\text{expl}} = \mathcal{POS} \cup \mathcal{NEG}$ contains only positive and negative examples of the predicates in $\mathcal{P}$ for optimizing $\mathcal{T}_{\text{expl}}$, as the classical

[2]We adopt a restricted version of the part-whole relation: nothing is part of itself.

[3]For the sake of presentation, we remove the subscript SII from $\mathcal{T}_{\text{SII}}$ and $\mathcal{K}_{\text{SII}}$.

Machine Learning setting requires. This first theory allows us to check the effectiveness of LTNs with respect to other methods. Moreover, we define another grounded theory $\mathcal{T}_{\text{prior}} = \langle \mathcal{K}_{\text{prior}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$, where $\mathcal{K}_{\text{prior}} = \mathcal{POS} \cup \mathcal{NEG} \cup \mathcal{BK}$ contains examples and the logical constraints in $\mathcal{BK}$ defined in the previous section. This second theory allows us to check if the background knowledge has effect on the optimization and improves the results.

Both grounded theories have the same definition of grounding $\hat{\mathcal{G}}_{\text{SII}}$. In this case, the grounding is learnt from examples (and constraints) without a rule-based definition. Regarding the constant symbols, the grounding is the vector of semantic and geometric features defined in Equation 7.1. Regarding the unary and the partOf predicate, the grounding is learnt according to Equation (6.11). For the partOf predicate, the grounding $\hat{\mathcal{G}}_{\text{SII}}(b, b')$ of a pair of bounding boxes, for the computation $\hat{\mathcal{G}}_{\text{SII}}(\mathsf{partOf}(b, b'))$, is defined with Equation 7.3 but with only two extra features: the inclusion ratios $ir(b, b')$, $ir(b', b)$, see Section 6.3.1. The mean function used both for the grounding of the universal quantifier (Equation (6.7)) and for the grounding of the whole knowledge base (Equation (7.5)) is the harmonic mean. The chosen t-norm is the Łukasiewicz's: $T(x, y) = \max(0, x + y - 1)$. The number of tensor layers in Equation (6.11) is $k = 6$ and the regularization parameter in Equation (7.5) is $\lambda = 10^{-10}$. Even though both grounded theories have the same grounding, the optimization processes are different as they are performed on two different knowledge bases: $\mathcal{K}_{\text{expl}}$ and $\mathcal{K}_{\text{prior}}$. We run 1000 training epochs of the RMSProp learning algorithm available in TensorFlow^{TM}.

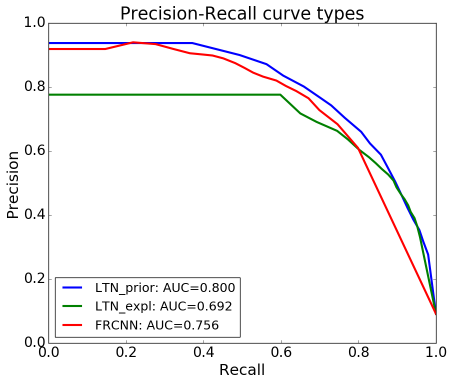
Once the grounded theories $\mathcal{T}_{\text{expl}}, \mathcal{T}_{\text{prior}}$ are trained, we evaluate them on the mentioned tasks with the following procedure. Every bounding box b, in the test set, is classified with $C \in \mathcal{P}_1$ if $\hat{\mathcal{G}}_{\text{SII}}(C(b)) \geq th$ for a threshold th that varies in the interval $[0, 1]$. In this manner, a bounding box can be classified with more than one class. For each class, precision and recall are calculated in the usual way. Every pair of bounding boxes $\langle b, b' \rangle$, belonging to the same picture of the test set, is classified with partOf if $\hat{\mathcal{G}}_{\text{SII}}(\mathsf{partOf}(b, b')) \geq th$, with $th \in [0, 1]$. Precision and recall are computed as in a usual binary classification.

The trained LTNs models are compared with a well-known object detector and a rule-based method according to the SII tasks[4]:

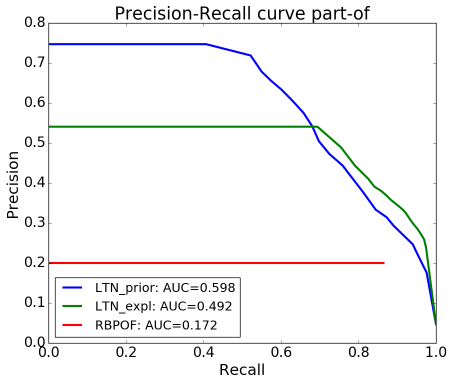
Object Types Classification A bounding box b is classified with the label C^* obtaining the highest score provided by the Fast R-CNN object detector (FRCNN) [36], that is, $C^* = \text{argmax}_{C \in \mathcal{P}_1} score(C, b)$, with $score(C, b) \geq th$, for $th \in [0, 1]$ and $C \in \mathcal{P}_1$.

Part-of Detection A pair of bounding boxes $\langle b, b' \rangle$ is classified with the partOf relation if the inclusion ratio $ir(b, b')$ is greater than a given threshold th (in this experiments

[4]A direct comparison with the SII system in [15] is not possible because the source code is not available.



(a) LTNs with prior knowledge improve the performance of Fast R-CNN on object type classification, achieving an Area Under the Curve (AUC) of 0.800 in comparison with 0.756.



(b) LTNs with prior knowledge outperform the rule-based approach in the detection of part-of relations, achieving AUC of 0.598 in comparison with 0.172.

Figure 8.1: Precision-Recall curves for indoor objects type classification and the partOf relation between bounding boxes.

$th = 0.7$). This baseline is similar to the one used for the grounding of the partOf relation, see Equation (6.8).

Results for indoor objects are shown in Figure 8.1 where AUC is the area under the precision-recall curve. The results show that, for both object types and the part-of relation, the LTNs trained with examples and background knowledge have better performance than the LTNs trained with only examples. Moreover, background knowledge allows LTNs to improve the performance of the Fast R-CNN object detector. Notice that the LTNs are trained using the Fast R-CNN results as features. Regarding the LTNs model $\mathcal{T}_{\text{expl}}$ without constraints, we can see that it outperforms only the rule-based method for the part-of detection. A possible explanation is that FRCNN assigns a bounding box to a class if the value of the corresponding semantic feature exceeds th. This is local to the specific semantic features. If such local features are very discriminative (which is the case in our experiments), then very good levels of precision can be achieved. Differently from FRCNN, the LTNs model $\mathcal{T}_{\text{expl}}$ makes a global choice which takes into consideration all (semantic and geometric) features together. This should offer robustness to the LTNs classifier at the price of a drop in precision. However, this drop is compensated by the logical constraints in $\mathcal{T}_{\text{prior}}$. For the other object types (animals and vehicles), LTNs have results comparable to FRCNN: FRCNN beats $\mathcal{T}_{\text{prior}}$ by 0.05 and 0.037 AUC, respectively,

for animals and vehicles. Finally, we perform an initial experiment on *small data*, on the assumption that the LTNs constraints should be able to compensate a reduction in training data. By removing the 50% of the training data for indoor objects, a similar performance to $\mathcal{T}_{\text{prior}}$ with the full training set can be achieved: 0.767 AUC for object types and 0.623 AUC for the part-of relation, that is, a little improvement in performance.

8.1.3 Robustness to Noisy Labels

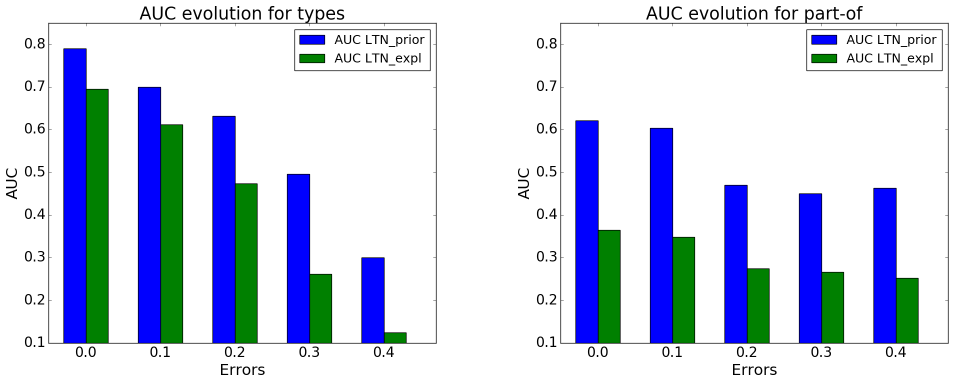
It has been acknowledged that, with the impressive growth of the size of training sets for visual recognition [55], many data annotations may be affected by noise, such as missing or erroneous labels, non-localised objects, and disagreements between annotations [80]. For example, human annotators could mistake the "part-of" relation with the "have" relation. Indeed, the label "have" could be used for both annotating possession ("person have umbrella") and part-of relationship ("person have leg"). In this experiment, we evaluate the robustness of LTNs models with respect to the presence of errors in the labels of the training data.

We artificially add an increasing amount of noise to the PASCAL-Part training data, and then the measures on how performance degrade in presence of noise are taken. For an error rate er in $\{0, 10, 20, 30, 40\}$, we randomly select the $er\%$ of the bounding boxes in the training data, and we randomly change their classification labels. In addition, we randomly select the $er\%$ of pairs of bounding boxes, and we flip the value of the part-of relation label. Notice that, the case for $er = 0$ is the case without errors described in the previous section. Then, two sets of LTNs grounded theories, $\{\mathcal{T}^{er}_{\text{expl}}\}^{40}_{er=0}$ and $\{\mathcal{T}^{er}_{\text{prior}}\}^{40}_{er=0}$, have been trained according to the values of er and evaluated on both SII tasks, as done before. As expected, adding too much noise to training labels leads to a large drop in performance. Figure 8.2 shows the AUC measures for indoor objects with increasing error er. Each pair of bars indicates the AUC of $\mathcal{T}^{er}_{\text{prior}}, \mathcal{T}^{er}_{\text{expl}}$ for a given $er\%$ of errors.

