Which tool to use? Grounded reasoning in everyday environments with assistant robots

Lydia Fischer[1] Stephan Hasler[1] Jörg Deigmöller[1] Thomas Schnürer[2]
Michael Redert[3] Ulrike Pluntke[3] Katrin Nagel[3] Chris Senzel[3]
Joern Ploennigs[4] Andreas Richter[1] Julian Eggert[1]

1 – Honda Research Institute EU, Offenbach, Germany, firstname.surname@honda-ri.de
2 – Technical University of Illmenau, Illmenau, Germany, thomas.schnuerer@tu-illmenau.de
3 – IBM Watson IoT Center, Munich, Germany, firstname.surname@de.ibm.com
4 – IBM Research, Ireland, firstname.surname@ie.ibm.com

Abstract

We present a cooperative reasoning agent embodied into a mobile robot enabled to explore its environment by a camera. The robot can infer missing knowledge given an action and an object by the user, like for example "*I want to open a wine bottle.*". Available tools are explored by the robot and it finally recommends the most suitable one. The reasoning is on the one hand based on a static ontology that describes how to relate the actions "*open*" and "*cut*" with a fixed set of tools. On the other hand, unknown actions and tools are resolved by looking up for synonyms or super/sub-class relations in Wikidata and WordNet. By this, the robot tries to map knowledge from linked data to its internal interpretable ontology. To retrace the reasoning process, the robot is able to explain its conclusions by text to speech. Finally, we show the performance of the system based on different settings for scanning the linked data.

1 Introduction

Cooperative Intelligence (CI) is an approach which takes up ideas from artificial intelligence and machine learning but puts the human in the center of all considerations. One of the prerequisites of CI is that for seamless interaction, an artificial agent and a human need to share concepts about the things that happen in their environment [Rebhan et al., 2009a, Rebhan et al., 2009b]. A viable approach for this is to use semantic knowledge in terms of ontologies. Problems here lie in the generalizability of the approach and in the grounding [Harnad, 1990] of the semantic concepts in the real-world. Besides the pure functionality of such a system the capability of making the internal processes transparent is important for user acceptance and trust [Hancock et al., 2011]. To tackle this problem, in a joint project between HRI-EU and IBM Watson IoT Center, we have studied a minimal real-world set-up in which a robot interacts with a user to reason about tools, objects and actions. The scenario was limited to a small set of actions like e. g. cutting and opening, which are common all day activities. However it was designed to be expandable to a broader set of involved elements and actions. In this paper, we present a system consisting of a robotics platform provided with a scenario-specific root ontology, dialog, reasoning, external knowledge search, object detection, and symbol grounding capabilities. The ontology provides seed knowledge about a small set of concepts, and we show the system capabilities of reasoning about tool-related queries within the ontology and additionally making use of external knowledge sources to extend its reasoning scope in case of "unknown" concepts. The analysed system is capable of

- interacting with the user via speech,
- processing user requests within the designed scope of the system using an internal static ontology and reasoning,
- providing a log file that contains the internal steps of the reasoning as well as an explanation via speech,

In: G. Steinbauer, A. Ferrein (eds.): Proceedings of the 11th International Workshop on Cognitive Robotics, Tempe, AZ, USA, 27-Oct-2018, published at http://ceur-ws.org

- connecting knowledge from external sources with the real world, and
- operating in the real world.

In the following we present a proof of concept framework that allows for future extensions. We have chosen a limited scenario that should show the main idea of dealing with external knowledge sources for a robot.

The scenario: Tools, objects, and actions

In our real world scenario, we show a mobile robot that is acting intelligently and able to reason by taking into account the current situation and the knowledge sources of the system. In the future, robots will assist humans in many tasks in their household or at work. We consider the tasks of cutting and opening objects which are typical in all day activities. These tasks are easy to understand and offer already enough complexity to show the challenges and solutions in reasoning.

