Deep Learning III : memory-based layers for learning sequences

Moacir Ponti
ICMC, Universidade de São Paulo

Contact: www.icmc.usp.br/~moacir — moacir@icmc.usp.br

Rio de Janeiro/Brazil – April, 2019

Agenda

When data sequence matters

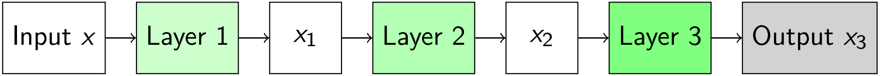
Basic recurrent layer (RNN)

Long Short Term Memory (LSTM)

Concluding remarks

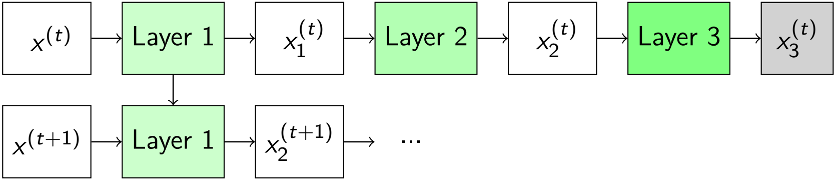
Non-sequence processing

- Dense and convolutional layers only consider the current example to compute their output
- In every iteration, the input moves forward, until reaching the output



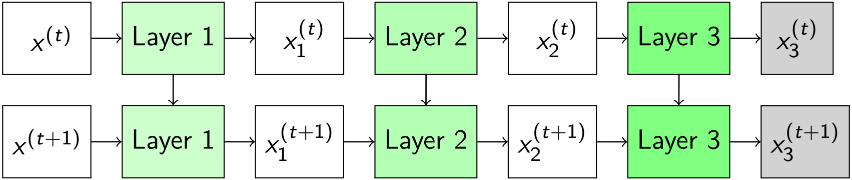
When sequence is important

- What if some output of a layer at iteration t is used as an additional input to the same layer at iteration $t+1$?



When sequence is important

- This way the output depends not only on the current input, but previous outputs of the same layers

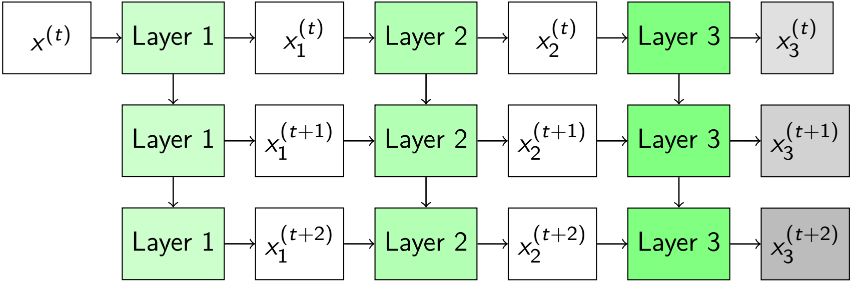


Different configurations of sequences

- One input, sequence output
- Sequence input, one output
- Sequence input, sequence output

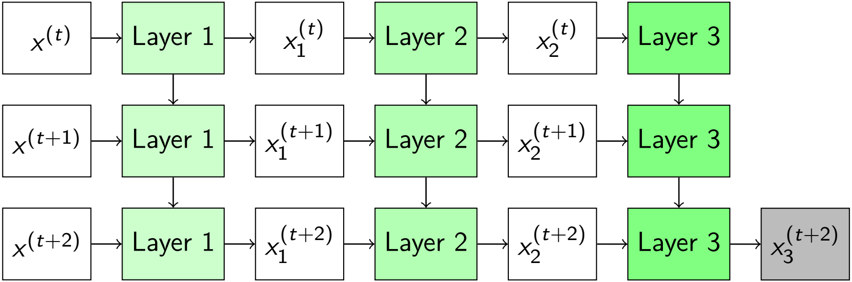
Different configurations of sequences

- One input, sequence output: e.g. one image is provided and the network outputs a sequence of words describing it



