Chapter 9

Adaptive cognitive systems

Timo Honkela, Aapo Hyvärinen, Krista Lagus, Ville Könönen, Kevin I. Hynnä, Juha Winter, Jaakko Väyrynen

9.1 Introduction

Our research on cognitive systems focuses on modeling and applying methods of unsupervised and reinforcement learning. The general aim is to provide a methodological framework for theories of conceptual development, symbol grounding, communication among autonomous agents, and constructive learning. We also work in close collaboration with other groups in our laboratory, e.g., related to multimodal environments and sensory fusion.

9.2 Unsupervised learning for agent communication

Traditional cognitive models and language technologies widely neglect the fact that language users learn the language in a large varity of contexts. This leads into varying interpretation of expressions. When this aspect is not carefully considered also the practical applications suffer from problems related to the basic underlying assumptions. Perhaps the most striking classical example can be given in the area of information retrieval. Namely, it has been found that in spontaneous word choice for objects in five domains, two people favored the same term with less than 20% probability. Moreover, it has been shown that different indexers, well trained in an indexing scheme, might assign index terms for a given document differently. It has also been observed that an indexer might use different terms for the same document at different times. In summary, while developing models of linguistic cognition or some computational tools, we cannot assume that there is a commonly shared model among the language users. On contrary, we have to be able to develop systems that are capable, among other things, to conduct meaning negotiations.

Abstract model of adaptive communicating agents

At an advanced level of multi-agent co-operation, as mentioned above, each agent has its own model of the environment. Thus, each agent has an individual interpretation for the relationship between the messages and the environment. These differences in the agents' models motivate the development of methods which provide the agents with the ability to learn, including learning to interpret messages from other agents.

The agents can perceive their environment, they are part of it, and possibly they can change it. The environment may be a computerized representation, constructed, or natural. The borderlines of these domains may, of course, be vague. A natural environment, in particular, is ever-changing, and consists of various continuous phenomena.

We have considered the possibility of applying a natural, or near-natural language as the communication medium. The general properties of natural languages necessitate some capabilities that autonomous agents will need to have. These basic properties of natural languages and their interpretation include ambiguity, contextuality, open-endedness, vagueness, and subjectivity.

Ambiguity or vagueness, then, can be considered a necessity when the communication medium is used in an open and changing environment in which having a distinct and a priori determined symbol, or combination of symbols, would be difficult, or even impossible. Finally, to ensure successful communication, both the sending and the receiving agent must share a similar enough framework of interpretation, and the message or the situation must contain enough information to activate a proper framework for the receiver.

Adaptive communicating agents based on Self-Organizing Map

We have developed a model of communicating agents based on the self-organizing map algorithm [5]. An agent has the following properties: it can perceive its simulated environment, it can move in its environment, it can perform some simple actions, and it can send and receive messages. The main components of its internal structure include a short-term and a long-term memory, and a decision making mechanism. Although these two memory types are closely interconnected, they have different implementations: episodic memory is dynamic and accurate in nature, whereas semantic memory is adaptive and approximative being based on the self-organizing map. The key idea is to provide the means for each agent to associate continuous-valued parameter spaces to sets of symbols, and furthermore, to "be aware" of the differences in this association and to learn those differences explicitly.

These kinds of abilities are especially required by highly autonomous agents that need to communicate using an open set of symbols or constructs in the agent language. [2,4]

The self-organizing map is especially suitable for the central processing element of autonomous agents because of the following reasons:

- The self-organizing map algorithm modifies its internal presentation, i.e., the codebook vectors, according to the external input which enables the adaptation of the agents.
- The self-organizing map is able to process natural language input to form "semantic maps" [6].
- Symbols and continuous variables may be combined in the input, and are associated by the self-organizing map. Continuous variables may be quantized, and a symbolic interpretation can be given for each section in the possibly very high-dimensional space of perceptual variables [1].
- Because the self-organizing map implements unsupervised learning, processing external input without any prior classifications is possible. The autonomous agent may form an individual model of the environment and of the relation between the expressions of the language and the environment [2]. In general, the basic approach is compatible with the framework of constructive learning theories [3].

