EU FP7 CogX
ICT-215181
May 1 2008 (52months)

DR 5.3: Representations of gaps in categorical knowledge

Danijel Skočaj, Matej Kristan, Aleš Leonardis and Sergio Roa

University of Ljubljana, DFKI Saarbrücken
⟨danijel.skocaj@fri.uni-lj.si⟩

Due date of deliverable:	31 July 2010
Actual submission date:	29 July 2009
Lead partner:	UL
Revision:	final
Dissemination level:	PU

A crucial requirement of a system that is supposed to self-extend based on self-understanding is the ability to detect gaps in its knowledge. Once these gaps are detected, it can plan what actions to perform to fill these gaps, and after the required information is obtained, it can extend its current knowledge accordingly. This deliverable addresses the problem of detection of knowledge gaps in categorical knowledge. We present a formal model for detection of knowledge gaps, define its requirements and show how it can be applied in different learning domains. We also address the problem of assessing the uncertainty and predictability of specific action consequences in the active affordance learning scenario, which is highly related to the detection of gaps in knowledge.

Executive Summary

Workpackage 5 addresses the problem of active continuous learning of cross-modal concepts. Active learning requires that the system identifies learning opportunities. This in turn requires that it must be able to represent and identify gaps in its categorical knowledge, which may indicate good learning opportunities. In this deliverable we focus on this problem. We first try to identify several types of knowledge gaps that arose in a scenario where the robot is acquiring categorical knowledge in an interaction with a tutor. We then formalise our approach and derive the method for knowledge gap detection. We also specify the requirements of our method and show how it can be used in various learning domains.

The active learning paradigm plays an even more important role in the case of learning object affordances and action effects. In this case, the robot interacts directly with objects in its environment and can plan further actions based on previous observations and detected uncertainty. Therefore, we also address the problem of knowledge gap detection in such situations.

Identification of incompleteness in knowledge is an integral part of the interactive continuous learning process, which also includes perception, learning, planning, and interaction. In general, it is very difficult to address these problems in isolation, since they are very intertwined and inter-dependent. Therefore, many of the principles presented in this deliverable are (or can be) used in the methods presented in some other deliverables and, vice versa, some of the approaches presented in those works are very related to the approaches presented here. These related deliverables include (at least): DR.5.1 Continuous learning of basic visual concepts, DR.5.2 Continuous learning of cross-modal concepts, DR.1.2 Unifying representations of beliefs about beliefs and knowledge producing actions, and DR.2.3 Representations of gaps in knowledge about objects.

The detection of knowledge gaps (and acting accordingly in order to fill these gaps) is an essential part of the project and WP5, therefore this problem has often been addressed in our work. The work has been mainly performed as envisioned in the workplan. We will continue our work on this problem in the future as well, building on the work we present in this deliverable.

Role of Representations of gaps in categorical knowledge in CogX

The main research topic, which is addressed in this deliverable, is the detection of gaps in categorical knowledge. The robot first has to understand what it does and does not know and then has to plan and perform actions that provide information to fill this gap. This research topic is therefore in the very core of the project whose main goal is to produce a system that is able to self-understand to be able to self-extend.

Contribution to the CogX scenarios and prototypes

Detection of gaps in categorical knowledge is one of the most important features we want to demonstrate in the George scenario (Interactive cross-modal learning scenario) [19, 21]. The formal method for identification of knowledge gaps presented in this paper is therefore integrated in the George system and is used for driving the system behaviour. The George system is capable of mixed initiative dialogue with the main goal of learning about objects and object properties. The detected knowledge gaps serve as a main motivation for the robot to take the initiative and ask the tutor for help.

Representing uncertainty and predictability in action effects are important aspects in the Dexter scenario. Skills in manipulation are to be represented by using qualitative models that can be used to assess these aspects. Models of object behaviour and motoric proprioception are combined within a cross-modal setting where this sensorimotor loop is key for the acquisition of skills and prediction abilities of a robot.

1 Tasks, objectives, results

1.1 Planned work

This deliverable mainly tackles the problems addressed in Task 5.5 of Workpackage 5:

> *Task 5.5: Representations of gaps in categorical knowledge. Investigate how a system can exhibit a certain level of self-understanding and self-criticism to detect the gaps in its knowledge and how to represent these beliefs about beliefs of cross-modal categorical knowledge.*

This task will be active throughout the entire duration of the project. This deliverable presents the work on this topic performed in the first two years of the project.

Our main objective was to specify the types of knowledge gaps we are facing and to define a methodology for their identification that could be used in the system we have been developing. The main research topic in WP5 has been interactive continuous learning of cross-modal concepts, with the focus on developing methods and mechanisms for learning in interaction with a tutor. Therefore we planned to develop a knowledge gap detection mechanism for such kinds of learning of categorical knowledge. However, the mechanism was supposed to be general enough to tackle this task in the case of affordance learning and related problems.

We also planned to study the problem of affordance learning from the point of view of analysis of dynamical systems by using probabilistic methods. Specifically, we planned to use the CrySSMEx algorithm[7], which applies an information-theoretic based method as a measure of uncertainty in the causal relationships that are produced through the evolution of the dynamical system. This method would be useful in the future to identify situations of uncertainty and predictability for specific actions. Moreover, the probabilistic models that are obtained from data sequences are basically stochastic automata that can infer a discretization of the sensorimotor spaces and the probabilistic transitions between the discrete states. In this way, these graphical models might be useful for knowledge gap representations by using information-theoretic strategies.

1.2 Actual work performed

In this section we briefly describe the main achievements related to the topic of this deliverable. For detailed descriptions of the work performed the reader is referred to the papers attached in the annex of this deliverable. Furthermore, since the detection of knowledge gaps is an integral part of an active learning approach, different aspects of the problem addressed in this deliverable are also present in several other deliverables and published papers [9, 20, 18, 19, 22, 21, 14]. Here we only present the work that focuses on the problem of knowledge gap detection.

The main contribution in this deliverable is the technical report about the principles of the detection of gaps in the categorical knowledge (Annex 2.1 [10]). We will describe how we detect the knowledge gaps, how we represent them and how we plan to fill them. First we describe these concepts conceptually, then we define them formally and finally, we show how these concepts can be used in different learning domains.

We divide *knowledge gap detection* into two groups based on the time, space and motivation for detecting the gap; either the lack of knowledge is related to a particular situation or not. In the first case, the lack of knowledge is *situational*; it is related to a particular situation or a particular task the robot has to perform, however due to the lack of knowledge, it fails. Here, this process can be triggered by a human (or a task given by the human), or the robot can be self-motivated to explore the current situation to learn something new. For instance, the robot fails to recognise an object in the scene; this indicates that the robot has not yet learned a representation of that particular object, or that the learned representation is not reliable enough, which indicates the incompleteness of the robot's knowledge about the object.

As well as that, knowledge gap detection might not be related to any currently perceived information; it can be achieved by *introspection*- In this case the robot is self-motivated to find the gaps by exploring its knowledge, either by trying to classify hallucinated samples (obtained by sampling the feature space), or by analysing the model. The robot therefore tries to find the problematic parts of its models without referring to the current sensorial input. It can find, for instance, that the models of colours red and orange are ambiguous, only by observing the learned representations.

We also identify three *types of knowledge gaps* in categorical knowledge: '*no model*', where the system detects that it does not have a model of a particular object, '*weak model*', where the model of a particular concept has been learnt, however it is not capable of reliable classification, and '*ambiguous models*', where the learned models tend to produce ambiguous results. We discuss these three types of gaps and describe when and how they can be detected.

We derive a *formal model* for detecting the knowledge gaps. This model refers to situational types of knowledge gaps; given an observation, the system tries to classify this sample in one of the previously learned classes and then to estimate if it was able to reliably classify the observation or it failed to do that due to a gap in its knowledge. Our approach does not make the closed world assumption; at every step the system also takes into account the probability that it has encountered a concept that has not been observed before. The approach first estimates a posteriori probability of classification of the observed sample into one of the previously learned classes, or in the "unknown" class, that models possible new (previously not encountered) classes. By analysing this a posteriori probability, we are able to measure the ambiguity of the decision and relate this to knowledge gaps. We also define the requirements for models that can be used in the proposed knowledge gaps detection framework. We then present an example of a probabilistic model suitable for our framework and show how a non-probabilistic class model can be converted into a suitable probabilistic model. We also show the applicability of the proposed approach in different learning domains. We present the experimental results that demonstrate that the effective knowledge gap detection mechanism can lead to an efficient active learning strategy, which speeds up the learning process.

The second paper, which is attached in Annex 2.2 [15], addresses the problem of detection of knowledge gaps in the case of *learning object affordances*. When a robot interacts with the environment producing changes through its own actions, it should find opportunities for learning and updating its own models of the environment. To do this efficiently, it first has to detect gaps in its knowledge and then plan the actions that would fill these gaps. We present a method on automatic entropy-based construction of probabilistic automata based on the CrySSMEx algorithm, which might be useful for these tasks. These probabilistic automata can be used as a prediction tool, as a means to assess the uncertainty or predictability of specific action consequences and thus to estimate the gaps in the current knowledge. As such, they can serve as a useful tool in the active learning process. In general, the CrySSMEx method only finishes when it converges to a deterministic automaton, but intermediate results can also be gathered. In the process of searching convergence, conditional entropy and a quantisation method are used to find discrete representations of the inherent dynamical system in the form of states and stochastic transitions in the automaton.

The paper shows preliminary experiments in the induction of probabilistic automata from data sequences obtained from pushing experiments with objects in simulation. The data sequences comprise motor command vectors, as well as finger effector and object poses. The algorithm finds deterministic automata after convergence. Thus, the models induce complete predictability of object behaviour, given the features obtained from a simulator and a finite number of actions, which is an expected result.

1.3 Relation to the state-of-the-art

In this section we discuss how our work is related to, and goes beyond the current state-of-the-art.

Detection of knowledge gaps in categorical knowledge is in the literature usually studied in the context of active learning. The approaches to active learning of categorical knowledge focus on estimating classifiers using minimal amount of data. They are motivated by the fact that there are many situations in which large quantities of unlabelled data are relatively easily obtained, however, the cost of labelling each sample can be high. Depending on how the data is accessed, we can divide the approaches to active learning into two major groups: (i) pool-based approaches to learning, and (ii) learning from a streaming data.

In pool-based learning, all data is available in advance, a selection procedure determines the learning points, queries an oracle for labels of these points, and uses these points to construct a classifier. Here, an important issue is which data-points to chose for querying. A plethora of papers have been published on this topic proposing numerous approaches [3, 17, 16, 13, 1, 23, 12, 5, 2, 8] using different kinds of classifiers and committees of classifiers, as well as probabilistic rules for selecting the next best sample for querying.

In the stream-based learning, the data comes sequentially, and possibly indefinitely. Here the challenge is to constantly adapt the classifier to the possibly changing properties of the data and identify in the observed sequence the potentially informative data-points for querying the oracle. Although the samples are introduces sequentially, most of the learning algorithms for streaming data process the data in small batches [24, 4, 6].

For situations in which a tutor is sequentially presenting objects to the robot, the pool-based approaches are not applicable, since they assume that the agent would have access to all observed objects. In that respect, the traditional streaming-data-based active learning approaches are also not applicable, since they assume a batch of data-points to be available for constructing the classifier. In real-life situations, it is desirable that the robot detects good candidates for querying on the fly and updates its classifiers accordingly, while posing minimal number of questions to the tutor. These are the requirements that we have set to our system and the approach presented in this deliverable meets these requirements.

Active selection of samples is even more important in robotics systems that learn object affordances or action effects by directly interacting with objects. In this scenarios usually a huge amount of training data is required, therefore it is very expensive to use passive strategies for acquiring knowledge only. In recent years, there has been an increasing interest in finding proper methods for active data sampling [11].

The use of probabilistic methods like Hidden-Markov Models or Gaussian Mixture Models for prediction in dynamical settings has increased recently. In our case, we chose the CrySSMEx algorithm because it has been used specifically for the analysis of some types of dynamical systems. The algorithm infers probabilistic automata similar to Markov models. Furthermore, it implements interesting features useful for representations of knowledge gaps and planning. For instance, a quantization method for finding discretization of states and an information-theoretic method to represent uncertainty between state transitions. Additionally, minimization strategies are also incorporated in the algorithm that compress the models even more.

