CS224n: NLP with Deep learning

Week4

Lecture: 7: Vanishing Gradients, Fancy RNNs

- vanishing gradient problem

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

Recall: $h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)} + b_1)$

$$\frac{\partial h^{(t)}}{\partial h^{(t-1)}} = \text{diag}(\sigma'(W_h h^{(t-1)} + W_x x^{(t)} + b_1)) \cdot W_h$$

(chain rule)

so $$\frac{\partial J^{(i)}(\theta)}{\partial h^{(j)}} = \frac{\partial J^{(i)}(\theta)}{\partial h^{(i)}} \cdot \prod_{j \le t \le i} \frac{\partial h^{(t)}}{\partial h^{(t-1)}}$$

$$= \frac{\partial J^{(i)}(\theta)}{\partial h^{(i)}} \cdot W_h^{(i-j)} \prod_{j \le t \le i} \text{diag}(\sigma'(W_h h^{(t+1)} + \cdots))$$

↑ if W_h is small, this term gets vanish

get small exponentially

consider the matrix L2 norms

[When largest eigenvalue > 1, there is a similiar problem called "graident exploding".

- Gradient signal far away get lost, because it's much smaller than near-by gradient signal.

<Gradient> : a measure of the effect of the past on future

we can not tell between t and t+n

① no dependency

② wrong parameter if vanishing gradient.

Syntactic recency V.S. Sequential recency

Why "gradient exploding" a problem?

SGD: $\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$

bad updates: took too large a step

reach a bad parameter configuration

Solution:

Gradient clipping:

if $\|\hat{g}\| \geq$ threshold then

$$\hat{g} \leftarrow \frac{\text{threshold}}{\|\hat{g}\|} \hat{g}$$

"cliff problem"

How to fix vanishing gradient problem?

- difficult to for RNN to learn/preserve long-term information.

Long short-Term Memory (LSTM)

- on step (t), there is a hidden state $h^{(t)}$ and a cell state $c^{(t)}$
  - both are vectors length n.
  - The cell store long-term information.
  - The LSTM can erase, write and read information from the cell.

There are "gate", dynamic, to compute action of erase.

write, read

How LSTM helps solve vanishing gradient?

- easier preserve information over many time stamps.

(for example, set forget gate always set to fully remember previous cell content)

Gated-recurrent Units (GRU)

- simpler alternative to LSTM
  - update gate: $u^{(t)} = \sigma(W_u h^{(t-1)} + V_u x^{(t)} + b_u)$
  - reset gate: $r^{(t)}$

LSTM v.s. GRU

- GRU is quicker to compute and has fewer parameters
- no evidence which's performance is better
- LSTM is a good default choice (particularly if data has long dependency)
- Vanishing gradient is common problem
  - lower lagers are harder & slower to train
- skip-connection: $\mathcal{F}(x) + x$

  makes deep networks much easier to train.
  - Dense-Net: connect everything to everything
  - Highway connection: dynamic gate.

- Bidirectional RNNs
- Multi-lager RNNs (stacked RNNs)

Lecture 8 : Translation, Seq2Seq, Attention

Machine Translation (MT)

Def. translate source language X into another target language y

- pre-neural : rule-based, bilingual dictionary
- statistical : learn a probabilistic model from data

$$\underset{y}{\operatorname{argmax}} P(y|x) = \underset{y}{\operatorname{argmax}} P(x|y) P(y)$$

$P(x|y)$: Translation model

$P(y)$: Language model

- need parallel data

how to learn $P(x|y)$?

