Inference in RNNs

$h_t = g(Uh_{t-1} + Wx_t)$

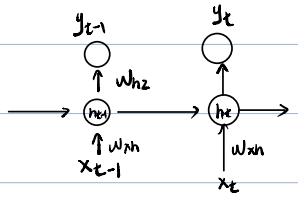
$y_t = f(Vh_t)$

Let input, hidden and output layer as d_{in}, d_h and d_{out}, Given this, our three parameter matrices are $W \in R^{d_h \times d_{in}}$, $U \in R^{d_h \times d_h}$, and $V \in R^{d_{out} \times d_h}$

$y_t = \text{softmax}(Vh_t)$

RNN as Language model

The input sequence $X = [x_1; \cdots x_t; \cdots x_N]$ consist of a series of word embedding each represented as a one-hot vector of size $|V| \times 1$, the output $\hat{y}$ is a vector representing a probability distribution over the vocabulary.



$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$

$y_t = \text{softmax}(W_{hz} h_t + b_z)$

$y_t[i] = P(w_{t+1} = i \mid w_1, \cdots w_t)$

The probability of entire sequence is $P(w_{1:n}) = \prod_{i=1}^{n} P(w_i \mid w_{1:i-1})$

$= \prod_{i=1}^{n} y_i[w_i]$

↗ the probability of the true word w_i at step i

To train an RNN as language model, we use a corpus of text as training material

The loss is $L_{CE} = -\sum_{w \in V} y_t[w] \log \hat{y}_t[w]$

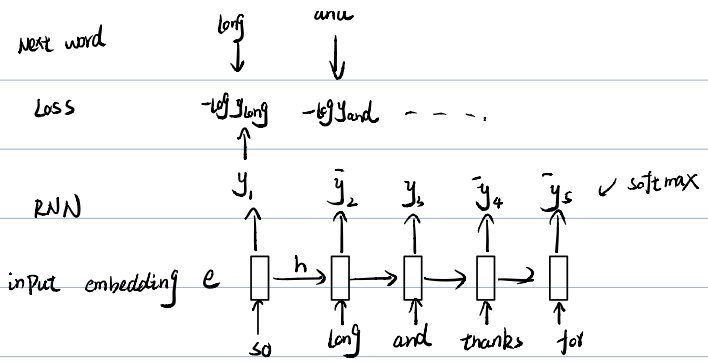
which measures the difference between a predicted probability distribution and the correct distribution

Language Model

Given sentence $S = w_1 w_2 \cdots w_n$

$P_M(S) = P(w_1 w_2 \cdots w_n) = P_M(w_1) P_M(w_2 \mid w_1) \cdots P_M(w_n \mid w_1 w_2 \cdots w_{n-1})$

and

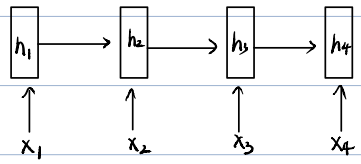


$e_t = E x_t$

$h_t = g(W h_{t-1} + W e_t)$

$y_t = \text{softmax}(E^{interval} h_t)$

Vanishing / Exploding Gradient



$$\frac{\partial J^{(4)}(\theta)}{\partial h^{(1)}} = \frac{\partial J^{(4)}(\theta)}{\partial h^{(4)}} \cdot \frac{\partial h^{(4)}}{\partial h^{(3)}} \cdot \frac{\partial h^{(3)}}{\partial h^{(2)}} \cdot \frac{\partial h^{(2)}}{\partial h^{(1)}}$$

9.5 Stacked and bidirectional RNN

Stacked RNNs consist of multiple networks where the output of one layer serves as the input to subsequence layer

When we need to have access to the entire input sequence, we would like to use bidirectional RNNs

How to solve Gradient exploding

Use regularization

$\theta^{(t+1)} = \theta^{(t)} - \eta_t \nabla_\theta f(\theta^t)$ if $|\nabla_\theta f(\theta^t)| \geq$ threshold, then $g^{(t)} = g^{(t)} \cdot \frac{\text{threshold}}{\|g^{(t)}\|}$

LSTM

Tanh activation squishes values to always be between -1 and 1, it regulate the values flowing through the network

Sigmoid is used to update or forget data because any number getting multiplied by 0 is 0 causing values to disappears, 1 represents keeping this value

$f^{(t)}$: forget gate, use it to decide what information we are going to forget

$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$

$i^{(t)}$: input gate, decide which information we need to keep

$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$

$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$

next, update cell state C_{t-1} to C_t

$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$ → how much we decide to update each state value

↑ whether forget previous information

Finally, decide which information we need to output

$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$

$h_t = o_t * \tanh(C_t)$