Exercícios Deep Learning
Aula 10

January 11, 2022

1 RNNs

1- Suponha que você esteja calculando o gradiente para algum parâmetro de uma RNN com várias camadas, considere que o gradiente para a primeira saída é dado por $\partial L_1 = 3$.

a) Quantas iterações são necessárias até que o vetor gradiente se iguale a zero (considere $< 2^{-31}$)? Considere que o gradiente no tempo t é dado por $\partial L_t = 0.5 * \partial L_{t-1}$.

b) Quantas iterações são necessárias até que ocorra um *overflow* (considere $> 2^{31}$)? Considere que o gradiente no tempo t é dado por $\partial L_t = 2 * \partial L_{t-1}$.

2- Você está treinando uma RNN, e descobre que seus pesos e ativações estão assumindo o valor de NaN ("Not a Number"). Diga uma causa provável desse problema.

3- Por que vanishing gradients/exploding gradients são problemas mais comum em RNNs simples do que em redes convencionais? Qual é uma possível solução para o problema de exploding gradients?

4- Por que vanishing gradients é um problema menos grave em RNNs do que em redes convencionais?

5- Por que às vezes não é viável fazer o gradient descent em todos os tempos da RNN? Quando isso é um problema?

2 GRU e LSTM

6- Suponha que você esteja treinando uma LSTM. Você tem um vocabulário de 10000 palavras e está usando uma LSTM com ativações de 100 dimensões

$a^{<t>}$. Qual é a dimensão de Γ_u em cada tempo?

7- Você tem um cão de estimação cujo humor depende muito do clima atual e dos últimos dias. Você coletou dados dos últimos 365 dias sobre o clima, que você representa como uma sequência como $x^{<1>}, ..., x^{<365>}$. Você também coletou dados sobre o humor do seu cão, que você representa como $y^{<1>}, .., y^{<365>}$. Você gostaria de criar um modelo para mapear a partir de $x \rightarrow y$. Você deve usar um RNN unidirecional ou RNN bidirecional para esse problema?

8- Considere as equações para o GRU e o LSTM:

GRU	LSTM
$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$	$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$
$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$	$\Gamma_u = \sigma(W_u[\, a^{<t-1>}, x^{<t>}] + b_u)$
$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$	$\Gamma_f = \sigma(W_f[\, a^{<t-1>}, x^{<t>}] + b_f)$
$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$	$\Gamma_o = \sigma(W_o[\, a^{<t-1>}, x^{<t>}] + b_o)$
$a^{<t>} = c^{<t>}$	$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$
	$a^{<t>} = \Gamma_o * c^{<t>}$

O Update Gate e o Forget Gate são análogos à quais parâmetros na GRU?

9- Considere uma GRU onde queremos usar somente a entrada no tempo t' para estimar a saída no tempo $t > t'$. Quais são os melhores valores para o Γ_u e Γ_r em cada tempo?

10- Considere uma GRU com dois neurônios na camada escondida e entrada em cada tempo como um valor real. Assuma que os bias são 0, $x^{<1>} = 3$, $\Gamma'_r = [1, 1]$ e

$$W_c = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 0 \end{bmatrix} W_u = \begin{bmatrix} 1 & 0 & 3 \\ 2 & 0 & 0 \end{bmatrix}$$

Calcule $c^{<1>}$.

3 Attention

11- Descreva a motivação e como funcionam os Modelos de Atenção.

Solução

1-
a) 34
b) 31

2- Exploding gradients.

3- Como em RNNs os mesmos pesos são usados para todas as entradas, o efeito delas pode ser repetido em cada tempo da rede. Se o efeito é aumentar os valores, eles podem se tornar grandes demais (exploding gradients). Analogamente, se o efeito é diminuir os valores, eles podem se tornar muito pequenos (vanishing gradients). Uma solução robusta para o problema de exploding gradients é o gradient clipping.

4- RNNs geralmente possuem uma entrada e saída em cada tempo do modelo. Isso significa que em cada tempo ela é semelhante a uma rede de uma ou duas camadas, onde vanishing gradients não são um grande problema. Além disso, como a RNN reutiliza os mesmos pesos, é preciso que apenas que uma das iterações tenha gradientes para que os pesos em todas as iterações sejam atualizados. Ou seja, é menos provável que uma RNN fique totalmente sem atualização de pesos.

5- Para fazer o backpropagation de uma rede, você precisa manter os valores de saída de cada camada, até chegar à etapa de backpropagation. Isso significa que o número de iterações possível de ser calculado é limitado pela memória do computador. Se sua RNN possui muitas camadas (ou seja, ela considera vários tempos diferentes), sua memória pode não ser suficiente para calcular o gradiente em todas camadas. Isso pode ser um problema onde seus dados possuem longas dependências temporais. Por exemplo, na previsão de novas palavras em um texto, um adjetivo pode aparecer muito depois do substantivo a que ele se refere, nesse caso a informação de concordância de gênero ou número pode ser perdida.

6- 100

7- RNN unidirecional, porque o valor de $y^{<t>}$ depende apenas de $x^{<1>}, ..., x^{<t>}$, e não de $x^{<t+1>}, ..., x^{<365>}$

8- Γ_u e $1 - \Gamma_u$

9- Para $c^{<t'>}$, $\Gamma_u = 1$ e $\Gamma_r = 0$. Para $c^{<i>}$, sendo $t' < i \leq t$, $\Gamma_u = 0$ e Γ_r não é utilizado.

10-

$$\tilde{c}^{<1>} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \Gamma_u = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}, c^{<1>} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

11- Tome como exemplo a tarefa de tradução. A arquitetura encoder-decoder funciona bem para sequências curtas (não muito curtas, que são um pouco difíceis de traduzir). Depois disso, o score BLEU começa a decair. Ideia dos attention models: a cada passo do decoder, use conexões diretas com o encoder para focar em uma parte específica da sequência de entrada. Reduz o problema da queda no score BLEU. Ideia básica: cada palavra de saída vem de uma ou de algumas poucas palavras da entrada. Talvez seja possível aprender a prestar atenção apenas às palavras relevantes conforme geramos a saída.

Resuminho: `https://stanford.io/3tqhVlG`
`shorturl.at/moCXY`

Attention:
`https://www.youtube.com/watch?v=SysgYptB198`
`https://www.youtube.com/watch?v=quoGRI-1l0A`