Department of Computer Science
University of Toronto
http://learning.cs.toronto.edu

6 King's College Rd, Toronto
M5S 3G4, Canada
fax: +1 416 978 1455

October 25, 2006

UTML TR 2006–003

Learning Multilevel Distributed Representations for High-Dimensional Sequences

Ilya Sutskever and Geoffrey Hinton
Department of Computer Science, University of Toronto

Learning Multilevel Distributed Representations for High-Dimensional Sequences

Ilya Sutskever and Geoffrey Hinton
Department of Computer Science
University of Toronto
Toronto, Ontario M5S 3G4

October 25, 2006

We describe a family of non-linear sequence models that is substantially more powerful than hidden Markov models or linear dynamical systems. Our models have simple approximate inference and learning procedures that work well in practice. Multilevel representations of sequential data can be learned one hidden layer at a time, and adding extra hidden layers improves the resulting generative models. The models can be trained with very high-dimensional, very non-linear data such as raw pixel sequences. Their performance is demonstrated using synthetic video sequences of two balls bouncing in a box.

Introduction

Many different models have been proposed for high-dimensional sequential data such as video sequences or the sequences of coefficient vectors that are used to characterize speech. Models that use latent variables to propagate information through time can be divided into two classes: *tractable* models for which there is an efficient procedure for inferring the exact posterior distribution over the latent variables and *intractable* models for which there is no exact and efficient inference procedure. Tractable models such as linear dynamical systems and hidden Markov models have been widely applied but they are very limited in the types of structure that they can model. To make inference tractable when there is componential hidden state, it is necessary to use linear models with Gaussian noise so that the posterior distribution over the latent variables is Gaussian. Hidden Markov Models combine non-linearity with tractable inference by using a posterior that is a discrete distribution over a fixed number of mutually exclusive alternatives, but the mutual exclusion makes them exponentially inefficient at dealing with componential structure: to allow the history of a sequence to impose N bits of constraint on the future of the sequence, an HMM requires at least 2^N nodes. Inference remains tractable in mixtures of linear dynamical systems [4], but if we want to switch from one linear dynamical system to another *during* a sequence, exact inference becomes intractable [4]. Inference is also tractable in products of hidden Markov models [2].[1]

[1] Products of linear dynamical systems are linear dynamical systems and mixtures of hidden Markov models are hidden Markov models.

To overcome the limitations of the tractable models, many different schemes have been proposed for performing approximate inference [10, 5]. Boyen and Koller [1] investigated the properties of a class of approximate inference schemes in which the true posterior density in the latent space is approximated by a simpler "assumed" density such as a mixture of a modest number of Gaussians [9]. At each time step, the model dynamics and/or the likelihood term coming from the next observation causes the inferred posterior density to become more complicated, but the inferred posterior is then approximated by a simpler distribution that lies in the space of assumed distributions. Boyen and Koller showed that the stochastic dynamics attenuates the approximation error created by projecting into the assumed density space and that this attenuation typically prevents the approximation error from diverging.

In this paper we describe a family of generative models for sequential data that can capture many of the regularities that cannot be modeled efficiently by hidden Markov models or linear dynamical systems. The family has an *undirected* model for the interactions between the hidden and visible (*i.e.* observed) variables. This ensures that the contribution of the likelihood term to the posterior over the hidden variables is exactly factorial which greatly facilitates inference. This model family has some attractive properties:

- It has componential hidden state which means it has an exponentially large state space[2].
- It has non-linear dynamics and it can make multimodal predictions.
- There is a very simple on-line filtering procedure which provides a good approximation to the true conditional distribution over the hidden variables given the data observed so far.
- Even though maximum likelihood learning is intractable, there is a simple and efficient learning algorithm that finds good values for the parameters.
- There is a simple way to learn multiple layers of hidden variables and this can greatly improve the overall generative model.

By using approximations for both inference and learning, we obtain a family of models that is much more powerful than those that are normally used for modeling sequential data.

The empirical question is whether our approximations are good enough to allow us to exploit its power for modeling real sequences in which each time-frame is high-dimensional and the past has high-bandwidth non-linear effects on the future.

The generative model

The Restricted Boltzmann Machine

We begin by reviewing the Restricted Boltzmann Machine (RBM) [6, 14]. It has a simple, exact inference procedure for the hidden variables and an efficient approximate learning algorithm

[2] The number of parameters is only quadratic, so there are strong limitations on how the exponentially large state space can be used, but for sequences in which there are several independent things going on at once, it is easy to use different subsets of the hidden units to model different components of the sequential structure.

for the parameters. These two properties make the RBM very useful as an observation model for sequential data. When an RBM is modified to be conditioned on previous hidden and/or visible states, we get a temporal RBM (TRBM) which can be used to model sequences.

