A Neural-Symbolic Approach to the Contemporary Theory of Metaphor

Guido Boella
Università di Torino
boella@di.unito.it

Artur d'Avila Garcez
City University London
aag@soi.city.ac.uk

Alan Perotti
Università di Torino
perotti@di.unito.it

Abstract

Lakoff defined the metaphor as a mapping between knowledge domains ([5]). The cognitive role of metaphor is to reuse the knowledge about a source domain we have expertiese on in order to reason about another target domain. We propose in this paper a model of metaphor to implement the idea of reusing existing knowledge about one domain in another one. We propose a model of metaphor using a neural approach, first of all to mimic the neural model of metaphor of our brain [4] and secondly to exploit the learning capability of neural networks to handle the limited knowledge about the target domain. Moreover, since we assume that in some cases partial knowledge about both the target and source domain can be already available in the form of symbolic declarative knowledge, we adopt a neural symbolic approach [1] to compile this knowledge into a neural network. This approach allows also for the inverse process: to extract symbolic declarative knowledge after the learning phase.

To build a neural model, first we formalize the definition of metaphor by Lakoff as a monomorphism. To model the invertibility of the mapping in a monomorphism we cannot simply use feedforward models but we resort to RBMs because they display symmetric connections between layers.

Our approach can be used for software reuse and flexible commitment in multiagent systems.

1 The contemporary theory of metaphor

In *Metaphors we Live by* ([5]), Lakoff claimed that the generalizations governing metaphorical expressions are not in language, but in thought: they are general mappings across conceptual domains. Moreover, these general principles -which take the form of conceptual mappings- shape the way we conceptualize one mental domain in terms of another. The main example featured in that work is the expression *Our relationship has hit a dead-end street*. Here love is being conceptualized as a journey, with the implication that the relationship is stalled, that the lovers cannot keep going the way they've been going, that they must turn back, or abandon the relationship altogether. Since knowledge about journeys is used to explore and explain issues in the domain of relationships, *Relationships* is called *target domain* and *Travelling* is the *source domain*. This makes metaphorical inferences possible; mapped source domain inferences combine with target domain knowledge via binding to produce new inferences: If lovers are "stuck" in relationship, if the relationship isn't "going anywhere," then they are not making progress toward common life goals. If the lovers are "going in different directions," then they may not be able get to the same destinations, which means metaphorically that their common life goals may be inconsistent.

Lakoff and Johnson's "Metaphors We Live By" was written back in 1979, before the era of brain science and neural computation. However, as discussed in [4], the theory is compatible with what we know about the brain. A metaphor mapping in neural terms is a complex circuit which, when activated, activates many other circuits via linking and binding circuitry. A metaphorical inference

at the neural level occurs when: a metaphorical mapping is activated in a neural circuit, there is an inference in the source domain of the mapping, and a consequence of the source domain inference is mapped to the target domain, activating a meaningful node. Inferences are new activations that arise via prior activations. In situations where the source and target domains are both active simultaneously, the two areas of the brain for the source and target domains will both be active. Via the Hebbian principle that neurons that fire together wire together, neural mapping circuits linking the two domains will be learned. Those circuits constitute the metaphor.

In Lakoff's work, the metaphor is defined as a *mapping*, without further specification. In order to give a more formal characterization of it, we model the metaphor as an injective *omomorphism*, called *monomorphism*.

Definition 1 (Monomorphism)
Given two algebraic structures A and B, a function m is a monomorphism iff:

- *m is injective*
- *$\forall$ n-ary operation f over the structures and $\forall$ n-tuple $x_1, .., x_n$ of A,*

$$m(f_A(x_1, .., x_n)) = f_B(m(x_1), .., m(x_n))$$

where f_A and f_B represent f over A and B respectively.

