Using an Effective Boltzmann Machine to Learn Context Dependencies of a Sequence

Matthew I Bellgard
Computer Science
Murdoch University
Murdoch, 6150, WA, Aust.
matthew@cs.murdoch.edu.au

Chi-Ping Tsang
Computer Science
The University of Western Australia
Nedlands, 6907, Aust
tsang@cs.uwa.edu.au

Abstract

It has been recognised that current Artificial Neural Network (ANN) systems that employ windowing techniques to learn context dependencies in sequences have many deficiencies. One variant of this type of system that attempts to overcome many of these deficiencies is the Effective Boltzmann Machine (EBM). The EBM which is based on the Boltzmann Machine (BM) has the ability to perform completion and to provide an energy measure for the solution. In this paper, we extend the EBM and show that the system has many desirable properties. This paper reports two major improvements to the EBM: First, whereas in the past, a BM is used to learn the local contexts of a sequence, we show that the EBM itself is an architecture suitable for learning local contexts. Our initial experiments show that it outperforms a window based BM. Second, we demonstrate that, with this new training scheme, a multilayer EBM can be constructed to extend the effective window size. This means that learning contextual dependencies is not limited by the window size. This is done by utilising the state information of the first EBM hidden layer to train that of the second EBM layer. The effect of this long range effective window is demonstrated by experiments.

1 INTRODUCTION

Capturing long-term contextual dependencies is an important area of research in Artificial Neural Networks. In the past, neural networks that use windowing (buffering) systems to learn contextual information have been criticised for their deficiencies [Elm90, Moz89]. One problem is that of choosing an appropriate window size. The window size must at least be large enough to account for the longest context which is generally unknown [Elm90, Moz91].

Simple recurrent networks for temporal sequence recognition are currently being demonstrated to possess many important features that are lacking in the explicit windowing approach [Elm90, Moz89]. These networks use time-delayed state information to reconstruct the context of a sequence and have been shown to exhibit decaying long range effects [Moz91]. However, these systems are not without their problems. For example, Mozer [Moz91] shows that learning fine-grain long-range temporal dependencies is limited. In addition, these systems are only able to learn either left-right or right-left rules of contextual sequences and not bidirectional ones.

This paper describes extensions to the Effective Boltzmann Machine (EBM) [TB90, BT92, BT94] which has already been shown to have many desirable properties. The EBM is able to complete arbitrary length contextual sequences from learned local context. A version of the EBM has been successfully applied to the synthesis of musical harmony for polyphonic music. Like the Boltzmann Machine (BM), the EBM is also a completion machine. The completion process is not directional in terms of the temporal/spatial nature of the sequences. That is, it can capture implicational constraints in both directions. Completions of partially specified sequences are produced in a non-deterministic manner.

In the past, a BM was initially trained to learn a set of local contexts (training stage), from which the EBM is constructed for the completion/evaluation stage. This is similar to learning using a window covering the range of local context. There are two main contributions from this paper. First, the EBM architecture is introduced during the learning stage. The main
advantage is that the weights are updated by the co-occurrence statistics of the sequence as a whole rather than individual local contexts. This will give more accurate statistics and consequently faster training than using the traditional BM. Furthermore, this leads to the second major improvement: that the state information in the hidden layer of the EBM can be used to train another EBM in the next layer. Hence an architecture of a hierarchy of EBMs is introduced. Using these improvements, we show that the effective window size can be lengthened.

The paper is structured as follows: section two describes the construction of the EBM and how it is used to perform completion/evaluation; section three introduces the learning procedure for a simple EBM; section four describes the hierarchy of the multi-layer EBM; section five describes some experiments demonstrating the EBM for learning (which is better than using a window-based BM) as well as showing the results of learning/completion using an hierarchical EBM; section six provides a discussion which is then followed by a conclusion of this research.

2 BACKGROUND

The EBM is an extension of the Boltzmann Machine [AK88] architecture. It is able to complete a partially specified contextual sequence from learned local context. A description of the EBM system can be found in references [TB90, BT92, BT94]. Hence only an overview will be given here. In the following sections we show how to learn the local contexts of sequences using a BM and then how to recognise new sequences by constructing an EBM.

