Unsupervised Neural-Symbolic Integration

Son N. Tran
The Australian E-Health Research Center, CSIRO
son.tran@csiro.au

Abstract

Symbolic has been long considered as a language of human intelligence while neural networks have advantages of robust computation and dealing with noisy data. Integration of neural-symbolic can offer better learning and reasoning while providing a means for interpretability through the representation of symbolic knowledge. Although previous works focus intensively on supervised feedforward neural networks, little has been done for the unsupervised counterparts. In this paper we show how to integrate symbolic knowledge into unsupervised neural networks. We exemplify our approach with knowledge in different forms, including propositional logic for DNA promoter prediction and first-order logic for understanding family relationship.

1 Introduction

An interesting topic in AI is integration of symbolic and neural networks, two different information processing paradigms. While the former is the key of higher level of intelligence the latter is well known for the capability of effective learning from data. In the last two decades, researchers have been working on the idea that combination of neural networks and symbolic representation of knowledge should offer joint benefits [Towell and Shavlik, 1994; Smolensky, 1995; Avila Garcez and Zaverucha, 1999; Valiant, 2006; Garcez *et al.*, 2008; Penning *et al.*, 2011; França *et al.*, 2014; Tran and Garcez, 2016].

In previous work, supervised neural networks have been used intensively for the integration based on the analogy of *modus ponens* inference with symbolic rules and forward passing in neural networks [Towell and Shavlik, 1994; Avila Garcez and Zaverucha, 1999]. In such networks, due to the discriminative structures only a subset of variables can be inferred, i.e. the variables in the left hand of *if-then* $\leftarrow$ formulas. This may limit their use in general reasoning. Unsupervised network, on the other hand, offers more flexible inference mechanism which seems more suitable for symbolic reasoning. Let us consider an XOR example $\mathrm{z} \leftrightarrow (\mathrm{x} \oplus \mathrm{y})$. Here, given the truth values of any two variables one can infer the rest. For supervised networks, a class variable must be discriminated from the others and only it can be inferred. An unsupervised network, in contrast, do not require such discrimination.

Encoding symbolic knowledge in an unsupervised neural network needs a mechanism to convert symbolic formulas to the network without loss of generality. In previous work, Penalty logic shows that any propositional formula can be represented in a symmetric connectionist network (SCN) where inference with rules is equivalent to minimising the network's energy [Pinkas, 1995]. However, SCN uses dense connections of hidden and visible units which make the inference very computational. Recent work shows that any propositional formula can be represented in restricted Boltzmann machines(RBMs) [Tran, 2017]. Different from Penalty logic, here the RBM is a simplified version of SCN where there is no visible-visible and hidden-hidden connections. This makes inference in RBMs is easier.

Several attempts have been made recently to integrate symbolic representation and RBMs [Penning *et al.*, 2011; Tran and Garcez, 2016]. Despite achieving good practical results they are still heuristic. In this paper, we show how to encode symbolic knowledge in both propositional and first-order forms into the RBM by extending the theory in [Tran, 2017].

The remainder of this paper is organized as follows. Section 2 reviews the idea of Confidence rule, a knowledge form to represent symbolic formulas in RBMs. In section 3 we show how to encode knowledge into RBMs. Section 4 presents the empirical verification of our encoding approach and Section 5 concludes the work.

2 Confidence Rules: Revisit

A confidence rule [Tran and d'Avila Garcez, 2013; Tran and Garcez, 2016] is a propositional formula in the form:

$$c : \mathrm{h} \leftrightarrow \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k \tag{1}$$

where h is called *hypothesis*, c is a non-negative real value called *confidence value*. Inference with a confidence rule is to find the model that makes the hypothesis h holds. If there exist a target variable y the inference of such variable will be similar to *modus ponens*, as shown in Table 1

An interesting feature of Confidence rules is that one can represent them in an RBM where Gibbs sampling can be seen

Confidence rule inference	Modus ponens
$\mathrm{h} \leftrightarrow \bigwedge_{t\in T} \mathrm{x}_t \wedge \bigwedge_{k\in K} \neg \mathrm{x}_k \wedge \mathrm{y}$	$\mathrm{y} \leftarrow \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k$
$\{\mathrm{x}_t, \neg \mathrm{x}_k \mid \text{for } \forall t \in T, \forall k \in K\}$	$\{\mathrm{x}_t, \neg \mathrm{x}_k \mid \text{for } \forall t \in T, \forall k \in K\}$
y	y

Table 1: Confidence rule and Modus ponens

equivalently as maximising the total (weighted) satisfiability [Tran, 2017]. If a knowledge base is converted into Confidence rules then we can take the advantage of the computation mechanism in such neural networks for efficient inference. The equivalence between confidence rules and an RBM is defined in that the satisfiability of a formula is inversely proportional to the energy of a network:

$$s_\varphi(\mathbf{x}) = -aE_{rank}(\mathbf{x}) + b$$

where s_φ is the truth value of the formula φ given an assignment $\mathbf{x}$; $E_{rank}(\mathbf{x}) = min_{\mathbf{h}} E(\mathbf{x}, \mathbf{h})$ is the energy function minimised over all hidden variables; $a > 0$,b are scalars.