References

[1] T. Honkela. Self-Organizing Maps in Symbol Processing. In *Hybrid Neural Systems*, Stefan Wermter, Ron Sun (eds.), Springer, Heidelberg, 2000, pp. 348-362.

[2] T. Honkela, K.I. Hynnä, and T. Knuuttila. Framework for Modeling Partial Conceptual Autonomy of Adaptive and Communicating Agents. *Proceedings of Cognitive Science*, 2003.

[3] T. Honkela, T. Leinonen, K. Lonka, and A. Raike. Self-Organizing Maps and Constructive Learning. *Proceedings of ICEUT'2000, International Conference on Educational Uses of Communication and Information Technologies*, Beijing, China, August 21-25, 2000, pp. 339-343.

[4] T. Honkela, and J. Winter. *Simulating Language Learning in Community of Agents Using Self-Organizing Maps.* Helsinki University of Technology, Publications in Computer and Information Science Report, 2003.

[5] T. Kohonen. *Self-Organizing Maps.* Springer, 2001.

[6] H. Ritter, and T. Kohonen. Self-Organizing Semantic Maps. *Biological Cybernetics*, 61:241-254, 1989.

9.3 Reinforcement learning in multiagent systems

Reinforcement learning methods have attained lots of attention in recent years. Although these methods and procedures were earlier considered to be too ambitious and to lack a firm foundation, they have been established as practical methods for solving, e.g., Markov decision processes (MDPs). However, the requirement for reinforcement learning methods to work is that the problem domain in which these methods are applied obeys the Markovian property. Basically this means that the next state of a process depends only on the current state, not on the history. In many real-world problems this property is not fully satisfied. However, many reinforcement learning methods can still handle these situations relatively well. Especially, in the case of two or more decision makers in the same system the Markovian property does not hold and more advanced methods should be used instead. A powerful tool for handling these highly non-Markovian domains is the concept of Markov game. In this project, we have developed efficient learning methods based on the asymmetric learning concept and tested the developed methods with different problem domains, e.g. with pricing applications.

Markov games

With multiple agents in the environment, the fundamental problem of single-agent MDPs is that the approach treats the other agents as a part of the static environment and thus ignores the fact that the decisions of the other agents may influence the state of the environment.

One possible solution is to use competitive multiagent Markov decision processes, i.e. *Markov games.* In a Markov game, the process changes its state according to the action choices of all agents and can thus be seen as a multicontroller MDP. In Fig. 9.1, there is an example of a Markov game with three states (s_1,s_2,s_3) and two agents. The process changes its state according to probability $P(s_i|s_1, a^1, a^2), i = 2, 3$, where a^1, a^2 are actions selected by the agents 1 and 2.

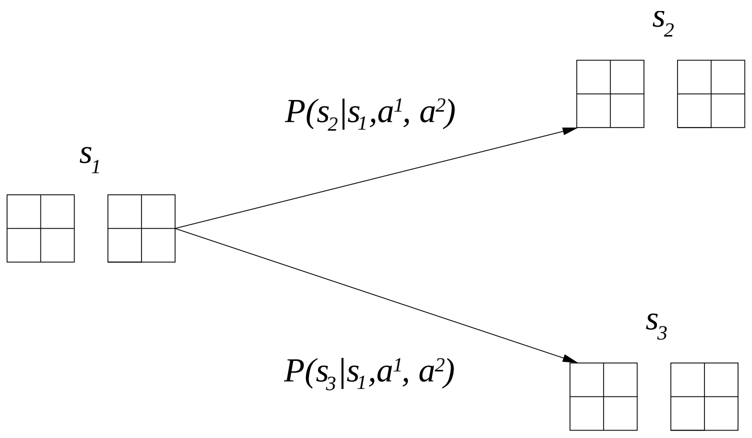

Figure 9.1: An example Markov game with three states.