Results indicate that the LTNs constraints offer robustness to noise: in addition to the expected overall drop in performance, an increasing gap can be seen between the drop in performance of the LTNs model trained with only examples and the LTNs model trained with examples and background knowledge.

8.2 Multiple Relationships Detection

In the previous section, the experiments were performed according to only one single binary predicate. However, in a real visual scene, there are many binary predicates



(a) Object types (b) Part-of predicate

Figure 8.2: AUCs for indoor object types and part-of relation with increasing noise in the labels of the training data. The drop in performance is noticeably smaller for the LTNs model trained with examples and background knowledge.

between bounding boxes. In this section, the experiments with LTNs are extended to the more challenging task of detecting multiple binary predicates between bounding boxes. This task is called *visual relationship detection* [65]. In SII literature, a binary relation between two labelled bounding boxes b and b' is called *visual relationship*. Formally, it is defined as a triple $\langle subject, predicate, object \rangle$ where the subject refers to the label (or the unary predicate in $\mathcal{P}_1$) of b, the object refers to the label (or the unary predicate in $\mathcal{P}_1$) of b' and the predicate refers to the label of the relationship (or the binary predicate in $\mathcal{P}_2$) between b and b'. A visual relationship is also visually grounded, that is, the subject and the label have a corresponding bounding box in the image. In LTNs a visual relationship is expressed with the atomic formula: $subject(b) \wedge predicate(b, b') \wedge object(b')$. Examples of visual relationships in LTNs are: $\mathsf{Person}(b_1) \wedge \mathsf{ride}(b_1, b_2) \wedge \mathsf{Motorcycle}(b_2)$, $\mathsf{Person}(b_1) \wedge \mathsf{on}(b_1, b_2) \wedge \mathsf{Motorcycle}(b_2)$ and $\mathsf{Cat}(b_3) \wedge \mathsf{under}(b_3, b_4) \wedge \mathsf{Table}(b_4)$. In SII visual relationships are very important as they are the building blocks of a semantically interpreted picture. The construction of a SII graph with labelled and direct edges from visual relationships is pretty forward. For each visual relationship $subject(b) \wedge predicate(b, b') \wedge object(b')$, if the graph nodes for b and b' do not exist, then create them and add the labels *subject* and *object*. If they exist, add the labels *subject* and *object* to the nodes. If the direct edge from the node corresponding to b to the node corresponding to b' does not exist, then create it and add the label *predicate*. If such an edge exists, then add the label *predicate* to it. These experiments are conducted on the Visual Relationship Dataset (VRD) [65] developed for the very same task, see next section. The performance of LTNs are tested

on these standard tasks for the detection of visual relationships[5] (see Figure 8.3):

Phrase Detection Given an image, the task is to predict a correct triple $\langle subject, predicate, object \rangle$ and localize it in a single bounding box containing both the subject and the object. The triple is a true positive if the labels are the same of the ground truth triple and if the predicted bounding box has at least 50% of overlap with a corresponding bounding box in the ground truth. The ground truth bounding box is the union of the ground truth bounding boxes of the subject and of the object.

Relationship Detection Given an image, the task is to predict a correct triple $\langle subject, predicate, object \rangle$ and the bounding boxes containing the subject and the object of the relationship. The triple is a true positive if both bounding boxes have at least 50% of overlap with the corresponding ones in the ground truth. In addition, the labels for the predicted triple have to match with the ones of the corresponding triple in the ground truth.

Predicate Detection Given an image with a set of pairs of bounding boxes, the task is to predict a set of correct binary predicates between them. As the pairs of bounding boxes are given, this prediction does not depend on the performance of an object detector. Thus, the focus is only on the ability of LTNs to predict binary predicates.

As in the previous section, the performance of LTNs on these tasks have been evaluated with two LTNs models: a first one trained only with positive and negative examples ($\mathcal{T}_{\text{expl}}$) and a second one trained with examples and constraints ($\mathcal{T}_{\text{prior}}$) to check the effect of the logical constraints. This evaluation has been performed by training the LTNs models on the VRD training set and then by evaluating them on the VRD test set. Both LTNs models are then compared with the original work on VRD [65] and with the state-of-the-art [6] on all the mentioned tasks. Moreover, all three tasks have been tested in a zero-shot learning scenario in order to test the ability of the LTNs models to generalize to relationships never seen (and learnt) during the training phase, see Section 8.2.4 for further details.

[5]In these experiments, we implement the LTNs starting from the file `LTN/logictensornetworks.py`. The files and resources for parsing the VRD and its knowledge base, for defining the grounded theories and for training and evaluation have been changed accordingly to the VRD and its tasks, see Section 7.4.

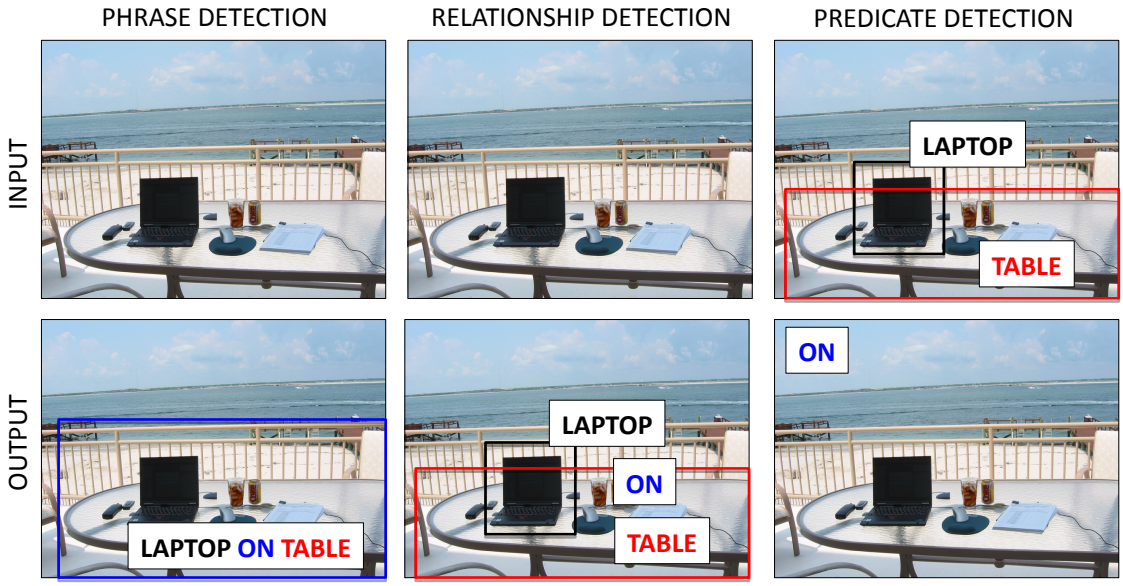


Figure 8.3: The three standard tasks of visual relationship detection.

8.2.1 The Visual Relationship Dataset

This dataset has been developed for the specific tasks of visual relationship detection. It contains 5000 images (4000 for training and 1000 for testing) annotated with visual relationships (bounding boxes and labels). Each bounding box is annotated with a label in the set $\mathcal{P}_1$ containing 100 unary predicates. These predicates include labels for animals (for

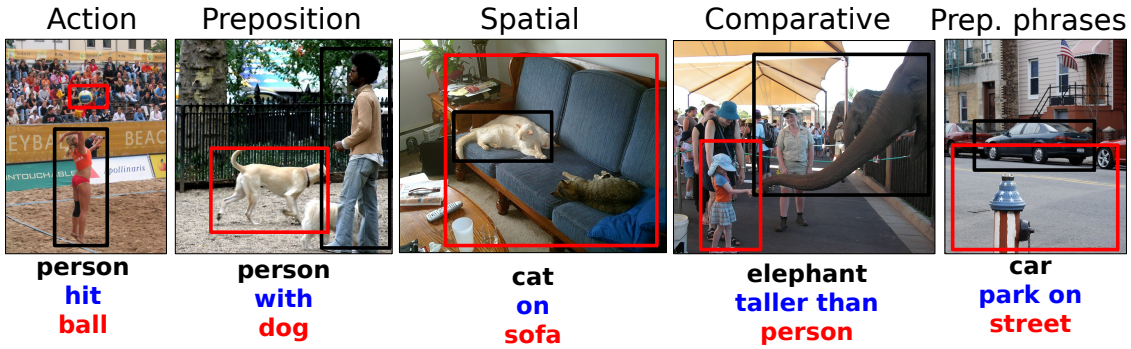


Figure 8.4: The binary predicates of the Visual Relationship Dataset can be grouped in categories (here shown 5 examples of them).

example, Horse and Elephant), vehicles (for example, Car and Bus), clothes (for example, Shirt and Jacket) and generic objects (for example, Camera and Cone). In addition, pairs of bounding boxes are annotated with a label in the set $\mathcal{P}_2$ containing 70 binary predicates.

These predicates include labels for actions (for example, ride and hit), prepositions (for example, with and at), spatial relations (for example, on and below), comparatives (for example, taller than) or preposition phrases (for example, park on and sit behind), see Figure 8.4. The dataset has 37993 instances of visual relationships, that is, every image has in average 7.60 triples that describe the content of the image. A visual relationship may appear several times in the images. The dataset has 6672 types of relationships. In addition, 1877 relationships occur only in the test set and they are used to evaluate the zero-shot tasks. On average, a unary predicate is connected with 24.25 binary predicates through visual relationships. Table 8.2 resumes these statistics.

