MicroPsi: Contributions to a Broad Architecture of Cognition

Joscha Bach[1], Colin Bauer[2], and Ronnie Vuine[3]

[1] University of Osnabrück, Institute for Cognitive Science, Osnabrück, Germany
jbach@uos.de
[2] Technical University of Berlin, Department for Computer Science, Berlin, Germany
kolynos@gmail.com
[3] Humboldt-University of Berlin, Institute for Computer Science, Berlin, Germany
vuine@informatik.hu-berlin.de

Abstract. The Psi theory of human action regulation is a candidate for a cognitive architecture that tackles the problem of the interrelation of motivation and emotion with cognitive processes. We have transferred this theory into a cognitive modeling framework, implemented as an AI architecture, called MicroPsi. Here, we describe the main assumptions of the Psi theory and summarize a neural prototyping algorithm that matches perceptual input to hierarchical declarative representations.

1 Introduction

Computational models of cognitive functioning usually emphasize problem solving, not emotion and motivation [1]. Thus they tend to fall short in modeling the interrelations between problem solving and memory functions and the context provided by emotional modulation and motivational priming, and they do not describe the cognitive system as an autonomous agent acting on its environment, but as a module within such an agent – and it is not clear if such a separation is warranted [2]. This has given rise to the suggestion of broader architectures of cognition which tightly integrate motivation and emotion with perceptual and reasoning processes. A very promising approach at such a broad architecture comprises the Psi theory of Dietrich Dörner [3, 4, 5]. Since this theory has not been extensively published in English, we will give a short summary on the following pages.

Psi is routed in a theory of problem solving [6] that makes use of neuro-symbolic models. Representations in the context of Psi are perceptual symbol systems [7], i.e. declarative and procedural descriptions are completely grounded in interaction contexts, which is achieved by using hierarchical spreading activation networks, with the lowest level of the hierarchy addressing sensor and motor systems. Depending on weights and link types, nodes within these hierarchies might carry their semantics individually (localist, symbolic) or as part of a configuration of jointly activated nodes (distributed, sub-symbolic). Cognitive processes are facilitated by control structures that are implemented as procedural representations within the same formalism.

Basic emotions in the Psi theory are understood as modulations of cognition, i.e. they emerge from configurations of various parameters (such as arousal, pleasure/distress signals and resolution level) that determine *how* cognition is carried out, and motivation is based on a finite set of competing drives, both physiological and cognitive.

Its focus on emotion, motivation and interaction make Psi very different from contemporary cognitive architectures like ACT-R [8] and Soar [9]. However, the design and integration of the motivational system bears a striking resemblance to the more recent, but independent CLARION architecture [10]. Like CLARION, the Psi theory proposes procedural reinforcement-learning based on pleasure/distress signals originating in the satisfaction and frustration of drives. The suggested cognitive drives however differ somewhat (they are less parsimonious in Psi). On the other hand, representations in Psi differ, because they are not separated into distinct symbolic and sub-symbolic formalisms – they use a single mode of representation for both.

Implementations of the Psi theory by Dörner's group have facilitated the successful evaluation of the emotional model against human emotions in a complex problem solving tasks [11, 12]; Psi is somewhat unique within cognitive architectures in offering such a validated model [13]. However, these implementations are unsuitable for independent experimentation, which we see as a primary requisite to turn the theory into a cognitive architecture, and they do not scale towards the integration of the representational mechanisms proposed by the theory. This demand is addressed by the *MicroPsi model*, by specifying an agent architecture and a scalable implementation framework, albeit not within the context of psychology, but computer science.