2 Annexes

2.1 Kristan, Skočaj and Leonardis "Principles of discovering gaps in categorical knowledge"

Bibliography D. Skočaj, M. Kristan and A. Leonardis: "Principles of discovering gaps in categorical knowledge", TR-LUVSS-03/10, University of Ljubljana, Faculty of Computer and Information Science, July 2010

Abstract In this work we address the problem of detecting gaps in knowledge representations. We treat detection of knowledge gaps within a broader context of an active learning paradigm. We describe how to detect knowledge gaps, how to represent them, and how to fill them. We present a taxonomy of types of knowledge gap detection by dividing it into two major groups based on the time, space and motivation for detecting the gap. We then further divide these groups based on how the knowledge gap detection is triggered or performed. We also identify three main types of knowledge gaps that we encounter when learning categorical knowledge. We then present a general probabilistic model for detection of knowledge gaps and propose methodology for measuring and discovering the various types of gaps. We provide requirements of the probabilistic models and propose how a non-probabilistic representation of knowledge can be implemented within our probabilistic framework. We also show how to apply the proposed mechanism for knowledge gap detection to different learning domains. We present the experimental results that demonstrate that the effective knowledge gap detection mechanism can lead to an efficient active learning strategy, which speeds up the learning process.

Relation to WP Detection and representation of knowledge gaps is an integral part of the interactive continuous learning process, which is the main topic of research in WP5, in particular Task.5.5. Therefore the approach presented in this paper plays an important role in the mechanisms we have been developing in this workpackage.

2.2 Roa and Kruijff "On the automatic Entropy-based construction of Probabilistic Automata in a Learning Robotic Scenario"

Bibliography S. Roa, and G.-J.M. Kruijff : "On the automatic Entropy-based construction of Probabilistic Automata in a Learning Robotic Scenario", Robotics: Science and Systems 2010 Workshop: Towards Closing the Loop: Active Learning for Robotics, Zaragoza, Spain, June 2010

Abstract When a robot interacts with the environment producing changes through its own actions, it should find opportunities for learning and updating its own models of the environment. A robot that is able to construct discrete models of the underlying dynamical system which emerges from this interaction can guide its own behavior and adapt it based on feedback from the environment. Thus, the induction of probabilistic automata from this sensorimotor loop might be useful for planning/learning tasks. These probabilistic automata can be used as a prediction tool, as a means to assess the uncertainty or predictability of specific action consequences and thus, as a tool for an active learning method.

Relation to WP This work is related to Tasks 5.2 and 5.5. Task 5.2 is related to learning of cross-modal concepts, while Task 5.5 is related to knowledge-gap representation, which is a crucial issue when dealing with active sampling of robot actions. This work is also related to WP2, specifically Task 2.10.

References

[1] C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*, page 111118, 2000.

[2] N. Cebron and M. R Berthold. Active learning for object classification: from exploration to exploitation. *Data Mining and Knowledge Discovery*, 18(2):283299, 2009.

[3] D. Cohn, L. Atlas, and R. Lander. Improving generalization with active learning. *Machine Learning*, 15(2):201221, 1994.

[4] W. Fan, Y. Huang, H. Wang, and P. S Yu. Active mining of data streams. In *Proceedings of the Fourth SIAM International Conference on Data Mining*, pages 457–461, 2004.

[5] A.D. Holub, P. Perona., and M. Burl. Entropy-based active learning for object recognition. In *Workshop on Online Learning for Classification, in conjunction with Conf. Comp. Vis. Pattern Recognition*, pages 1–8, 2008.

[6] S. Huang and Y. Dong. An active learning system for mining time-changing data streams. *Intelligent Data Analysis*, 11(4):401419, 2007.

[7] H. Jacobsson. The crystallizing substochastic sequential machine extractor - `CrySSMEx`. *Neural Computation*, 18(9):2211–2255, 2006.

[8] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Active learning with gaussian processes for object categorization. In *International Conference on Computer Vision*, 2007.

[9] M. Kristan, D. Skočaj, and A. Leonardis. Online kernel density estimation for interactive learning. *Image and Vision Computing*, 28(7):1106–1116, July 2010.

[10] M. Kristan, D. Skočaj, and A. Leonardis. Principles of discovering gaps in categorical knowledge. Technical Report TR-LUVSS-03/10, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia, July 2010.

[11] R. Martinez-Cantin, J. Peters, and A. Krause, editors. *Towards Closing the Loop: Active Learning for Robotics, Robotics: Science and Systems 2010 Workshop*, Zaragoza, Spain, June 2010.

[12] A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In *Proceedings of the fifteenth international conference on machine learning*, pages 350–358, 1998.

[13] H. T Nguyen and A. Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 79, 2004.

[14] S. Roa, G.-J. Kruijff, and H. Jacobsson. Curiosity-driven acquisition of sensorimotor concepts using memory-based active learning. In *Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics*, pages 665–670, 2008.

[15] S. Roa and G.-J.M. Kruijff. On the automatic entropy-based construction of probabilistic automata in a learning robotic scenario. In *Robotics: Science and Systems 2010 Workshop: Towards Closing the Loop: Active Learning for Robotics*, 2010.

[16] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. page 839846, 2000.

[17] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, page 287294, 1992.

[18] D. Skočaj, M. Kristan, and A. Leonardis. Continuous learning of simple visual concepts using Incremental Kernel Density Estimation. In *VISSAP 2008*, pages 598–604, 2008.

[19] D. Skočaj, M. Janiček, M. Kristan, G.-J. M. Kruijff, A. Leonardis, P. Lison, A. Vrečko, and M. Zillich. A basic cognitive system for interactive continuous learning of visual concepts. In *Proceedings of the ICRA 2010 Workshop on Interactive Communication for Autonomous Intelligent Robots (ICAIR) Making robots articulate what they understand, intend, and do.*, Anchorage, AK, USA, May 2010.

[20] D. Skočaj, M. Kristan, and A. Leonardis. Formalization of different learning strategies in a continuous learning framework. In *EPIROB'09*, pages 153–160, 2009.

[21] D. Skočaj, M. Kristan, A. Leonardis, M. Mahnič, A. Vrečko, M. Janíček, G.-J. M. Kruijff, P. Lison, M. Zillich, C. Gretton, M. Hanheide, and M. Göbelbecker. A system approach to interactive learning of visual concepts. 2010. Submitted.

[22] J. L. Wyatt, A. Aydemir, M. Brenner, M. Hanheide, N. Hawes, P. Jensfelt, M. Kristan, G.-J. M. Kruijff, P. Lison, A. Pronobis, K. Sjöö, D. Skočaj, A. Vrečko, H. Zender, and M. Zillich. Self-understanding & self-extension: A systems and representational approach. *IEEE Transactions on autonomous mental development*, 2010. Accepted for publication.

[23] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang. Representative sampling for text classification using support vector machines. In *Proceedings of the 25th European conference on IR research*, pages 393–407, 2003.

[24] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from stream data using optimal weight classifier ensemble. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, (99):1–15, 2010.

Principles of discovering gaps in categorical knowledge

TR-LUVSS-03/10

Matej Kristan Danijel Skočaj Aleš Leonardis

July 29, 2010

Abstract

In this work we address the problem of detecting gaps in knowledge representations. We treat detection of knowledge gaps within a broader context of an active learning paradigm. We describe how to detect knowledge gaps, how to represent them, and how to fill them. We present a taxonomy of types of knowledge gap detection by dividing it into two major groups based on the time, space and motivation for detecting the gap. We then further divide these groups based on how the knowledge gap detection is triggered or performed. We also identify three main types of knowledge gaps that we encounter when learning categorical knowledge. We then present a general probabilistic model for detection of knowledge gaps and propose methodology for measuring and discovering the various types of gaps. We provide requirements of the probabilistic models and propose how a non-probabilistic representation of knowledge can be implemented within our probabilistic framework. We also show how to apply the proposed mechanism for knowledge gap detection to different learning domains. We present the experimental results that demonstrate that the effective knowledge gap detection mechanism can lead to an efficient active learning strategy, which speeds up the learning process.

1 Introduction

Cognitive systems are often characterised by their ability to learn, interact with the environment and act autonomously. They are able to respond to the requests of human users and other cognitive agents, and they are also able to take the initiative and engage in a dialogue with a human or in interaction with the environment. Very importantly, they are able to learn from such interactions; they are able to acquire novel knowledge and update the previously learned conceptual models in an incremental manner. They can passively receive the information they need in this incremental learning process. In this case they simply rely on the environment, or on a human tutor, for being provided with with appropriate information for efficient learning. Or they can take an active part in this incremental learning process and try to infer what kind of information is needed to make the learning more efficient. The latter learning approach is known as *active learning.*

Active learning requires that the system identifies learning opportunities. This in turn requires that it must be able to *detect gaps in its knowledge*, which may indicate good learning opportunities. Once these gaps are detected, it can plan what actions to perform to fill these gaps, and after the required information is obtained, it can extend its current knowledge accordingly. A crucial requirement of a system that is supposed to self-extend is therefore a certain level of self-understanding enabling the detection of gaps in its knowledge.

In this work we focus on this problem. We address the problem of knowledge gap detection in the context of an active learning paradigm and propose a methodology that tackles this problem. Since our research has been concentrated around interactive continuous learning of conceptual knowledge in dialogue with a tutor, most of this document has been written with this problem in mind. However, the proposed solutions are general enough such that they can be applied to other learning domains as well.

The document is organised as follows. In Section 2 we first motivate our work and present the problem of knowledge gap detection in the context of active learning and present related work. In Section 3 we first identify several types of knowledge gap detection and several types of knowledge gaps that arose in a scenario where the robot is acquiring categorical knowledge in interaction with a tutor. We then formalise our approach and derive the method for knowledge gap detection in Section 4. We also specify the requirements of our method and show how it can be used in various learning domains in Section 5. In Section 6 we present experimental results and then conclude the paper in In Section 7 with a discussion and conclusions.

2 Interactive learning

2.1 Active learning paradigm

First, we place the detection of knowledge gaps within the broader context of the active learning paradigm. We envision one self-extension action (incremental update of the current knowledge) comprising of four main steps:

1. **Detection of knowledge gaps.** The system should first self-understand, it should understand what it does and what it does not know. It should detect using its internal modal representations what information is missing.

2. **Issuing a request.** Based on the detected ignorance the learner should determine (plan) what information is needed to fill the gap in the knowledge and issue a request (a desire, a motive) to the overall system about **what** information it would like to obtain (e.g., a view from the opposite side is needed, more similar objects are needed, a push from another direction is needed, disambiguation between the concepts of red and blue is needed, etc.). It will not, however, tell how this information should be given. And it will also not represent externally its ignorance using its internal representations.

3. **Planning and execution of actions.** The system should then plan the sequence of actions that would lead to the state that would reveal information asked by the particular modal learner. It would thus determine (plan) **how** and **when** to obtain a particular piece of information (e.g., the robot would move to get a novel view-point, or it would grasp and rotate the object, or it would ask a human to rotate the object, or it would push the object, or it would initiate a dialogue with the tutor like: *"I would like to see the object from the opposite side."* - *"I would like to see more similar objects."* - *"I would like the object to be pushed from another direction."*, etc.) or requesting some clarification : *"Is this a red object?"* - *"Is this a cube?"* - *"I have detected a new object. What is it?"*.). This planning should be done by a planner at a higher level of the system architecture and not within the modality.

4. **Updating the current knowledge.** After the action has been executed the modal learner will gather novel information and use it for updating the current internal representations.

Steps 1, 2, and 4 are modality-specific and also representation-specific and learner-specific. They are highly dependent on the types of representations and learning methods used. Step 3 is not modality specific, it operates with abstracted representations interfacing other modalities as well. In this step it is decided whether to learn or not, and how to provide the required information. It should deal with motivation for learning, it should decide about the importance of learning, it should consider other tasks the system can and should perform and the gains they can bring, as well as the costs that are needed, and plan accordingly.

Each modality keeps its internal modal representation of the world and concepts that are being learned. These internal representations have to enable detection of knowledge gaps (Step 1) and determine what kind of information is needed to fill these gaps (Step 2). Such next best view planning or next move planning, etc., has to be performed within a single modality, using the internal modal representations (although some information from other modalities could also be indirectly used). However, these modality-specific representations cannot be used to represent the knowledge gap externally (since other modalities know nothing about the particular representations used). The knowledge gap is represented/described with a request for information, which would be required to fill the gap. Therefore, the modality-specific learner should inform higher level processes, like motivation and planning, **what** information is needed to extend the current knowledge (e.g., the appearance of the object from the opposite side). These higher level processes then have to decide whether the request will be approved and plan the sequence of actions that would describe **how** and **when** the requested information will be provided (e.g., move around the object or rotate the object). This interface between modality-specific learners and the higher level processes should be carefully designed to enable situated dialogue for socially guided learning - clarification, establishing transparency, verbalising knowledge and knowledge gaps, verbalising actions that are to be undertaken to fill these gaps, to enable scaffolding, etc.

