Visualization of Processes in Self-Learning Systems

Gabriele Peters*, Kerstin Bunte†, Marc Strickert‡, Michael Biehl§ and Thomas Villmann¶
*Human-Computer Interaction, FernUniversität in Hagen, Germany
E-mail: gabriele.peters@fernuni-hagen.de
†CITEC-Cognitive Interaction Technology Center of Excellence, Bielefeld University, Germany
E-mail: kbunte@techfak.uni-bielefeld.de
‡Department of Computer Science and Mathematics, University of Marburg, Germany
E-mail: marc.strickert@uni-marburg.de
§Johann Bernoulli Inst. for Mathematics & Computer Science, University of Groningen, The Netherlands
E-mail: m.biehl@rug.nl
¶Computational Intelligence Group, University of Applied Sciences Mittweida, Germany
E-mail: villmann@hs-mittweida.de

***Abstract*—One aspect of self-organizing systems is their desired ability to be *self-learning*, i.e., to be able to adapt dynamically to conditions in their environment. This quality is awkward especially if it comes to applications in security or safety-sensitive areas. Here a step towards more trustful systems could be taken by providing transparency of the processes of a system. An important means of giving feedback to an operator is the visualization of the internal processes of a system. In this position paper we address the problem of visualizing dynamic processes especially in self-learning systems. We take an existing self-learning system from the field of computer vision as an example from which we derive questions of general interest such as possible options to visualize the flow of information in a dynamic learning system or the visualization of symbolic data. As a side effect the visualization of learning processes may provide a better understanding of underlying principles of learning in general, i.e, also in biological systems. That may also facilitate improved designs of future self-learning systems.**

***Keywords*-Visualization of dynamic processes, transparency and controllability of self-organization processes, self-learning systems, symbolic representations**

I. Introduction

A frequent demand on self-organizing systems is their capability to react and adapt dynamically to conditions and events in their surroundings. For these acquirements they have to be able to *learn* autonomously from the environment, i.e., they have to be *self-learning*. This quality is delicate especially if it comes to applications in security or safety-sensitive areas, because the flexibility of self-learning may seem to be conficting with security and safety requirements. Here a step towards more trustful systems could be taken by providing insight to the self-organizing processes of the system. Both, usability and controllability of such a system can be increased by giving feedback to the operator on the internal status of the system [1], [2]. This includes the knowledge the system has learned so far but also information on the learning process itself, i.e., the learning characteristic. An important means for system monitoring is the visualization of the internal processes which, in addition, also facilitates interaction with the system [3]. In the field of machine learning one can distinguish at least two fundamentally different approaches. On the one hand, methods exist which represent learned knowledge in *symbolic* form, such as realized by *belief revision* [4]. Here the result of the learning process is given explicitly, e.g., in the form of *it-then rules*. On the other hand, there are learning approaches that represent learned knowledge in subsymbolic, i.e., *numeric* form, such as realized by *reinforcement learning* [5]. Here the result of the learning process is given only implicitly, e.g., in the form of a *table of numbers*. The main benefits of a combination of symbolic and numeric representations in self-learning systems are described in [6]. Biological learning systems inspire the formulation of further requirements for self-learning systems. Among them is the demand that learning should take place at several *levels of hierarchy* and that a *multi-directional transfer* of information between the separate levels is necessary to acquire cognitive capabilities autonomously from the environment. The current situation in classical machine learning is characterized by a rather strict separation of symbolic and numeric learning approaches, although results from, e.g., psychological research also provide evidence that humans are able to learn implicitly as well as explicitly and that these levels of learning interact with each other [7]. Only recently these fields of machine learning slowly begin to merge, taking advantage of the mutual benefits of both concepts. One such approach that combines both ideas is the *Sphinx* system described in [8], [9]. We take this system as an example and a starting point to develop our ideas on the visualization of learning processes. Despite of strong research efforts no broad intelligent, self-organizing system exists which features essential properties of biological learning systems (such as the ability to explore the environment and to learn the inherent structure of perceived data) and at the same time is able to solve complex problems (such as cognitive tasks in computer vision). The few

approaches (including the above mentioned system), which try to implement the above mentioned learning principles, only provide first decent results in autonomously gaining cognitive abilities. Consequently, it is not surprising that - so far - no attempt has been made to visualize the internal processes of such a broad learning system. An inherent challenge for the development of such systems is the limited access to the information flow between different learning levels. Visualization of dynamically attained system states helps to monitor and identify transition patterns important for the system design task.