2 Assumptions of the Psi Theory

The Psi theory describes cognition in the terms of a homeostatic system: as a structure consisting of relationships and dependencies that is designed to maintain a homeostatic balance in the face of a dynamic environment. It consists of a set of assumptions that could be summarized as follows:

1. Explicit symbolic representations: The Psi theory suggests *hierarchical networks of nodes* as a *universal* mode of representation for declarative, procedural and tacit knowledge: representations in models of the Psi theory (Psi *agents*) are neuro-symbolic. These nodes may encode *localist and distributed* representations. The activity of the system is modeled using *modulated* and *directional* spreading of activation within these networks. Plans, episodes, situations and objects are described with a semantic network formalism that relies on a fixed number of pre-defined link types, which especially encode *causal/sequential ordering*, and *partonomic hierarchies* (the theory specifies four basic link-types to denote predecessor und successor, has-part and is-part relations).

There are special nodes (representing neural circuits) that control the spread of activation and the forming of temporary or permanent *associations* and *disassociations*.

2. Memory: The Psi theory posits a world model (*situation image*). The current situation image is extrapolated into a branching *expectation horizon* (consisting of anticipated developments and active plans). Working memory also contains an *inner screen*, a hypothetical world model that is used for comparisons during recognition, and for planning.

The situation image is gradually transferred into an episodic memory (*protocol*). By selective *decay* and *re-inforcement*, portions of this long-term memory provide *automated behavioral routines*, and *elements for plans* (procedural memory). The fundamental atomic element of plans and behavior sequences is a *triplet* of a (partial, hierarchical) situation description, forming a condition, an operator (a hierarchical action description) and an expected outcome of the operation as another situation description.

Object descriptions (mainly declarative) are also part of long-term memory and the product of perceptual processes and affordances. Situations and operators in long-term memory may be associated with *motivational relevance*, which is instrumental in retrieval and reinforcement. Operations on memory content are subject to emotional *modulation*.

3. Perception: Perception is based on conceptual hypotheses, which guide the recognition of objects, situations and episodes. *Hypothesis based perception ('HyPercept')* is understood as a *bottom-up* (data-driven and context-dependent) cueing of hypotheses that is interleaved with a *bottom-down* verification.

The acquisition of schematic hierarchical descriptions and their gradual adaptation and revision can be described as *assimilation* and *accommodation* [14]. Hypothesis based perception is a *universal principle* that applies on visual perception, auditory perception, discourse interpretation and even memory interpretation. Perception is subject to emotional *modulation*.

4. Urges/drives: The activity of the system is directed on the satisfaction of a *finite set* of primary, *pre-defined urges* (drives). All goals of the system are situations that are associated with the satisfaction of an urge, or situations that are instrumental in achieving such a situation (this also includes abstract problem solving, aesthetics, the maintenance of social relationships and altruistic behavior). These urges reflect *demands* of the system: a mismatch between a target value of a demand and the current value results in an *urge signal*, proportional to the deviation, which might give rise to a motive. There are three categories of urges:

- *physiological urges* (such as food, water, maintenance of physical integrity), which are relieved by the *consumption* of matching resources and increased by the metabolic processes (food, water) of the system, or inflicted damage (integrity).
- *social urges (affiliation)*. The demand for affiliation is an individual variable and adjusted through early experiences. The urge for affiliation needs to be satisfied in regular intervals by *external legitimity signals* (provided by other agents as a signal of acceptance and/or gratification) or *internal legitimity signals* (created by the fulfillment of social norms). It is increased by social frustration (*anti-legitimity signals*) or *supplicative signals* (demands of other agents for help, which create both a suffering by frustration of the affiliation urge, and a promise of gratification).
- *cognitive urges* (*reduction of uncertainty*, and *competence*). Uncertainty reduction is maintained through exploration and frustrated by mismatches with expectations and/or failures to create anticipations. Competence consists of *task specific competence* (and can be acquired through exploration of a task domain) and *general competence* (which measures the ability to fulfill the demands in general). The urge for competence is frustrated by actual and anticipated failures to reach a goal. The cognitive urges are subject to individual variability and need regular satisfaction.

