Discussion of “The Neural Autoregressive Distribution Estimator”

Yoshua Bengio
DIRO, Université de Montréal
Montréal, QC, Canada
bengioy@iro.umontreal.ca

1 Introduction

This is a discussion of Larochelle and Murray (2011).

The Restricted Boltzmann Machine (Smolensky, 1986; Hinton *et al.*, 2006) has inspired much research in recent years, in particular as a building block for deep architectures (see Bengio (2009) for a review). The Restricted Boltzmann Machine (RBM) is an undirected graphical model with latent variables, exact inference, rather simple sampling procedures (block Gibbs), and several successful learning algorithms based on approximations of the log-likelihood gradient. However, when it comes to actually computing the distribution or density function, it is intractable, except when either the number of inputs or latent variables is very small (about 25 binary hidden units with current computers and about an hour of computing, on MNIST).

With applications in mind where the exact likelihood would be useful (e.g. when combining the model with other graphical models, or in order to perform exact likelihood comparisons between different models), Larochelle and Murray have introduced a new probabilistic model that is inspired by the RBM but whose likelihood can be computed very cheaply.

2 The NADE

Larochelle and Murray called this model the Neural Autoregressive Distribution Estimator (NADE) because it actually is a fully visible directed graphical model without any latent variable and with a left-to-right connectivity, i.e., each variable gets to be predicted by the previous ones in some order (like in autoregressive statistical models):

$$P(x) = \prod_i P(x_i|x_{<i})$$

where $x_{<i}$ denotes $(x_1, \ldots x_{i-1})$. However, NADE's *parametrization* is obtained by performing a *single step* of a mean-field recursion to approximate $P(x_i|x_{<i})$ in an RBM, yielding a very simple parametric form:

$$P(x_i = 1|x_{<i}) = \text{sigm}(b_i + W_{i,.}\text{sigm}(c + W_{.,<i}x_{<i}))$$

with $W_{.,<i}$ denoting the submatrix of W with columns 1 to $i-1$, c a vector of length H, and W having D rows and H columns. Note this is a neural network with a very special *shared weights structure*. The hidden units are organized in groups $h_i = \text{sigm}(c + W_{.,<i}x_{<i})$ (one per variable), each input variable x_i is connected to all groups h_j for $j > i$, the same weight matrix (or submatrices of it) is used across the different groups, both in the hidden layer (to compute h_i) and in the output layer (to compute the probability prediction for each binary variable). Because the likelihood is tractable, so is its gradient, so the model is trained by stochastic gradient ascent on the log-likelihood. Note that computations can be greatly speeded-up by noting that most of the work to compute h_i has already been done when computing h_{i-1}, and can be re-used.

Apart from the weight sharing, this is the same architecture that we already proposed (Bengio and Bengio, 2000) eleven years ago, a model that is a non-linear generalization of the *logistic autoregressive Bayesian network* (Frey, 1998) proposed just before (where $P(x_i = 1|x_{<i})$ is just a logistic regression), called FVSBN here (Fully Visible Sigmoid Belief Network).

3 Main Results

NADE is compared with several other models for which the likelihood can be computed tractably, including small RBMs (with 23 hidden units), FVSBNs, mixtures of multivariate Bernoulli's, and two other recently proposed models (Larochelle *et al.*, 2010) for which the likelihood is tractable (to some extent), the RBM multinomial (an RBM with a small number of groups of multinomials as hidden units) and the RBForest (similar to the RBM multinomial, but with each multinomial structured into a tree of hidden units). The comparisons are performed on 8 datasets (with several of the likelihood values obtained from

Appearing in Proceedings of the 14^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15. Copyright 2011 by the authors.

earlier papers). The results are striking, showing a very strong advantage of NADE (or of FVSBN, in two cases, but where NADE also performs well).

