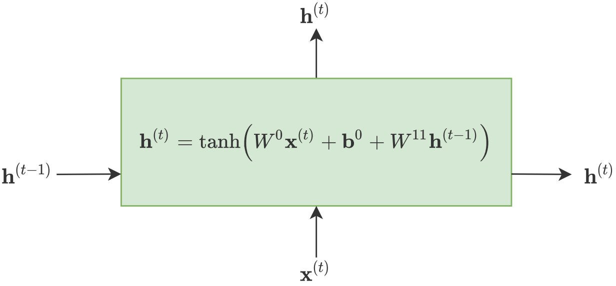
CS-568 Deep Learning

Long Short-Term Memory (LSTM)

Nazar Khan
Department of Computer Science
University of the Punjab

Weakness of standard RNN

- We have already seen that RNNs do not possess long-term memory.
- Input at time t is soon forgotten because of the recurrent weights W^{11}.
- Would be nice to decide what and how much to forget/remember based on the input itself.



RNN Cell: Operations at the hidden layer.

Long Short-Term Memory (LSTM)
Building blocks

Let $\mathbf{v}^{(t)} = \begin{bmatrix} \mathbf{h}^{(t-1)} \\ \mathbf{x}^{(t)} \end{bmatrix} \in \mathbb{R}^{(M+D)\times 1}$

Perform 4 affine transformations of $\mathbf{v}^{(t)}$ followed by non-linearities.

$$\mathbf{f}^{(t)} = \sigma\left(W_f \mathbf{v}^{(t)} + \mathbf{b}_f\right) \tag{1}$$

$$\mathbf{i}^{(t)} = \sigma\left(W_i \mathbf{v}^{(t)} + \mathbf{b}_i\right) \tag{2}$$

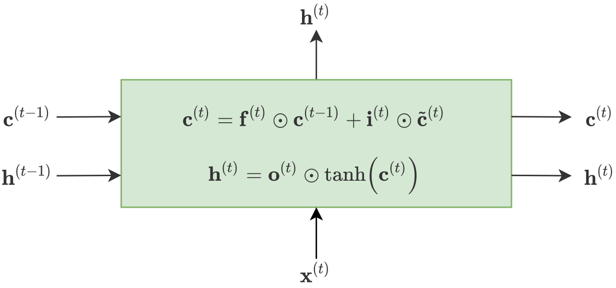
$$\mathbf{o}^{(t)} = \sigma\left(W_o \mathbf{v}^{(t)} + \mathbf{b}_o\right) \tag{3}$$

$$\tilde{\mathbf{c}}^{(t)} = \tanh\left(W_c \mathbf{v}^{(t)} + \mathbf{b}_c\right) \tag{4}$$

All 4 matrices of size $M \times (M+D)$ and therefore all 4 transformations produce M-dimensional vectors.
Vectors $\mathbf{f}^{(t)}, \mathbf{i}^{(t)}, \mathbf{o}^{(t)}$ contain values in $(0,1)$ and $\tilde{\mathbf{c}}^{(t)}$ in $(-1,1)$.

LSTM

Putting everything together



LSTM Cell: Operations at the hidden layer.

- Vector $\mathbf{c}^{(t)}$ is *interpreted* as the *cell state*.
- Cell state is recurrent as well.
- Notice that $\mathbf{c}^{(t)}$ is not forced to contain values in $(0, 1)$ or $(-1, 1)$.

Role of the Gates

$\mathbf{f}^{(t)}$: *Forget Gate*

$$
\begin{aligned}
\mathbf{f}^{(t)} &= \sigma\left(W_f \mathbf{v}^{(t)} + \mathbf{b}_f\right) \\
c_j^{(t)} &= f_j^{(t)} c_j^{(t-1)} + i_j^{(t)} \tilde{c}_j^{(t)}
\end{aligned}
$$

- $f_j^{(t)} \in (0,1)$ due to the logistic sigmoid.
- If $f_j^{(t)} = 0$, then $c_j^{(t-1)}$ is *forgotten* in the next state $c_j^{(t)}$.
- If $f_j^{(t)} = 1$, then $c_j^{(t-1)}$ is *retained completely* in the next state $c_j^{(t)}$.

$\mathbf{f}^{(t)}$ acts as a forget gate on the previous cell state $\mathbf{c}^{(t)}$.

Role of the Gates

$\mathbf{i}^{(t)}$: *Input Gate*

$$\mathbf{i}^{(t)} = \sigma\left(W_i \mathbf{v}^{(t)} + \mathbf{b}_i\right)$$
$$c_j^{(t)} = f_j^{(t)} c_j^{(t-1)} + i_j^{(t)} \tilde{c}_j^{(t)}$$

- $i_j^{(t)} \in (0, 1)$ due to the logistic sigmoid.
- If $i_j^{(t)} = 0$, then no new information will be added to $c_j^{(t)}$.
- If $i_j^{(t)} = 1$, then the potential cell state $\tilde{c}_j^{(t)}$ is *added completely* in the next state $c_j^{(t)}$ *irrespective of forget level*.

