Learning a Knowledge Base of Ontological Concepts for High-Level Scene Interpretation

Johannes Hartz, Bernd Neumann

Cognitive Systems Laboratory
Department of Informatics
Hamburg University
{*hartz, neumann*}*@informatik.uni-hamburg.de*

July 9, 2007

Zusammenfassung

Ontologische Konzeptbeschreibungen von Aggregaten spielen eine wesentliche Rolle bei modellbasierter Szeneninterpretation. Ein Aggregat spezifiziert eine Objektmenge mit bestimmten Eigenschaften und Beziehungen, die zusammengenommen ein bedeutungstragender Bestandteil einer Szene sind. In diesem Bericht zeigen wir, wie Aggregatkonzepte von Objekten mit räumlichen Beziehungen von positiven und negativen Beispielen erlernt werden können. Unser Ansatz basiert auf dem von Mitchell ([1]) eingeführten Version Space Learning und zeichnet sich durch eine ausdrucksstarke Repräsentationssprache mit quantitativen sowie qualitativen Attributen und Relationen aus. Anhand von Beispielen aus der Gebäudedomäne zeigen wir, dass Aggregatkonzepte für Fensterreihen, Balkons und andere Strukturen in der Tat anhand von annotierten Bildern gelernt und erfolgreich in der konzeptuellen Wissensbasis eines Szeneninterpretationssystems eingesetzt werden können. Darüber hinaus legen wir dar, dass unser Ansatz verwendet werden kann um ontologische Konzepte jeder Art zu lernen, mit sehr wenigen Einschränkungen.

Abstract

Ontological concept descriptions of scene objects and aggregates play an essential role in model-based scene interpretation. An aggregate specifies a set of objects with certain properties and relations which together constitute a meaningful scene entity. In this paper we show how ontological concept descriptions for spatially related objects and aggregates can be learnt from positive and negative examples. Our approach, based on Version Space Learning introduced by Mitchell ([1]), features a rich representation language encompassing quantitative and qualitative attributes and relations. Using examples from the buildings domain, we show that aggregate concepts for window arrays, balconies and other structures can in fact be learnt from annotated images and successfully employed in the conceptual knowledge base of a scene interpretation system. Furthermore we argue that our approach can be extended to cover ontological concepts of any kind, with very few restrictions.

1 Introduction

In computer vision, growing interest in artificial cognitive systems has brought about increased efforts to extend vision systems towards capabilities for high-level vision or scene interpretation. These are terms commonly used for vision tasks going beyond single-object recognition, such as inferring the existence and location of occluded aggregate parts from already observed ones. As explicated in [3], scene interpretation can be modelled formally as a knowledge-based process. The burden of the interpretation process lies on the conceptual descriptions, and the richer a domain, the more demanding is the task of designing these descriptions. It is foreseeable that designing knowledge bases for larger applications using a handcrafting approach will be prohibitively costly and error-prone.

We therefore started to investigate supervised learning in the eTRIMS project, with the belief that in the long run high-level vision can only be achieved by leading the system through a supervised learning phase where the concepts for a particular domain are acquired based on examples. Different from a probabilistic approach (e.g. [4], [5], [6]), we chose the representation language used in our scene interpretation system SCENIC [7], [20] which represents variability in terms of ranges with crisp boundaries or enumeration of possible values. Apart of the fact that this way we can evaluate the learnt concepts by applying them to real-world scenes through the SCENIC system, this approach also allows us to invoke and extend well-known learning methods from symbolic AI.

Our approach is in the spirit of the seminal work of Winston [18] who showed how spatial structures in the blocks-world could be learnt. We rephrase this problem for a more general domain by using the Version Space Learning framework.

Our main contributions are

- developing a description language for spatial object arrangements,
- applying the learning procedure to a concrete real-world domain, and
- evaluating the results in an operational scene interpretation system.