The RBM defines a distribution over $(V, H) \in \{0,1\}^{N_V} \times \{0,1\}^{N_H}$ via the equation

$$P(V, H) = \exp\left(V'WH + \mathbf{a}'V + \mathbf{b}'H\right)/Z, \tag{1}$$

where W are the connection weights of the RBM, $\mathbf{a}, \mathbf{b}$ are the biases for V and H, and the variables N_V and N_H are the number of dimensions of V and H. We use the notation V' for V transpose, since the standard notation creates confusion in later sections. The variable V stands for visible and H for hidden, and we use $P(V, H)$ to mean either a distribution or a single probability, depending on the context. We use the more cumbersome notation $P(V = \mathbf{v}, H = \mathbf{h})$ to clarify the ambiguous cases.

The joint distribution $P(V, H)$ is from the exponential family and VH' is the sufficient statistics, subject to the constraint that (V, H) is a binary vector. The conditional distributions $P(V|H)$ and $P(H|V)$ are factorial and are given by

$$\begin{aligned} P\left(H_j = 1|V\right) &= \sigma\left(b_j + W'_{:,j}V\right) && (2)\\ P\left(V_i = 1|H\right) &= \sigma\left(a_i + W_{i,:}H\right), && (3) \end{aligned}$$

where $\sigma(z) = (1 + \exp(-z))^{-1}$ is the logistic function and $W_{:,j}, W_{i,:}$ are the jth column and the ith row of W.

The derivative of the log likelihood $\mathcal{L}$ with respect to the parameters is given by the very simple equations

$$\begin{aligned} \Delta W_{ij} &\propto \langle V_i H_j \rangle_{P(H|V)\tilde{P}(V)} - \langle V_i H_j \rangle_{P(V,H)} && (4)\\ \Delta a_i &\propto \langle V_i \rangle_{\tilde{P}(V)} - \langle V_i \rangle_{P(V)} && (5)\\ \Delta b_j &\propto \langle H_j \rangle_{P(H|V)\tilde{P}(V)} - \langle H_j \rangle_{P(H)}, && (6) \end{aligned}$$

where $\tilde{P}(V)$ denotes empirical the data distribution which is the average of the datapoints in the training set. Maximum likelihood estimation is difficult due to the need to compute expectations with respect to the model's distribution, $\langle \cdot \rangle_{P(V,H)}$. An obvious way to compute these expectations is to use alternating Gibbs sampling. Starting from an arbitrary initial distribution, we alternate between updating all of the hidden units in parallel using Eq. 2 and updating all of the visible units in parallel using Eq. 3. After a sufficient number of iterations, this method gives unbiased samples from the distribution $P(V, H)$ [12]. It is generally much better than brute-force calculation of the expectation which takes exponential time in the size of the RBM, but it is still slow in practice, since the Markov chain needs to be run for *each* iteration of the learning algorithm.

Fortunately, there is another parameter estimation method which we call Contrastive Divergence (CD) because it follows the approximate gradient of an objective function that is the difference of two Kullback-Liebler divergences [6]. CD is much more efficient than maximum likelihood learning and it works well in practice – RBMs learned with CD produce high-quality generative models [3]. The weight updates for CD are given by

$$\begin{aligned} \Delta W_{ij} &\propto \langle V_i H_j \rangle_{P(H|V)\tilde{P}(V)} - \langle V_i H_j \rangle_{P_1(V,H)} && (7)\\ \Delta a_i &\propto \langle V_i \rangle_{\tilde{P}(V)} - \langle V_i \rangle_{P_1(V,H)} && (8)\\ \Delta b_j &\propto \langle H_j \rangle_{P(H|V)\tilde{P}(V)} - \langle H_j \rangle_{P_1(V,H)}, && (9) \end{aligned}$$

where a sample from the distribution P_1 is obtained by running Gibbs sampling for 1 full step, having initialized (V, H) by $\tilde{P}(V)P(H|V)$. More specifically, we sample the visibles V from $\tilde{P}(V)$, the hiddens H from $P(H|V)$, and then sample the visibles and then the hiddens once more to get a sample from P_1. For CD to work, V *must* be initialized with $\tilde{P}(V)$.

There is a simple intuitive way to understand what CD learning is doing. Instead of sampling from the model's distribution, we allow the model to slightly distort the data distribution towards a distribution that the model prefers. Then we lower the free energy of the data and raise the free energy of the distorted data to prevent the model distorting the data in that direction in future. This can be viewed as a way of making the gradient of the free energy *w. r. t.* the distribution be zero at the data distribution.

