A Neural-Symbolic Cognitive Agent for Online Learning and Reasoning

H.L.H. (Leo) de Penning[1], A.S. d'Avila Garcez[2], Luís C. Lamb[3], John-Jules C. Meyer[4]

[1]TNO Behaviour and Societal Sciences, Soesterberg, The Netherlands, leo.depenning@tno.nl
[2]Department of Computing, City University, London, UK, aag@soi.city.ac.uk
[3]Instituto de Informática, UFRGS, Porto Alegre, Brazil, lamb@inf.ufrgs.br
[4]Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, jj@cs.uu.nl

Abstract

In real-world applications, the effective integration of learning and reasoning in a cognitive agent model is a difficult task. However, such integration may lead to a better understanding, use and construction of more realistic models. Unfortunately, existing models are either oversimplified or require much processing time, which is unsuitable for online learning and reasoning. Currently, controlled environments like training simulators do not effectively integrate learning and reasoning. In particular, higher-order concepts and cognitive abilities have many unknown temporal relations with the data, making it impossible to represent such relationships by hand. We introduce a novel cognitive agent model and architecture for online learning and reasoning that seeks to effectively represent, learn and reason in complex training environments. The agent architecture of the model combines neural learning with symbolic knowledge representation. It is capable of learning new hypotheses from observed data, and infer new beliefs based on these hypotheses. Furthermore, it deals with uncertainty and errors in the data using a Bayesian inference model. The validation of the model on real-time simulations and the results presented here indicate the promise of the approach when performing online learning and reasoning in real-world scenarios, with possible applications in a range of areas.

Introduction

The effective integration of automated learning and cognitive reasoning in real-world applications is a difficult task [Valiant, 2003]. Usually, most applications deal with large amounts of data observed in the real-world containing errors, missing values and inconsistencies. Even in controlled environments, like training simulators, integrated learning and reasoning is not very successful [Sandercock, 2004; Heuvelink, 2009]. Although the use of training simulators simplifies the data and knowledge acquisition, it is still very difficult to construct a cognitive model of an (intelligent) agent that is able to deal with the many complex relations in the observed data. When it comes to the assessment and training of high-order cognitive abilities (e.g. leadership, tactical manoeuvring, safe driving, etc.) training is still guided or done by human experts [Bosch and Riemersma, 2004]. The reason is that expert behaviour on high-level cognition is too complex to model, elicit and represent in an automated system. There can be many temporal relations between low and high-order aspects of a training task. Human behaviour is often non-deterministic and subjective (i.e. biased by personal experience and other factors like stress or fatigue) and what is known is often described vaguely and limited to explicit (i.e. "explainable") behaviour.

Several attempts have been made to tackle these problems. For instance [Fernlund et al., 2006] describes a number of systems that use machine learning to learn the complex relations from observation of experts and trainees during task execution. Although these systems are successful in learning and generalization, they lack the expressive power of logic-based (symbolic) systems and are therefore difficult to understand and validate [Smith and Kosslyn, 2006]. Alternatively, one could add probabilistic reasoning to logic-based systems [Heuvelink, 2009]. These systems perform better in expressing their internal knowledge as they are logic based and are able to deal with inconsistencies in the data because they reason with probabilities. Unfortunately, when it comes to knowledge representation and modelling these systems still require either statistical analysis of large amounts of data or knowledge representation by hand. Therefore, both approaches are time expensive and are not appropriate for use in real-time applications, which demand online learning and reasoning.

In this paper, we present a new cognitive agent model that is able to: (i) perform learning of complex temporal relations from uncertain observations, (ii) reason probabilistically about the knowledge that has been learned, and (iii) represent the agent's knowledge in a logic-based format for validation purposes. This is achieved by taking advantage of neural learning to perform robust learning and adaptation, and symbolic knowledge to represent qualitative reasoning. In addition, the model is validated in a training simulator employed in real-world scenarios, illustrating the effective use of the approach. The results show that the agent model is able to learn to perform automated driver assessment from observation of real-time simulation data and assessments by

driving instructors and that this knowledge can be extracted in the form of temporal logic rules.