In the next section we present Mitchell's Version Space Learning framework and the basic types of our representation language. The language comprises conventional symbolic descriptions as well as ranges of numerical values and spatial relations designed to allow realistic concept descriptions. Section 3 deals with the problem of hypothesis selection which arises when several concept descriptions correctly cover all positive and negative examples. This is the rule rather than the exception in Version Space Learning. In Section 4 we argue that a comprehensive knowledge base of ontological concepts of any kind can be learnt in a Version Space framework, given general-specific orderable concept attributes and a finite set of concept relations. We also introduce the term *concept differentiation* as a quality measure for a conceptual knowledge base and present an approach for learning maximally diverse concepts. In Section 5 we present experimental results for the application domain of building facades. Section 6, finally, presents conclusions and an outlook on further work.

Figure 1: Annotated training image with four instances of aggregate "Entrance"

2 Version Space Learning of ontological concepts

2.1 Learning Procedure

Version Space Learning ([1], [2]) is a framework for supervised concept learning, i.e. learning by means of positive and negative examples given by a teacher. During the learning process, the space of possible concept hypotheses VS is implicitly represented through an upper and a lower bound on the generality of the hypotheses $h \in VS$. The General Boundary GB contains all maximally general members of VS, the Specific Boundary SB contains all maximally specific members of VS. GB and SB completely determine VS as the set of hypotheses h being *more-general-or-equal* to an element of SB and *more-specific-or-equal* to an element of GB.

Initially, GB includes all possible training examples and SB excludes all possible training examples. As a positive example e^+ is presented, SB has to be generalised to include e^+. As a negative example e^- is presented, GB must be specialised to exclude e^-. Both, generalisation and specialisation steps, are chosen to be minimal in the sense that as few instances as possible besides e^+ or e^- are included or excluded, respectively. In contrast to minimal generalisations, the minimal specialisation of a hypothesis h leads to a set of hypotheses $\{h', h'', ...\}$, at least for non-trivial cases. For the sake of compactness, more elaborated representation schemes ([8], [9], [10]) and training procedures (([11], [12]) for Version Space Learning are omitted here. Theoretic considerations for inductive concept learning can be found in [13].

The representation of the Version Space by the two boundaries, together with an appropriate revision strategy, allows every hypothesis that is consistent with the training data to be generated. Note, that what is required in order to realise this type of representation is a concept language that constitutes concepts which satisfy the properties of a concept lattice ([14]), i.e. a partial general-specific ordering can be imposed on them.

For the application to the eTRIMS domain, annotated images of building facades are used as input for the learning process. In these images meaningful

scene objects have been segmented and labeled (Fig. 1). Scene aggregates are specified solely through a set description of their parts.
For the actual training process we use a concept language with attribute types presented in Section 2.2. To give an intuition of the nature of the concept language beforehand, we present an abbreviated concept description for the aggregate "Entrance", generalised from the four positive examples in Fig. 1:

Size and configuration	
Aggregate Width =	[184..216] cm
Aggregate Height =	[299..366] cm
Composition	
Has-Parts =	[3..4]
door =	[1..1]
stairs =	[1..1]
canopy =	[0..1]
railing =	[0..1]
sign =	[0..1]
Symbolic attributes	
Shape =	{ Quadratic }
Internal spatial relations	
(stairs011) BelowNeighbourOf	[0..2] (door012)
(door012) AboveNeighbourOf	[0..2] (stairs011)

Table 1: Generalised aggregate description "Entrance"

2.2 Representation

In this section we describe the attribute types used to formulate concept descriptions. We also specify generalisation and specialisation criteria which can be used to determine their general-specific ordering (denoted $\leq$ and $\geq$). A methodology to compute minimal concept attribute generalisations (denoted $\uparrow$) is presented, which is needed to extend attributes of concept hy-

potheses $h \in SB$ to cover attribute values of positive examples e_i^+. Specialisation methods (denoted $\downarrow$) to exclude attribute values of negative examples e_i^- from attributes in concept hypotheses $h \in GB$ are also presented.

2.2.1 Symbol set type

The symbol set type describes disjunctive symbolic or discrete numerical attribute values.