By using disjunctive normal form (DNF) to present knowledge Confidence rules attract some criticism for practicality since it is more popular to convert a formula to a conjunctive normal form of polynomial size. However, we will show that Confidence rules are still very useful in practice. In fact, in such tasks as knowledge extraction, transfer, and integration Confidence rules have been already employed [Penning *et al.*, 2011; Tran and d'Avila Garcez, 2013; Tran and Garcez, 2016]. For knowledge integration previous work separates the *if-and-only-if* symbol in Confidence rules into two *if-then* rules to encode in a hierarchical network [Tran and Garcez, 2016]. In this work, we show that such separation is not necessary since any propositional *if-then* formulas can be efficiently converted to Confidence rules. The details are in the next section.

3 Knowledge Encoding

In many cases background knowledge presents a set of *if-then* formulas (or equivalent Horn clauses). This section shows how to convert them into Confidence rules for both propositional and first-order logic forms

3.1 Proposition Logic

A propositional *if-then* formula has the form

$$c : \mathrm{y} \leftarrow \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k$$

which can be transformed to a DNF as:

$$c : (\mathrm{y} \wedge \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k) \vee \bigvee_t (\neg \mathrm{x}_t) \vee \bigvee_k (\mathrm{x}_k)$$

and then to the confidence rules:

$$c : \mathrm{h}_y \leftrightarrow \mathrm{y} \wedge \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k$$
$$c : \mathrm{h}_t \leftrightarrow \neg \mathrm{x}_t \text{ for } \forall t$$
$$c : \mathrm{h}_k \leftrightarrow \mathrm{x}_k \text{ for } \forall k$$

Encoding these rules into an RBM does not guarantee the equivalence. This is because it violates the condition that the DNF of a formula should *have at most one conjunct is true given an assignment* [Tran, 2017]. Fortunately this can be solved by grouping $\neg \mathrm{x}_t$, x_k with a max-pooling hidden unit which results in an RBM with the energy function as[1][2]:

$$\begin{aligned} E = &- c \times h_y(y + \sum_t x_t - \sum_k x_k - |T| - 1 + \epsilon) \\ &- c \times h_p \max(\{-x_t + \epsilon, x_k - 1 + \epsilon | t \in T, k \in K\}) \end{aligned} \quad (2)$$

Here a max pooling hidden unit represents a hypothesis: $\mathrm{h}_p \leftrightarrow \bigvee_t \mathrm{h}_t \vee \bigvee_k \mathrm{h}_k$ which, in this case, can be written as: $c : \mathrm{h}_p \leftrightarrow \bigvee_t \neg \mathrm{x}_t \vee \bigvee_k \mathrm{x}_k$. The final set rules are:

$$c : \mathrm{h}_y \leftrightarrow \mathrm{y} \wedge \bigwedge_t \mathrm{x}_t \wedge \bigwedge_k \neg \mathrm{x}_k$$
$$c : \mathrm{h}_p \leftrightarrow \bigvee_t \neg \mathrm{x}_t \vee \bigvee_k \mathrm{x}_k$$

Example 1. *Let us consider the formula:* $5 : \mathrm{y} \leftarrow \mathrm{x}_1 \wedge \neg \mathrm{x}_2$ *which would be converted to DNF as:* $5 : (\mathrm{y} \wedge \mathrm{x}_1 \wedge \neg \mathrm{x}_2) \vee (\neg \mathrm{x}_1) \vee (\mathrm{x}_2)$*, and then to an RBM with the energy function:* $E = -5h_1(y + x_1 - x_2 - 1.5) - 5h_2 \max(-x_1 + 0.5, x_2 - 0.5)$. *Table 2 shows the equivalence between the RBM and the formula.*

x_1	x_2	y	s_φ	E_{rank}
0	0	0	1	-2.5
0	0	1	1	-2.5
0	1	0	1	-2.5
0	1	1	1	-2.5
1	0	0	0	0
1	0	1	1	-2.5
1	1	0	1	-2.5
1	1	1	1	-2.5

