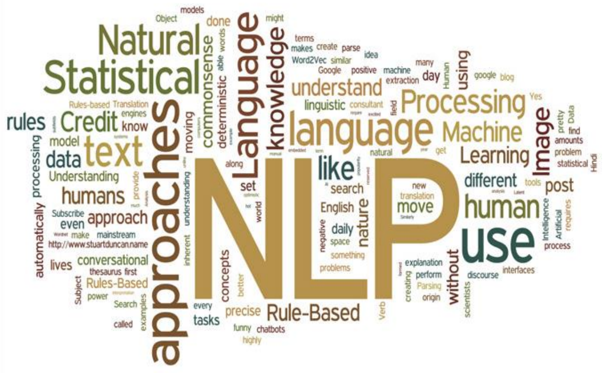


به نام خدا

گروه هوش مصنوعی، دانشکده مهندسی کامپیوتر

بخش سیزدهم

ترجمه ماشینی مبتنی بر شبکه های عصبی

(Machine Translation based on NNs)

حمیدرضا برادران کاشانی

Machine Translation

Figure from deeplearning.ai

Sequence-to-sequence

- Introduced by Google in 2014
- Maps variable-length sequences to fixed-length memory
- Inputs and outputs can have different lengths
- LSTMs and GRUs to avoid vanishing and exploding gradient problems

Figure from deeplearning.ai

Sequence-to-sequence

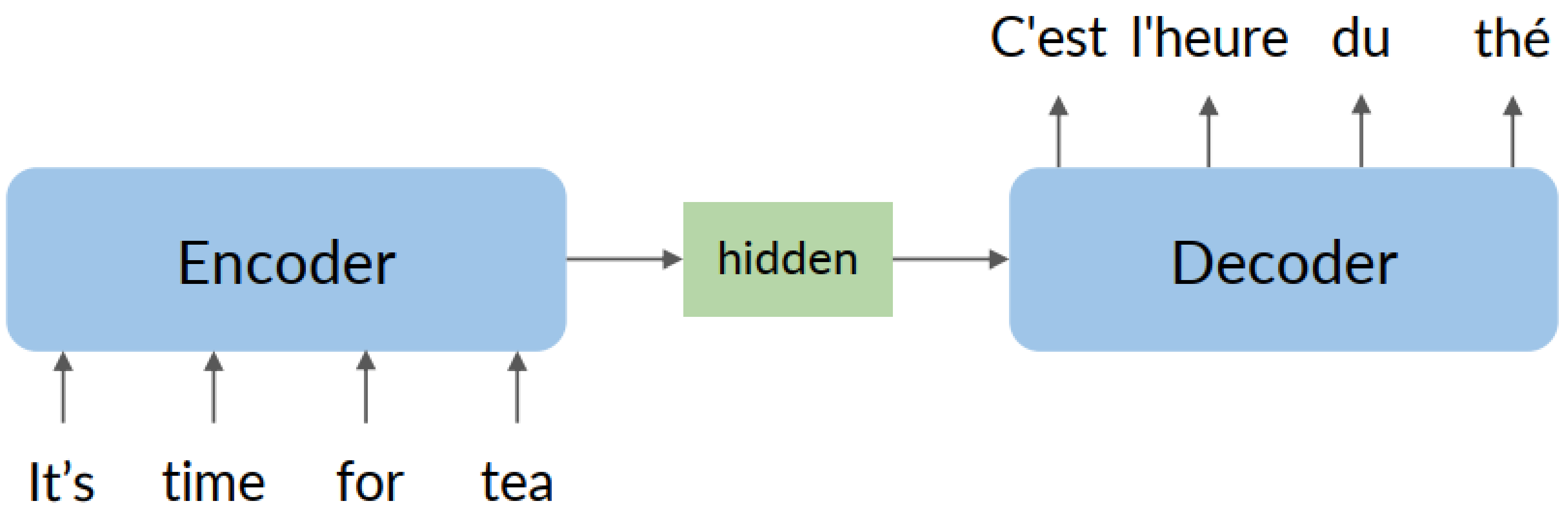
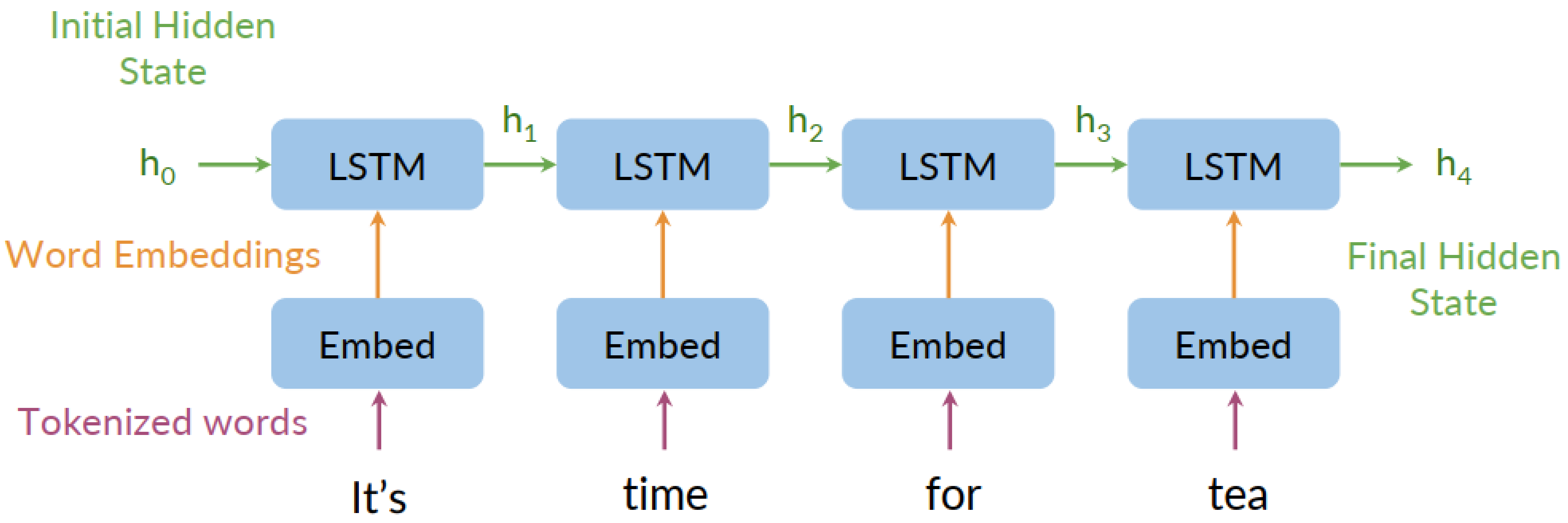


