KiSTi
www.kisti.re.kr
한국과학기술정보연구원
Korea Institute of Science and Technology Information

UST
과학기술연합대학원대학교
UNIVERSITY OF SCIENCE & TECHNOLOGY

Seminar II: Deep Learning-based Natural Language Processing

“On the difficulty of training Recurrent Neural Networks”
Paper review

LSTM Networks (The gated copy trick)

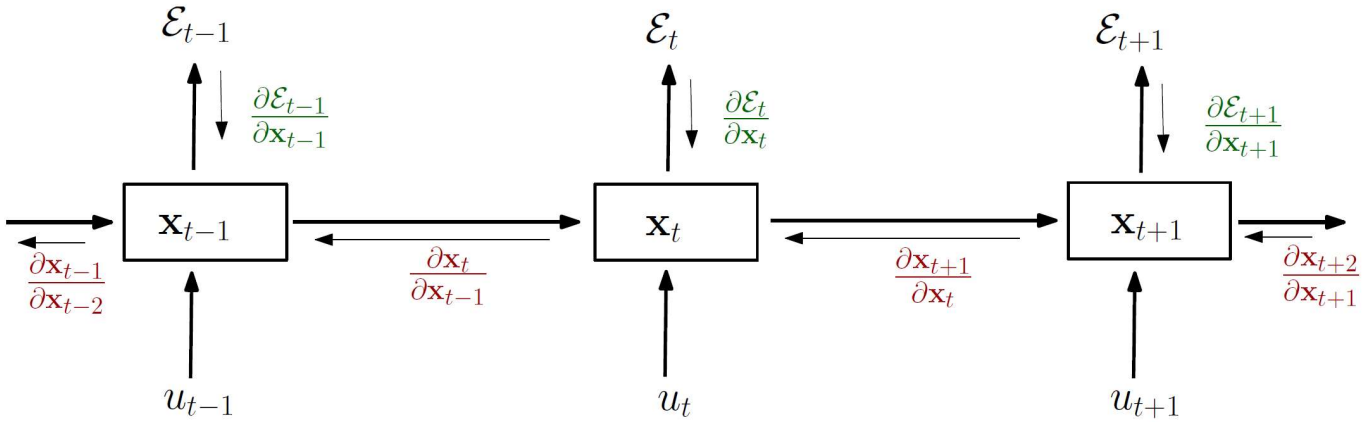
Contents

Introduction. Training recurrent networks

Hidden state

$$x_t = f(W_{rec}x_{t-1} + W_{in}u_t + b)$$

Backpropagation Through Time (BPTT)



Unrolling recurrent neural networks in time by creating a copy of the model for each time step.
Source: On the difficulty of training Recurrent Neural Networks

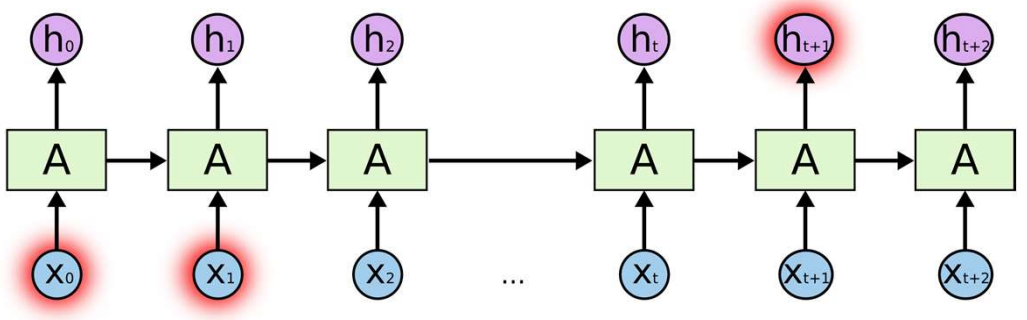
Introduction. Exploding and Vanishing Gradients

$x_t = f(W_{rec}x_{t-1} + W_{in}u_t + b)$

Shakespeare dataset has 100,000 character

Backpropagating through 100,000 character is impossible

Long sequence means long backpropagation chain!



Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Introduction. Key Contribution

OBJECTIVE

To improve **the understanding of the underlying issues by exploring these problems from**

1. **Analytical perspective**
2. **Geometric perspective**
3. **Dynamical systems perspective**

To improve **previous solutions to deal with these problems**

1. **Clipping strategy (exploding gradient)**
2. **Soft constraint (vanishing gradient)**

Methodology – Analytical perspective

$$x_t = f(W_{rec}x_{t-1} + W_{in}u_t + b)$$

The condition for exploding gradients:

$$\lambda_1 > \frac{1}{\gamma}$$

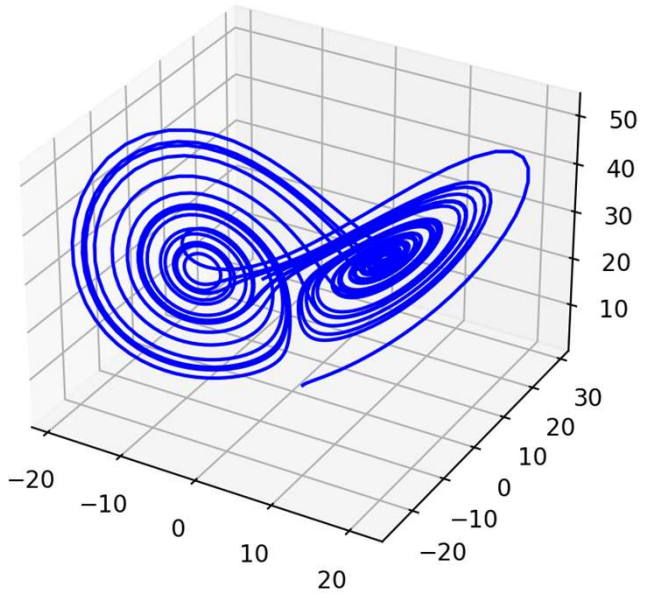
The condition for vanishing gradients:

$$\lambda_1 < \frac{1}{\gamma}$$

where λ_1 is the absolute value of the largest eigenvalue of the recurrent weight matrix W_{rec}

For *tanh* we have $\gamma = 1$ while for *sigmoid* we have $\gamma = 1/4$

Methodology – Dynamical systems perspective



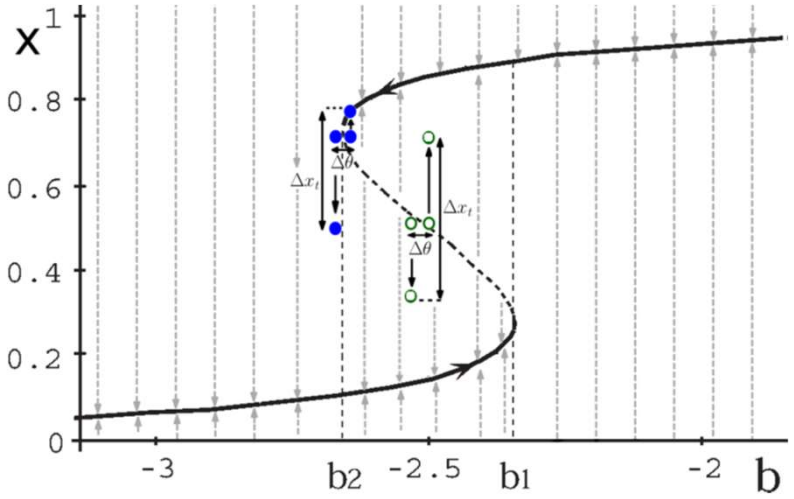
Lorenz 3D phase-space representation

Edward Lorenz accidentally discovered chaotic behaviour in the following dynamical system.

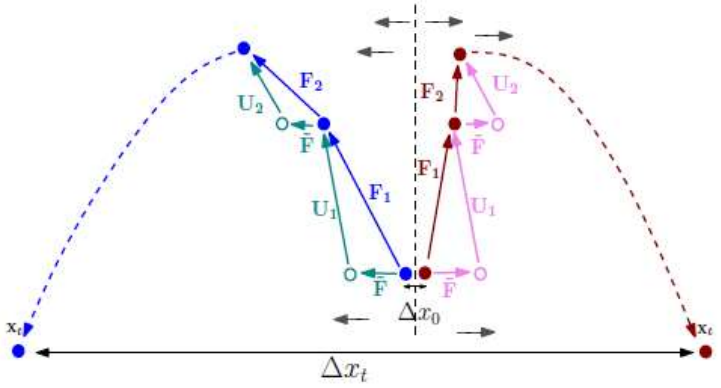
next_x = x + (s * (y - x)) * Dt
next_y = y + (r * x - y - x * z) * Dt
next_z = z + (x * y - b * z) * Dt

In this experiment, the parameters are:
s = 10, r = 28, b = 8/3, Dt = 0.01, and
x = y = z = 1