Table 2: Energy of the RBM and truth values of the formula $5 : \mathrm{y} \leftarrow \mathrm{x}_1 \wedge \neg \mathrm{x}_2$

where s_φ is (unweighted) truth values of the formula. This indicates the equivalence between the RBM and the formula as:

$$s_\varphi(\mathrm{x}_1, \mathrm{x}_2, \mathrm{y}) = -\frac{1}{2.5} E_{rank}(x_1, x_2, y) + 0$$

3.2 First-order Logic

A first order logic formula can also be converted into a set of Confidence rules. First, let us consider a predicate: $P(x, y)$ which one can present in a propositional DNF as:

$$\bigvee_{a,b|P(a,b)=true} \mathrm{p}_{x=a} \wedge \mathrm{p}_{y=b} \wedge \mathrm{p}_P$$

where (a, b) are the models of $P(x, y)$; $\mathrm{p}_{x=a}$ and $\mathrm{p}_{y=b}$ are the propositions that are $true$ if $x = a$ and $y = b$ respectively, otherwise they are $false$; p_P is the proposition indicating if

[1]The proof is similar as in [Tran, 2017]

[2]$0 < \epsilon < 1$

the value of $P(a, b)$. Each conjunct in this DNF then can be represented as a Confidence rule.

Now, let us consider a first-order formula which we are also able to present in a set of Confidence rules. For example, a clause φ as:

$$\forall_{x,y,z} \text{son}(x, z) \leftarrow \text{brother}(x, y) \wedge \text{has\_father}(y, z)$$

can be converted into:

$$\bigvee_{a,b,c|\varphi=true} (\text{p}_{x=a} \wedge \text{p}_{y=b} \wedge \text{p}_{z=c} \wedge \text{p}_{\text{son}} \wedge \text{p}_{\text{brother}} \wedge \text{p}_{\text{has\_father}}) \vee (\neg\text{p}_{x=a} \vee \neg\text{p}_{y=b} \vee \neg\text{p}_{z=c} \vee \neg\text{p}_{\text{brother}} \vee \neg\text{p}_{\text{has\_father}})$$

If one want to encode the background knowledge through its samples, for example:

$$\text{son}(James, Andrew) \leftarrow \text{brother}(James, Jen) \wedge \text{has\_father}(Jen, Andrew)$$

then we can convert it into confidence rules:

$c : \text{h}_1 \leftrightarrow \text{james} \wedge \text{jen} \wedge \text{andrew} \wedge \textbf{son} \wedge \textbf{brother} \wedge \textbf{has\_father}$

$c : \text{h}_p \leftrightarrow \neg\text{james} \vee \neg\text{jen} \vee \neg\text{andrew} \vee \neg\textbf{brother} \vee \neg\textbf{has\_father}$

In practice, in many cases we are only interested in inferring the predicates therefore we can omit $\neg$james, $\neg$jen, $\neg$andrew from the second rule.

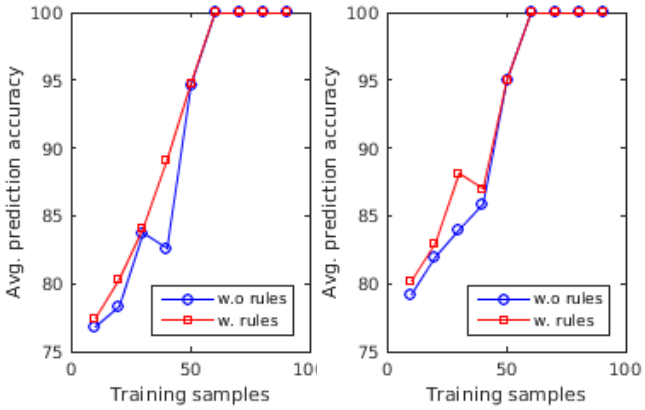
4 Empirical Evaluation

In this section we apply the encoding approaches discussed in the previous section to integrate knowledge into unsupervised networks.

4.1 DNA promoter

The DNA promoter dataset consist of a background theory with 14 logical *if-then* rules [Towell and Shavlik, 1994]. The rules includes four symbols $contact$, $minus_{10}$, $minus_{35}$, $conformation$ which are not observed in the data. This is suitable for hierarchical models as shown in previous works [Towell and Shavlik, 1994; Tran and Garcez, 2016]. In this experiment we group the rules using *hypothetical syllogism* to eliminate the unseen symbols. After that we encode the rules in an RBM following the theory in Section 3.1. The confidence values are selected empirically.