Figure from deeplearning.ai

Sequence-to-sequence



Encodes the overall meaning of the sentence

Figure from deeplearning.ai

Sequence-to-sequence

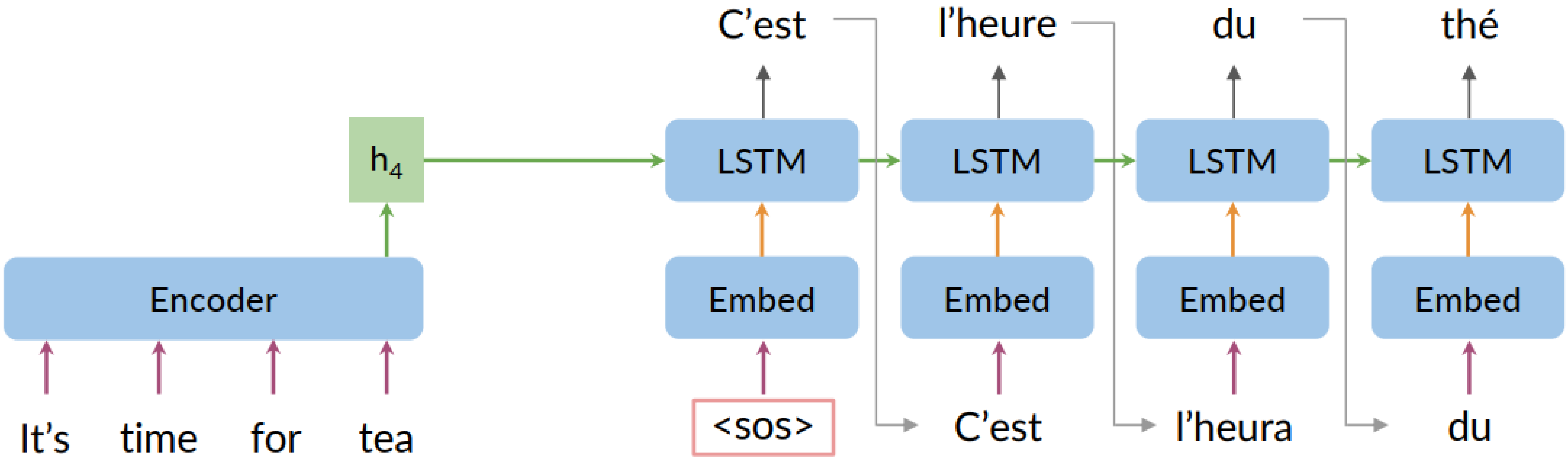


Figure from deeplearning.ai

Major limitation

The information bottleneck

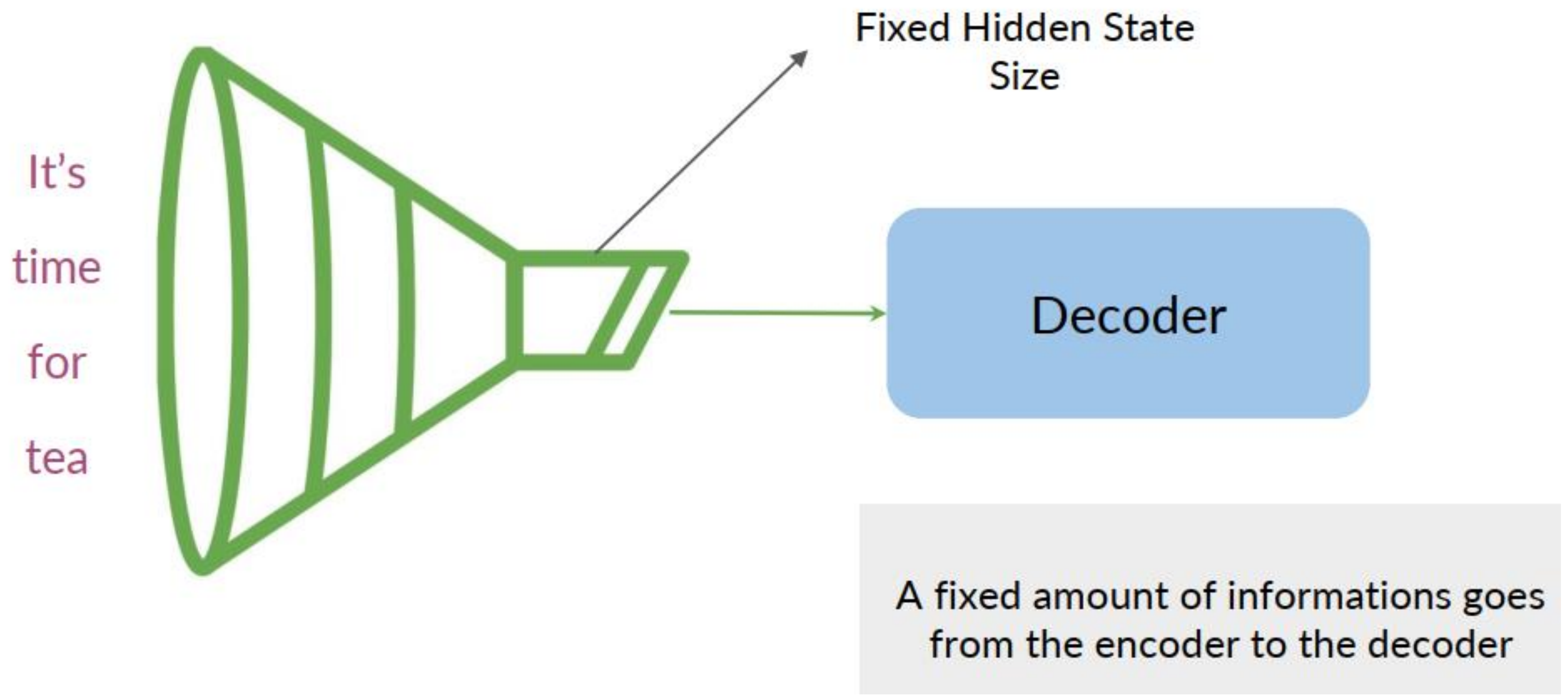
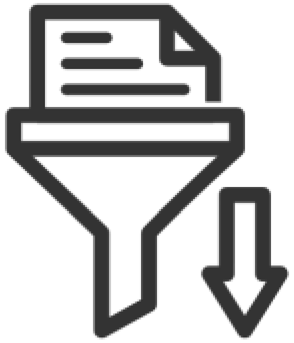


Figure from deeplearning.ai

Major limitation

- Variable-length sentences + fixed-length memory =

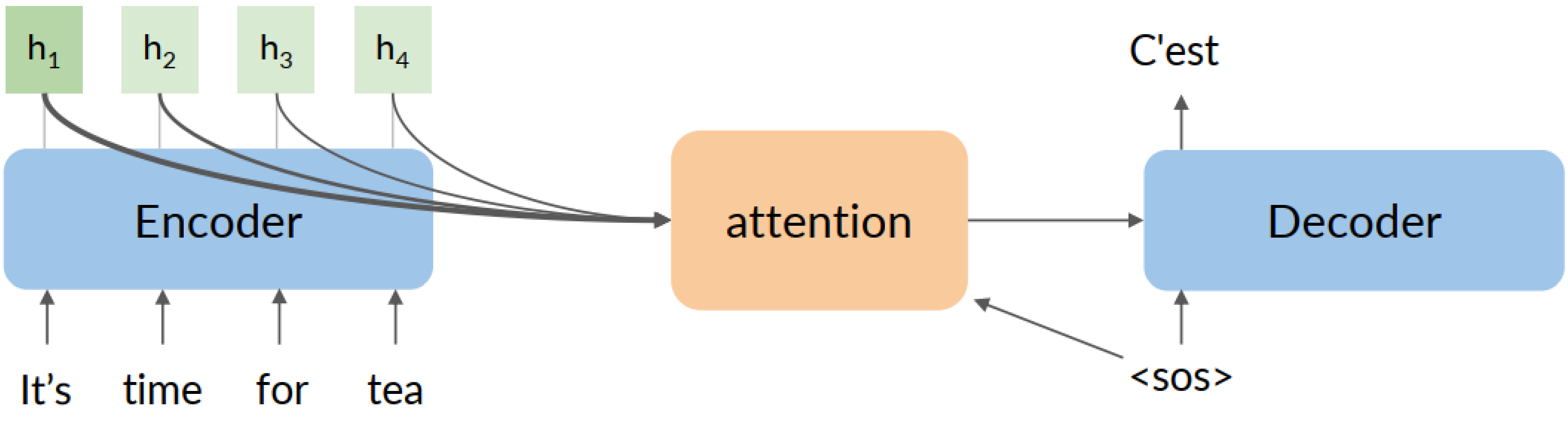


- As sequence size increases, model performance decreases

What is the solution?