VRD Statistics			
Images	5000	Relationship Types	6672
Images Train Set	4000	Relationship Instances	37993
Images Test Set	1000	Relationships Only in the Test Set	1877
Unary Predicates	100	Avg. Triples per Image	7.60
Binary Predicates	70	Avg. Predicates per Obj. Category	24.25

Table 8.2: Main figures for the Visual Relationship Dataset.

The dataset presents some issues that make the learning challenging. The first one is the so-called *long tail phenomenon* [62]: many binary predicates are involved in only few visual relationships, see Figure 8.5. Indeed, the dataset respects the 80-20 rule: the 80%

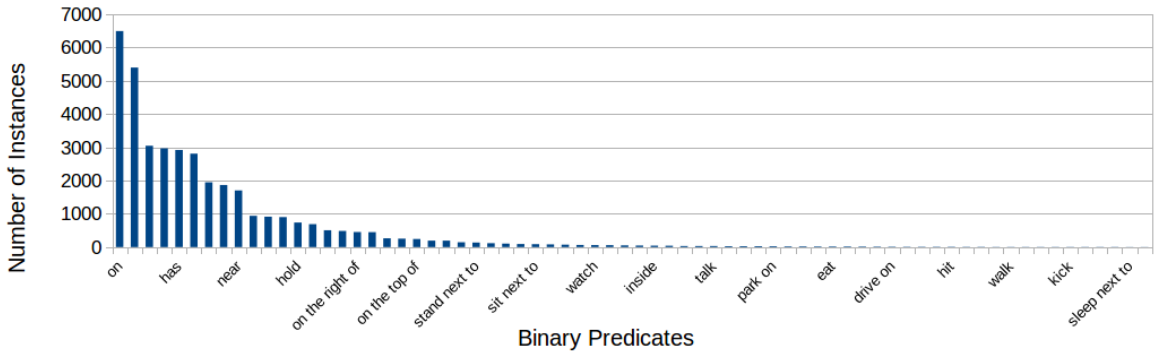


Figure 8.5: Long tail phenomenon: many predicates have only few relationships as examples. The figure shows the number of relationship instances for every predicate in the Visual Relationship Dataset.

of the triples is covered by only the 16% of the binary predicates (11 predicates such as

on, wear and has). This makes the learning of the single binary predicates very difficult. Another issue is that a single pair of bounding boxes is involved in only one single visual relationship (on average 1.17). This is quite unlikely in a real visual scene where a pair of bounding boxes occur in more relationships, for example, if $\langle \mathsf{Person}, \mathsf{ride}, \mathsf{Horse} \rangle$ then it also holds that $\langle \mathsf{Person}, \mathsf{on}, \mathsf{Horse} \rangle$.

8.2.2 The Visual Relationship Background Knowledge

In this evaluation, we manually build the background knowledge $\mathcal{BK}$ with logical constraints related to the unary and the binary predicates of the VRD. In the specific, we focus on the constraints that join binary and unary predicates. The reason is that a visual relationship is not independent from its subjects and objects. For example, there is a strong dependence between the sleep on relationship and its subjects that are only animals. Another strong dependence is between the drive relationship and its objects that are only vehicles. For this reason, the background knowledge contains the following category of constraints:

Negative Domain and Range These constraints do not list all the possible subjects and objects for a predicate but, rather, they exclude some semantic classes as subjects and objects. For example, physical objects cannot sleep on. Therefore, for every unary predicate PhyObj in $\mathcal{P}_1$ that refers to a physical object, the constraint $\forall xy(\mathsf{sleep\ on}(x,y) \rightarrow \neg \mathsf{PhyObj}(x))$ is added to $\mathcal{BK}$. Another example is that clothes cannot be driven. For every unary predicate Dress in $\mathcal{P}_1$ that refers to a dress, the constraint $\forall xy(\mathsf{drive}(x,y) \rightarrow \neg \mathsf{Dress}(y))$ is added to $\mathcal{BK}$.

However, more complex constraints on the binary predicates can be added to $\mathcal{BK}$, such as the IsA, the mutual exclusivity, the inverse, the symmetry and reflexivity properties, the positive domain and range constraint. Or even, a subject/object can be added in the negative domain/range constraints in $\mathcal{BK}$, for example, a person eats only edible things. However, adding too much knowledge could overfit the optimization with poor generalization results. For this reason, we add only the simple negative domain/range constraints. As future work, we study how performance change by introducing in $\mathcal{BK}$ other categories of constraints.

8.2.3 Performance With and Without Constraints

As done for the experiments on the PASCAL-Part-dataset, in order to evaluating the LTNs performance on phrase, relationship and predicate detection we define a grounded

theory $\mathcal{T}_{\text{expl}} = \langle \mathcal{K}_{\text{expl}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$, where the knowledge base $\mathcal{K}_{\text{expl}} = \mathcal{POS} \cup \mathcal{NEG}$ contains only positive and negative examples of the predicates in $\mathcal{P}$. This theory gives us the first results on the effectiveness of LTNs on visual relationship detection with respect to the state-of-the-art. Moreover, a second grounded theory $\mathcal{T}_{\text{prior}} = \langle \mathcal{K}_{\text{prior}}, \hat{\mathcal{G}}_{\text{SII}} \rangle$ is defined with its knowledge base $\mathcal{K}_{\text{prior}} = \mathcal{POS} \cup \mathcal{NEG} \cup \mathcal{BK}$ containing examples and the logical constraints in $\mathcal{BK}$. As before, with this theory we check the contribution of the logical constraints with respect to a standard Machine Learning approach.

Both grounded theories have the same definition for the grounding $\hat{\mathcal{G}}_{\text{SII}}$. Differently from the previous experiments, here the grounding is a mixture of the rule-based and learnt from examples approaches. Indeed, the grounding of the unary predicates is the result given by an object detector according to Equation (7.4). On the other hand, the grounding of the binary predicates is learnt from examples (and constraints) according to Equation (6.11). This choice is due to the focus of the experiments, our aim is to perform the detection of many relationships between bounding boxes instead of classifying bounding boxes. This last goal is left to the object detector. Regarding the constant symbols, the grounding, defined in Equation 7.2, is the vector that combines the one-hot encoding of the semantic features with the geometric features. The grounding of pairs of bounding boxes is the concatenation of the groundings of the single bounding boxes with the extra features defined in Equation (7.3). The mean function used for both the grounding of the universal quantifier (Equation (6.7)) and for the grounding of the whole knowledge base (Equation (7.5)) is the harmonic mean. The chosen t-norm is the Łukasiewicz one. The number of tensor layers in Equation (6.11) is $k = 5$ and the regularization parameter in Equation (7.5) is $\lambda = 10^{-10}$. Even though both grounded theories have the same grounding, the optimization processes are performed on the different knowledge bases $\mathcal{K}_{\text{prior}}$ and $\mathcal{K}_{\text{expl}}$. We run 20000 training epochs of the RMSProp optimizer in TensorFlow^{TM}.

The LTNs models $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$ are compared with the methods in [65] and in [6] on all the tasks:

Lu et al. [65] This method detects visual relationships by combining visual and semantic information. The visual information is the classification score given by two convolutional neural networks (VGG net [95]). The first network classifies single bounding boxes according to the semantic classes in $\mathcal{P}_1$. The second one classifies the union of two bounding boxes (subject and object) according to the relations in $\mathcal{P}_2$. These scores are then combined with a language prior score for modelling the semantics underlying the visual relationships. This prior is based on word embeddings. This combination allows the generalization from few (or zero) examples.

Baier et al. [6] This method also combines visual and semantic information. However, the work improves the results of the previous one by using link prediction methods (*RESCAL*, *MultiwayNN*, *CompleEx*, *DistMult* [6, 72]) in place of word embeddings for modelling the visual relationship semantics.

At the time of the experiments, these were the results of the state-of-the-art. Successfully, the method in [109] improves the results above. Here, a visual relationship is predicted with a network trained on visual features and on the word embeddings of the labels for the subject and the object. This network is regularized with a semantic term that encodes knowledge about the statistical dependencies between the relationships and the subjects/objects. We do not compare LTNs results with the results in [109] as they are successive to our experiments. Moreover, a full comparison is not possible as in [109] a different object detector, with respect to the one used in LTNs, is adopted for discovering the bounding boxes. Therefore, we limit at reporting the results in [109] and comparing the work with LTNs in Chapter 9.