Even though RBM's define a distribution over $\{0, 1\}^{N_V}$, they can sometimes be used to model $[0, 1]$ valued variables by treating intermediate values as probabilities. For the MNIST images of handwritten digits, for example, the normalized pixel intensities are mostly very close to 0 or 1 and it works well to treat the intermediate values as probabilities. Contrastive divergence learning works well when the we update the visible variables in Eq. 3 to have the real values produced by the logistic without using random sampling. In Eq. 2, we still use random sampling to obtain binary stochastic values for H, but on the RHS we simply use the real values of V which amounts to using a mean field approximation. In the simple videos described later in the paper, we used the mean-field approximation for the pixels.

Treating intermediate values as probabilities and using the mean-field approximation in Eq. 2 does not work well for most real-valued images, such as images of faces, because it cannot assign a sharply peaked probability distribution to an intermediate pixel intensity. For these images we can replace Bernoulli-distributed binary visible units by Gaussian-distributed real-valued ones [8].

The Temporal Restricted Boltzmann Machine for sequence modeling

Figure 1 shows an RBM that has been been augmented by adding directed connections from previous states of the visible and hidden units. We call this a Temporal Restricted Boltzmann Machine (TRBM). The TRBM defines a joint distribution over (V_t, H_t) that is conditional on earlier hidden and visible states. The effect of these earlier states is to dynamically adjust the effective biases of the visible and hidden units at time t:

$$\begin{aligned} P(V_t, H_t | V_{t-m}^{t-1}, H_{t-m}^{t-1}) = \exp(&V_t' C_0 H_t + \mathbf{b}' H_t + \mathbf{a}' V_t + \\ &V_{t-1}' C_1 H_t + V_{t-2}' C_2 H_t + \cdots + V_{t-m}' C_m H_t + \\ &H_{t-1}' B_1 H_t + H_{t-2}' B_2 H_t + \cdots + H_{t-m}' B_m H_t + \\ &V_t' A_1 V_{t-1} + V_t' A_2 V_{t-2} + \cdots + V_t' A_m V_{t-m})/Z, \end{aligned} \tag{10}$$

where V_t, H_t denote the state of the variables at time t and the notation V_{t-m}^{t-1} stands for $V_{t-1}, \ldots, V_{t-m}$, and likewise, H_{t-m}^{t-1}. Z depends on the states of $V_{t-m}^{t-1}, H_{t-m}^{t-1}$, as well as on the weight matrices $\{A_j\}_{j \le m}, \{B_j\}_{j \le m}, \{C_j\}_{j \le m}$ and the usual biases $\mathbf{a}, \mathbf{b}$. That is, we have a standard RBM with C_0 as its weight matrix, but the bias for H_t is $\mathbf{b} + B_1 H_{t-1} + \cdots + B_m H_{t-m} + C_1 V_{t-1} + \cdots + C_m V_{t-m}$ and the bias for V_t is $\mathbf{a} + A_1 V_{t-1} + \cdots + A_m V_{t-m}$, both of which depend on the states of the variables in the previous time steps. Whenever the TRBM needs to use a value of V_τ or H_τ where τ is less than 1, we use learned "initial" values that depend on τ to replace the product of the output of a unit and the weight on the directed connection.

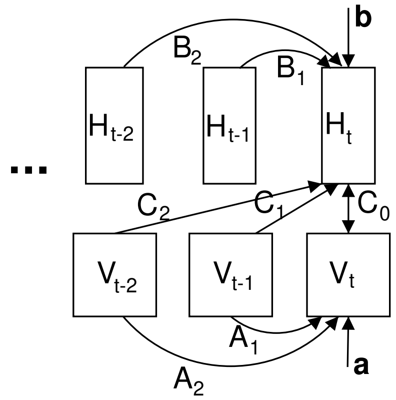

Figure 1: The TRBM

We model the probability of a whole sequence using a product of the distributions defined by a separate TRBM for each time step, with all of the TRBM's sharing the same parameters:

$$P(V_1^T, H_1^T) = \prod_{t=1}^{T} P(V_t, H_t | V_{t-m}^{t-1}, H_{t-m}^{t-1}) \tag{11}$$

There are three reasons we chose the TRBM as the building block for our sequence model. The first is that given V_1^t and H_1^{t-1} the distribution over H_t is factorial as long as the future variables V_{t+1}^T, H_{t+1}^T are unknown. This helps to ensure that a factorial approximation to the filtering distribution ($P(H_t|V_1^t)$) is reasonably accurate.

