# Sequence to sequence models

# Refinements to beam search

deeplearning.ai

# Length normalization

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

it unnaturally tends to prefer very short translation, it tends to prefer very short outputs

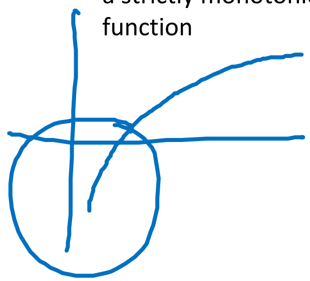
$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \ldots, y^{<t-1>})$$

log

numerical underflow

too small for the floating part representation in your computer to store accurately

a strictly monotonically increasing function



$\log P(y|x) \leftarrow$

$P(y|x) \leftarrow$

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

So by taking logs, you end up with a more numerically stable algorithm that is less prone to rounding errors, numerically rounding errors, or to really numerical underflow.

$T_y = 1, 2, 3, \ldots, 30.$

normalize

take the average of the log of the probability of each word.

And this significantly reduces the penalty for outputting longer translations

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

normalized log probability objective

normalized log likelihood objective

this is a heuristic or this is a hack, there isn't a great justification for it but people have found this work well.

$\alpha = 0.7$

$\alpha = 1$

$\alpha = 0$

this is somewhere in between full normalization and no normalization

alpha is another hyperparameter of algorithm that you can tune to get the best result

Andrew Ng

# Beam search discussion

large B: better result, slower
small B: worse result, faster

Beam width B?

$1 \rightarrow 3 \rightarrow 10$, 100, $1000 \rightarrow 3000$

Unlike exact search algorithms like BFS (Breadth First Search) or DFS (Depth First Search), Beam Search runs faster but is not guaranteed to find exact maximum for $\arg\max_{y} P(y|x)$.

And there are some simple things you can compute to give you guidance on what you need to work on improving your algorithm.