In this position paper we address the problem of visualizing processes in self-learning systems. The purpose of such a visualization is at least twofold. On the one hand, it represents one of the best possibilities to improve the design and safety of a system by increasing its transparency, its usability and its possibilities for user interaction. On the other hand, visualizations of processes in artificial learning systems may also allow for insights in learning mechanism in general, including biological systems. In section II we introduce briefly the learning system we refer to in this paper. In section III we first identify relevant questions which are able to open future research directions in the field of visualization of dynamic learning processes and give an example of a concrete visualization of a learned symbolic representation in the form of rules.

II. Example for a Self-Learning System

In this section we give a brief overview of the self-learning system *Sphinx* we will take as an example to introduce our suggestions for visualizations. We restrict ourselves to describing only those aspects of the system, which are relevant for this purpose. For details refer to [8]. The iterative two-level architecture of *Sphinx* is displayed in figure 1. *Sphinx* is able to autonomously learn strategies for visual object acquisition and recognition from scratch. For that purpose the system interacts with its environment by rotating objects depending on past perceptions to acquire those views which are advantageous for recognition. The goal is that after the learning process the system is able to recognize objects with as few actions as possible. *Sphinx* is endowed with some simple visual recognition mechanisms, such as the categorization of object textures into the three different categories *simple*, *medium* and *complex*. In the beginning of the learning process a set of unfamiliar objects (such as a bottle) is presented. *Sphinx* starts to chose whatever object it likes, rotates the object as often as it likes to any view it likes, and perceives the visual parameters of the view. This is repeated several times. During the learning process a set of *if-then rules* is learned that allows for an increasingly efficient acquisition process, i.e., an increasingly goal-oriented selection of views for decisions for the next action such as the rotation to a particular view or the recognition of a special object. The if-then rules are ranked according to their plausibility, with increasing ranks for decreasing plausibilities, starting with a rank of 0 for the most plausible rules. During the learning process new rules can be created, existing rules can be deleted or modified, and the ranks of rules can be adapted. In the end of the learning process *Sphinx* displays high recognition rates using its learned rules. Examples for learned rules are

- *IF the shape of the front view is an upright triangle AND the size of the front view is tall THEN recognize a bottle.*
- *IF the shape of the front view is a circle AND the shape of the side view is unknown AND the texture is simple THEN rotate the object to the left.*

A. System Parameters of the Low-Level Learning Part

The lower learning level of *Sphinx* is realized by methods of reinforcement learning, i.e., *Q-learning* [5]. Learning proceeds over several episodes. One episode consists of serveral steps in which a state of the environment is perceived, an action is performed, a reward is received for the last action, and a Q-table is updated using the received reward. (Figure 1 shows theses stages of one learning step.) Hence, the central data structure on the lower learning level is the Q-table, which is a sparsely populated table with quality values of state-action pairs. This is the main data structure for the numeric representation of learned knowledge.

B. System Parameters of the High-Level Learning Part

The above mentioned rules are derived from an interplay between the rewards received on the lower learning level and a central data structure on the higher level which is learned simultaneously, namely the *ordinal conditional function* (OCF). This function assigns a rank to each world model according to its plausibility. A world model is a conjunction of literals of all possible variables. The OCF is the main data structure for the symbolic representation of learned knowledge. The symbolic representations of the states are constituted by values of visual parameters that characterize an object. In detail, that are the 7 boolean variables summarized in table I with their possible values in parentheses. (The deviation variables code the deviations from the idealized shapes.) The symbolic representation s

Table I
VARIABLES

variable	possible semantic
1 *FrontViewShape*	{*Unknown*, *Circle*, *Square*, *TriangleUp*, *TriangleDown*}
2 *FrontViewSize*	{*Unknown*, *Flat*, *Regular*, *Tall*}
3 *FrontViewDeviation*	{*Unknown*, *Little*, *Medium*, *Much*}
4 *SideViewShape*	{*Unknown*, *Circle*, *Square*, *TriangleUp*, *TriangleDown*}
5 *SideViewSize*	{*Unknown*, *Flat*, *Regular*, *Tall*}
6 *SideViewDeviation*	{*Unknown*, *Little*, *Medium*, *Much*}
7 *Texture*	{*Unknown*, *Simple*, *Medium*, *Complex*}

