Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

2025-07-08

ICLR 2014

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio

Content

Introduction

- What is a Recurrent Neural Network?
- Types of a Recurrent Neural Network
- Long-Term Dependency Problem

Gated Recurrent Neural Networks

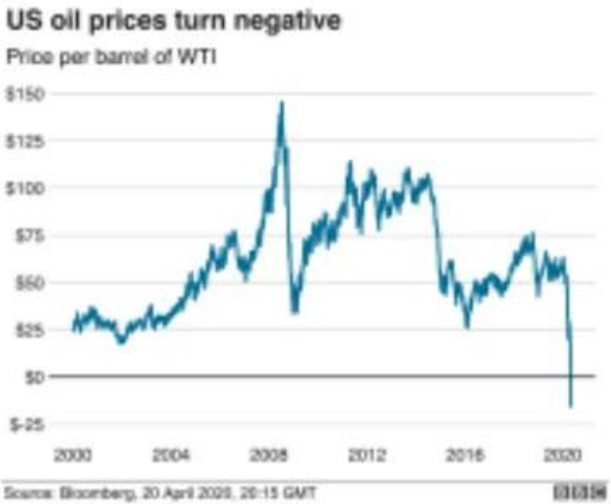
- Long Short-Term Memory (LSTM)
- Gated Recurrent Unit (GRU)

Experiments

Conclusion

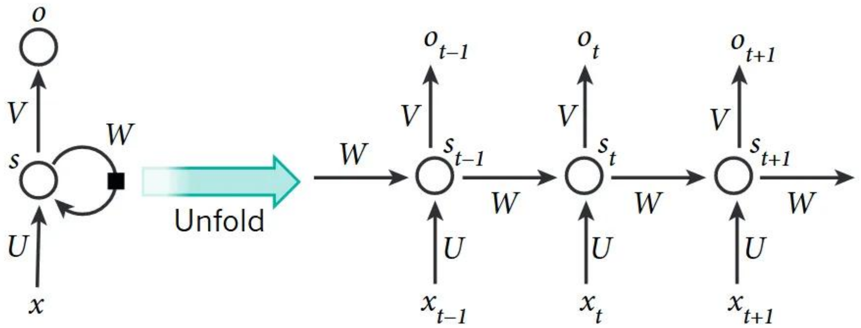
What is a Recurrent Neural Network?

- An extension of a conventional feedforward neural network, which is able to handle a variable-length sequence input
  - Text, Music, Stock price, …



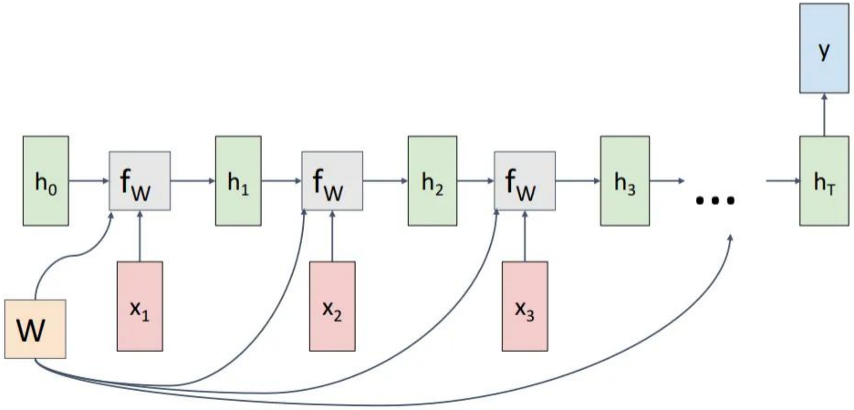
What is a Recurrent Neural Network?