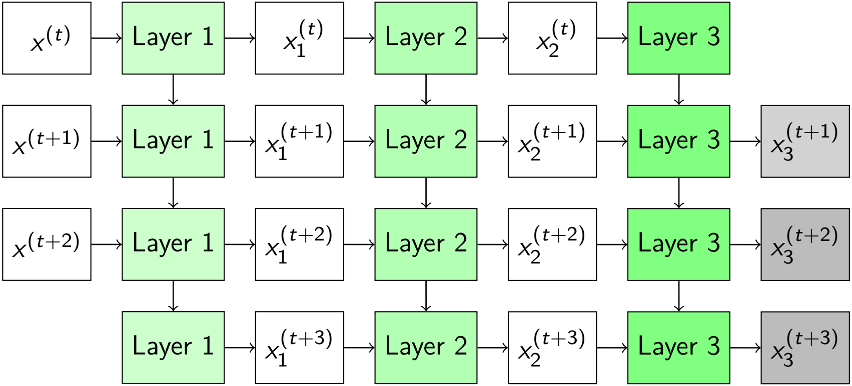
Different configurations of sequences

- Sequence input, one output: e.g. sentence (text) is given and the output is a sentiment analysis, into positive or negative content.



Different configurations of sequences

- Sequence input, sequence output: e.g. machine translation of sentences in different languages (it may or not have a delay)



Agenda

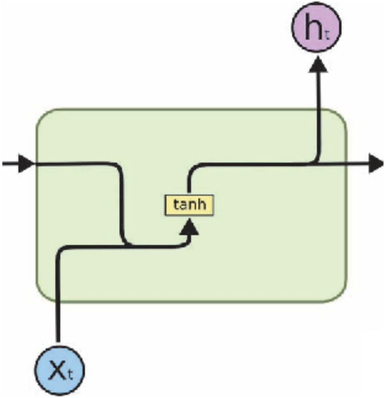
When data sequence matters

Basic recurrent layer (RNN)

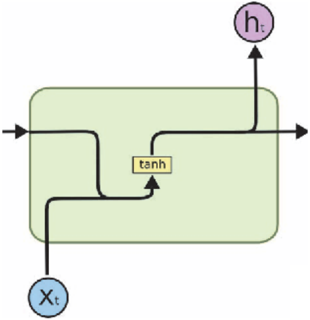
Long Short Term Memory (LSTM)

Concluding remarks

Recurrent layer: to remember or to forget?



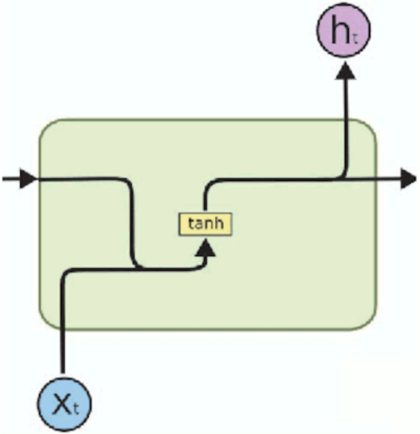
Recurrent layer: to remember or to forget?



$$\mathbf{h}_t = \tanh\left(W_h \mathbf{h}_{t-1} + W_x \mathbf{x}_t + b_h\right)$$
$$y = \left(W_y \mathbf{h}_t + b_y\right)$$

Recurrent layer: to remember or to forget?

$$\mathbf{h}_t = \tanh\left(W_h \mathbf{h}_{t-1} + W_x \mathbf{x}_t + b_h\right)$$



$$y = \left(W_y \mathbf{h}_t + b_y\right)$$

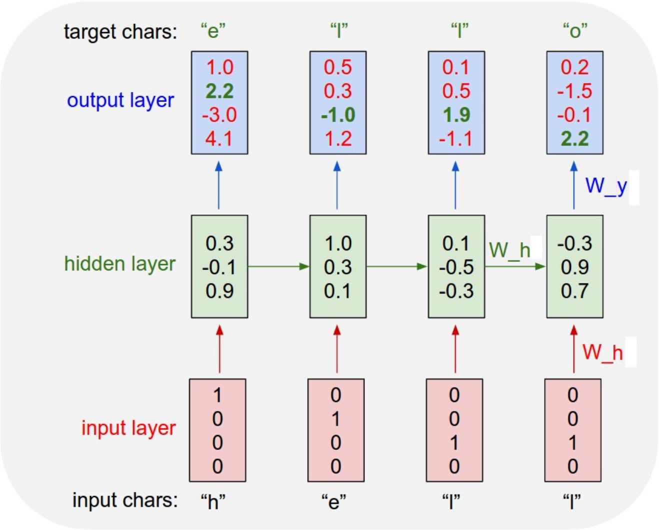
Example: predicting next character

Let us define a one-hot vector for characters so that:

- $\mathtt{h} = [1, 0, 0, 0]$
- $\mathtt{e} = [0, 1, 0, 0]$
- $\mathtt{l} = [0, 0, 1, 0]$
- $\mathtt{o} = [0, 0, 0, 1]$

