On Multi-layered Connectionist Models Adding Layers vs. Increasing Width

Chung-jen Ho
Courant Institute of Mathematical Sciences
New York University
251 Mercer Street
New York, New York 10012

Abstract

In this paper, we explore the computational potential and limitations of the multi-layered connectionist models [Minsky and Papert, 1968]. We found that the number of layers and the width are two crucial parameters for the multi-layered connectionist models. If each layer has the same size n and we increment the number of layers by 1, then the number of problems solved will increase $0(n^3)$ times. On the other hand, suppose the number of layers is equal to 2. If we increment the width by 1, then the number of problem solved will increase $<D(n^n)$ times, where n is the input size. Hence, we can extend a 2-layered connectionist model by adding layers or increasing width. Our conclusion is that increasing width is better than adding layers.

1 Introduction

For studying learning in the multi-layered connectionist models, It is important to understand the computational potential and limitations of the multi-layered connectionist models. The single layered connectionist models have limited computational ability. For example, there is no single layered connectionist machine which can compute the 'exclusive or' of two bits. But the 'exclusive or' problem can be solved by a 2-layer connectionist machine. However, the computational ability of 2-layer connectionist models depends on the width of the models. We define the *width* of a connectionist machine to be the maximum number of neurons in one layer of the connectionist machine. The number of layers and the width are two crucial parameters for the multi-layered connectionist models. In this paper, we examine two extreme cases of the multi-layered connectionist models. In the first case, we fix the width of the multi-layered connectionist models and see what happen when the number of layers is increased. In the second case, we fix the number of layers to be two and see what happen when the width is increased. Our result is that in the first case, if each layer has the same size n and we increment the number of layers by 1, then the number of problems solved will increase $0(n^n)$ times; in the second case, when we increment the width by 1, the number of problem solved will increase $0(n^n)$ times, where n is the input size.

2 Layered Connectionist models

A *layered connectionist machine* is a special case of connectionist models. It has t layers and one input layer (layer zero). The input (bottom) layer contains n input neurons. The last (top) layer contains m output neurons. Each of the remainder layers contains w neurons. For $i = 0, \ldots, t-1$, there are links connecting the neurons in layer i to the neurons in layer $(i+1)$; no other links exist. The following is a formal definition of layered connectionist machines.

Definition A layered connectionist machine is a 7-tuple (t, w, n, m, C, B, A), where

1. t is a positive integer defining the number of layers.
2. w is a positive integer defining the width of the model.
3. n is a positive integer defining the input size of the model.
4. m is a positive integer bounding the output size of the model.
5. C is a set of neurons. $C = L_0 \cup L_1 \cup \cdots \cup L_t$ where $L_0 = \{C_{01}, C_{02}, \ldots, C_{0n}\}$ is the set of input neurons, $L_t = \{C_{t1}, C_{t2}, \ldots, C_{tm}\}$ is the set of output neurons, and $L_i = \{C_{ij} : j = 1, \ldots, w\}$ is the set of neurons in layer i for $i = 1, \ldots, t-1$. We denote by $|L_i|$ the number of elements in L_i for $i = 0, 1, \ldots, t$. Clearly, $|L_0| = n$, $|L_t| = m$ and $|L_i| = w$ for $i = 1, \ldots, t-1$.
6. B is a set of real numbers. $B = \{b_{ij} : i = 1, \ldots, t;$ and $j = 1, \ldots, |L_i|\}$ where b_{ij} serves as the *threshold* of C_{ij}.
7. A is a set of matrices. $A = \{A_1, A_2, \ldots, A_t\}$ where $A_i = (a_{ijk})$ is a matrix of real numbers whose entry a_{ijk} represents the connection weight with which $C_{i-1,j}$ affects C_{ik} for $i = 1, \ldots, t$ and $j = 1, \ldots, |L_{i-1}|$ and $k = 1, \ldots, |L_i|$.

Each neuron C_{ij} has a binary value v_{ij}. Initially, the values $v_{01}, v_{02}, \ldots, v_{0n}$ are set to the input values. At time i, $i = 1, \ldots, t$, the values of neurons in the ith layer are determined by the following formula :

$$v_{ik} = \begin{cases} 1 & \text{if } \left(\sum_{j=1}^{|L_{i-1}|} a_{ijk} v_{i-1,j} > b_{ik} \right) \\ & \end{cases}$$

for $k = 1, 2, \ldots, |L_i|$. Clearly, the computation time for a layered connectionist machine with t layers is t.