Example: $Colour = \{Red, Green, Blue\}$

- General-specific ordering of symbol sets S_1 and S_2 of disjoint symbols $\{s_1, s_2, ...\}$:
  - Iff $S_1 \supseteq S_2$: $S_1 \geq S_2$
- Obtaining symbol set S_3 from $S_1 \in h \uparrow S_2 \in e^+$:
  $S_3 = S_1 \cup S_2$
- Obtaining symbol sets S_i from $S_1 \in h \downarrow S_2 \in e^-$:
  $\forall s_i \in S_1 \wedge s_i \in S_2 : S_i = S_1 \setminus \{s_i\}$

2.2.2 Range type

The range type describes a convex range of metric attribute values. Specialized ranges can have (half) open boundaries due to the exclusion of discrete values. Ranges can contain symbolic infinity values -INF and INF.

Example: $Aggregate\ Height = [160..\text{INF}]$

- General-specific ordering for ranges $R_1 = [l_1..u_1]$ and $R_2 = [l_2..u_2]$:
  - Iff $R_1 \supseteq R_2$: $R_1 \geq R_2$
- Obtaining range R_3 from $R_1 \in h \uparrow R_2 \in e^+$:
  - Iff $l_2 < l_1$: $R_3 = [l_2..u_1]$
  - Iff $u_2 > u_1$: $R_3 = [l_1..u_2]$
  - Iff $l_2 < l_1 \wedge u_2 > u_1$: $R_3 = [l_2..u_2]$

- Obtaining range R_3 from $R_1 \in h \downarrow R_2 \in e^-$:
  - Iff $u_1 > u_2 \wedge l_1 \leq u_2$: $R_3 =]u_2..u_1]$
  - Iff $l_1 < l_2 \wedge u_1 \geq l_2$: $R_3 = [l_1..l_2[$

Note that theoretically the two cases of specialisation may be applicable at the same time, but because the Version Space enforces all $h_i \in GB$ to be more general than any $h_i \in SB$, at most one of the two possible specialisations is valid.

2.2.3 Composition type

The composition type describes
1. the number of aggregate parts by the range attribute N and
2. the different part types by the symbol set attribute TN and
3. the number of parts of each type by the subordinate range attributes $T_{1..n}$ in TN.

$N = [MAX_{i \in n}(l_i) \leq nl \leq \Sigma_{i \in n}(l_i)..MIN_{i \in n}\ (u_i) \leq nu \leq \Sigma_{i \in n}(u_i)]$[1]
$TN = \{T_1 = [l_1..u_1],\ T_2 = [l_2..u_2],...,\ T_n = [l_n..u_n]\}$

Example: $Has - Parts = [3..6]$
$Triangle = [2..3]$
$Square = [1..3]$

- General-specific ordering for compositions C_1 and C_2:
  - Iff $N_1 \geq N_2 \wedge \forall T_i \in TN_2 \geq T_i \in TN_1$: $C_1 \geq C_2$

To generalise the compositional properties of an aggregate, all ranges in the composition can be treated individually.

- Obtaining composition C_3 from $C_1 \in h \uparrow C_2 \in e^+$:
  - $N_3 = N_1 \uparrow N_2$, $TN_3 = TN_1$,
    $\forall T_i \in TN_3 = T_i \in TN_1 \uparrow T_i \in TN_2$

[1]The actual values of nl and nu depend on preceding generalisation or specialisation steps

[1]The actual values of nl and nu depend on preceding generalisation or specialisation steps

When specialising the composition we have to consider dependencies between the total number of parts and the number of parts per type explicitly.

- Obtaining C_3 from $N_1 \in h \downarrow N_2 \in e^-$ might lead to the same specialisation step for subranges T_i in set TN_3:
  - Iff $nu_1 > nu_2 \wedge nl_1 \leq nu_2$: $N_3 =]nu_2..nu_1]$,
    $TN_3 = TN_1$
  - Iff $nl_1 < nl_2 \wedge nu_1 \geq nl_2$: $N_3 = [nl_1..nl_2[$,
    $TN_3 = TN_1$, $\forall T_i \in TN_3$: $T_i = [l_i..\ MIN(u_i, nl_2)[$
- Obtaining C_3 from $T_i \in TN_1 \in h \downarrow T_i \in TN_2 \in e^-$ might lead to a specialisation step of N_3:
  - Iff $u1_i > u2_i \wedge l1_i \leq u2_i$:
    $N_3 = [MAX(nl_1, \Sigma_{i \in n}(l3_i))..nu_1]$,
    $TN_3 = TN_1$, $T_i \in TN_3 =]u2_i..u1_i]$
  - Iff $l1_i < l2_i \wedge u1_i \geq l2_i$:
    $N_3 = [nl_1..MIN(nu_1, \Sigma_{i \in n}(u3_i))]$,
    $TN_3 = TN_1$, $T_i \in TN_3 = [l1_i..l2_i[$