Preliminaries

The construction of effective cognitive agent models is a long standing research endeavour in artificial intelligence, cognitive science, and multi-agent systems [Valiant, 2003; Wooldridge, 2009]. One of the main challenges toward achieving such models is the provision of integrated cognitive abilities, such as learning, reasoning and knowledge representation. Recently, cognitive computational models based on artificial neural networks have integrated inductive learning and deductive reasoning, see e.g. [Garcez et al., 2009; Lehmann et al., 2010]. In such models, neural networks are used to learn and reason about (an agent's) knowledge about the world, represented by symbolic logic. In order to do so, algorithms map logical theories (or knowledge about the world) T into a neural network N which computes the logical consequences of T. This provides also a learning system in the network that can be trained by examples using T as background knowledge. In agents endowed with neural computation, induction is typically seen as the process of changing the weights of a network in ways that reflect the statistical properties of a dataset, allowing for generalizations over unseen examples. In the same setting, deduction is the neural computation of output values as a response to input values *(stimuli* from the environment) given a particular set of weights. Such network computations have been shown equivalent to a range of temporal logic formalisms [Lamb et al., 2007]. Based on this approach the agent architecture of our model can be seen as a *Neural Symbolic Cognitive Agent* (NSCA). In our model, the agent architecture uses temporal logic as theory T and a Restricted Boltzmann Machine (RBM) as neural network N. A RBM is a partially connected neural network with two layers, a visible V and a hidden layer H, and symmetric connections W between these layers [Smolensky, 1986].

A RBM defines a probability distribution $P(V=v,H=h)$ over pairs of vectors v and h encoded in these layers, where v encodes the input data in binary or real values and h encodes the posterior probability $P(H|v)$. Such a network can be used to infer or reconstruct complete data vectors based on incomplete or inconsistent input data and therefore implement an auto-associative memory. It does so by combining the posterior probability distributions generated by each unit in the hidden layer with a conditional probability distribution for each unit in the visible layer. Each hidden unit constrains a different subset of the dimensions in the high-dimensional data presented at the visible layer and is therefore called an expert on some feature in the input data. Together, the hidden units form a so-called "Products of Experts" model that constrains all the dimensions in the input data.

A RBM has no connections between visible and other visible or hidden and other hidden units, making the experts conditionally independent. This restriction makes it possible to infer the posterior probabilities for each expert in parallel and train the network very effectively using Contrastive Divergence [Hinton, 2002]. This makes the learning and inference of a RBM very fast and therefore a suitable candidate for N in our agent architecture.

The Cognitive Model and Agent Architecture

Figure 1 presents the Neural-Symbolic Cognitive Agent (NSCA) architecture of our cognitive model. This model is able to encode domain knowledge in the form of temporal logic rules as hypotheses in a RBM, learn new hypotheses from observed data, reason with these hypotheses and extract new rules from these hypotheses in temporal logic form to update the domain knowledge.

The NSCA uses a Recurrent Temporal Restricted Boltzmann Machine (RTRBM) that encodes the temporal rules in the form of hypotheses about beliefs and previously-applied rules. This is possible due to recurrent connections between hidden unit activations at time t and the activations at time t-1, see [Sutskever et al., 2008]. Based on the Bayesian inference mechanism of the RTRBM each hidden unit H_j represents a hypothesis about a specific rule R_j that calculates the posterior probability that the rule implies a certain relation in the beliefs b being observed in the visible layer V, given the previously applied rules r^{t-1} (i.e. $P(R|B=b,R^{t-1}=r^{t-1})$). From these hypotheses the RTRBM selects the most applicable rules r using random Gaussian sampling of the posterior probability distribution (i.e. $r \sim P(R|B=b,R^{t-1}=r^{t-1})$) and calculates the conditional probability or likelihood of all beliefs given the selected rules are applied (i.e. $P(B|R=r)$). The difference between the observed and inferred beliefs can be used by the NSCA to train the RTRBM (i.e. update its weights) in order to improve the hypotheses about the observed data. There the RTRBM uses a combination of Contrastive Divergence and backpropagation through time, see [Sutskever et al., 2008].