In the beginning of the demonstration, a user enters the scene and sets a task like *"I want to cut a wooden block."*. The system shall suggest a suited tool with respect to the available tools in the current scene. The robot IRA (Intelligent Reasoning Agent) is placed in a fixed and static environment. In our demonstration, we set up a room with a table that is placed in the middle of the room (Figure 1). On the table there are tools

Figure 1: Illustration of the scenario. The robot IRA and the user are in a room. The user can ask IRA how to manipulate an object. IRA explores the table, detects the tools and proposes a suitable tool via speech.

which are available for the task. We assume that the tools do not change and are not moved while exploring.

In case the user states a request which cannot be answered using the predefined ontology, e. g. *"I want to cube an onion."* (the action to 'cube' and the object 'onion' is unknown by the system), the system uses external knowledge sources in order to match the unknown terms with related terms known in the internal ontology. Hence, our system is able to work beyond the designed task scope with this mechanism.

One run of the scenario contains five main steps:

1. User sets a task via speech interaction
2. IRA provides a first suggestion by reasoning in its internal ontology (and external knowledge sources if required)
3. IRA explores the table with the available tools and recognizes them visually
4. IRA suggests the most suited tool that is available on the table for the given task and points to it
5. IRA provides an explanation of its decision

The subsequent section provides related work followed by a description of the system architecture. Afterwards the functionality of single system components are explained. Then the results of the quantitative tests are described. The conclusion completes the paper.

2 Related work

Most work in robotics in the area of reasoning focuses on detailed execution of manipulation tasks and robot planning. For instance, [Hogg et al., 2017, Tenorth and Beetz, 2013] and [Haidu and Beetz, 2016] focus on learning tasks from observing human activities in kitchen scenarios. In the area of robot planning, the reasoning usually deals with uncertainties from real world measurements [Hanheide et al., 2017].

Only little work exists in covering more broader everyday knowledge for robots like usual relations between objects and properties to allow a more natural and less targeted interaction. Early work tried to build own crowd sourced databases [Gupta and Kochenderfer, 2004]. Later on, more extensive databases emerged from the linked data wave [Antoniou et al., 2012, Wood et al., 2014]. [Daoutis et al., 2009] used the common sense database Cyc [Matuszek et al., 2006] for communicating with the robot about objects and their properties that have been seen. [Kaiser et al., 2014] crawl text documents that fit to the current context, build their own ontology and improve the content by linking against WordNet [Stallman and Krempl, 2010].

One major problem in making data interpretable for robots is the type of representation. In [Pustejovsky and Krishnaswamy, 2016], a semantic knowledge representation is defined that describes real-world objects from a functional point of view, which might be beneficial for robotics.

Still, the previous approaches lack in judging if an observed object by the robot is usable for a certain task. This relates to the problem that linked data is not directly machine interpretable, since the naming of symbols in the ontology depends on the designers choice. Hence, our approach differs as we map linked data onto an internal, machine-interpretable ontology that represents a known concept to the robot. The advantage is that by doing this we are able to evaluate

if external knowledge sources are usable for the robot.

3 The system architecture

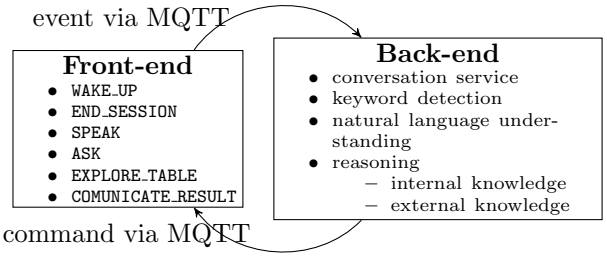

Figure 2: The system overview. The back-end controls the dialog and applies modules of the front-end to communicate with the user or to explore the environment. Commands/ events from and to the front-end are sent by MQTT.

The system consists of two main parts (Figure 2): the front-end (interaction platform) and the back-end (dialog management and reasoning). We assume a system with a predefined static ontology about objects and tool types as well as relations between them with respect to the actions cut and open, e. g. a wooden block can be cut by a wood saw. The platform of the system is a MetraLabs SCITOS G5 robot [MetraLabs, 2018], additionally equipped with a Kinect camera mounted on a pan-tilt-unit for moving the head. Laser scanners allow the robot to localize itself in the room and a KINOVA JACO arm [Kinovarobotics, 2018] enables the robot to point at tools and objects in the environment.