The second reason is that parameter estimation for an individual TRBM can be done efficiently using CD if we know the previous hidden and visible states, so by using the filtering distribution to approximate the posterior distribution over the hidden variables we can reduce the problem of modeling whole sequences to the problem of modeling the distribution of the current time frame of data given the previous data and the previous hidden states computed by the filtering distribution.

The third reason is that the TRBM model can easily be extended to include additional hidden layers and by adding more hidden layers we get a better representation and a better generative model.

Approximate Filtering

Our model is designed to make it easy to approximate the filtering distribution $P(H_t|V_1^t)$. Let $P_{\text{approx}}(H_{ti} = 1|V_1^t)$ be the probability that the i^{th} hidden unit is on in the factorial approximation to the filtering distribution. For each time t we maintain a vector $\mathbf{p}_t \in [0,1]^{N_H}$ such that $\mathbf{p}_{ti} = P_{\text{approx}}(H_{ti} = 1|V_1^t)$. We show how to compute $\mathbf{p}_t$, which clearly shows how to immediately obtain P_{approx}.

We derive our factorial approximation from the following observation. Suppose that V_1^t and H_1^{t-1} are known with certainty. In that case, the filtering distribution is factorial and is given by

$$P(H_{ti} = 1|V_1^t, H_1^{t-1}) = \sigma(C_0 V_t + B_1 H_{t-1} + \cdots + B_m H_{t-m} + C_1 V_{t-1} + \cdots + C_m V_{t-m} + b_i), \tag{12}$$

which is simply the sigmoid of all the inputs from the previous time frames to unit H_{ti}.

In the general case, we assume that V_1^t is given by the data with certainty but H_1^{t-1} is unknown and its uncertainty is represented by a factorial distribution P_{approx} (and $\mathbf{p}$). We use the mean-field equations [13] to compute $\mathbf{p}_t$ from $\mathbf{p}_1^{t-1}$ and V_1^t. The resulting equation is very similar to equation 12, except that we replace the values of the variables H_t with their probabilities $\mathbf{p}_t$, thus getting the equation

$$\mathbf{p}_{ti} = \sigma(C_0 V_t + B_1 \mathbf{p}_{t-1} + \cdots + B_m \mathbf{p}_{t-m} + C_1 V_{t-1} + \cdots + C_m V_{t-m} + b_i). \tag{13}$$

Learning

To allow online learning, we ignore the effect of future data on the inferred distribution over H_t and use the approximate filtering distribution as an approximate posterior (*i.e.* we do not do smoothing). Consider the following standard lower bound to the log likelihood [11]:

$$\log P(V_1^T) \geq \left\langle \log P\left(V_1^T, H_1^T\right)\right\rangle_{P_{\text{approx}}} + \mathbb{H}\left(P_{\text{approx}}\right), \tag{14}$$

where $\mathbb{H}$ is the entropy of a distribution, and $P_{\text{approx}}(H_1^T|V_1^T)$ is the approximate filtering distribution. We would like to maximize this lower bound with respect to P and P_{approx}. Maximizing this lower bound with respect to P amounts precisely to learning each TRBM separately using the factorial hidden distribution provided by P_{approx}, but as a result of this maximization with respect to P, P_{approx} changes as well, and can possibly reduce the value of the bound. The fact that the learning works in practice suggests that this ignored effect is not too serious.

Learning a TRBM when the hidden states are known is simple. It is just an RBM with dynamic biases which can be learned in the same way as normal biases.

In the equation below we write the weight update for a single TRBM. In our sequence model there are T such TRBMs, and the sum of their weight updates constitutes the full weight update. To simplify the notation we assume that there is only one training sequence in which case the weight update for time step t is

$$\Delta(C_n)_{ij} \propto \langle (V_{t-n})_i (H_t)_j \rangle_{Q_1} - \langle (V_{t-n})_i (H_t)_j \rangle_{Q_2^t} \tag{15}$$

$$\Delta(B_n)_{ij} \propto \langle (H_{t-n})_i (H_t)_j \rangle_{Q_1} - \langle (H_{t-n})_i (H_t)_j \rangle_{Q_2^t} \tag{16}$$

$$\Delta(A_n)_{ij} \propto \langle (V_{t-n})_i (V_t)_j \rangle_{Q_1} - \langle (V_{t-n})_i (V_t)_j \rangle_{Q_2^t}, \tag{17}$$