2.1 Capturing Local Contexts in Sequences Using an EBM

In our earlier work, the EBM is only constructed for recognition purposes. During the training stage, we use a window-based BM to learn the local context for a fixed size window. These local contexts are binary patterns that constitute the training set. The standard BM learning algorithm [AHS85] is used to obtain a set of weights that characterises these binary patterns. This trained BM is used to construct an EBM.

The behaviour of a BM is governed by an energy function \( E \). The energy \( \bar{E} \) of the BM is defined to be the summation of the weighted connections of pairs of units that are activated. Unit updates are made in relation to this energy function. The energy of a particular unit (known as the energy gap) is determined by a difference in the global energy of the system when that unit is either "1" or "0". In conjunction with the Simulated Annealing algorithm [AK88] the unit is stochastically updated according to a probability acceptance criterion. These equations may found elsewhere. For instance, [AHS85, AK88].

2.2 Completions of an Arbitrary Length Sequence using an EBM

Given a BM which has learnt the local contexts of a sequence(s), an arbitrary length sequence \( S \) consisting of \( L \) number of events may be synthesised using an EBM. Similar to the BM, each event in \( S \) will be represented by a set of binary units equal to the number of bits required to encode the events. The units in the events of \( S \) may be clamped or unclamped. If they are unclamped, they are initialised randomly to "0" or "1", otherwise the units in an event are clamped to binary values such that the set of units represent an instantiated event.

An EBM may be constructed as shown schematically in Figure 1 to complete the sequence \( S \) (of length \( L = 5 \)). Assuming that the BM with a window size of \( M = 3 \), is used to learn the local contexts of a sequence(s) (training set), then this machine is replicated and placed over \( S \). There will be \( L - M + 1 \) copies of the BM. In Figure 1 each machine is enclosed by the dotted lines. The BM is replicated 3 times. Each \( B^1 \) has identical weights and an identical number of hidden units.

The EBM is constructed such that the units in the IO layers of the individual BMs are no longer independent but are constrained to have the same values as those in \( S \). The units in \( S \) represent the IO layer of the EBM. Because the individual machines are constrained, the entire system (not the individual BMs) is subjected to the annealing process to produce a solution (to synthesise a sequence). As may be seen in Figure 1, unit updates in \( S \) (as the system is annealed) take into account the information arriving from neighbouring contexts (from more than one
individual BM). For example the shaded IO unit in Figure 1 is contained in all three individual BMs. The whole sequence $S$ is completed at the same time. As the system is slowly relaxed to a solution,

$$\Delta E_{S_i}(\xi) = \sum_r \sum_j W_{ij} H_j^k(e^k)$$

where $$1 \leq j \leq N, 0 \leq i-r \leq L-M$$

$$E(\alpha, \xi) = -\sum_i \sum_j \sum_r W_{ij} H_j^k(e^k) S_i(\alpha)$$

$$P_{S_i}(\xi) = \frac{1}{1 + e^{-\frac{\Delta E_{S_i}(\xi)}{T}}}$$

Note that the energy gap for hidden units in the EBM is the same as that for a BM. To anneal this system we effectively extend the algorithm for a BM. For example, the unclamped units in $S$ are initialised with random activations. Then both the hidden units and the unclamped $S$ units are alternatively selected for simulation. When "thermal equilibrium" is achieved, the temperature is lowered. This method constructs an EBM such that completion can be performed for a sequence of arbitrary length.

3 LEARNING SEQUENCES USING THE EBM

While the scheme described in section 2 works fairly well, we understood that it could be further improved by using the EBM to learn the sequence directly instead of using the window-based BM. Learning in the EBM is similar to that of BM. In order to learn the contexts of a sequence, an EBM using a fixed sized BM is constructed over the sequence and is shown in Figure 1. There are two phases in the learning stage: phase+ and phase-. In phase+, the IO layer of the EBM is clamped to the value of the sequence. The hidden units for the fixed sized BM are then annealed and updated for each context sequentially. At the end of the annealing schedule, co-occurrence statistics are recorded. In phase-, all units in the system are unclamped and at the end of the annealing schedule, co-occurrence statistics are recorded. Note that the IO units in the EBM are updated (in phase-) using the energy gap equation described in section 2. This is the difference that distinguishes learning in the EBM from that of the window-based BM learning. Weights in the network are then updated using the difference between the co-occurrence statistics in the two phases. This process (known as a sweep) is
repeated many times until the network converges on a correct set of weights that learns the sequences. Of course, if more than one sequence is to be learnt by the EBM, an EBM is constructed for each individual sequence. The co-occurrence statistics are calculated for each sequence in turn and the weights are updated.