The four self-extension steps mentioned above reflect the need for **detecting the gaps in knowledge** and show how the active learning can be performed in a continuous learning framework. In the rest of this document we will focus on Step 1, thus on principles of detecting the gaps in the knowledge.

2.2 Related Work

The approaches to active learning focus on estimating classifiers using minimal amounts of data. They are motivated by the fact that there are many situations in which large quantities of unlabelled data can be relatively easily obtained, however, the cost of labelling each data-point can be high. Depending on how the data is accessed, we can divide the approaches to active learning into two major groups: (i) pool-based approaches to learning, and (ii) learning from streaming data. In pool-based learning, all data is available in advance, a selection procedure determines the learning points, queries an oracle for labels of these points, and uses these points to construct a classifier. In stream-based learning, the data comes sequentially, and possibly indefinitely. Here the challenge is to constantly adapt the classifier to the possibly changing properties of the data and identify in the observed sequence the potentially informative data-points for querying the oracle.

An important issue in pool-based active learning approaches is which data-points to choose for querying. Cohn et al. [4] used a feed-forward neural network and suggested to select samples in the areas of the feature space where the most specific and the most general hypotheses disagree. Seung [16] proposed an approach which generates a number of viable classifiers from the labelled data-set to measure which unlabelled data-point gets confused most under these classifiers. The most confused point is then selected for the next query. Schon et al. [15] used a Support Vector Machine (SVM) classifier and proposed to query for samples that lie most closely to the decision hyperplane between the two classes. A similar approach is applied by Nguyen et al. [14]. Campbell et al. [1] select the sample that results in the maximal decrease of the margin between two classes. To better explore the feature space, they precluster the data-points and use only the cluster centers for querying candidates. Xu et. al [21] applied preclustering of the data to identify the clusters only within the margin of the classifier. Again, only the centers of the clusters are used for querying. McCallum et al. [13] combined an Expectation Maximization (EM) algorithm with the Query-By-Committee algorithm. In their approach, labelled training samples are used to construct a committee of classifiers and the query points are identified as those points for which the committee's variance of the classification is the greatest. Holub [6] applied a committee of classifiers to calculate the expected amount of information gain for each unlabelled data-point. The data-point that maximises this gain selected for querying. Cebron et al. [2] determines the query points by a measure that combines the typicality of the data-points with their uncertainty under the learnt classifier. The typicality is measured via the estimated probability density function that is calculated from the pool of data-points. The uncertainty of a particular data-point is evaluated by calculating the Shannon entropy over the posterior distribution of classes for that data-point. Kapoor [8] applied a Gaussian Process prior to derive a probabilistic rule for selecting the next best sample for querying.

The learning algorithms for streaming data process the data in small batches. Some researchers apply an ensemble of classifiers learnt from chunks of data to arrive at a stronger classifier. Zhu et al. [22] divide the observed stream of data-points into chunks and learn a classifier from each chunk. These classifiers are then combined in a weighted ensemble of classifiers, by optimising the weights such that the classification variance of the unlabelled data-points is minimised. From each new chunk of data, a random subset is selected and labelled by the oracle. This subset is then used to initialise a new classifier and used to identify additional points for labelling. Fan et al. [5] analyse chunks of data-stream to determine whether a drift has occurred in the classes distribution. In case of detected drift, data-points are randomly sampled for querying. Huang and Dong [7] also apply a drift detection, and use a light-weight uncertainty sampling to determine the points for querying.

For situations in which a tutor is sequentially presenting objects to the robot, the pool-based approaches are not applicable, since they assume that the agent would have access to all observed objects. In that respect, the traditional streaming-data-based active learning approaches are also not applicable, since they assume a batch of data-points to be available for constructing the classifier. In real-life situations, it is desirable that the robot detects good candidates for querying on the fly and updates its classifiers accordingly, while posing a minimal number of questions to the tutor.

3 Detection of knowledge gaps

3.1 Types of knowledge gap detection

Knowledge gaps can in general be divided into two groups based on the time, space and motivation for detecting the gap; either the lack of knowledge is related to a particular situation or not (see Fig. 1).

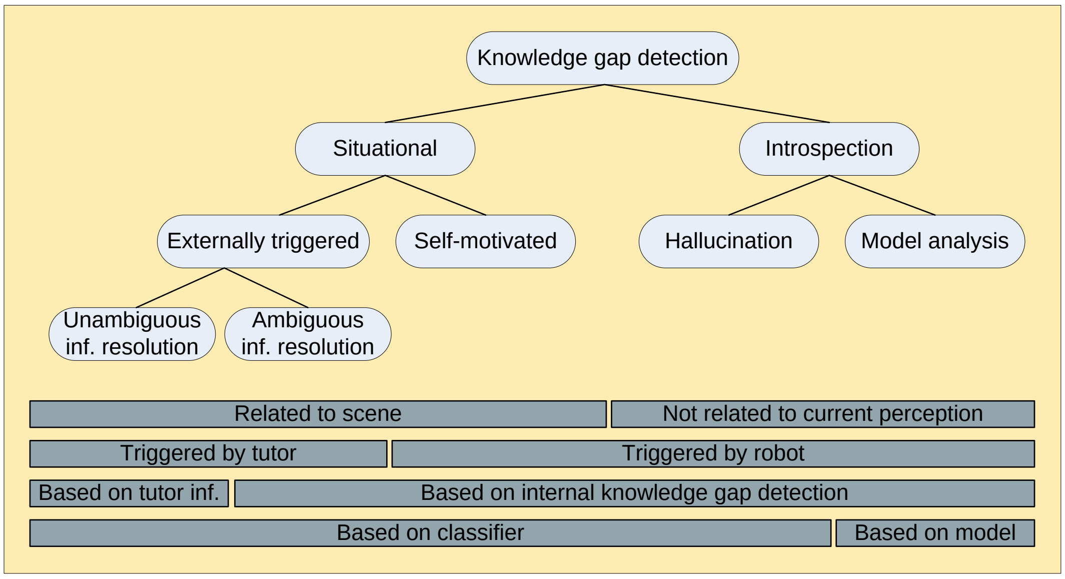


Figure 1: Different types of knowledge gap detection.

In the first case, the lack of knowledge is **situational**; it is related to a particular situation or a particular task the robot has to perform, however due to the lack of knowledge, it fails. Maybe it can't answer a question, it can't find a way, it cannot recognise an object, it cannot find its way, it cannot perform an action it was requested to, it cannot predict action effects, etc. The robot therefore tries to complete a certain task, however it is not capable of doing that due to a gap in its knowledge. At this point, it should detect this gap, represent it adequately, find out what information is needed to fill this gap and try to obtain this information accordingly. It can try to perform some action itself (e.g., move to a new position), or to ask for a help from a human (e.g., by asking a question).

In this case we can distinguish two sub-cases depending how the knowledge gap detection is triggered. This process can be **externally triggered**; the robot is asked (by a human or some other agent) to perform a task, but it fails (e.g., it can't answer a question). Or it is given an information, it cannot interpret. Or it tries to complete some task, but it fails (it can't move from one place to another as planned). The robot therefore faces a problem during execution of a task that it can not solve due to a gap in its knowledge. The other option is that the robot is **self-motivated** to look around (without being asked to do so) and is intentionally looking for situations (places, objects, actions) that it can not interpret successfully due to gaps in its knowledge. In both cases the gap is detected based on a particular situation the robot is engaged in.

The externally-triggered knowledge gap detection can be further decomposed. As mentioned above, in this type of knowledge gap detection the robot can receive the information from a human tutor. Based on the type of information it receives we can distinguish two cases. If the robot can resolve the received information (**Unambiguous information resolution** in Fig. 1), it can use this information directly to detect the gap. E.g., if it is asked about a concepts, he has not encountered yet, this indicates a gap in its knowledge. Or, if its interpretation of the scene does not agree with the information asserted by the tutor, this indicates its erroneous recognition, which again implicates the incompleteness of its knowledge (providing that the tutor always provides a correct information). Therefore, in this case the robot can base the knowledge gap detection solely on the information provided by the tutor. In contrary, when the tutor can not attach the obtained information to particular objects (**Ambiguous information resolution**), the robot can not rely on this information only and should use internal knowledge gap detection mechanisms. For instance, when there are several objects in the scene, and the robot is asked to pick up a mug, but it can't recognise any mug in the scene, this indicates that there is a gap in its knowledge. However, it can not relate the gap to a particular object in the scene and has to apply its knowledge gap detection mechanisms to do that.

Another way of detecting knowledge gaps is through **introspection**. In this case the detection of knowledge gaps is completely self-driven and is not related to any particular situation or task. It is not triggered by any external problem; it is triggered by an inner motivational mechanism with the goal of detecting ignorance and of proposing the actions that would provide the information needed to extend the current knowledge. No sensorial inputs are used in this case; the detection of knowledge gaps is based solely on the current knowledge.

Also in this case we can discuss two different subcases, based on the type of methodology used for detecting the knowledge gaps. In the case of **hallucination**, the robot tries to hallucinate sensorial inputs (basically, it samples over distributions of feature values it uses), and tries to interpret these hallucinated situations. Failing to do that would indicate a knowledge gap. This type of knowledge gap detection is thus also based on the output of the classifier, which is built on the top of the models; the only difference is that the input is hallucinated and not perceived. This sampling cannot be random, especially in the case of high-dimensional feature spaces; it should be driven by the structure of the current knowledge and by the output of previous classifications. This brings us to another type of introspection, **model analysis**. Here, the system tries to directly analyse the current models and to determine which models are ambiguous, weak or partially undefined analytically or algorithmically (e.g. two models of different concepts overlap, a model is weakly defined, the model is not defined for certain feature values, etc.). Such introspection strongly depends on the nature of the underlying models; if such model analysis is not possible, only a variant of hallucination-based approach can be used.

In the case of non-situational introspection, however, it is more difficult to communicate this gap, since the detection is not based on a particular situation the robot could refer to. In some cases the robot can communicate its hallucination or ambiguous interpretation of this hallucination (e.g., *"Please, show me a red object that looks almost as blue."* to refine the models of concepts of red and blue). In other cases the robot can remember this gap and try to fill it when it encounters the situation that would help it to resolve the problem and it could plan to undertake actions that would lead it to such situations. There are also some cases, when the robot does not need to communicate its knowledge gap in order to obtain information needed to fill this gap. It can create a situation that was found to lead in a knowledge gap. For instance, in the affordance learning scenario, the robot first detects which part of the feature space is not modelled well and then performs an action that produces the corresponding features. Of course, it this case, the robot has to know the inverse mapping from the feature values to the action parameters (as opposed to the mapping from action parameters to feature values, which is a part of the regular feature extraction process). These kinds of learning scenarios can take full advantage of introspective knowledge gap detection, which may lead to a very efficient training sample selection in an active learning settings.

3.2 Types of knowledge gaps

Above we identified different means of detecting knowledge gaps. Here we will define different types of knowledge gaps. Every detected gap in the conceptual knowledge falls in one of the following three categories:

- **No model**. This type of knowledge gap occurs during situated gap discovery, when the current observation cannot be explained by any of the previously learned models, or when the robot is asked to recognise a concept (e.g., an object) it has not encountered before.
- **Weak model.** The robot has already learned the concept, but the model is very weak and non reliable for robust recognition. During the situated knowledge gap discovery this would reflect in a weak response of the classifier, which could not reliably determine whether the observation belongs to a particular concept or not. Also, this type of knowledge gap can also be determined by introspection (e.g., the robot notices that it has observed a very small number of instances to create a reliable model).
- **Ambiguous models.** In an interaction with a tutor, the robot's interpretation of the scene does not agree with the information asserted by a human tutor. Or, the classifier produces a similar response for two or more classes and therefore fails

to reliably classify the observation. Such a response indicates that the underlying models overlap and are therefore ambiguous. This overlap could also be detected by model analysis.