We test the normal RBMs and the RBMs with encoded rule using leave-one-out method, both achieve 100% accuracy. In order to evaluate the effectiveness of our approach we partition the data into nine different training-test sets with number of training samples are 10, 20, 30, 40, 50, 60, 70, 80, 90. All experiments are repeated 50 times and the average results are reported in Figure 1. We perform the prediction using both Gibbs sampling and conditional distribution $P(y|\mathbf{x})$. In particular, Figure 1a shows the prediction results using 1-step Gibbs sampling where the input is fixed to infer the hidden states and then to infer the label unit. In Figure 1b the results show the prediction accuracy achieved by inferring the label unit from the conditional distribution. As we can see, in both cases the integrated RBMs perform better than the normal RBMs on small training sets with number of training sample is less than 60. With larger training sets, the rules are no longer advantageous to the learning since the training samples are adequate to generalise the model to achieve 100% accuracy.



(a) Infer with Gibbs sampling (b) Infer with $P(y|\mathbf{x})$

Figure 1: RBMs without rules v.s RBMs with rules

4.2 Kinship

In this experiment, we use the approach discussed in Section 3.2 for relation discovery and reasoning tasks with Kinship dataset [Hinton, 1986; Sutskever and Hinton, 2008]. Here given a set of examples about relations we perform two type of reasoning: (1) what is relation between two people, i.e $?(x, y)$; and (2) a person has a relation R with whom, i.e. $R(x, ?)$. Previous approaches are using matrices/tensors to represents the relations making it difficult to explain [Sutskever and Hinton, 2008]. In this work, since only predicates are given, we encode the examples for the predicates in an unsupervised network as shown earlier in Section 3.2. This constructs the left part of the integrated model in Figure 2. In the right part, we model the unknown clauses by using a set of hidden units. The idea here is that by inferring the predicates using the encoded rules in the left part we can capture the relationship information, from which the desired relation is inferred by reconstruction of such relationship in the right part. In this experiment, we use auto-encoder [Bengio, 2009] for the right part for the purpose of efficient learning. The whole process is described in Algorithm 1.

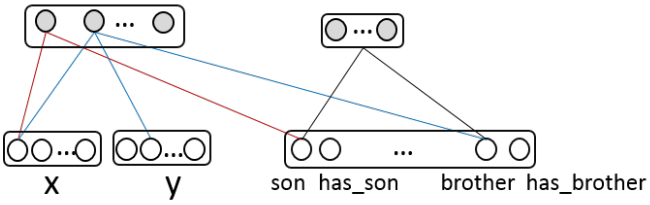


Figure 2: Encoding Kinship examples

Let us take an example where one wants to find the relation between two people $R(Marco, Pierro) =?$. First, we use the other examples to construct an integrated model as in Figure 2. After that, we train the auto-encoder in the right part using unsupervised learning algorithm, then we extract the relation features from Marco and Pierro as shown in Ta-

Marco, Pierro	1.000: has_father	Marco has father who is Pierro
Marco	0.191:has_wife	Marco has wife
	0.191:husband	Marco is a husband of someone
	0.191:has_mother	Marco has mother
	0.191:father	Marco is a father of someone
	0.191:has_daughter	Marco has daughter
	0.191:son	Marco has is a son of someone
	0.191:has_son	Marco has son
	0.191:has_sister	Marco has sister
	0.191:brother	Marco is a brother of someone
Pierro	0.191: has_wife	Pierro has wife
	0.191: husband	Pierro is a husband of someone
	0.191: father	Pierro is a father of someone
	0.191: has_daughter	Pierro has daughter
Reconstruct (possible relations)	0.008:wife 0.008:husband 0.001:mother 0.016:father 0.045:daughter **0.252:son** 0.005:sister 0.013:brother 0.001:aunt 0.002:uncle 0.003:niece 0.002:nephew	

Table 3: Relation features

ble 3. In that table we also show the reconstructed scores for all the relations where son is the correct one.

Algorithm 1

Data: Examples: E, Question: R(a,b)
Result: R
Encode all examples in an RBM: N
Initialise $\mathcal{D} = \emptyset$
for *each example* $R(x, y)$ *in E* **do**
  f = INFER(N,x,y)
  Add f to $\mathcal{D}$
end
Train an Auto-Encoder (AE) on $\mathcal{D}$
f = INFER(N,a,b)
Reconstruct $\hat{f}$ using AE
Return R = $\arg\max_R(\hat{f}_R)$ ▷ Return the unseen relation where the reconstruction feature have the highest value.