4 THE HIERARCHICAL EBM

4.1 Lack of long-term contextual dependencies with this approach

With this formalism (using the BM to learn local contexts of sequences and then using the EBM to complete sequences of arbitrary length), context dependencies larger than $2M - 1$ cannot be learned. As a simple example of this situation, consider two sequences “xabcx” and “habch”. Assume that each character in these two sequences represents an event. Whenever there is an “x” in the start (end) of the sequence, then there should be an “x” at the end (start) of the sequence. Similarly for the sequence “habch”. Note that the context, “abc” is common to both sequences. A window size $M = 3$ would produce the following local contexts: “xab”, “abc”, “bcx”, “hab”, “abc”, and “bch”.

A BM made up of three events (each event of a fixed number of binary units), and a set of hidden units could be trained to learn these contexts. An EBM could then be constructed with an IO layer consisting of 5 events (as shown in Figure 1). If the EBM was then required to complete the partial sequence “xabc?” (where “?” is the event that the EBM must complete), how would it complete it?

The EBM will complete this partial sequence in one of two ways: either “xabcx” or “xabch”. It will choose an “x” or an “h” non-deterministically - having no real bias to either completion. However, the system should complete the partial sequence “xabc?” to “xabcx” with a higher probability than completing it to “xabch”. Similarly, the system should complete the partial sequence, “habe?” to “habch” with a higher probability than completing “habech”. In addition, the system should similarly account for the partial sequences, “?abcx” and “?abch”.

4.2 The Hierarchical EBM

Learning with the EBM can be extended further to hierarchical networks. Because the hidden units for each machine in the EBM are explicitly articulated, it is then possible to treat the hidden layer as an IO layer and construct another EBM (for a given window size). This may be shown diagrammatically in Figure 2. The first EBM learns the sequence and then the states of the first hidden layer are clamped so that the second EBM can learn contextual dependencies of the hidden states. W2 represents the set of weights for the second EBM.

During completion, the hidden layer HIDDEN1 is shown in Figure 2 is updated from information coming from the weights to the IO layer as well as from weights from the second hidden layer. That is, the energy gap of a unit in the HIDDEN1 layer is the sum of the energy gap of the first EBM plus the energy gap of the second EBM. This process may be extended to multiple layers.

5 EXPERIMENTAL RESULTS

A number of experiments have been carried out to determine the performance of learning with the EBM. The first experiment compares the EBM with the standard BM and the second experiment demonstrates the potential of the hierarchical EBM.
Experiment 1: Comparing of the EBM to the BM

The experiment was set up as follows: The sequence, “ABCDEFGHJKLMNO”, (15 events) is to be learnt using a window size of \( M = 3 \). Each event is represented in 6 bits, where only 2 bits are set to “1” for each event. That is, “A” is represented by “110000”, “B” is represented by “101000”, “C” is represented by “100100”, and so on for each character in the sequence. The EBM and the BM were trained on the sequence. For the BM, there are thirteen patterns in the training set: “ABC”, “BCD”, “CDE”, etc. The EBM has only one pattern - the original sequence. The learning parameters were the same for both. That is, the starting (700) and final (5) temperature, the number of co-occurrence statistics (50), number of hidden units (7) and the temperature decay rate (0.99). The weights of two machines are recorded for different sweeps. To determine the effectiveness of learning, various partial inputs (all of length 15) were clamped to an EBM system (for completion). This system was then used to complete for ten trials using first the weights obtained from EBM learning and then the weights from BM learning. The results are shown in Table 1.