Once the grounded theories $\mathcal{T}_{\text{expl}}, \mathcal{T}_{\text{prior}}$ are trained, we evaluate them on the mentioned tasks by using the test set of the VRD. For each image in the test set, we use $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$ to compute the ranked set of groundings $\{\hat{\mathcal{G}}_{\text{SII}}(\mathsf{r}(b,b'))\}_{\mathsf{r}\in\mathcal{P}_2}$, with $\langle b, b'\rangle$ bounding boxes computed with an object detector (phrase and relationship detection tasks) or taken from the image ground truth (predicate detection task). As object detector, we use the one (R-CNN) provided by [65][6]. Then, we select the top 100 and 50 relationships from the ranked set and compute the **recall@100** and **recall@50** [65, 1] as evaluation metrics. The choice of classifying the input pair $\langle b, b'\rangle$ of bounding boxes with *all* the predicates in $\mathcal{P}_2$ is due to the fact that many relationships can occur between two objects. For example, a person rides and is on a horse at the same time, or a person could be on the right of, beside and near to a motorcycle simultaneously. However, it is not possible to define a preference notion between relationships, for example, if two objects are one beside the other than they are automatically in the near relation. Other works [65] do not consider this fact and take the relation with highest score on $\langle b, b'\rangle$. This approach assumes that *all* the relations in $\mathcal{P}_2$ are mutually exclusives. Our choice is counterbalanced by ranking the predictions and taking the first 100 and 50. Therefore, the challenge is to predict correct relationships with a high score. Table 8.3 shows the results for the visual relationships detection tasks. The first line presents the results in [65] and the following four lines the results in [6] according to the link prediction models implemented. The last two lines show the LTNs results for $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$, respectively. The LTNs models outperform the original work

[6] https://github.com/Prof-Lu-Cewu/Visual-Relationship-Detection

Task	Phrase Det.	Phrase Det.	Rel. Det.	Rel. Det.	Pred. Det.	Pred. Det.
Evaluation	R@100	R@50	R@100	R@50	R@100	R@50
Lu et al. [65]	17.03	16.17	14.7	13.86	47.87	47.87
RESCAL [6]	19.17	18.16	16.88	15.88	52.71	52.71
MultiwayNN [6]	18.88	17.75	16.65	15.57	51.82	51.82
ComplEx [6]	19.36	18.25	17.12	16.03	53.14	53.14
DistMult [6]	15.42	14.27	13.64	12.54	42.18	42.18
$\mathcal{T}_{\text{expl}}$	28.12	**23.21**	24.86	**20.82**	91.23	77.32
$\mathcal{T}_{\text{prior}}$	**28.24**	22.72	**25.1**	20.63	**91.88**	**78.63**
Yu et al. [109]	29.43	26.32	31.89	22.68	94.65	85.64

Table 8.3: Results on the Visual Relationship Dataset (R@K stands for recall at K). The $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$ models are comparable and always outperform the state-of-the-art.

on VRD [65] and the state-of-the-art [6] on all the tasks for every measure. The phrase and the relationship detection tasks are the hardest, as they include also the detection of the bounding boxes of the subject and of the object. Therefore, the errors coming from the object detector propagate also to the visual relationship detection models. Adopting the same object detector used by our competitors allows us to compare LTNs results starting from the same level of error coming from the bounding boxes detection. The good results obtained by LTNs models show that LTNs deal with the object detection errors in a better way. The predicate detection task, instead, is easier as it is independent from object detection. We can see the improvement of performance on all the methods. In this task, it is possible to see all the effectiveness of LTNs due to the good improvement of performance. Indeed, the LTNs models are able to correctly classify more than the 90% of all the given pairs of bounding boxes within the first 100 results. In this evaluation, the performance of $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$ are comparable. This fact suggests us that the negative domain and range constraints do not take particular effect on the optimization. Therefore, the improvement given by LTNs is mostly determined by the underlying Neural Tensor Network (Equation 6.11) and the encoding of the bounding boxes with semantic and (extra) geometric features.

8.2.4 Performance on Zero-Shot Learning

In a SII dataset, many types of visual relationships have only few, or even zero, examples (the long tail problem). This is due to the big effort of annotation. Thus, it is important

for a SII system to perform this kind of generalization. The zero-shot learning scenario evaluates the ability of a method to generalize to never seen types of visual relationships. In this setting, both $\mathcal{T}_{\text{expl}}$ and $\mathcal{T}_{\text{prior}}$ models are tested on the defined visual relationship detection tasks and compared with the presented methods. The test is performed only on the 1877 never seen types of visual relationships in the training set (for example, $\langle \mathsf{Elephant}, \mathsf{stand\ on}, \mathsf{Street} \rangle$). In this evaluation, we test the ability of LTNs to generalize from unseen relationships by exploiting similarity with already seen relationships and the logical constraints. Table 8.4 shows the results for the visual relationships detection tasks in the zero-shot learning scenario. The difficulty of this setting can be seen in the huge

Task	Phrase Det.	Phrase Det.	Rel. Det.	Rel Det.	Pred. Det.	Pred. Det.
Evaluation	R@100	R@50	R@100	R@50	R@100	R@50
Lu et al. [65]	3.75	3.36	3.52	3.13	8.45	8.45
RESCAL [6]	6.59	5.82	6.07	5.3	16.34	16.34
MultiwayNN [6]	6.93	5.73	6.24	5.22	16.6	16.6
ComplEx [6]	6.5	5.73	5.82	5.05	15.74	15.74
DistMult [6]	4.19	3.34	3.85	3.08	12.4	12.4
$\mathcal{T}_{\text{expl}}$	13.94	8.98	12.57	8.13	64.5	40.12
$\mathcal{T}_{\text{prior}}$	**15.91**	**11.12**	**14.37**	**10.09**	**70.15**	**46.28**
Yu et al. [109]	17.24	12.96	15.89	12.02	74.65	54.20

Table 8.4: Results on the Visual Relationship Dataset in the zero-shot learning scenario (R@K stands for recall at K). The LTNs models always outperform the state-of-the-art. The use of the logical constraints in $\mathcal{T}_{\text{prior}}$ leads to the best results.

drop in performance for all the methods on all the tasks. The LTNs models $\mathcal{T}_{\text{expl}}$, $\mathcal{T}_{\text{prior}}$ outperform the state-of-the-art. This proves the LTNs ability to generalize to never seen relationships. Indeed, the LTNs learning model exploits the structure in the data to infer similarity between relationships. For example, the triple $\langle \mathsf{Elephant}, \mathsf{stand\ on}, \mathsf{Street} \rangle$ can be inferred by the triple $\langle \mathsf{Horse}, \mathsf{stand\ on}, \mathsf{Street} \rangle$ due to the similarity between $\mathsf{Elephant}$ and Horse. Indeed, both concepts can eat/stand on Grass and both have Legs, Body and Tail. As above, LTNs better deals with the errors of the object detection (phrase and relationship detection) and predicts correctly the majority of the predicates between given pairs of bounding boxes (predicate detection). Moreover, the LTNs model $\mathcal{T}_{\text{prior}}$ trained with data and constraints outperforms the model $\mathcal{T}_{\text{expl}}$ trained with only constraints. The negative domain and range constraints are leveraged by LTNs to exclude some binary predicates for a bounding box with a given subject or object. For example, if the subject

is a physical object, then the predicate cannot be sleep on. Comparing these results with the ones in the normal scenario, we can state that the background knowledge is useful when the training data are scarce. These good results are of great help as many Neural Networks datasets suffer of the long tail problem.

Chapter 9

Related work

In this Chapter, we compare PWCA and LTNs for SII with some works of the-state-of-the-art. However, not all the following methods can be compared with both PWCA and LTNs. In [50] the authors propose a fuzzy DL ontology of spatial relations and an algorithm for building scene graphs. The scene graph is constructed starting from some basic objects in the scene, then, through logical reasoning, semantic relations between objects (or new objects) are inferred. This approach is effective in domains where the objects are tightly related with one or few spatial relations (for example, for detecting particular areas during a brain image analysis). However, in a general scene understanding many actions can occur between two objects. This method is unsupervised as PWCA, the scene is built with logical fuzzy reasoning whereas PWCA exploits clustering of numeric and semantic features and then logical reasoning. This method deals with the uncertainty coming from the object detection with a fuzzy approach as LTNs do. However, LTNs do not use a strict reasoning procedure to build the scene graph but rather exploit logical constraints to locally predict visual relationships.

In [57] the labelling of the scene graph is found through energy minimization of a CRF. The potentials are given by (i) the object/attribute detection on objects; (ii) the manual definition of 16 spatial relations between bounding boxes according to their geometric properties; (iii) text priors, such as co-occurrence of objects and relations found in the image descriptions of Flickr. This work limits the description of a scene to spatial relations between objects. These relations are manually defined and the extension to other relations is not straightforward. LTNs use a different methodology: it maximizes training data and logical knowledge instead of the statistical textual knowledge in Flickr. LTNs are not limited to spatial relations and are able to generalize to new relations simply by learning from examples.

Also in [15] the scene graph is encoded as a graphical model and the task is to correctly find a labelling for nodes and edges through energy minimization. The energy term combines visual information coming from the object detection and logical constraints of a DL knowledge base of the domain. This method needs a dataset to learn the parameters of the graphical model, whereas PWCA is an unsupervised method. PWCA handles multiple labels for the detected objects, whereas [15] does not handle such uncertainty of the object detectors. Finally, PWCA is able to reconstruct composite objects from the detection of their parts with a sort of abductive reasoning. The idea in [15] of combining axioms and visual information is close in spirit to LTNs but with a totally different methodology: LTNs combine Neural Tensor Networks with Fuzzy Predicate Logic instead of a graphical model as in [15]. Moreover, in LTNs it is easier to add more types of constraints, whereas the work in [15] is tailored to domain/range and cardinality constraints.

The work in [78] predicts visual relationships by jointly considering, in the same embedding space, visual information and logical constraints (derived from WordNet) between the relationships. Also LTNs consider visual information and constraints jointly but as a task of maximization of the satisfiability of a knowledge base. Differently from LTNs, this work predicts only a single triple for image without the visual grounding of bounding boxes. Moreover, it is tailored to only 3 logical constraints: visual relationship implication, type-of constraint and mutual exclusion, defined with a rule-based approach. LTNs are more general as they predict relations between bounding boxes and encode many types of constraints with minimal effort.