2.2.4 Predicate type

The predicate type represents a freely definable n-ary boolean function over part attribute values. Predicates $p_1..p_n$ are organised in a set P.

Example: $Predicates = \{FuzzyEqual(Parts - Area)\}$

Note that since predicates constrain attribute values, they behave contrarily to symbol sets!

- General-specific ordering for predicates in sets P_1 and P_2:
  - Iff $P_1 \subseteq P_2$: $P_1 \geq P_2$
- Obtaining predicate set P_3 from $P_1 \in h \uparrow P_2 \in e^+$:
  - $P_3 = P_1 \cap P_2$

- Obtaining predicate set P_3 from $P_1 \in h \downarrow P_2 \in e^-$:
  - $\forall p_i \notin P_2 : P_i = P_1 + p_i$

2.2.5 Spatial relation type

Spatial relations are learnt between the parts of an aggregate and between the aggregate and possible sourrounding entities, which might be scene objects or other aggregates. To represent the spatial relation between two objects, we employ an 8-neighbourhood to obtain a finite set of possible relations. For this purpose the bounding box of an object induces the eight octants of its neighbourhood: {Left, AboveLeft, Above, AboveRight, Right, BelowRight, Below, BelowLeft}. To quantise spatial relations we use the Euclidean distance d between the related objects' boundaries.

Example: $SR = \{$(triangle003) Above [45..45] (Square012)$\}$[2]

Each spatial relation is a 4-tuple. Spatial relations $l_1..l_n$ are organized in a set L and are treated like predicates.

$l_{i \in n} =$ (object $p1$, relation r, object $p2$, range d)
$L = \{l_1, l_2, .., l_n\}$

In general, for object p_1 and relation type r_j several relations $l_i = (p_1, r_j, p_i, d_i)$ involving different objects p_i may be possible. The relation minimising d_i is called the neighbour relation. Neighbour relations are a specialisation of spatial relations, hence spatial relations form their own general-specific hierarchy. This hierarchy must be considered when performing generalisation and specialisation steps on spatial relations.

- General-specific ordering for spatial relations in sets L_1 and L_2:
  - Iff $L_1 \subseteq L_2 \wedge \forall l_i \in L_1 \geq l_i \in L_2 : L_1 \geq L_2$
- Obtaining spatial relation set L_3 from $L_1 \in h \uparrow L_2 \in e^+$:
  - $L_3 = L_1 \cap L_2, \forall l_i \in L_3 = l_i \in L_1 \uparrow l_i \in L_2$

[2]For a textual representation of spatial relations, arbitrary object indices are kept to disambiguate relational structures

- Obtaining spatial relation set L_3 from $L_1 \in h \downarrow L_2 \in e^-$:
  - $L_3 = L_1, \forall l_i \in L_3 = l_i \in L_1 \downarrow l_i \in L_2$
  - $\forall l_i \notin L_2 : L_3 = L_1 + l_i, d_i = [0..\text{INF}]$[3]

Note that the particular spatial relation type presented here is just one example of how to impose a symbolic relation. Any other finite set of symbolic relations could be treated accordingly. The spatial relation type includes a range attribute to represent the parts distance. In general, a relation can be enhanced with concept attributes of any type, however the specialization methodology needs to be enhanced then, too.