Example 1 The following is an example of layered connectionist models with 3 input neurons, 2 output neurons, 4 layers and width 5:

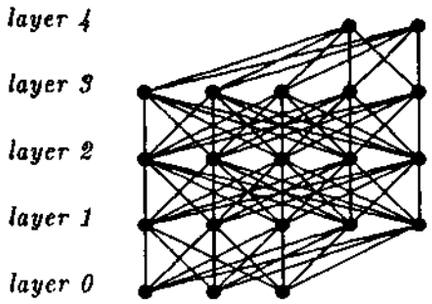

Figure 1 An example of layered connectionist models.

Let $H_{nm}(t)$ denote a class of problems solved by layered connectionist machines with n input neurons, m output neurons, t layers and width $w = \max\{n, m\}$. We denote by $|H_{nm}(t)|$ the number of elements in $H_{nm}(t)$.

We regards each problem in $H_{nm}(t)$ as a mapping from $\{0,1\}^n$ to $\{0,1\}^m$. Thus, each problem in $H_{nm}(t)$ can be represented as a function $f(x_1, x_2, \ldots, x_n) = y_1 \cdots y_m$ where $x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_m \in \{0,1\}$. For $j = 1, 2, \ldots, m$, each y_j is a Boolean function in $x_1, x_2, \ldots, x_n$. We write $y_j = f_j(x_1, x_2, \ldots, x_n)$. f can be specified with m Boolean functions, $f_1, f_2, \ldots, f_m$. There are 2^n possible combinations of binary input values. Therefore, $|H_{nm}(t)| \leq 2^{m2^n}$. As we know, the Boolean functions can be expressed in *sum of products form*. The product terms are AND terms and the sum denotes the ORing of product terms.

Example 2 $f(x_1, x_2, x_3) = \bar{x}_1 x_2 x_3 + x_1 \bar{x}_2 x_3 + x_1 x_2 \bar{x}_3$ is the sum of products form of f where $\bar{x}_j$ is the complement of x_j for $j = 1, 2, 3$.

Lemma 1 *Suppose the sum of products form of a Boolean function $f(x_1, x_2, \ldots, x_n)$ contains single product term. Then, $f \in H_{n1}(1)$. In other words, f can be computed by a single layered connectionist machine.*

Proof. In the product term, either x_i or $\bar{x}_i$ appears for $i = 1, \ldots, n$. Let $U = \{i : x_i$ appears in the product term$\}$ and $V = \{j : \bar{x}_j$ appears in the product term$\}$. Suppose the number of elements in U is p and the number of elements in V is q. Clearly, $U \cup V = \{1, 2, \ldots, n\}$ and $p + q = n$. We construct a single layered connectionist machine for computing $f(x_1, \ldots, x_n)$ as follows:

The set of neurons $C = L_0 \cup L_1$ where $L_0 = \{x_1, x_2, \ldots, x_n\}$ is the set of input neurons and $L_1 = \{y\}$ is the set of the output neuron. Let $b = p - 1$ be the threshold of y. For $i = 1, \ldots, n$, suppose a_i is the connection weight with which x_i affects y. We let $a_i = 1$ if $i \in U$; let $a_i = -1$ otherwise. Clearly, $y = 1$ only when all x_i $(i \in U)$ and $\bar{x}_j$ $(j \in V)$ are equal to 1. Hence, this single layered connectionist machine computes f. □

Example 3 $y = x_1 \bar{x}_2 x_3 \bar{x}_4 x_5$ can be computed by the following single layered connectionist machine:

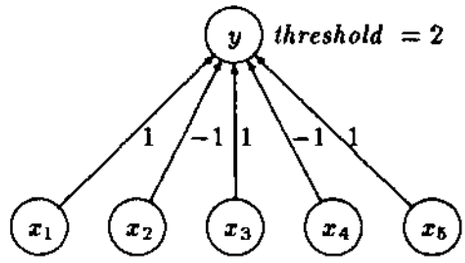

Figure 2 $y = x_1 \bar{x}_2 x_3 \bar{x}_4 x_5$.