The distribution Q_1 is the filtering distribution $P_{\text{approx}}(H|V)$ and distribution Q_2^t is identical to Q_1 for frames $1, \ldots, t-1$. For timestep t it is the TRBM distribution over (V_t, H_t) conditioned on the previous states H_1^{t-1} and V_1^{t-1}, averaged over by filtering distribution, hence the superscript t. Note that even though the values of H_1^{t-1} are uncertain and are averaged over, in practice we substitute the value of each coordinate of H_1^{t-1} by $\mathbf{p}_1^{t-1}$, the vector of probabilities of each coordinate being 1 under the filtering distribution Q_1 of V_1^T. This makes the biases to the TRBM deterministic and eases learning. In practice also we cannot use the TRBM distribution, so we use a CD update, in which $Q_2^t(V_t, H_t|V_1^{t-1}, H_1^{t-1})$ is replaced by the distribution obtained from running Gibbs sampling in the TRBM at time t for one step starting at V_t, exactly as for an RBM. We assumed that there was only one training case in the above description, but actually it is sampled from the training set, so the gradients are averaged by the empirical data distribution.

Experiments with a single layer model

To demonstrate that our learning procedure works we used it to learn synthetic video sequences composed of 20×20 pixel time-frames of two balls bouncing in a box. The first row in figure 2 shows a sample from the training data. A movie can be viewed at
www.cs.utoronto.ca/∼ilya/aistats2006_filter/index.html

In the pixel space, the dynamics are highly non-linear. Even if we could extract the positions and velocities of the centers of both balls, the dynamics would be highly non-linear when the balls bounce off the walls or off each other. Also, the underlying coordinates are related to the pixel intensities in a very non-linear way. For all these reasons, modeling the raw sequence of pixel intensities is a challenging task which is made even more difficult if the model class cannot handle componential structure efficiently. An HMM, for example, would need about 10^4 hidden states to distinguish 10 positions and velocities of one ball on each axis, and 10^8 states for both balls.

We used several different TRBM models that had 400 visible units, 300 hidden units, and direct access to the hidden and visible states for the 4 previous time steps (*i. e.* $m = 4$). The full TRBM has 3 kinds of connections[3]. In addition to trying the full TRBM we also tried leaving out each set of connections in turn. We call these special cases TRBM-VV, TRBM-HH, and TRBM-VH where the last part of the name indicates which connections are omitted. TRBM-VV, for example, has no visible-to-visible connections. Despite its name, TRBM-VH retains the undirected connections between the current instantiations of V and H.

The TRBM-HH model is an interesting special case because the lack of hidden-to-hidden connections makes exact inference possible. This model is particularly well suited for hierarchical learning, as will be seen in future sections.

We trained each model using 100,000 training sequences of $T = 128$ frames. The weights were updated at the end of each sequence, with a learning rate of $0.05/T$ and momentum of 0.9. The learning signal for the connections between the visible variables was reduced by a factor of 100, since without doing so the learning procedure settles into poor local minima. Any parameter setting with a small enough learning rate and a momentum of 0.9 works well whenever the number of learning iterations is sufficient. All four variations of the TRBM learned quite good generative models that could continue an initial segment of a video (see the URL for examples of sequences generated by these models). The models could also be used for online denoising of sequences by performing approximate filtering and then reconstructing the visible state from the approximate filtering distribution. Figure 2 shows a typical image sequence and the same sequence corrupted by noise. The noise is correlated in both time and space which makes denoising much more difficult. All four variations of the TRBM denoise the sequence quite well, except for frames 4-7 where the noise is too severe. Figure 2 shows the

[3]It would also be possible to have connections from previous hidden states to the current visible units, but the model is complicated enough already.

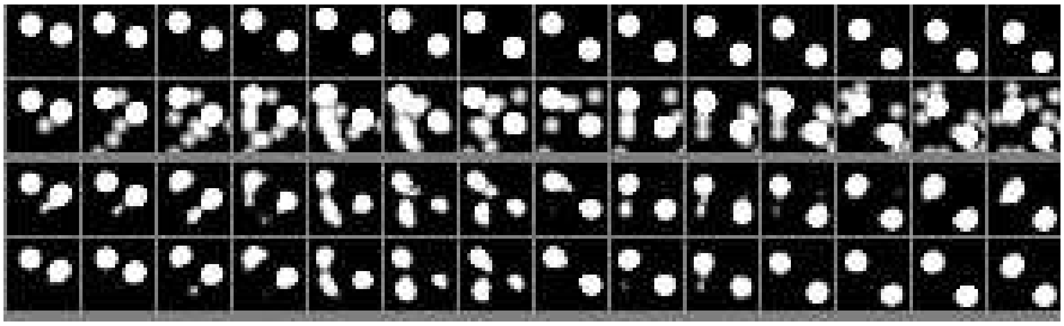

Figure 2: Top row: An image sequence. Second row: The same sequence corrupted by noise that is highly correlated in space and time. Third row: Denoising by a TRBM-VV using a single hidden layer. Bottom row: Denoising by a TRBM-VV with two hidden layers.

denoised sequence produced by the TRBM-VV which must use the hidden states to combine information across frames. When an extra hidden layer is added to any of the TRBM's, there is a noticeable improvement in the denoising, as well as in the generation. To denoise with two hidden layers we first compute the approximate filtering distribution for the second hidden layer and then reconstruct each frame of the data from the second hidden layer.