Methodology – Dynamical systems perspective



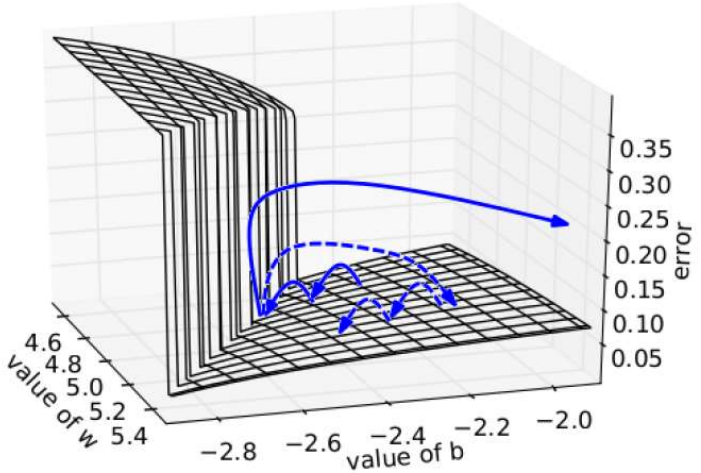
Bifurcation diagram of a single hidden unit RNN (with fixed recurrent weight of 5.0 and adjustable bias b
Source: On the difficulty of training Recurrent Neural Networks



The dotted vertical line represents the boundary between basins of attraction, and the straight dashed arrow the direction of the map F on each side of the boundary.
Source: On the difficulty of training Recurrent Neural Networks

Methodology – Geometric perspective

$$x_t = wf(x_{t-1}) + b$$



We plot the error surface of a single hidden unit recurrent network, highlighting the existence of high curvature walls
Source: On the difficulty of training Recurrent Neural Networks

Methodology – Scaling down the gradients

Algorithm 1 Pseudo-code for norm clipping the gradients whenever they explode

$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$
if $\|\hat{\mathbf{g}}\| \geq threshold$ **then**
  $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$
end if

Rescale gradients whenever they go over a threshold

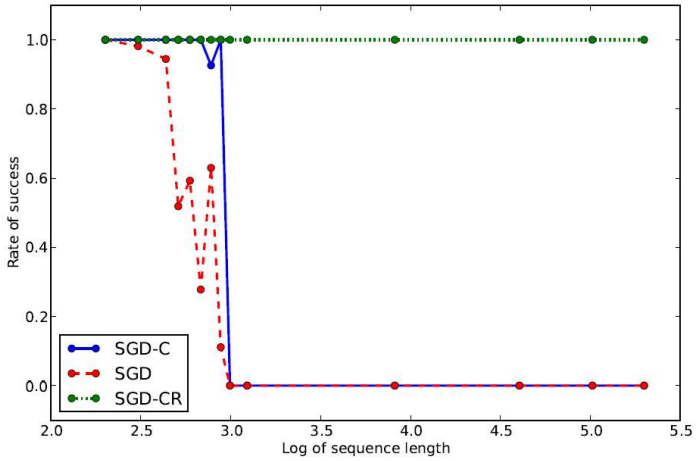
Methodology – Vanishing gradient regularization

$$\Omega = \sum_k \Omega_k = \sum_k \left(\frac{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k} \right\|}{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \right\|} - 1 \right)^2$$

$$\begin{aligned} \frac{\partial^+ \Omega}{\partial \mathbf{W}_{rec}} &= \sum_k \frac{\partial^+ \Omega_k}{\partial \mathbf{W}_{rec}} \\ &= \sum_k \frac{\partial^+ \left(\frac{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_k)) \right\|}{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \right\|} - 1 \right)^2}{\partial \mathbf{W}_{rec}} \end{aligned}$$

Vanishing gradient is not easy to deal with.
Input from a distant past is forgotten! (Memory decay)
The problem is the model (Need to change this model, maybe LSTM)

Results and Discussions



Rate of success for solving the temporal order problem versus log of sequence length
Source: On the difficulty of training Recurrent Neural Networks

Data set	Data fold	SGD	SGD+C	SGD+CR
Piano-midi.de	train	6.87	**6.81**	7.01
	test	7.56	7.53	**7.46**
Nottingham	train	3.67	**3.21**	3.24
	test	3.80	3.48	**3.46**
MuseData	train	8.25	6.54	**6.51**
	test	7.11	7.00	**6.99**

