LSTMs and GRUs

Kate Farrahi

ECS Southampton

March 14, 2019

Vanishing Gradients in RNNs

The influence of a given input on the hidden layer, and therefore on the network output, decays as it cycles around the network's recurrent connections. This effect is referred to as the vanishing gradient problem.

Vanishing Gradients in RNNs

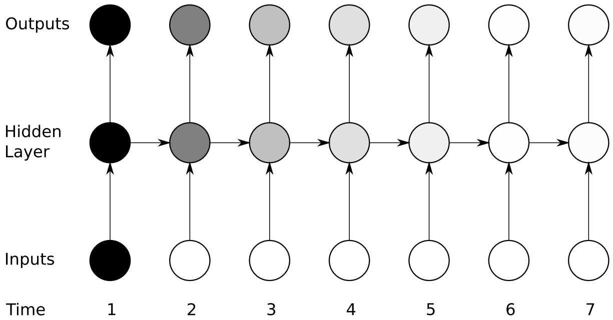


Figure 4.1: **The vanishing gradient problem for RNNs.** The shading of the nodes in the unfolded network indicates their sensitivity to the inputs at time one (the darker the shade, the greater the sensitivity). The sensitivity decays over time as new inputs overwrite the activations of the hidden layer, and the network 'forgets' the first inputs. [1]

[1] Graves, Alex. Supervised sequence labelling with recurrent neural networks. Springer, Berlin, Heidelberg, 2012.

Vanishing Gradients in RNNs

- RNNs have difficulties learning long-range dependencies, i.e. interactions between words that are several steps apart.
- For example, consider the subject-verb agreement in the sentences below:
- *Our experiments* are based on a variety of datasets to observe the information trajectory of cascade learning and *show* consistency across the results.
- *Our experiment* is based on a variety of datasets to observe the information trajectory of cascade learning and *shows* consistency across the results.

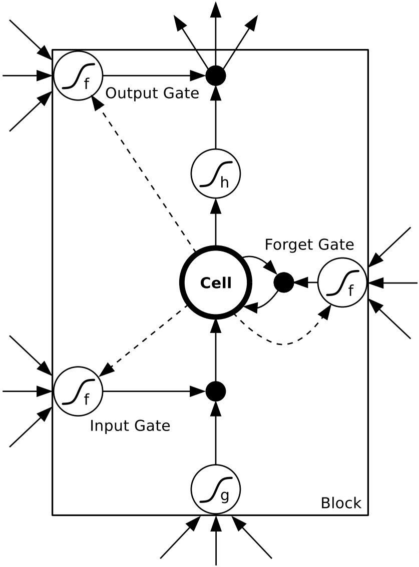
Long Short-Term Memory (LSTM)

- LSTM was proposed in 1997 by S. Hochreiter and J. Schmidhuber [2]
- The initial version of LSTM block included cells, input and output gates.
- In 1999, Felix Gers and his advisor J. Schmidhuber and Fred Cummins introduced the forget gate into LSTM architecture, enabling the LSTM to reset its own state.
- In 2014, K. Cho et al. put forward a simplified variant called Gated recurrent unit (GRU)[3].

[2] Hochreiter, S., and J. Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[3] Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". arXiv:1406.1078

LSTM Memory Block



[4]

[4] Graves, Alex. Supervised sequence labelling with recurrent neural networks. Springer, Berlin, Heidelberg, 2012.

LSTM Network

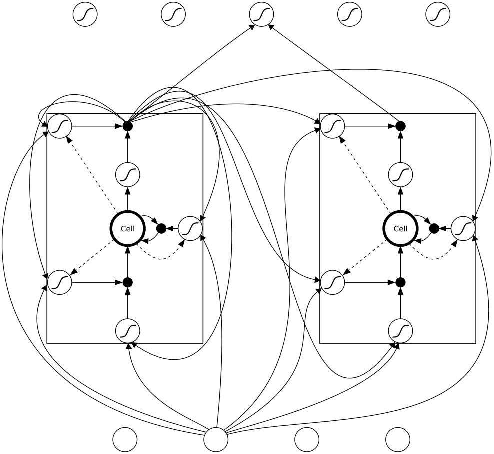


Figure 4.3: **An LSTM network.** The network consists of four input units, a hidden layer of two single-cell LSTM memory blocks and five output units. Not all connections are shown. Note that each block has four inputs but only one output.[5]

[5] Graves, Alex. Supervised sequence labelling with recurrent neural networks. Springer, Berlin, Heidelberg, 2012.

LSTM Network

- The multiplicative gates allow LSTM memory cells to store and access information over longer periods of time, thereby mitigating the vanishing gradient problem.
- As long as the input gate remains closed, the activation of the cell will not be overwritten and can be available to the network much later in the sequence.
- This preservation of information is shown in the next figure.

