Sequence To Sequence Model

Machine Translation

French → Encoder → Decoder → English

Image Captioning

Image → CNN → RNN → Caption

Picking the most likely Sentence

Build A Conditional Language Model to optimize

$P(y^{<1>}, y^{<2>}, \cdots, y^{<T_y>}) \times$

$$\underset{y^{<1>}, y^{<2>}, \cdots, y^{<T_y>}}{\operatorname{argmax}} \left(P(y^{<1>}, y^{<2>}, \cdots, y^{<T_y>} | x^{<1>}, x^{<2>}, \cdots x^{<T_x>}) \right)$$

Greedy Search: pick the "best" word one by one

local optimal $\nrightarrow$ global optimal

Beam Search

Beam Width (B = 3) # best candidates

Beam Width = 1 → Greedy Search

Step 1: Pick 3 Best first words $P(y^{<1>} | x)$

Step 2: Pick 3 Best "Pairs" (first, second) $P(y^{<1>}, y^{<2>} | x) = P(y^{<1>} | x) P(y^{<2>} | x, y^{<1>})$

Instantiate 3 Copies of Network to calculate 30000 probabilities

(3 x 10000)

Beam Width ↵ ↳ Vocab Size

... ...

... ...

Refinements of Beam Search

→ Length Normalization

$$\arg\max_y \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \cdots, y^{<t-1>})$$

Take log to avoid numerical underflow

$$P(y^{<1>}, y^{<2>}, \cdots, y^{<t>} | x) = P(y^{<1>}|x)\, P(y^{<2>}|y^{<1>}) \cdots P(y^{<T_y>} | x, y^{<1>}, y^{<2>}, \cdots, y^{<T_y-1>})$$

$$\arg\max_y \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \cdots, y^{<t-1>})$$

normalize sentence length to allow longer output sentences

$$\arg\max_y \frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \cdots, y^{<t-1>})$$

$\alpha = 0$ No Normalization

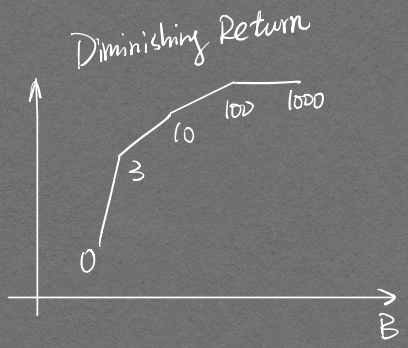
$\alpha = 1$ Full Normalization

Normalized Log likelihood

→ Choose Beam Width B

Large B → Better Results, Slower

Small B → Worse Results, Faster



Beam Search is an approximate optimization algorithm, or heuristic algorithm

Error Analysis of Beam Search

Use RNN to compute $P(\hat{y}|x)$, $P(y^*|x)$

Case 1: $P(y^*|x) > P(\hat{y}|x)$ → Beam Search doesn't find the optimal y

Case 2: $P(y^*|x) \leq P(\hat{y}|x)$ → RNN computes $P(y|x)$ wrong

But y^* is better than $\hat{y}$

BLEU Score

Bilingual Evaluation Understudy

Bleu : A method for automatic of machine translation [Papineni et al, 2002]

$$P_1 = \frac{\sum_{\text{unigram} \in \hat{y}} \text{Count}_{\text{clip}}(\text{unigram})}{\sum_{\text{unigram} \in \hat{y}} \text{Count}(\text{unigram})}$$

P_1: Position, Unigram

$$P_n = \frac{\sum_{\text{n-gram} \in \hat{y}} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{\text{n-gram} \in \hat{y}} \text{Count}(\text{n-gram})}$$

P_n: Position, n-gram

Combined BLEU Score

$$\underline{BP} \exp\left(\frac{1}{4}\sum_{n=1}^{4} P_n\right)$$

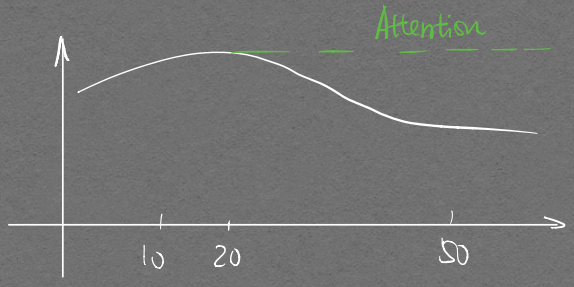
Brevity Penalty

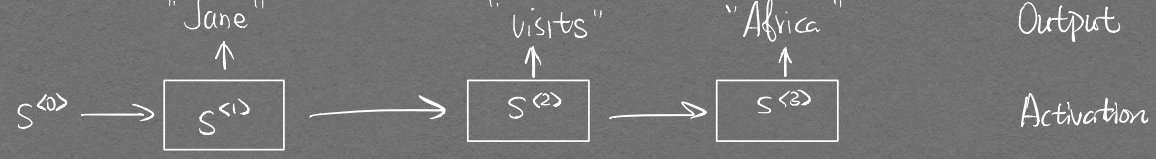
$$BP = \begin{cases} 1 & \text{if Machine Translation Output Length} < \text{Reference Length} \\ \exp\left(1 - \frac{\text{Reference Output Length}}{\text{Machine Translation Output Length}}\right) & \text{otherwise} \end{cases}$$