Our models denoise much better than a simple RBM which cannot make use of previous frames. They are not as good as an autoregressive model that has been trained to predict the clean image from the four previous noisy ones, but our model is not trained with noise so it can denoise without requiring training data that contains both the noisy and the noisy-free sequence.

The biggest disadvantage of our models is that they currently take several days to train and even then the training is not complete. We also tried training a full TRBM with 400 hidden units for two weeks after which it had a model that generated extremely well (see the URL).

Multilayer Models

Adding more hidden layers to an RBM

In this section we describe how to improve an RBM by introducing additional hidden layers, and creating a hierarchical representation of the data, as described in [7]. Let $\tilde{P}(V)$ denote the data distribution and $P(V, H)$ denote the joint distribution defined by the RBM. The idea is to get another RBM, $Q(H, U)$, which has H as its visible and U as its hidden variables, to learn to model the aggregated posterior distribution, $\tilde{Q}(H)$, of the first RBM

$$\tilde{Q}(H) = \sum_V P(H|V)\tilde{P}(V). \tag{18}$$

Provided $Q(H)$ models $\tilde{Q}(H)$ better than $P(H)$ does, it can be shown that there is an augmented model $M_{PQ}(V, H, U)$ which will be defined shortly that is a better model of the original data than the $P(V, H)$ defined by the first RBM alone [7]. $M_{PQ}(V, H, U)$ uses the undirected connections learned by Q between H and U, but it uses *directed* connections from H to V. It thus inherits $P(V|H)$ from the first RBM but discards $P(H|V)$ and hence $P(H)$ from its generative model. Data can be generated from the augmented model by sampling from $Q(H,U)$ (by running a Markov chain), discarding the value of U, and then sampling from $P(V|H)$ (in a single step) to obtain V. This implements $M_{PQ}(V) = \sum_H P(V|H)Q(H)$. Provided $N_U \geq N_V$, Q can be initialized by using the parameters from P to ensure that the two RBM's define the same distribution over H. Starting from this initialization, optimization then ensures that $Q(H)$ models $\tilde{Q}(H)$ better than $P(H)$ does.

The second RBM, $Q(H,U)$, learns by fitting the distribution $\tilde{Q}(H)$, which is not equivalent to maximizing $\log M_{PQ}(V)$. Nevertheless, it can be proved [7] that this learning procedure maximizes a variational lower bound on $\log M_{PQ}(V)$. Even though $M_{PQ}(V,H,U)$ has discarded $P(H|V)$ from its generative model, we can still approximate the posterior distribution $M_{PQ}(H|V)$ by $P(H|V)$. Applying the standard variational bound, we get

$$\mathcal{L} \geq \langle \log Q(H)P(V|H) \rangle_{P(H|V)} + \mathbb{H}(P(H|V)). \tag{19}$$

where $\mathbb{H}(P(H|V))$ is the entropy of $P(H|V)$. Maximizing this lower bound with respect to the parameters of Q whilst holding the parameters of P and the approximating posterior $P(H|V)$ fixed is precisely equivalent to fitting Q to $\tilde{Q}(H)$. Note that the details of Q are unimportant; Q could be any kind of a model, and not just an RBM. The main advantage of using another RBM is that it is possible to initialize $Q(H)$ to be equal to $P(H)$, so the variational bound starts as an equality and any improvement in the bound guarantees that $M_{PQ}(V)$ is a better model of the data than $P(V)$.

This procedure can be repeated recursively as many times as desired, creating very deep hierarchical representations. For example, a third RBM, $R(U,X)$, can be used to model the aggregated approximate posterior over U obtained by

$$\tilde{R}(U) = \sum_V \sum_H Q(U|H)P(H|V)\tilde{P}(V) \tag{20}$$

Provided $R(U)$ is initialized to be the same as $Q(U)$, $M_{QR}(H)$ will be a better model of $\tilde{Q}(H)$ than $Q(H)$, but this does not mean that $M_{PQR}(V)$ is necessarily a better model of $\tilde{P}(V)$ than $M_{PQ}(V)$. It does mean, however, that learning R will improve the variational bound obtained by using $P(H|V)$ and $Q(U|H)$ to approximate the posterior distribution $M_{PQR}(U|V)$.