Table 1 Results showing that EBM learning is more effective than window-based BM learning

<table>
<thead>
<tr>
<th>No. of events unclamped (U)</th>
<th>Machine for learning</th>
<th>Correct completions from 10 trials after different number of sweeps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EBM</td>
<td>150</td>
</tr>
<tr>
<td>aUUuUg...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>aUUuUg...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>aUUuUg...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>aUUuUg...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>aUUuUg...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>...UuUuG...</td>
<td>EBM</td>
<td>4</td>
</tr>
<tr>
<td>BM</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

As it can be seen, the weights from the EBM (for learning) is able to complete the partial sequences in less sweeps than the conventional BM. After 150 sweeps EBM learning managed to complete the sequence correctly (4 times) whereas the BM still produced an incorrect sequence.

In fact the number of sweeps required for the BM to perform as well as the EBM was at least twice as many. This experiment was repeated and similar results were obtained.

Experiment 2: The hierarchical EBM

In this experiment, the Hierarchical EBM (HEBM) was used to learn the two sequences, “xabcx” and “habch” using a window size of 3. To calculate the co-occurrence statistics, each sequence was treated one at a time. The weights were updated after both sequences were presented. There were 5 hidden units in the first hidden layer and 5 hidden units in the second layer. The first EBM was trained for 700 sweeps and then the second EBM was trained for 700 sweeps using the hidden states of the first EBM as the IO layer. The start temperature was 2000, final temperature 10, and decay rate of 0.99. During completion, the HEBM completed the partial sequence “xabo?” to “xabcx”. Similarly, the partial sequence “?abcx” is completed to “xabcx”, “?abch” completed to “habch”, and “?abcx” is completed to “habch”. Table 2 shows the completions along with energies of the four sequences: “xabcx”, “xabcx”, “habch”, and “habcx”. Each sequence is clamped to the HEBM which is then annealed. The energies for each layer of HEBM is shown. Note that HEBM attempts to minimise energy. The combined energy of the two layers for the two sequences: “xabcx” and “habcx” is lower than that for the other two sequences.

Table 2 Results showing that HEBM is able to correctly learn the two sequences: “xabcx” and “habcx” using window size of 3.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;xabcx&quot;</td>
<td>-656 (1st Layer)</td>
</tr>
<tr>
<td></td>
<td>-504 (2nd Layer)</td>
</tr>
<tr>
<td>&quot;habch&quot;</td>
<td>-644 (1st Layer)</td>
</tr>
<tr>
<td></td>
<td>-443 (2nd Layer)</td>
</tr>
<tr>
<td>&quot;habc?&quot;</td>
<td>-688 (1st Layer)</td>
</tr>
<tr>
<td></td>
<td>-370 (2nd Layer)</td>
</tr>
</tbody>
</table>

6 DISCUSSION

From Experiment 1, it can be concluded that it takes fewer sweeps to learn a sequence with the EBM (for learning) than it takes with the window-based BM. In fact EBM learning took less than half the time than that of BM.
This is not a totally surprising result because in effect, the EBM may be viewed as a standard BM except that some of the weights are constrained to be equal. It is this constraining that enables the EBM to find the correct minima faster than the BM learning with windows. In a windowing system during learning, each individual context is clamped separately and the system annealed, thus conflicts may occur between some contexts. This would explain why it takes longer to average out the conflicts in a sequence.

The BM has to find a set of weights that learns all 13 patterns correctly. The EBM, on the otherhand, is required to learn just one pattern. The new update rule for the IO units in the EBM (described in section 2), assists in removing any conflicts that arise (during learning) in having to learn 13 separate patterns.

Experiment 2 demonstrates another important result in using the EBM for learning. It is possible to construct a hierarchical EBM which is able to learn the hidden states of the EBM above. This means that it is possible to learn longer-term dependencies than have been possible with current windowing systems.

7 CONCLUSIONS

From our initial experiments, we have demonstrated that it is possible to learn with the EBM and that it outperforms the window-based BM in the number of sweeps that are required to learn a sequence. We have also shown how to construct a hierarchical network of EBMs in order to learn longer-range dependencies from local context. More extensive experiments are currently being conducted to the learning of sequences. We are also investigating extensions to the architecture of the network and other properties associated with this type of system.

6 REFERENCES