The front-end provides basic modules for the robot like speaking or listening. Those modules are running in ROS (Robot Operating System [ROS, 2018]). The back-end comprises mainly the dialog control and the reasoning. Further it acts as a state machine that decides for example if the robot needs to explore the environment for available tools.

4 Front-end

The front-end provides modular components that can be independently triggered by the back-end through a MQTT (Message Queue Telemetry Transport, [ISO/IEC, 2016]) bridge. The MQTT bridge translates messages from the back-end to ROS messages of the front-end and vice versa. As depicted in Figure 2, the front-end modules are limited to `SPEAK`, `ASK`, `EXPLORE_TABLE`, and `COMMUNICATE_RESULT`. `WAKE_UP` and `END_SESSION` only send unidirectional messages for starting and stopping the dialog. The modules are explained in the following:

`SPEAK` receives a text message from the back-end, transfers it to speech and returns if the robot has finished speaking.

`ASK` works similar as `SPEAK`, as it is a combination of `SPEAK` at first - for asking a question - with a subsequent listening function. It finally returns the answer of the user as text to the back-end.

`EXPLORE_TABLE` is the most complex module. If triggered, it moves the robot around the table while keeping the camera view on the table. The robot stops approximately every 0.5 m on the trajectory and captures an RGB and a depth image from the Kinect camera. For each RGB image the objectness score of the YOLO detection framework [Redmon and Farhadi, 2017] is used to predict bounding boxes of object candidates (see Figure 3). For this the pre-trained YOLO model for the COCO dataset is used, which yields an 81-dimensional class vector for each bounding box. This vector is the feature input to a linear SVM [Cortes and Vapnik, 1995] that is trained to predict the final tool class together with a confidence. Using

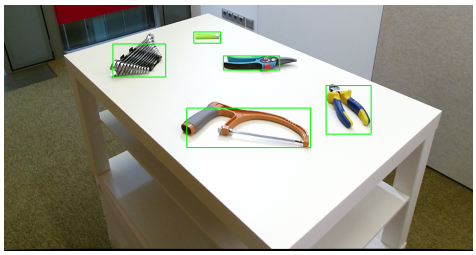

Figure 3: Robot camera view with predicted bounding boxes.

the depth information, a 3D world position is computed for each object detection. After all views have been taken, nearby 3D detections are clustered using DBSCAN [Ester et al., 1996]. Each cluster is assigned with a unique instance ID, a label, and a confidence. The label indicates the tool class with the highest accumulated confidence, and the confidence of the cluster equals the ratio between the accumulated confidence of the winning class and the sum of all detection confidences (Figure 4). This integration step strongly reduces the effects of noisy 3D estimations and wrong class predictions from certain view-points. Finally the cluster labels with their confidences are sent to the back-end. We expect that the single view detection performance can be strongly improved by directly fine-tuning YOLO on our data. Having this, a more efficient exploration strategy of the robot can be implemented, e. g. like in [Andreopoulos et al., 2011] where occlusions of objects on a table are taken into account. This module relates to step three of a demonstration run explained at the end of the Introduction.

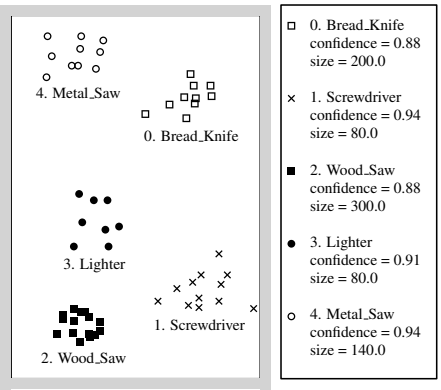

Figure 4: Integrated object detection result.

COMMUNICATE_RESULT receives the inferred information from the back-end which tool is the most suitable one for the given query. This contains the instance ID of the tool and the text for speaking. The text is forwarded to the SPEAK module and the instance ID is used to point at the tool on the table with the robot arm (step 4).

5 Back-end

The complete dialog between the user and IRA is controlled by the back-end. In addition, keyword detection based on natural language understanding techniques extract the action and the object from tasks stated by the user.

