Category-based Inductive Learning in Shared NeMuS

Ana Carolina Melik Schramm[1], Edjard de Souza Mota[1],
Jacob M. Howe[2], and Artur S. d'Avila Garcez[2]

[1] Universidade Federal do Amazonas,
Instituto de Computação, Campus Setor Norte
Coroado - Manaus - AM - Brasil CEP: 69080-900
{acms, edjard}@icomp.ufam.edu.br,
[2] City, University of London, London, EC1V 0HB, UK
{J.M.Howe,a.garcez}@city.ac.uk

1 Introduction

One of the main objectives of cognitive science is to use abstraction to create models that represent accurately the cognitive processes that constitute learning, such as categorisation. Relational knowledge is important in this task, since through the reasoning processes of induction and analogy over relations that the mind "creates" categories (it later estabilishes causal relations between them by using induction and abduction), and analogies exemplify crucial properties of relational processing, like structure-consistent mapping[2].

Given the complexity of the task, no model today has accomplished it completely. The associacionist/connectionist approach represents those processes through associations between different informations. That is done by using artificial neural networks. However, it faces a great obstacle: the idea (called propositional fixation) that neural networks could not represent relational knowledge. A recent attempt to tackle the symbolic extraction from artificial neural networks was proposed in [1]

The cognitive agent Amao uses a shared Neural Multi-Space (Shared NeMuS) of coded first-order expressions to model the various aspects of logical formulae as separate spaces, with importance vectors of different sizes. Amao [4] uses inverse unification as the generalization mechanism for learning from a set of logically connected expressions of the Herbrand Base (HB). Here we present an experiment to use such learning mechanism to model a simple version of train set from Michalski's train problem[3].

2 Shared NeMuS Approach to Train Problem

In Michalski's train problem, there are 10 trains: 5 eastbound and 5 westbound. Whether a train is going east or west is determined by its properties. Using these trains, a simple base has been created, taking into account the size of the train wagons (short or not) and whether these wagons are closed or not. The number of wheels, wagon format and other attributes have been ignored in order to make the base simpler.

All the eastbound trains have at least one wagon which is both short and closed. That is what determines whether a train is eastbound or westbound. The idea is to use the shared NeMuS structure to induce the rule eastbound knowing that `t1` (the first train) is going east. Having that information, we can directly get all predicate instances, called as *bindings*, which have `t1` is an attribute. They are the following:

```
train(t1).
car(t1, c1_t1).
car(t1, c2_t1).
car(t1, c3_t1).
car(t1, c4_t1).

short(c1_t1).
closed(c1_t1).
```

The predicate `car` links `t1` to all its wagons (or carriages), so `car(t1, c1_t1)` means that `c1_t1` is a wagon that belongs to `t1`. Taking the first instance of the predicate `car`, we now know that `t1` has a wagon named `c1_t1`. Amao, through its shared NeMuS, accesses `c1_t1`'s bindings and using a polynomial search, finds both occurrences of `c1_t1` in `short` and `closed`, as seen above. This mechanism is called linkage pattern in Amao's learning mechanism.

At this point `t1` is a train that has `c1_t1` as a wagon, and this wagon is not closed. Amao also has the linkage predicate connecting both `c1_t1` and `t1`. Thus, a candidate hypothesis generated would look like `eastbound(X)` ← `car(X, Y)` ∧ ∼ `short(Y)` ∧ ∼ `closed(Y)`. However, this may not be the only possible hypothesis, so the other wagons being carried by `t1` need to be considered.

```
short(c2_t1).       ∼short(c3_t1).       short(c4_t1).
closed(c2_t1.       ∼closed(c3_t1).      ∼closed(c4_t1).
```

Among the possible hypotheses that may define a train as being eastbound, we have:

```
eastbound(X) ← car(X, Y) ∧ ∼short(Y) ∧ ∼closed(Y).
eastbound(X) ← car(X, Y) ∧ short(Y) ∧ closed(Y).
eastbound(X) ← car(X, Y) ∧ short(Y) ∧ ∼closed(Y).
```

