# STAT 153 & 248 - Time Series
# Lecture Twenty Six

**Spring 2025, UC Berkeley**

Aditya Guntuboyina

May 01, 2025

## 1 RNN

RNN is given by

$$\begin{aligned}
r_0 &= 0 \\
s_t &= W_r r_{t-1} + W x_t + b \\
r_t &= \sigma_{\tanh}(s_t) \\
\mu_t &= \beta_0 + \beta^T r_t
\end{aligned} \tag{1}$$

This formula can also be written as

$$\begin{aligned}
r_0 &= 0 \\
r_t &= \sigma_{\tanh}(W_r r_{t-1} + W x_t + b) \\
\mu_t &= \beta_0 + \beta^T r_t
\end{aligned} \tag{2}$$

Here the activation function $\sigma_{\tanh}$ is the tanh activation function given by

$$\sigma_{\tanh}(u) := \frac{e^u - e^{-u}}{e^u + e^{-u}}.$$

The parameters now are $W_r$ ($k \times k$ matrix), $W$ ($k \times p$ matrix), $b$ ($k \times 1$ vector), $\beta_0$ (scalar) and $\beta$ ($k \times 1$ vector).

In the last lecture, we saw that RNNs have a "lack of long memory" problem. This means that even though $r_t$ technically depends on all of $x_t, x_{t-1}, \dots$, in practice, it is mainly controlled by $x_u$ for $u$ close to $t$. This problem is fixed, to some extent, by GRUs and LSTMs.

## 2 GRU (Gated Recurrent Unit)

GRU is

$$\begin{aligned}
r_0 &= 0 \\
g_t &= \sigma_{\text{sigmoid}}(W_r^g r_{t-1} + W^g x_t + b^g) \\
z_t &= \sigma_{\text{sigmoid}}(W_r^z r_{t-1} + W^z x_t + b^z) \\
\tilde{r}_t &:= \sigma_{\tanh}(W_r(r_{t-1} \odot g_t) + W x_t + b) \\
r_t &= z_t \odot r_{t-1} + (1 - z_t) \odot \tilde{r}_t \\
\mu_t &= \beta_0 + \beta^T r_t.
\end{aligned} \tag{3}$$

$z_t$ is called the update gate while $g_t$ is called the reset gate. The unknown parameters in this model (which need to be estimated from the data) are $W_r^g, W^g, b^g, W_r^z, W^z, b^z, W_r, W, b, \beta_0, \beta$.

Because of the presence of $z_t$, it is possible for $r_t$ to be quite close to $r_{t-1}$ for many time points $t$. This allows $r_t$ to have a relatively long memory.

# 3 LSTM (Long Short Term Memory)

LSTM is another modification to the basic RNN for enabling long memory. It also uses gates and has one more gate compared to the GRU. Instead of a recursion directly between $r_{t-1}$ and $r_t$, the LSTM recursions are between the pairs $(s_{t-1}, r_{t-1}) \to (s_t, r_t)$:

$$
\begin{aligned}
&r_0 = 0 \text{ and } s_0 = 0 \\
&f_t = \sigma_{\text{sigmoid}}(W_r^f r_{t-1} + W^f x_t + b^f) \\
&i_t = \sigma_{\text{sigmoid}}(W_r^i r_{t-1} + W^i x_t + b^i) \\
&o_t = \sigma_{\text{sigmoid}}(W_r^o r_{t-1} + W^o x_t + b^o) \\
&\tilde{r}_t := \sigma_{\tanh}(W_r r_{t-1} + W x_t + b) \\
&s_t = f_t \odot s_{t-1} + i_t \odot \tilde{r}_t \\
&r_t = o_t \odot \sigma_{\tanh}(s_t) \\
&\mu_t = \beta_0 + \beta^T r_t
\end{aligned}
\tag{4}
$$

$f_t$ is called the forget gate, $i_t$ is called the input gate and $o_t$ is called the output gate. The presence of these gates allow $r_t$ to draw information from $x_u$ even for $u$ quite far from $t$.

The unknown parameters in this model are $W_r^f, W^f, b^f, W_r^i, W^i, b^i, W_r^o, W^o, b^o, W_r, W, b, \beta_0, \beta$.

The LSTM unit is all the equations in (4) excluding the last linear layer $\mu_t = \beta_0 + \beta^T r_t$:

$$
\begin{aligned}
&r_0 = 0 \text{ and } s_0 = 0 \\
&f_t = \sigma_{\text{sigmoid}}(W_r^f r_{t-1} + W^f x_t + b^f) \\
&i_t = \sigma_{\text{sigmoid}}(W_r^i r_{t-1} + W^i x_t + b^i) \\
&o_t = \sigma_{\text{sigmoid}}(W_r^o r_{t-1} + W^o x_t + b^o) \\
&\tilde{r}_t := \sigma_{\tanh}(W_r r_{t-1} + W x_t + b) \\
&s_t = f_t \odot s_{t-1} + i_t \odot \tilde{r}_t \\
&r_t = o_t \odot \sigma_{\tanh}(s_t)
\end{aligned}
\tag{5}
$$

The output of the LSTM unit is $r_t$. In PyTorch (see `https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html`), the notation used for the LSTM unit differs slightly from (5). The LSTM PyTorch unit is given by:

$$
\begin{aligned}
&i_t = \sigma(W_{ii} x_t + b_{ii} + W_{hi} h_{t-1} + b_{hi}), \\
&f_t = \sigma(W_{if} x_t + b_{if} + W_{hf} h_{t-1} + b_{hf}), \\
&g_t = \tanh(W_{ig} x_t + b_{ig} + W_{hg} h_{t-1} + b_{hg}), \\
&o_t = \sigma(W_{io} x_t + b_{io} + W_{ho} h_{t-1} + b_{ho}), \\
&c_t = f_t \odot c_{t-1} + i_t \odot g_t \\
&h_t = o_t \odot \tanh(c_t),
\end{aligned}
\tag{6}
$$

where $\sigma = \sigma_{\text{sigmoid}}$.

Both (5) and (6) implement the same LSTM update, but they differ in notation. Our $r_t$ is named $h_t$ in PyTorch, and our $s_t$ is named $c_t$ in PyTorch. $h_t$ is referred to as the hidden state and $c_t$ is referred to as the cell state.

The three gates have the same notation in both formulae: forget $f_t$, input $i_t$, output $o_t$. The candidate or potential feature vector $\tilde{r}_t$ in (5) is denoted by $g_t$ in (6). In (6), there are two bias vectors in each of the formulae for $i_t, f_t, g_t, o_t$. We combined these to one bias vector in (5). Other than these notational differences, the formulae (5) and (6) are identical.

## 4 Additional Optional Reading

1. Check out the PyTorch documentation for LSTM, GRU and RNN (`https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html`, `https://pytorch.org/docs/stable/generated/torch.nn.GRU.html` and `https://pytorch.org/docs/stable/generated/torch.nn.RNN.html`)
2. This is a popular blog post (by Andrej Karpathy) on how RNNs can be used for many interesting tasks in NLP: `https://karpathy.github.io/2015/05/21/rnn-effectiveness/`.