When defining these different types of knowledge gaps we assumed that the data (features) that were used for classification in the situated knowledge gap discovery were reliably estimated, and that the system failed to reliably classify the sample due to incompleteness of the classifier and of the current representations. However, it may also happen that the feature values that are used in the learning process are non-reliably estimated (e.g., due to missdetection, bad segmentation, occlusions, noise, etc.). In this case the interpretation of the observation does not fail due to the knowledge gap in the learned models but rather to the deficiencies in the preprocessing steps.

The knowledge gaps listed above are addressed in the next section. The first type of the gap (No model) in the list is of particular interest in the continuous learning framework. The knowledge gap due to the unknown concept is quite common at the beginning of the incremental learning process since the systems knowledge is rather empty initially, and only slowly includes novel concepts during the online learning process. Therefore, the learning mechanism should contain an "unknown" class that should encompass all the concepts that have not been encountered until that particular moment. This lack of knowledge due to immaturity with regard of the development of the system is very difficult to model. In the next section we present a general and flexible solution to this problem and also formally propose how to detect other types of knowledge gaps.

4 Formal model

4.1 Derivation of the model

To be more specific, we will relate our formal model for knowledge gaps detection to the problem of learning visual concepts. The model, however, is more general and could be applied on other learning domains as well.

Generally speaking the robot collects the visual information about its environment as follows. First it determines a region in an image which contains the interesting information, then it "segments" that region and extracts the feature values $\mathbf{z}$ from which it later builds models of objects, concepts, etc. The visual information may be ambiguous by itself, and segmentation may not always be successful. We will assume that some measure of how well the segmentation was carried out exists and we will denote it by $s \in [0, 1]$. High values of s (around one) mean high confidence that a good observation $\mathbf{z}$ was obtained, while low values relate to low confidence.

Probability of observation $\mathbf{z}$, given the observer's confidence score s, can be written in terms of the fact that the observation might come (or not) from an existing internal model. Let $m \in \{m_k, m_u\}$ denote two possible events: (i) the observation came from a existing internal model m_k, and (ii) the observation came from an unknown model m_u. We define the knowledge model as a probability of observation $\mathbf{z}$, given the confidence score s:

$$p(\mathbf{z}|s) = p(\mathbf{z}|m_k, s)p(m_k|s) + p(\mathbf{z}|m_u, s)p(m_u|s). \tag{1}$$

The function $p(\mathbf{z}|m_k, s)$ is the probability of explaining $\mathbf{z}$ given that $\mathbf{z}$ comes from one of the learnt models, $p(m_k|s)$ is the a priori probability of any learnt model given the observer's score s. The function $p(\mathbf{z}|m_u, s)$ is the probability of $\mathbf{z}$ corresponding to the unknown model, and $p(m_u|s)$ is the probability of the model "unknown" given the score s.

Assume that the robot has learnt K separate alternative internal models $M = \{M_i\}_{i=1:K}$ from previous observations. The probability $p(\mathbf{z}|m_k, s)$ can then be further decomposed in terms of these K models,

$$p(\mathbf{z}|m_k, s) = \sum_{i=1}^{K} p(\mathbf{z}|M_i, m_k, s)p(M_i|m_k, s). \tag{2}$$

If we define the "unknown" model by M_0 and set $p(\mathbf{z}|m_u, s) = p(\mathbf{z}|M_0, m_u, s)p(M_0|m_u, s)$, then (1) becomes

$$\begin{aligned} p(\mathbf{z}|s) = p(m_k|s) &\sum_{i=1}^{K} p(\mathbf{z}|M_i, m_k, s)p(M_i|m_k, s) \\ &+ p(m_u|s)p(\mathbf{z}|M_0, m_u, s)p(M_0|m_u, s). \end{aligned} \tag{3}$$

Note that the "unknown model", M_0, accounts for a poor classification, by which we mean that none of the learnt models supports the observation $\mathbf{z}$ strongly enough. We assume that the probability of this event is uniformly distributed over the feature space, which means that we can define the likelihood of model M_0, given observation $\mathbf{z}$ by a uniform distribution, i.e., $p(\mathbf{z}|M_0, m_u, s) = \mathcal{U}(\mathbf{z})$. Note also, that the only possible unknown model comes from the class M_0, therefore $p(M_0|m_u, s) = 1$.

The observation $\mathbf{z}$ can be classified into the class M_i which maximises the a posteriori probability (AP). The a posteriori probability of a class M_i is calculated as

$$p(M_i|\mathbf{z}, s) \propto p(\mathbf{z}|M_i, m, s)p(M_i|m, s)p(m|s), \tag{4}$$

where $m = m_k$ for $i \in [1, K]$ and $m = m_u$ for $i = 0$. Note that the terms $p(M_i|m_k, s)$ are the prior probabilities of the learnt known models and can be in practice approximated through the frequency at which each class is observed. Another important term is $p(m_u|s)$, which represents the prior probability of an unknown class. In practice, we can model this term by a function that decreases with the number of observed samples (e.g., [3]) or approximate it with a constant.

The gap in knowledge can be discovered through inspection of the AP distribution. In particular, if the AP distribution exhibits an *ambiguous classification* of the observation $\mathbf{z}$, or classifies it as an "unknown" (or unaccounted), then this is a good indication that we are dealing with a gap in knowledge.

4.2 Detecting knowledge gaps

Let us assume that we want to test our internal knowledge model about some observation $\mathbf{z}$. The first step is to calculate the a posteriori distribution over all classes, including the class "unknown" as described above.

If we want to classify the observation in one of the K classes, we would intuitively assign it to the class with the maximum a posteriori probability, therefore into the class

$$M_{\text{maxap}} = \arg\max_{M_i}\{p(M_i|\mathbf{z}, s)\}. \tag{5}$$

However, we may then want to check how reliable this classification is. We shall not consider only the strongest response, but also others, including the response of the "unknown model".

Intuitively we might want to evaluate a level of certainty of maximum a posteriori classification through hypothesis testing. One hypothesis, $h$class, is that $\mathbf{z}$ corresponds to M_{maxap}, while the alternative hypothesis, $h$alt, is that $\mathbf{z}$ might be explained by some other class (including the class "unknown"). The score which tells the certainty of the hypothesis $h$class is the likelihood ratio of $\mathbf{z}$ under hypothesis $h$class and the alternative hypothesis $h$alt

$$R_{\text{hyp}} = \frac{p(h\text{class}|\mathbf{z}, s)}{p(h\text{alt}|\mathbf{z}, s)}, \tag{6}$$

where $p(h\text{class}|\mathbf{z}, s)$ is the likelihood of $\mathbf{z}$ given the classification hypothesis $h$class and $p(h\text{alt}|\mathbf{z}, s)$ is the likelihood of $\mathbf{z}$ given the hypothesis $h$alt. A high score R_{hyp} corresponds to a confident hypothesis $h$class and a *low score* (close or below 1) corresponds to a weak hypothesis, which indicates *a gap in knowledge.*

The likelihood of $\mathbf{z}$, given the hypothesis $h$class, is defined as the likelihood of $\mathbf{z}$, given the maximum a posteriori class label M_{maxap}, i.e.,

$$p(h\text{class}|\mathbf{z}, s) = p(M_{\text{maxap}}|\mathbf{z}, s). \tag{7}$$

Various definitions of the meaning of the alternative hypothesis lead to various definitions of the likelihood of $\mathbf{z}$ under that hypothesis. We provide here two examples of such definitions.

Definition 1 of the alternative hypothesis: *The feature value* $\mathbf{z}$ *can be explained by any of the* K *classes, including* $j = 0$*, and excluding* $j =$ maxap. This leads to definition

$$p({}^{(1)}h\text{alt}|\mathbf{z}, s) = \sum_{j \neq \text{maxap}} p(M_j|\mathbf{z}, s), \tag{8}$$

$$^{(1)}R_{\text{hyp}} = \frac{p(h\text{class}|\mathbf{z})}{p({}^{(1)}h\text{alt}|\mathbf{z})}. \tag{9}$$

Definition 2 of the alternative hypothesis: *The feature value* $\mathbf{z}$ *can be explained by either the next best class or the "unknown class".* This leads to the definition

$$p({}^{(2)}h\text{alt}|\mathbf{z}, s) = p(M_{\text{maxap2}}|\mathbf{z}, s) + p(M_0|\mathbf{z}, s), \tag{10}$$

$$^{(2)}R_{\text{hyp}} = \frac{p(h\text{class}|\mathbf{z}, s)}{p({}^{(2)}h\text{alt}|\mathbf{z}, s)}, \tag{11}$$

where we have defined

$$M_{\text{maxap2}} = \underset{M_i, M_i \neq M_{\text{maxap}}}{\arg\max} \{p(M_i|\mathbf{z}, s)\}. \tag{12}$$

Alternative measure of ambiguity: Another way to measure the ambiguity of decision is to calculate the entropy of the a posteriori distribution over classes. We can use the Bayes entropy measure [11], which we normalise to make it independent of the number of classes

$$H = (1 - \sum_{i=0}^{K} p(M_i|\mathbf{z}, s)^2)/(1 - \frac{1}{K}). \tag{13}$$

Intuitively, a high entropy indicates more uniform distribution, which hints ambiguity of decision. Therefore, a *high entropy* indicates a *knowledge gap*. On the other hand, a low entropy indicates a dominant mode in the a posteriori distribution which indicates certain decision, with one exception, that is, if the *strongest response* is produced by the *"unknown model"*. Such response indicates a *knowledge gap* as well.

4.2.1 Illustrative example

Figure 2 shows two examples of a posteriori distributions, where the number of the learnt classes is $K = 3$ and the class unknown is denoted by label M_0. The first distribution shows a strong response for the M_1, however, there is some nonzero probability that the observation might correspond to some other class – most likely M_2 and there is even some probability that $\mathbf{z}$ might have been poorly classified (M_0). The second distribution in Figure 2 shows a different example, in which the observation most likely corresponds to an unknown class. Therefore, in the first case the recognition would be considered reliable enough, while in the second case a knowledge gap would be detected.

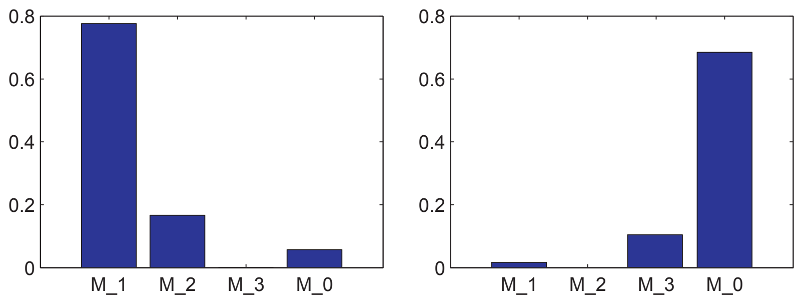


Figure 2: An example of a posteriori pdf over the classes M_1, M_2, M_3 and the unknown class M_0. The left image shows an unambiguous decision with only one dominant known class, while the right image shows an ambiguous decision, with the unknown model as the dominant model.

4.3 Determining the type of knowledge gaps

In the previous subsection we described how the knowledge gap is detected.To summarise, a low likelihood ratio R_{hyp} (or a high entropy H) or a strong response of the "unknown model" (i.e., if $M_{maxap} = 0$) indicate a knowledge gap. Now we will show how to determine also the type of the knowledge gap, i.e., how to classify the knowledge gap in one of three classes defined in Section 3.2. This is also very important, since the actions that follow the knowledge gap discovery are dependent on the type of the detected knowledge gap.

Three types of knowledge gaps can be identified as follows:

- **No model**. The response of all learned models is very low, the "unknown model" clearly wins, i.e.,
$$p(M_0|\mathbf{z}, s) >> p(M_{\text{maxap}}|\mathbf{z}, s).$$
Since the observation can not be modelled by any previously learned model this indicates that it should probably be modelled with a new model.

- **Weak model.** The unknown model still produces the highest response, however, also one of the learned models responds quite well, or alternatively, one of the known models produce the highest response, which is however, quite similar to the response of the unknown model, i.e.,
$$p(M_0|\mathbf{z}, s) + thr > p(M_{\text{maxap}}|\mathbf{z}, s) >> p(M_{\text{maxap2}}|\mathbf{z}, s).$$
Such situation would indicate that the observation should probably be classified in the class M_{maxap}, however, since the corresponding model is weak, this classification can not be performed reliably.

- **Ambiguous models.** The unknown model does not produce a high response, however, there is no single dominant response of one of the known models, i.e.,
$$p(M_0|\mathbf{z}, s) << p(M_{\text{maxap}}|\mathbf{z}, s) < p(M_{\text{maxap2}}|\mathbf{z}, s) + thr.$$
In such situations the models can not distinguish between two classes reliably, which indicates that they are ambiguous.