The model strives for *minimal parsimony* in the specification of urges. For instances, there is no need to specify a specific urge for social power, because this may be reflected by the competence in reaching affiliative goals, while an urge for belongingness partially corresponds to uncertainty reduction in the social domain. The model should only expand the set of basic urges if it can be shown that the existing set is unable to produce the desired variability in behavioral goals. Note that none of the aforementioned urges may be omitted without affecting the behavior.

5. Pleasure and distress: A *change* of a demand of the system is reflected in a *pleasure* or *distress signal*. The strength of this signal is *proportional* to the amount of change in the demand measured over a short interval of time. Pleasure and distress signals are *reinforcement* values for the learning of behavioral procedures and episodic sequences and define *appetitive* and *aversive* goals.

6. Modulation: Cognitive processing in subject to *global modulatory parameters*, which adjust the cognitive resources of the system to the environmental and internal situation. Modulators control behavioral tendencies (action readiness via *general activation* or *arousal*), stability of active behaviors/chosen goals (*selection threshold*), the rate of orientation behavior (*sampling rate* or *securing threshold*) and the width and depth of activation spreading in perceptual processing, memory retrieval and planning (*activation* and *resolution level*). The effect and the range of modulator values are subject to *individual variance*.

7. Emotion: Emotion is not an independent sub-system, a module or a parameter set, but an intrinsic *aspect of cognition*. Emotion is an emergent property of the modulation of perception, behavior and cognitive processing, and it can therefore not be understood outside the context of cognition, that is, to model emotion, we need a cognitive system that can be modulated to adapt its use of processing resources and behavior tendencies. (According to Dörner, this is necessary *and* sufficient.) In the Psi theory, emotions are understood as a configurational setting of the *cognitive modulators* along with the *pleasure/distress dimension* and the assessment of the *cognitive urges*. This perspective addresses *primary emotions*, such as joy, anger, fear, surprise, relief, but not *attitudes* like envy or jealousy, or emotional responses that are the result of modulations which correspond to specific demands of the environment, such as disgust.

The *phenomenological qualities* of emotion are due to the effect of modulatory settings on perception on cognitive functioning (i.e. the perception yields different representations of memory, self and environment depending on the modulation), and to the experience of accompanying physical sensations that result from the effects of the particular modulator settings on the physiology of the system (for instance, by changing the muscular tension, the digestive functions, blood pressure and so on). The *experience of emotion* as such (i.e. as *having an emotion*) requires reflective capabilities. Undergoing a modulation is a necessary, but not a sufficient condition of experiencing it as an emotion.

8. Motivation: Motives are *combinations of urges and a goal* that is represented by a situation that affords the satisfaction of this urge. (Motives are terminologically and conceptually different from urges and emotions. *Hunger*, for instance, is an urge signal, an association of hunger with an opportunity to eat is a motive, and *apprehension* of an expected feast may be an emergent emotion.)

There may be several motivations active at a time, but *only one* is chosen to determine the choice of behaviors of the agent. The choice of the dominant motive depends on the anticipated probability to satisfy the associated urge and the strength of the urge signal. (This means also that the agent may opportunistically satisfy another urge, if it presented with that option.) The *stability of the dominant motive* against other active motivations is regulated using the selection threshold parameter, which depends on the *urgency* of the demand and individual variance.

9. Learning: Perceptual learning comprises the *assimilation/ accommodation* of new/existing schemas by hypothesis based perception. Procedural learning depends on *reinforcing* the associations of actions and preconditions (situations that afford these actions) with *appetitive or aversive* goals, which is triggered by pleasure and distress signals. *Abstractions* may be learned by evaluating and reorganizing episodic and declarative descriptions to generalize and fill in missing interpretations (this facilitates the organization of knowledge according to conceptual frames and scripts). Behavior sequences and object/situation representations are *strengthened by use*. Tacit knowledge (especially sensory-motor capabilities) may be acquired by *neural learning*. Unused associations *decay*, if their strength is below a certain threshold: highly relevant knowledge may not be forgotten, while spurious associations tend to disappear.

