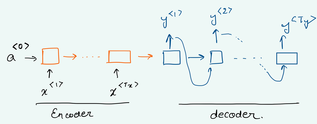
Sequence to Sequence models :-

French sentence :. → Encoder
↳ English sentence → Decoder



* Finding the most likely sentence is challenging
* Eg → French Statement
→ English 1
→ English 2 → Best answer.
→ English 3

$P(\hat{y}^{<1>}, \hat{y}^{<2>}, \hat{y}^{<3>}, \hat{y}^{<4>} \cdots \hat{y}^{<T_y>} \mid x)$ → Maximize it

↓ ↓
Jane is visiting Africa in September

↓ ↓ |
Jane is going to be visiting Africa in September.

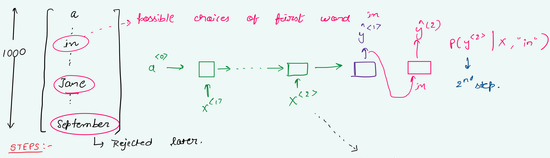
$P(\text{Jane is going} \mid x) > P(\text{Jane is visiting} \mid x)$

* Selecting one word at a time is expensive and inaccurate.
* The final sentence might not be correct.

Beam Search :. (Most likely sentence)

Beam width = 3. (width: 1 is greedy approach)

It tries to pick the first word of english translation that it is going to output.
↳ Considering multiple outputs



STEPS :-

1. Predict the first word using beam width (top 3)
2. Consider their next word.

P(1) in September ... out of $10,000$
P(2) Jane visits
P(3) Jane is
top-3 is now updated
↳ September → Rejected → * * Important

Maximize this in second step

$P(y^{<1>}, y^{<2>} \mid x) = P(y^{<1>} \mid x) * P(y^{<2>} \mid x)$

then the search moves forward, top 3 always updated accordingly

* Length Normalization :-

$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{<t>} \mid x, y^{<1>}, \ldots y^{<t-1>})$$

↳ Try maximizing it.

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

T_y → length of sentence

$\alpha = 0.7$ $\alpha = 1$ $\alpha = 0$
↳ hyperparameter.

{ Normalized log probability objective }

* → Large B → More computationally expensive.

Error Analysis :-

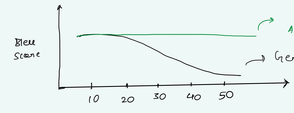
French → Jane visite l'Afrique en Septembre
Human → Jane visits Africa in September
Algorithm → Jane visited Africa last september.

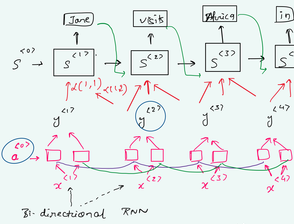
Case 1 :- $P(y^* \mid x) > P(\hat{y} \mid x)$ Beam search is at fault
↳ increase beam width

Case 2 :- $P(y^* \mid x) < P(\hat{y} \mid x)$ RNN is at fault
↳ deeper layers → overfitting / underfitting

* Do a statistical analysis on the results.

Attention Models :-

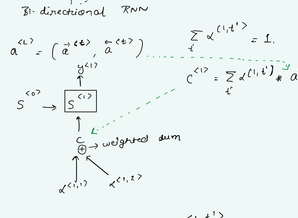




$a^{<t>} = (\vec{a}^{<t>}, \overleftarrow{a}^{<t>})$

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

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



Computing attention $\alpha^{<t,t'>}$

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

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

softmax.

$s^{<t-1>}$, $a^{<t'>}$ → [0 0 0 0] → $e^{<t,t'>}$

→ $\alpha^{<t,t'>}$ and $e^{<t,t'>}$ depend on $s^{<t-1>}$ and $a^{<t'>}$

④ Quadratic time to train.
$n \to 10 \to O(10^2)$ → very slow.