Visual relationships are detected in [65, 6] by multiplying a visual term (the visual features of the union of the bounding boxes of the subject and of the object computed with a CNN) with a semantic term. This semantic term is a pre-trained word embedding (word2vec) [67] of the subject and of the object labels, such that similar triples are close in the embedding space. In this manner, even if no examples of a visual relationship are in the data, the relationship can be inferred from similarity with already seen visual relationships (zero-shot learning). In [6] the semantic term is computed with statistical link prediction methods [72] that model the statistical information encoded in the Visual Relationship Dataset. The proposed statistical methods improve the results of [65] both in seen and unseen triples. However, this statistical knowledge is tailored to the training set whereas LTNs exploit logical knowledge that can be found in available knowledge bases. In both [65, 6] inconsistent triples, such as $\langle man, eat, chair \rangle$, can be predicted due to the lack of a consistency checking. LTNs are able to avoid this problem by considering, in the training phase, logical axioms (for example, "chairs are not normally edible") that bind predicates with subjects and objects. The ability of LTNs to handle exceptions makes a

SII system able to deal with both real-value data and crisp axioms.

In [109] every visual relationship is predicted with a Neural Network trained on visual features and on the word embeddings of the labels (returned by an object detector) for the subject and the object. This network is regularized with a semantic term that encodes semantic knowledge about the statistical dependencies between the relationships and subjects/objects. This knowledge is taken from the training set and Wikipedia documents. LTNs are different as they exploit only external logical knowledge encoded with constraints. The difference is that the statistical knowledge states only the statistical dependencies (co-occurrences) between subjects/objects and relationships. Whereas logical knowledge can express more information, for example, dependencies between subjects/objects and relationships, properties of the relationships, hierarchies of classes or relationships, mutual exclusions, cardinality constraints and negations of subjects/objects for a given relationship. Logical knowledge is more flexible as it allows a more accurate reasoning on the visual relationships. Moreover, the network in [109] has to predict $\mathcal{O}(|\mathcal{P}_1|^2|\mathcal{P}_2|)$ possible relationships, whereas the searching space of LTNs is $\mathcal{O}(|\mathcal{P}_1|+|\mathcal{P}_2|)$. This implies an important reduction of the number of parameters of the network and a substantial advantage on scalability.

In [107] a scene graph is built with an iterative message passing algorithm where information of the latent state (computed with a Gated Recurrent Unit [18]) of a node (or edge) is passed to the edges (or nodes) to compute the states of the next iteration. Every iteration refines the prediction of the nodes and edges. The features for objects (and pairs of objects) are computed with an object detector. The advantages of this method are (i) the refinement of its predictions through iterations and (ii) the fact that the graph is predicted globally and not locally by triple predictions. This work does not exploit background knowledge and is different in spirit from LTNs. LTNs, instead, are focused on combining logical knowledge with training data for improving the predictions of objects and relationships and thus leading to the scene graph construction.

Some of the previous methods are difficult to compare with PWCA, the main difference with respect to the others (a part of [50]) is the unsupervision of PWCA and the reasoning procedure it uses for finding partial models. PWCA is more suitable to compare with part-based models for object detection. In these models, an object is determined by the composition of its parts. In [17] a CRF is built for discovering whole objects and their parts. The nodes are the objects (whole and its parts), the edges represent the part-whole relation and the fact that two parts belong to the same whole object. The unary potentials are the scores returned by a trained object detector about an object, the binary potentials are manually defined as geometric relations between bounding boxes. As in PWCA, a

whole object is determined by geometric and semantic (the labels) properties of its parts. However, PWCA is more fine grained as it is able to deal with more specific parts (the parts in [17] are only the head, the torso and the legs). Indeed, PWCA can reason about the number of parts composing a whole object: a horse cannot have more than four legs. PWCA does not need to be trained as [17]. PWCA is scalable in the number of objects in the domain, whereas in [17] the possible configurations of parts and wholes grows exponentially.

In [70] a whole object is determined by two trained models: an appearance model about the parts and a location model about the location of the whole object and its parts. These models are learnt in an iterative way: first they are trained on examples from Google images, then on examples mined (with the first version of these models) from the previous Google images and finally on examples mined (with the second version of the models) from the PASCAL VOC dataset [33]. The models of the last iteration are composed with the R-CNN object detector [37]. This method is different in spirit from PWCA as it requires 3 steps of training. PWCA does not need an object detector for whole objects as, in principle, it is able to detect a whole object only from its parts. Moreover, PWCA can reason about the number of parts of a whole object.

Chapter 10

Conclusion

In this thesis, we addressed the Semantic Image Interpretation as the problem of extracting a labelled direct graph, also known as semantically interpreted picture or scene graph, that describes the content of an input image. The labelled nodes represent the objects in the image, the labelled edges represent relationships between the objects. The labels describe the semantic types of the objects and the semantic relations between objects. Labels are taken from the signature of a knowledge base.

We devised two well-founded methods for solving the SII problem. The first one formalizes a semantically interpreted picture as a partial model of a knowledge base. That is, a logical interpretation, defined on a subset of the signature of the knowledge base, such that there exists a model that extends the interpretation. Partial models are constructed with a clustering algorithm (PWCA) that considers geometric and semantic features and performs reasoning on the objects in the scene. The evaluation on the part-whole relation shows that PWCA outperforms methods based only on numerical features. This method is unsupervised, thus it does not need a training set. The research question of dealing with both numeric and geometric features is addressed by considering geometric and semantic distances in the clustering. The research question of inferring new objects with few information is addressed by predicting whole objects from the presence of few part objects. The dealing with constraints of a knowledge base is dealt with the construction of a partial model of the knowledge base.

The second method uses the framework Logic Tensor Networks to predict both nodes and visual relationships. This framework is based on the principle of jointly maximizing both the likelihood of training data and the satisfiability of logical constraints of a knowledge base. The evaluation tested the ability of LTNs to predict both object types of bounding boxes and visual relationships between objects. The results show that LTNs

outperforms the state-of-the-art and, moreover, the use of logical background knowledge improves the results both in a standard setting and in settings with noisy training labels or missing data (zero-shot learning). The research question of dealing with both numeric and geometric features is addressed by encoding the data with a vector containing both kinds of features. The research question of discarding some proposals can be addressed by training LTNs to recognize "background" objects. The dealing with constraints of a knowledge base is dealt with the jointly optimization of the likelihood of both training data and constraints. This method, at the time of writing (autumn 2017), outperformed the state-of-the-art on visual relationship detection on the VRD [6]. Now it is second to a Computer Vision top conference paper [109]. The difference is that our proposal is based on a well-founded and strong framework (LTNs) that tightly integrates data and constraints. Moreover, it easily deals with very expressive background logical knowledge and not only with statistical knowledge

As future work, we want to test LTNs for Semantic Image Interpretation on different settings on VRD. The first setting is the LTNs evaluation with noisy training labels. In the second setting we progressively remove some amounts of data from the training set and see how performance change. Another evaluation of LTNs regards the logical constraints. Our aim is to study the difference in performance by using different categories of constraints (for example, by adding the mutual exclusion and removing the IsA constraints) in the grounded theory of LTNs. In this manner, we want to check if there are constraints more effective than others. In addition, we want to use different object detectors that provide the proposals to LTNs. Another direction of investigation is the application of LTNs to a specific domain of images, such as, the images obtained with the scanning electron microscope (SEM). The idea is to find relations between nanoscience objects detected from images obtained with the SEM. These images are provided by a startup company which performed an initial work on the classification of nanoscience images [69].

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2189–2202, 2012.

[2] Jamal Atif, Céline Hudelot, and Isabelle Bloch. Explanatory reasoning for image understanding using formal concept analysis and description logics. *IEEE Trans. Systems, Man, and Cybernetics: Systems*, 44(5):552–570, 2014.

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2^{ND} Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, pages 722–735, Berlin, Heidelberg, 2007. Springer-Verlag.

[4] S. Baadel, F. Thabtah, and J. Lu. Overlapping clustering: A review. In *2016 SAI Computing Conference (SAI)*, pages 233–237, July 2016.

[5] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.

[6] Stephan Baier, Yunpu Ma, and Volker Tresp. Improving visual relationship detection using semantic modeling of scene descriptions. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21–25, 2017, Proceedings, Part I*, pages 53–68, Cham, 2017. Springer International Publishing.

[7] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*

- *Volume 1*, ACL '98, pages 86–90, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

[8] Jon Barwise. *Handbook of mathematical logic*, volume 90. Elsevier, 1982.

[9] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA, 1995.

[10] Isabelle Bloch. Fuzzy spatial relationships for image processing and interpretation: A review. *Image Vision Comput.*, 23(2):89–110, feb 2005.

[11] Grady Booch, James Rumbaugh, and Ivar Jacobson. *Unified Modeling Language User Guide, The (2Nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005.

[12] Alex Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial intelligence*, 82(1-2):353–367, 1996.

[13] João Carreira and Cristian Sminchisescu. CPMC: automatic object segmentation using constrained parametric min-cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1312–1328, 2012.

[14] Neelima Chavali, Harsh Agrawal, Aroma Mahendru, and Dhruv Batra. Object-proposal evaluation protocol is 'gameable'. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 835–844. IEEE Computer Society, 2016.

[15] Na Chen, Qian-Yi Zhou, and Viktor Prasanna. Understanding web images by object relation network. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 291–300, New York, NY, USA, 2012. ACM.

[16] Peter Pin-Shan Chen. The entity-relationship model-toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, 1(1):9–36, 1976.

[17] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[18] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In

Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi, editors, *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111. Association for Computational Linguistics, 2014.

[19] Wongun Choi, Yu-Wei Chao, Caroline Pantofaru, and Silvio Savarese. Understanding indoor scenes using 3d geometric phrases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 33–40, 2013.

