NLP Techniques and Applications

- Text prediction using GRU

Outline

- Gated recurrent unit (GRU) structure
- Comparison between GRUs and vanilla RNNs
- Vanishing Gradients and Exploding Gradients

GRU

Ants are really interesting. ___________ are everywhere.”

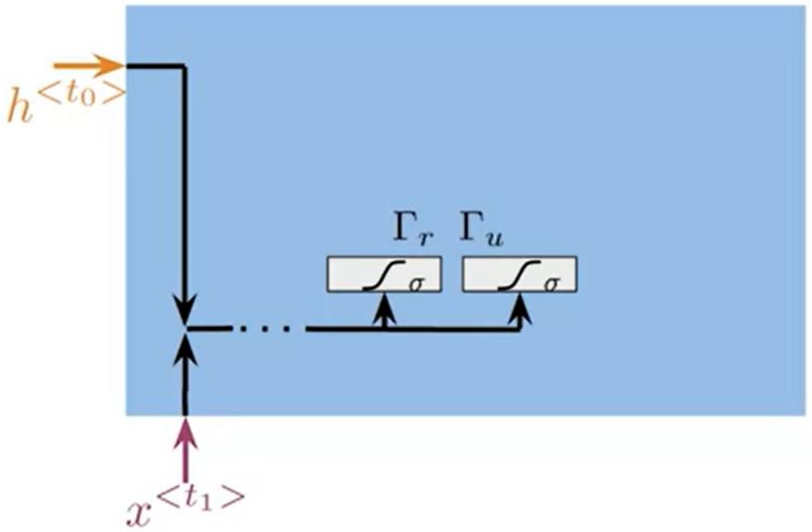
GRU

Ants are really interesting. _They_ are everywhere.”