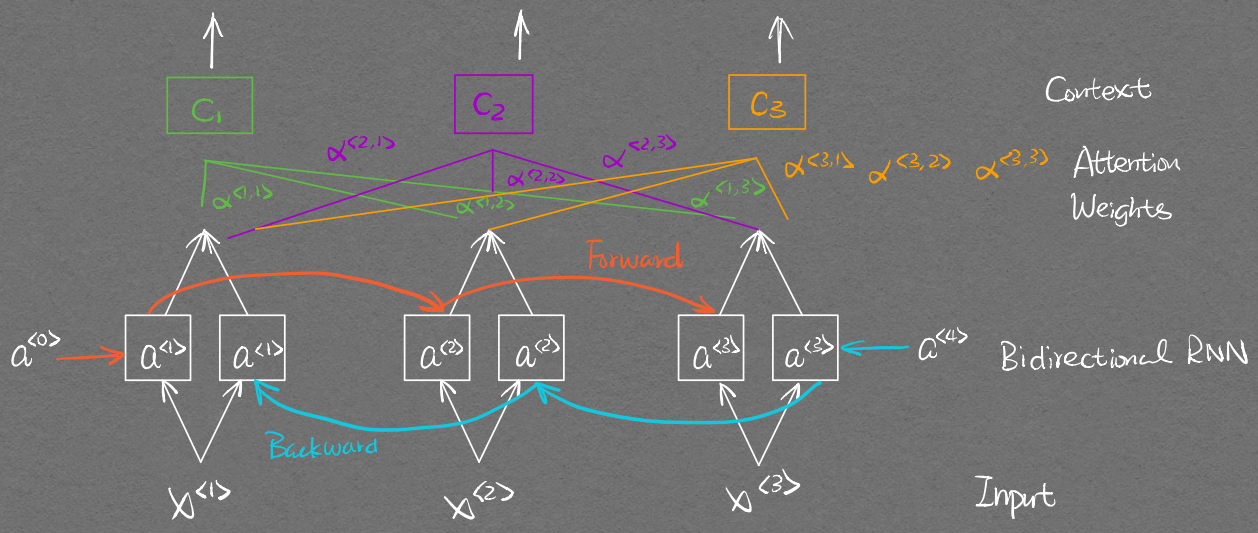
Attention Model Intuition

Problem of Long Sequence.

Neural Machine Translation by jointly learning to align and translate
Bahdanau et al. 2014







Attention Model

$t' \rightarrow$ Input Index $\quad t \rightarrow$ Output Index

$$a^{<t'>} = (\overrightarrow{a}^{<t'>}, \overleftarrow{a}^{<t'>})$$ Feature Vector

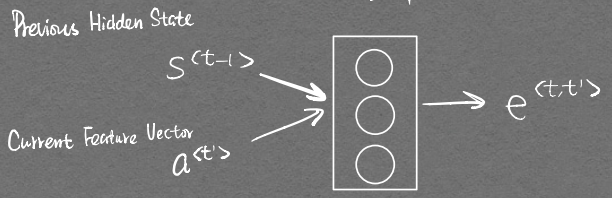
$$\sum_{t'} \alpha^{<1,t'>} = 1$$

$\alpha^{<t,t'>}$ = amount of "attention" $y^{<t>}$ should pay to $a^{<t'>}$

$$c^{<1>} = \sum_{t'} \alpha^{<1,t'>} a^{<t'>}$$

How to compute $\alpha^{<t,t'>}$

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$



Total # Parameters = $T_x \cdot T_y$ Quadratic!

Attention also applies to:

1) Image Caption
2) Date Format 23 April, 1564 → 1564-04-23 !
3) Speech Recognition

Speech Recogition

x → y

Audio clip → transcript

Before : phonemes (units of sound)

Academia 300h
3000h

Commercial 10,000h

CTC cost (Connectionist Temporal Classification)

10s 100Hz Audio → 5 words

"the quick brown fox"

Generate output with blank spaces

ttt_ _ _ h_ e_ _ _␣_ qq_ _

Blank

Space

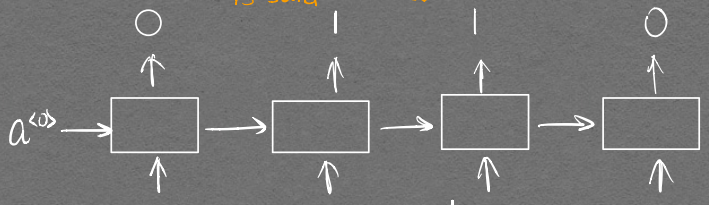
Triggerword Detection

First time the triggerword is said

Some blank time

2nd

3rd



Imbalance Data

Audio Clip