5. Sequence modeling with recurrent neural networks

5.4. Long Short-Term Memory Networks

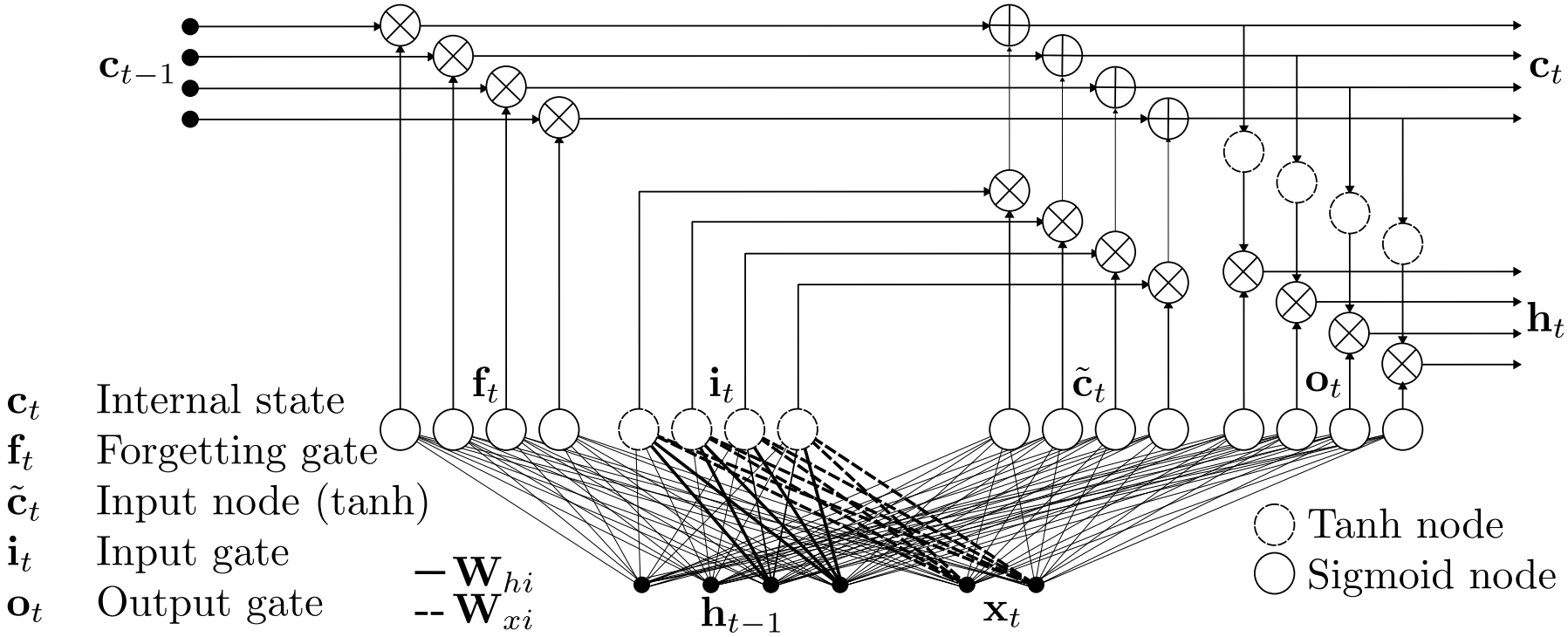
Manel Martínez-Ramón
Meenu Ajith
Aswathy Rajendra Kurup

Introduction

- Short-term dependencies can be stored by RNNs , but long-time ones cannot be captured due to the exploding and vanishing gradient phenomena in BPTT
- The LSTMs is a variation of RNNs that can handle long-term connections.
- Introduced by Sepp Hochreiter and Jürgen Schmidhuber in 1997 to tackle the problems posed by standard RNNs.
- The LSTM passes a linear internal state from one time instant to the next one with unit gain to propagate the gradient without exploding or vanishing.
- In the original LSTM paper by Hochreiter and Schmidhuber this mechanism is called constant error carousel (CEC).