Adding negative examples, we can reduce the number of possible hypotheses. In this case, the simplest way to do that is to use the 10th train `t10` as a negative example. Using the same method as explained above, the structure can select all predicates that have `t10` as an attribute:

```
car(t10, c1_t10).
car(t10, c2_t10).
```

Then, all the predicates that have `t10s` wagons as attributes:

```
short(c1_t10).          ∼short(c2_t10).
∼closed(c1_t10).        ∼closed(c2_t10).
```

Thus, the hypotheses that definitely do not define a train as being eastbound are:

eastbound(X) ← car(X, Y) ∧ short(Y) ∧ ∼closed(Y).
eastbound(X) ← car(X, Y) ∧ ∼short(Y) ∧ ∼closed(Y).

Both hypotheses are among the possible options defined above. Excluding them, the correct option remains. The target eastbound(X) can be defined by:

eastbound(X) ← car(X, Y) ∧ short(Y) ∧ closed(Y).

Formalizing what was explained above:

1. With the positive example (t1), get all predicates (bindings) that have t1 as an attribute;
2. Access bindings of attributes linked to t1 using polynomial search (linkage pattern)
   – in this case, the attributes are c1_t1, c2_t1 and c3_t1
3. repeat the first two steps for the negative example (t10)]
   – in this case, the attributes linked to t10 are c1_t10 and c2_t10
4. if there are hypotheses generated by using the positive example that are repeated in the negative example, they will not be in the list of possible hypotheses.
   – some of the hypotheses generated by using only the positive example are:
     eastbound(X) ← car(X, Y) ∧ ∼short(Y) ∧ ∼closed(Y).
     eastbound(X) ← car(X, Y) ∧ short(Y) ∧ closed(Y).
     eastbound(X) ← car(X, Y) ∧ short(Y) ∧ ∼closed(Y).
     However, using only the negative example, the first and third hypotheses would also be generated. By using both examples, these two don't make it into the list of possible hypotheses, and the correct one, which is eastbound(X) ← car(X, Y) ∧ short(Y) ∧ closed(Y), remains.

3 Concluding Remarks

The knowledge base created is only a simplification of the original train problem. As explained before, many attributes such as number of wheels, wagon format, load shape and roof shape have been ignored. Had they been included, more hypotheses could have been generated through Amao's inductive learning mechanism over the shared NeMuS. One current limitation is not being able to deal with predicate invention, that would allow to automatically create categories by means of abstraction/new predicates.

One possible road to explore is to take advantage of shared NeMuS weights to integrate a neural network classification method to help identify categories. In the train set, we know which trains are eastbound, but whatever rule defines the eastbound category is not known before using Amao to define it. Understanding what makes a train eastbound or not can help us categorize any train that might be added to the set in the future.

Another goal we aim to pursue is to make use of weights to implement neural mechanisms. We expect to envisage more efficient heuristics to guide hypotheses generation, improving Amao's learning mechanism.

References

1. França, M.V.M., D'Avila Garcez, A.S., Zaverucha, G.: Relational knowledge extraction from neural networks. In: Proceedings of the 2015th International Conference on Cognitive Computation: Integrating Neural and Symbolic Approaches - Volume 1583. pp. 146–154. COCO'15, CEUR-WS.org, Aachen, Germany, Germany (2015), http://dl.acm.org/citation.cfm?id=2996831.2996849
2. Halford, G.S., Wilson, W.H., Phillips, S.: Relational knowledge: the foundation of higher cognition 14, 597–505 (2010)
3. Larson, J.B., Michalski, R.S.: Inductive Inference of VL Decision Rules 14, 16–20 (1977)
4. Mota, E.d.S., Howe, J., Garcez, A.: In: Besold, T.R., d'Avila Garcez, A., Noble, I. (eds.) To appear NeSy 2017 Neural-Symbolic Learning and Reasoning (July)