10. Problem solving: Problem solving is directed on *finding a path* between a given situation and a goal situation, on completing or *reorganizing mental representations* (for instance, the identification of relationships between situations or of missing features in a situational frame) or it serves an *exploratory* goal. It is organized in stages: If no *immediate response* to a problem is found, the system first attempts to resort to a behavioral routine *(automatism)*, and if this is not successful, it attempts to construct a *plan*. If planning fails, the system resorts to *exploration* (or switches to another motive).

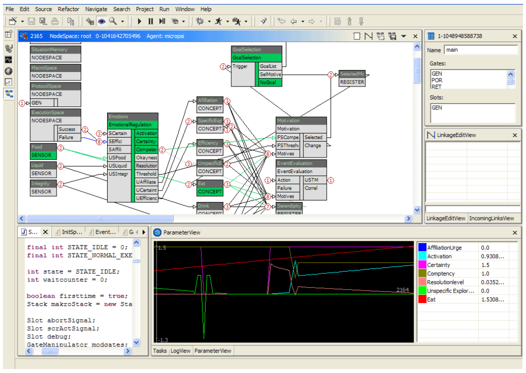
Problem solving is *context dependent* (contextual priming is served by associative pre-activation of mental content) and subject to *modulation*. The strategies that encompass problem solving are *parsimonious*. They can be reflected upon and reorganized according to learning and experience. According to the Psi theory, many advanced problem solving strategies can not be adequately modeled without assuming *linguistic capabilities*.

11. Language: Language has to be explained as syntactically organized symbols that designate conceptual representations and a model of language thus starts with a model of mental representation. Language extends cognition by affording the categorical organization of concepts and by aiding in meta-cognition. (Cognition is not an extension of language.) The understanding of discourse may be modeled along the principles of hypothesis based perception and assimilation/ accommodation of schematic representations.

The Psi theory is largely qualitative, not quantitative, which makes it slightly unusual in contemporary research in cognitive science, but very useful as a frame of thought when addressing cognitive phenomena. After all, most pressing with respect to understanding human intelligent behavior start with "how" and "what" instead of "how much". Yet, to evaluate its proposals, it needs to be implemented as a model, which itself has to include commitments to concrete algorithms, representational formalisms and parameter settings. Dörner's own implementations as partial computer models do not favor such an evaluation, because they do not specify most of these commitments, nor do they separate between theory, architecture and model.

3 MicroPsi

MicroPsi translates the Psi theory into a cognitive architecture that eventually allows the comparison with other approaches. It comprises a development and simulation framework, written in Java, that allows implementing multi-agent systems according to the principles of the Psi theory, and it specifies an agent architecture that is implemented within the framework. MicroPsi is also used as a robot control architecture.



```
final int STATE_IDLE = 0;
final int STATE_NORMAL_EXE

int state = STATE_IDLE;
int waitcounter = 0;

boolean firsttime = true;
Stack makroStack = new Sta

Slot abortSignal;
Slot scrActSignal;
Slot debug;
GateManipulator modgates;
```

🟦	AffiliationUrge	0.0
🟦	Activation	0.9308...
🟪	Certainty	1.5
🟫	Comptency	1.0
🟥	Resolutionlevel	0.0352...
🟩	Unspecific Explor...	0.0
🟥	Eat	1.5308...

Fig. 1. MicroPsi toolkit

The framework offers an editor for hierarchical spreading activation networks, which is the principal tool for the design of Psi agents, a graphical simulation world that facilitates multi-agent interaction and several customizable environmental designs offering different tasks and tools for the visualization and evaluation of experiments.