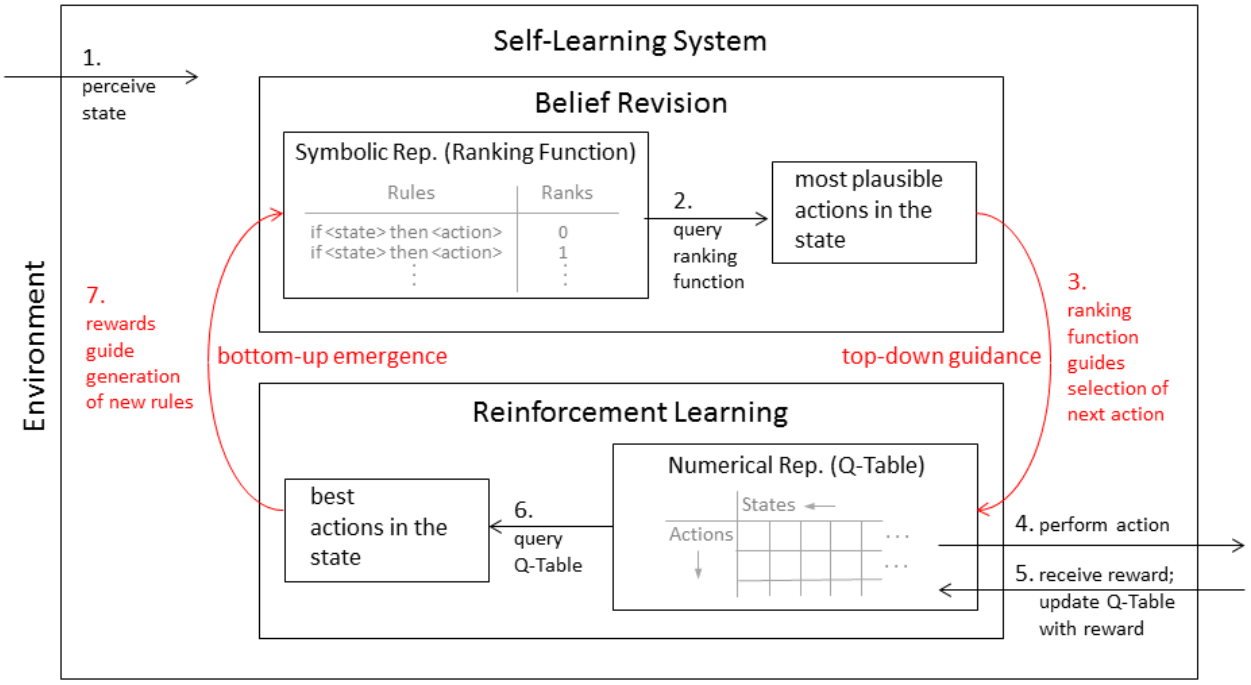

Figure 1. The seven stages of a single learning step of the self-learning system *Sphinx*. Learning takes place on *two levels of hierarchy*, and there is an *exchange of information* between both levels in both directions. The subsymbolic level is realized by means of reinforcement learning. The knowledge learned here are values of state-action pairs and it takes the *numeric* form of a *table of numbers*. The symbolic level of *if-then rules* is realized by means of belief revision. Before the execution of the very first learning step neither any symbolic knowledge nor any numeric knowledge has been acquired, as detailed in [8].

of a specific state is the conjunction of the corresponding literals of all of these variables.

The possible actions are two rotation actions (to the left and to the right), one recognition action for each of 9 objects

- $Recognize = \{Ball, Bird, Bottle, DVD, Football, House, Pott, TetraPack, Tree\}$

and the decision to not recognize anything. Thus, here we have one variable with 12 possible values:

- $Action = \{RotateLeft, RotateRight, RecognizeUnkown\} \cup R$, where R is the set of '*Recognize*' actions.