Figure from deeplearning.ai

Solution: focus attention in the right place



The model can focus on specific hidden states at every step

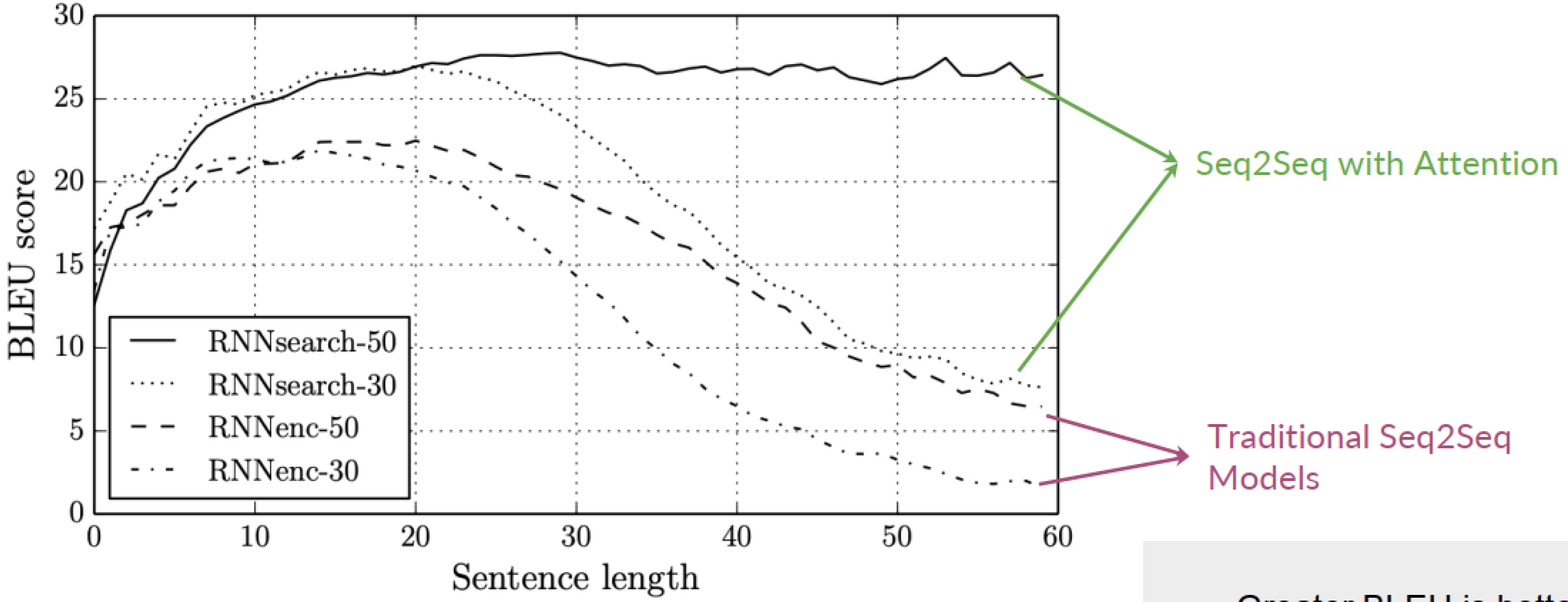
Figure from deeplearning.ai

Seq-to-Seq model with attention

NEURAL MACHINE TRANSLATION
BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal



Greater BLEU is better

Figure from deeplearning.ai

Traditional Seq-to-Seq Models

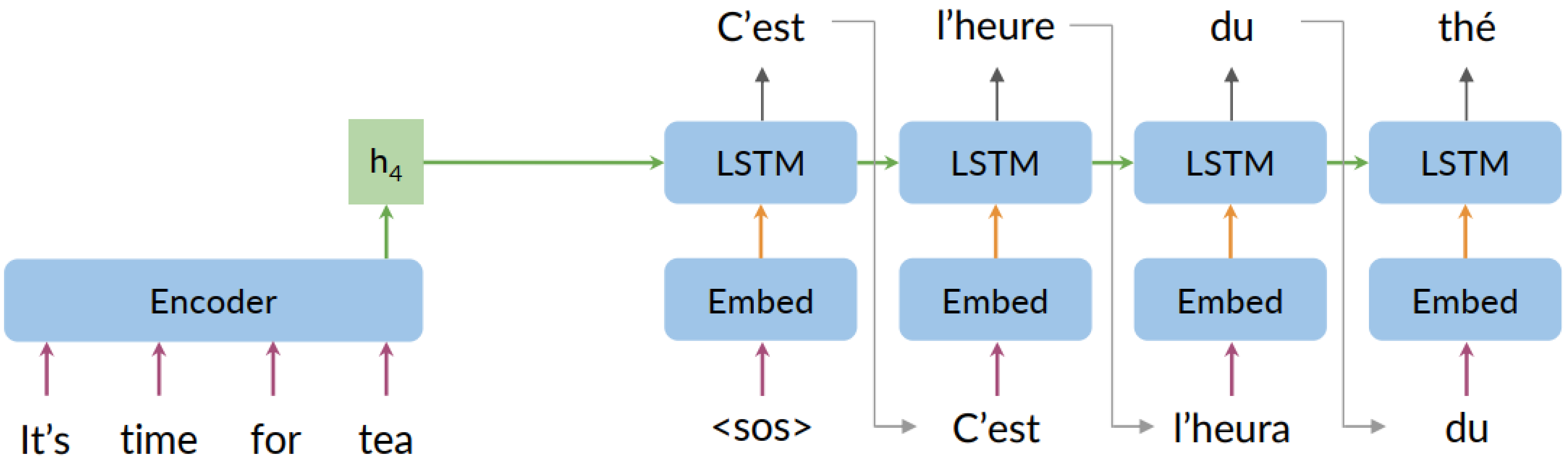


Figure from deeplearning.ai

How to use all the hidden states?