Plural

Relevance and update gates to remember important prior information

GRU

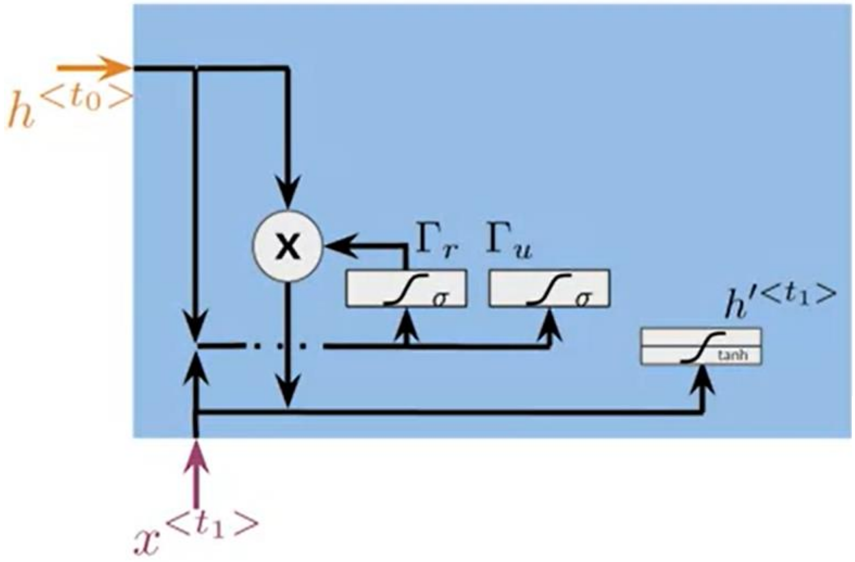


Gates to keep/update relevant information in the hidden state

$$\Gamma_r = \sigma(W_r[h^{<t_0>}, x^{<t_1>}] + b_r)$$

$$\Gamma_u = \sigma(W_u[h^{<t_0>}, x^{<t_1>}] + b_u)$$

GRU



Gates to keep/update relevant information in the hidden state

$$\Gamma_r = \sigma(W_r[h^{<t_0>}, x^{<t_1>}] + b_r)$$

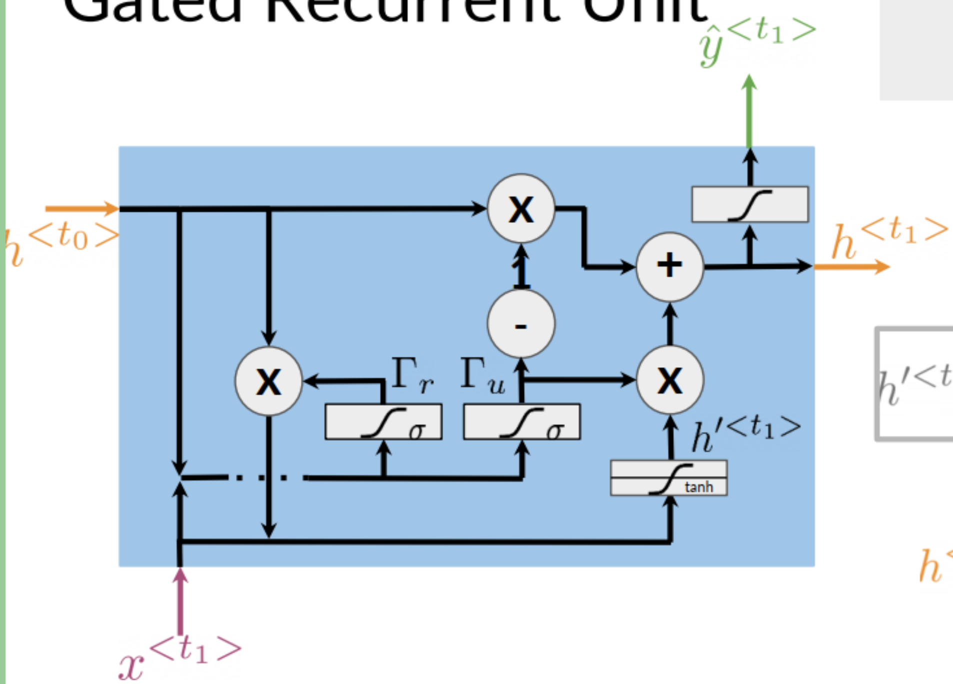
$$\Gamma_u = \sigma(W_u[h^{<t_0>}, x^{<t_1>}] + b_u)$$

$$\boxed{h'^{<t_1>} = \tanh\left(W_h[\Gamma_r * h^{<t_0>}, x^{<t_1>}] + b_h\right)}$$

Hidden state candidate

GRU

Gated Recurrent Unit



Gates to keep/update relevant information in the hidden state

$$\Gamma_r = \sigma(W_r[h^{<t_0>}, x^{<t_1>}] + b_r)$$

$$\Gamma_u = \sigma(W_u[h^{<t_0>}, x^{<t_1>}] + b_u)$$

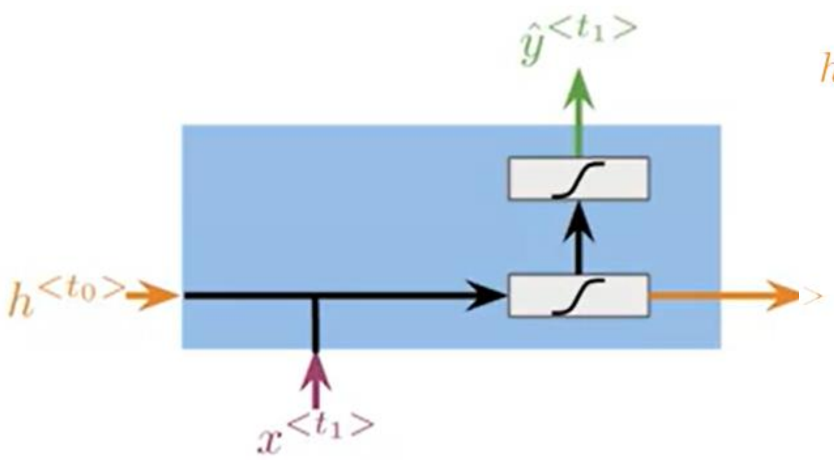
$$h'^{<t_1>} = \tanh\left(W_h[\Gamma_r * h^{<t_0>}, x^{<t_1>}] + b_h\right)$$