We don't know if there exists an integer t_0 such that $|H_{nm}(t_0)| = 2^{m2^n}$. However, the following lemma shows that any boolean function $f(x_1, \ldots, x_n) = y_1 \cdots y_m$ can be computed by a layered connectionist machine $M = (2^n, w, n, m, C, B, A)$ if $w \geq n + m$.

Lemma 2 *Let $T_{nm}(t)$ denote a class of problems solved by layered connectionist machines with n input neurons, m output neurons, t layers and width $w = n + m$. Then, $|T_{nm}(2^n)| = 2^{m2^n}$.*

Proof. It is equivalent to show that for an arbitrary function $f : \{0,1\}^n \longmapsto \{0,1\}^m$ there is a layered connectionist models $M = (2^n, n + m, n, m, C, B, A)$ which computes f. Let $t_1 = 2^n$. f can be specified with m Boolean functions, $f_1, f_2, \ldots, f_m$. Each f_j $(j = 1, 2, \ldots, m)$ can be represented by a sum of products form. The sum of products forms can be embedded in a layered connectionist model $M = (2^n, n + m, n, m, C, B, A)$ as follows:

The input values pass to neurons $C_{i1}, C_{i2}, \ldots, C_{in}$ at time i for $i = 1, 2, \ldots, t_1 - 1$. This can be achieved by setting weight $a_{ikk} = 1$ and threshold $b_{ik} = 0$ for $k = 1, 2, \ldots, n$. $v_{i1}, v_{i2}, \ldots, v_{in}$ serve as inputs to $C_{i+1,n+j}$ for $j = 1, 2, \ldots, m$. The ith product term of f_j is corresponding to the single layered subnetwork with nodes $C_{i-1,1}, C_{i-1,2}, \ldots, C_{i-1,n}$ and $C_{i,n+j}$. For $k = 1, 2, \ldots, n$, we set the weight $a_{ik,n+j}$ (with which $C_{i-1,k}$ affects $C_{i,n+j}$) and threshold $b_{i,n+j}$ by using the method described in Lemma 1 and Example 3. Now, for fixed j $(1 \leq j \leq m)$, we need only to sum (OR) all of the values in $C_{1,n+j}, C_{2,n+j}, \ldots, C_{t_1,n+j}$. This can be done by setting $a_{i,n+j,n+j} = 2n$ for $i = 2, \ldots, t_1$. All the other weights and thresholds which haven't been mentioned are set to zero.

We have finished the description of the layered connectionist model M. The number of product terms in f_j $(j = 1, 2, \ldots, m)$ can not be greater than 2^n. Hence, M computes f. □

Example 4 The following layered connectionist model for computing $f = \bar{x}y + x\bar{y}$ is constructed by using the method described in Lemma 2. Note: all the thresholds are zero.

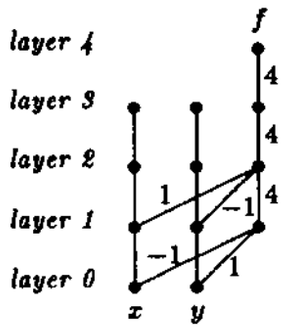

Figure 3 $f = \bar{x}y + x\bar{y}$.

Next, we will separate $H_{nm}(t+1)$ from $H_{nm}(t)$. Let

$$K = \max\{|H_{nm}(t)| : t \geq 1\}.$$

Then, clearly $|H_{nm}(t)| \leq K \leq 2^{m2^n}$ for $t \geq 1$.

Theorem 3 *For $t \geq 1$, if $|H_{nm}(t)| < K$, then $|H_{nm}(t)| < |H_{nm}(t+1)|$.*