The control module of the back-end keeps track of the current state of the dialog and triggers the needed commands for the robot. It also provides the exchange of information between the back-end components. The main intelligence of the system is contained in the reasoning module that is explained in more detail later.

The architecture of the back-end (Figure 5) follows the micro service architectural pattern. It has been implemented on the IBM Cloud using multiple IBM Watson services. Connectivity between the robot and the back-end is established through the Watson Internet of Things Platform using the MQTT protocol. The back-end services can be organized in three layers, which are briefly described below.

Natural language understanding: The components within this layer are responsible to drive the dialog with the user. If an utterance is received by the Dialog Controller, it decides, which of potentially several skills is best suited to handle it. In case, none fits well, the Default Skill creates a response that tells the user that the system cannot handle the request. If the utterance is related to the described scenario, the Cut & Open Skill will produce a response. In order to do so, it first analyses the intent of the utterance and extracts relevant information like the action and the object using the Keyword Extraction component. In

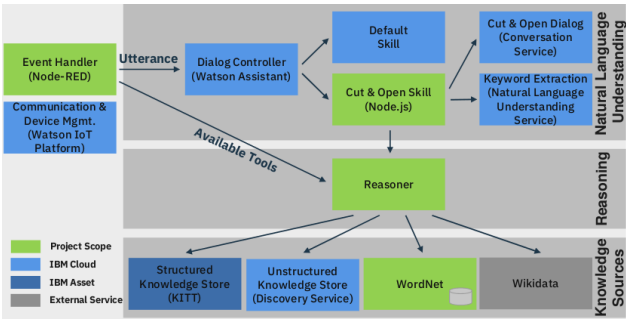

Figure 5: Back-end architecture

case action and object are identified, the reasoning is performed. The result is then translated into natural language using the Cut & Open Dialog component. This is also involved, if the system must ask for missing information or explains the result.

Structured reasoning: This layer contains the main reasoning component, which provides a structured API for concept-based as well as instance-based reasoning. It uses knowledge sources from the layer below. Details of both algorithms are described in the next section of the paper.

Knowledge sources: The components of this layer provide information for the reasoning component. Most prominent is the Structured Knowledge Store component, which contains structured domain knowledge (ontology) about the use case scenario. In order to enhance manually added knowledge, it applies reasoning rules that generate additional knowledge. The Unstructured Knowledge Store is considered as a fallback that returns a paragraph of text as description in case no specific tool can be recommended. WordNet and Wikidata wrap the corresponding knowledge sources.

WordNet [Stallman and Krempl, 2010] is a lexical database of English containing nouns, verbs, adjectives and adverbs. Words are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. This knowledge is helpful in resolving the meaning of unknown words. Wikidata [Vrandečić and Krötzsch, 2014] is an open source knowledge database containing factual knowledge extracted for instance from Wikipedia. Both sources are selected because of their complementary content.

The reasoning:

The reasoning components are the pre-defined static ontology and reasoning rules on top of it, the interfaces to Wikidata and WordNet. The static ontology contains

- objects ($\approx$ 30), e. g. apple, glass, rope
- tools ($\approx$ 20), e. g. bread knife, corkscrew, wood saw

- concepts (≈ 15), e. g. vegetable, vessel, plant
- relations (`hasComponent`, `cutBy`, `cutByPref`, `openBy`, `openByPref`)
- the internal structure encodes an `isA` relation of objects, tools, and concepts, e. g. an apple `isA` fruit

The objects and the tools are attached with the properties: stability $\mathcal{S}$ and their 3D shape. The 3D shape is described in the dimensions width $\mathcal{W}$, height $\mathcal{H}$ and length $\mathcal{L}$. These qualitative properties are used for reasoning. The stability $\mathcal{S}$ of a tool is specified in a range from completely deformable to completely stiff. Currently, the shape of a tool is not measured online but is a static property stored in the ontology. Later on it is planned to use the visual input of IRA to measure the length of the recognized tools. For reasoning within cutting tasks there are two main reasoning rules using this properties:

$$\text{(i) } \mathcal{S}_{\text{object}} \leq \mathcal{S}_{\text{tool}}\,, \text{and (ii) } \mathcal{W}_{\text{object}} \leq 2 \cdot \mathcal{L}_{\text{tool}}\,.$$