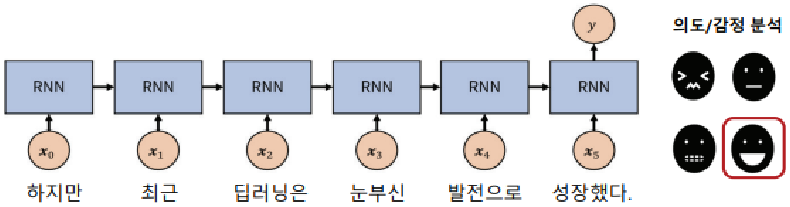
- $x = (x_1, x_2, \ldots, x_T)$
  - Input sequence
- $h_t = g(Ux_t + Wh_{t-1})$
  - Recurrent hidden state (state vector)
  - Contain information about the history of all the past elements of the sequence
- $y = (y_1, y_2, \ldots, y_T)$
  - Output (optional)



Types of Recurrent Neural Networks

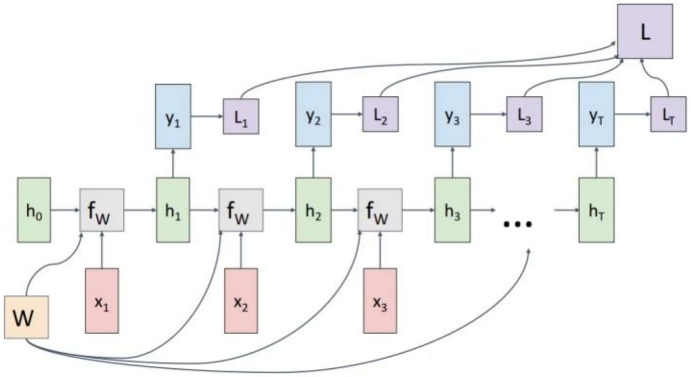
- **Many to One**
  - $e.g.$, Emotional analysis