In the spirit of BDI agents [Bratman, 1999], the observed data (e.g. simulation data, human assessments) is encoded as beliefs and the difference between the observed and inferred beliefs are the actual implications or intentions of the agent on its environment (e.g. adapting the assessment scores in the training simulator). The value of a belief represents either the probability of the occurrence of some event or state in the environment (e.g. *Raining=true*), or a real value (e.g. *Speed*=31.5). In other words, the NSCA deals with both binary and continuous data, by using a continuous stochastic visible layer [Chen and Murray, 2003]. As will become clear in our experiments, this improves the agent's ability to model asymmetric data, which in turn is very useful since measured data coming from a simulator is often asymmetric (e.g. training tasks typically take place in a restricted region of the simulated world).

Due to the stochastic nature of the sigmoid activation functions used in our model, the beliefs can be regarded as fuzzy sets with a Gaussian membership function. This allows us to represent vague concepts like *fast* and *slow*, as well as approximations of learned values, which is useful when reasoning with implicit and subjective knowledge [Sun, 1994].

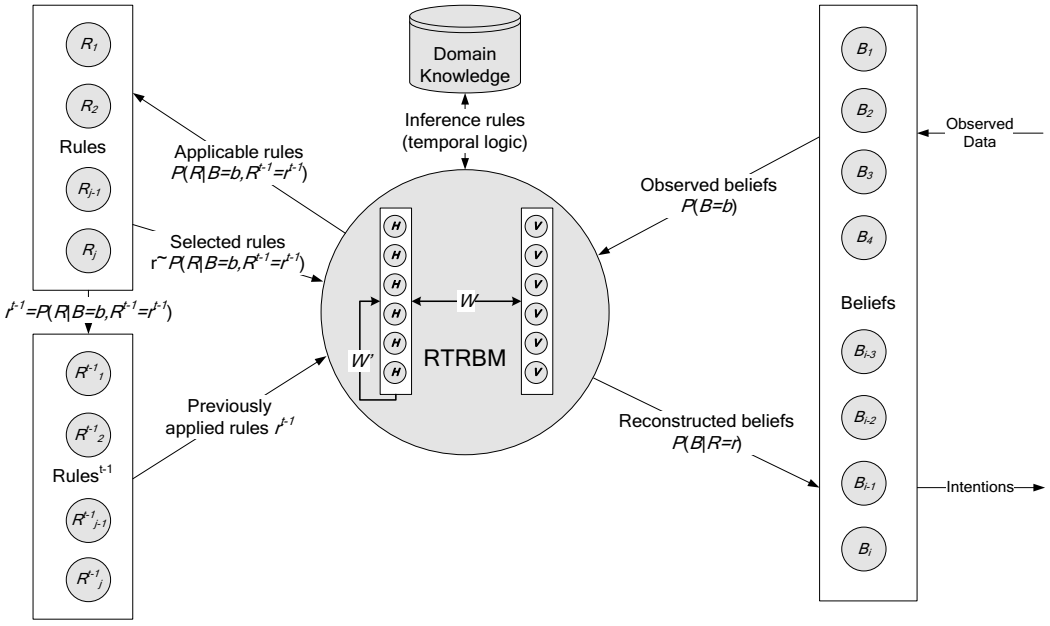

Figure 1. Global architecture of the Neural-Symbolic Cognitive Agent Model with Recurrent Temporal RBM.

Temporal Knowledge Representation

To represent domain knowledge in terms of beliefs and previously-applied rules we use the cognitive temporal logic described in [Lamb et al., 2007]. This logic contains several modal operators that extend classical modal logic with a notion of past and future. To express beliefs on continuous variables we extend this logic with the use of equality and inequality formulas (e.g. *Speed*<30, *Raining=true*). As an example let us take the task depicted in Figure 2. In this task, a trainee drives on an urban road and approaches an intersection. In this scenario the trainee has to apply the yield-to-the-right-rule. Using our extended temporal logic, we can describe rules about the conditions, scenario and assessment related to this task. In rules 1 to 4 below, $\Diamond A$ denotes "A is true sometime in the future" and $A \mathbb{S} B$ denotes "A has been true since the occurrence of B".