In our setting, we can't compute f_A, and we wonder what could $f_A(x_1, .., x_n)$ be. Since m is injective, it can be inverted. Let n be the inverse function of m. The following transformations hold:

$$f_A(x_1, .., x_n) \equiv^1 n(m(f_A(x_1, .., x_n))) \equiv^2 n(f_B(m(x_1), .., m(x_n)))$$

Where ($\equiv^1$) is justified because m and n are inverse functions (and therefore $n(m(x)) \equiv x$) and ($\equiv^2$) follows from the definition of monomorphism.
This equivalence is displayed in Figure 1, where S and T represent the source and target domain respectively.

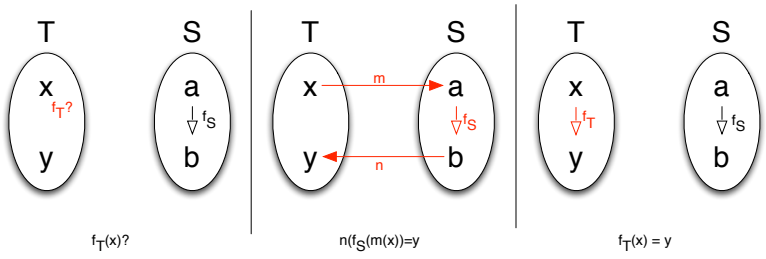

Figure 1: Monomorphism

Thanks to this equivalence, we can use the source domain S to perform computations over elements of the target domain T. Intuitively, we move the problem from the target domain to the source domain, we resolve it, and then map back the results onto the target domain, thus linking previously unrelated elements by a causal relation.

2 Neural models

To model metaphor as a monomorphism, we need to model the source domain and then to connect it to the target domain with an invertible mapping. Since we only connect the input and output of the source domain with the target one, we could consider it as a black box. However, in order to exploit the possibility that the source domain is created from a symbolic knowledge base and incrementally updated with new knowledge coming from examples, we model it as a connectionist

inductive learning and logic programming (CILP) system in the neural-symbolic paradigm of [1]. The mapping, however, being invertible, cannot be modeled by this approach, so we resort to RBM still in the neural-symbolic paradigm ([2]).

2.1 The neural-symbolic paradigm

The goal of neural-symbolic computation is to provide a coherent, unifying view for symbolic reasoning and connectionist learning and produce better computational tools for integrated machine learning and reasoning. Symbolic and subsymbolic models are combined to form an integrated model of computation; such hybrid systems are designed for explaining both paradigms one in terms of the other (that is, providing a neural implementation of a logic and a logical characterization of a neural system) and for combining advantages of the two approaches, building robust tools that can offer principled knowledge representation, learning and computation.
The CILP system ([1]) is a neural-symbolic system showing a one-to-one correspondence between logic programming and neural networks that are trainable by backpropagation.

2.2 RBM and contrastive divergence

We model the mapping functions m and n as a single restricted Boltzmann machine (RBM). Note that the bidirectionality of the link in the network (and the symmetry in the weights) structurally corresponds to our imposed inverse relation between m and n. If some information about the mapping between domains is given, we can use the encoding algorithms by de Penning et al. [2] to encode it into the RBM. After the reasoning phase has been performed, contrastive divergence can be used [3] to fine-tune the RBM. 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 ([6]). 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). 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 ([3]), thus making the learning and inference of a RBM very fast.

2.3 Example

We propose an example to illustrate and visualize our proposal. For the sake of explanation, let the mapping be given and let the knowledge over the domains be logic programs.
As a working example, consider the CAREER IS A JOURNEY example. Suppose that an agent, with a previous knowledge about the vehicles domain, is faced with an issue from the business domain: *My career is stuck*. The agent does not know how to treat such information, but she is given a mapping between the vehicles domain and the business one. The initial situation is visualized in Figure 2.