2.3 Dependencies between attributes

After adopting the above attribute types to constitute our concept language we have to consider dependencies between certain attribute types. Obviously the spatial relation attribute depends not only on the presence of a certain spatial relation, but primary on the presence of the parts forming the relation. Thus we have to keep in mind that if we specialize spatial relations we also have to specialize possible composition attributes to keep the resulting concept description coherent, i.e. we have to demand a part to be present if we introduce a spatial relation containing it (otherwise the partonomy attribute type would dominate the spatial relation attribute type).
In addition to the above dependency which is inherent in our selection of attribute types, one might introduce attributes of the above types which are also dependant on each other. A simple example would be introducing a range w for the width and h for the height of an object. If we now introduce a predicate c testing if $h = 2 * w$ holds, we have a dependency between c and w and h. As long as the dependency is based on a monotonous relation between the attributes (as in the above case) resulting concept descriptions will always be coherent. If we allow predicates imposing non-monotonous relations between attributes we need to track this dependency manually and

[3]The range is opened maximally to satisfy the requirement for minimal specialisation

update dependant predicates if we generalize or specialize attributes on which they depend.

3 Hypothesis selection

After the learning process has been conducted, the boundary sets SB and GB contain the minimally and the maximally generalised concept hypotheses over all training examples. The space of applicable hypotheses VS covers these two boundary sets and the space in between them. In principle, one could use any member of VS as a classifier. To use the whole VS as a classifier, one could employ a voting scheme.
For high-level scene interpretation, however, we are interested in a concise concept description which can be included in the conceptual knowledge base. To achieve this, we have to define additional criteria by which to select a concept description.

3.1 Learning objective

Assuming a set of positive and negative examples as training data we propose the following learning objective:

> For a given set of training examples we want to learn a defined concept description with the most specific representation of attributes descriminating positive from negative examples, and the most general representation of all others.

3.2 Selection methods

Trivially, every hypothesis $h_i \in GB$ satisfies our learning objective. If the set GB has converged to one concept hypothesis, this hypothesis is chosen as concept description for the knowledge base. If GB contains multiple concepts (which is likely considering [15]), we need a selection method to choose a concept hypothesis from GB. But since the hypotheses $h \in GB$ cannot be ordered in a general-specific manner, there is no preference measure to choose

a concept hypothesis that can be derived from our learning objective. Several approaches can be considered to overcome this selection problem:

1. A concept hypothesis can be chosen randomly from GB, as proposed in [16], [17].

2. The minimum amount of attribute specialization can be considered to be the selection criterion. This criterion is not applicable for concept languages with a mixture of symbolic and metric attribute types, because these types cannot be compared with regard to the amount of specialisation.

3. The logical conjunction of all $h_i \in GB$ yields a single hypothesis h_c. Hypothesis h_c is the most general concept hypothesis excluding negative examples through all discriminating attributes. Hypothesis h_c is defined for any state of GB, hence it can be chosen as concept hypothesis.

Since every form of hypothesis selection is in fact a form of biasing, we consider the last approach to be the soundest, because it emphasises all attributes that have been used to discriminate negative examples. Approach 1. and 2. (if applicable) lead to an arbitrary selection of discriminating attributes. A refinement of the selected concept is possible for all above approaches via feedback learning as presented in Section 5.4.

4 Building a Knowledge Base of Ontological Concepts

An ontological concept consists of concept attributes and relations to other concepts. Concept attributes can be represented as shown in 2.2 - more elaborated attribute types can also be employed, as long as a general-specific ordering can be imposed. Basic relations between ontological concepts are compositional relations and taxonomical relations. The composition of concepts is learnt as presented in 2.2.3. Taxonomical relations between concepts can be inferred after learning by applying the general-specific ordering

methodology to concepts. Further symbolic relations between concepts can be represented and learnt analogous to the spatial relation type presented in 2.2.5. Any symbolic relation can be enriched with further attribute types as mentioned above.

So far, we have considered learning and hypothesis selection with the goal of establishing individual concept descriptions. For scene interpretation and many other applications, however, we want to learn a comprehensive knowledge base of ontological concepts. Since our learning approach covers all properties of individual ontological concepts, learning a set of concepts of this type yields a comprehensive knowledge base of concepts with a multitude of possible relations between them. This knowledge base can be exploited through any form of ontology reasoning, which is typically performed using description logics and a constraint system solver.