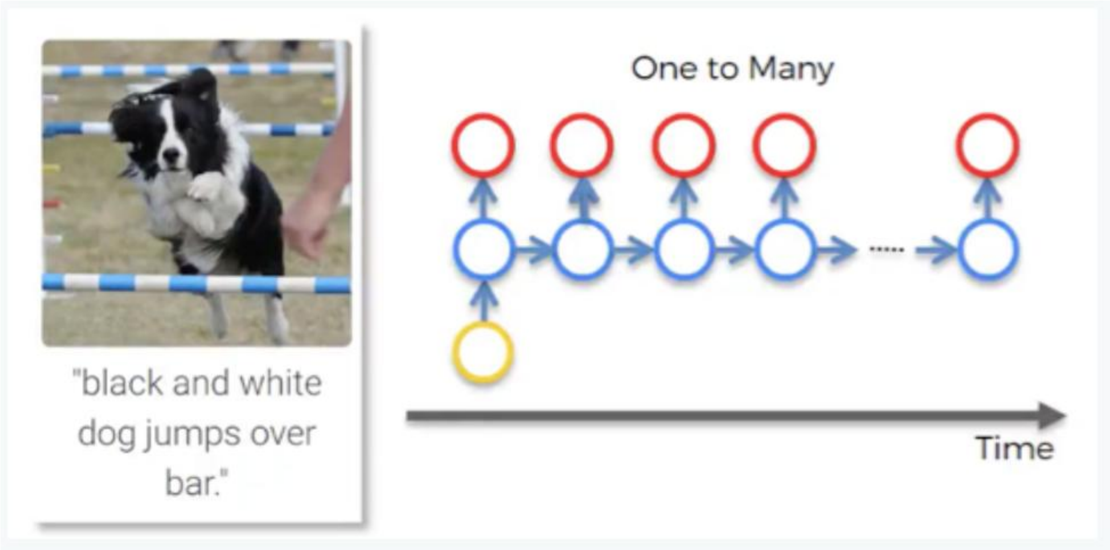
Types of Recurrent Neural Networks

- ☐ **Many to Many**
  - $e.g.$, Named Entity Recognition



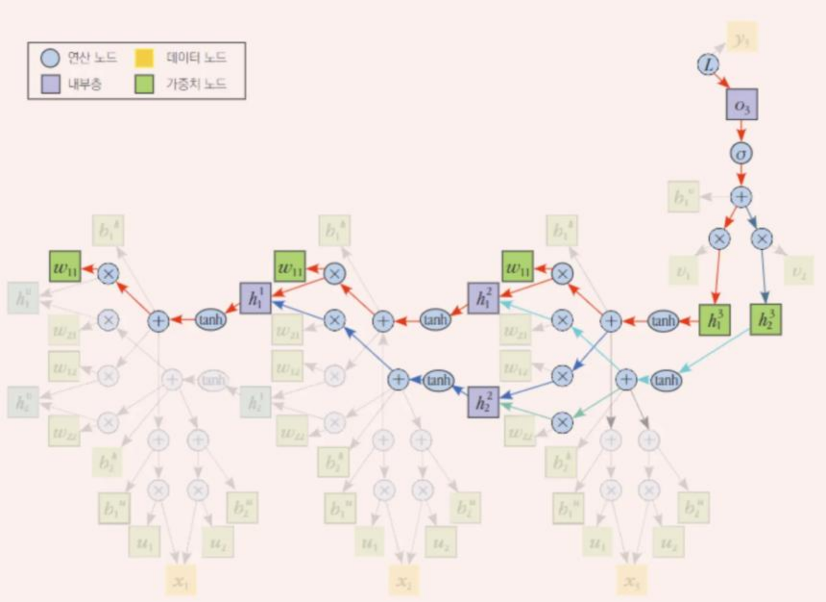
Types of Recurrent Neural Networks

- ☐ **One to Many**
  - $e.g.$, Image Captioning



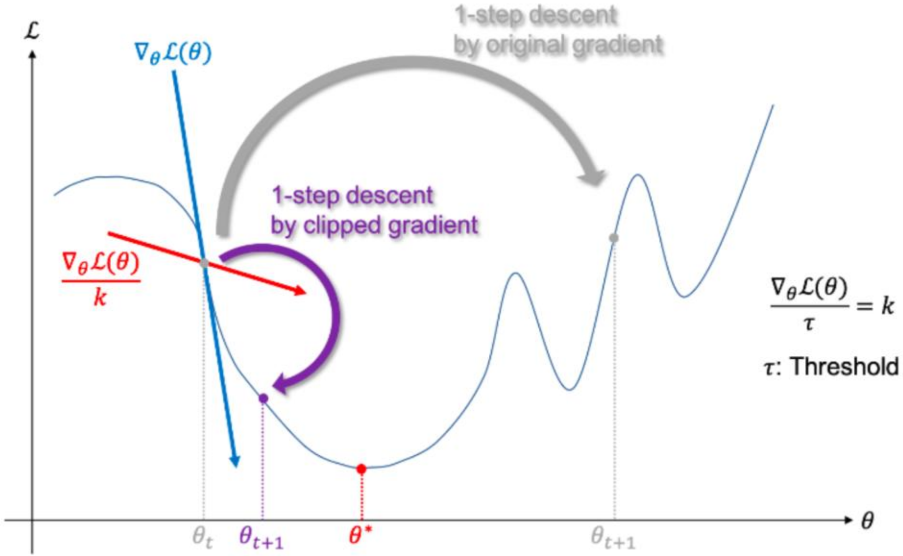
Long-Term Dependency Problem

- ☐ **Difficult to train RNNs to capture Long-Term Dependencies**
  - ■ The gradients tend to either vanish or explode