There have been many previous attempts to train multilayer models in a greedy, layer-by-layer way. These attempts have not met with much success because they generally use a directed model of the form

$$P(V) = \sum_H P(V|H)P(H) \tag{21}$$

where $P(H)$ is a factorial prior over H that is defined by a separate set of parameters. The use of a factorial prior encourages the learning to make the aggregated posterior over H as factorial as possible and this leaves little structure to be modeled by the next hidden layer. In an RBM, the posterior over H is factorial for each possible value of V, but both the implicitly defined prior over H and the aggregated posterior over H are typically very far from factorial, thus leaving plenty of structure for the next layer to model.

Multilayer TRBMs

We straightforwardly generalize the idea to our sequence model. First we learn a TRBM, and then learn another TRBM that learns to model the hidden states of the first TRBM, which is precisely analogous to the way the RBM was augmented.

Recall that we denote by V_1^T the set of all the visible time frames and by H_1^T the set of all the hidden time frames. Denote by $P(V_1^T, H_1^T)$ the distribution defined by the first TRBM.

$P(H_1^T|V_1^T)$ is not factorial, so we approximate it by the filtering distribution, $P_{\text{approx}}(H_1^T|V_1^T)$. Let $Q(H_1^T, U_1^T)$ be a TRBM that we use to learn the aggregated approximate filtering distribution $\sum_{V_1^T} P_{\text{approx}}(H_1^T|V_1^T)\tilde{P}(V_1^T)$, where U_1^T is the sequence of the hidden variables of the TRBM Q. The approximate posterior of U_1^T, $Q_{\text{approx}}(U_1^T|H_1^T)P_{\text{approx}}(H_1^T|V_1^T)$ allows U_1^T to represent higher-level features that can be computed on-line, since neither P_{approx} nor Q_{approx} make use of future frames. The resulting augmented generative model, $M_{PQ}(V_1^T)$, is one where we first sample from $Q(H_1^T)$, then from $P(V_1^T|H_1^T)$, so, as with the RBMs, $M_{PQ}(V_1^T) = \sum_{H_1^T} P(V_1^T|H_1^T)Q(H_1^T)$. If we can initialize Q so that $Q(H_1^T) = P(H_1^T)$, then the augmented model is identical to the P and has the same likelihood. By making Q learn the distribution of the hidden states of P, which is $\sum_V P_{\text{approx}}(H_1^T|V_1^T)\tilde{P}(V_1^T)$, we maximize the lower bound

$$\mathcal{L} \geq \left\langle \log Q\left(H_1^T\right) P\left(V_1^T|H_1^T\right)\right\rangle_{P_{\text{approx}}} + \mathbb{H}\left(P_{\text{approx}}\right), \tag{22}$$

with respect to Q. This is very similar to the bound in equation 19, except for the use of the approximate posterior. At the beginning of the optimization this bound is strictly *less* than $\log P(V_1^T)$ even when $Q(H_1^T) = P(H_1^T)$, because an approximate posterior is used. It could, therefore, remain less than $\log P(V_1^T)$. This is not the case for RBMs, since the exact posterior $P(H|V)$ is easily computable, so the lower bound is equal to $\log P(V)$ at the beginning of the optimization.

Although our learning procedure maximizes a lower bound that is initially smaller than $\log P(V_1^T)$, it is very likely that by the end of learning the bound will exceed $\log P(V_1^T)$. In addition, since we use an approximate posterior during the learning of $P(V_1^T)$ (recall that inference is intractable in our TRBM model), we are performing approximate maximization of a lower bound on $\log P(V_1^T)$ (this is also equation 14):

$$\log P(V_1^T) \geq \left\langle \log P\left(H_1^T\right) P\left(V_1^T|H_1^T\right)\right\rangle_{P_{\text{approx}}} + \mathbb{H}\left(P_{\text{approx}}\right) \tag{23}$$

(the maximization is approximate in that we ignore the effect that changing the approximate posterior has on the bound), so by introducing Q the new lower bound of equation 22 will be equal to the bound in equation 23 if Q is properly initialized. Therefore, the lower bound in Eq. 22 will be greater than the lower bound in Eq. 23.

In order to initialize Q such that $Q(H_1^T) = P(H_1^T)$, it is necessary for Q to have directed connections between its visible variables (the variables H_1^T) so that $Q(H_1^T)$ can represent every distribution $P(H_1^T)$ can. For RBM's, learning one hidden layer at a time works well even if $Q(H)$ is not initialized to be equal to $P(H)$ [7], so in our experiments, we did not initialize $Q(H_1^T) = P(H_1^T)$.