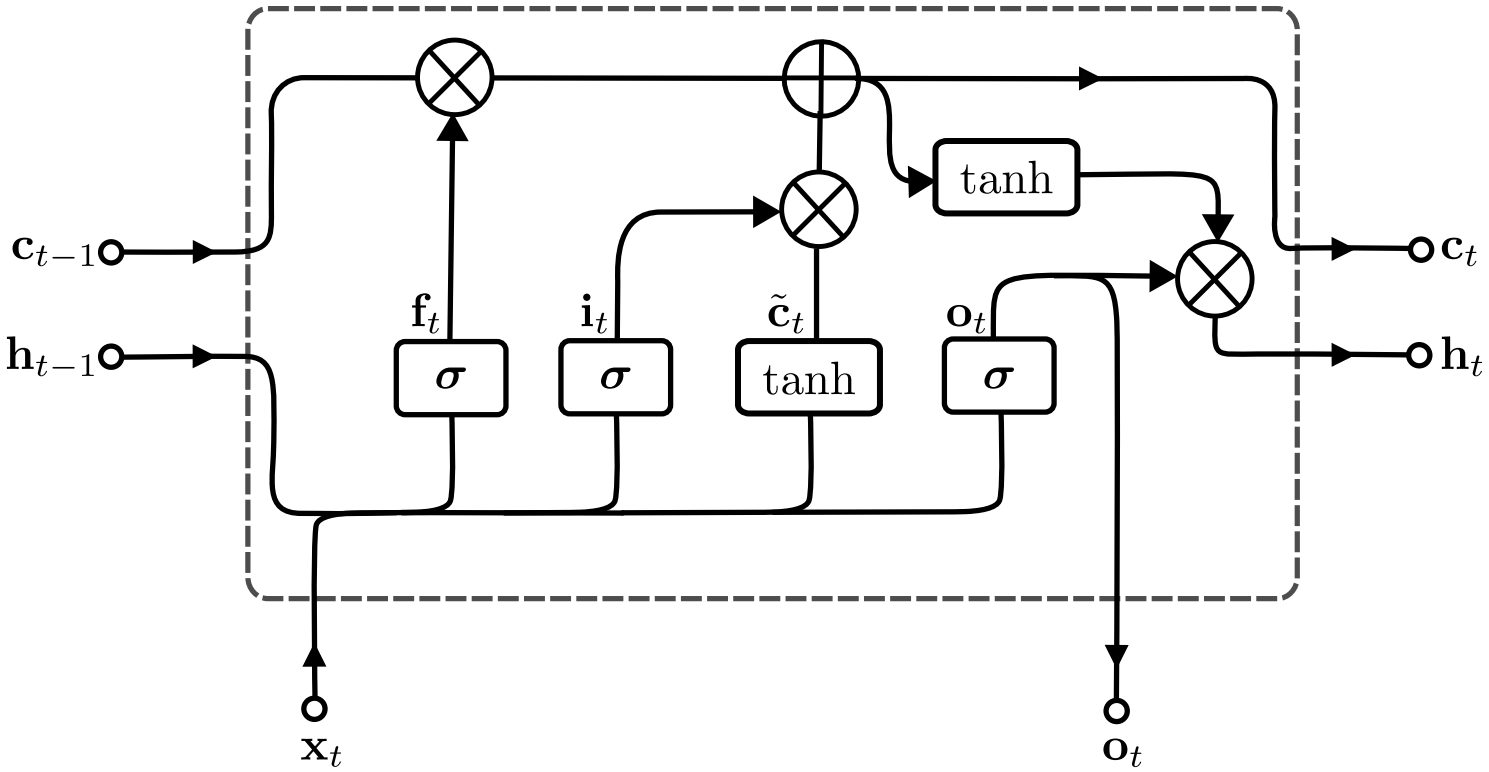
Structure

This really is an LSTM



Structure

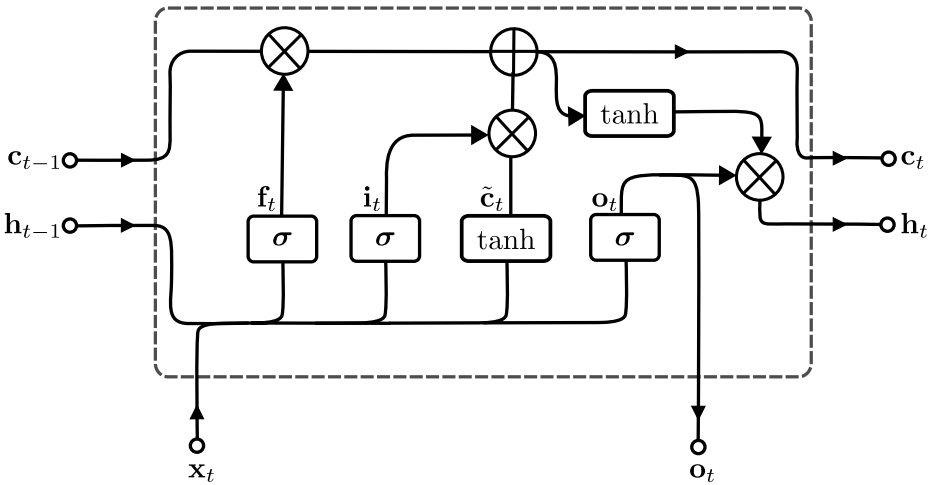
Compact (and more reasonable) structure representation



Structure

LSTM gates

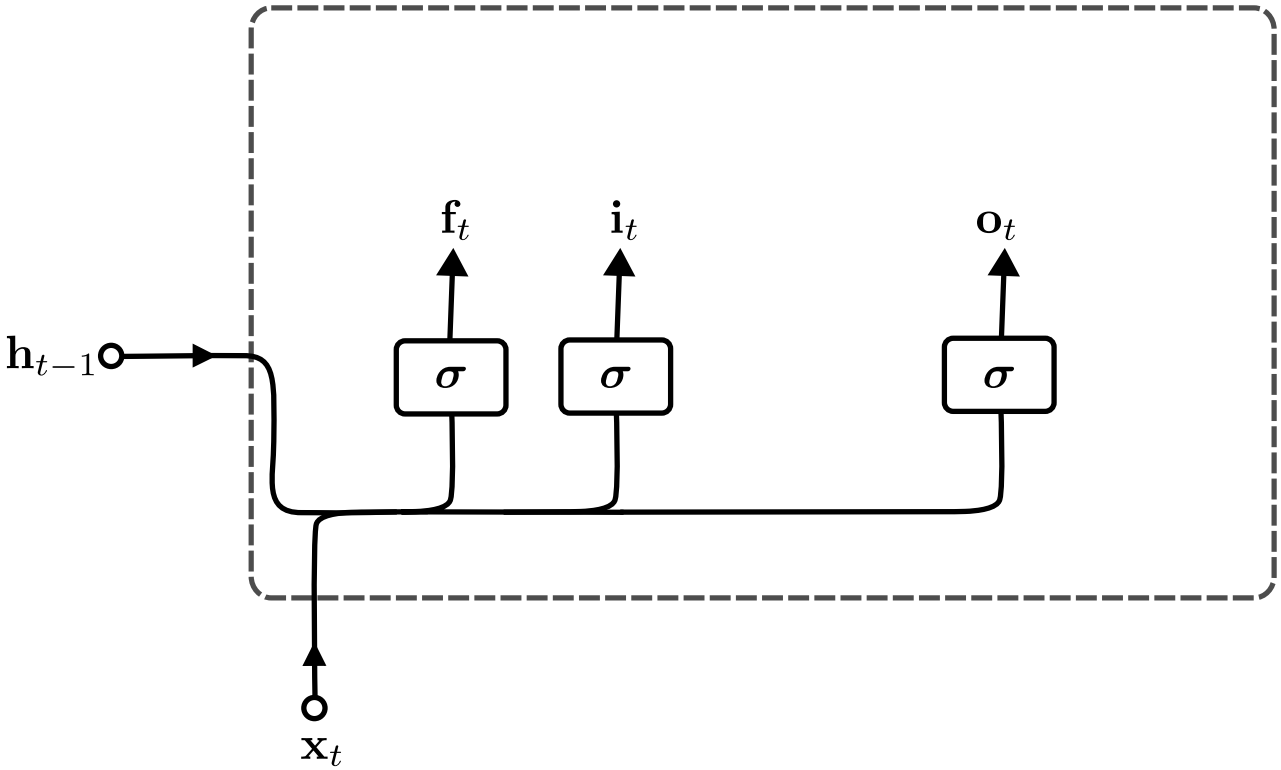
- The LSTM inputs are the internal state $\mathbf{c}_{t-1}$, the hidden state $\mathbf{h}_{t-1}$, and the present input pattern $\mathbf{x}_t$.
- The LSTM unit computes the values of the forgetting gate $\mathbf{f}_t$, the input gate $\mathbf{i}_t$, and the output gate $\mathbf{o}_t$.
- The expressions of these three activations are:



$$\begin{aligned}
\mathbf{f}_t &= \boldsymbol{\sigma}\left(\mathbf{W}_{xf}^{\top}\mathbf{x}_i + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f\right) \\
\mathbf{i}_t &= \boldsymbol{\sigma}\left(\mathbf{W}_{xi}^{\top}\mathbf{x}_i + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i\right) \\
\mathbf{o}_t &= \boldsymbol{\sigma}\left(\mathbf{W}_{xo}^{\top}\mathbf{x}_i + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o\right)
\end{aligned} \tag{1}$$

Structure

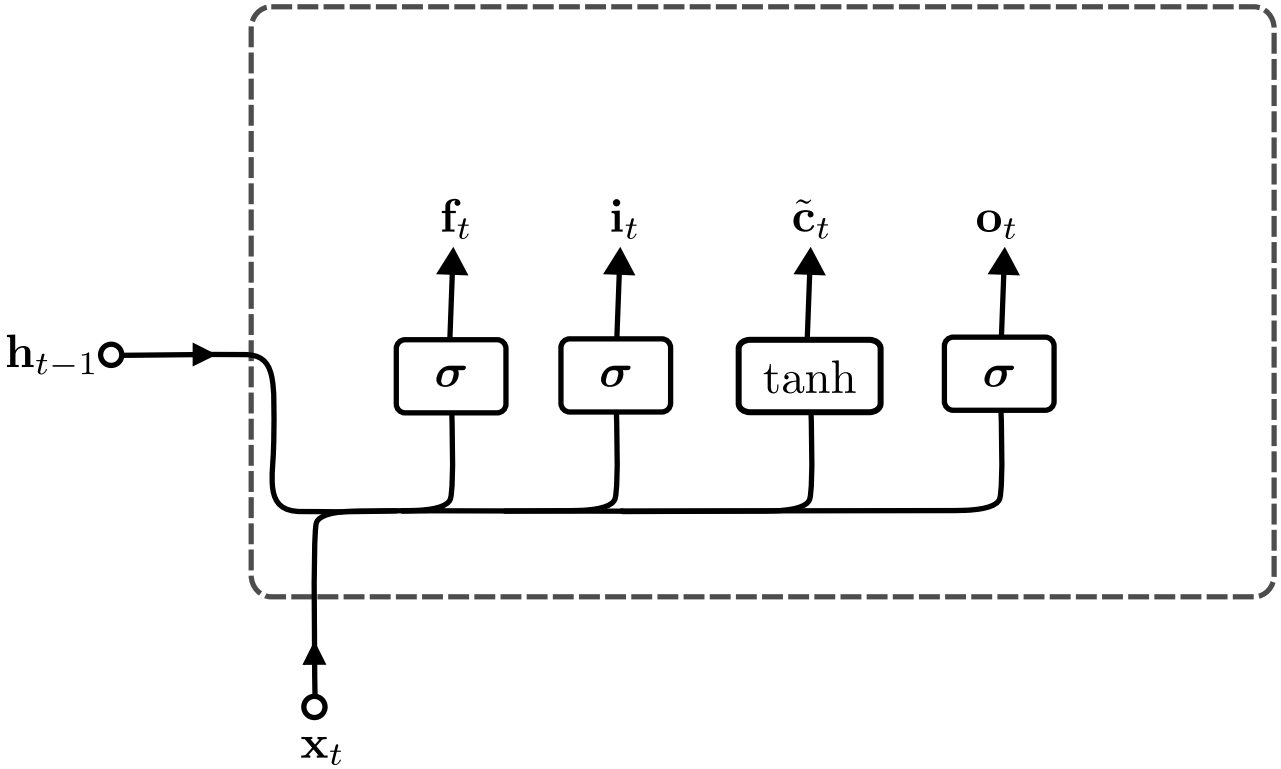
LSTM gates $\mathbf{f}_t$, $\mathbf{i}_t$, and $\mathbf{o}_t$



- $\mathbf{f}_t$ is trained to forget the previous internal state if necessary.
- $\mathbf{i}_t$ determines how much of the input must be used to modify $\mathbf{c}_t$.
- $\mathbf{o}_t$ will modulate $\mathbf{c}_t$ to compute state $\mathbf{h}_t$.