Proof. If $|H_{nm}(t)| < K$ and $|H_{nm}(t)| = |H_{nm}(t+1)|$, we will show that $|H_{nm}(t)| = |H_{nm}(t+i)|$ for $i \geq 1$ which lead to a contradiction. Clearly, $H_{nm}(t) \subseteq H_{nm}(t+1)$. If $|H_{nm}(t)| = |H_{nm}(t+1)|$, then $H_{nm}(t) = H_{nm}(t+1)$. For showing $H_{nm}(t) = H_{nm}(t+i)$ for $i \geq 1$, we need only to show that $H_{nm}(t+1) = H_{nm}(t+2)$. Let us consider the case of $t+2$. Obviously, $H_{nm}(t+1) \subseteq H_{nm}(t+2)$ and $|H_{nm}(t+1)| \leq |H_{nm}(t+2)|$. For any $f(x_1, x_2, \ldots, x_n) \in H_{nm}(t+2)$, we consider two cases:

Case 1, $m \geq n$: there exists a layered connectionist model $M = (t+2, m, n, m, C, B, A)$ which computes f. Each $v_{t+1,j}$ $(j = 1, 2, \ldots, m)$ in M can be represented as a Boolean function in $x_1, x_2, \ldots, x_n$. Since $H_{nm}(t) = H_{nm}(t+1)$, there is a layered connectionist model $N = (t+1, m, n, m, C', B', A')$ such that $v'_{tj} = v_{t+1,j}$ for $j = 1, 2, \ldots, m$, where v'_{tj} is the value of C'_{tj}. We can let $A'_{t+1} = A_{t+2}$ so that N also computes f.

Case 2, $m < n$: there exists a layered connectionist model $M = (t+2, n, n, m, C, B, A)$ which computes f. We can write $f = g(v_{11}, v_{12}, \ldots, v_{1n})$. Since $H_{nm}(t) = H_{nm}(t+1)$, there is a layered connectionist model $N' = (t+1, n, n, m, C'', B'', A'')$ such that $f = g(v''_{11}, v''_{12}, \ldots, v''_{1n})$, where v''_{1j} is the value of C''_{1j} for $j = 1, 2, \ldots, n$. We can let $A''_1 = A_1$. Thus, $v''_{1j} = v_{1j}$ for $j = 1, 2, \ldots, n$; so that N' also computes f.

Thus, $f \in H_{nm}(t+1)$; hence, $H_{nm}(t+2) \subseteq H_{nm}(t+1)$. Therefore, $H_{nm}(t+1) = H_{nm}(t+2)$; and thus $H_{nm}(t) = H_{nm}(t+i)$ for $i \geq 1$. Let t_0 be an integer auch that $|H_{nm}(t_0)| = K$. If $|H_{nm}(t)| < K$, then it is impossible that $t \geq t_0$. However, if $t < t_0$, then we have $H_{nm}(t_0) < K$. Contradiction. Hence, $|H_{nm}(t)| < |H_{nm}(t+1)|$. □

Corollary 4 *If $|H_{nm}(t)| < K$ then $H_{nm}(t) \subset H_{nm}(t+1)$.*

Proof. Clearly, $H_{nm}(t) \subseteq H_{nm}(t+1)$. But, by Theorem 3, $H_{nm}(t) \neq H_{nm}(t+1)$. Hence, $H_{nm}(t) \subset H_{nm}(t+1)$. □

By Corollary 4, $H_{nm}(t+1)$ can be separated from $H_{nm}(t)$. There exists a function $f \in H_{nm}(t+1)$ but $f \notin H_{nm}(t)$.

Suppose $M = (t, w, n, m, C, B, A)$ is a layered connectionist machine. If we add one layer to M, we will add w^2 links to M; and thus w^2 weights are added to M.

The following theorem is due to [Hong, 1987].

Theorem 5 (Hong) *Suppose $r_1, r_2, \ldots, r_n$ are real numbers. There exists integers $m_1, m_2, \ldots, m_n$ such that the length of m_j $(j = 1, 2, \ldots, n)$ is $O(n \log n)$ and for any $c_1, c_2, \ldots, c_n$ in $\{0, 1\}$, we have*

$$\sum_{j=1}^{n} c_j r_j > 0 \text{ iff } \sum_{j=1}^{n} c_j m_j > 0.$$

$r_1, r_2, \ldots, r_n$ are said to be simulated *by $m_1, m_2, \ldots, m_n$.*

By Theorem 5, each weights added can be simulated by an integer with binary length $O(w \log w)$. Thus, w^2 weights has total binary length $O(w^3 \log w)$. Therefore, by adding one more layer to M, we can create $O(w^{w^3})$ different layered connectionist machines. Hence, $|T_{nm}(t+1)|$ is $O((n+m)^{(n+m)^3})$ times of $|T_{nm}(t)|$. But $O((n+m)^{(n+m)^3}) = O(2^{(n+m)^4})$. Thus, $|T_{nm}(t)| = O(2^{t(n+m)^4})$. We conclude that if $|T_{nm}(t)| = 2^{m2^n}$, then

