Knowledge Extraction from DBNs for Images

Son N. Tran and Artur d'Avila Garcez

Department of Computer Science
City University London

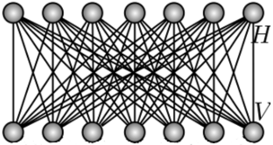
Contents

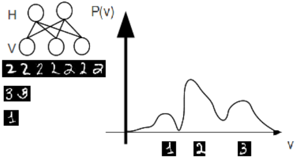
Motivation

- Deep networks have shown good performance in image, audio, video and multimodal learning
- We would like to know why by studying the role of symbolic reasoning in DBNs. In particular, we would like to find out:
    - How knowledge is represented in deep architectures
    - Relations between Deep Networks and a hierarchy of rules
    - How knowledge can be transferred to analogous domains

Restricted Boltzmann Machine

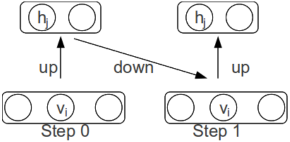
- Two-layer symmetric connectionist system [Smolensky, 1986]
- Represents a joint distribution $P(V,H)$
- Given training data, learning by Contrastive Divergence (CD) seeks to maximize $P(V) = \sum_h P(V,H)$
- It can be used to approximate the data distribution given new data (rather like an associative memory)





Restricted Boltzmann Machine (details)

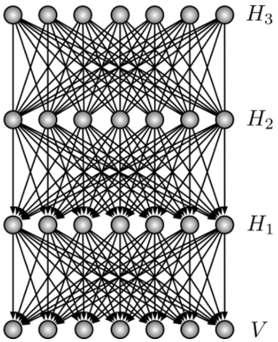
- Generative model that can be trained to maximize log-likelihood $\mathcal{L}(\theta|\mathcal{D}) = \log(\prod_{x \in D} P(v = x))$, where θ is set of parameters (weights and biases) and $\mathcal{D}$ is a training set of size n
- $P(v = x) = \frac{1}{Z} \sum_h \exp(-E(v,h))$, where E is the energy of the network model
- This log-likelihood is intractable since it is not easy to compute partition function $Z = \sum_{v,h} \exp(-E(v,h))$
- But it can be approximated efficiently using CD [Hinton, 2002]; $\Delta w_{ij} = \frac{1}{n} \sum_n (v_i h_j)_{step0} - \frac{1}{n} \sum_n (v_i h_j)_{step1}$



Deep Belief Networks

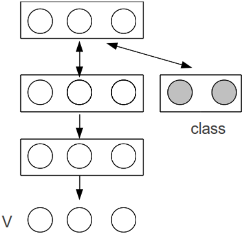
Deep Belief Networks [Hinton et al., 2006]

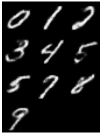
- Stack of RBMs
- Greedily learns each pair of layers bottom-up with CD
- Fine tuning option 1: Split weight matrix into up and down weights (wake-sleep algorithm)
- Fine tuning option 2: Use as feedforward neural network and update weights using BP



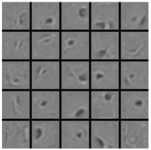
Deep Belief Networks (example)

- The lower level layer is expected to capture low-level features
- Higher level layers combine features to learn progressively more abstract concepts
- Label can be attached at the top RBM for classification




(class layer - 0 to 9)


(second hidden layer - shapes)


(first hidden layer - edges)

Rule Extraction from RBMs: related work

- [Pinkas, 1995]: rule extraction from symmetric networks using *penalty logic*; proved equivalence between conjunctive normal form and energy functions
- [Penning et al., 2011]: extraction of temporal logic rules from RTRBMs using sampling; extracts rules of the form $hypothesis_t \leftrightarrow belief_1 \wedge, ..., \wedge belief_n \wedge hypothesis_{t-1}$
- [Son Tran and Garcez, 2012]: rule extraction using confidence-value similar to penalty logic but maintaining implicational form; extraction without sampling

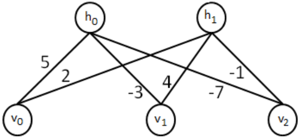
Rule Extraction from RBMs (cont.)