Finally, a world model which is ranked by the OCF during the learning process takes the form of conjunctions of literals, where the first part is a symbolic representation s of a state as described above and the last literal is a symbolic representaion a of an action. For the interpretation, s constitutes the predicate of an if-then rule, whereas a constitutes the consequent, with the meaning, that if such a rule is ranked plausible, then action a is a plausible action in state s.

III. Questions of General Interest

Our goal is the visualization of dynamic processes in learning systems in general. Future learning systems will be most successful, as explained in section I, if they combine symbolic and numeric knowledge on different hierarchy levels. The questions of general interest in this context concern the visualization of the dynamic transfer of information and of the symbolic data.

In our example the status of the self-learning system *Sphinx* is defined, on the one hand, by the OCF (i.e., the list of learned rules with plausibility ranks, values of zero indicating highest trustworthiness) and, on the other hand, the Q-table (i.e., a sparsely populated table or matrix of qualities of state-action pairs).

For the visualization of numeric data, especially of matrices of values, a number of suggestions already exist. For this reason we concentrate in the following subsections on general considerations concerning the visualization of the dynamic flow of information in self-learning systems (subsection III-A) and on a concrete example of a visualization of lists of rules of the *Sphinx* learning system (subsection III-B).

A. General Considerations on the Visualization of Self-Learning Systems

Internal states of self-learning systems are driven by external stimuli from the environment but also from internal

states that account for previous recursively dependent system states. A rigorous mathematical analysis would opt for quantification of partial auto-correlation characteristics or the identification of attractor states [19]. Semi-quantitative, yet informative, visualization-based characterization of dynamical systems is possible using the visual recurrence analysis approach [20]. This method characterizes quasi-periodic states and phase transitions by seeking pairs of states vectors over time being marked in a so-called recurrence matrix suitable for visual assessment and interpretation. Restriction to rather small data sets, complete data knowledge, and real-valued uni-variate input data prevent us from using visual recurrence analysis; however, the strong idea of relying on neighborhood relations is used for visual local reconstructions of state relationships in this work.

1) Visualization graphs: Chart plots against time are commonly useful in process characterization if system goals can be defined. For example, rates of learning success or internal stimulus representation accuracies can be computed and plotted during different learning phases of the model. If such externally quantifiable goals are not available, measures of structural complexity can be calculated. The numbers of active rules per learning episode or, more precisely, the activity distribution and credibility of rules can be used for model characterization. Progress is then defined relative to previous system states by calculating differential values such as relative entropy measures such as the Kullback-Leibler divergence [21].

2) Visualization of internal states: The relationships of internal states of a developing system are the result of a process, thus, they are supposed to be more meaningful relative to the previous stages rather than to a global origin. Moreover, the combination of symbolic and subsymbolic architectures requires methods that are able to deal with internal representation ranging between numeric and categorial data. A very general approach is the definition of scoring schemes to assess the similarity of internal states. For numeric data, such as the Q-table, this can be done using norms for potentially sparse vectors. For symbolic string data it is more natural to employ edit-distances, alignment scoring strategies, or Hausdorff metrics [13], which for general boolean rules would require prior canonicalization using algebraic normal forms (Zhegalkin polynomials) [14]. Alternatively, Karnaugh-Veitch-Diagrams can be used to simplify Boolean algebra expressions [15]. Once the (dis-)similarity measure is defined, it is possible to extract rankings of state neighbours for use with relational data processing tools [16].

Visualization methods like stochastic neighbor embedding (SNE) allow to use information on object neighborhoods to re-create their approximations in simple-structured spaces, such as 2D-scatter plots [10], [11]. These plots can be equipped with additional labeling information for printing or further be used in interactive systems like GGobi for the visual inspection of attained state configurations [12].