1: **function** INFER(N, a, b)
2: Infer direct relation between a,b
3: Infer possible relations of a: R(a,*)
4: Infer possible relations of b: R(*,b)
5: f = concatenation of all relations
6: Return f
7: **end function**

We test the model on answering the question $R(x, y) =?$ using leave-one-out validation which achieve 100% accuracy. We also use the integrated model to reason about whom one has a relation with. This question may have more than one answer, for example $\mathrm{son}(Athur, ?)$ can be either $Cristopher$ or $Penelope$. We randomly select 10 examples for testing and repeat it for 5 times. If the designate answers are in the top relations with highest reconstructed features then we consider this as correct, otherwise we set it as wrong. The average error of this test is 0%. However, when we increase the number of test samples to 20 and 30 the average errors grow to 2.8% and 6.8% respectively. For comparison, the matrices based approach such as [Sutskever and Hinton, 2008] achieves 0.4%,1.2%,2.0% average error rates for 10, 20, 30 test examples respectively. Note that, such approach and many others [Socher *et al.*, 2013] model each relation by a matrix/tensor while in this experiment we share the parameters across all relations. Also, the others use discriminative learning while we use unsupervised learning. The purpose of this is to exemplify the encoding technique we proposed earlier in this paper. Improvement can be achieved if similar methods are employed.

5 Conclusions

The paper shows how to integrate symbolic knowledge into unsupervised neural networks. This work bases on the theoretical finding that any propositional formula can be represented in RBMs [Tran, 2017]. We show that converting background knowledge in the form of *if-then* rules to Confidence rules for encoding is efficient. In the experiments, we evaluate our approaches for DNA promoter prediction and relationship reasoning to show the validity of the approach.

References

[Avila Garcez and Zaverucha, 1999] Artur S. Avila Garcez and Gerson Zaverucha. The connectionist inductive learning and logic programming system. *Applied Intelligence*, 11(1):5977, July 1999.

[Bengio, 2009] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[França *et al.*, 2014] Manoel V. M. França, Gerson Zaverucha, and Artur S. d'AvilaGarcez. Fast relational learning using bottom clause propositionalization with artificial neural networks. *Machine Learning*, 94(1):81–104, 2014.

[Garcez *et al.*, 2008] Artur S. d'Avila Garcez, Lus C. Lamb, and Dov M. Gabbay. *Neural-Symbolic Cognitive Reasoning*. Springer Publishing Company, Incorporated, 2008.

[Hinton, 1986] Geoffrey E. Hinton. Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 1–12. Hillsdale, NJ: Erlbaum, 1986.

[Penning *et al.*, 2011] Leo de Penning, Artur S. d'Avila Garcez, Lus C. Lamb, and John-Jules Ch Meyer. A neural-symbolic cognitive agent for online learning and reasoning. In *IJCAI*, pages 1653–1658, 2011.

[Pinkas, 1995] Gadi Pinkas. Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. *Artificial Intelligence*, 77(2):203–247, September 1995.

[Smolensky, 1995] Paul Smolensky. Constituent structure and explanation in an integrated connectionist/symbolic cognitive architecture. In C. Mcdonald, editor, *Connectionism: Debates on Psychological Explanation*, pages 221–290. Blackwell, Cambridge, 1995.

[Socher *et al.*, 2013] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, NIPS'13, pages 926–934, USA, 2013. Curran Associates Inc.

[Sutskever and Hinton, 2008] Ilya Sutskever and Geoffrey E Hinton. Using matrices to model symbolic relationship. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1593–1600. Curran Associates, Inc., 2008.

[Towell and Shavlik, 1994] Geoffrey G. Towell and Jude W. Shavlik. Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1-2):119–165, 1994.

[Tran and d'Avila Garcez, 2013] Son N. Tran and Artur d'Avila Garcez. Knowledge extraction from deep belief networks for images. In *IJCAI-2013 Workshop on Neural-Symbolic Learning and Reasoning*, 2013.

[Tran and Garcez, 2016] Son Tran and Artur Garcez. Deep logic networks: Inserting and extracting knowledge from deep belief networks. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99):1–13, 2016.

[Tran, 2017] Son N. Tran. Propositional knowledge representation in restricted boltzmann machines. https://arxiv.org/abs/1705.10899, 2017.

[Valiant, 2006] Leslie G. Valiant. Knowledge infusion. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pages 1546–1551, 2006.