Hidden state candidate

$$h^{<t_1>} = (1 - \Gamma_u) * h^{<t_0>} + \Gamma_u * h'^{<t_1>}$$

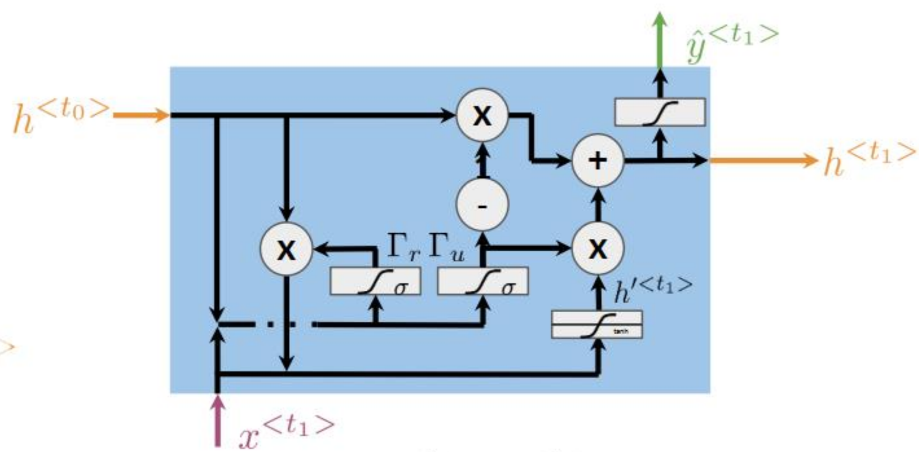
$$\hat{y}^{<t_1>} = g(W_y h^{<t_1>} + b_y)$$

Comparison of Vanilla RNN and GRU



$$h^{<t>} = g(W_h[h^{<t-1>}, x^{<t>}] + b_h$$

$$\hat{y}^{<t>} = g(W_{yh}h^{<t>} + b_y)$$



$$\Gamma_u = \sigma(W_u[h^{<t_0>}, x^{<t_1>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[h^{<t_0>}, x^{<t_1>}] + b_r)$$

$$h'^{<t_1>} = \tanh\left(W_h[\Gamma_r * h^{<t_0>}, x^{<t_1>}] + b_h\right)$$

$$h^{<t_1>} = (1 - \Gamma_u) * h^{<t_0>} + \Gamma_u * h'^{<t_1>}$$

$$\hat{y}^{<t_1>} = g(W_y h^{<t_1>} + b_y)$$

Summary

- GRUs “decide” how to update the hidden state
- GRUs help preserve important information

Question

What problem, related to vanilla RNNs, do GRUs tackle?

- Restricted flow of information from the past to the present
- Overfitting
- Loss of relevant information for long sequences of words
- High computational time for training and prediction.

RNNs: Advantages

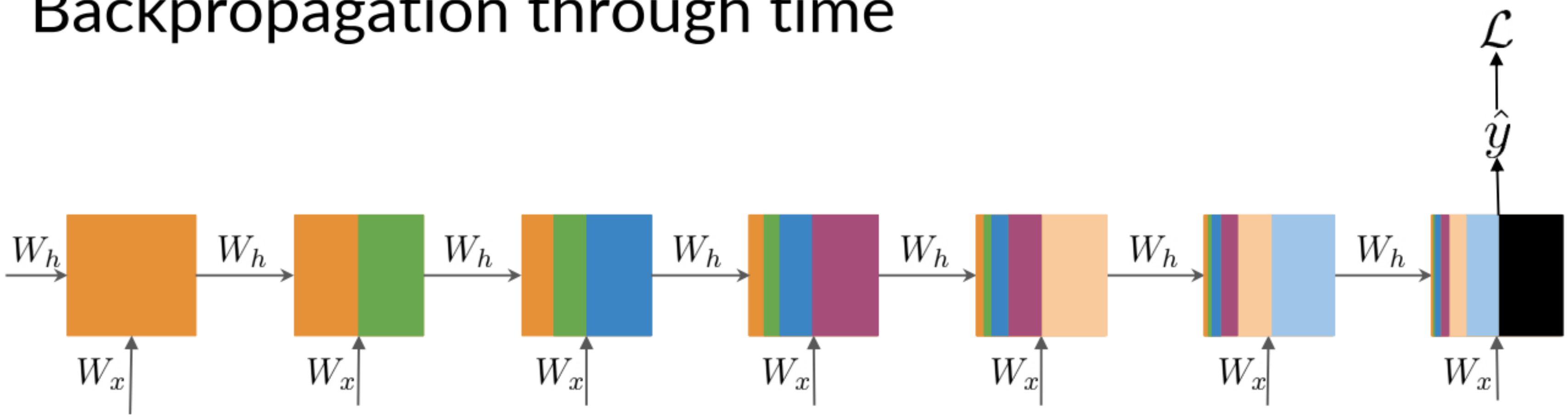
- Captures dependencies within a short range
- Takes up less RAM than other n-gram models