[20] Philipp Cimiano, Alexander Mädche, Steffen Staab, and Johanna Völker. Ontology learning. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 245–267. Springer, 2009.

[21] Ramazan Gokberk Cinbis, Jakob J. Verbeek, and Cordelia Schmid. Segmentation driven object detection with fisher vectors. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 2968–2975. IEEE Computer Society, 2013.

[22] P. Cintula, P. Hájek, and C. Noguera. *Handbook of Mathematical Fuzzy Logic*. Number v. 1 in Handbook of Mathematical Fuzzy Logic. College Publications, 2011.

[23] Jose Alfredo Ferreira Costa and Marcio Luiz De Andrade Netto. Estimating the number of clusters in multivariate data by self-organizing maps. *International Journal of Neural Systems*, 9(03):195–202, 1999.

[24] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3298–3308. IEEE Computer Society, 2017.

[25] Ivan Donadello. Ontology based semantic image interpretation. In Elena Bellodi and Alessio Bonfietti, editors, *Proceedings of the Doctoral Consortium (DC) co-located with the 14th Conference of the Italian Association for Artificial Intelligence (AI*IA 2015), Ferrara, Italy, September 23-24, 2015.*, volume 1485 of *CEUR Workshop Proceedings*, pages 19–24. CEUR-WS.org, 2015.

[26] Ivan Donadello and Luciano Serafini. Mixing low-level and semantic features for image interpretation - A framework and a simple case study. In Lourdes Agapito, Michael M. Bronstein, and Carsten Rother, editors, *Computer Vision - ECCV 2014*

Workshops - Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part II, volume 8926 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2014.

[27] Ivan Donadello and Luciano Serafini. Integration of numeric and symbolic information for semantic image interpretation. *Intelligenza Artificiale*, 10(1):33–47, 2016.

[28] Ivan Donadello, Luciano Serafini, and Artur S. d'Avila Garcez. Logic tensor networks for semantic image interpretation. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1596–1602. ijcai.org, 2017.

[29] E.R. Dougherty. *An introduction to morphological image processing*. Tutorial texts in optical engineering. SPIE Optical Engineering Press, 1992.

[30] Bob DuCharme. *Learning SPARQL*. O'Reilly Media, Inc., 2011.

[31] Desmond Elliott and Arjen de Vries. Describing images using inferred visual dependency representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 42–52, 2015.

[32] Ian Endres, Kevin J. Shih, Johnston Jiaa, and Derek Hoiem. Learning collections of part models for object recognition. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 939–946. IEEE Computer Society, 2013.

[33] Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[34] Christiane Fellbaum, editor. *WordNet: an electronic lexical database*. MIT Press, 1998.

[35] Pedro F. Felzenszwalb, Ross B. Girshick, David A. McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2010.

[36] Ross Girshick. Fast r-cnn. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1440–1448, Washington, DC, USA, 2015. IEEE Computer Society.

[37] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(1):142–158, 2016.

[38] Georgia Gkioxari, Ross B. Girshick, and Jitendra Malik. Actions and attributes from wholes and parts. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2470–2478. IEEE Computer Society, 2015.

[39] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web. (Advanced Information and Knowledge Processing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[40] Nicola Guarino and Giancarlo Guizzardi. On the reification of relationships. In Mario A. Bochicchio and Giansalvatore Mecca, editors, *24th Italian Symposium on Advanced Database Systems, SEBD 2016, Ugento, Lecce, Italy, June 19-22, 2016, Ugento, Lecce, Italia, June 19-22, 2016.*, pages 350–357. Matematicamente.it, 2016.

[41] Ramanathan Guha and Dan Brickley. RDF schema 1.1, feb 2014.

[42] Saurabh Gupta and Jitendra Malik. Visual semantic role labeling. *arXiv preprint arXiv:1505.04474*, 2015.

[43] Volker Haarslev, Kay Hidde, Ralf Möller, and Michael Wessel. The racerpro knowledge representation and reasoning system. *Semantic Web Journal*, 3(3):267–277, 2012.

[44] P. Hájek. *Metamathematics of Fuzzy Logic*. Trends in Logic. Springer, 2001.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1904–1916, 2015.

[46] Shengfeng He and Rynson W. H. Lau. Oriented object proposals. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 280–288. IEEE Computer Society, 2015.

[47] Graeme Hirst and David St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332, 1998.

[48] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible SROIQ. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2-5, 2006*, pages 57–67. AAAI Press, 2006.

[49] Jan Hendrik Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals? *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(4):814–830, 2016.

[50] Céline Hudelot, Jamal Atif, and Isabelle Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15):1929–1951, 2008.

[51] Celine Hudelot, Nicolas Maillot, and Monique Thonnat. Symbol grounding for semantic image interpretation: From image data to semantics. In *Proc. of the 10th IEEE Intl. Conf. on Computer Vision Workshops*, ICCVW '05. IEEE Computer Society, 2005.

[52] Yunjae Jung, Haesun Park, Ding-Zhu Du, and Barry L. Drake. A decision criterion for the optimal number of clusters in hierarchical clustering. *Journal of Global Optimization*, 25(1):91–111, 2003.

[53] Teuvo Kohonen. The self-organizing map. *Neurocomputing*, 21(1-3):1–6, 1998.

[54] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

[55] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012.

[57] Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. Baby talk: Understanding and generating simple image descriptions. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pages 1601–1608. IEEE Computer Society, 2011.

[58] Alina Kuznetsova, Sung Ju Hwang, Bodo Rosenhahn, and Leonid Sigal. Expanding object detector's horizon: Incremental learning framework for object detection in videos. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 28–36. IEEE Computer Society, 2015.

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

[60] Yikang Li, Wanli Ouyang, Xiaogang Wang, and Xiaoou Tang. Vip-cnn: Visual phrase guided convolutional neural network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7244–7253. IEEE Computer Society, 2017.

[61] Xiaodan Liang, Lisa Lee, and Eric P. Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4408–4417. IEEE Computer Society, 2017.

[62] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[63] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, volume 9905 of *Lecture Notes in Computer Science*, pages 21–37. Springer, 2016.

[64] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014.

[65] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I*, pages 852–869, Cham, 2016. Springer International Publishing.

[66] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A knowledge base from multilingual wikipedias. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org, 2015.

[67] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[68] Koji Miyajima and Anca Ralescu. Spatial organization in 2d segmented images: Representation and recognition of primitive spatial relations. *Fuzzy Sets Syst.*, 65(2-3):225–236, aug 1994.

[69] Mohammad Hadi Modarres, Rossella Aversa, Stefano Cozzini, Regina Ciancio, Angelo Leto, and Giuseppe Piero Brandino. Neural network for nanoscience scanning electron microscope image recognition. *Scientific reports*, 7(1):13282, 2017.

[70] Davide Modolo and Vittorio Ferrari. Learning semantic part-based models from google images. *CoRR*, abs/1609.03140, 2016.

[71] Bernd Neumann and Ralf Möller. On scene interpretation with description logics. *Image Vision Comput.*, 26(1):82–101, 2008.

[72] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[73] Daniel Nyga, Ferenc Balint-Benczedi, and Michael Beetz. PR2 looking at things - ensemble learning for unstructured information processing with markov logic networks. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 3916–3923. IEEE, 2014.

[74] Irma Sofía Espinosa Peraldí, Atila Kaya, and Ralf Möller. Formalizing multimedia interpretation based on abduction over description logic aboxes. In Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, and Ulrike Sattler, editors, *Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009*, volume 477 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[75] Giulio Petrucci, Chiara Ghidini, and Marco Rospocher. Ontology learning in the deep. In *Knowledge Engineering and Knowledge Management: 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings 20*, pages 480–495. Springer, 2016.

[76] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan T. Barron, Ferran Marqués, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(1):128–140, 2017.

[77] Luc De Raedt. *Logical and Relational Learning: From ILP to MRDM (Cognitive Technologies)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2008.

[78] Vignesh Ramanathan, Congcong Li, Jia Deng, Wei Han, Zhen Li, Kunlong Gu, Yang Song, Samy Bengio, Chuck Rosenberg, and Fei-Fei Li. Learning semantic relationships for better action retrieval in images. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1100–1109. IEEE Computer Society, 2015.

[79] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 779–788. IEEE Computer Society, 2016.

[80] Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *CoRR*, abs/1412.6596, 2014.

[81] Raymond Reiter. On closed world data bases. In Hervé Gallaire and Jack Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, 1977.*, Advances in Data Base Theory, pages 55–76, New York, 1977. Plemum Press.

[82] Raymond Reiter and Alan K. Mackworth. A logical framework for depiction and image interpretation. *Artificial Intelligence*, 41(2):125–155, 1989.

[83] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017.

[84] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

[85] Matteo Ruggero Ronchi and Pietro Perona. Describing common human visual actions in images. In Xianghua Xie, Mark W. Jones, and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*, pages 52.1–52.12. BMVA Press, 2015.

[86] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. *Int. J. Comput. Vision*, 77(1-3):157–173, may 2008.

[87] Mohammad Amin Sadeghi and Ali Farhadi. Recognition using visual phrases. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1745–1752. IEEE, 2011.

[88] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[89] Carsten Schröder and Bernd Neumann. On the logics of image interpretation: model-construction in a formal knowledge-representation framework. In *Proceedings 1996 International Conference on Image Processing, Lausanne, Switzerland, September 16-19, 1996*, pages 785–788. IEEE Computer Society, 1996.

[90] Karin Kipper Schuler. *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon.* PhD thesis, University of Pennsylvania, Philadelphia, PA, USA, 2005. AAI3179808.