4.4 Applying the model in the different types of knowledge gap detection

As discussed in Section 3.1 and depicted in Fig. 1, we can distinguish several types of knowledge gap detection. In this subsection we will describe how we can utilise the proposed formal model in each of these different types.

4.4.1 Situational knowledge gap detection

From the point of view of using the proposed formal model, we can distinguish two cases. In the case of unambiguous information resolution in externally triggered situational knowledge gap detection, there is no need to use the knowledge gap detection mechanism described in the previous subsection. The knowledge gap can be detected and classified only by considering information provided by the tutor and the results of the classification of the current observation (by observing the a posteriori distribution as described above). Here, the three types of knowledge gaps can be identified simply as follows:

- **No model**. The tutor provides information about the observation, e.g., attributes a class to an object, which the robot is not aware of, i.e., the particular class the robot has not encountered before. In this case, the robot has no model of the particular class, and has to initialise the model of a new class M_{K+1}.

- **Weak model.** The robot can not classify the current observation and compare it to the asserted information. The model is therefore weak, and should be updated.

- **Ambiguous models.** The robot is able to classify the current observation, however the classification result differs from the information asserted by the tutor. This implies that the model is ambiguous, and should be updated.

In other types of situational knowledge gap discovery, the robot is not given the exact unambiguous information about the observation to be compared with the outcome of its own classification. Therefore it has to rely on the knowledge gap detection mechanism described in the previous subsection, i.e., to calculate a posteriori distribution over all $M_i, i = 0..K$, then to check the values of R_{hyp} or H and to detect the type of knowledge gap by comparing $p(M_i|\mathbf{z}, s), i = 0..K$. This implies to both, externally triggered and self-motivated situational knowledge gap detection. In both cases the knowledge gap detection refers to the current observation. Where the motivation for the discovery of knowledge gaps comes from is not important for the actual mechanism used.

4.4.2 Gap discovery through introspection

In the case of non-situated knowledge gap detection, we identified two different types:

- **Hallucination.** If the robot has resources available, e.g., in an idle state, or during sleep it can *hallucinate* various feature values $\mathbf{z}$ and trie to classify them. Failure of unambiguous classification leads to discovery of the gap in knowledge. In practice this means that it first determines the possible bounds on the feature values using the internal knowledge models. Then it commences a Markov-Chain-Monte-Carlo-like (MCMC) sampling of feature values, and for each generated feature it calculates R_{hyp}. By minimizing R_{hyp} it can determine parts of its internal models which lack information. This is the hardest example when it comes to communicating the knowledge gap to the tutor. One approach may be that, as MCMC generates a sufficient number of hallucinated examples in the knowledge gap, the robot might *remember* them by fitting a model to those data (e.g., Kernel Density Estimates or some other models).

- **Model analysis.** Another way to introspect, is to identify which internal models of the learnt concepts are ambiguous. Take for example the models that relate to the concept of colour, and suppose that the robot has learnt three colours so far. Each colour is some model in some feature space. The robot might want to inspect whether significant parts of these models overlap in this feature space, thus discovering how ambiguous each model is. If a large portion of a particular model overlaps with one or more other models, then this is a good indication that the model is ambiguous. To detect this type of gap in knowledge, the robot simply needs to go through all concepts and needs to be able to calculate a measure of overlap between them. If generative models are used for knowledge representation, the ambiguity can be determined for each model as follows: First we generate samples from a concept model, calculate for each sample a measure of ambiguity (e.g., one of the measures discussed in section 4.2 could be used), and report the median value as the ambiguity of that model.

5 Using the proposed model

5.1 Requirements of models

The gap-discovery methodology described in the previous section can be applied to any knowledge model, which meets the following requirements:

1. The model has to define the probability of observation $\mathbf{z}$ as in (1) in terms of alternative explanations:

$$p(\mathbf{z}|s) = p(\mathbf{z}|m_k, s)p(m_k|s) + p(\mathbf{z}|m_u, s)p(m_u|s). \tag{14}$$

2. The explanations in terms of the learnt models should be written as a mixture model of the K learnt classes M_i:

$$p(\mathbf{z}|m_k, s) = \sum_{i=1}^{K} p(\mathbf{z}|M_i, m_k, s)p(M_i|m_k, s). \tag{15}$$

3. The models should thus define the following pdfs:
   - $p(\mathbf{z}|M_i, m, s)$; $i \in \{1, \ldots, K\}$ ... probability of the observation $\mathbf{z}$ under the i-th learnt model.
   - $p(M_i|m, s)$; $i \in \{1, \ldots, K\}$ ... a priori probability of observing the i-th model.
   - $p(m|s)$; $m \in \{m_\mathrm{k}, m_\mathrm{u}\}$ ... a priori probability that observations will correspond to a known m_k or an unknown m_u model.

4. To allow hallucination, it must be possible to generate samples from the internal knowledge models.

5.2 Example of a probabilistic knowledge model

A probabilistic knowledge model can be formed directly by estimating a probability density functions over the input features for each of the learnt models. Particularly, we can apply the online discriminative Kernel Density Estimator (odKDE) [10], which is a discriminative variant of the online Kernel Density Estimators (oKDE) [9] to estimate these models. The odKDE estimates the probability density functions by a mixture of Gaussians, is able to adapt using only a single data-point at a time, automatically adjusts its complexity and does not assume specific requirements on the target distribution. A particularly important feature of the odKDE is that is allows adaptation from the positive as well as negative examples [12]. Therefore, the distribution of each i-th alternative of the known model $p(\mathbf{z}|M_i, m_k, s)$ can be continually updated by the odKDE, while the a priori probability $p(M_i|m_k, s)$ for each model is calculated from the frequency at which each of the alternative classes M_i, $i > 0$, has been observed. The a priori probability of an unknown model (and implicitly of a known model), $p(m_\mathrm{u}|s)$ is assumed non-stationary in that it changes with time. The following function decreases the "unknown" class probability with increasing number of observations N. With above definitions, the knowledge model is completely defined and allows discovery of knowledge gaps.

Note that the last requirement in Section 5.1 states that the models should enable hallucination. In practice this means that we have to be able to (i) determine the bounds on the feature values $\mathbf{z}$ and (ii) we have to be able to sample from the part of the feature space close to the learnt models. Since we are dealing with Gaussian mixture models with an additional uniform component, the bounds on feature space can be trivially estimated by fitting a single Gaussian to the mixture model (15). The bounds are then obtained at the distance of three standard deviations from the center of the fitted Gaussian. Sampling from a Gaussain is also trivial, therefore the last requirement of Section 5.1 is met.

5.2.1 Illustrative example of gap discovery

We will illustrate the concept of gap discovery on a simple example with $K = 3$ one-dimensional models $\{M_1, M_2, M_3\}$. To further simplify the illustration, we assume that each of the models is described by a single Gaussian (note that, in general, the oKDE generates a mixture of Gaussians). Figure 3 shows this model for parameters $\mu_{1:3} = \{1, 2, 3\}$, $\sigma^2_{1:3} = \{0.1, 0.4, 0.3\}$, $p(M_{1:3}) = \{0.3, 0.5, 0.17\}$.

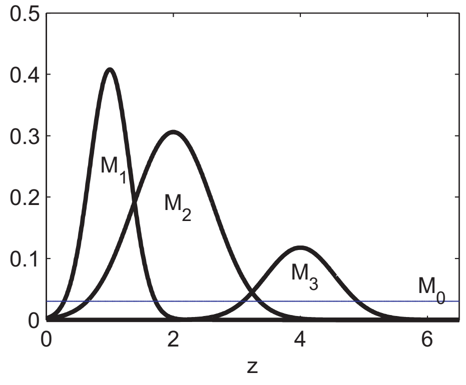
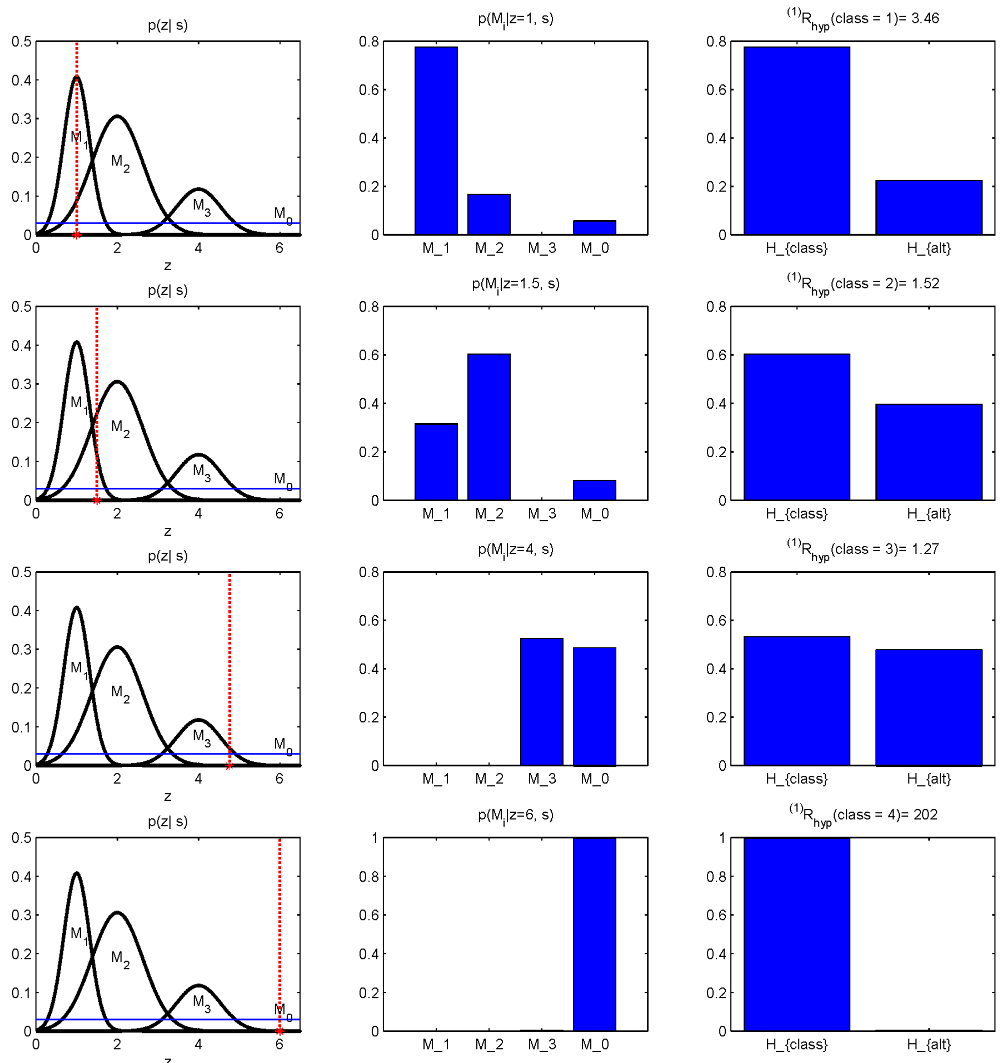
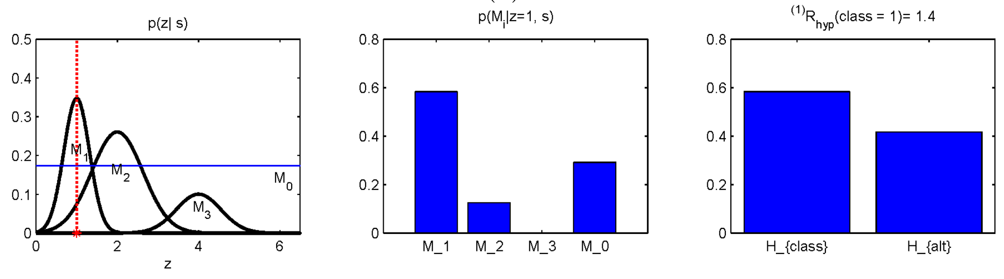


Figure 3: An example of the knowledge model using single Gaussians (black) to describe each learnt class (M_1, M_2, M_3) and a uniform distribution to describe the *unknown* class M_0 (blue).