For our application to the eTRIMS domain, the learnt concepts in the knowledge base are related compositionally, taxonomically and through spatial relations (Table 2). An additional composition attribute is kept to trace transitive compositional relations (e.g. object o_1 is part of o_3 through being part of o_2), which simplifies interpretation. The interpretation process is performed by the SCENIC system.

A basic property of ontological concepts in a knowledge base is disjointness. To test two concepts for disjointness one can simply construct the logical conjunction of these concepts and check the resulting concept for consistency. If the resulting concept description is inconsistent, the basic concepts are disjoint. Disjointness of concepts is important for our approach to concept differentiation, which is presented in the next section.

4.1 Concept differentiation

Learnt concept descriptions must be evaluated with respect to existing concepts. Intuitively, we want to make sure that the conceptual descriptions in the knowledge base do not only reflect arbitrarily chosen positive and negative examples but are also constructed to differentiate between each other. We call this quality criterion *concept differentiation*.

Fortunately, the concept learning process can be controlled to yield a knowledge base of maximally differentiated concepts by selecting training examples in a prudent way. Note that positive examples represent information about intra-concept similarities, whereas negative examples represent information about inter-concept differentiation. Hence to achieve a set of maximally differentiated concept descriptions, one can employ all positive examples of a given concept as negative examples for all other disjoint concepts. Furthermore one can employ any given ontological concept as negative example for any other disjoint ontological concept.[4] Since the learning objective presented above leads us to a selection of a concept hypothesis from the boundary set GB, all inter-concept discriminating attributes introduced through these negative examples will be represented in the selected concept hypothesis. This provides a theoretic foundation for learning a knowledge base of maximally differentiated concept descriptions.

5 Experimental results

5.1 Application to eTRIMS domain

In the context of the eTRIMs project, the concept learning approach presented here has been applied to the domain of terrestrial views of building facades. Typical aggregates of this domain are window arrays (consisting of aligned and regularly spaced windows as parts), balconies (consisting of railing, door and optional windows) or entrances (consisting of a door, stairs and ground).

To conduct the actual training sequence, positive learning examples are extracted from annotated pictures directly. An enriched instance description of the positive example is generated from the information contained in the annotation of the aggregate parts.

We automatically generate negative examples from annotated pictures by selecting random sets of parts. To be precise, we select a negative example N as any set of annotated objects that is not a subset or equal to a positive

[4] A formal description of specialization through concepts is omitted, but very similar to specialization through examples (2.2)

example P in the same picture. This requires, of course, that positive examples are annotated to their maximal extent. Following Winston's insight about "near-miss" examples [18], one can assume a negative example N to be most useful if it differs from a positive example P as little as possible. Hence an ideal negative example differs from a positive example only in one discriminating attribute. This kind of negative example leads to the generation of a most general concept description which is only specialised to exclude the attribute value of the discriminating attribute in the negative example. A straight-forward approach to generate negative examples with near-miss properties is to define a distance metric d for N and P, to randomly generate possible training examples $N_1..N_n$ and finally choose the examples minimizing d. This approach is computationally inexpensive as the distance measure from N_i to P is available at nearly no cost (compared to the training procedure). Hence a large number of negative examples can be evaluated for their near-miss properties.
For a typical training sequence, about 10 to 15 positive examples are used. Since negative examples have stronger concept differentiation qualities, we apply about 100 to 300, keeping a ratio between positive and negative examples of 1/10 to 1/20.

5.2 Example concept

As an example result of the learning process we present the General Boundary conjunction hypothesis for the aggregate "Window Array", learnt from 13 annotated positive examples and 260 generated negative examples:

Size and configuration

Aggregate Width =	]549..INF] cm
Aggregate Height =	[0..208[cm
Parts Width =	[0..INF] cm
Parts Height =	[0..INF] cm
Parts Top-Left-X Variance =	]131..INF] cm
Parts Top-Left-Y Variance =	[0..33[cm
Parts Bottom-Right-X Variance =	]115..INF] cm
Parts Bottom-Right-Y Variance =	[0..9[cm

Composition

Has-Parts =	[3..INF]
window =	[3..INF]
door =	[0..0]
Part-Of =	[1..1]
facade =	[0..1]
roof =	[0..1]