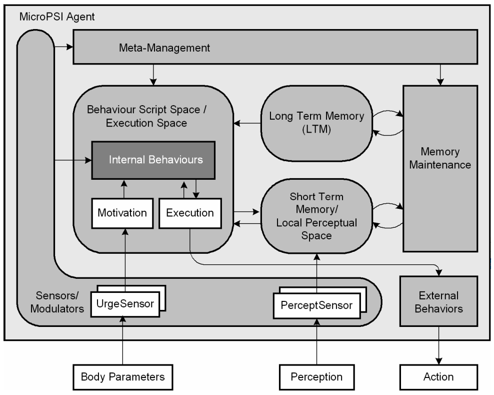


Fig. 2. MicroPsi agent architecture

MicroPsi agents are partial implementations of the Psi theory. They provide a motivational system in which the cognitive processes are embedded. The agent implementations so far address simple hypothesis based perception, means-end analysis, behavior execution, emotional modulation, reinforcement-learning based on satisfaction and frustration of drives, simple neural learning of low-level stimuli and environmental mapping.

3.1 Current Experiments: Neural Prototyping

Our current work deals with the extension of individual components of MicroPsi agents, such as the integration of neural learning of perceptual patterns from camera images with high-level concepts and the acquisition of hierarchical object descriptions. Here, we describe an approach to structure classification using *neural prototyping*.

When recognizing objects, planning and retrieving object hypotheses from long-term memory, MicroPsi agents need to classify hierarchical representations, which is a computationally expensive task, when structure matching is involved. Thus we need a strategy to minimize the structure comparison operations, and we address this need with an algorithm using class prototypes. These prototypes are represented as neural networks, where topology and weights are changed through learning. A first classification is performed by sending activation through these networks and to select only the most active prototypes. These few remaining structures can then be matched using a subgraph matching technique to identify the most similar prototype and with it the most suitable class. The advantages of this approach are that the pruning takes only as many steps as the depth of the largest prototype, and that the expensive structure matching is only used for a very small subset of items in memory.

The first step of the algorithm is to convert the class prototypes (that can be predetermined or acquired through learning) into neural networks. We used the approach developed by Towell and Shavlik [15], who describe the conversion of hierarchical structures into labelled neural networks (KBANN: knowledge-based artificial neural networks) and the properties of those networks. The main idea is to convert the nodes and links into neurons and connections in a network, setting the weights and biases in such a way as to preserve the logic of the structure.

After all prototypes have been converted, the neural classification can be performed on a sample structure. This means that all sensors (the leaf nodes in the hierarchical neural networks) that have the same label as one of the sensors in the example provided get activated and spread their activation through the network. A smooth activation function has to be chosen, otherwise only the examples that are isomorphic to a prototype will be able to activate its root node and thus be classified.

Next, structure matching is performed between the example and the prototypes with the highest activation of their root nodes. This step is necessary, because the neural activation phase is only able to give a rough estimate of similarity. It only takes a very small part of the structures' topologies into account. Since the topology is an important factor for classification, the example is then assigned to the class with the most similar structure. There are many different algorithms for structure matching ([16, 17], an overview in [18]). We decided to use the method described by Schädler and Wysotzki [18, 19], based on a Hopfield-network, where each node corresponds to two nodes whose similarity exceeds a certain threshold, because it can be integrated nicely into the general MicroPsi framework of representation. In our context, the similarity is measured by comparing the sensors of the subtrees rooted at the respective nodes, and the structures of these subtrees. Connections exist between nodes in the Hopfield network if they exist between the nodes in the original structures. In addition, connections with negative weights are added to the network between nodes that correspond to mapping a node in one graph to two nodes in the other. After the construction has been completed, the network is run until it reaches a stable state. The active nodes in this stable state represent a maximum common subgraph of the two original structures.

After the example has been compared to each of the prototypes, there are three possibilities to be considered:

- The example's similarity to one of the prototypes exceeds a predefined threshold. In that case, the example is assigned the class of the prototype and the classification procedure is finished.
- Maximum similarity is high, but not high enough. Here the intuition is that the class of the closest prototype should be correct, but that the prototype is not good enough to capture the individual properties of the example. In this case, the maximum common subgraph computed during structure matching can be used to add the example-specific parts to the prototype. This is achieved by converting the example into a KBANN network, attaching the parts not included in the maximum common subgraph and changing the weights of the network (by backpropagation learning).
- No prototype is sufficiently similar. In that case, no statement can be made about the class of the example, or, in the case of training, a new prototype is added to the example's class.

