Formulate RNN with Implicit Lifted Net Framework

Alicia Tsai

August 2019

1 Implicit lifted net framework

The implicit prediction model defines a prediction rule that processes an input point $u \in \mathbb{R}^p$ to produce a predicted ouput vector $\hat{y}(u) \in \mathbb{R}^q$ via an implicit equation in some vector $x \in \mathbb{R}^n$:

$$\hat{y}(u) = Cx, \; x = \phi(Ax + Bu) \tag{1}$$

where ϕ is the activation function applied component-wise to a vector. Parameters (weights) of the model are contained in the matrices $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times p}, C \in \mathbb{R}^{q \times n}$. For notation simplicity, we do not include bias terms in the notation. The vector $x \in \mathbb{R}^n$ can be thought of as "state" corresponding to n "hidden" features extracted from the inputs.

2 Formulate feedforward neural networks

Standard feedforward neural network prediction rules can be formulated as the above model with (A, B) strictly upper block diagonal, where the number of blocks is equal to that of hidden layers. Consider a feedforward neural network with $L > 1$ layers:

$$\hat{y}(u) = W_L x_L, \; x_{l+1} = \phi(W_l x_l), \; x_0 = u \tag{2}$$

Here $W_l,\ l = 1, \cdots, L$, are given weight matrices. Equation (2) can be expressed as (1), with $x = (x_L, \cdots, x_1)$, and the weight matrices

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right] = \left[\begin{array}{cccc|c} 0 & W_{L-1} & \cdots & 0 & 0 \\ 0 & 0 & \ddots & \vdots & \vdots \\ & & \ddots & W_1 & 0 \\ & & & 0 & W_0 \\ \hline W_L & 0 & \cdots & 0 & 0 \end{array}\right] \tag{3}$$

where A, B are strictly upper block triangular. The implicit rule is computed via the block matrix multiplication

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right] \left[\begin{array}{c} x \\ \hline u \end{array}\right] = \left[\begin{array}{c} Ax + Bu \\ \hline Cx \end{array}\right] \tag{4}$$

The implicit equation $x = \phi(Ax + Bu)$ can be solved via a forward pass through the network.

3 Formulate recurrent neural networks

A simple RNN has three components which are inputs, recurrent hidden states, and outputs. The input is a sequence of vectors through time t, such as $u_1, \cdots, u_{t-1}, u_t$, where each input u_t has p input units $u_t = (u_t^1, u_t^2, \cdots, u_t^p)$. At each time step, the network takes in a input u_t and the previous hidden state h_{t-1} to produce the next hidden state h_t. The first hidden state h_0 is initialized randomly at first. The hidden state h_t defines the state space or "memory" of the network.

$$h_t = \phi_H(W_H h_{t-1} + W_I u_t) \tag{5}$$

Here W_H is the "shared" weight matrix for hidden state and W_I is the "shared" weight matrix for the input. ϕ_H is the activation function for the hidden state. The model can output at every time step except $t = 0$ or only output at the end of the sequence depending on the task. The output at each time step y_t has q units $y_t = (y_t^1, y_t^2, \cdots, y_t^q)$ that are computed via

$$y_t = \phi_O(W_O h_t) \tag{6}$$

where ϕ_O is the activation function for the output layer.

If we consider the case when the network only outputs at the end of the sequence at time step t. Then, equation (5) and (6) can be expressed as (1), with $x = (h_t, h_{t-1}, \cdots, h_0)$, $u = (u_t, \cdots, u_1)$, and weight matrices

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right] = \left[\begin{array}{ccccc|cccc} 0 & W_H & \cdots & 0 & 0 & W_I & \cdots & 0 & 0 \\ 0 & 0 & W_H & \vdots & \vdots & 0 & W_I & \vdots & \vdots \\ & & \ddots & \ddots & 0 & & \ddots & \ddots & 0 \\ & & & 0 & W_H & & & 0 & W_I \\ \hline W_O & 0 & \cdots & 0 & 0 & 0 & \cdots & \cdots & 0 \end{array}\right] \tag{7}$$

where A, B are strictly upper block triangular and have the same block diagonal submatrices, W_H and W_I respectively, shared across all hidden state and input. The implicit rule is computed via the block matrix multiplication