To provide some intuition in the knowledge gap score (6), R_{hyp}, we have calculated it for different values of $\mathbf{z}$ and different values of $p(m_{\mathrm{u}}|s)$. The results for $s = 1$ are shown in Figure 4a and for $s = 0.5$ are shown in Figure 4b. The first row in Figure 4a shows an example in which classification indicates a strong model – the posterior $p(M_i|\mathbf{z}, s)$ is dominated by the class M_1 and the value of ${}^{(1)}R_{hyp}$ measure is high; this measurement thus does not indicate a knowledge gap. The second line in Figure 4a shows a classification that indicates an *ambiguous model* – according to the posterior, two classes, M_1 and M_2, support the observed data-point. The ambiguity is also indicated by the value of ${}^{(1)}R_{hyp}$, which is nearly one. The third row in Figure 4a shows an example in which the classification indicates a *weak model* – although the maximum in the posterior corresponds to one of the known models, there is large probability that the sample can also be explained by the unknown model. The fourth row in Figure 4a shows an example in which the classification indicates an *unobserved model* – the posterior is dominated by the unknown class, and the ambiguity measure indicates that this decision is highly unambiguous. In the Figure 4b we see another example of a *weak model*, however the model has become weak due to the a low certainty score s, which in effect raised the probability of the unknown model.



(a)



(b)

Figure 4: Knowledge models and the feature value **z** (left), a posteriori pdf over class labels (middle), pdf over the two hypotheses and the R_{hyp} score(right).

5.2.2 Illustrative example of filling the knowledge gaps

In the second illustrative example we present the knowledge gaps detection in the context of the active learning paradigm. We show how the particular detection of knowledge gap triggers a corresponding action, which provides the required information. Based on this information, the knowledge is updated and the knowledge gap is filled in.

An illustration of this process is shown in Fig. 5 for three observations (feature values) denoted by the circle, the diamond, and the square. The plots in the left column depict the models and the recognition at a particular step in the learning process, while the right column depicts the situation after the system has updated these models considering the detected knowledge gaps and the answers from the tutor.

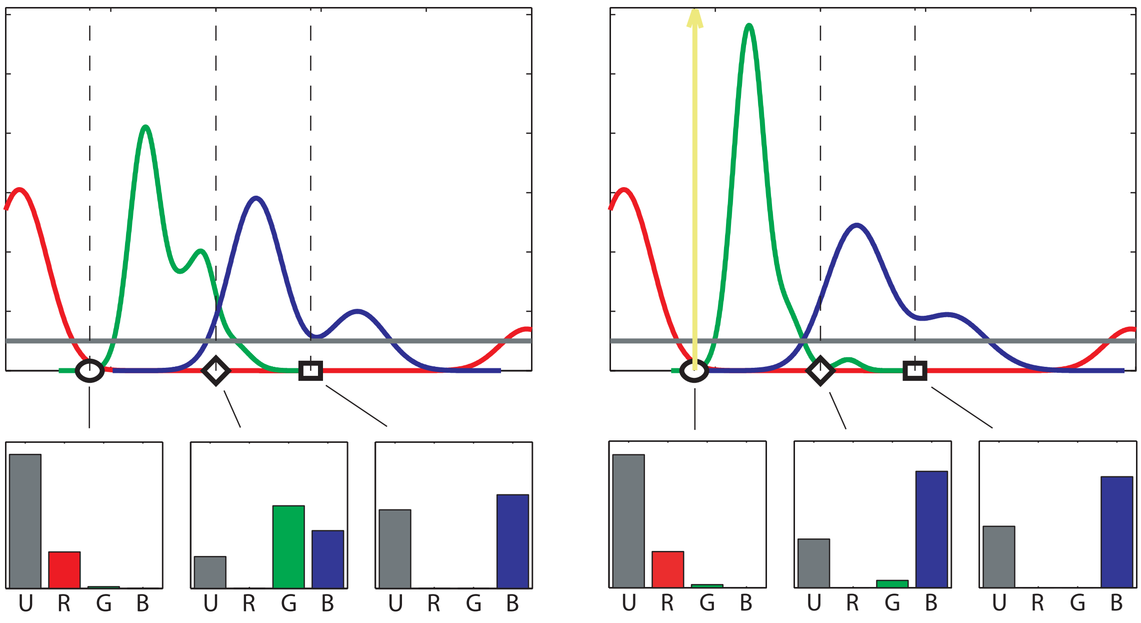


Figure 5: Example of detecting the knowledge gaps and updating the 1D KDE representations. Top row: probability distributions for three colours (red, green, blue lines) and unknown model (gray line) in 1D feature space. Bottom row: a posteriori probabilities for the unknown model (U) and three colours (R, G, B) for three feature values denoted by the circle, the diamond and the square. Left column: before updates, right column: after updates.

The circle represents a yellow object. Since the yellow colour has not been presented to the robot before, the corresponding model has not yet been learned and the feature value fails in a not yet modeled area, therefore this value is best explained by the "unknown model", which has the highest a posteriori probability. The '*No model*' knowledge gap is detected, which triggers the action of asking the tutor " *What colour is this object?*". After the tutor provides the correct information, the robot initialises a model for yellow colour. However, since only one sample does not suffice to build a reliable representation, the yellow colour will only be able to be recognised after some additional yellow object is observed.

The feature value denoted by a diamond is best explained by a green model, however this recognition is not very reliable, since the response of the model blue is similar. Therefore, the '*Ambiguous models*' knowledge gap is detected and the robot asks the tutor: "*Is this object green?*" to verify its belief. After the tutor replies "*No. It is blue.*", the robot first unlearns the representation of green and updates the representation of blue. The corrected representations, depicted in the pdfs in the right column, then enable the correct recognition as indicated by the second bar plot in the right column of the Fig. 5.

The last case denoted by the square shows another example of non-reliable recognition. In this case the model of blue responds best, however its response is quite similar to the response of unknown model. Therefore, the '*weak model*' knowledge gap is detected , which triggers the additional clarification question to the tutor: "*Is this object blue?*" After the robot gets a positive answer, it updates the representation of blue, which increases the probability of the recognition.

5.2.3 Comparison of gap discovery measures

As discussed in Section 4.2, the ambiguity of classifying some sample $\mathbf{z}$ (a potential gap) can be measured by calculating some measures over the a posteriori distribution of that sample. We have proposed three alternative measures of ambiguity and denoted them by ${}^{(1)}R_{hyp}$, ${}^{(2)}R_{hyp}$ and H. In classification of a sample $\mathbf{z}$ an ambiguity can be detected if either the sample is classified as the *unknown model* or if the ambiguity measure is above or below a specified threshold. To compare the different ambiguity measures, we have designed a simple experiment, in which 200 samples were generated from four partially overlapping classes, and used one at a time to update the odKDE model. Before the updating a sample was classified using the odKDE. If it was not classified as an unknown but was still classified into a wrong class, then this was considered as a *true positive* of the knowledge gap. Alternatively, if the sample was correctly classified, it was considered as a *true negative* of the knowledge gap. For all *true positives* and *true negatives* we have recorded the three ambiguity measures. This experiment was repeated 10 times and the Figure 6 shows the ROC curves corresponding to each of the three ambiguity measures. We see that from the gap-detection point of view, the measures are equivalent. Since the entropy-based measure H has roots in the information theory, we have decided to use this measure in subsequent experiments for gap detection. For convenience, we plotted some values of the measure H on the ROC in Figure 6.

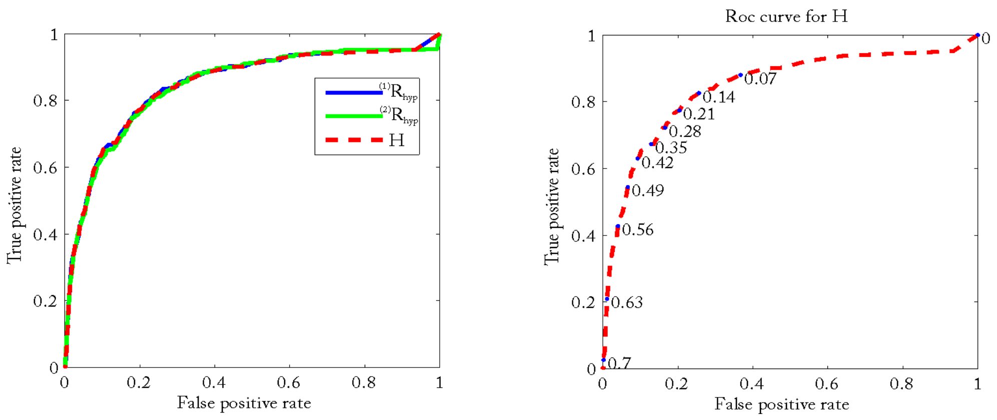


Figure 6: ROC curves for the three classifiers, corresponding to ${}^{(1)}R_{hyp}$, ${}^{(2)}R_{hyp}$ and H ambiguity measure (left), and a ROC curve corresponding to H measure along with some values of that measure (right).

When the robot observes a sample and classifies it using its internal models, a gap can therefore be detected if either the sample is classified as an unknown model or if the ambiguity measure H exceeds a predefined threshold H_{th}. For example, by setting

$H_{\mathrm{th}} = 0.1$, we can expect from the ROC curve in Figure 6 that we would be able to detect 85% of true gaps and 30% of false gaps.

5.3 From non-probabilistic class models to probabilistic knowledge models

The proposed methodology for discovering gaps in knowledge can be applied not only to inherently probabilistic models as in previous section but to non-probabilistic models as well. Here we provide some directions how to achieve that with an example of object detection/recognition.

Assume that we have a task of detecting objects from visual data and that, for each i-th object class, we have a prototype model $\mathbf{S}(M_i)$[1]. Assume that we have learnt K such models, which means that we have a set of K prototype models

$$\mathbf{S} = \{\mathbf{S}(M_i)\}_{i=1:K}. \tag{16}$$

The object detection proceeds by determining a query region of image which might contain an object and comparing this query with the object prototypes $\mathbf{S}$. We also have a distance function $d(\mathbf{S}(M_i), \mathbf{z})$, which calculates the distance of a query region to the object prototype $\mathbf{S}(M_i)$. If the distance $d(\mathbf{S}(M_i), \mathbf{z})$ is small enough, the region $\mathbf{z}$ is classified as the i-th object. The question now is how to determine the probabilistic knowledge model, i.e., how to meet the requirements summarised in the Section 5.1.

The a priori probability of observing the i-th known model, $p(M_i|m_{\mathrm{k}}, s)$ can be set to $\frac{1}{K}$ if all the models are equally probable, or can be estimated from the number of times we have encountered the class M_i during learning. The a priori probability $p(m_{\mathrm{u}}|s)$ that the observations will correspond to an unknown model can be set as in (**??**) and the pdf of the feature value $\mathbf{z}$ given the "unknown" model M_0 is set to a uniform distribution. The only distributions which remain to be estimated are the likelihood functions of each known i-th class, i.e., $p(\mathbf{z}|M_i, m_{\mathrm{k}}, s)$. These are in fact probabilistic models that relate the likelihood of M_i being responsible for generating $\mathbf{z}$, given the distance function value is $d(\mathbf{S}(M_i), \mathbf{z})$. Formally,

$$p(\mathbf{z}|M_i, m_k, s) = p(y_i|\Theta_i), \tag{17}$$

where $y_i = d(\mathbf{S}(M_i), \mathbf{z})$ and Θ_i are the parameters of the probability density function corresponding to the i-th model.

The functional form of $p(y_i|\Theta_i)$ is arbitrary and can be measured by evaluating the distance function $d(\mathbf{S}(M_i), \mathbf{z})$ for regions which contain the i-th model. A common approach is to use common parameters for all distributions, thus $\Theta_i = \Theta$ for all i. Under the assumption that low distances $d(\mathbf{S}(M_i), \mathbf{z})$ imply high probability of $\mathbf{z}$ corresponding to the model M_i and large distances imply low probability, a straightforward approach would be to define $p(\cdot|\Theta)$ as an exponential distribution, which requires estimating only a single parameter λ_d, e.g.,

$$p(y_i|\Theta) = {\lambda_d}^{-1} e^{-y_i/\lambda_d}. \tag{18}$$

The last requirement in Section 5.1 states that the models should enable hallucination. In practice this means that we have to be able to (i) determine the bounds on the feature values $\mathbf{z}$ and (ii) we have to be able to sample from the part of the feature space close to the learnt models. The bounds on observations $\mathbf{z}$ depend on the way in which the object

[1]For example, $\mathbf{S}(M_i)$ might be a collection of SIFT descriptors corresponding to the objects of class i.

model is specified and in case of SIFT features may be achieved perhaps by specifying the maximum amount of the random noise we can add to the SIFT histograms. Generating SIFTS from different parts of object model by adding random noise is then straightforward.