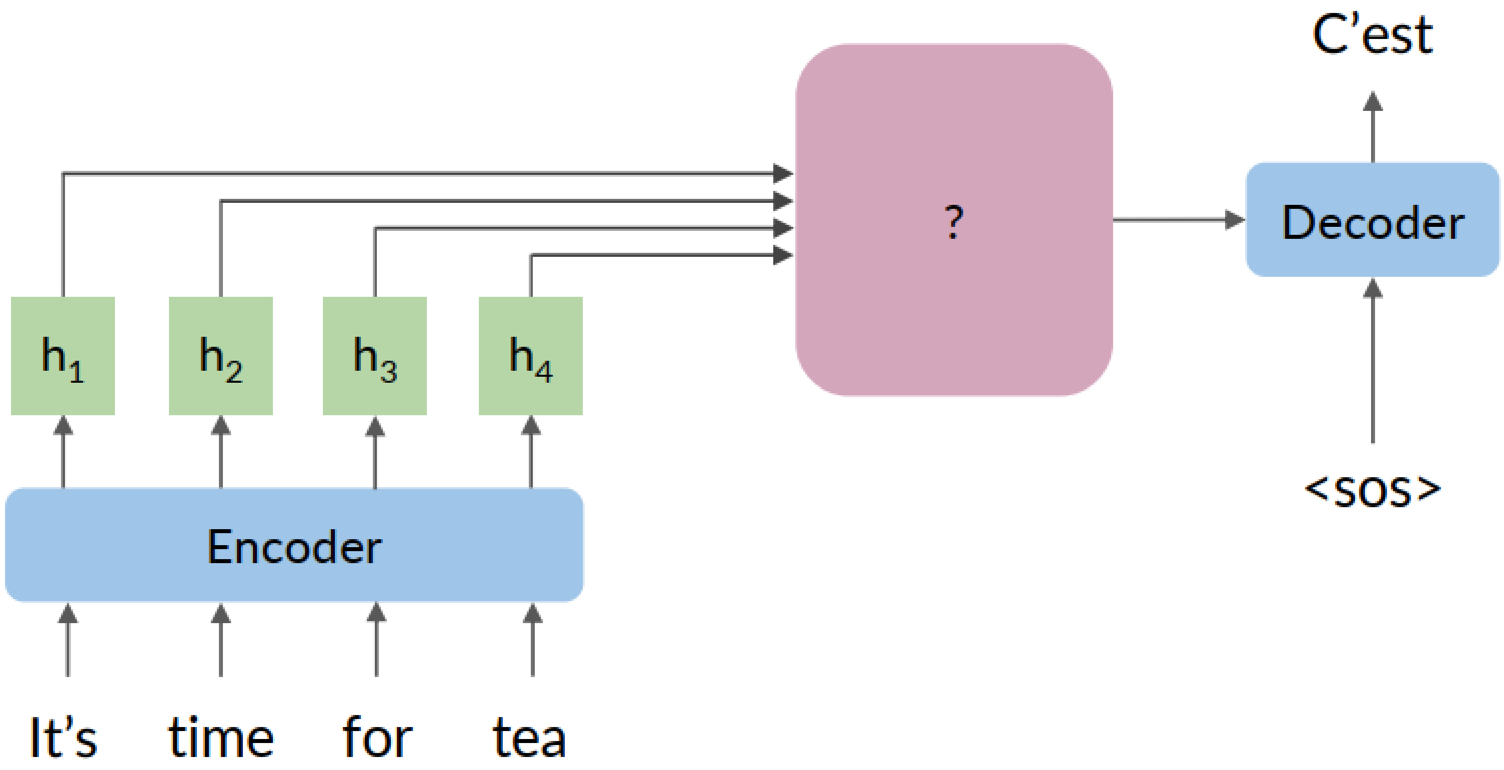
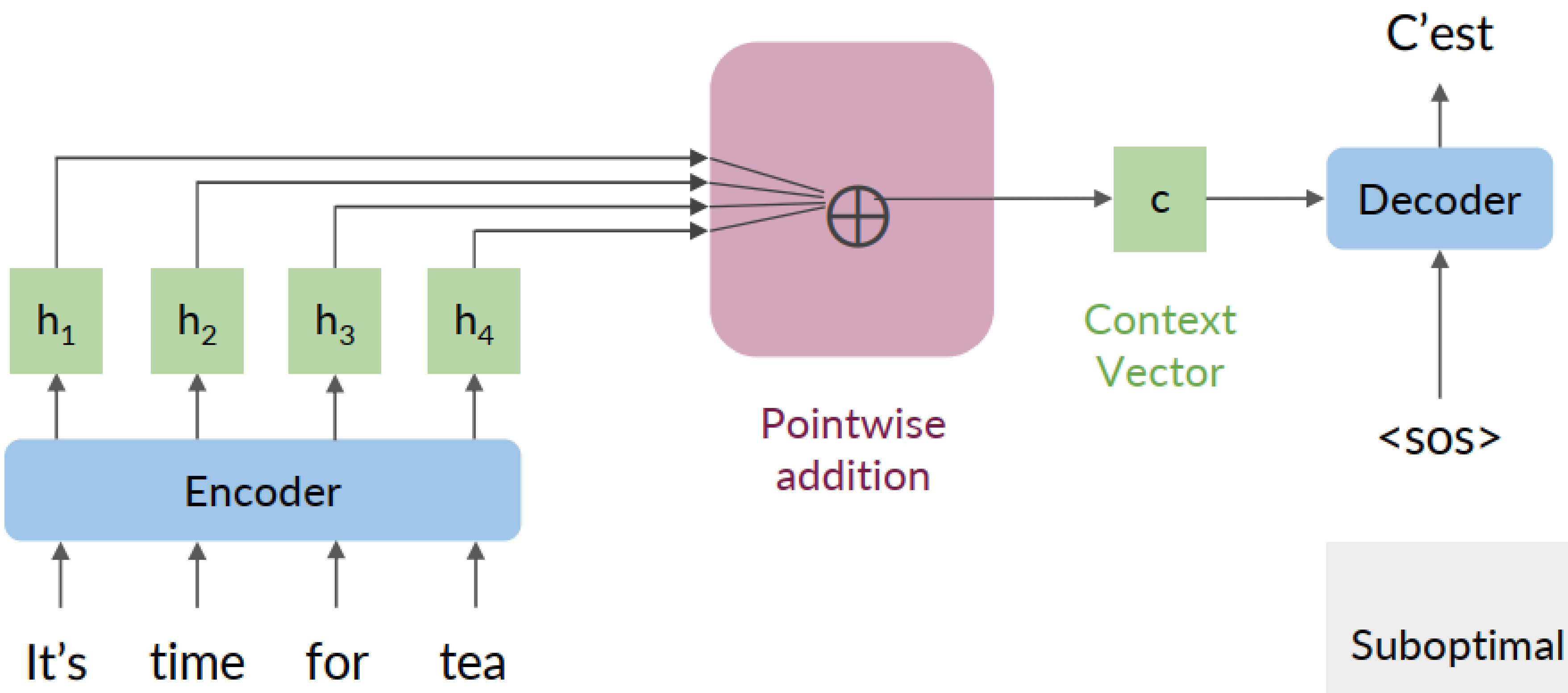


Figure from deeplearning.ai

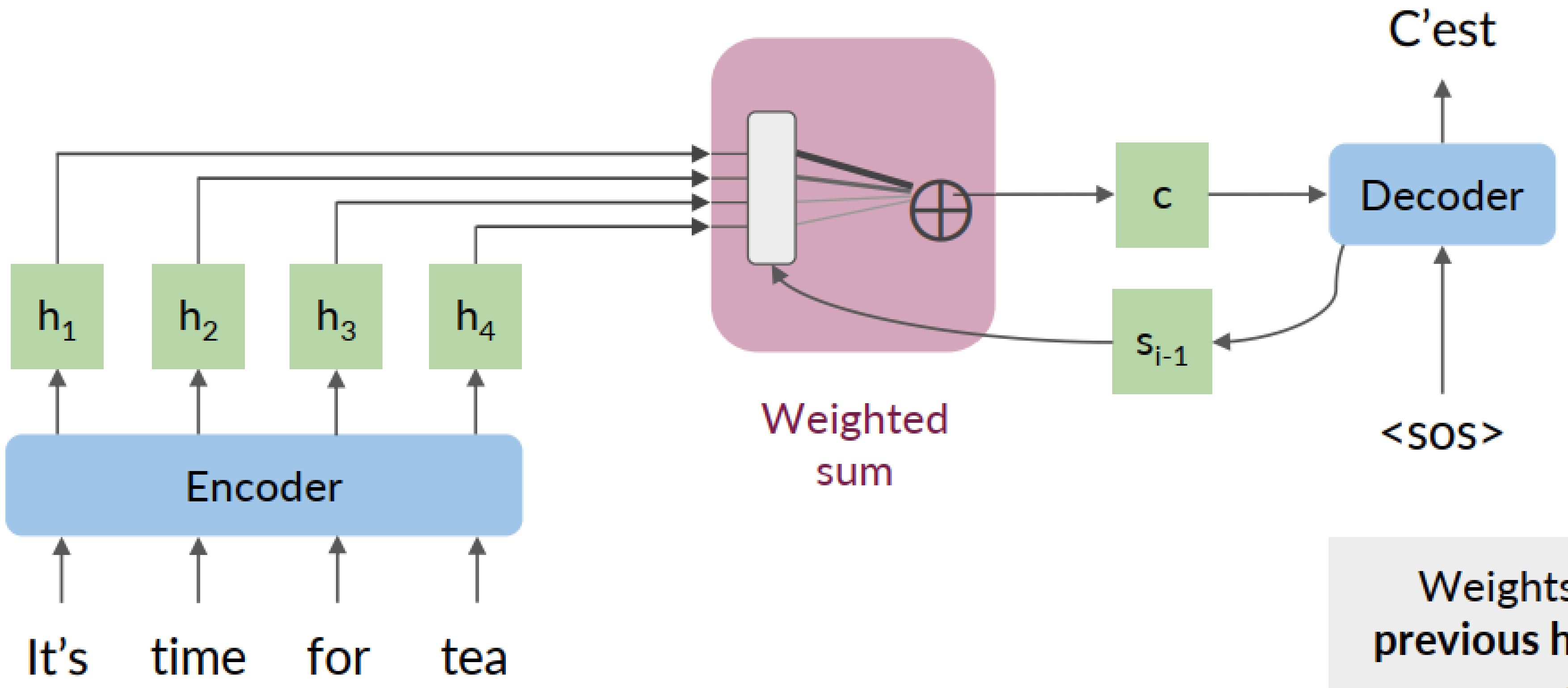
How to use all the hidden states?