What is impressive with those results is that the improvements in terms of log-likelihood are not just statistically significant, they are plainly large. NADE is also compared with large RBMs, using Annealed Importance Sampling to estimate the log-likelihood on a binarized version of the MNIST. It was found that NADE actually yields similar likelihoods, suggesting that *tractability was achieved at almost no cost in generalization performance*. Finally, note that NADE samples appeared good, and furthermore can be obtained through an exact left-to-right sampling (not requiring convergence of an MCMC).

4 Discussion

The model without weight constraints, i.e., from Bengio and Bengio (2000), was not included in the comparison. However, on two of the datasets (DNA and Mushroom) one can compare with Bengio and Bengio (2000), and indeed NADE is doing better[1], suggesting that the RBM-inspired constraint on the parametrization of the neural network indeed buys something in terms of generalization performance. The weight sharing also greatly reduces the number of free parameters, of course, as well as the actual computation (because of the shared computation between successive h_i's). An approach explored in Bengio and Bengio (2000) to reduce capacity was to prune the directed belief network structure, by keeping a connection (in the graphical model) between the pairs of variables for which a statistical dependency test was above a threshold. On the DNA dataset, the pruned but otherwise unconstrained network slightly outperforms NADE (but then the same strategy could be used to possibly improve NADE as well).

An important discussion element raised by reviewers is the fact that NADE is dependent on a particular *ordering of the variables*. Although any ordering yields a valid model, some orderings could be more or less favorable. To address this issue, the authors ran tests in which a dozen separate models were trained, each with a different randomly chosen order. They found that the variation induced by the order was an order of magnitude smaller than the uncertainty due to the finite test set, which is very reassuring.

One should note that part of the explanation for the better performance of NADE with respect to the small RBMs, RBM multinomial, and RBForest, could be due simply to the smaller capacity enforced upon the latter to achieve tractability. The improvement with respect to the mixture of Bernoulli's might be due to other reasons, though, such as the use of a distributed representation in the latent variables (an RBM is just a mixture model with a huge number of components but very strong constraints on their parametrization, so the number of parameters remains exponentially small compared to the number of mixture components).

One intriguing question is that as one goes from h_1 to h_D, the hidden units will saturate more and more (since we are summing the contributions from more and more of the variables). This is unusual for a neural network, and one may wonder if it would make optimization inefficient. On the other hand, it does make sense that h_i become more saturated as one considers more evidence (more variables). Since we are in the realm of "inspiration" from the RBM, one could easily try variants in which the saturation effect could be reduced (e.g., by defining $h_i = \text{sigm}(\alpha_i(c + W_{\cdot,<i} x_{<i}))$ with α_i a free parameter initialized to $1/i$).

To summarize NADE is a very easy to implement and train model for joint distributions, yielding a tractable distribution function. It should be easy to extend it to continuous variables or a mix of discrete and continuous ones (e.g., either taking inspiration from a corresponding RBM parametrization, such as the Gaussian RBM, or simply parametrizing the output densities appropriately based on the hidden units). In this context, it is not clear if the constraint that the input-to-hidden weights are the transpose of the hidden-to-output weights is strictly necessary.

References

Bengio, S. and Bengio, Y. (2000). Taking on the curse of dimensionality in joint distributions using neural networks. *IEEE Trans. Neural Networks*, **11**(3), 550–557.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, **2**(1), 1–127. Also published as a book. Now Publishers, 2009.

Frey, B. J. (1998). *Graphical models for machine learning and digital communication*. MIT Press.

Hinton, G. E., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**, 1527–1554.

Larochelle, H. and Murray, I. (2011). The Neural Autoregressive Distribution Estimator. In *AISTATS'2011*.

Larochelle, H., Bengio, Y., and Turian, J. (2010). Tractable multivariate binary density estimation and the restricted Boltzmann forest. *Neural Computation*, **22**(9), 2285–2307.

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge.

[1] On mushroom the comparison is more difficult because of differences in data splits and input representation.