The algorithm has very interesting features: it does not need a complete set of prototypes in order to work efficiently, because it adds new prototypes or changes existing ones during training. This is required for its application in the MicroPsi architecture, because MicroPsi agents start with very limited or no knowledge about the world and have to build their knowledge base over time with only limited help from the outside. The system works under the supervised as well as unsupervised paradigms. In supervised learning, a training set of labeled examples can be used to generate a good set of prototypes. When dealing with unsupervised learning, the prototypes correspond to a set of clusters that can be built continuously, without having to make assumptions about size or locations of clusters beforehand, as is necessary with other algorithms. The system is very efficient, because the computationally expensive process of pruning the search space is done by neural networks of limited depth that can be run in parallel. Neural learning techniques and the topological modification of these classifying networks enable the system to generalize faster and better than other generalization algorithms that are based on finding the maximum common subgraphs between elements of one class that serve as generalized structures. The fact that the nodes of the prototype networks have semantic meaning can be exploited during training, to achieve faster convergence and less training time.

We ran two different experiments to test the performance of our approach and to compare its performance to other structure classification algorithms. The task of both experiments was to classify visually given objects. To represent them, they were represented as *shock graphs* [20], which are derived by transforming the skeleton of a two-dimensional shape into a hierarchical graph (Fig. 3), where nodes correspond to the vertices and end points of the skeleton and edges to their interrelations. (Using shock-graph was a somewhat arbitrary decision, which aimed at providing a basic abstraction over visual input.) The algorithm first computes the skeleton of the given shape and identifies its shock points. Given the shock points, the so called shock graph grammar can be used to create a hierarchical structure using the shocks as nodes and connecting them according to the grammar rules.

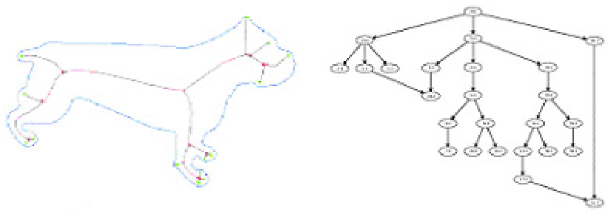


Fig. 3. Skeleton and shock points, sample shock graph taken from [21]

3.1.1 Supervised Learning

We trained the algorithms to classify 8 different shapes from a data-base of visual objects [21]. From each category we chose 5 shapes that were used for training, and the accuracy of each method was measured using 50 objects per class.

The five prototypes of each class, presented one at a time, were used to build the prototype networks. A prototype only became part of the prototype networks, if none of the already present prototypes of the same class got activated during spreading activation. At the end of the training phase, 30 of the 40 prototypes remained. In 10 cases there were other prototypes that already classified the given prototype correctly. Then the 400 examples were presented to the algorithm. For each example, the five prototypes with the highest activation were selected for structural comparison, and the example assigned to the one with the highest similarity.

We have compared our algorithm to several other approaches: to *eigenvalue-based indexing* [22], which represents the structure of the graph as a vector capturing the branching structure and node distribution; similarity is determined by computing the euclidean distance between the eigen-vectors of two graphs; to attributed graph indexing, where the eigenvectors are not derived from the standard adjacency matrix, but from the *attributed graph* – here, the entries in the diagonal are the labels of the nodes, and the other entries the labels of the links connecting the respective nodes; *subgraph prototyping* [19], where – using our own subgraph matching procedure – we chose the prototype with the largest common subgraph. For neural prototyping and the eigenvector-based methods, we use the respective algorithms to retrieve the five most similar graphs for each of the 400 examples and then perform a complete structural comparison. As a base-line, we include linear search, which performs a complete graph-matching with every example (and is prohibitively slow).