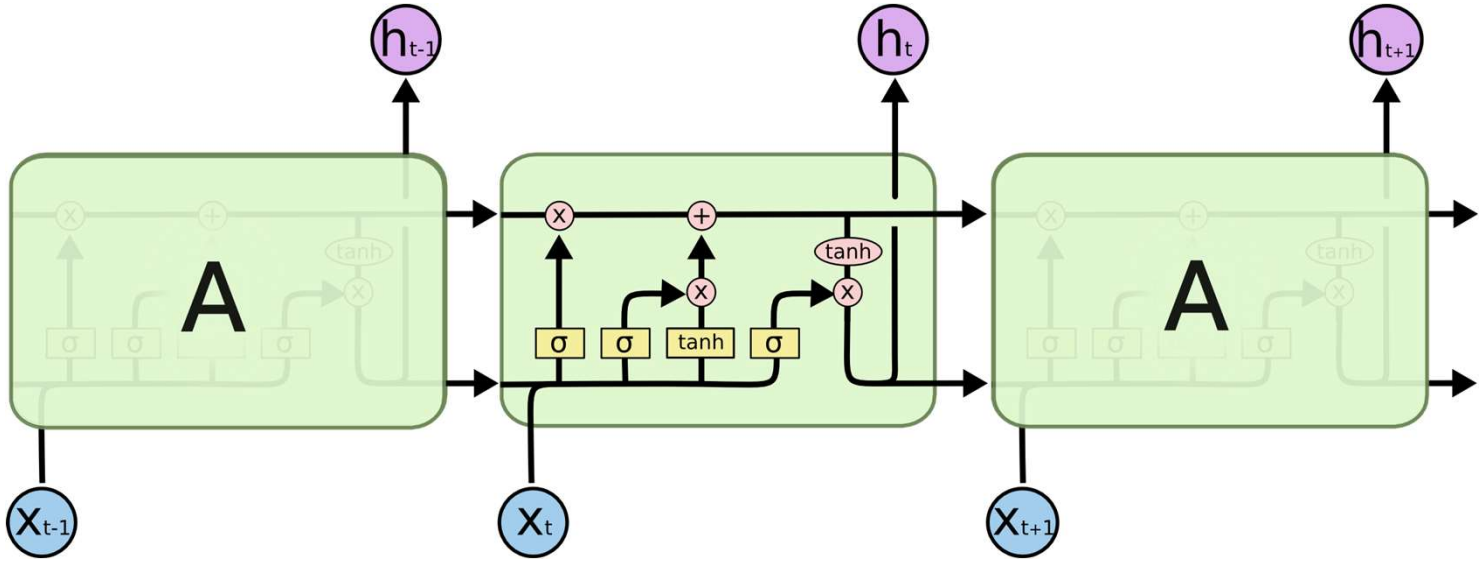
Results on polyphonic music prediction in negative log likelihood per time step. Lower is better

Data set	Data fold	SGD	SGD+C	SGD+CR
1 step	train	1.46	**1.34**	1.36
	test	1.50	1.42	**1.41**
5 steps	train	N/A	3.76	**3.70**
	test	N/A	3.89	**3.74**

Results on the next character prediction task in entropy (bits/character)

GRU and LSTM

LSTM and GRU create the next hidden state by copying the previous hidden state and then adding or removing information as necessary.
This mechanisms for adding and removing information are called gates.



Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

GRU

- Standard RNN computes hidden layer at next time step directly: $h_t = f\left(W^{(hh)}h_{t-1} + W^{(hx)}x_t\right)$
- GRU first computes an update **gate** (another layer) based on current input word vector and hidden state

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$

- Compute reset gate similarly but with different weights

$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$

Source: https://cs224d.stanford.edu/lectures/CS224d-Lecture9.pdf

GRU

- Update gate $z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$
- Reset gate $r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$
- New memory content: $\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$
  If reset gate unit is ~0, then this ignores previous memory and only stores the new word information
- Final memory at time step combines current and previous time steps: $h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$

Source: https://cs224d.stanford.edu/lectures/CS224d-Lecture9.pdf

LSTM

- We can make the units even more complex
- Allow each time step to modify
  - Input gate (current cell matters) $i_t = \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1}\right)$
  - Forget (gate 0, forget past) $f_t = \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1}\right)$
  - Output (how much cell is exposed) $o_t = \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1}\right)$
  - New memory cell $\tilde{c}_t = \tanh\left(W^{(c)}x_t + U^{(c)}h_{t-1}\right)$
- Final memory cell: $c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$
- Final hidden state: $h_t = o_t \circ \tanh(c_t)$

Source: https://cs224d.stanford.edu/lectures/CS224d-Lecture9.pdf

Conclusions

- Exploding gradient -> gradient clipping (Popular remedy)
- Vanishing gradient -> Regularization term

- The problem is the model itself (that is why LSTM is devised)