$\mathbf{i}^{(t)}$ acts as an input gate on the potential cell state $\tilde{\mathbf{c}}^{(t)}$.

Role of the Gates

$\mathbf{o}^{(t)}$: *Output Gate*

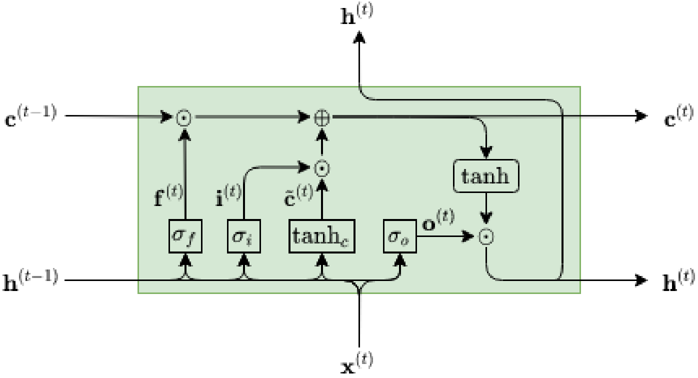
$$\mathbf{o}^{(t)} = \sigma\left(W_o \mathbf{v}^{(t)} + \mathbf{b}_o\right)$$

$$h_j^{(t)} = o_j^{(t)} \tanh(c_j^{(t)})$$

- $o_j^{(t)} \in (0, 1)$ due to the logistic sigmoid.
- If $o_j^{(t)} = 0$, then cell state $c_j^{(t)}$ will be *completely hidden*.
- If $o_j^{(t)} = 1$, then cell state $c_j^{(t)}$ is *completely exposed* in both space ($\uparrow$) and time ($\rightarrow$).

> $\mathbf{o}^{(t)}$ acts as an output gate on $\mathbf{c}^{(t)}$.

LSTM



LSTM Cell: Operations at the hidden layer in detail.

Information flow

- Depending on $\mathbf{f}^{(t)}$ and $\mathbf{i}^{(t)}$, an LSTM cell has the ability to push through its cell state $\mathbf{c}^{(t-1)}$ exactly or almost unchanged into the next time step $\mathbf{c}^{(t)}$.
- This ensures *flow of the cell state (memory) through time*. Hence long-term memory.
- This is similar to how other deep learning techniques ensure flow of information in space.
  - ReLU
  - Weight initialization
  - Batchnorm
  - Residual block

Remembering the past

- Consider a sentence containing brackets.

  England (last year's winners) are expected to put up a good fight.
- The LSTM cell can learn to set $c_j = 1$ if an opening bracket is seen at time t.
- It can also learn to keep $c_j = 1$ for a long time until a closing bracket is seen *in the input*.
- Some other c_k can similarly be used to handle nested brackets and so on.
- Even the value of c_j itself can be used to signify the level of nesting. It all depends on how and what the LSTM learns.

Peephole Connections

- Allow gates to look at the cell state as well before deciding what to forget, what to add, and what to output.

$$\mathbf{v}_{f,i}^{(t)} = \begin{bmatrix} \mathbf{c}^{(t-1)} \\ \mathbf{h}^{(t-1)} \\ \mathbf{x}^{(t)} \end{bmatrix} \in \mathbb{R}^{(2M+D)\times 1}$$

$$\mathbf{v}_{o}^{(t)} = \begin{bmatrix} \mathbf{c}^{(t)} \\ \mathbf{h}^{(t-1)} \\ \mathbf{x}^{(t)} \end{bmatrix} \in \mathbb{R}^{(2M+D)\times 1}$$

Coupled forget and input

- Use a single forget gate for interpolation.

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \odot \mathbf{c}^{(t-1)} + (1 - \mathbf{f}^{(t)}) \odot \tilde{\mathbf{c}}^{(t)}$$

- Fewer parameters due to removal of input gate.

Gated Recurrence Unit (GRU)

- Coupled forget and input gates.
- Merged hidden and cell state.

$$\mathbf{z}^{(t)} = \sigma\left(W_z\mathbf{v}^{(t)} + \mathbf{b}_z\right)$$
$$\mathbf{r}^{(t)} = \sigma\left(W_r\mathbf{v}^{(t)} + \mathbf{b}_r\right)$$
$$\tilde{\mathbf{h}}^{(t)} = \tanh\left(W_h[\mathbf{r}^{(t)} \odot \mathbf{h}^{(t-1)}; \mathbf{x}^{(t)}] + \mathbf{b}_h\right)$$
$$\mathbf{h}^{(t)} = \left(1 - \mathbf{z}^{(t)}\right) \odot \mathbf{h}^{(t-1)} + \mathbf{z}^{(t)} \odot \tilde{\mathbf{h}}^{(t)}$$

- Always expose the hidden state.
- In some variants, the weight matrices can be set to 0.
- In other variants, the bias vectors can be set to 0.
- Fewer parameters, faster training, learning from lesser data.

Summary

- RNNs do not retain long-term context in practice.
- An LSTM cell has the ability to push through its cell state exactly or almost unchanged into the next time step.
- This ensures flow of the cell state (memory) through time. Hence long-term memory.