In single-agent MDPs, it suffices to maximize the utility of the agent in each state. In Markov games, however, there are multiple decision makers and more elaborated solution concepts are needed. Game theory provides a reasonable theoretical background for solving this interaction problem. In the single-agent learning, our goal is to find the utility maximizing rule (policy) that stipulates what action to select in each state. Analogously, in a multiagent setting the goal is to find an equilibrium policy between the learning agents.

Practical learning methods

We have concentrated on the case where the state transition probabilities and utility values are not known to the learning agents. Instead, the agents observe their environment and learn from these observations. In general, we use the update rule in the following form:

$$Q^i_{t+1}(s_t, a^1_t, \dots, a^N_t) = (1 - \alpha_t)Q^i_t(s_t, a^1_t, \dots, a^N_t) + \alpha_t[r^i_{t+1} + \gamma f(s_{t+1})], \qquad (9.1)$$

where $Q^i_t(s_t, a^1_t, a^2_t)$ is the estimated utility value for the agent i at the time instance t when the system is in the state s_t and agents select actions $a^1_t, \dots, a^N_t$. r^i_{t+1} is the immediate reward for the agent i and γ is the discount factor. f is the function used to evaluate values of the games associated with states. If a symmetric evaluation function is used, i.e. Nash or correlated equilibrium function, the update rule is similar for each agent. In the asymmetric case, there is an ordering (some agents make their decisions prior other agents) among learning agents and thus the learning rules are different on different levels of the corresponding agent hierarchy. Further discussion about symmetric learning methods can be found in [1] and [2]. Respectively, fundamental principles and theoretical analysis of the asymmetric model can be found in [3].

Grid world example

In this section we provide a simple example of multiagent reinforcement learning. Let us consider a grid world containing nine cells, two competing agents and two goal cells (Fig. 9.2). Initial positions of the agents are the bottom corners 1 and 2, respectively, and they can move to adjacent cells (4-neighborhood) on each round. An agent gets a large positive payoff when it founds the right goal cell. Additionally, it gets a small negative payoff if it collides with its opponent, i.e. both agents move into the same cell, and the agents are returned back to their original cells.

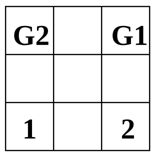

Figure 9.2: The game board used in the grid world example. Agents are initially located in the cells marked with numbers **1** and **2**. Goal cells are marked with symbols **G1** and **G2**.

When this problem is modeled as a Markov game, a state is a pair containing the positions of the agents and the actions are the directions of movement. The problem was solved by using the asymmetric multiagent reinforcement learning method with discount factor $\gamma = 0.99$. In this asymmetric setting, the agent 1 decides his action first (leader) and the agent 2 (follower) reacts optimally to this selection. The learning process converged to the optimal paths (policy functions) shown in Fig. 9.3. Corresponding convergence curves can be found in Fig. 9.4, in which changes in Q-values, i.e. the Euclidean distance between two vectors containing Q-values of the consecutive iterations of the learning algorithm, are plotted against iteration rounds. More detailed empirical evaluations of the asymmetric learning method can be found in [4] and [5].

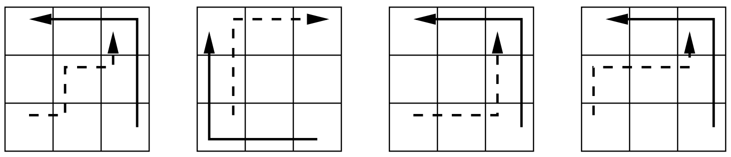

Figure 9.3: Some optimal paths generated by the asymmetric multiagent reinforcement learning model.