$$t = \Omega(\frac{m2^n}{(n+m)^4}).$$

If $m = O(n)$, then $t = \Omega(2^n/n^3)$. It means that if $w = O(n)$ then $\Omega(2^n/n^3)$ layers are needed for computing an arbitrary function with input size n and output size $O(n)$.

3 2-Layer Connectionist models

Let $M = (t, w, n, m, C, B, A)$ be a layered connectionist machine. If $t = 2$, we call M a *2-layer connectionist machine*. A 2-layer connectionist machine is a special case of connectionist models. It has an input layer, an intermediate layer and an output layer. We denote by w the size of the intermediate layer. There are links connecting the neurons in the input layer to the neurons in the intermediate layer and there are links connecting the neurons in the intermediate layer to the neurons in the output layer; no other links exist. Clearly, the computation time for a 2-layer connectionist machine is 2.

Let $W_{nm}(w)$ denote a class of problems solved by 2-layer connectionist machines with input size n, output size m and width w. We denote by $|W_{nm}(w)|$ the number of elements in $W_{nm}(w)$.

Each problem in $W_{nm}(w)$ can also be regarded as a function

$$f(x_1, x_2, \ldots, x_n) = y_1 \cdots y_m$$

where $x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_m \in \{0, 1\}$. Moreover, each y_j $(j = 1, 2, \ldots, m)$ is a Boolean function in $x_1, x_2, \ldots, x_n$. There are 2^n possible combinations of binary input values. Therefore, we have $|W_{nm}(w)| \leq 2^{m2^n}$. In fact, for an arbitrary function $f : \{0,1\}^n \longmapsto \{0,1\}^m$ there is a 2-layer connectionist models $M = (2, 2^n, n, m, C, B, A)$ which computes f. f can be specified with m Boolean functions, $f_1, f_2, \ldots, f_m$. Each f_j $(j = 1, 2, \ldots, m)$ can be represented by a sum of products form. Each n-digit input value has a corresponding

product term such that the value will produce 1 for this product term. M can be constructed as follows:

Let $v_{11}, v_{12}, \ldots, v_{1,2^n}$ represent 2^n different product terms. This can be done by using the method described in Lemma 1 to set the weights $a_{1ij}(i = 1, \ldots, n;\ j = 1, \ldots, 2^n)$ and thresholds $b_{1j}(j = 1, \ldots, 2^n)$ to proper values. Then, for $i = 1, \ldots, 2^n$ and $j = 1, \ldots, m$, let $a_{2ij} = 1$ if v_{1i} is a product term in the sum of products form of f_j; let $a_{2ij} = 0$ otherwise. Clearly, $v_{2j} = f_j$ for $j = 1, 2, \ldots, m$. Hence, M computes f. We thus proved the following lemma:

Lemma 6 $|W_{nm}(2^n)| = 2^{m2^n}$.

Theorem 7 *For $w \geq 1$, if $|W_{nm}(w)| < 2^{m2^n}$, then $W_{nm}(w) \subset W_{nm}(w+1)$.*

Proof. Since $|W_{nm}(w)| < 2^{m2^n}$, there exists a function $f(x_1, \ldots, x_n) \notin W_{nm}(w)$. Suppose f is specified with m Boolean functions, $f_1, f_2, \ldots, f_m$. Let f_j $(j = 1, 2, \ldots, m)$ be represented as the sum of product form. We define

$$\begin{aligned} S = \ & \{(d_1, \ldots, d_m) : \text{ for } j = 1, 2, \ldots, m, \\ & \text{after delete } d_j \text{ product terms from } f_j, \\ & f \text{ will be in } W_{nm}(w)\} \\ d = \ & \min\{i : i = \min(d_1, \ldots, d_m) \\ & \text{for some } (d_1, \ldots, d_m) \in S\} \end{aligned}$$