Structure

Input gate $\tilde{\mathbf{c}}_t$

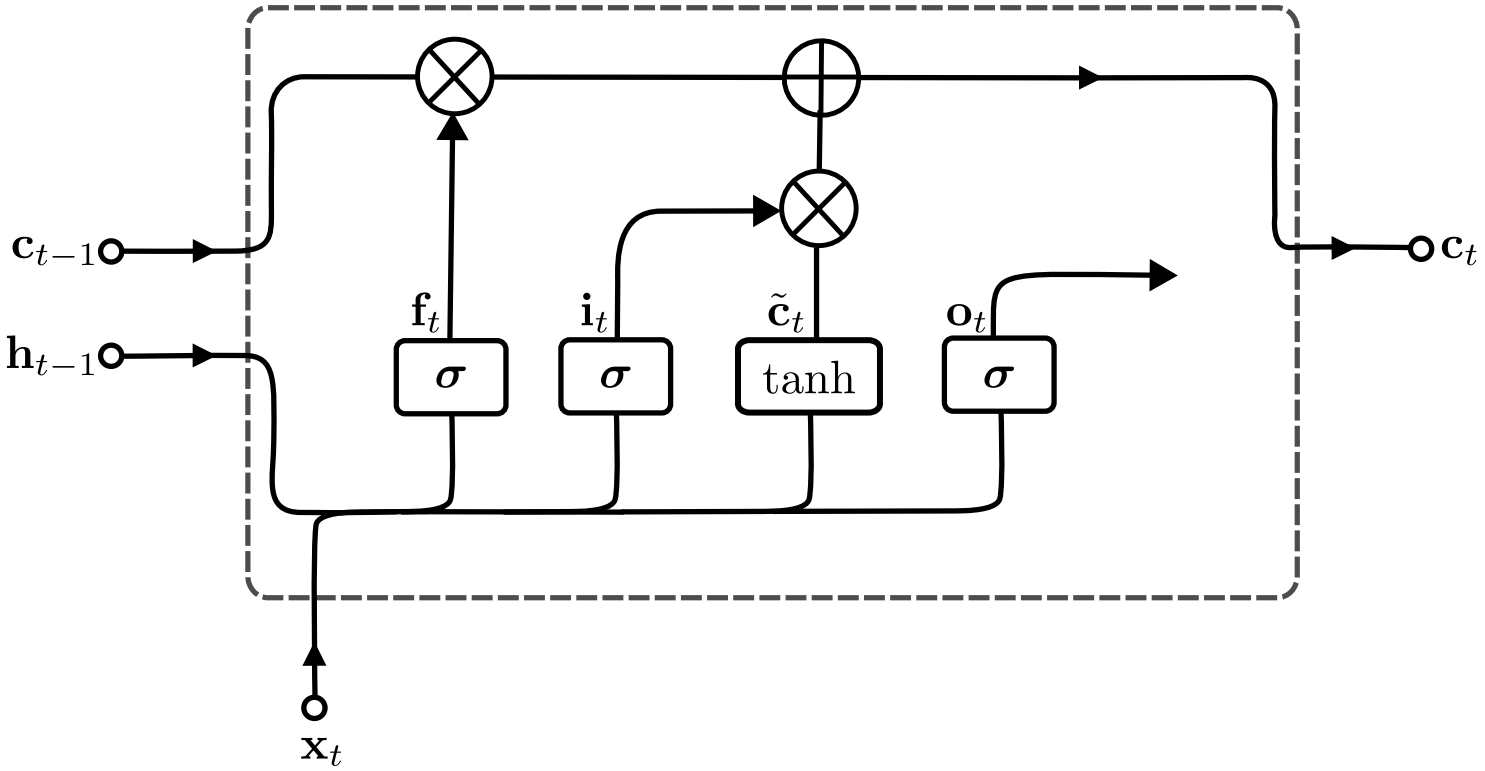


$$\tilde{\mathbf{c}}_t = \boldsymbol{\sigma}\left(\mathbf{W}_{xc}^{\top}\mathbf{x}_i + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right) \tag{2}$$

- $\tilde{\mathbf{c}}_t$ is used to update the previous internal state.