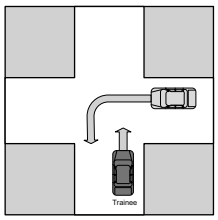

Figure 2. Example training task for driving simulation.

Conditions:

$(Weather \geq good)$ (1)

meaning: the weather is at least good

Scenario:

$ApproachingIntersection \wedge \Diamond(ApproachingTraffic = right)$ (2)

meaning: the car is approaching an intersection and sometime in the future traffic is approaching from the right

$((Speed > 0) \wedge HeadingIntersection)\ \mathbb{S}\ (DistanceIntersection < x) \rightarrow ApproachingIntersection$ (3)

meaning: if the car is moving and heading towards an intersection since it has been deemed close to the intersection*, then the car is approaching the intersection.*

Assessment:

$ApproachingIntersection \wedge (DistanceIntersection = 0) \wedge (ApproachingTraffic = \text{right}) \wedge (Speed = 0) \rightarrow (Evaluation = good)$ (4)

meaning: If the car is approaching an intersection and arrives at the intersection when traffic is coming from the right and stops then the trainee gets a good evaluation.

Rule 4 is an example of an uncertain notion that is highly subjective (the distance x at which a person is regarded as approaching an intersection is dependent on the situation and personal experience). When this rule is encoded in a RTRBM, it becomes possible to learn a more objective value for x based on the observed behaviour of different people in various scenarios. This exemplifies our main objective at combining reasoning and learning.

In our model, the temporal logic rules are represented by setting the weights W and W' (see Figure 1) in the RTRBM. Therefore the rules need to be translated to a form that relates only to the immediately previous time step (denoted by the temporal operator $\bullet$). A transformation algorithm for this is described in [Lamb et al., 2007]. Then we can encode any rule as a stochastic relation between the hidden unit that represents the rule, the visible units that represent the beliefs and the previous hidden unit activations that represent the applied rules in the previous time step. For example, the rule $\alpha \mathbb{S} \beta$ can be translated to the following rules: $\beta \rightarrow \alpha \mathbb{S} \beta$ and $\alpha \wedge \bullet(\alpha \mathbb{S} \beta)$

$\rightarrow \alpha S \beta$, where α and β are modelled by visible units, $\alpha S \beta$ by a hidden unit and $\bullet(\alpha S \beta)$ is modelled by a recurrent connection to the same hidden unit. [Pinkas, 1995] shows how to map these logic-based propositions into the energy function of a symmetric network and how to deal with uncertainty by introducing a notion of confidence, called "penalty". We have adapted those algorithms to temporal logic and the RTRBM and show how to encode temporal logic rules in RTRBM and how to extract revised rules from a trained RTRBM.

Rule Extraction Algorithm

Step 1: Finding maximum likelihood. In order to extract the temporal rules from the RTRBM, we need to find the states of the visible units (V) and hidden units (H) that lower the total energy in its energy function. This means finding the states that maximize the likelihood of each rule. These states form the local minima (stable states) in the energy function and depend on the weights encoded in the RTRBM. Using the RTRBM's inference mechanism we can find these states by sampling the belief states from the conditional probability distribution calculated for each rule at a time, given only that rule is selected (i.e., $H^{(j)}=1 \wedge H^{(x \neq j)}=0$). More formally, for each hidden unit $H^{(j)}$ we create a rule R_j and for each visible unit $V^{(i)}$ a belief B_i for which the conditional probabilities of the beliefs and previously applied rules $R^{t-1}{}_j$ (associated with the hidden unit activations at time $t-1$ denoted by $H_{t-1}{}^{(j)}$) are defined by:

$$P(B_i|R_j) = P(V^{(i)}|H^{(j)}=1, H^{(x \neq j)}=0) \quad (6)$$

$$P(R^{t-1}{}_j|R_j) = P(H_{t-1}{}^{(j)}|H^{(j)}=1, H^{(x \neq j)}=0) \quad (7)$$