Exact runtimes could not be compared because some algorithms used different implementation techniques. Nevertheless, from the experiments we conducted it could be observed that the runtimes of the neural prototyping and eigenvalue-based approaches were comparable, whereas the other algorithms were significantly slower.

The histograms in Figure 4 depict the results for the five algorithms, where the x-axis represents the eight categories and the y-axis shows the percentage of correctly classified examples. The bar graphs represent the percentage of the examples in each class that were correctly classified by the respective algorithm. For example, the neural prototyping algorithm was able to classify 44 of the 50 examples of category 1 correctly, leading to an accuracy of 0.88, as shown in the figure. Since all algorithms were presented with the same test set, their performance can be compared directly by comparing their accuracies across classes.

As expected, linear search gives the best results, but is followed by our algorithm, and then the eigenvector-based approaches. (The results from linear search also show that the examples of the different classes had distinct enough shock graph representations for the structure comparison algorithm to find the correct prototype in most of the cases.) With respect to runtime, neural prototyping is on a par with the eigenvector-based algorithms, because all of them use a pruning strategy to minimize the amount of structural comparisons to be performed. Our algorithm and the subgraph prototyping approach were the only ones that showed generalization effects. In our case, generalization happened during the training phase, where from the total number of 40 prototypes, only 30 generalized prototypes remained and were used for classification.

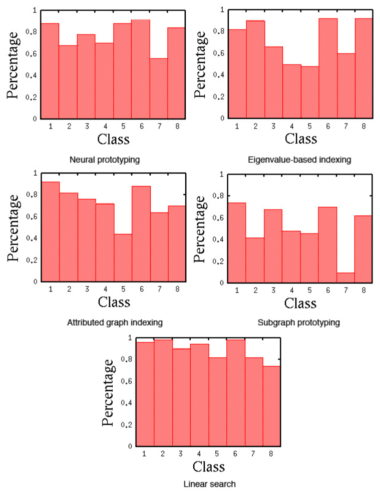


Fig. 4. Results for the different approaches

3.1.2 Unsupervised Learning

25 of the objects of each class that had already been used in the previous experiment were presented to our algorithm one by one without providing class information. The objects in this training pool were used to build a set of prototypes (or cluster points)

for each class. Contrary to the resulting set of prototypes in the previous experiment, the resulting generalized graphs in this case could contain subparts from different, but similar, classes. From a total of 200 prototypes that could be generated, the algorithm constructed 60 partially overlapping prototypes. The other 25 examples from the test pool in the supervised learning setup were used to measure the accuracy of these prototypes. To assess the difficulty of the task, the eigenvalue-based indexing algorithm (algorithm two in the previous experiment) was run on the data. Each graph from the test set was compared to those in the training pool, and the top five matches identified. As in the previous experiment, structure matching was used to select the closest structure out of these candidates. (Linear search in this setting was computationally infeasible.)

Under general circumstances, where no a-priory prototypes are known, the prototype-generating feature of our algorithm should prove advantageous over the eigenvalue-based indexing, where each new object must be compared to all available structures in memory. In our experiment, there were 60 prototypes left at the end of training, as opposed to the originally 200 shapes that were used in the eigenvalue-based approach. In addition, the speed of our algorithm increased over time during training, because as the size of the prototype networks increased, fewer examples needed to modify the topology.

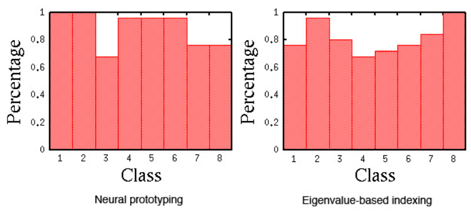


Fig. 5. Results from the unsupervised learning environment

Both algorithms perform well, with some advantage for our algorithm. The inclusion of neural prototyping is a useful extension of MicroPsi, because it enables the agents to explore their world and generate categories without external help. The algorithm generalizes, since only about one fourth of the potential number of prototypes were used to generate prototypes, while maintaining its performance on the test set.