Structure

Internal state $\mathbf{c}_t$



- $\mathbf{f}_t$ is elementwise multiplied with internal state $\mathbf{c}_{t-1}$.
- If the outputs of $\mathbf{f}_t$ are low, a forgetting factor is applied to $\mathbf{c}_t$.
- At the same time, $\mathbf{i}_t$ elementwise multiplies $\tilde{\mathbf{c}}_t$, and the product is added to $\mathbf{c}_{t-1}$ state to produce $\mathbf{c}_t$.

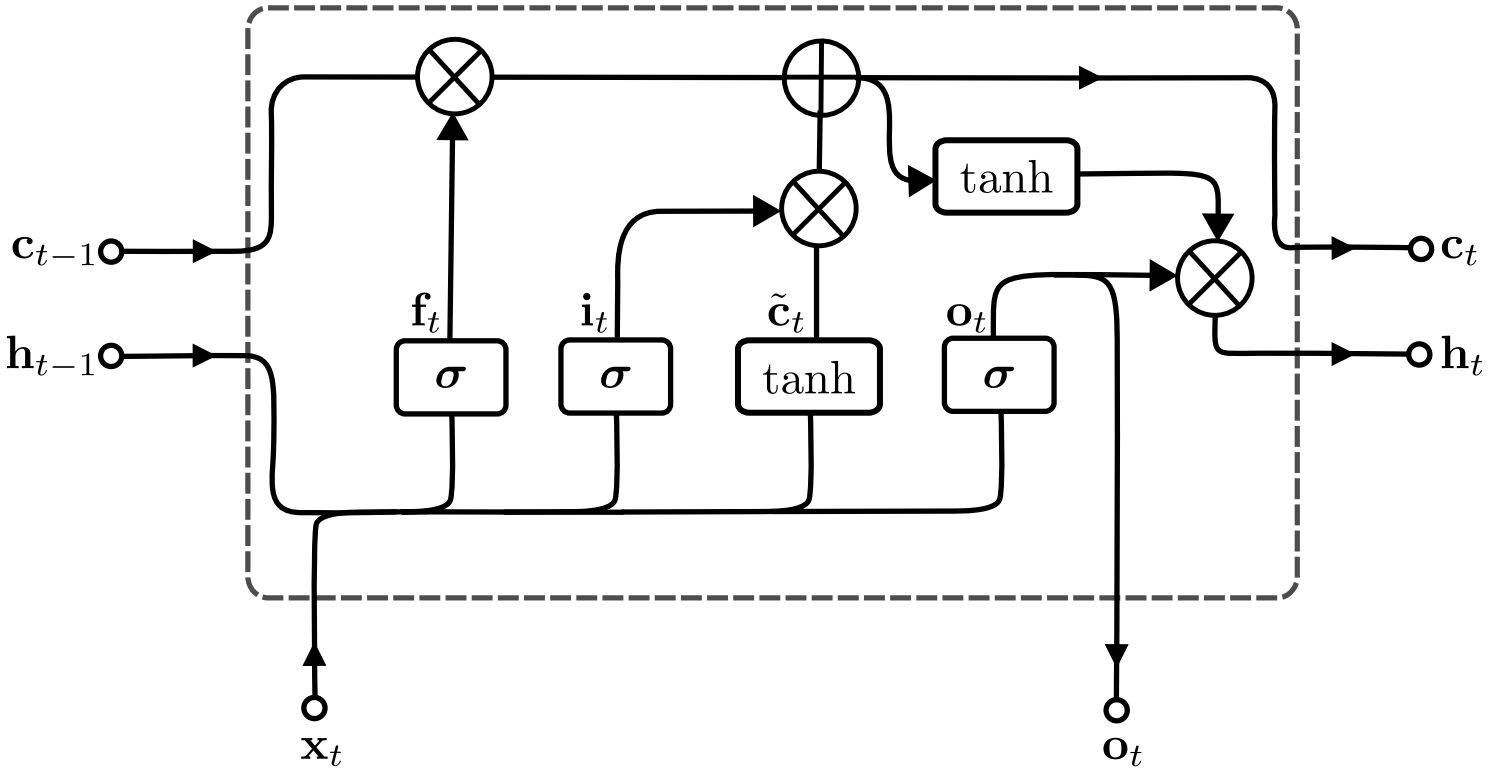
The operation to compute the internal state is

$$\mathbf{c}_t = \mathbf{c}_{t-1} \odot \mathbf{f}_t + \tilde{\mathbf{c}}_i \odot \mathbf{i}_t \tag{3}$$

- Notice the equivalence of the above structure with the unit gain self-feedback described as constant error carousel (CEC) in the original paper by Hochreiter and Schmidhuber.
- The backpropagation of the error through the internal state line is done without transformation through any nonlinear function derivative or weight matrix, therefore avoiding the phenomena of vanishing or exploding gradients.