Suboptimal use of information

Figure from deeplearning.ai

How to use all the hidden states?



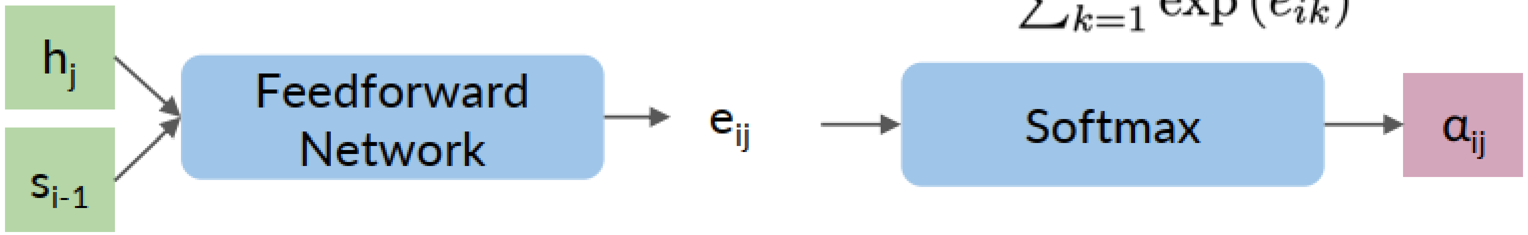
Weights depend on the **previous hidden state** in the decoder

Figure from deeplearning.ai

The attention layer in more depth

$$e_{ij} = a(s_{i-1}, h_j)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$



Weights used for the sum of hidden states

Learnable parameters

$$c_i = \sum_{j=1}^{T_x} \boxed{\alpha_{ij}}\boxed{h_j}$$

$\alpha_{i1}h_1 + \alpha_{i2}h_2 + \alpha_{i3}h_3 + \cdots + \alpha_{iM}h_M \longrightarrow c_i$

Context Vector is an expected value

Figure from deeplearning.ai

Queries, Keys, values and Attention

Figure from deeplearning.ai

Queries, Keys, Values

Query

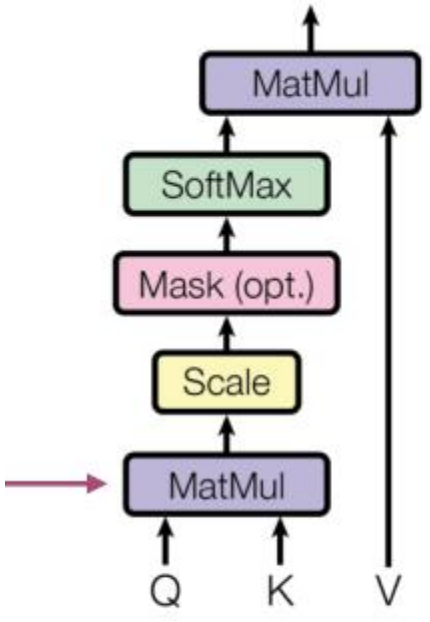
l'heure

Key	Value
It's	[0.5, 0.2, -1.2, ...,]
time	[0.2, -0.7, 0.9, ...,]
for	[1.3, 0.3, 0.8, ...,]
tea	[-0.4, 0.6, -1.1, ...,]

Similarity is used in for weighted sum

Figure from deeplearning.ai

Scaled dot-product attention



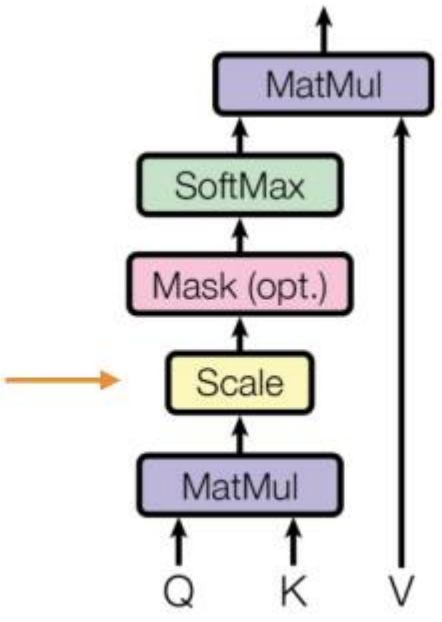
(Vaswani et al., 2017)

Similarity Between
Q and K

$$\mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

Figure from deeplearning.ai

Scaled dot-product attention



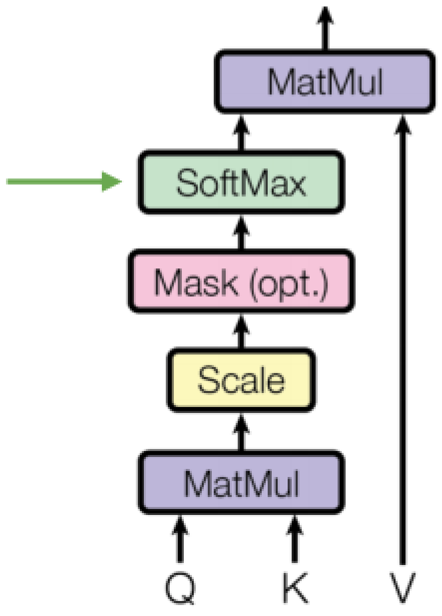
(Vaswani et al., 2017)

$$\text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

Scale using the root of the key vector size

Figure from deeplearning.ai

Scaled dot-product attention



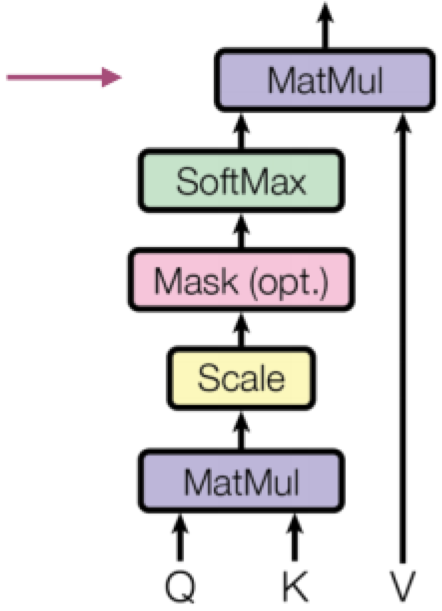
(Vaswani et al., 2017)

$$\mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

Weights for the weighted sum

Figure from deeplearning.ai

Scaled dot-product attention



(Vaswani et al., 2017)