[91] Luciano Serafini and Artur S. d'Avila Garcez. Learning and reasoning with logic tensor networks. In Giovanni Adorni, Stefano Cagnoni, Marco Gori, and Marco Maratea, editors, *AI*IA 2016 Advances in Artificial Intelligence: XVth International Conference of the Italian Association for Artificial Intelligence, Genova, Italy, November 29 – December 1, 2016, Proceedings*, pages 334–348, Cham, 2016. Springer International Publishing.

[92] Luciano Serafini, Ivan Donadello, and Artur S. d'Avila Garcez. Learning and reasoning in logic tensor networks: theory and application to semantic image interpretation. In Ahmed Seffah, Birgit Penzenstadler, Carina Alves, and Xin Peng, editors,

Proceedings of the Symposium on Applied Computing, SAC 2017, Marrakech, Morocco, April 3-7, 2017, pages 125–130. ACM, 2017.

[93] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013.

[94] Rob Shearer, Boris Motik, and Ian Horrocks. Hermit: A highly-efficient owl reasoner. In *OWLED*, volume 432, page 91, 2008.

[95] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[96] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *J. Web Sem.*, 5(2):51–53, 2007.

[97] Barry Smith, Christian von Ehrenfels, and Philosophia Verlag. *Foundations of Gestalt theory*. Philosophia Verlag Munich, Germany, 1988.

[98] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934, 2013.

[99] Richard Socher, Milind Ganjoo, Christopher D. Manning, and Andrew Y. Ng. Zero-shot learning through cross-modal transfer. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 935–943, 2013.

[100] Robert Speer and Catherine Havasi. Conceptnet 5: A large semantic network for relational knowledge. In *The Peoples Web Meets NLP*, pages 161–176. Springer, 2013.

[101] Christian Szegedy, Scott E. Reed, Dumitru Erhan, and Dragomir Anguelov. Scalable, high-quality object detection. *CoRR*, abs/1412.1441, 2014.

[102] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[103] Christopher Town. Ontological inference for image and video analysis. *Mach. Vision Appl.*, 17(2):94–115, apr 2006.

[104] Yi-Hsuan Tsai, Onur C. Hamsici, and Ming-Hsuan Yang. Adaptive region pooling for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 731–739. IEEE Computer Society, 2015.

[105] Ian Tsarkov, Dmitryand Horrocks. Fact++ description logic reasoner: System description. In Natarajan Furbach, Ulrichand Shankar, editor, *Automated Reasoning: Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006. Proceedings*, pages 292–297, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[106] Jasper R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, and Arnold W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[107] Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[108] Mark Yatskar, Luke Zettlemoyer, and Ali Farhadi. Situation recognition: Visual semantic role labeling for image understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[109] Ruichi Yu, Ang Li, Vlad I. Morariu, and Larry S. Davis. Visual relationship detection with internal and external linguistic knowledge distillation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1068–1076. IEEE Computer Society, 2017.

[110] Alan Yuille and Aude Oliva. Frontiers in computer vision: Nsf white paper, November 2010. `http://www.frontiersincomputervision.com/WhitePaperInvite.pdf`.

[111] Lotfi A Zadeh. Fuzzy sets. In *Fuzzy Sets, Fuzzy Logic, And Fuzzy Systems: Selected Papers by Lotfi A Zadeh*, pages 394–432. World Scientific, 1996.

[112] Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. Visual translation embedding network for visual relation detection. In *2017 IEEE Conference on*

Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 3107–3115. IEEE Computer Society, 2017.

[113] Yuke Zhu, Alireza Fathi, and Li Fei-Fei. Reasoning about object affordances in a knowledge base representation. In *European conference on computer vision*, pages 408–424. Springer, 2014.

[114] Yukun Zhu, Raquel Urtasun, Ruslan Salakhutdinov, and Sanja Fidler. segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4703–4711. IEEE Computer Society, 2015.

[115] C. Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 391–405. Springer, 2014.

Appendix A

KnowPic, the Semantic Image Interpretation Demo

A Semantic Image Interpretation system has been developed during the doctoral years. The name of the system is KnowPic, and it refers to a system that *knows your pictures*. Indeed, this system is a Web Application that allows a user to upload a picture and see the results of the semantic interpretation of the input picture. The result is a set of visual relationships: triples $\langle subject, predicate, object \rangle$. For every triple, it is possible to show the corresponding bounding boxes by simply clicking on the "show bounding boxes" button. This set of grounded (that is, linked with image bounding boxes) visual relationships composes the scene graph associated to the input picture.

The technologies used for developing this demo are:

Web Application KnowPic is a Web Application built with Flask (`http://flask.pocoo.org/`): a micro web-development framework for Python.

Object Detection The object detector used here is Fast R-CNN [36]. It is trained on the VRD dataset and provides the bounding boxes corresponding to the nodes of the scene graph. These bounding boxes are filtered with a threshold on the classification score of each bounding box.

Visual Relationship Detection The bounding boxes of the object detection are provided to the LTNs model trained with data and prior knowledge on the VRD and on the VRD knowledge base of Section 8.2. Every pair of bounding boxes is classified with the LTNs model according to the binary predicates of the VRD. The triples are then filtered with a threshold on the classification score.

In the following some screenshots of the results of KnowPic on images downloaded from Google.

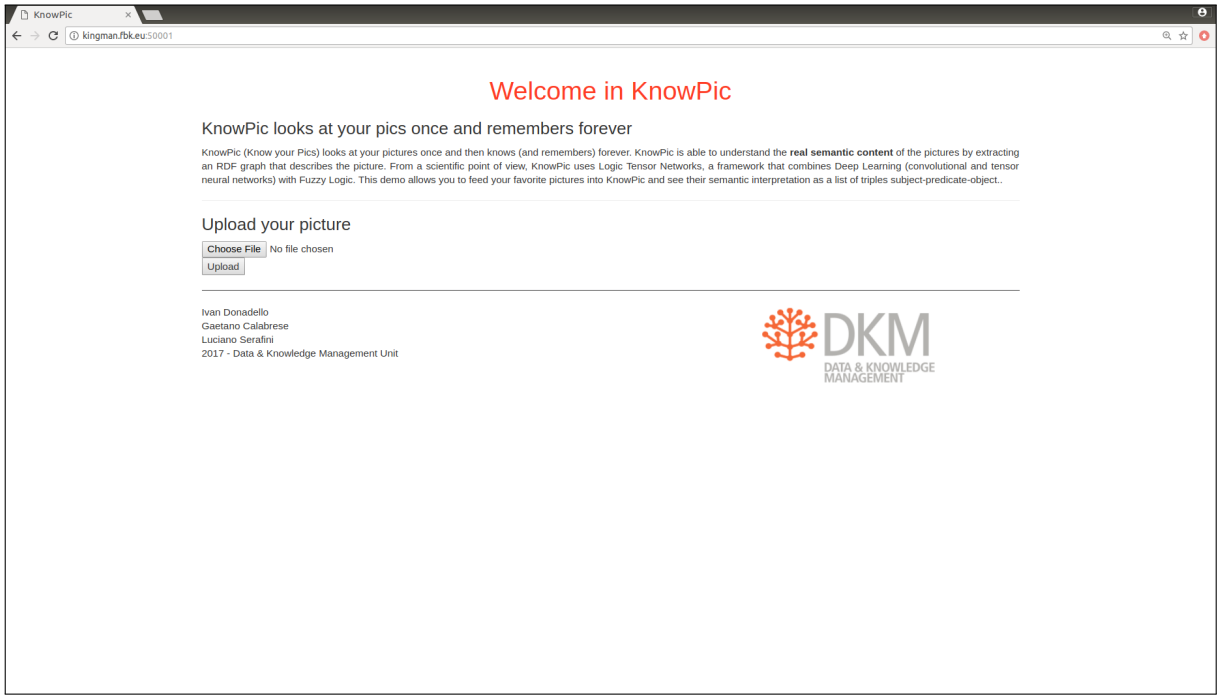


Figure A.1: The input page of KnowPic: here it is possible to upload a picture and run the system.

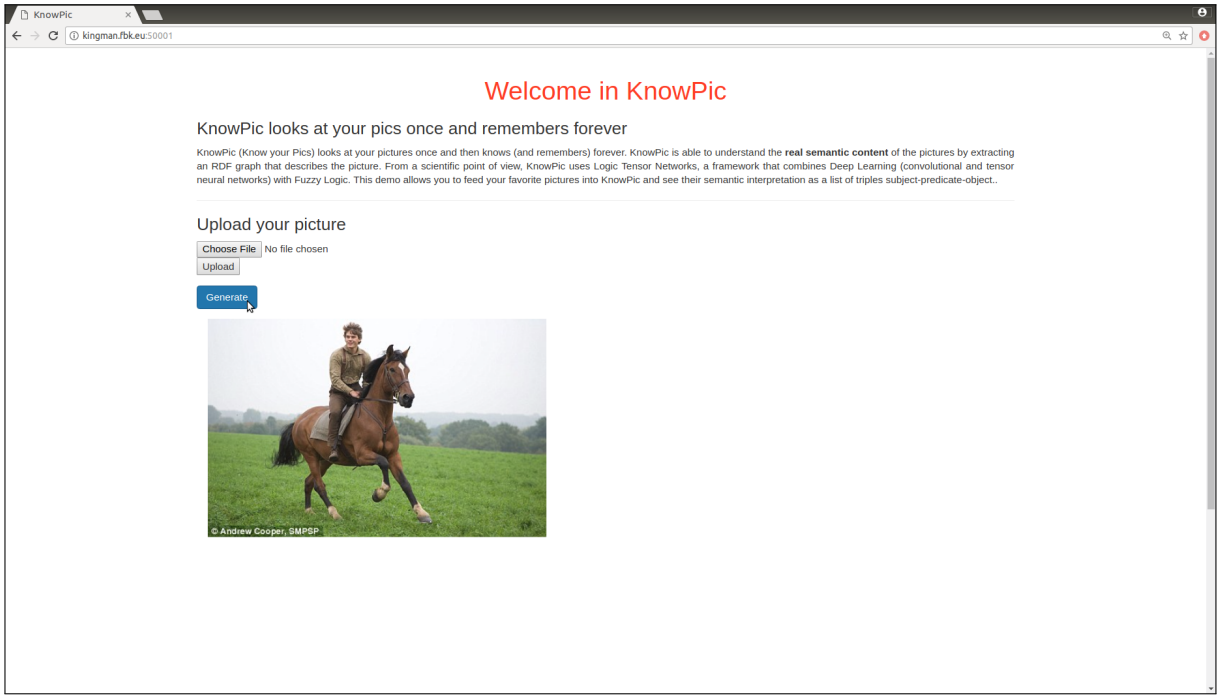


Figure A.2: Once the picture is uploaded, the "generate" button starts the SII of the picture.