Structure

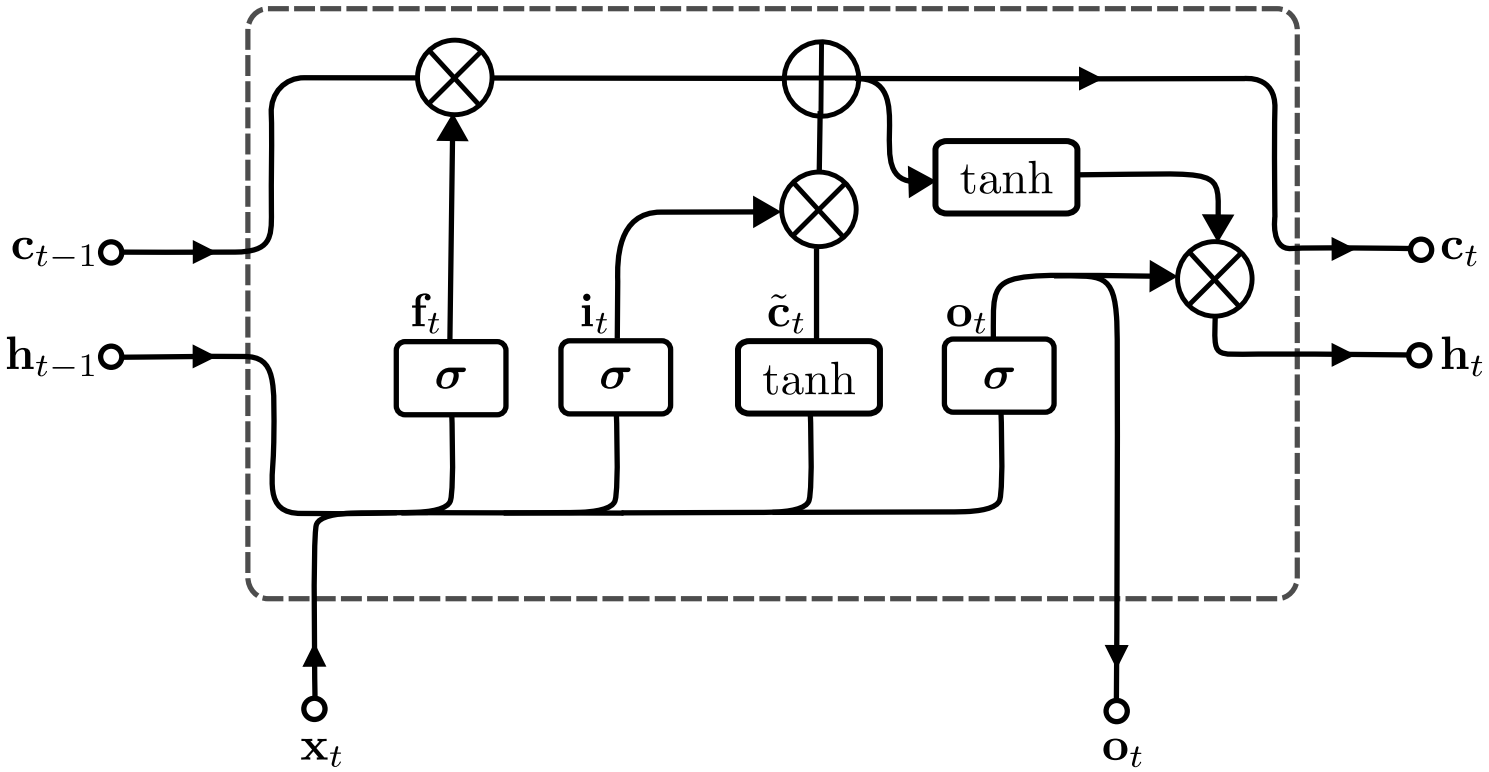
Hidden state $\mathbf{h}_t$



- The computation of the LSTM ouput gate is

$$\mathbf{h}_t = \tanh(\mathbf{c}_t) \odot \mathbf{o}_t \tag{4}$$

Backpropagation



- The backpropagation through time is computed essentially as in an RNN.
- It can be done by following the paths in the direction opposite to the arrows

