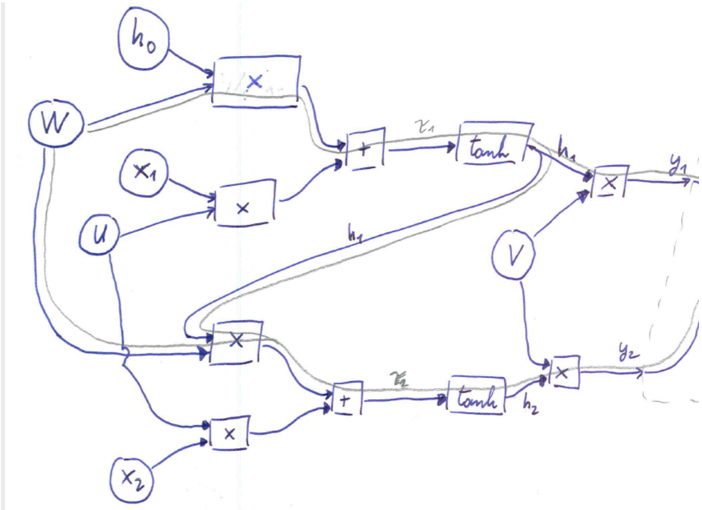
Recurrent Neural Networks - Tutorial Solutions

Matthias Rowold

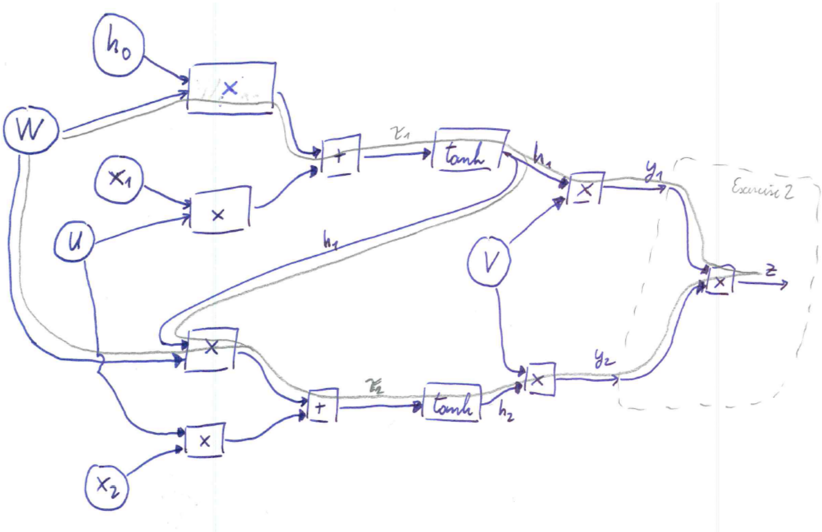
January 19, 2023

1 Written Exercises

1.1 Exercise 1



1.2 Exercise 2



•

- $$\begin{aligned}
\frac{dz}{dW} &= \underbrace{\frac{\partial z}{\partial y_1}}_{y_2}\frac{dy_1}{dW} + \underbrace{\frac{\partial z}{\partial y_2}}_{y_1}\frac{dy_2}{dW} \\
\frac{dy_1}{dW} &= \frac{\partial y_1}{\partial h_1}\frac{dh_1}{dW} = V\frac{dh_1}{dW} \\
\frac{dh_1}{dW} &= \frac{\partial h_1}{\partial \tau_1}\frac{d\tau_1}{dW} = (1-\tanh(\tau_1)^2)h_0 \\
\frac{dy_2}{dW} &= \frac{\partial y_2}{\partial h_2}\frac{dh_2}{dW} = V\frac{dh_2}{dW} \\
\frac{dh_2}{dW} &= \frac{\partial h_2}{\partial \tau_2}\frac{d\tau_2}{dW} = (1-\tanh(\tau_2)^2)\frac{d\tau_2}{dW} \\
\frac{d\tau_2}{dW} &= \frac{\partial \tau_2}{\partial W}\frac{dW}{dW} + \frac{\partial \tau_2}{\partial h_1}\frac{dh_1}{dW} = h_1 + W\frac{dh_1}{dW} = h_1 + W(1-\tanh(\tau_1)^2)h_0
\end{aligned}$$

  In total:

  $$\frac{dz}{dW} = y_2 V h_0(1-\tanh(\tau_1)^2)) + y_1 V(t-\tanh(\tau_2)^2)\left[h_1 + W h_0(1-\tanh(\tau_1)^2)\right] \tag{1}$$

- We can see in (1), that we need $y_1, y_2, V, W, h_0, h_1, \tau_1, \tau_2$. V and W are parameters, so we need additional storage for $y_1, y_2, h_0, h_1, \tau_1, \tau_2$.
- For each additional time-step we need to store τ_i, h_i, and y_i. This makes $98 \cdot 3 = 294$ more variables.
- As the *tanh*() itself appears in the derivative of *tanh*() we can save one variable per time-step by re-using the result for h_i instead of $\tanh(\tau_i)$. This makes $98 \cdot 2 = 196$ more variables.

1.3 Exercise 3

Even though x_1 and x_2 do not appear explicitly in (1), $\frac{dz}{dW}$ depends on them as y_1, y_2, h_1, h_2 depend on x_1 and x_2. The values of x_1 and x_2 are only needed in the forward pass. The values for y_1, y_2, h_1, h_2 are stored for the backward pass.

1.4 Exercise 4

- Gradient clipping
- Option 1 (formula /algorithm):

$\mathbf{g} \leftarrow \frac{\partial L}{\partial \theta}$
if $\|\mathbf{g}\| \geq \nu$ **then**
  $\mathbf{g} \leftarrow \frac{\nu}{\|\mathbf{g}\|}\mathbf{g}$
end if

- Option 2 (text):
  The norm of the gradient is reduced to a predefined threshold. The direction of the gradient remains the same.

1.5 Exercise 5

There are two hidden states. h_t depends on the previous h_{t-1}, while k_t depends on the next k_{t+1}.

- Bidirectional RNN
- Answer must meet the two requirements: 1.) sequential data and 2.) future data available
- E.g. handwriting recognition, video analysis

1.6 Exercise 6

- With parameter set (1):

$$
\begin{aligned}
h_1 &= \tanh(20 \cdot 0.5 + 1 \cdot 0 + 1) = \tanh(11) = 1 = y_1 \\
h_2 &= \tanh(20 \cdot (-1) + 1 \cdot 1 + 1) = \tanh(-18) = -1 = y_2 \\
L &= 1^2 + (-1)^2 = 2
\end{aligned}
$$

- With parameter set (2):

$$
\begin{aligned}
h_1 &= \tanh(1 \cdot 0.5 + 1 \cdot 0 + 0) = 0.46 = y_1 \\
h_2 &= \tanh(1 \cdot (-1) + 1 \cdot 1 + 0) = -0.49 = y_2 \\
L &= 0.46^2 + (-0.49)^2 = 0.45
\end{aligned}
$$

- Parameter set (1) leads to the saturation of tanh() and the gradient vanishes. L will almost not change, if we slightly change a, b, or c. Therefore, we should choose parameter set (2) or parameters in the same order of magnitude.

1.7 Exercise 7

Disadvantage:

- The gradient of the loss function with respect to the recurrent parameters is now biased. The short sequences do not allow to represent long-term dependencies and the gradients for short sequences to not match the true gradients for the whole sequences that potentially show long-term dependencies.

Advantages:

- Less memory required (RAM, GPU memory)
- Faster computation of gradients
- The gradient is better conditioned (less prone to exploding / vanishing gradient)