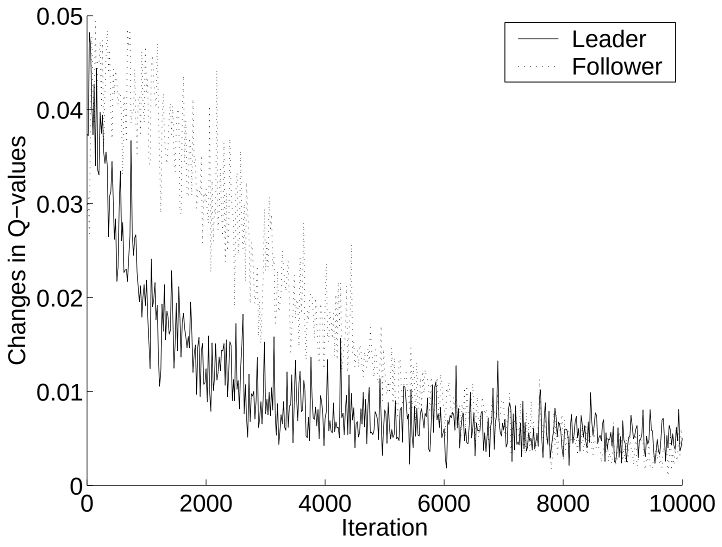

Figure 9.4: The convergence of the asymmetric learning method in the grid world example problem.

References

[1] A. Greenwald and K. Hall. Correlated-Q learning. In *Proceedings of the AAAI-2002 Spring Symposium Workshop on Collaborative Learning Agents*, Stanford, CA, 2002. AAAI Press.

[2] J. Hu and M. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML'98)*, Madison, WI, 1998. Morgan Kaufmann Publishers.

[3] V. Könönen. Asymmetric multiagent reinforcement learning. In *Proceedings of the 2003 WIC International Conference on Intelligent Agent Technology (IAT-2003)*, Halifax, Canada, 2003. IEEE Press.

[4] V. Könönen. Gradient based method for symmetric and asymmetric multiagent reinforcement learning. In *Proceedings of the Fourth International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2003)*, Hong Kong, China, 2003. Springer-Verlag.

[5] V. Könönen. Policy gradient method for multiagent reinforcement learning. In *Proceedings of the 2nd International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2003)*, Singapore, 2003.

9.4 Emergence of linguistic features using Independent Component Analysis

Language technology is very central area in the development of intelligent systems. Traditionally, much of the development work has been manual: encoding linguistic and domain knowledge even for a single system may even take years. We have been studying how aspects of human language learning could be modeled, potentially resulting into (a) more realistic cognitive models than those based on, e.g., rule-based representations, and (b) efficient tools for applications in, for instance, information retrieval, natural language interfaces, machine translation and computer supported collaborative work. In the following, we consider one specific aspect of language learning, i.e, how categories of words can be learned from input data without supervision.

Word Category Learning

A word can belong to several syntactic categories simultaneously. The number of categories is even higher if one takes into account the semantic categories. Such categorization has traditionally been determined by hand: the categories into which a word belongs to are described in a manually collected dictionary.

In order to facilitate learning of word categories, the self-organizing map has earlier been used in the analysis of word context data, e.g., in [4] (artificially generated short sentences), and [1] (Grimm fairy tales). The result of a word context analysis based on the self-organizing map algorithm can be called a word category map. Areas or local regions on a word category map are implicit categories that have emerged during the learning process. Single nodes in the map can be considered as adaptive prototypes. Each prototype is involved in the adaptation process in which the neighbors influence each other and the map is gradually finding a form in which it can best represent the input.

One classical approach for defining concepts is based on the idea that a concept can be characterized by a set of defining attributes. In contrast, the prototype theory of concepts involves that concepts have a prototype structure and there is no delimiting set of necessary and sufficient conditions for determining category membership that can also be fuzzy. Instances of a concept can be ranked in terms of their typicality. Membership in a category is determined by the similarity of an object's attributes to the category's prototype.

The emergent categories on a word category map are implicit. The borderlines for any categories have to be determined separately. It would be beneficial if one could find more explicit categories in an automated analysis. Moreover, each word appears in one location of the map. This means, among other things, that one cannot have a map in which several characteristics or categories of one word would be represented unless the the categories overlap and accordingly the corresponding areas of the map overlap. In some cases, this is the case: it is possible to see the area of modal verbs inside the area of verbs, e.g., on the map in our earlier research [1]. However, one might wish to find a sparse encoding of the words in such a way that there would be a collection of features associated with each word. For instance, a word can be a verb, a copula and in past tense. It is an old idea in linguistics to associate words with features. The features can be syntactic as well as semantic However, these features are, as already mentioned, given by hand, and the membership is crisp.