Suppose $(e_1, \ldots, e_m) \in S$ and there is e_k, $1 \leq k \leq m$, such that $d = e_k$. Let $g(x_1, \ldots, x_n) \in W_{nm}(w)$ be specified with m Boolean functions, $g_1, g_2, \ldots, g_m$ such that g_j is obtained by deleting e_j product terms from f_j for $j = 1, \ldots, m$. Suppose $p_1, \ldots, p_d$ are the product terms deleted from f_k. Let $h_k = g_k + p_1$ and let h be specified with $g_1, g_2, \ldots, g_{k-1}, h_k, g_{k+1}, \ldots, g_m$. Clearly, $h \notin W_{nm}(w)$, otherwise it will contradict to the definition of d. Suppose g is computed by a 2-layer connectionist model $M = (2, w, n, m, C, B, A)$. We construct a 2-layer connectionist model N by adding one more neuron $C_{1,w+1}$ to M such that $v_{1,w+1} = p_1$ (which can be done by using the method described in Lemma 1) and $a_{2,w+1,k} = \sum_{j=1}^{w} |a_{2jk}| + b_{2k} + 1$. Clearly, N computes h. Thus, $h \in W_{nm}(w+1)$; but $h \notin W_{nm}(w)$. Hence $W_{nm}(w) \neq W_{nm}(w+1)$. But $W_{nm}(w) \subseteq W_{nm}(w+1)$. Hence, we have $W_{nm}(w) \subset W_{nm}(w+1)$. □

Theorem 7 explains the space hierarchy of the 2-layer connectionist models. Suppose $N = (2, w, n, m, C, B, A)$ is a 2-layer connectionist machine. If we add one neuron to layer 1 of N, we will add $(n+1)$ links to N; and thus $(n+1)$ weights are added to N. By Theorem 5, each weight added can be simulated by an integer with binary length $O(n \log n)$. Thus, $(n+1)$ weights has total binary length $O(n^2 \log n)$. Therefore, by adding one more neuron in layer 1 of N, we can create $O(n^{n^2})$ different 2-layer connectionist machines. Hence, $|W_{nm}(w+1)|$ is $O(n^{n^2})$ times of $|W_{nm}(w)|$. But $O(n^{n^2}) = O(2^{n^3})$. Thus, $|W_{nm}(w)| = O(2^{wn^3})$, we conclude that if $|W_{nm}(w)| = 2^{m2^n}$, then

$$w = \Omega(\frac{m2^n}{n^3}).$$

If $m = O(n)$, then $w = \Omega(2^n/n^2)$. It means that there exists a problem with input size n and output size $O(n)$ which can only be solved by the 2-layer connectionist machines with width $\Omega(2^n/n^2)$.

4 Conclusion

Computational complexity for the connectionist models has not been studied much. In the complexity theory, we usually consider the uniform models. However, The connectionist models are non-uniform models. Little is known about the non-uniform models. [Hong, 1987] has proved the similarity between the connectionist models and other non-uniform models. Therefore, for the complexity theory in the non-uniform models, we can just focus upon the connectionist models. In this paper, we presented time and space hierarchy in two extreme cases of the layered connectionist models. Open question: is there time and space hierarchy in the multi-layered connectionist models?

If a function can not be learned by a 2-layer connectionist model with width w = O(n), then, by Corollary 4 and Theorem 7, we may extend the 2-layer connectionist model by adding layers or increasing width such that the extended model can learn this function. The total number of neurons in a connectionist machine is said to be the size of this machine. For computing any function with input size n and output size O(n), both 2-layer connectionist models and layered connectionist models with O(n) width need size $\Omega(2^n/n^2)$. But the 2-layer connectionist models always take time 2 whereas the layered connectionist models with O(n) width may take time $\Omega(2^n/n^3)$. Therefore, for extending a 2-layer connectionist model, we suggest that increasing width is better than adding layers.

Acknowledgments

I thank Professor Jiawei Hong for many useful discussions.

References

[Hong, 1987] Jiawei Hong. On Connectionist Models. Courant Institute, NYU, 1987.

[Minsky and Papert, 1968] Marvin L. Minsky and Seymour A. Papert. Perceptions. The MIT Press, Cambridge, Massachusetts, 1968.