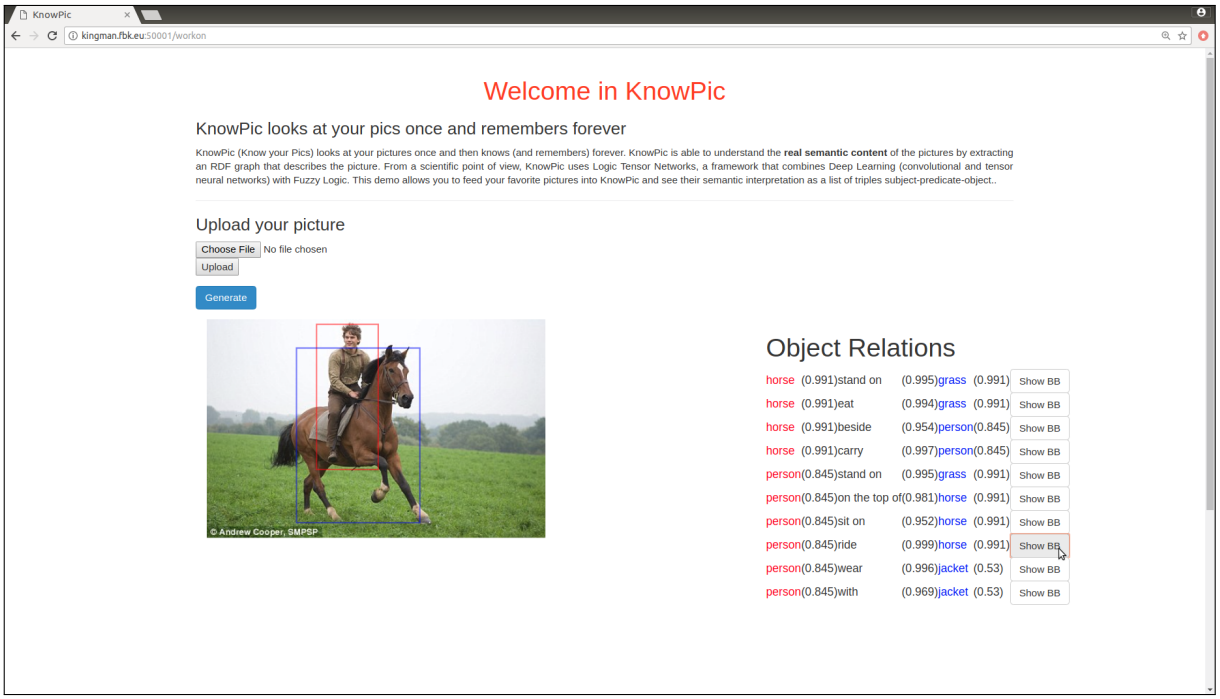


Figure A.3: The SII result for the running example. The number following the label for the subject (in red) and the object (in blue) is the confidence score returned by Fast R-CNN. The number following the predicate is the confidence score of the triple returned by LTNs.

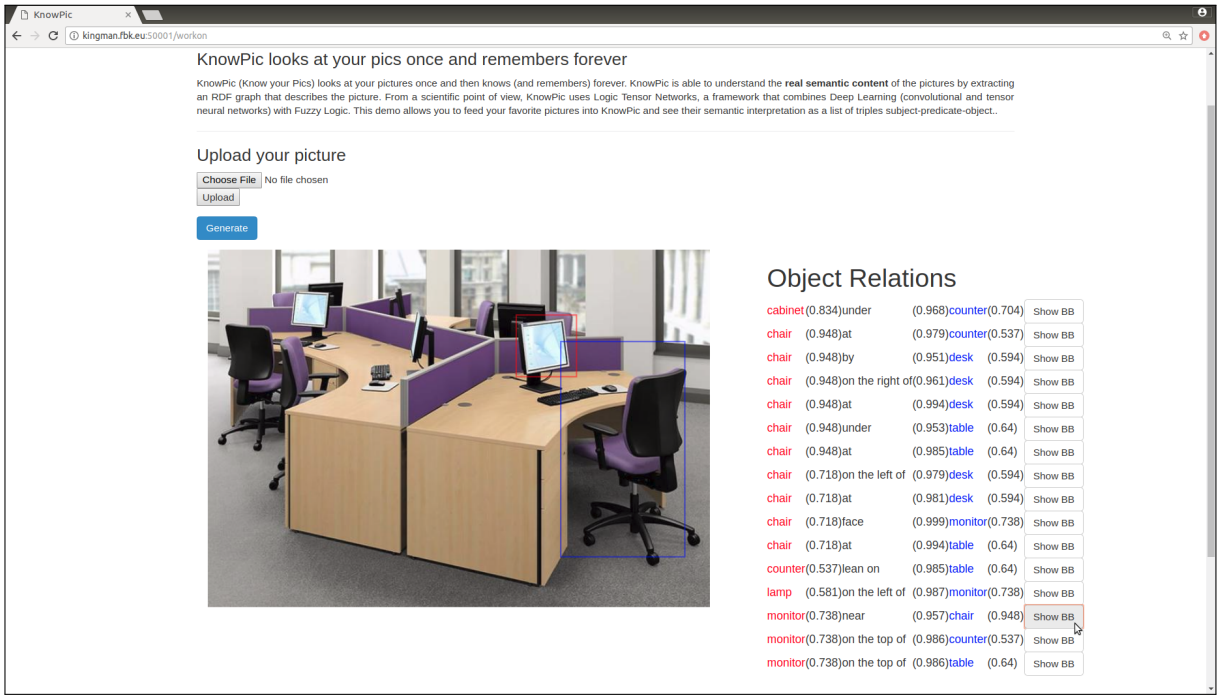


Figure A.4: The SII result for a picture containing an indoor scene.

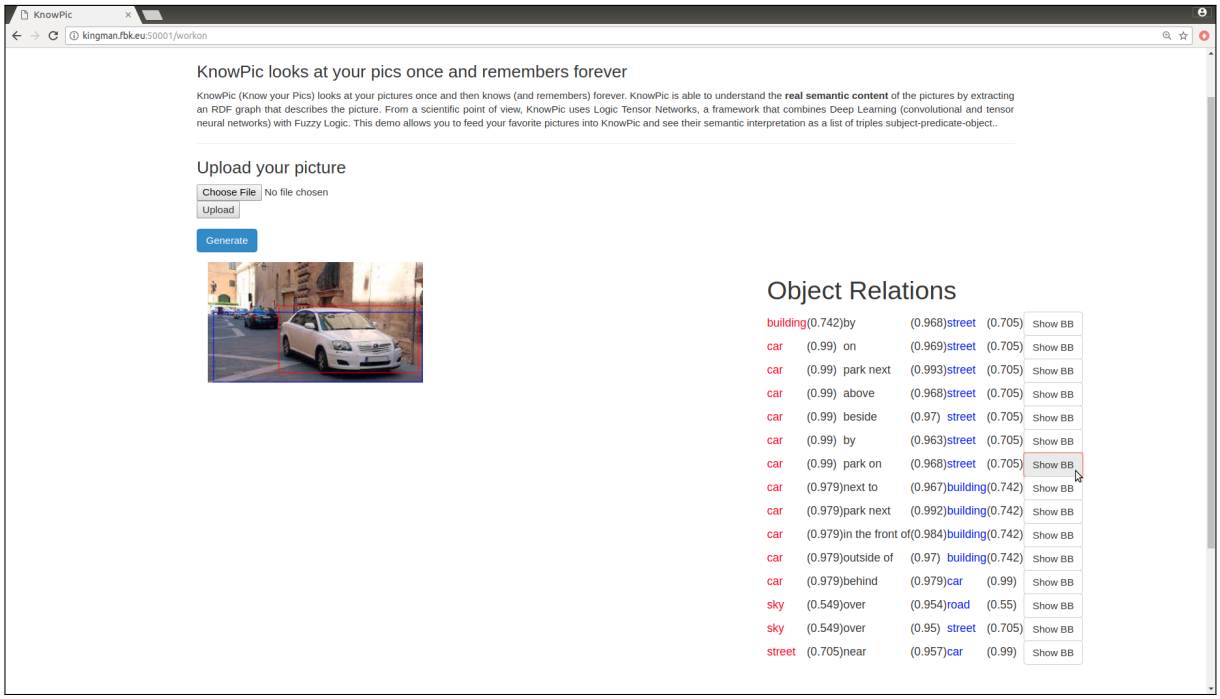


Figure A.5: The SII result for a picture containing an urban scene.