break it down further: consider alignment

- one-to-many
- many-to-one
- many-to-many

↑ very complex

consider $P(x, a|y)$

includes : aligning, fertile words

Decoding for MT: $\underset{y}{\operatorname{argmax}} P(x|y) \cdot P(y)$

- use a efficient algorithm, keep only parts of hypothesis

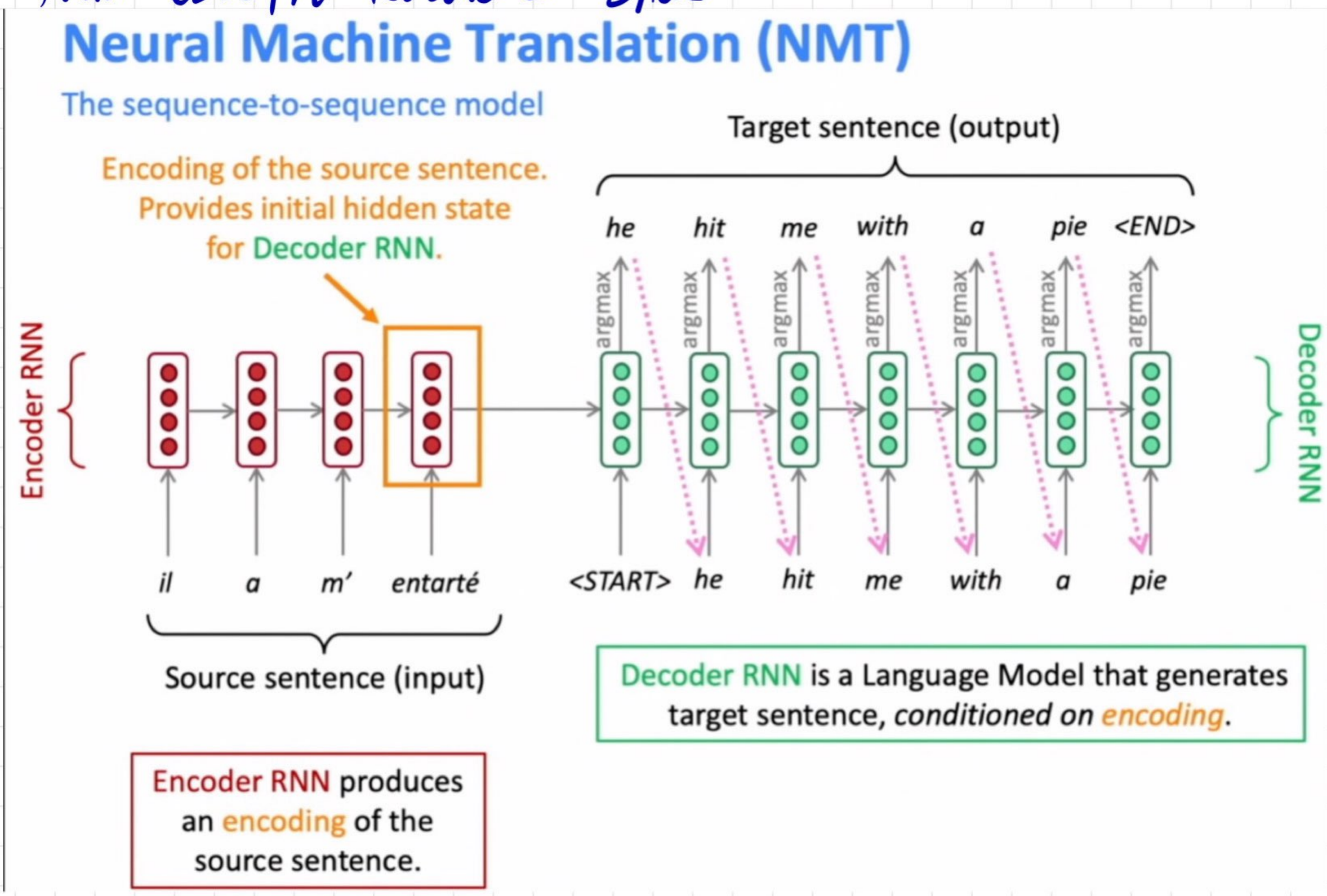
Overall, SMT system (statistical machine translation)

- a huge research field
- extremely complex :
  - separately-designed subcomponents
  - Lot of feature engineering
  - extra resources
  - human efforts

- Section 2: Neural Machine Translation

Sequence-to-sequence with 2 RNNs

from CS224n: lecture 8 Slide



2 sets of word-embeddings

Seq2Seq is very versatile:

- Summarization (long text → short text)
- Dialogue
- Parsing
- Code generation

The Seq2Seq is a Conditional Language Model

$$P(y|x) = P(y_1|x) \cdot P(y_2|y_1,x)\, P(y_3|y_1,y_2,x) \cdots P(y_T|y_1 \cdots y_{T-1},x)$$

Question: how to train a Seq2Seq?

Seq2Seq is optimized as a single-system.

Backpropogation operates end to end

- Greedy decoding : taking argmax on each step of the decoder
  decoding has no way to undo decisions
- Exhaustive search : $O(V^T)$ time-complexity