Table II
CHARACTER CODES OF THE RULES IN SPHINX AND THEIR ANNOTATION.

character	denotation	character	denotation
A	Unknown	M	Pott
B	Tall	N	Triangle Down
C	Little	O	Tree
D	Complex	P	Much
E	Tetra Pack	Q	Bird
F	Circle	R	rotate left
G	Flat	S	rotate right
H	Medium	T	House
I	Football	U	Ball
J	Square	V	Bottle
K	Normal	W	Triangle Up
L	Simple	X	DVD

Another challenge is the varying number of internal states generated dynamically during the learning process. Adding and removing objects from the display may lead to flickering and undesired clutter. Therefore, prototype representations of average or median states are desirable for conveying global pictures of internal states. Vector quantization methods like k-means or self-organizing maps can be used for clustering vectorial data [18], and generalizations for relational data have been recently developed [17].

Thus, some demands for the visualization of learning process data are that it should be

- usually unsupervised;
- stable (continuous) between successive plots;
- turning similarity information into neighborhood proximities;
- able to handle a varying number of objects (clustering option);
- time & memory efficient.

In the following example we make use of a method for the reconstruction of neighborhoods in Euclidean space for the visualization of rule sets.

B. Concrete Example of a Visualization of Learned Rules

As an illustrative example we visualized of rules of the Sphinx learning system after episode 1 and after episode 10 of learning. A rule is given by an 8 bit character code containing the 7 variables explained in table I and the action of the system in the last bit, which can be the recognition of 9 possible objects or a rotation (see section II-B for details). The character definitions can be found in table II. As explained before the rules are accompanied with a numeric plausibility value, where lower values denote that the system trust the rules more than bigger ones.

First we created a pairwise dissimilarity score between all rules taking into account the first 7 variables. As we can see from table I some variables, like 1 *FrontViewShape* and 4 *SideViewShape*, should be considered as equally far from each other. We see now reason to consider *Circle* more similar to *Square* than *TriangleUp*. In the special

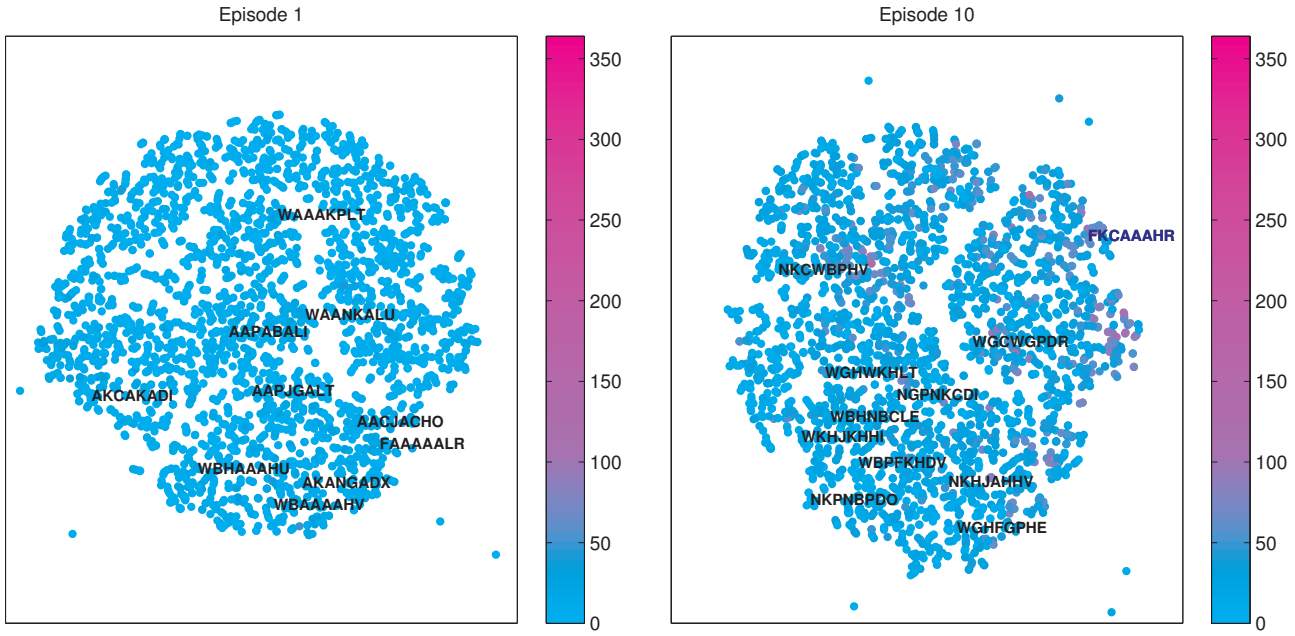

Figure 2. Visualization of the neighborhood relations of rules after the first training episode of the self-learning system Sphinx and after 10 episodes. Neighboring points denote similar rules according to a scoring system explained in the text. The color denotes the plausibility, cyan denoting highest plausibility (OCF=0), magenta denoting least plausibility (OCF=360). At the beginning every rules is trusted almost equally well, because the system did not yet develop proper differentiation into pragmatic values. After 10 episodes some implausible rules are identified, which are highlighted in the visualization by a magenta color.