5.4 Learning domains

The formal model presented in Section 4 is very general and does not assume any strong constraints for the underlying representations. It only assumes that the underlying classifier is able to return the probability distribution over all concepts in question, or even only some confidence measures (and corresponding distance function), which are then converted in the probability distribution (also by modeling the class "unknown").

Due to the generality of the proposed formalism, it is applicable to different areas. Here, we discuss its application for learning object properties and object recognition.

5.4.1 Learning object properties

The formalism presented in the previous section is very well suited for learning object properties; also several examples given to illustrate the proposed principles are taken from this learning domain. If we assume that we have classifiers for visual properties we want to learn, we can use the proposed formalism for detecting knowledge gaps. Since we use for modeling individual concepts representations based on Mixture of Gaussians, the obtained representations fit very well with the proposed probabilistic framework.

5.4.2 Object recognition

The presented model can be used for object detection as described in Section 5.3. To successfully describe the probabilistic model, two problems have to be solved: the definition of the distance function $d(S(M_i), \mathbf{z})$ and the modelling of the "unknown" object.

The object model $S(M_i)$ is represented with a set of views of the object from different viewpoints, $S(M_i) = \{V_j, j = 1 \dots N_i\}$. A view V_j^i is described with viewpoint angles (ϕ, λ) and a set of features $\{F_k, k = 1 \dots N_{i,j}\}$ extracted from the image taken at this viewpoint. A feature $F_k^{i,j}$ is a SIFT feature described with its location (x, y) in the image, scale σ, orientation θ and a 128-dimensional vector $\mathbf{X}_k^{i,j}$.

An observation $\mathbf{z}$ is represented with a set of features extracted from an image: $Z = \{F_k^{\mathbf{z}}, k = 1 \dots N_z\}$. When matching an object M_i to an observation, the Euclidean distance between each pair of feature descriptors is calculated:

$$d_f(k, l) = d(\mathbf{X}_k^{i,j}, \mathbf{X}_l^{\mathbf{z}}), k = 1 \dots N_{i,j}, l = 1 \dots N_z \tag{19}$$

The distances between a single view V_j^i and the observation $\mathbf{z}$ are converted to a view score D_j^i. The minimal distance d_0 is used to set an upper bound for the contribution of each matched feature to the final score. It can be either constant or estimated for each view in the learning phase.

$$D_j^i = \sum_{k,l} \frac{1}{max(d_0, d_f(k, l))} \tag{20}$$

If we take the view with the highest score D^i_{max}, then the appearance distance $d(S(M_i), \mathbf{z})$ can be expressed as

$$d(S(M_i), \mathbf{z}) = \frac{1}{D^i_{max}} \tag{21}$$

To estimate the PDF for $d(S(M_i), \mathbf{z})$ we make a bunch of observations $\mathbf{z}_k$ of the object M_i in different poses with the camera at the approximate distance r_0 from which the object model was learnt. A histogram of appearance distances is created and an exponential curve is fitted to the histogram and scaled appropriately (see Eq. (18)).

To build an observation model for the object M_i, the observations $\mathbf{z}_k$ are additionally made at different distances from the object. For each observation $O_k = (\phi_o, \lambda_o, r_o)$ the viewpoint and score $P_b = (\phi_b, \lambda_b, D_b)$ of the best matching view are recorded and also the viewpoint and score $P_a(\phi_a, \lambda_a, D_a)$ of each matching view where score $D_a > 0.5 * D_b$ to account for pose ambiguity.

6 Experimental results

To gain a better insight into the performance of gap discovery, we have applied the methodology to active learning of simple object properties. In this section we present preliminary experimental results.

The task was to learn the models for four colours from the observed objects. We have used a database of 442 images containing single objects in which each object was of one of four colours. From each object a six-dimensional feature vector was extracted – this was the input to our learning algorithm. One object at a time was used to train the classifier. For the online classifier we have used the online discriminative Kernel Density Estimator [10], which was integrated according to 5.2 into the probabilistic knowledge model with the entropy measure H as a gap detector. The experiment was performed with five-fold cross validation – in each fold, 354 images were used for learning and 88 for testing the accuracy of recognition.

We have explored four learning strategies: *oracle random*, *oracle proactive*, *self verified* and *self verified proactive.*

In the *oracle random* strategy, the tutor (oracle) randomly choose a learning sample and always provided a label. The learner learned from each sample. In this learning strategy, no knowledge gap detection was involved.

In the *oracle proactive* strategy, the learner inspected its internal models through introspection, determined which class it would like to see next, and asked the tutor to present him with a sample from that class. Again the learner learned from each sample, but this time the learner guided the concept class selection, not the tutor. Therefore, in this strategy the active learning process was driven my *introspection* using *model analysis*, as defined in Section 3.1.

In the third *self verified* strategy, the tutor as in *oracle random* randomly selected samples for the learner, but did not label them. For each sample, the learner performed gap detection and asked the tutor about the label only if the gap was detected. This case, therefore, applies the *situational* type of knowledge gap detection.

The fourth, *self verified proactive*, strategy was similar to the *self verified* strategy with the addition that the learner was able to introspect its models and requested specific classes from the tutor (as in *oracle proactive*).

Figure 7 shows the results of the experiment with the five-fold cross validation. We see that the active involvement of the learner in the learning process by gap detection improved the trend of recognition accuracy. By comparing how the recognition accuracy evolved with the number of samples (Figure 7a), in the case of *oracle random* and *oracle proactive* strategy, we see that the model introspection allowed the learner to determine which models are ambiguous and successfully guided the tutor to select samples from those classes that improved the models. As a result, the learner with *oracle proactive* learning strategy achieved better accuracy during the first hundred samples than the learner with the *oracle random* strategy. However, both strategies required the tutor to label all the samples, which demands a certain effort from the tutor. This is shown in (Figure 7b), which plots the cumulative number of the labelled samples with respect to the number of observed samples.

On the other hand, a significant reduction in the number of labels was obtained in the *self verified* learning strategy. From Figure 7b we can see that in contrast to *oracle random* and *oracle proactive* strategy, the number of labelled samples increased sublinearly in the *self verified* strategy. In fact, up to the 150th sample, the trend was already sublinear (meaning that the learner required labels only for a subset of samples), and after that point it remained nearly constant (meaning that almost no additional label was required). Even further reduction in the number of required labels was achieved in the *self verified proactive* strategy. This strategy also achieved best performance with the smallest number of explicitly required labels.

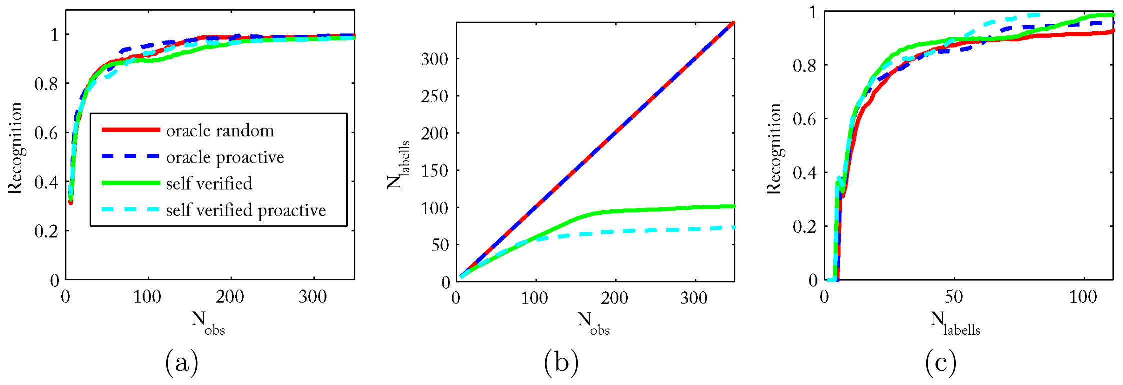


Figure 7: (a) Evolution of average recognition accuracy w.r.t the number of observed samples for the four learning strategies. (b) Evolution of the number of questions w.r.t. the number of observed samples. (c) The average recognition accuracy for the first 111 questions.

Experiment shows that, on average, the learner with *self verified* strategies required approximately 100 samples to be labelled out of the entire set of 354. This means that the *self verified* learning strategy with gap detection required only 27% of the labels to achieve equal performance to the *oracle random* and *oracle proactive* strategy which required 100% of labels. The dependance of the recognition accuracy and each provided label for the three strategies is show in Figure 7c. We see that both, *oracle proactive* and *self verified* strategy, outperform the *oracle random* strategy for the first 95 labelled samples. The improved performance is exclusively due to the probabilistic formulation of the knowledge model and the mechanisms for the knowledge gap discovery. These preliminary results therefore show that the ability to detect knowledge gaps can lead to an efficient active learning strategy, which speeds up the learning process.

7 Conclusion

A crucial requirement of a system that is supposed to self-extend based on self-understanding is the ability to detect gaps in its knowledge. Once these gaps are detected, it can plan what actions to perform to fill these gaps, and after the required information is obtained, it can extend its current knowledge accordingly.

In this work we addressed the problem of detection of gaps in categorical knowledge. First, we positioned the problem of knowledge gap detection in a broader context of self-understanding for self-extension and active learning paradigm. Then we conceptually described different types of knowledge gap detection and different types of knowledge gaps. Then we formalised this process and presented a formal probabilistic approach to detection of ignorance. We specified the requirements of the probabilistic models and showed how to apply the proposed mechanism for knowledge gap detection to different learning domains. We also presented the experimental results that demonstrate that the effective knowledge gap detection mechanism can lead to an efficient active learning strategy, which speeds up the learning process.

The model, presented in this work, seems to perform well and can efficiently guide the learning process. We still need to perform a thorough evaluation and analyse its performance, scalability, and robustness. We will integrate it in our interactive continuous learning framework [18], where it will serve as a main drive for guiding different learning strategies exploiting different levels or the robot autonomy. In addition, we have been integrating this mechanism in the robotic system we have been developing [17, 20, 19]. This work will serve as a firm basis for increasing the robot's autonomy and efficiency in determining its own ignorance, therefore for increasing the ability to self-understand and act autonomously. This will be an important step towards our final goal, which is to produce an autonomous robot that will be able to efficiently learn and adapt to an ever-changing world by capturing and processing cross-modal information in an interaction with the environment and other cognitive agents.

References

[1] C. Campbell, N. Cristianini, and A. Smola, *Query learning with large margin classifiers*, Proceedings of the Seventeenth International Conference on Machine Learning, 2000, p. 111118.

[2] N. Cebron and M. R Berthold, *Active learning for object classification: from exploration to exploitation*, Data Mining and Knowledge Discovery **18** (2009), no. 2, 283299.

[3] A. Chao, *On estimating the probability of discovering a new species*, The Annals of Statistics **9** (1981), no. 6, 1339–1342.

[4] D. Cohn, L. Atlas, and R. Lander, *Improving generalization with active learning*, Machine Learning **15** (1994), no. 2, 201221.

[5] W. Fan, Y. Huang, H. Wang, and P. S Yu, *Active mining of data streams*, Proceedings of the Fourth SIAM International Conference on Data Mining, 2004, pp. 457–461.

[6] A.D. Holub, P. Perona., and M. Burl, *Entropy-based active learning for object recognition*, Workshop on Online Learning for Classification, in conjunction with Conf. Comp. Vis. Pattern Recognition, 2008, pp. 1–8.

[7] S. Huang and Y. Dong, *An active learning system for mining time-changing data streams*, Intelligent Data Analysis **11** (2007), no. 4, 401419.

[8] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, *Active learning with gaussian processes for object categorization*, International Conference on Computer Vision, 2007.

[9] M. Kristan and A. Leonardis, *Multivariate online kernel density estimation*, Computer Vision Winter Workshop, 2010, pp. 77–84.

[10] ———, *Online discriminative kernel density estimation*, International Conference on Pattern Recognition, 2010.

[11] M. Kristan, J. Perš, M. Perše, and S. Kovačič, *A Bayes-spectral-entropy-based measure of camera focus using a discrete cosine transform*, Pattern Recognition Letters **27** (2006), no. 13, 1419–1580.

[12] M. Kristan, D. Skočaj, and A. Leonardis, *Online kernel density estimation for interactive learning*, Image and Vision Computing **28** (2010), no. 7, 1106–1116.

[13] A. McCallum and K. Nigam, *Employing EM in pool-based active learning for text classification*, Proceedings of the fifteenth international conference on machine learning, 1998, pp. 350–358.