For a task within opening scenarios there is a special reasoning rule in addition to the direct links that says, if a vessel v has a fastening f, and this fastening f can be opened by a tool t, then vessel v can also be opened by tool t:

$$\texttt{hasComponent}(v, f)\,\&\,\texttt{openBy}(f, t) \rightarrow \texttt{openBy}(v, t)\,.$$

The reasoning consists of two steps: concept-based reasoning, and instance-based reasoning. The concept-based reasoning uses the internal ontology and the external knowledge without reference to the tools on the table. Hence the system provides a first suggestion of a tool, as mentioned in the second step of the scenario description. Figure 6 shows a visualisation of the relations of a wooden block in the internal ontology which represents the main structure of objects and tools. For the task *"I want to cut a wooden block."*

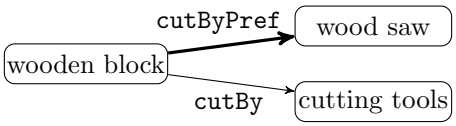

Figure 6: Instance of the relations attached to an object

the first suggestion would be a wood saw since it is connected via a `cutByPref` relation. The suffix `Pref` indicates that a tool is always the preferred one given a related tool. If such a relation is missing, an alternative tool that is connected with the object via the non preference relations `cutBy`, `openBy` is suggested. In the example of the object wooden block (Figure 6) all cutting tools (e. g. metal saw, pocket knife) are connected with the wooden block and hence they are alternative tools to cut it. Of course this holds for tools only that full-fill the constraints for cutting respectively opening the objects. To make the internal process of the reasoning transparent for the user there is a so-called reasoning log stating single steps of the reasoning process. It is easily accessible in real-time on a web page. An example for a concept-based reasoning log can be found in Figure 7.

- Checking ontology...
- The action 'to cut' is known.
- The object 'Wooden_Block' is known.
- Best suited tool to cut a Wooden_Block: Wood_Saw
- Unknown if it is available, hence exploring the table.

Figure 7: Example of a concept-based reasoning log for the task *"I want to cut a wooden block."*.

After the first reasoning step, the robot explores the available tools on the table. Subsequently, the instance-based reasoning takes place. The recognized tools together with their recognition confidence are taken into account in addition to the internal and external knowledge. The differences to the concept-based reasoning are that only the available tools are considered including their confidence values and that the reasoning rules mentioned above are applied. With this additional information, the system is able to suggest an available tool in the current scene that fits the task with a high certainty. Figure 8 shows an example of instance-based reasoning.

- On the table there are best suited tools to cut a Wooden_Block: Wood_Saw (id_1)
- The best suited tool with the highest recognition confidence is the Wood_Saw (id_1).
- The recognition confidence (0.95) is above a defined threshold (0.80).
- Recommendation: Wood_Saw (id_1)

Figure 8: Example of an instance-based reasoning log for the task *"I want to cut a wooden block."*

Now, let us assume that the given task consists of known actions (cut and open) and objects that are contained in the ontology, we refer to this kind of tasks as *standard tasks*. For standard tasks the reasoning process is described above, external knowledge is not required. As an extension assume that at least the action or the object, or both terms are unknown. In this case the internal knowledge is unable to provide an adequate suggestion because of lacking information.

Using external knowledge sources to find a *proper match* of the unknown terms to terms known in the system is the first step before the reasoning steps of the standard tasks are performed. We refer to this kind of

tasks as *non standard tasks*. Within this paper we define a match of two terms (unknown term, term known in the ontology) is proper when the semantic meaning (linguistic) fits. In other words, a proper match is a path in the external knowledge source that connects an unknown term with a term known in the internal ontology. For example proper matches of actions are (cube, cut) and (uncork, open) and on the other hand for objects (meat, food), and (rose, plant). From WordNet we only search for synonym and hypernym links, whereas from Wikidata we search for material used, part of, and subclass relations. The number of allowed links indicating relations is denoted as search depth. A proper path is a path of the structure: unknown term (e. g. cube) – link-type (e. g. is synonym) – known term in the ontology (e. g. cut). Such a path can possibly have more links and unknown terms in-between. The important part is that the path connects the unknown term via allowed links and potentially other terms with a term known in the ontology. In order to choose the most reliable path out of possibly many paths ending in different known terms, we introduce a scoring in three steps. First, each existing link l_i is scored with 1:

$$l_i = \begin{cases} 1, & \text{if link exists} \\ 0, & \text{if no link exists} \end{cases}$$