For each rule R_j, the value of each belief in the belief base, denoted as a vector b_j is set to $P(B|R_j)$ (denoted by $\leftarrow$ in Eq. 8) and the value of each previously applied rules, denoted as $r^{t-1}{}_j$ is sampled (using a random Gaussian sampling) from $P(R^{t-1}|R_j)$ (this is denoted by $\sim$ in Eq. 9).

$$b_j \leftarrow P(B|R_j) \quad (8)$$

$$r^{t-1}{}_j \sim P(R^{t-1}|R_j) \quad (9)$$

Step 2. Generating a rule. When we have calculated the belief and previous rule states that maximize the likelihood of each rule, we can construct a temporal logic formula Ψ with temporal modalities using the following equations (where k is the number of beliefs, m the number of rules and w_{ij} is the weight of the connection between $V^{(i)}$ and $H^{(j)}$):

$$\psi = \left\{ \left\langle c_j : R_j \leftrightarrow \bigwedge_{i=1}^{k} \varphi_j^{(i)} \wedge \bigwedge_{l=1}^{m} \rho_j^{(l)} \right\rangle, \forall R_j \in R \right\} \quad (10)$$

$$\varphi_j^{(i)} = \begin{cases} B_i \leq b_j(i) & ,if \quad w_{ij} < 0 \\ B_i \geq b_j(i) & ,if \quad w_{ij} > 0 \\ \varnothing & ,if \quad w_{ij} = 0 \end{cases} \quad (11)$$

$$\rho_j^{(l)} = \begin{cases} \bullet R_l & ,if \quad r_j^{t-1}(l) = 1 \\ \bullet \neg R_l & ,if \quad r_j^{t-1}(l) = 0 \end{cases} \quad (12)$$

$$c_j = P(R_j = 1 \mid b_j, r_j^{t-1}) \quad (13)$$

The propositions on beliefs, denoted by $\varphi_j^{(i)}$, are calculated using Eq. 11 and depend on the weight w_{ij} of the connection between the hidden unit $H^{(j)}$ that represents rule R_j and visible unit $V^{(i)}$ that represents belief B_i. A negative weight between $V^{(i)}$ and $H^{(j)}$ will increase the probability of R_j when we decrease the value of B_i. So all values for belief B_i less or equal to $b_j(i)$ will increase the probability of rule R_j. The inverse applies to a positive weight. When the weight is zero a belief has no influence on the rule and can be left out.

The propositions on previously applied rules, denoted by $\rho_j^{(i)}$, are calculated using Eq. 12 and use the temporal operator $\bullet$. By applying the transformation algorithm in [Lamb et al., 2007] the generated rules can be further simplified for readability.

In the spirit of [Pinkas, 1995] and [Lee, 1997] we also calculate a confidence level c_j for each rule that denotes the probability that rule R_j is actually implied by the state defined in b_j and $r^{t-1}{}_j$ (see Eq. 13).

Rule Encoding Algorithm

The knowledge extraction algorithm above shows that temporal logic rules can be extracted from RTRBM reasonably efficiently. Encoding these rules is the dual of the extraction algorithm.

Step 1. Encoding a rule. For each rule defined by Eq. 10:

1. Add a hidden unit $H^{(j)}$ to the RTRBM to represent R_j.
2. For each belief B_i in the rule, add a visible unit $V^{(i)}$.
3. Randomize the weights connecting $V^{(i)}$ with $H^{(j)}$.

Step 2. Setting the weights. For each rule calculate the weights that maximize the likelihood of the beliefs B_j and previously applied rules $R^{t-1}{}_j$:

1. For each hidden unit $H^{(j)}$, calculate its activation $h^{(j)}$ using the inference algorithm in [Sutskever et al., 2008].
2. Set the value of hidden unit $H^{(j)}$ to 1 and the other hidden units $H^{(x \neq j)}$ to 0.
3. Infer the conditional probabilities of beliefs b' and previously applied rules r' given the current weights using eqs. 8 and 9.
4. Minimize the difference by minimizing the contrastive divergence between, resp., $h^{(j)}$, b' and r' and the values for c_j, B_j and $R^{t-1}{}_j$ as defined by rule R_j.