Symbolic attributes

Shape =	{ Protracted-X }

Attribute predicates

Fuzzy-Equal	(top-left-y)
Fuzzy-Equal	(bottom-right-y)
Fuzzy-Equal	(parts-height)
Fuzzy-Equal	(parts-distance-x)
Value-Equal	(parts-type)

Internal spatial relations

(window000) LeftNeighbourOf [132..324] cm (window001)
(window000) LeftOf [339..649] cm (window002)
(window001) LeftNeighbourOf [206..325] cm (window002)
(window001) RightNeighbourOf [132..324] cm (window000)
(window002) RightNeighbourOf [206..325] cm (window001)
(window002) RightOf [339..649] cm (window000)

External spatial relations

(concept013) BelowOf [44..1865] cm (sky020)
(sky020) AboveOf [44..1865] cm (concept013)

Table 2: Learnt aggregate description "Window Array"

5.3 Evaluation

A simple kind of evaluation is to test learnt concepts on instances from annotated images which have not been used for training (Table 3). This method evaluates how many false negative detections occur. An evaluation of false

Aggregate	Instances	Detections	Success Ratio
"window array"	18	18	1
"balcony"	14	13	0.93
"entrance"	9	7	0.78

Table 3: Preliminary evaluation results

positive detections is not possible this way, but will be performed on annotated instances of disjoint concepts.

Ontological concepts can also be evaluated using the interpretation facilities of the SCENIC system. The scene interpretation process is based on the hypothesise-and-test paradigm. Hypotheses are posed mainly through part-whole-reasoning, which emphasises the role of conceptual aggregate descriptions. Fig. 2 shows the result of an interpretation process applying the learnt concept description in Table 2 to an image, where scene objects have been automatically detected ([19], [20]). The interpretation system tries to interpret the scene by finding object aggregations based on the detected scene objects and the ontological aggregate descriptions in the knowledge base. SCENIC infers four instances of the window array concept and poses four additional window hypotheses:

Figure 2: Detected scene objects and SCENIC interpretation result

5.4 Feedback learning

Refinement of learnt concepts is possible through an interpretation process applying them to annotated pictures which have not been used for training. The results of the interpretation process are automatically evaluated against the ground truth given by the annotation.
There are two possible cases of misinterpretation:

- An annotated instance of the concept to be evaluated is not recognised.
- A set of annotated objects is wrongly interpreted as an instance of the aggregate concept to be evaluated.

Both cases may lead to an automatic feedback learning step. In the case of false negative recognition (i.e. an annotated instance is not recognized based on the learnt concept), the learnt concept description is too specific. As a feedback step, the unrecognised aggregate instance is introduced to the learning module as a positive example, generalising the learnt concept description. In the case of false positive recognition (i.e. a set of annotated objects is wrongly interpreted as an instance of the evaluated concept), the learnt concept description is too general. As a feedback step, this set of annotated objects is introduced to the learning module as a negative example, specialising the learnt concept description. For both cases another misclassification of this instance becomes impossible, regardless which concept hypothesis is chosen from VS after feedback learning.

6 Conclusions

We have shown that conceptual descriptions for real-world knowledge-based scene interpretation can be obtained by Version Space Learning. A concept description language has been presented which allows to express quantitative as well as qualitative attributes and relations suitable for the descriptions of ontological concepts. Novel results also pertain to concept selection and concept differentiation for a conceptual knowledge base. Version Space

Learning can be used to obtain maximally differentiated concepts. The success of learning has been demonstrated by scene interpretation experiments employing the learnt concepts. By making use of annotated images, an automatic feedback learning cycle can be entered where wrong interpretations serve as correcting examples. An extended evaluation using a database of several hundred annotated facade images will be carried out soon.
Version Space Learning is attractive for stepwise extension and refinement of conceptual knowledge bases because individual examples count and mistakes can be easily corrected by feedback learning. As a drawback, standard Version Space Learning is highly sensitive to bad teaching. A single inconsistent example, wrongly annotated as positive or negative, may cause the version space to collapse. Methods for extending Version Space Learning to cope with noisy data are known in the literature (e.g. [9], [11]), and will be exploited for our concept learning approach.
We consider learning the aggregate storey as a future effort, which yields a machine learnt concept description very similar to a basic component of human scene interpretation in this domain. An interesting topic for further research is to use the knowledge about near-miss properties of negative examples to derive an approach to Active Learning. If a model of best near-miss examples can be generated from a given concept description, the learner will be able to actively choose appropriate negative examples to learn a most discriminating concept description. In fact, the learner will transfer his knowledge about intra-concept similarity to derive a model for inter-concept discrimination.