Second, a path into the ontology has a weight p_j which is based on the number of links:

$$p_j = \prod_{i=1}^{n} (0.9 \cdot l_i)$$

The factor 0.9 penalizes long paths, since we assume the longer the path the more the uncertainty increases.

The maximum score p_{opt} of all m path scores defines the optimal path. It describes the "properness" of a match between two terms (unknown term, known term):

$$p_{\text{opt}} = \max(p_1, ..., p_m)$$

Methods for link and path scoring as well as the selection of the best result are subject to future research. For example, the method for scoring a link might use the number of links of the same type starting from the same object. Further the types of links, e. g. made of, can be scored differently.

Figure 9 shows an excerpt of a dialog that highlights the reasoning of IRA if it is asked for "cube an onion". This example contains an unknown action (cube in the sense of cutting) and an unknown object (onion). If IRA is asked to explain its reasoning it describes the relations to an external knowledge source based on the mentioned scoring process.

- User: "I want to cube an onion"
- IRA: "To cube is like to cut. Onion is a kind of vegetable. The best tool would be a Fruit_Knife. Let me check the available tools."
- IRA: "A Fruit_Knife is available, I would recommend to use it."
- User: "Please explain."
- IRA: "According to Wikidata a vegetable is a superclass of an onion. According to WordNet to cut is a hypernym of to cube. A Fruit_Knife is the preferred tool to cut a vegetable. This tool is available on the table. Is there anything else you want to do?"

Figure 9: Part of the dialog if action (cube) and object (onion) are unknown.

6 Performance analysis of the system

In the first part we evaluate if the suggested tool matches the task. For instance for the unknown terms *"cube an onion"* a suited tool would be a knife. This would count as a successful suggestion. An example for a failure would be the suggestion to use a wood saw for *"carving wood"* (unknown term). The suggested tool is reasonable but of course it would be very difficult to carve wood with a saw. In the second part of this section we evaluate the performance of our system in finding proper matches for unknown terms to terms in the internal ontology. This mechanism is the basement of the reasoning.

We created a test set that consists in total of 50 test examples. Each example is a triplet (object, action, tool). The objects and actions are taken from 15 unknown actions, 44 unknown objects, and the known objects and actions in the ontology. We consider only known tools for the test examples. The test set consists of 5 examples with only an unknown action, 32 examples with only an unknown object, and 13 examples where both terms are unknown. For a meaningful evaluation, different settings have been applied in the experiments. Within a setting the search depth of the available relations of the external knowledge sources are defined. The search depth is varied from direct links (length one) up to search paths of length six. In the figures 10 and 11, the search depth is varied on the x-axis and the legend denotes the used relation. The term Wikidata refers to the used relations: material used, part of, and subclass. WordNet refers to the relations: synonym and hypernym.

Evaluation of tool suggestion:

Figure 10 shows the evaluation of the defined test cases. It can be seen that single relations like synonym and hypernym contained in WordNet help in resolving few test cases only, independently of the search depth. The

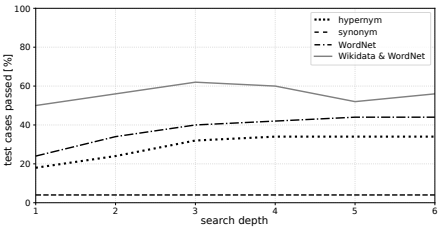

Figure 10: The evaluation of the test cases if accessing external knowledge sources. The performance is shown for different knowledge sources depending on search depth.

combination of both relations is beneficial (WordNet graph) but the best performance occurs when combining WordNet and Wikidata. As mentioned the content of both sources is complementary and it is consequential that their combination is beneficial. The associated graph displays that a search depth of three provides the best results while higher depths leads to a decreasing performance. In total the system is able to recommend suited tools for 35 examples while 15 failed. The main reason is missing information such that there is no possibility to find a proper match for at least one unknown term. Another reason of failing is the mismatch of the suggested tool with the expected one in the test case triplet. An example for this case is (wooden block, carve (unknown), pocket knife). Here the system suggests the wood saw but of course carving wood with a wood saw is difficult.