[14] H. T Nguyen and A. Smeulders, *Active learning using pre-clustering*, Proceedings of the twenty-first international conference on Machine learning, 2004, p. 79.

[15] G. Schohn and D. Cohn, *Less is more: Active learning with support vector machines*, 2000, p. 839846.

[16] H. S. Seung, M. Opper, and H. Sompolinsky, *Query by committee*, Proceedings of the fifth annual workshop on Computational learning theory, 1992, p. 287294.

[17] D. Skočaj, M. Janiček, M. Kristan, G.-J. M. Kruijff, A. Leonardis, P. Lison, A. Vrečko, and M. Zillich, *A basic cognitive system for interactive continuous learning of visual concepts*, Proceedings of the ICRA 2010 Workshop on Interactive Communication for Autonomous Intelligent Robots (ICAIR) Making robots articulate what they understand, intend, and do. (Anchorage, AK, USA), May 2010.

[18] D. Skočaj, M. Kristan, and A. Leonardis, *Formalization of different learning strategies in a continuous learning framework*, EPIROB'09, 2009, pp. 153–160.

[19] D. Skočaj, M. Kristan, A. Leonardis, M. Mahnič, A. Vrečko, M. Janíček, G.-J. M. Kruijff, P. Lison, M. Zillich, C. Gretton, M. Hanheide, and M. Göbelbecker, *A system approach to interactive learning of visual concepts*, 2010, Submitted.

[20] J. L. Wyatt, A. Aydemir, M. Brenner, M. Hanheide, N. Hawes, P. Jensfelt, M. Kristan, G.-J. M. Kruijff, P. Lison, A. Pronobis, K. Sjöö, D. Skočaj, A. Vrečko, H. Zender,

and M. Zillich, *Self-understanding & self-extension: A systems and representational approach*, IEEE Transactions on autonomous mental development (2010), Accepted for publication.

[21] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang, *Representative sampling for text classification using support vector machines*, Proceedings of the 25th European conference on IR research, 2003, pp. 393–407.

[22] X. Zhu, P. Zhang, X. Lin, and Y. Shi, *Active learning from stream data using optimal weight classifier ensemble*, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on (2010), no. 99, 1–15.

On the automatic Entropy-based construction of Probabilistic Automata in a Learning Robotic Scenario

Sergio Roa and Geert-Jan Kruijff
German Research Center for Artificial Intelligence / DFKI GmbH
{sergio.roa,gj}@dfki.de

***Abstract*— When a robot interacts with the environment producing changes through its own actions, it should find opportunities for learning and updating its own models of the environment. A robot that is able to construct discrete models of the underlying dynamical system which emerges from this interaction can guide its own behavior and adapt it based on feedback from the environment. Thus, the induction of probabilistic automata from this sensorimotor loop might be useful for planning/learning tasks. These probabilistic automata can be used as a prediction tool, as a means to assess the uncertainty or predictability of specific action consequences and thus, as a tool for an active learning method.**

I. Introduction

In the face of continuously changing environments, a robot needs to learn from the world by interacting with it. A robot should then accurately predict the consequences of actions on an object given its own body configuration. With the acquired skills knowledge, it can then solve more complex tasks in a hierarchical manner. We consider here the task of predicting consequences of pushing simple geometrical objects called *polyflaps*. Polyflaps have been proposed to design simple learning scenarios [1].

In this paper we discuss a scenario where a simulated robotic arm interacts with a polyflap. In the implementation we use the NVidia® PhysX$^{\text{TM}}$ library that allows us to perform realistic physical simulations and to obtain 3-dimensional feature vectors. The learning machines we use are able to process spatio-temporal features. Specifically, we use the Crystallizing Substochastic Sequential Machine Extractor (CrySSMEx) [2] algorithm for the extraction of probabilistic finite state automata. The robotic arm pushes the object and then a sequence of polyflap poses is stored, encoded as rigid body transformations during a certain time interval. To reduce the space- and time complexity of the problem, we select a discrete set of possible actions and starting positions for the arm to start the pushing movement. This reduction of dimensionality affords us also to evaluate and analyse more easily and carefully the inference algorithm and its corresponding outcomes. In general, *sliding* and *flipping* affordances are obtained by applying pushing actions. The experiments show that the machines are able to model the tuples $\langle$action, state, output, next state$\rangle$.

But how can we find state abstractions? Since we want to understand the behaviour of an object given a specific action, it is reasonable to encode a state on the basis of polyflap poses (rigid body transformations). A sequence of polyflap poses will then be encoded as a sequence of states (not necessarily a 1 to 1 relation), where transitions depend on the corresponding action. Moreover, state abstractions can possibly be obtained not only from quantized vectors but also from the output dynamics. Given the complexity of these physical processes, transition probabilities between states are also quite possible and an information-theoretic measure is used to help quantize the space and to generate the transitions by using the CrySSMEx algorithm. The extracted automata are useful as a verification tool, that can be employed by active learning techniques in order to evaluate uncertainty in specific actions.

II. Learning Scenario

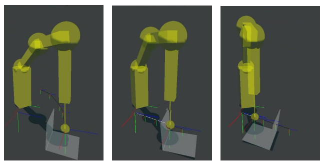


Fig. 1. Learning scenario with a polyflap

The learning scenario is shown in Fig. 1. The simulated arm corresponds to a Neuronics® Katana 6M$^{\text{TM}}$ arm with a ball as a simple finger. In order to simulate a pushing action we apply a linear trajectory over a specified time period until it reaches the desired pose. The arm has 6 joints, including the last joint for the finger which is static. The representation of object poses are in Euler angles with respect to a reference frame which is the origin in the scene (6-D pose).

The features corresponding to the arm are a starting 6-D pose vector for the finger arm $\mathbf{f}$, a feature value s encoding 3 velocity values (low, medium, high), obtained from applying 3 different movement durations, and an integer value denoting a direction angle Θ ranging from 60 to 120 degrees, parallel to the ground plane in the direction to the center of the standing polyflap side. Together, these features form the motor command feature vector denoted as $\mathbf{m}_t$ at time t. The values are all normalized to obtain vectors with mean 0 and standard deviation 1.0. A 6-D pose vector corresponding to the polyflap pose is denoted as $\mathbf{p}_t$. In order to perform preliminary experiments and avoid ambiguities and difficulties in analyzing data (cf. section III), we artificially enumerate (discretize) the set of possible actions. Therefore, we obtain a symbolic feature m_t denoting the motor information. In the enumeration, we encode the time step t into the symbol m_t in order to discretely represent the finger transformations (change of poses) through an action sequence. Moreover, CrySSMex requires that we define an output label associated to each state or transition. This feature is very useful for evaluation and also for the convergence of the CrySSMEx algorithm itself (cf. section III). We then define the set $Y = \{-1, 1, 0, -0.5, 0.5\}$ of possible values for an output symbol y_t, denoting respectively when θ polyflap angle decreases (this happens when the polyflap tilts but does not completely flips over, so that it returns to the original angle), θ angle increases (polyflap falling down), polyflap does not move, polyflap moves backwards (negative X direction without angle change) and polyflap moves forwards. Thus, the tuples $\langle m_t, \mathbf{p}_t, y_t \rangle_{t=1}^{n}$ encode the rigid body transformations of polyflaps through these n steps and also encode the given robot control command and abstract behaviour after the pushing movement. In order to discretize and reduce the dimensionality of the task, we only used 18 different starting positions for the arm to start the pushing movement.

III. Automata induction method

Let us define a substochastic sequential machine (SSM) as a quadruple $\langle Q, M, Y, \mathcal{P} = \{p(q_j, y_l | q_i, m_k)\} \rangle$ where Q is a finite set of state elements (SEs), M is a finite set of input symbols, i.e. our motor command representation, Y is a finite set of output symbols, and $\mathcal{P}$ is a finite set of conditional probabilities (cf. explanation in [2] and eqs.1-3) where $q_i, q_j \in Q, m_k \in M$ and $y_l \in Y$. In practice, CrySSMEx can automatically induce discretizations of the input and output spaces by means of quantizer functions on these spaces but we will avoid this step in order to not introduce more difficulties in analyzing data and to increase the chance of convergence of the CrySSMEx algorithm. Thus, we assume the artificial enumeration for input and output spaces described in section II and we use the vector quantization method described in [2] for state discretization. A SSM models a situated discrete time dynamical system for which input (M), output (Y) and state (P) spaces are defined and a transition function $\gamma : P \times M \to P \times O$. A stochastic dynamical model of such a system is a joint probability mass function p_Ω induced from a transition event set Ω, and quantizers Λ_y, Λ_m and Λ_p for output, input and state spaces respectively. Ω consists of selected transition events recorded from a given set of input sequences. Thus, the joint probabilities of observed and quantized transitions (p_Ω) are translated into joint probabilities of SSM transitions according to $\mathcal{P}$. As already mentioned, we define $\Lambda_m(\mathbf{m}_t)$ and $\Lambda_y(t)$ according to the discretization described above and $\Lambda_p(\mathbf{p}_t)$ using the vector quantization method. Thus, we have:

$$\begin{gathered} p(q_i, m_k, y_l, q_j) = \\ p_\Omega(\Lambda_p(\mathbf{p}_t) = i, \Lambda_m(\mathbf{m}_t) = k, \Lambda_y(t+1) = l, \Lambda_p(\mathbf{p}_{t+1}) = j) \end{gathered} \tag{1}$$

The conditional probability is calculated with:

$$p(q_i, m_k) = \sum_{j=1}^{|Q|} \sum_{l=1}^{|Y|} p(q_i, m_k, y_l, q_j) \tag{2}$$

$$p(q_j, y_l | q_i, m_k) = \begin{cases} \frac{p(q_i, m_k, y_l, q_j)}{p(q_i, m_k)} & \text{if } p(q_i, m_k) > 0 \\ 0 & \text{if } p(q_i, m_k) = 0 \end{cases} \tag{3}$$

The substochasticity of the extracted machines is due to the possibility that the sample of input sequences in Ω will not necessarily provide examples of all possible input symbols in all possible enumerations of the quantized space of the dynamical system. As a consequence, the probability distributions can become substochastic [2]. The details of the procedure for extracting substochastic sequential machines is described in [2]. In summary, there is a recursive state splitting starting from only one SE. Then, a decision to split data into different SEs is based primarily on the output entropy $H(Y|Q = q_i, M = m_k) = H(\mathcal{P}_y(q_i, m_k))$ and then on the next state entropy $H(Q|Q = q_i, M = m_k) = H(\mathcal{P}_q(q_i, m_k))$, i.e., choosing state vectors that convey the most information (i.e., highly indeterministic) [2], where $H(\mathcal{P}) = -\sum_{i=1}^{n} p_i \log p_i$ and $p(q_{t+1}) = \mathcal{P}_q(q_i, m_k)$ and $p(y_{t+1}) = \mathcal{P}_y(q_i, m_k)$ are marginal distributions of $\mathcal{P}$. Additionally, states are possibly merged if there exists an equivalence relation between two states based on determining when two SEs are not equivalent if they, in their outgoing transitions, share some input symbols and transitions that lead to discrepancies in the future output. The procedure finishes when the machine is deterministic, i.e., when the entropies for all states equal to 0.

IV. Preliminary experimental results

We extracted an event set consisting of 90 actions. The algorithm converges after 45 iterations, when 74 states were obtained. Due to space constraints we do not show the extracted automaton, but for illustration purposes we show an automaton (Moore machine) extracted from 5 actions (cf. fig. 2) after convergence. Not all possible input symbols (in the boxes) are shown.

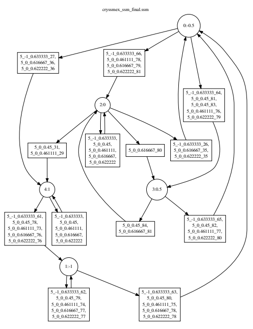


Fig. 2. Extracted finite state automaton from 5 actions. Circles are states and boxes contain input symbols

References

[1] A. Sloman, "Polyflaps as a domain for perceiving, acting and learning in a 3-D world," in *Position Papers for 2006 AAAI Fellows Symposium*. Menlo Park, CA: AAAI, 2006. [Online]. Available: http://www.cognitivesystems.org/publications/Fellows16.pdf

[2] H. Jacobsson, "The crystallizing substochastic sequential machine extractor - CrySSMEx," *Neural Computation*, vol. 18, no. 9, pp. 2211–2255, 2006.