Experiments and Results

The cognitive model has been developed as part of a three year research project on assessment in driving simulators, carried out by TNO in cooperation with the Dutch licensing authority (CBR), Research Centre for Examination and Certification (RCEC), Rozendom Technologies, and ANWB driving schools. The NSCA is implemented using a multi-agent platform for Virtual Instruction [de Penning et al., 2008] and was used for an experiment on one of the

ANWB driving simulators. In this experiment 5 students participated in a driving test consisting of 5 test scenarios each. For each attempt all data from the simulator (i.e. 43 measurements, like relative positions and orientations of all traffic, speed, gear, and rpm of the student's car) and assessments scores (0-100) on several driving skills (i.e. vehicle control, economic driving, traffic flow, social-, and safe driving) that were provided by 3 driving instructors during the attempts were observed by the NSCA in real-time. The results in figures 3 and 4 show that the NSCA was able to learn from these observations and infer assessment scores that are similar to those of the driving instructors. For readability, the figures only show the actual (i.e. given by the instructor) and inferred beliefs (i.e. calculated by the NSCA) on assessment scores, but in fact the NSCA reconstructs beliefs over all aspects, including the expected position and orientation of other vehicles. This generative property of the RTRBM enables the NSCA to deal with uncertainty and missing data related to the observed beliefs based on the encoded rules. Also, the difference between actual and inferred beliefs can be translated into intentions to change a simulation in order to adapt to the student's level of expertise (i.e. adaptive training).

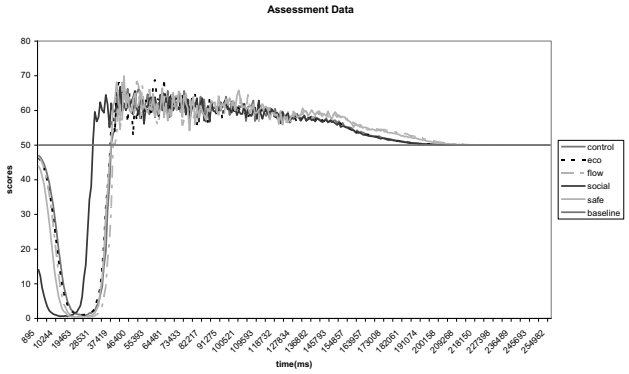

Figure 3. Actual and inferred assessment scores (i.e. vehicle control, economic driving, traffic flow, social driving and safe driving) when the student is doing nothing.

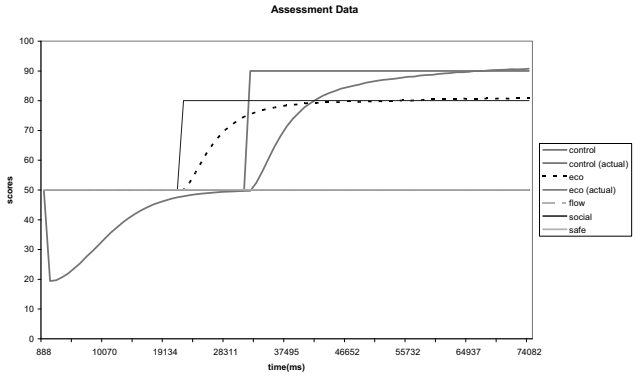

Figure 4. Actual and inferred assessment scores when instructor is assessing vehicle control and economic driving skills.

Eqs. 14 and 15 show some examples of learned hypotheses on temporal relations between observations, that were extracted from the NSCA. These rules imply relations on beliefs and previously- applied rules with a certain probability (i.e. confidence). In case of driving assessment, the implications are the assessment scores that are inferred from observed behaviour in a driving simulator. For readability we left out the beliefs that have insignificant impact on the truth-value of the rules based on the normalized weights of the beliefs in the rules.