Subsequently we evaluate more deeply the performance of resolving unknown actions and objects.

Evaluation of proper object matches:

To evaluate the performance regarding proper object matches using Wikidata and WordNet, we analyzed the true positive rates and false positive rates (Figure 11). The true positive rate reports the number of proper matches divided by the overall number of unknown objects from our test set. The false positive rate denotes the number of mismatches, that means matches into the ontology which are semantically wrong, again divided by the overall number of unknown objects. The displayed positive rate shows the sum of both rates. From Figure 11, we derive that a search depth of three provides a good balance between true positives (67,3 %) and false positives (19 %). This is in accordance with observations from Figure 10 and holds especially for using WordNet and Wikidata. As can be seen from the positive rate, the system finds matches into the ontology for 95.45 % of the unknown objects. During our experiments it turned out, that WordNet is more suited than Wikidata for resolving lacking information of unknown actions due to its lexical content. Using the relations synonym or hypernym from WordNet, the

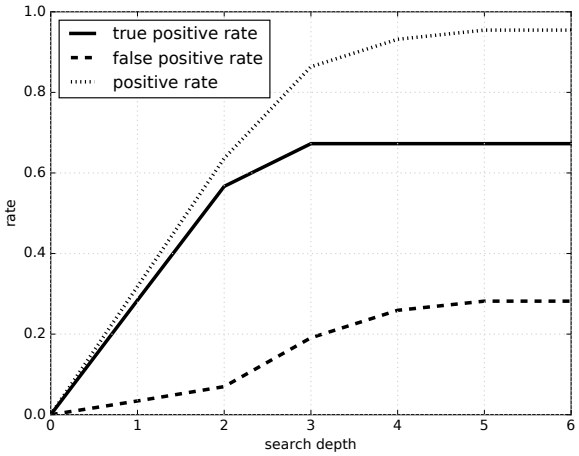

Figure 11: The true positive rate, false positive rate, and the positive rate for the unknown objects in the test set using Wikidata and WordNet.

system finds proper matches for 75 % of the unknown actions of the test set.

In summary, the most promising setting is the combination of sources with complementary content (in our case WordNet and Wikidata). There are many more sources available [Paulheim, 2017] possibly useful for intelligent systems.

7 Conclusions

We proposed an intelligent system that is able to provide an answer for a stated task by the user. The answer is retrieved through reasoning taking an internal ontology, external knowledge sources and the explored resources in the current scene into account. The reasoning takes place in order to suggest a suited tool for a task like *"I want to cut a wooden block."* that contains an action and an object. For unknown actions and objects, external knowledge sources are used to map these terms onto known terms into the ontology. For our defined test cases it turned out that the combination of complementary external knowledge sources is beneficial and a promising direction for future research. The system shows on a web page a reasoning log containing the internal steps of the reasoning process and an explanation of the found suggestion via speech. This mechanism makes the system transparent and nicer to engage with.

In future, we would like to extend the scenario to more actions and objects to allow a more natural and comprehensive interaction with the robot.

Acknowledgements

The authors would like to thank all the people that supported this project at both companies. We thank MetraLabs for the set-up and support of the robot.

References

[Andreopoulos et al., 2011] Andreopoulos, A., Hasler, S., Wersing, H., Janßen, H., Tsotsos, J., and Körner, E. (2011). Active 3D Object Localization Using a Humanoid Robot. *IEEE Transactions on Robotics*, 27(1):47–64.