`http://karpathy.github.io/2015/05/21/rnn-effectiveness/`

Example: predicting next character



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Agenda

When data sequence matters

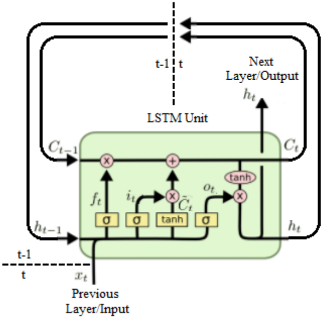
Basic recurrent layer (RNN)

Long Short Term Memory (LSTM)

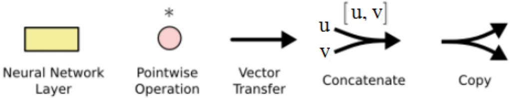
Concluding remarks

Long Short Term Memory Unit (LSTM)

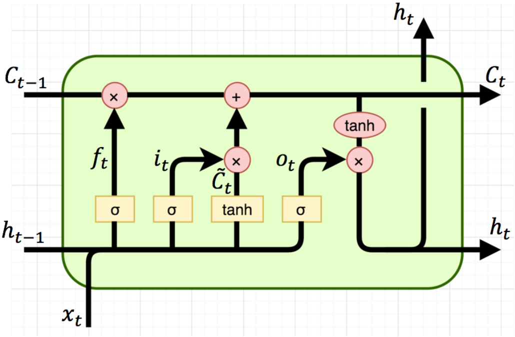
Understanding LSTM Networks



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

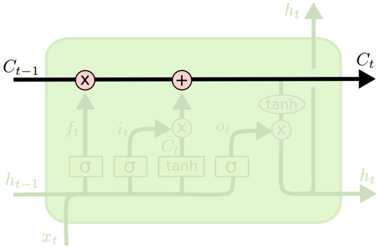


Long Short Term Memory Unit (LSTM)



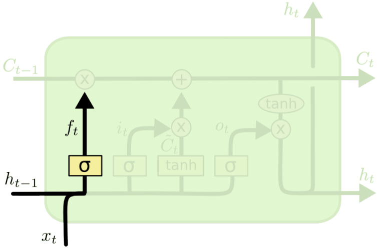
This and following figures are from `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`

LSTM network: Cell line



- Runs down the entire chain, with minor linear interactions
- LSTM may remove or add information to the cell state, via 3 **gates**

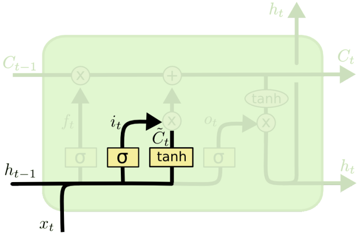
LSTM network: forget gate



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

- decide what to cancel out from the cell state
- outputs values between 0 (forget) and 1 (keep entirely) for each value of the cell state vector

LSTM network: input and update gate

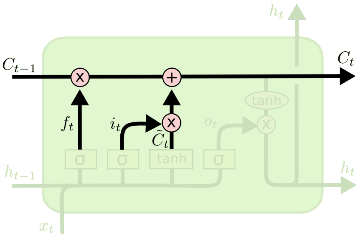


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

- first, it combines previous output h_{t-1} and the input x_t
- then, it filters out those by learning $\tilde{C}_t$, which are candidate values for updating the cell state

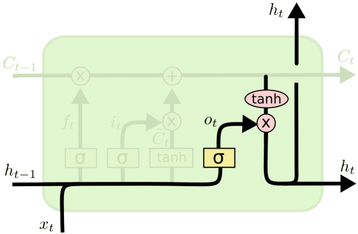
LSTM network: update Cell state



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- now the previous and current cell state are combined

LSTM network: output gate



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

- decide what to output
- the output is based on the computed cell state C_t, which weights the vector formed by the recurrence h_{t-1} and input x_t

Concluding remarks

- Recurrent layers are essential when sequential data is concerned
- It is paramount to format the input to as simple as possible configurations
- Example: one-hot vectors for words or characters.

Further reading

- Try to look for the Attention Networks: the idea is to let every step of an RNN pick information to look at from some larger collection of information.
- For example, a recurrent net to output caption of an image, it might pick a different part of the image to decide every word it outputs.

References

- Goodfellow, I., Bengio, Y., and Courville, A. Deep learning. MIT press, 2016.
- A. Karpathy. Understanding LSTM Networks. `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`
- C. Olah. Understanding LSTM Networks `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`