Backpropagation

- Gradient with respect to the hidden state $\mathbf{h}_t$

$$\nabla_{\mathbf{h}_t} J_{ML} = \mathbf{h}_t - \mathbf{y}_t = \boldsymbol{\delta}_t \tag{5}$$

- Gradient with respect to the output gate $\mathbf{o}_t$

$$\nabla_{\mathbf{o}_t} J_{ML} = \boldsymbol{\delta}_t \odot \tanh(\mathbf{c}_t) \tag{6}$$

- Gradient with respect to internal state $\mathbf{c}_t$

$$\nabla_{\mathbf{c}_t} J_{ML} = \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \tag{7}$$

$$\nabla_{\mathbf{c}_{t-1}} J_{ML} = \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{f}_t \tag{8}$$

Backpropagation

- Gradient with respect to the input gate $\mathbf{i}_t$

$$\nabla_{\mathbf{i}_t} J_{ML} = \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \tilde{\mathbf{c}}_t \tag{9}$$

- Gradient with respect to the input state $\tilde{\mathbf{c}}_t$

$$\nabla_{\tilde{\mathbf{c}}_t} J_{ML} = \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{i}_t \tag{10}$$

- Gradient with respect to the forgetting gate $\mathbf{f}_t$

$$\nabla_{\mathbf{f}_t} J_{ML} = \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_{t-1}) \tag{11}$$

All these gradients are used to compute the gradient with respect to the weights. They are 8 matrices and 4 biases.

Backpropagation

- Gradient with respect to output gate weights:

$$\begin{aligned}\nabla_{\mathbf{W}_{xo}} J_{ML} &= \boldsymbol{\delta}_t \odot \tanh(\mathbf{c}_t) \odot \boldsymbol{\sigma}'(\mathbf{z}_o) \odot \mathbf{x}_t \\ \nabla_{\mathbf{W}_{ho}} J_{ML} &= \boldsymbol{\delta}_t \odot \tanh(\mathbf{c}_t) \odot \boldsymbol{\sigma}'(\mathbf{z}_o) \odot \mathbf{h}_{t-1} \\ \nabla_{\mathbf{b}_o} J_{ML} &= \boldsymbol{\delta}_t \odot \tanh(\mathbf{c}_t) \odot \boldsymbol{\sigma}'(\mathbf{z}_o)\end{aligned} \tag{12}$$

- Gradient with respect to forgetting gate weights:

$$\begin{aligned}\nabla_{\mathbf{W}_{xf}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f) \odot \mathbf{x}_t \\ \nabla_{\mathbf{W}_{hf}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f) \odot \mathbf{h}_{t-1} \\ \nabla_{\mathbf{b}_f} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f)\end{aligned} \tag{13}$$

Backpropagation

- Gradient with respect to the input gate weights:

$$\begin{aligned}
\nabla_{\mathbf{W}_{xi}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \tilde{\mathbf{c}}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_g) \odot \mathbf{x}_t \\
\nabla_{\mathbf{W}_{hi}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \tilde{\mathbf{c}}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_g) \odot \mathbf{h}_{t-1} \\
\nabla_{\mathbf{b}_{bi}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \tilde{\mathbf{c}}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_g)
\end{aligned} \tag{14}$$

- Gradient with respect to the input state weights:

$$\begin{aligned}
\nabla_{\mathbf{W}_{xc}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f) \odot \mathbf{x}_t \\
\nabla_{\mathbf{W}_{hc}} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f) \odot \mathbf{h}_{t-1} \\
\nabla_{\mathbf{b}_c} J_{ML} &= \boldsymbol{\delta}_t \odot \mathbf{o}_t \odot \tanh'(\mathbf{c}_t) \odot \mathbf{c}_t \odot \boldsymbol{\sigma}'(\mathbf{z}_f)
\end{aligned} \tag{15}$$