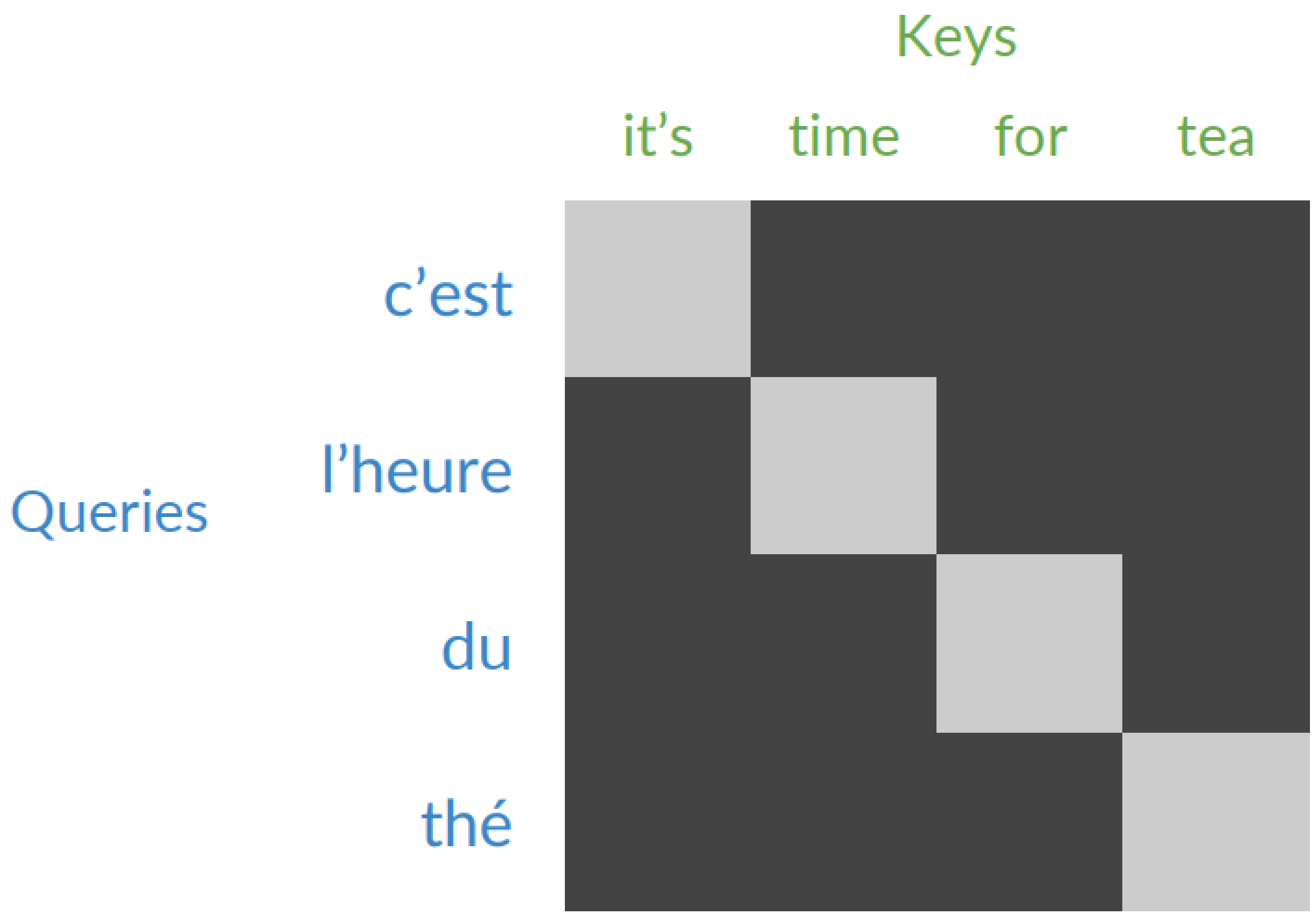
$$\mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

Weighted sum of values V

Just two matrix multiplications and a Softmax!

Figure from deeplearning.ai

Alignment Weights

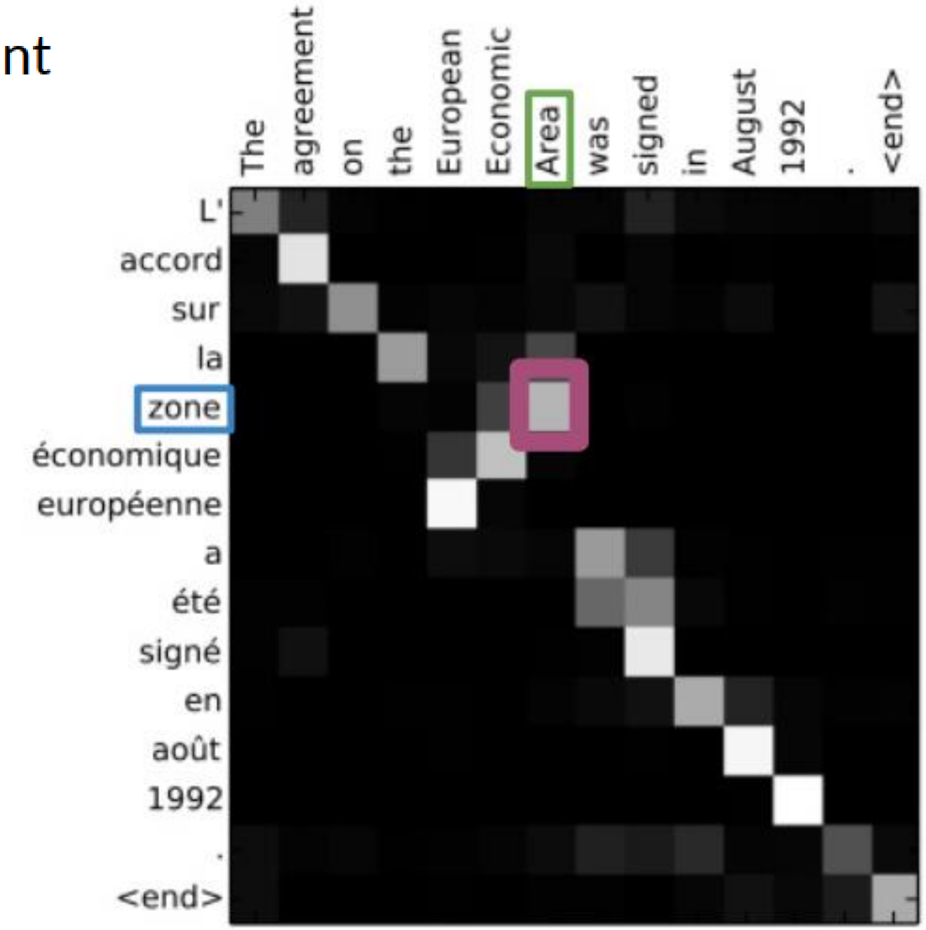


Similar words have large weights

Figure from deeplearning.ai

Alignment Weights

Works for languages with different grammar structures!