- Both penalty [Pinkas, 1995] and confidence-value [Penning et al., 2011, Son Tran and Garcez, 2012] represent the reliability of a rule
- Inference with penalty logic is to optimize a ranking function, thus similar to weighted-SAT
- In [Penning et al., 2011], confidence-value is not used for inference, whilst confidence-values extracted by our method can be used for hierarchical inference

Our method: partial-model extraction

- Extracts rules $c_j : h_j \leftrightarrow \bigwedge_{w_{pj}>0} v_p \wedge \bigwedge_{w_{nj}<0} \neg v_n$
- $c_j = \sum_{w_{ij}>0} w_{ij} - \sum_{w_{ij}<0} w_{ij}$ (i.e. sum of absolute values of weights); also applies to visible units v_i
- Example:

$$15 : h_0 \leftrightarrow v_1 \wedge \neg v_2 \wedge \neg v_3$$
$$7 : h_1 \leftrightarrow v_1 \wedge v_2 \wedge \neg v_3$$



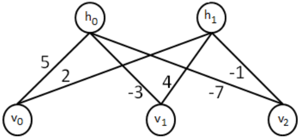
- These rules are called *partial-model* because they capture partially the architecture and behavior of the network

Our method: complete-model extraction

- Confidence-vector: $\mathbf{h}_j = [|w_{1j}|, |w_{2j}|, ...]$
- Complete rules: $c_j : h_j \overset{\mathbf{h}_j}{\leftrightarrow} \bigwedge_{w_{ij}>0} v_i \wedge \bigwedge_{w_{ij}<0} \neg v_i$

$$15 : h_0 \overset{[5,3,7]}{\leftrightarrow} v_1 \wedge \neg v_2 \wedge \neg v_3$$

$$7 : h_1 \overset{[2,4,1]}{\leftrightarrow} v_1 \wedge v_2 \wedge \neg v_3$$



Inference

- Inference

$$c : h \overset{[w_1, w_2, \ldots, w_n]}{\leftrightarrow} b_1 \wedge \neg b_2 \wedge \cdots \wedge b_n$$

$$\alpha_1 : b_1, \alpha_2 : \neg b_2, \ldots, \alpha_n : b_n$$

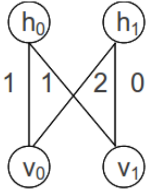
$$\overline{\hspace{10cm}}$$

$$c_h : h \text{ where } c_h = f(c \times (w_1\alpha_1 - w_2\alpha_2 + \ldots w_n\alpha_n))$$

- $\alpha_i : b_i$ means that b_i is believed to hold with confidence α_i
- f is a monotonically nondecreasing function. We use either sign-based ($f(x) = 1$ if $x > 0$ otherwise $f(x) = 0$) or logistic function; f normalizes the confidence value to [0,1].
- c is the confidence of the rule; c_h is the confidence of h
- In partial-models, $w_i = \frac{c}{n}$.
- The inference is deterministic (but stochastic inference is possible)

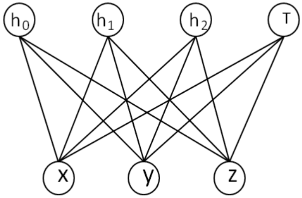
Partial-model vs. Complete-model

Partial model: equalizes weights, can help generalization, good if weights are similar; information loss, otherwise
Complete model: very much like the network, but difficult to visualize rules; baseline
Example:



$2 : h_0 \leftrightarrow v_1 \wedge v_2$
$2 : h_1 \leftrightarrow v_1 \wedge v_2$
Both rules have the same confidence-value but the first is a better match to h_0 than the second is to h_1

XOR problem



X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

$$W = \begin{pmatrix} -10.0600 & 3.9304 & -9.8485 \\ 9.6408 & 9.5271 & -7.5398 \\ 5.0645 & -9.9315 & -9.8054 \end{pmatrix}$$

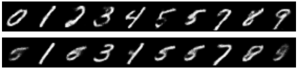
$$\text{visB} = [4.5196 \quad -4.3642 \quad 4.5371]^\top$$