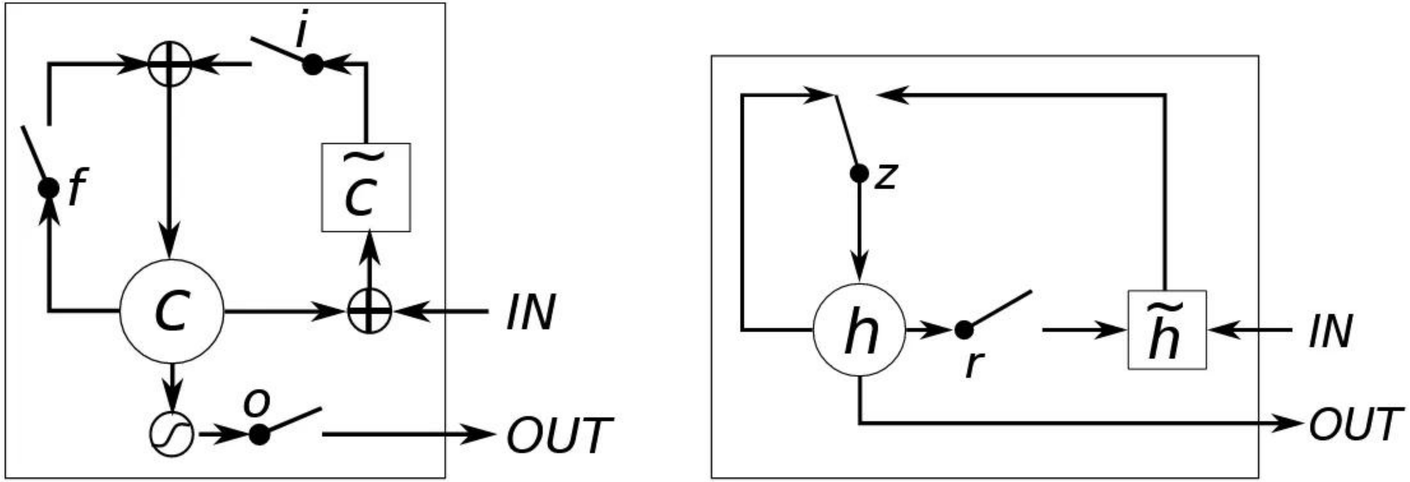
Long-Term Dependency Problem

- **Solution**
  - Devise a better learning algorithm
    - $e.g.$, Clip gradients above a given threshold



Long-Term Dependency Problem

- **Solution**
  - Use gate units
    - Perform well in tasks that require capturing long-term dependencies

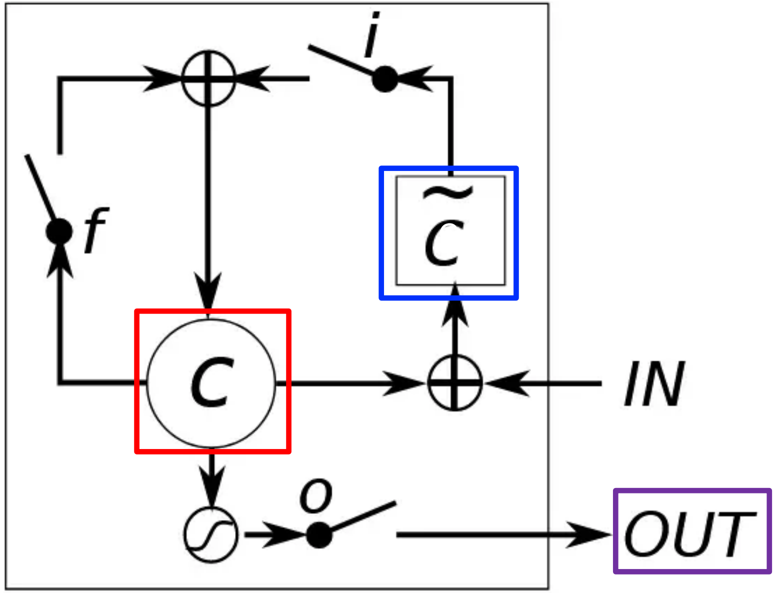


(a) Long Short-Term Memory (b) Gated Recurrent Unit

Long Short-Term Memory (LSTM)

Memory cell

- Explicitly store information up to this time
- Updated by
  - 1) Partially forgetting the existing memory
  - 2) Adding a new memory content
- $c_t^j = f_t^j c_{t-1}^j + i_t^j \tilde{c}_t^j$
- $\tilde{c}_t^j = \tanh(W_c x_t + U_c h_{t-1})^j$



Output

- The activation of the LSTM unit
- $h_t^j = o_t^j \tanh(c_t^j)$

Long Short-Term Memory (LSTM)

Forget gate

- Modulate **the extent to which the existing memory is forgotten**
- $f_t^j = \sigma\left(W_f x_t + U_f h_{t-1} + V_f c_{t-1}\right)^j$

Input gate

- Modulate **the degree to which the new memory content is added to the memory cell**
- $i_t^j = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1})^j$