Independent component analysis of word contexts

We have studied the emergence of linguistic representations through the analysis of words in contexts using the Independent Component Analysis (ICA) [3]. The ICA learns features automatically in an unsupervised manner. Several features for a word may exist, and the ICA gives the explicit values of each feature for each word. In our experiments, we have shown that the features coincide with known syntactic and semantic categories. As a simple example, the method is able to find a feature that is shared by words such as "must", "can" and "may", i.e. modal verbs.

In one of our experiments, we formed a context matrix $\mathbf{C}$ in which c_{ij} denotes the number of occurrences jth word in the immediate context of ith word, i.e, ith word followed by jth word with no words between them. This provided a 100×2000 matrix. A logarithm of the number of occurrences was taken in order a reduce the effect of the very most common words in the analysis and finally each word vector was normalized to unit length.

The results of the ICA analysis corresponded in most cases very well or at least reasonably well with our preliminary intuitions. The system was able to automatically create distributed representations as a meaningful collection of emergent linguistic features; each independent component was one such feature. For instance, Fig. 9.5 shows how the third component is strong in the case of nouns in singular form. A similar pattern was present in all the nouns with three exceptional cases with an additional strong fourth component indicated in Fig. 9.6. The reason appears to be that "psychology", "neuroscience", and "science" share a semantic feature of being a science or a scientific discipline. This group of words provide a clear example of distributed representation where, in this case, two components are involved.

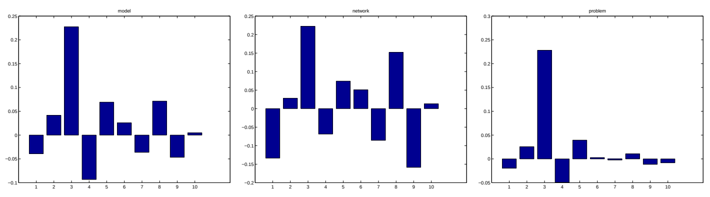

Figure 9.5: ICA features for "model", "network" and "problem".

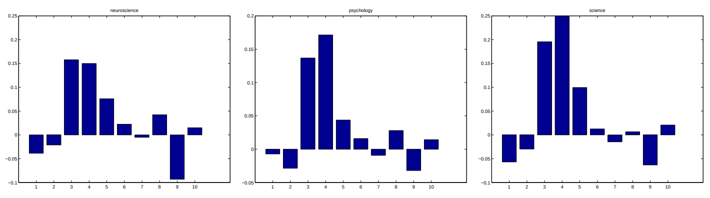

Figure 9.6: ICA features for "neuroscience", "psychology" and "science".

We have been able to show how independent component analysis can bring an additional advantage of finding explicit features that characterize words in an intuitively appealing manner. Independent component analysis appears to make possible a qualitatively new kind of result which have earlier been obtainable only through hand-made analysis. In the future, we will study more in detail what is the relationship between automatically acquired categories and manually defined ones.

References

[1] T. Honkela, V. Pulkki, and T. Kohonen. Contextual Relations of Words in Grimm Tales, Analyzed by Self-Organizing Map. Proceedings of ICANN'95, International Conference on Artificial Neural Networks, Paris, France, October 9-13, 1995, vol. 2, pp. 3-7.

[2] T. Honkela, A. Hyvärinen, and J. Väyrynen. Emergence of Linguistic Features using Independent Component Analysis. Helsinki University of Technology, Publications in Computer and Information Science Report, 2003.

[3] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* J. Wiley, 2001.

[4] H. Ritter, and T. Kohonen. Self-Organizing Semantic Maps. *Biological Cybernetics*, 61:241-254, 1989.