$25 : h_0 \leftrightarrow \neg x \wedge y \wedge z$
$23 : h_1 \leftrightarrow x \wedge y \wedge \neg z$
$27 : h_2 \leftrightarrow \neg x \wedge \neg y \wedge \neg z$
$13 : \top \leftrightarrow x \wedge \neg y \wedge z$

If z is ground-truth then the combined, normalized rule is:
$0.999 : z \leftarrow (x \wedge \neg y) \vee (\neg x \wedge y)$

Logical inference vs. Stochastic inference

- DBN with 748-500-500-2000 nodes (+10 label nodes) was trained on MNIST handwritten digits dataset
- Figure shows the result of downward inference from the labels using the network (top) and using its complete model with a sigmoid function f for logical inference (bottom)
- To reconstruct the images from the labels using the network, we run up-down inference several times; to reconstruct the images from the rules, Gibbs sampling is not used, and we go downwards once through the rules



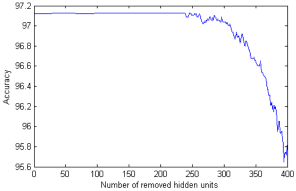
System pruning

One can use rule extraction to prune the network by removing hidden units corresponding to rules with low confidence-value

- Reconstruction of images from pruned RBM

(a) 500 units (b) 382 units (c) 212 units (d) 145 units

- Classification by SVM using features from pruned RBMs



Transfer Learning

Problems in Machine Learning:

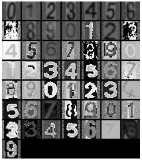
- Data in problem domain is limited
- Data in problem domain is difficult to label
- Prior knowledge in problem domain is hard to obtain

Solution: Learn the knowledge from unlabelled data from related domains which are largely available and transfer the knowledge to the problem domain.

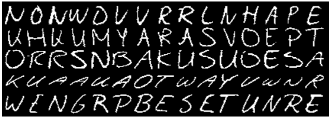
Transferring Knowledge to Learn

Source domain: MNIST handwritten digits
Target domains: ICDAR (digit recognition), TiCC (writer recognition)

(a) MNIST dataset



(b) ICDAR dataset



(c) TiCC dataset

Experimental Results

Source:Target	SVM	RBM	PM Transfer	CM Transfer
MNIST : ICDAR	68.50 38.14	65.50 50.00	66.50 50.51	66.50 51.55
MNIST : TiCC	72.94 73.44	78.82 80.23	79.41 83.05	81.18 80.79

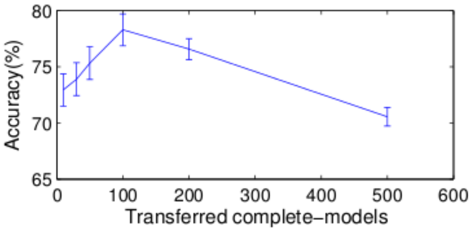


Figure : TiCC average accuracy vs. size of transferred knowledge

Conclusion and Future Work

- New knowledge extraction method for Deep Networks
- Initial results on image datasets and transfer learning
- Future work: More results and analysis of rules' applicability to transfer learning (domain dependent?)
- Extraction of partial-models that approximate the network well (midway between complete and current partial model)
- Best way of generalizing and revising rules after transferring them (knowledge insertion to close the learning cycle)

References I

Hinton, G. E. (2002).
Training products of experts by minimizing contrastive divergence.
Neural Comput., 14(8):1771–1800.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006).
A fast learning algorithm for deep belief nets.
Neural Comput., 18(7):1527–1554.

Penning, L. d., Garcez, A. S. d., Lamb, L. C., and Meyer, J.-J. C. (2011).
A neural-symbolic cognitive agent for online learning and reasoning.
In *IJCAI*, pages 1653–1658.

References II

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

Smolensky, P. (1986).
Information processing in dynamical systems: Foundations of harmony theory.
In *Rumelhart, D. E. and McClelland, J. L., editors, Parallel Distributed Processing: Volume 1: Foundations*, pages 194–281. MIT Press, Cambridge.

Son Tran and Garcez, A. (2012).
ICML logic extraction from deep belief networks.
In *ICML 2012 Representation Learning Workshop*, Edinburgh.