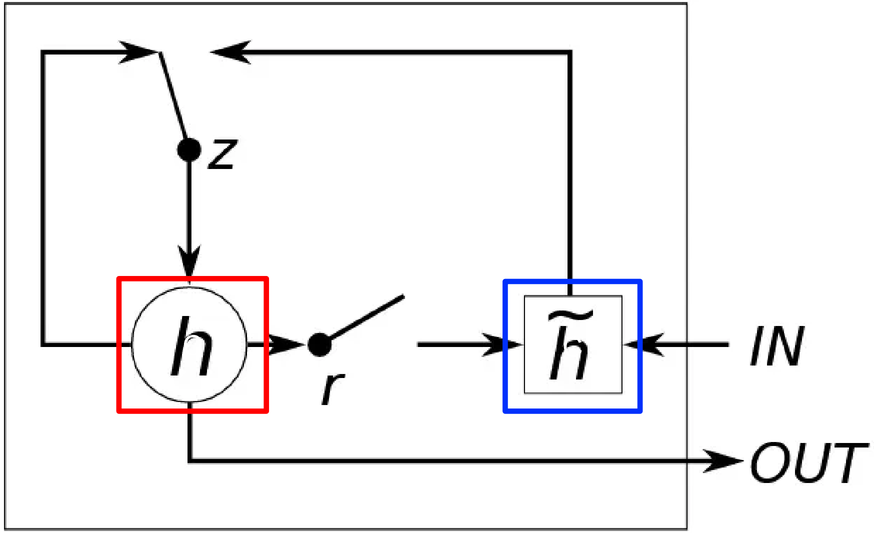
Output gate

- Modulate **the amount of memory content exposure**
- $o_t^j = \sigma(W_o x_t + U_o h_{t-1} + V_o c_{t-1})^j$

Gated Recurrent Unit (GRU)

- Have no separate memory cells
- Linear interpolation
  - The previous activation h_{t-1}^j
  - The candidate activation $\tilde{h}_t^j$

  - $h_t^j = (1 - z_t^j)h_{t-1}^j + z_i^j \tilde{h}_t^j$
  - $\tilde{h}_t^j = \tanh(W x_t + U(r_t \odot h_{t-1})^j$



Gated Recurrent Unit (GRU)

Update gate

- Decide **how much the unit updates its activation, or content**
- $z_t^j = \sigma(W_z x_t + U_z h_{t-1})^j$

Reset gate

- Determine **the percentage of oblivion to past information** based on

  1) Information of the current timestamp

  2) Past information

- $r_t^j = \sigma(W_r x_t + U_r h_{t-1})^j$

Gated Recurrent Neural Networks

Similarities

- Keep the existing content and add the new content on top of it

Differences

- Control of the amount of information to the next timestamp
- Oblivion and renewal of information

Advantages

- Easy for each unit to remember the existence of a specific feature in the input stream
- Reduce the difficulty due to vanishing gradients via the addition

Experiments

Tasks

- Sequence modeling
  - Aim at learning a probability distribution over sequences
    → *Maximize the log-likelihood of a model!*
  - $\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n; \theta)$

Datasets

- Music Datasets
  - Polyphonic music modeling
  - Nottingham, JSB Chorales, MuseData, Piano-midi
- Ubisoft Datasets
  - Speech signal modeling
  - Ubisoft A, Ubisoft B

Experiments

☐ Models

- Make models have approximately the same number of parameters

Unit	# of Units	# of Parameters
Polyphonic music modeling		
LSTM	36	$\approx 19.8 \times 10^3$
GRU	46	$\approx 20.2 \times 10^3$
tanh	100	$\approx 20.1 \times 10^3$
Speech signal modeling		
LSTM	195	$\approx 169.1 \times 10^3$
GRU	227	$\approx 168.9 \times 10^3$
tanh	400	$\approx 168.4 \times 10^3$

Table 1: The sizes of the models tested in the experiments.