We can also add further hidden layers in the same was as is done for RBM's and each time another layer is added we should get a better generative model.

Notice that for the model TRBM-HH, for which $P(H_1^T|V_1^T)$ is exactly factorial, the situation is significantly better. Not only does it have an exact learning procedure (if we ignore the approximations introduced by contrastive divergence), but its augmented model *always* has a greater likelihood since the lower bound (Eq. 23) is equal to the log likelihood if $Q(H_1^T) = P(H_1^T)$, because $P_{\text{approx}}(H_1^T|V_1^T) = P(H_1^T|V_1^T)$.

A drawback of all of our TRBM models is that $P(V_1^T|H_1^T)$ is not factorial because of the directed connections into V. This makes it intractable to generate unbiased samples from the augmented models, so further approximations must be used.

Results for multilevel models

We conducted experiments to determine whether adding an extra hidden layer improves the quality of generative models. For each of TRBM, TRBM-VV, TRBM-VH, TRBM-HH, we used the same type of TRBM with 400 hidden units (and 300 visibles) to learn the aggregated posterior distribution of the hidden units in the first-level model. The learning parameters of all these models were the same as those for the original TRBM's and training lasted for 100,000 updates. All of the generative models improved and they all became better at denoising (see figure for a typical denoising example, or the URL for many movies of denoising and generation).

Despite the improved performance, we cannot generate exactly from the improved multilevel models. Recall that to generate, we first need to use $Q(H_1^T, U_1^T)$ to sample the activities of H_1^T and then we need to sample from $P(V_1^T|H_1^T)$, which is the distribution over *sequences* of visible frames given a sequence of hidden frames. This distribution is intractable for the same reasons inference is intractable in our models, and we approximate it in a similar spirit.

The conditional distribution $P(V_1^T|H_1^T)$ is intractable to sample from, but if H_1^t and V_1^{t-1} are known and the rest of the variables H_{t+1}^T are not given, then generating V_t is easy. Indeed, all the explaining away effects disappear and the distribution over V_t is factorial. We therefore use an "on-line" approximation to $P(V_1^T|H_1^T)$, one where we go over t from 1 to T, sampling V_t given H_1^t and V_1^{t-1} *ignoring* H_{t+1}^T.

Conclusions and discussion

In this paper we introduced a family of sequence models that can learn good generative models in an online fashion. We demonstrated that the learning works despite relying on several approximations:

- The filtering distribution is approximate because it uses a mean-field approximation to model the effects of the previous filtering distribution on the current one.
- There is no smoothing so the learning is using the filtering distribution to approximate the posterior. This means that it is ignoring the effect of changing parameters on the likelihood of the future observations.
- Even if the posterior was correct, the learning would be following the approximate gradient of the contrastive divergence instead of the exact gradient of the log likelihood.

We believe that there are two main reasons why the family of models that we have described will work much better than the more familiar family of directed models (*i. e.* dynamic Bayes nets) for modeling video sequences at the pixel level. The first is that our observation model leads to a factorial posterior over the hidden variables. The second is that we have an effective way to decompose the task of learning a model with many hidden layers into a series of one hidden layer tasks.

References

[1] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In G. Cooper and S. Moral, editors, *Proc. of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 33–42, San Francisco, 1998. Morgan Kaufmann.

[2] A. Brown and G. Hinton. Relative density nets: A new way to combine backpropagation with HMM's. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 1149–1156. MIT Press, 2001.

[3] M. Carreira-Perpinan and G. Hinton. On contrastive divergence learning. *Artificial Intelligence and Statistics, 2005*, 2005.

[4] Z. Ghahramani and G. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12(4):831–864, 2000.

[5] Z. Ghahramani and M. Jordan. Factorial hidden markov models. *Machine Learning*, 29(4):245–273, 1997.

[6] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1711–1800, 2002.

[7] G. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18, 2006.

[8] G. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, 2006.

[9] A. Ihler, J. Fisher, III, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self-calibration in sensor networks. In *Proceedings of the third international symposium on Information processing in sensor networks (IPSN-04)*, pages 225–233, New York, 2004. ACM Press.

[10] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of the 4th European Conference on Computer Vision*. Springer-Verlag, 1996.

[11] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37(2):183–233, 1999.

[12] R. Neal. Probabilistic inference using MCMC methods. Technical report, CRG-TR-93-1, Department of Computer Science, University of Toronto, Canada, 1993.

[13] C. Peterson and J. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1(5):995–1019, 1987.

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