References

1. Detje, F.: Handeln erklären. DUV, Wiesbaden(1999)
2. Clark, A., Grush, R.: Towards a Cognitive Robotics. Adaptive Behavior 7(1) (1999) 5-16
3. Dörner, D.: Eine Systemtheorie der Motivation. Memorandum Lst Psychologie II Universität Bamberg, 2,9 (1994)
4. Dörner, D.: Bauplan für eine Seele. Rowohlt, Reinbeck (1999)
5. Dörner, D., Bartl, C., Detje, F., Gerdes, J., Halcour D., Schaub H., Starker, U.: Die Mechanik des Seelenwagens. Eine neuronale Theorie der Handlungsregulation. Verlag Hans Huber, Bern Göttingen Toronto Seattle (2002)

6. Dörner, D.: Die kognitive Organisation beim Problemlösen. Versuche zu einer kybernetischen Theorie der elementaren Informations-verarbeitungsprozesse beim Denken. Kohlhammer, Bern (1974)
7. Barsalou, L. W.: Perceptual Symbol Systems. In: Behavioral and Brain Sciences, 22,4. (1999) 577-660
8. Anderson, J. R., Lebiere, C.: The atomic components of thought. Erlbaum, Mahwah, NJ (1998)
9. Laird, J. E., Newell, A., Rosenbloom, P. J.: Soar: An architecture for general intelligence. Artificial Intelligence, 33(1) (1987) 1-64
10. Sun, R.: Cognition and Multi-Agent Interaction. Cambridge University Press (2005) 79-103
11. Detje, F.: Comparison of the PSI-theory with human behaviour in a complex task. In: Taatgen, N., Aasman, J. (eds.): Proceedings of the Third International Conference on Cognitive Modelling. Universal Press, Veenendaal, The Netherlands (2000) 86-93
12. Dörner, D.: The Mathematics of Emotion. Proceedings of ICCM-5, International Conference on Cognitive Modeling. Bamberg, Germany (2003)
13. Ritter, F. E., Shadbolt, N, R., Elliman, D., Young, R. M., Gobet, F., Baxter, G. D.: Techniques for Modeling Human Performance in Synthetic Environments: A Supplementary Review. Human Systems Information Analysis Center, State of the Art Report (2003)
14. Piaget, J.: The construction of reality in the child. Basic Books, New York (1954)
15. Towell, G. G., Shavlik, J. W.: Knowledge-based artificial neural networks. In: Artificial Intelligence, 70(1-2). (1994) 119–165
16. Sebastian, T. B., Klein, P. N., Kimia, B. B.: Recognition of shapes by editing shockgraphs. IEEE International Conference on Computer Vision. (2001) 755–762
17. Pelillo, M., Siddiqi, K., Zucker, S. W.: Matching hierarchical structures using association graphs. In: IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol 21. (1999) 1105–1120
18. Schädler, K., Wysotzki, F.: Comparing structures using a hopfield-style neural network. Applied Intelligence, Vol. 11. (1999) 15–30
19. Schädler, K., Wysotzki, F.: A connectionist approach to structural similarity determination as a basis of clustering, classification and feature detection. PKDD. (1997) 254–264
20. Siddiqi, K., Shokoufandeh, A., Dickinson, S., Zucker, S.: Shock graphs and shape matching. IEEE International Journal on Computer Vision. (1998) 222–229
21. Macrini, D.: Indexing and matching for view-based 3-d object recognition using shock graphs. Master's thesis, University of Toronto (2003)
22. Shokoufandeh, A., Dickinson, S.: A unified framework for indexing and matching hierarchical shape structures. 4th International Workshop on Visual Form. (2001) 28–46