$$0.72: R_{47} \leftrightarrow car\_Spatial\_Orientation\_Psi >= 0.062 \wedge car\_Spatial\_WorldLocation\_Y >= -2701220 \wedge car\_Spatial\_WorldLocation\_Z >= 1632605 \wedge cyclist\_Spatial\_Orientation\_Psi <= -0.088 \wedge cyclist\_Spatial\_WorldLocation\_X <= -163262 \wedge cyclist\_Spatial\_WorldLocation\_Z <= 1598489 \wedge mycar\_BrakeDepression <= 49.5 \wedge mycar\_EmergencyLight = false \wedge \mathbf{mycar\_Gear >= 2} \wedge mycar\_HighBeam = \text{false} \wedge mycar\_ParkingBrake = false \wedge mycar\_RightIndicator = false \wedge \text{objectives\_flow\_score} >= 50.56101934052543 \wedge \bullet R_3 \wedge \bullet R_4 \wedge \bullet R_6 \wedge \bullet R_8 \wedge \bullet R_9 \wedge \bullet R_{11} \wedge \bullet R_{13} \wedge \bullet R_{16} \wedge \bullet R_{23} \wedge \bullet R_{24} \wedge \bullet R_{25} \wedge \bullet R_{28} \wedge \bullet R_{42} \quad (14)$$

$$0.78: R_{13} \leftrightarrow car\_Spatial\_Orientation\_Phi <= -0.031 \wedge car\_Spatial\_WorldLocation\_Y >= -2702291 \wedge cyclist\_Spatial\_Orientation\_Phi >= 0.085 \wedge cyclist\_Spatial\_WorldLocation\_X <= -164763 \wedge mycar\_BrakeDepression <= 49.2 \wedge mycar\_EmergencyLight = false \wedge mycar\_FuelConsumption >= 0.0045 \wedge mycar\_Gear >= 1 \wedge mycar\_HighBeam = false \wedge \mathbf{mycar\_SteeringAngleVelocity >= -57.5} \wedge objectives\_flow\_score <= 49.78478771808131 \wedge objectives\_safe\_score <= 49.55145948925322 \wedge \bullet R_1 \wedge \bullet R_4 \wedge \bullet R_6 \wedge \bullet R_7 \wedge \bullet R_8 \wedge \bullet R_{10} \wedge \bullet R_{11} \wedge \bullet R_{12} \wedge \bullet R_{17} \wedge \bullet R_{21} \wedge \bullet R_{25} \wedge \bullet R_{28} \wedge \bullet R_{34} \wedge \bullet R_{44} \quad (15)$$

The rules in eq. 14 and 15 are the result of a small experiment and require further validation by experts, but one can see plausible effects that certain beliefs have on the outcome of the assessment scores. For example, when the gear in rule R_{47} is in position 2 or higher we can conjecture that this facilitates better traffic flow. And when the steering angle velocity in rule R_{13} is very high (i.e. the student's steering must have been very rough), one can imagine lower assessment scores on safe driving. More experiments with larger populations of students and instructors have been planned for this year for which we expect further improvement of the learned knowledge on driving assessment.

Conclusions and Future Work

The cognitive model and agent architecture presented in this paper offer an effective approach that integrates symbolic reasoning and neural learning in a unified model. This approach allows the agent to learn rules about observed data in complex, real-world environments (e.g. expert behaviour for training and assessment in simulators). Learned behaviour can be extracted to update existing domain knowledge for validation, reporting and feedback. Furthermore the approach allows domain knowledge to be encoded in the model and deals with

uncertainty in real-world data. Results described in this paper show that the agent is able to learn new hypotheses from observations and extract them into a temporal logic formula. Although initial results are promising, the model requires further validation by driving experts. More experiments are planned on ANWB driving simulators. This will allow further validation of the model in an operational setting with many scenarios, a large trainee population and multiple assessments by driving instructors. In parallel, the system will also be tested in other simulation domains, like jetfighter pilot training and for strategic command and control training. Other future work includes research on using Deep Boltzmann Machines [Salakhutdinov and Hinton, 2009] to find higher-level rules and the application of an RTRBM to facilitate adaptive training. In summary, we believe that our work provides an integrated model for knowledge representation, learning and reasoning which may indeed lead to realistic computational cognitive agent models, thus answering the challenges put forward in [Valiant, 2003; Wooldridge, 2009].