References

[1] T.M. Mitchell, "Version spaces: A candidate elimination approach for rule learning", Proc. of the International Joint Conference on Artificial Intelligence, pp. 305–310, 1977.

[2] T.M. Mitchell, "Version Spaces: An Approach to Concept Learning", PhD thesis, Stanford University, Cambridge, MA, 1978.

[3] B, Neumann, "A Conceptual Framework for High-level Vision", Technical report FBI-HH-B-241/02, Universität Hamburg, 2002.

[4] K. Sage, J. Howell, H. Buxton, "Recognition of Action", Activity and Behaviour in the ActIPret Project, Künstliche Intelligenz, 2/2005, BöttcherIT Verlag, Bremen, pp. 30–33.

[5] K. Murphy, A. Torralba, and W. T. Freeman, "Using the forest to see the trees: A graphical model relating features, objects, and scenes", Proc. of Neural Information Processing Systems, 2003.

[6] M. Boutell, J. Luo, "Scene parsing using region-based generative models", IEEE Transactions on Multimedia 9(1), pp. 136-146, 2007.

[7] L. Hotz, B. Neumann, "Scene Interpretation as a Configuration Task", Künstliche Intelligenz, 3/2005, BöttcherIT Verlag, Bremen, pp. 59–65.

[8] H. Hirsh, "Polynomial-Time Learning with Version Spaces", National Conference on Artificial Intelligence, pp. 117–122, 1992.

[9] H. Hirsh, N. Mishra, L. Pitt, "Version Spaces without boundary sets", Proc. AAAI-97, pp. 491-496, 1997.

[10] M. Sebag, "Using Constraints to Building Version Spaces", Proc. of the 7th European Conference on Machine Learning, pp. 257-271, 1994.

[11] T.-P. Hong, S.-S. Tseng, "A Generalized Version Space Learning Algorithm for Noisy and Uncertain Data", IEEE Transactions on Knowledge And Data Engineering, Vol. 9, No. 2, 1997.

[12] L. De Raedt, S. Kramer, "The Levelwise Version Space Algorithm and its Application to Molecular Fragment Finding", Proc. of the International Joint Conference on Artificial Intelligence, pp. 853–862, 2001.

[13] R.S. Michalski, "A Theory and Methodology of Inductive Learning", Machine Learning - An Artificial Intelligence Approach, pp. 83–143, 1983.

[14] B. Ganter, R. Wille, "Formal Concept Analysis - Mathematical Foundations", Springer Verlag, 1999.

[15] D. Haussler, "Quantifying inductive bias: AI learning algorithms and Valiant's learning framework", Artificial Intelligence 36, pp. 177–221, 1988.

[16] S.W. Norton, H. Hirsh, "Classifier learning from noisy data as reasoning under uncertainty", Proc. of the National Conference on Artificial Intelligence, 1992.

[17] S.W. Norton, H. Hirsh, "Learning DNF via probabilistic evidence combination", Machine Learning: Proc. of the Seventh Internation Conference, 1990.

[18] P.H. Winston, "Learning structural descriptions from examples", The psychology of computer vision, pp. 157–209, 1975.

[19] J. Šochman, J. Matas, "WaldBoost - Learning for Time Constrained Sequential Detection", Proc. of the Conference on Computer Vision and Pattern Recognition, pp. 150–157, 2005.

[20] L. Hotz, B. Neumann, K. Terzic, J. Šochman, "Feedback between Low-Level and High-Level Image Processing", eTRIMS Project Deliverable D2.4, 2007.