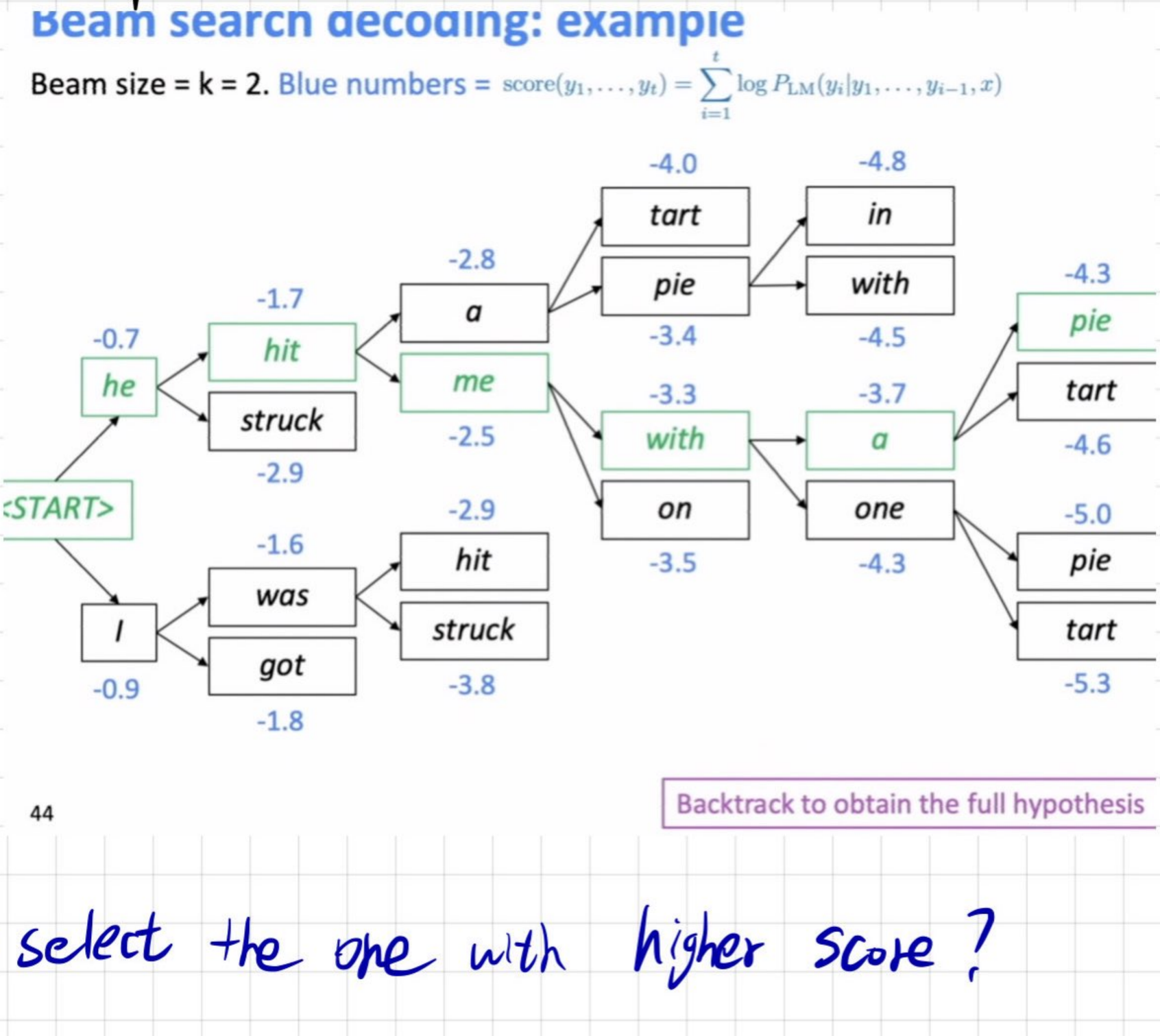
☆ : **Beam search decoding**

- Core idea: on each step of decoder, keep track of the k most probable partial translations (called hypothesis)
  - k is the beam size
- A hypothesis $y_1, \cdots, y_t$ has a score which is its log probability

$$\text{score}(y_1, \cdots, y_t) = \log P_{LM}(y_1, \cdots, y_t | x)$$
$$= \sum_{i=1}^{t} \log P_{LM}(y_i | x, y_1, \cdots y_{i-1})$$

- not guaranteed to find optimal solution
  but efficient enough

an example:



how to select the one with higher score?

- longer hypothesis have lower scores
- Fix: normalize by length

- advantage of NMT:
  - Better performance
    - △ more fluent
    - △ better use of context
    - △ better use of phrase similarities
  - a single neural network to be optimized end-to-end
  - less human engineering effort
  - Same method for different language-pairs

- disadvantage of NMT:
  - less interpretable
    - △ hard to debug
  - difficult to control
    - △ hard to specify a rule or guidelines
  - Safety concerns

- How do we measure Machine Translation?

BLEU (Bilingual Evaluation Understudy)

- n-gram precision
- plus a penalty for too-short system translations

BLEU is useful but not perfect

- there are many valid ways to translate
- so a good translation can have a low BLEV score.

Many difficulties remain:

- Out-of-vocabulary words
- Domain-mismatch
- maintain context over longer context
- Low-resource language pairs
- using common-sense is hard

• Section 3: Attentation is All you Need

encoder: might loss long-term information

too much pressure on a single vector

Attenation: on each step of the decoder, use direct connection to the encoder to focus on a particular part of the source sequence.

- dot product, softmax
- attention output (probability distribution)
- weighted sum → attention out.
- concatenate with decoder hidden state.

Attention

- allow decoder to focus on certain parts of the source
- solve the information bottleneck
- solve vanishing gradient problem by direct connection
- attention provides interpretability (get alignment for free)

Attention is a general Deep Learning technique