References

[Bosch and Riemersma, 2004] K. van den Bosch, and J.B.J. Riemersma. Reflections on scenario-based training in tactical command. *Scaled Worlds: Development, validation and applications*: 1–21, 2004.

[Bratman, 1999] M.E. Bratman. *Intention, Plans, and Practical Reason*. Cambridge University Press, 1999.

[Chen and Murray, 2003] H. Chen, and A.F. Murray. Continuous restricted Boltzmann machine with an implementable training algorithm. In *IEEE Proc. on Vision, Image and Signal Processing*, pages 153–158, 2003.

[Fernlund et al., 2006] H.K.G. Fernlund, A.J. Gonzalez, M. Georgiopoulos, and R. F. DeMara. Learning tactical human behavior through observation of human performance. *IEEE trans. on systems, man, and cybernetics, Part B*. 36: 128-140, 2006.

[Garcez et al., 2009] A.S. d'Avila Garcez, L.C. Lamb, and D.M. Gabbay. *Neural-Symbolic Cognitive Reasoning*. Springer-Verlag New York Inc, 2009.

[Heuvelink, 2009] A. Heuvelink. *Cognitive Models for Training Simulations*. Dutch Graduate School for Information and Knowledge Systems (SIKS), PhD. diss. no. 2009-24, Vrije Universiteit Amsterdam, 2009.

[Hinton, 2002] G.E. Hinton. Training products of experts by minimizing contrastive divergence. In *Neural computation* 14(8): 1771-1800, 2002.

[Lamb et al., 2007] L.C. Lamb, R.V. Borges, and A.S. d'Avila Garcez. A connectionist cognitive model for temporal synchronization and learning. In *Proc. of the AAAI Conference on Artificial Intelligence*, AAAI Press, pages 827-832, 2007.

[Lee, 1997] P.M. Lee. *Bayesian Statistics: An Introduction*. Third Ed. Arnold Publication, 1997.

[Lehmann et al., 2010] J. Lehmann, S. Bader, and P. Hitzler. Extracting reduced logic programs from artificial neural networks. *Applied Intelligence* 32(3): 249–266, 2010.

[de Penning et al., 2008] L. de Penning, E. Boot, and B. Kappé. Integrating Training Simulations and e-Learning Systems: The SimSCORM Platform. In *Proc. of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*, Orlando, USA, pages 1-9, 2008.

[Pinkas, 1995] G. Pinkas. Artificial Intelligence Reasoning , nonmonotonicity and learning in connectionist networks that capture propositional knowledge. *Artificial Intelligence* 77: 203-247, 1995.

[Salakhutdinov and Hinton, 2009] R. Salakhutdinov, and G.E. Hinton. Deep Boltzmann machines. In *Proc. of the Int. Conference on Artificial Intelligence and Statistics*, , pages 448–455, 2009.

[Sandercock, 2004] J. Sandercock. *Lessons for the construction of military simulators: a comparison of artificial intelligence with human-controlled actors. Technical Report, DSTO TR-1614*. Adelaide, South Australia, 2004.

[Smith and Kosslyn, 2006] E. Smith, and S. Kosslyn. *Cognitive Psychology: Mind and Brain*. Prentice-Hall, 2006.

[Smolensky, 1986] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In *Parallel Distributed Processing: Volume 1: Foundations*, eds. D. E. Rumelhart and J. L McClelland. MIT Press, Cambridge, MA, pages 194-281, 1986.

[Sun, 1994] R. Sun. A neural network model of causality. *IEEE Transactions on Neural Networks* 5(4): 604-611, 1994.

[Sutskever et al., 2008] I. Sutskever, G.E. Hinton, and G. Taylor. The recurrent temporal restricted boltzmann machine. *Neural Information Processing Systems (NIPS)*, 2008.

[Valiant, 2003] L.G. Valiant. Three problems in computer science. *Journal of the ACM (JACM)* 50(1): 96–99, 2003.

[Wooldridge, 2009] M. Wooldridge. *An introduction to multiagent systems*. 2nd ed. Wiley, 2009.