RNNs: Disadvantages

- RNNs: Struggles to capture long term dependencies
- Prone to vanishing or exploding gradients

RNNs

Backpropagation through time



W_x, W_h → Same at every step

$$\frac{\partial L}{\partial W_h} \propto \sum_{1 \le k \le t} \left(\prod_{t \ge i > k} \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W_h}$$

Gradient is proportional to a sum of partial derivative products

Backpropagation through time

$$\frac{\partial L}{\partial W_h} \propto \sum_{1 \leq k \leq t} \boxed{\left(\boxed{\prod_{t \geq i > k} \frac{\partial h_i}{\partial h_{i-1}}} \right) \frac{\partial h_k}{\partial W_h}} \longrightarrow \text{Contribution of hidden state } \boldsymbol{k}$$

Length of the product proportional to how far ***k*** is from ***t***

$$\boxed{\frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial h_{t-3}} \frac{\partial h_{t-3}}{\partial h_{t-4}} \frac{\partial h_{t-4}}{\partial h_{t-5}} \frac{\partial h_{t-5}}{\partial h_{t-6}} \frac{\partial h_{t-6}}{\partial h_{t-7}} \frac{\partial h_{t-7}}{\partial h_{t-8}} \frac{\partial h_{t-8}}{\partial h_{t-9}} \frac{\partial h_{t-9}}{\partial h_{t-10}} \frac{\partial h_{t-10}}{\partial W_h}}$$

Contribution of hidden state ***t-10***

Backpropagation through time

$$\frac{\partial L}{\partial W_h} \propto \sum_{1 \le k \le t} \left(\prod_{t \ge i > k} \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W_h}$$

Contribution of hidden state ***k***

Length of the product proportional to how far ***k*** is from ***t***

Partial derivatives <1	Contribution goes to 0	Vanishing Gradient
Partial derivatives >1	Contribution goes to infinity	**Exploding** Gradient

Solving for vanishing or exploding gradients

Identity RNN with ReLU activation

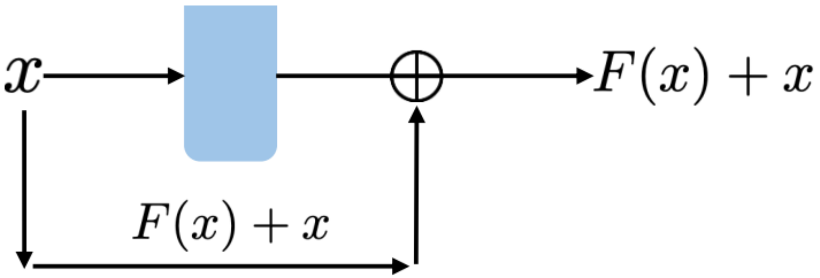
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