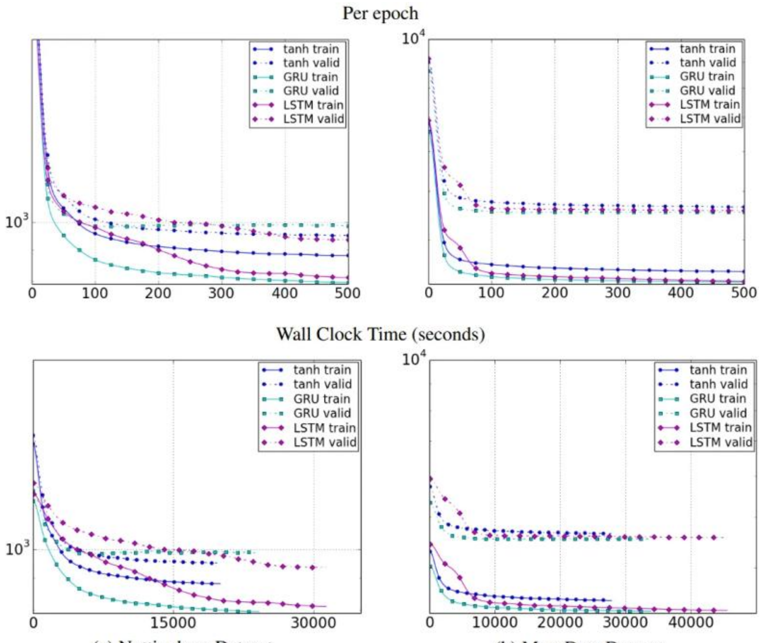
Experiments

- The average negative log-probabilities of the training and test sets
  - GRU-RNN outperformed all the others on all the datasets except for the Nottingham

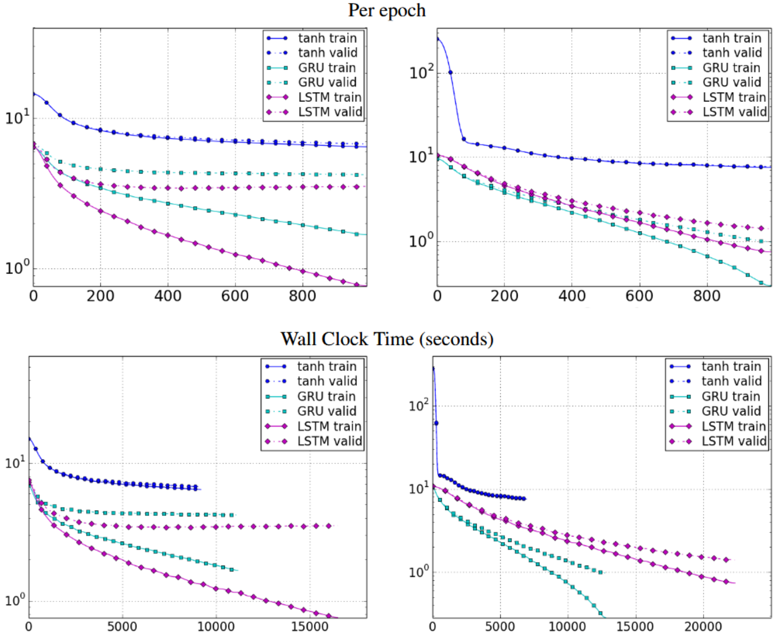
			tanh	GRU	LSTM
Music Datasets	Nottingham	train	3.22	2.79	3.08
		test	**3.13**	3.23	3.20
	JSB Chorales	train	8.82	6.94	8.15
		test	9.10	**8.54**	8.67
	MuseData	train	5.64	5.06	5.18
		test	6.23	**5.99**	6.23
	Piano-midi	train	5.64	4.93	6.49
		test	9.03	**8.82**	9.03
Ubisoft Datasets	Ubisoft dataset A	train	6.29	2.31	1.44
		test	6.44	3.59	**2.70**
	Ubisoft dataset B	train	7.61	0.38	0.80
		test	7.62	**0.88**	1.26

Experiments

☐ Learning curves of the best validation runs



(a) Nottingham Dataset (b) MuseData Dataset



(a) Ubisoft Dataset A (b) Ubisoft Dataset B

Conclusion

What is a Recurrent Neural Network?

- Address sequence data
- Many to One, Many to Many, One to Many
- Long-Term Dependency Problem

Gated Recurrent Neural Network

- Use several gates to catch long term dependencies

Experiments

- Gated Neural Networks outperform traditional RNN