Neural-Symbolic Learning:How to play Soccer

Silvano Colombo Tosatto
University of Luxembourg
silvano.colombotosatto@uni.lu

Abstract

In the present extended abstract we describe the simulator developed. The purpose of this simulator is to test our neural symbolic approach towards normative reasoning.

After the translation process provided by the simulator I describe one of the case study used during the experiments. To be more precise, the case study regards RoboCup scenario.

1 The Simulator

The task of the simulator is to build a neural network starting from a knowledge base. For the translation, the simulator uses an approach similar to the one described in the book[d'Avila Garcez *et al.*, 2002]. More precisely the approach used is the one described in [Boella *et al.*, 2011a].

The simulator has been developed using *Java*[1]. To develop the necessaries functions for the neural networks and the networks themselves, *Joone*[2]. Joone is a object oriented neural engine, a framework written in Java that collects the functions needed to build neural networks and work with them.

2 Translation process

The translation process receives in input two *XML*[3] files. The first file contains the lists of input and output perceptrons to be included in the neural network. The second file contains the rules that have to be translated within the neural network by the *N-CILP* algorithm described in [Boella *et al.*, 2011a].

The input file containing the lists of inputs and outputs is needed for training purposes. Otherwise the neural network would be constructed using the inputs and outputs used in the rules inside the knowledge base, if this is the case, then the neural network would not be able to learn new rules containing literals which are not already known.

The knowledge base file contains I/O logic rules, shaped in XML format. The priority relationships between the rules are encoded within the rules themselves as described in [Boella *et al.*, 2011a].

[1]http://www.java.com/en/

[2]http://sourceforge.net/projects/joone/

[3]http://www.w3.org/XML/

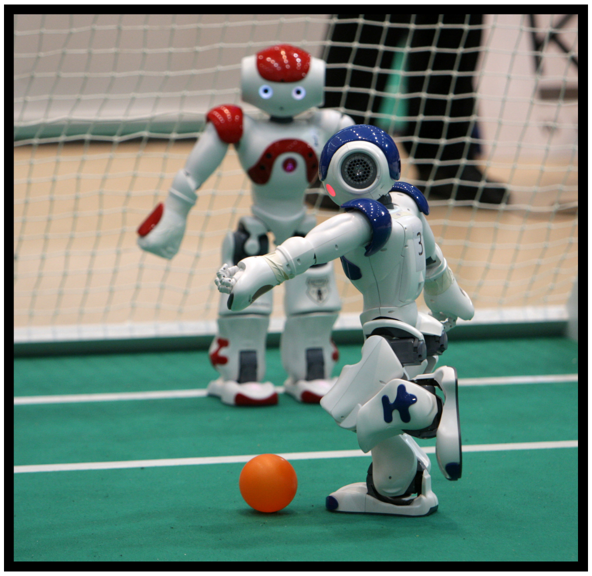

Figure 1: A snapshot of a RoboCup match.

With the mentioned files the simulator builds the neural network following the N-CILP algorithm. In addition, the simulator allows some optional translation features. The first option allows to set the weights calculated by N-CILP as untrainable. By activating this option, for the resulting neural network would be very difficult to forget the starting rules. Anyway a possible drawback of this approach is that would be difficult to forget *wrong* rules. The second option allows to randomize the weights and thresholds of the network[4]. The principal use of this option is to generate non-symbolic networks and compare their performances with the ones built with N-CILP.

3 Neural Network

The network built by the N-CILP algorithm uses a step activation function for the perceptrons in the input level, because they just need to pass the information from the inputs to the following layer. For the hidden and output level instead, the

[4]The number of hidden perceptrons is defined by the N-CILP algorithm.

perceptrons used adopts an bipolar sigmoid activation function [Karlik and Olgac, 2010].

After being built, the neural network can be used to process data. Thanks to the translation process, the neural network is capable to process the data by following the rules contained in the initial knowledge base.

In order to cope with a dynamically changing environment, the network can be trained in a supervised fashion. By instance learning the network is capable to extend its knowledge and after the training. In this way the trained neural network, should be able to correctly process data which the starting knowledge base is not.

4 Case Study

One of the case studies used with the simulator involves the RoboCup scenario (Figure 1). In our work described in [Boella *et al.*, 2011a] we focused on the normative aspect of the game. For this reason, we have took into consideration some of the rules that the robots had to follow. We used the rulings used in 2007 contained in [Menegatti, 2007].