Bahdanau et al., 2015

Figure from deeplearning.ai

Traditional seq2seq models

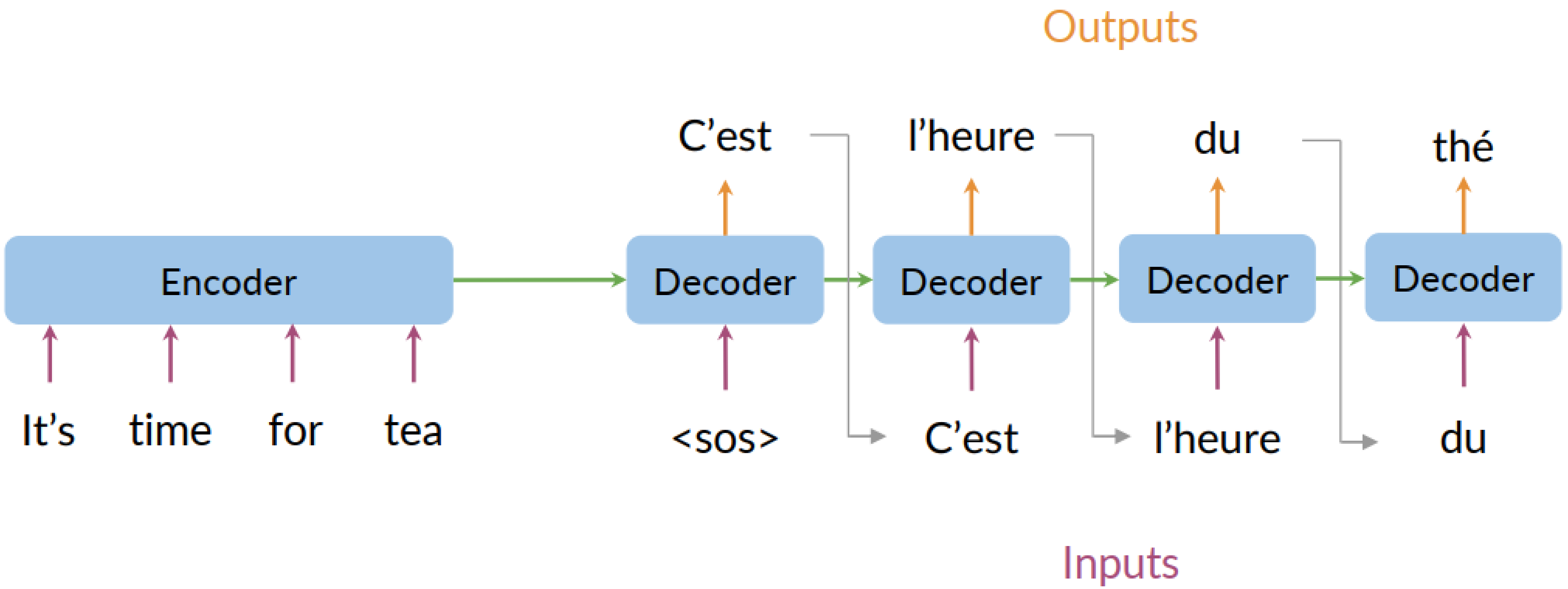


Figure from deeplearning.ai

Teacher Forcing

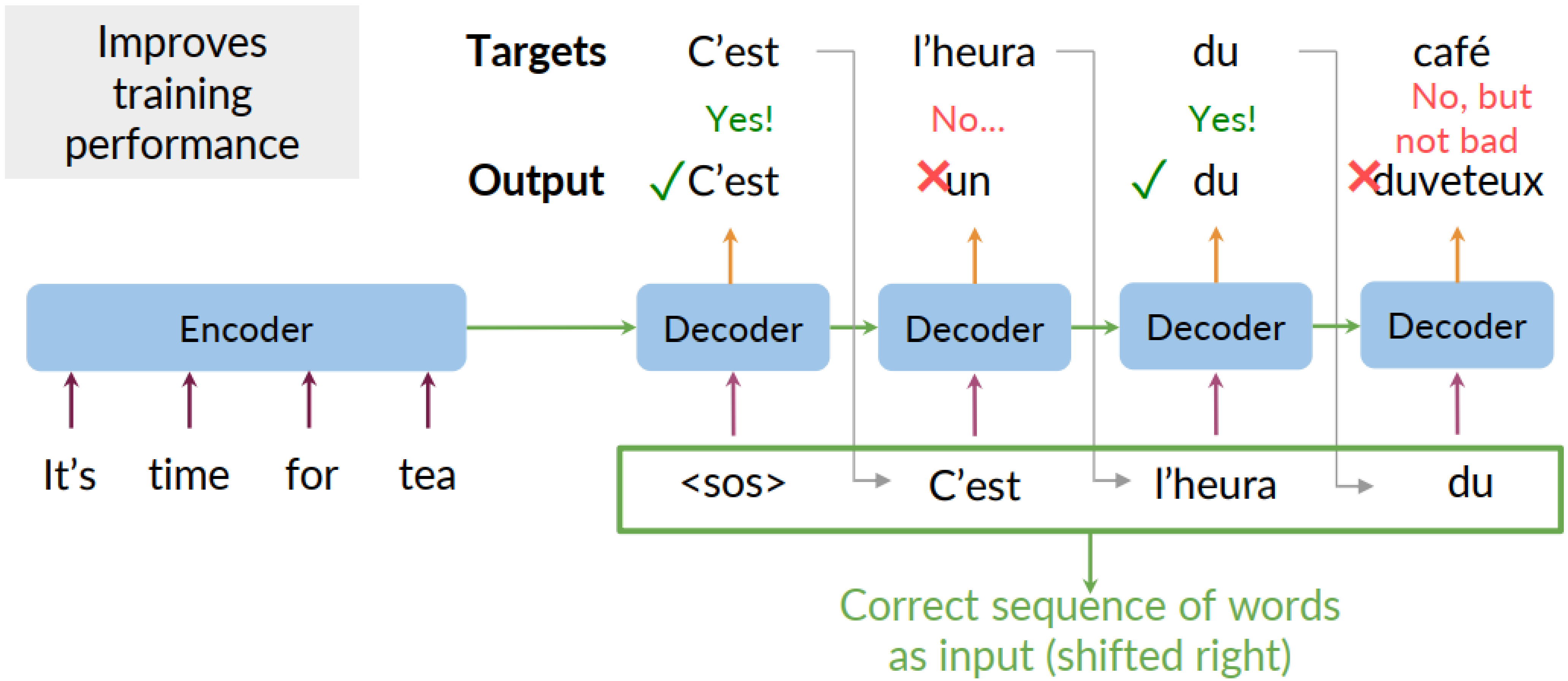
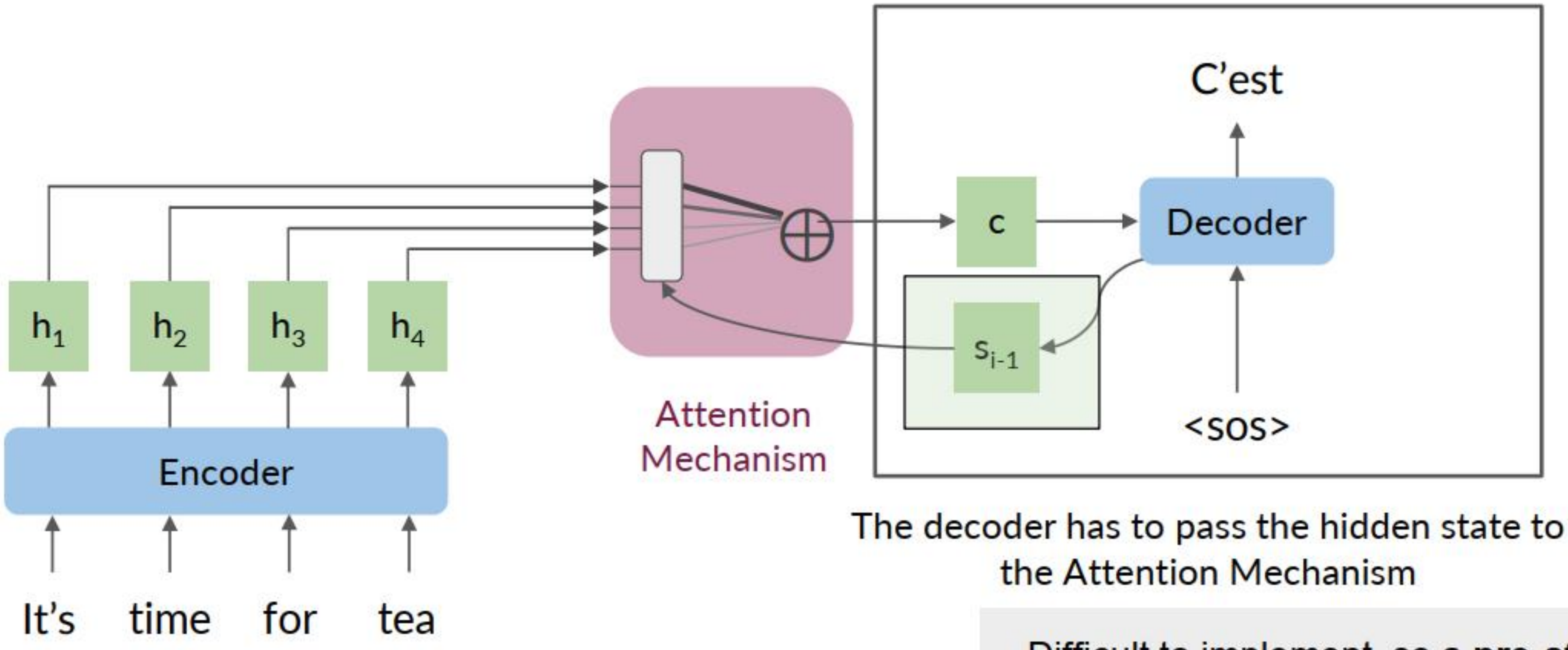