LSTM preservation of gradient information

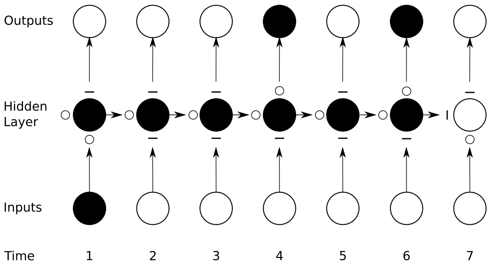


Figure 4.4: **Preservation of gradient information by LSTM.** As in Figure 4.1 the shading of the nodes indicates their sensitivity to the inputs at time one; in this case the black nodes are maximally sensitive and the white nodes are entirely insensitive. The state of the input, forget, and output gates are displayed below, to the left and above the hidden layer respectively. For simplicity, all gates are either entirely open ('O') or closed ('—'). The memory cell 'remembers' the first input as long as the forget gate is open and the input gate is closed. The sensitivity of the output layer can be switched on and off by the output gate without affecting the cell.[6]

[6] Generating sequences with recurrent neural networks. A Graves - arXiv preprint arXiv:1308.0850, 2013

LSTM preservation of gradient information

Adam's visualisations of LSTMs for COMP6208 are excellent:

https://secure.ecs.soton.ac.uk/notes/comp6208/lectures/lstm.pdf

LSTMs

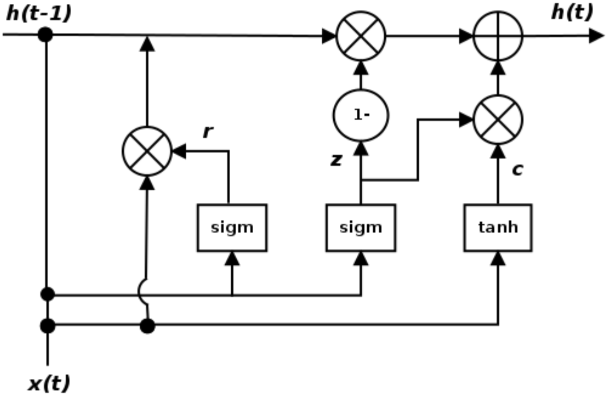
$$
\begin{aligned}
z(t) &= (x(t), y(t-1)) && (1)\\
f(t) &= \sigma(W_f z(t)) && (2)\\
g(t) &= \sigma(W_g z(t)) && (3)\\
h(t) &= tanh(W_h z(t)) && (4)\\
c(t) &= f(t) \odot c(t-1) + g(t) \odot h(t) && (5)\\
o(t) &= \sigma(W_o z(t)) && (6)\\
y(t) &= o(t) \odot tanh(c(t)) && (7)\\
& && (8)
\end{aligned}
$$

LSTM Success Stories

- LSTMs have been used to win many competitions in speech and handwriting recognition
- Major technology companies including Google, Apple, and Microsoft are using LSTMs as fundamental components in new products.
- Google used LSTM for speech recognition on the smartphone, for Google Translate.
- Apple uses LSTM for the "Quicktype" function on the iPhone and for Siri.
- Amazon uses LSTM for Amazon Alexa.
- In 2017, Facebook performed some 4.5 billion automatic translations every day using long short-term memory networks [7]

[7] https://en.wikipedia.org/wiki/Long_short-term_memory

Gated Recurrent Unit (GRU)



Gated Recurrent Unit (GRU)

- x_t: input vector
- h_t: output vector
- z_t: update gate vector
- r_t: reset gate vector
- W, U, and b: parameter matrices and vector
- *sigm* or σ_g is the sigmoid function
- *tanh* or σ_h is the hyperbolic tangent

Gated Recurrent Unit (GRU)

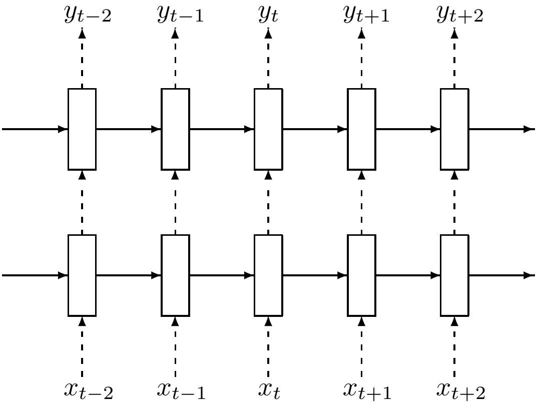
Initially, for $t = 0$, $h_0 = 0$

$$\begin{aligned}
z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \\
r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \sigma_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)
\end{aligned}$$

GRU vs. LSTM?

- GRUs have two gates (reset and update) whereas LSTM has three gates (input/output/forget)
- GRU performance on par with LSTM but computationally more efficient (less complex).
- In general, if you have a very large dataset then LSTMs will likely perform better.
- GRUs are a good choice for smaller datasets.

Regularization in RNNs with LSTM units



[8]

[8] https://arxiv.org/pdf/1409.2329.pdf

Regularization in RNNs with LSTM units

$$\mathbf{LSTM}: h_t^{l-1}, h_{t-1}^l, c_{t-1}^l \rightarrow h_t^l, c_t^l$$
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \mathrm{sigm} \\ \mathrm{sigm} \\ \mathrm{sigm} \\ \tanh \end{pmatrix} T_{2n,4n} \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$
$$c_t^l = f \odot c_{t-1}^l + i \odot g$$
$$h_t^l = o \odot \tanh(c_t^l)$$

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \mathrm{sigm} \\ \mathrm{sigm} \\ \mathrm{sigm} \\ \tanh \end{pmatrix} T_{2n,4n} \begin{pmatrix} \mathbf{D}(h_t^{l-1}) \\ h_{t-1}^l \end{pmatrix}$$
$$c_t^l = f \odot c_{t-1}^l + i \odot g$$
$$h_t^l = o \odot \tanh(c_t^l)$$
[9]

[9] `https://arxiv.org/pdf/1409.2329.pdf`