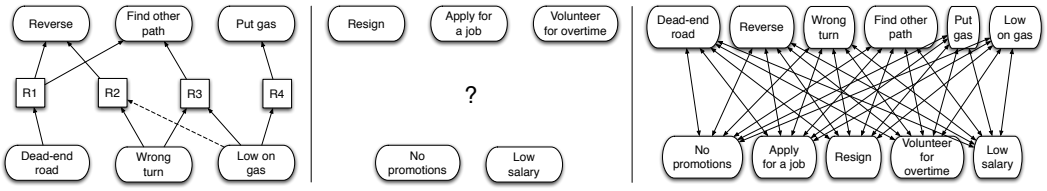

Figure 2: Initial setting

Figure 3 visualizes how, exploiting the aforementioned equivalence, the agent can learn a new rule over the business domain. First of all, the given information is mapped from the target domain to the source domain (Figure 3.a), discovering that *my career is stuck* is mapped onto *I reached a*

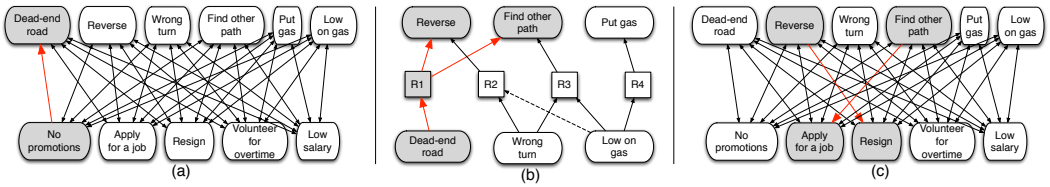

Figure 3: Applying the metaphor

dead-end road. Now the agent can manipulate this new chunk of information, inferring conclusions from it: the agent knows that, in case of *dead-end road*, the solution is to *put the car in reverse* and *find an alternative route* (Figure 3.b). This can be mapped back to the business domain: *put the car in reverse* corresponds to *resign* and *find an alternative route* to *apply for a new job* (Figure 3.c). Using the metaphor as a mapping, the agent has learned that, in order to overcome a *stuck-career* issue, she can *resign* and *apply for a new job*.

Note that the new rules over the target domain are learned as weights in a network and then extracted as symbolic knowledge. In the general case, the agent can assign a score to the new rule learned, and this can be used for supervisedly fine-tune the networks.

3 Applications

First of all, our approach allows agents to recycle knowledge from one domain to another (in our example, from the vehicles domain to the business one). Secondly, agents can recycle their *mental schemata*, since even with the same mapping, agents following some patterns over the same source domain will behave analogously over the same target domain. The metaphor encapsulates previous knowledge and makes it applicable to new domain. The neural-symbolic approach allows for our proposal to be robust and have explanatory power at the same time, since new symbolic rules can be extracted from the network modeling the target domain.
Moreover, a library of metaphors-as-interfaces would make commitment in multi-agent systems way more flexible and easy, since given an interface each agent could perform local learning/reasoning over an unknown domain, providing suggestions from her own experience at the same time.

4 Conclusions

In this work, we model the cognitive theory of metaphor, as defined by Lakoff, as a monomorphism. With this approach we are able to prove that local computation can be performed over a more familiar domain. We propose a framework that relies on the CILP system and RBMs and allows to perform learning and reasoning over unknown domains.

References

[1] A. d'Avila Garcez, K. B. Broda, and D. M. Gabbay. *Neural-Symbolic Learning Systems*. Perspectives in Neural Computing. Springer, 2002.

[2] L. de Penning, A. S. d'Avila Garcez, L. C. Lamb, and J.-J. C. Meyer. A neural-symbolic cognitive agent for online learning and reasoning. In *IJCAI*, pages 1653–1658, 2011.

[3] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14:1771–1800, August 2002.

[4] G. Lakoff. *The Neural Theory of Metaphor and Thought*, page 17–39. Cambridge University Press, Cambridge, 2008.

[5] G. Lakoff and M. Johnson. *Metaphors we Live by*. University of Chicago Press, Chicago, 1980.

[6] P. Smolensky. *Information processing in dynamical systems: foundations of harmony theory*, pages 194–281. MIT Press, Cambridge, MA, USA, 1986.