4.1 Rules of the RoboCup

In this section we will take a look at some of the rules that the robots should follow in order to play properly. I represent the rules using the I/O logic format [Makinson and van der Torre, 2000] with modalities as described in [Boella *et al.*, 2011a].

$R_1 : (\top, \mathbf{O}(\neg impact\_opponent))$

$R_2 : (\top, \mathbf{O}(\neg use\_hands))$

$R_3 : (goalkeeper \wedge inside\_own\_area, \mathbf{P}(use\_hands))$

$R_4 : (kickoff, \mathbf{O}(\neg score))$

$R_5 : (kickoff \wedge mate\_touch\_ball, \mathbf{P}(score))$

The first rule refers to the prohibition to voluntarily impact into an opponent.

The second rule also states a prohibition, the interdiction to use the hands to play the ball. Differently, the third rule says that the goalkeeper can use its hands inside its own goal area.

The fourth rule refers to the prohibition to score from the kickoff. The last rule, the fifth, is related with the fourth. It states that it is permitted to score in a kickoff situation if a team mate touches the ball.

4.2 Adding dynamism in the system

By only considering the rules of the game, the system is static. This because the rules of the game does not change during a match. In order to include dynamism, we add to the system another ruling element[5]. The additional ruling element is represented by the *coach*.

The coach gives directions to the robot about how they should play during the match. The situation during the match can change, and the coach may want to change accordingly the strategies of the robot team. To do so, the coach can add additional rules to the knowledge base or retract some of them.

[5]The first ruling element is the *referee* which enforces the rules of the game.

In the following list we take a look at some of the possible rules that a coach may enforce.

$R_6 : (ball \wedge close\_to\_opponent\_goal, \mathbf{O}(shoot))$

$R_7 : (ball \wedge opponent\_approaching, \mathbf{O}(pass))$

$R_8 : (ball \wedge opponent\_approaching \wedge team\_mate\_marked, \mathbf{O}(\neg pass))$

$R_9 : (opponent\_shooting, \mathbf{O}(impact\_opponent))$

The sixth rule refers to the ought to try to score when a robot with the ball, is close to the opponent's goal.

The seventh rule states the obligation to pass the ball if an opponent is approaching. Differently, the eight rule states the prohibition to pass, if the same condition holds and additionally the team mate is marked by an opponent. In this case the rules can be ordered by a priority relation in order to avoid *dilemmas*[Boella *et al.*, 2011b], like $R_8 \succ R_7$.

The last rule, the ninth, states that a robot should try to prevent an opponent to scoring by impacting into him if it is shooting. This rule clearly goes against R_1 given by the referee. Even in this case is possible to enforce the decision of the robot with a priority (like $R_9 \succ R_1$) or leave at the robot both the possibilities available ($impact\_opponent$ or $\neg impact\_opponent$).

Acknowledgements

Silvano Colombo Tosatto is supported by the National Research Fund, Luxembourg.

References

[Boella *et al.*, 2011a] Guido Boella, Silvano Colombo Tosatto, Artur S. d'Avila Garcez, Valerio Genovese, and Leendert van der Torre. Embedding normative reasoning into neural symbolic systems. In *7th International Workshop on Neural-Symbolic Learning and Reasoning*, 2011.

[Boella *et al.*, 2011b] Guido Boella, Silvano Colombo Tosatto, Artur S. d'Avila Garcez, Dino Ienco, Valerio Genovese, and Leendert van der Torre. Neural symbolic systems for normative agents. In *10th International Conference on Autonomous Agents and Multiagent Systems*, 2011.

[d'Avila Garcez *et al.*, 2002] Artur S. d'Avila Garcez, Krysia B. Broda, and Dov M. Gabbay. *Neural-Symbolic Learning Systems*. Perspectives in Neural Computing. Springer, 2002.

[Karlik and Olgac, 2010] Bekir Karlik and A Vehbi Olgac. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1:111–122, 2010.

[Makinson and van der Torre, 2000] David Makinson and Leendert van der Torre. Input-output logics. *Journal of Philosophical Logic*, 29, 2000.

[Menegatti, 2007] Emanuele Menegatti. Robocup soccer humanoid league rules and setup, 2007.