-1 → 0

Gradient clipping

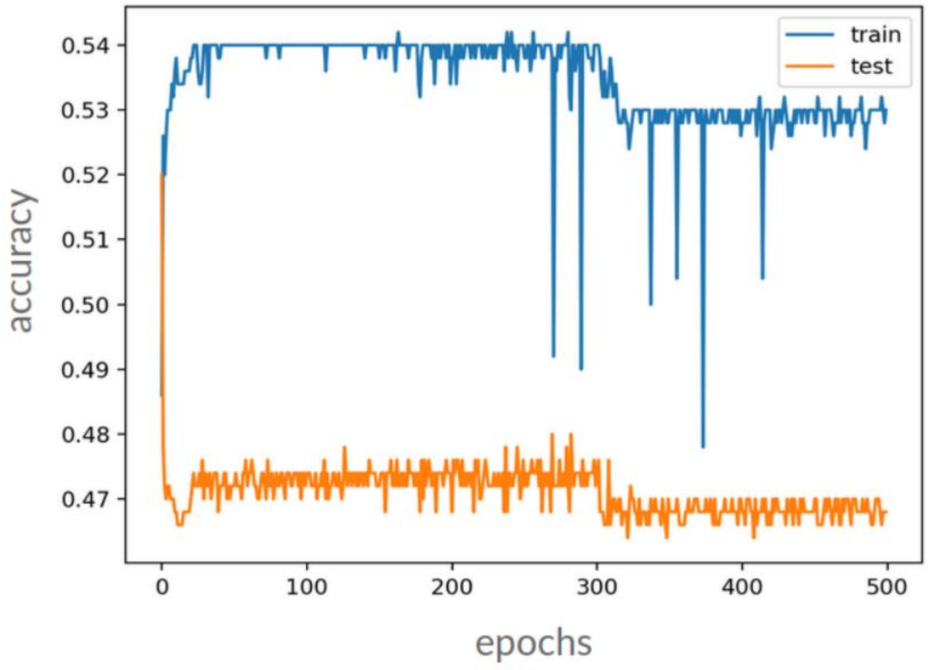
32 → 25

Skip connections



Poor Model Performance

This plot shows poor model performance and may indicate a vanishing gradient problem.



LSTMs: a memorable solution

- Learns when to remember and when to forget
- Basic anatomy:
  - A cell state
  - A hidden state
  - Multiple gates
- Gates allow gradients to avoid vanishing and exploding

LSTMs: Based on previous understanding

Starting point with some irrelevant information

↓

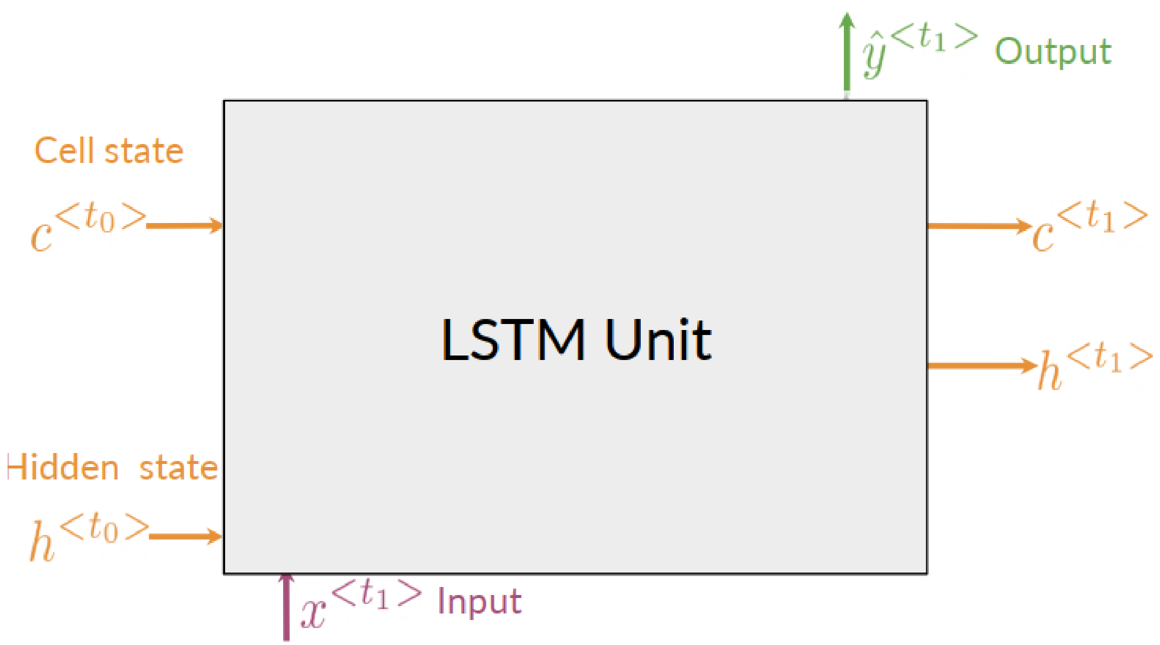
Cell and Hidden States

Discard anything irrelevant

Add important new information → Gates

Produce output

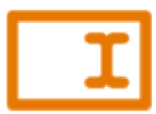
Gates in LSTM



1. **Forget Gate:** information that is no longer important
2. **Input Gate:** information to be stored
3. **Output Gate:** information to use at current step

Applications of LSTMs

Next-character prediction



Chatbots

Music composition

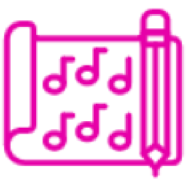


Image captioning

Speech recognition

Summary

LSTMs offer a solution to vanishing gradients

Typical LSTMs have a cell and three gates:

- Forget gate
- Input gate
- Output gate