[Antoniou et al., 2012] Antoniou, G., Groth, P., Harmelen, F. v., and Hoekstra, R. (2012). *A Semantic Web Primer*. The MIT Press.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[Daoutis et al., 2009] Daoutis, M., Coradeshi, S., and Loutfi, A. (2009). Grounding Commonsense Knowledge in Intelligent Systems. *Journal of Ambient Intelligence and Smart Environments*, 1(4):311–321.

[Ester et al., 1996] Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231.

[Gupta and Kochenderfer, 2004] Gupta, R. and Kochenderfer, M. J. (2004). Common Sense Data Acquisition for Indoor Mobile Robots. In *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, pages 605–610. AAAI Press.

[Haidu and Beetz, 2016] Haidu, A. and Beetz, M. (2016). Action recognition and interpretation from virtual demonstrations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2833–2838.

[Hancock et al., 2011] Hancock, P. A., Billings, D. R., Schaefer, K. E., Chen, J. Y. C., de Visser, E., and Parasuraman, R. (2011). A meta-analysis of factors affecting trust in human-robot interaction. *Human Factors*, 53(5):517–527.

[Hanheide et al., 2017] Hanheide, M., Göbelbecker, M., Horn, G. S., Pronobis, A., Sjöö, K., Aydemir, A., Jensfelt, P., Gretton, C., Dearden, R., Janicek, M., Zender, H., Kruijff, G.-J., Hawes, N., and Wyatt, J. L. (2017). Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*, 247:119–150.

[Harnad, 1990] Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346.

[Hogg et al., 2017] Hogg, D. C., Alomari, M., Duckworth, P., Hawasly, M., Bore, N., and Cohn, A. G. (2017). Grounding of Human Environments and Activities for Autonomous Robots. In *Proceedings of International Joint Conference of Artificial Intelligence (IJCAI)*, pages 1395–1402.

[ISO/IEC, 2016] ISO/IEC (2016). Information technology – Message Queuing Telemetry Transport (MQTT). Website. `https://www.iso.org/standard/69466.html`.

[Kaiser et al., 2014] Kaiser, P., Lewis, M., Petrick, R. P. A., Asfour, T., and Steedman, M. (2014). Extracting common sense knowledge from text for robot planning. In *2014 IEEE International Conference on Robotics and Automation, ICRA*, pages 3749–3756.

[Kinovarobotics, 2018] Kinovarobotics (2018). Kinova. Website. `www.meetjaco.com/about/`.

[Matuszek et al., 2006] Matuszek, C., Cabral, J., Witbrock, M., and Deoliveira, J. (2006). An introduction to the syntax and content of Cyc. In *Proceedings of the AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, pages 44–49.

[MetraLabs, 2018] MetraLabs (2018). Metralabs. Website. `www.metralabs.com/en/mobile-robot-scitos-g5/`.

[Paulheim, 2017] Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

[Pustejovsky and Krishnaswamy, 2016] Pustejovsky, J. and Krishnaswamy, N. (2016). VoxML: A visualization modeling language. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*.

[Rebhan et al., 2009a] Rebhan, S., Einecke, N., and Eggert, J. (2009a). Consistent modeling of functional dependencies along with world knowledge. In *Proceedings of the International Conference on Cognitive Information Systems Engineering*, pages 341–348.

[Rebhan et al., 2009b] Rebhan, S., Richter, A., and Eggert, J. (2009b). Demand-driven visual information acquisition. In *Computer Vision Systems, 7th International Conference on Computer Vision Systems, ICVS*, pages 124–133.

[Redmon and Farhadi, 2017] Redmon, J. and Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525.

[ROS, 2018] ROS (2018). Robot Operating System. Website. `www.ros.org`.

[Stallman and Krempl, 2010] Stallman, R. M. and Krempl, S. (2010). Princeton university 'About WordNet'. Website. `https://wordnet.princeton.edu/`.

[Tenorth and Beetz, 2013] Tenorth, M. and Beetz, M. (2013). KnowRob: A Knowledge Processing Infrastructure for Cognition-enabled Robots. *International Journal of Robotic Research*, 32-5:566–590.

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

[Wood et al., 2014] Wood, D., Zaidman, M., Ruth, L., and Hausenblas, M. (2014). *Linked Data: Structured Data on the Web*. Manning Publications, Shelter Island, first edition.