case of *Unknown* in one of these two variables we even consider no contribution at all to the dissimilarity value for the comparison of the rules. For the other variables the content can be ordered, such that we simply exchanged them by numerical values ranging from 0 to 3 and taking the squared Euclidean distance. The contribution of the non-ordered variables is just added to this distance. This way we end up with pairwise dissimilarity scores of the rules, which we can use in state-of-the-art nonlinear visualization techniques, such as t-SNE.

Figure 2 shows the pairwise dissimilarities of the rules embedded in the 2-dimensional Euclidean space after the first training episode (=150 learning iterations) of the learning system and after 10 episodes (=1.500 iterations). We used t-SNE on the dissimilarity scores using the publicly available code and with a neighborhood contiguity parameter (perplexity) of 10. Since neighborhood relationships of embedded rules in the point cloud do not change under rotation and scaling operations, the axes express extent of space only. In the beginning we can find a lot of rules with 'A'-*Unknown* feature content and nearly all rules are equally trusted, because the system did not develop a differentiation into levels of usefulness at that early stage. We plotted some example rules on the position of their corresponding dot in the map. After 10 episodes, rules with a lot of 'A's are denoted by a higher OCF value like the rule 'FKCAAAHR' shown in the upper right corner.

One example for a rule that is ranked implausible, i.e., is colored magenta after 10 episodes, is 'NKCWBPHV'. This rule means 'IF Front View Shape = Triangle Down AND Front View Size = Normal AND Front View Deviation = Little AND Right View Shape = Triangle Up AND Right View Size = Tall AND Right View Deviation = Much AND Texture = Medium THEN Recognize Bottle'.

On the other hand, one example for a rule that is still ranked plausible after 10 episodes, is 'NKPNBPDO'. This rule means 'IF Front View Shape = Triangle Down AND Front View Size = Normal AND Front View Deviation = Much AND Right View Shape = Triangle Down AND Right View Size = Tall AND Right View Deviation = Much AND Texture = Complex THEN Recognize Tree'.

Besides the different colors another difference between teh diagrams can be recognitzed. There are also more trenches between groups of rules visible in the right panel, expressing further differentiation of the trained system.

Summarizing, in this example we derived dissimilarity codes in an heuristic way using our prior knowledge of the nature of the variables, but custom dissimilarity measures can be employed as desired.

IV. Conclusion

We discussed the problem of visualization of processes in self-learning systems. This topic becomes relevant especially when self-learning systems are applied in safety critical environments because the visualization of internal system

statuses and processes can provide transparency and thus increase usability and trust. The most promising approaches in the field of self-learning systems combine learning methods on several hierarchies, including symbolic representations of the world. As such representations are understandable by humans they are especially suited to be applied in domains, where usability and safety is critical.

By means of an example of an existing self-learning system we developed some ideas how to visualize dynamic learning processes and gave a concrete example of a visualization of symbolic knowledge in the form of learned rules.

In particular, it seems to be reasonable to visualize symbolic rules using state-of-the-art nonlinear techniques. This way the user can get an impression of the system state via visual inspection. Therefore suitable metrics are necessary, especially if the code presentation accounts for defined semantic meanings.

We visualized learned rules of a self-learning system in a way that enables an operator to monitor its internal learning state. By a comparision of the neighborhood relations of learned rules at different instants of time an operator can recognize an increasing development of clusters of rules (indicated by an increasing width of trenches between rule clusters). Furthermore, she is enabled to follow the process of an increasing rejection of implausible rules (indicated by an increasing number of rules colored magenta). Both artifacts can be used as an indicator of a proper functioning of the system, thus contributing to more saftey and security.