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right] \left[\begin{array}{c} x \\ \hline u \end{array}\right] = \left[\begin{array}{ccccc|cccc} 0 & W_H & \cdots & 0 & 0 & W_I & \cdots & 0 & 0 \\ 0 & 0 & W_H & \vdots & \vdots & 0 & W_I & \vdots & \vdots \\ & & \ddots & \ddots & 0 & & \ddots & \ddots & 0 \\ & & & 0 & W_H & & & 0 & W_I \\ \hline W_O & 0 & \cdots & 0 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & \ddots & \vdots & 0 & 0 & \ddots & \vdots & 0 \\ & & \ddots & \ddots & 0 & & & \ddots & 0 \end{array}\right] \left[\begin{array}{c} h_t \\ h_{t-1} \\ \vdots \\ h_1 \\ h_0 \\ \hline u_t \\ \vdots \\ u_1 \end{array}\right]$$

Now, if we consider the case when the network outputs at every time step t, then the matrix C would become

$$C = \begin{bmatrix} W_O & 0 & \cdots & 0 \\ 0 & W_O & 0 & \vdots \\ & \ddots & W_O & 0 \\ & & 0 & W_O \end{bmatrix}$$

3.1 Bi-directional RNN

Conventional RNN only consider the previous context of data. A bi-directional RNN (BRNN) considers the input sequence in both the past and the the future. BRNN uses one RNN layer to process the sequence from start to end in a forward time direction and another RNN to process the sequence backwards from end to start in a backward time direction. The forward and backward hidden sequences are denoted by $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ at time t. The forward hidden sequence is computed as

$$\overrightarrow{h}_t = \phi_H\big(W_{\overrightarrow{H}}\overrightarrow{h}_{t-1} + W_{\overrightarrow{I}}u_t\big) \tag{8}$$

where it is iterated over $t = (1, ..., T)$. The backward layer is

$$\overleftarrow{h}_t = \phi_H\big(W_{\overleftarrow{H}}\overleftarrow{h}_{t-1} + W_{\overleftarrow{I}}u_t\big) \tag{9}$$

which is iterated backward over time $t = (T, ..., 1)$. The output sequence y_t at time t is

$$y_t = \phi_O\big(W_{\overrightarrow{O}}\overrightarrow{h}_t + W_{\overleftarrow{O}}\overleftarrow{h}_t\big) \tag{10}$$

Let us first look at the hidden state and input. Equation (8) and (9) can be expressed with $\overrightarrow{x} = (\overrightarrow{h}_t, \overrightarrow{h}_{t-1}, \cdots, \overrightarrow{h}_0)$, $\overleftarrow{x} = (\overleftarrow{h}_t, \overleftarrow{h}_{t-1}, \cdots, \overleftarrow{h}_0)$, $u = (u_t, \cdots, u_1)$, and weight matrices

$$\left[\begin{array}{c|c} A & B \end{array}\right] = \left[\begin{array}{cc|c} W_{\overrightarrow{H}} & 0 & W_{\overrightarrow{I}} \\ 0 & W_{\overleftarrow{H}} & W_{\overleftarrow{I}} \end{array}\right] \tag{11}$$

where only A is strictly upper block triangular. Again, the implicit rule is computed via the block matrix multiplication

$$\left[\begin{array}{cc|c} W_{\overrightarrow{H}} & 0 & W_{\overrightarrow{I}} \\ 0 & W_{\overleftarrow{H}} & W_{\overleftarrow{I}} \end{array}\right] \left[\begin{array}{c} \overrightarrow{x} \\ \overleftarrow{x} \\ \hline u \end{array}\right] \tag{12}$$

Similarly, the output layer can be expressed via the matrix C

$$C = \left[\begin{array}{c:c} W_{\overrightarrow{O}} & W_{\overleftarrow{O}} \end{array}\right] = \left[\begin{array}{cccc:cccc} W_{\overrightarrow{O}} & 0 & \cdots & 0 & 0 & \cdots & 0 & W_{\overleftarrow{O}} \\ 0 & W_{\overrightarrow{O}} & 0 & \vdots & \vdots & 0 & W_{\overleftarrow{O}} & 0 \\ & \ddots & W_{\overrightarrow{O}} & 0 & 0 & W_{\overleftarrow{O}} & \ddots & \\ & & 0 & W_{\overrightarrow{O}} & W_{\overleftarrow{O}} & 0 & & \end{array}\right]$$

Finally, equation (8), (9), and (10) can be formatted as

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right] \left[\begin{array}{c} \overrightarrow{x} \\ \overleftarrow{x} \\ \hline u \end{array}\right] = \left[\begin{array}{cc|c} W_{\overrightarrow{H}} & 0 & W_{\overrightarrow{I}} \\ 0 & W_{\overleftarrow{H}} & W_{\overleftarrow{I}} \\ \hline W_{\overrightarrow{O}} & W_{\overleftarrow{O}} & 0 \end{array}\right] \left[\begin{array}{c} \overrightarrow{x} \\ \overleftarrow{x} \\ \hline u \end{array}\right]$$