We have demonstrated our ideas by means of a special learning system where learned rules can be compared by means of a customized distance metric. Of course, there is a huge variety of learning approaches, and visualization techniques will depend on the special algorithms and data structures at hand. Future work has to deal with the broader question of a possible matching of different visualization techniques to different classes of learning systems.

ACKNOWLEDGMENT

The authors like to thank Schloss Dagstuhl International Conference and Research Center for Computer Science in Germany and the organizers of the seminar 12081 "Information Visualization, Visual Data Mining and Machine Learning" for providing the opportunity to develop the ideas discussed in this paper. M. Strickert is supported by the LOEWE Center for Synthetic Microbiology (SYNMIKRO).

REFERENCES

[1] A. Dix, J. Finlay, G. Abowd, and R. Beale, *Human Computer Interaction*, 3rd ed. Prentice Hall, 2004.

[2] J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human-Computer Interaction*. John Wiley, 2002.

[3] D. Benyon, *Designing Interactive Systems: A Comprehensive Guide to HCI and Interaction Design*. Addison Wesley, 2010.

[4] W. Spohn, *A Survey of Ranking Theory*. In *Degrees of Belief*. Springer, 2009.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

[6] G. Peters, *Six Necessary Qualities of Self-Learning Systems - A Short Brainstorming*. Int. Conf. on Neural Computation Theory & Applications (NCTA 2011). Paris, France, 2011.

[7] R. Sun, X. Zhang, P. Slusarz, and R. Mathews, *The Interaction of Implicit Learning, Explicit Hypothesis Testing Learning and Implicit-to-Explicit Knowledge Extraction*. Neural Networks, 20(1):34-37, 2007.

[8] T. Leopold, G. Kern-Isberner, and G. Peters, *Combining Reinforcement Learning and Belief Revision - A Learning System for Active Vision*. Brit. Mach. Vis. Conf. (BMVC 2008), 473–482, 2008.

[9] T. Leopold, G. Kern-Isberner, and G. Peters, *Belief Revision with Reinforcement Learning for Interactive Object Recognition*. Europ. Conf. Artif. Intellig. (ECAI 2008), 65–69, 2008.

[10] G.E. Hinton and S.T. Roweis, *Stochastic Neighbor Embedding*. In Advances in Neural Information Processing Systems, volume 15, MIT Press, 833–840, 2002.

[11] L. J. P. van der Maaten and G. E. Hinton. *Visualizing High-Dimensional Data Using t-SNE*. Journal of Machine Learning Research 9: 2579–2605, 2008.

[12] D. Cook and D.F. Swayne, *Interactive and Dynamic Graphics for Data Analysis with R and GGobi*. Springer, 2007.

[13] A. Hutchinson, *Metrics on terms and clauses*. Machine Learning: ECML-97, LNCS 1224, Springer, 138–145, 1997.

[14] I.I. Zhegalkin, *On the Technique of Calculating Propositions in Symbolic Logic*. Matematicheskii Sbornik 43: 9–28, 1927.

[15] M. Karnaugh, *The Map Method for Synthesis of Combinational Logic Circuits* Transactions of the American Institute of Electrical Engineers part I 72(9): 593–599, 1953

[16] B. Hammer and A. Hasenfuss, *Clustering Very Large Dissimilarity Data Sets*. Artificial Neural Networks in Pattern Recognition, LNCS 5998, 259–273, 2010.

[17] A. Gisbrecht, B. Mokbel, and B Hammer, *Relational generative topographic mapping*. Neurocomputing 74(9): 1359–1371, 2011.

[18] T. Kohonen, *Self-Organizing Maps*. Springer, 3rd ed., 2001

[19] H. Broer and F. Takens, *Dynamical Systems and Chaos*. Applied Mathematical Sciences 172, Springer, 2011.

[20] N. Marwan, *Encounters with Neighbors*. PhD Thesis, University of Potsdam, 2003.

[21] T. Villmann, S. Haase, F.-M. Schleif, B. Hammer, and M. Biehl, *The Mathematics of Divergence Based Online Learning in Vector Quantization*. Artificial Neural Networks in Pattern Recognition, LNCS 5998, 108–119, 2010.