Figure from deeplearning.ai

NMT Model



The decoder has to pass the hidden state to the Attention Mechanism

Difficult to implement, so a **pre-attention decoder** is introduced.

Figure from deeplearning.ai

NMT Model

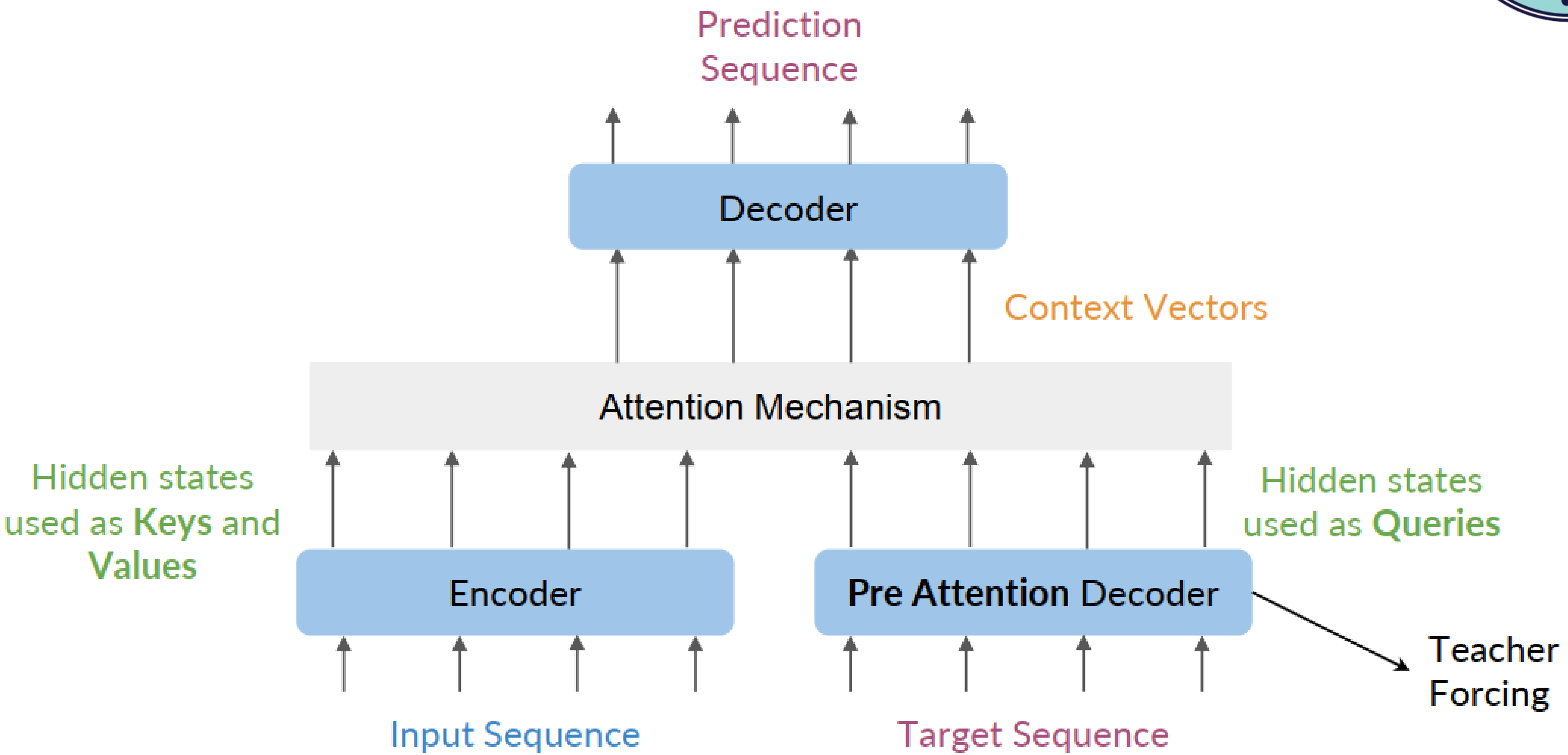


Figure from deeplearning.ai

Bleu Score

BiLingual Evaluation Understudy

Compares candidate translations to reference (human) translations

The closer to 1, the better

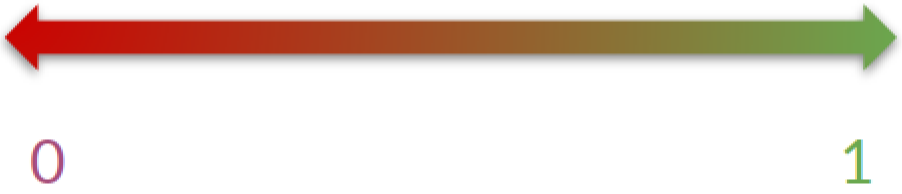


Figure from deeplearning.ai

Bleu Score

Candidate	I	I	am	I	
Reference 1	Younes	said	I	am	hungry
Reference 2	He	said	I	am	hungry

How many words from the candidate appear in the reference translations?

Figure from deeplearning.ai

Bleu Score

Candidate	$\boxed{\text{I}}$	$\boxed{\text{I}}$	$\boxed{\text{am}}$	$\boxed{\text{I}}$	
Reference 1	Younes	said	I	am	hungry
Reference 2	He	said	I	am	hungry

Count: $\frac{1+1+1+1}{4} = 1$

A model that always outputs common words will do great!

Figure from deeplearning.ai

Bleu Score (Modified Precision)

Candidate	$\boxed{\text{I}}$	$\boxed{\text{I}}$	$\boxed{\text{am}}$	$\boxed{\text{I}}$	
Reference 1	Younes	said			hungry
Reference 2	He	said			hungry

Count: $\frac{1+1}{4} = 0.5$

Better than the previous implementation version!

Figure from deeplearning.ai

Bleu Score (Modified Precision): unigram to 4-gram

- **Target Sentence:** The guard arrived late because it was raining
- **Predicted Sentence:** The guard arrived late because of the rain

Step 1: Compute BLEU scores from 1-gram to 4-grams

Step 2: Compute Geometric Average Precision

Step 3: Compute Brevity Penalty

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Precision 1-gram

Precision 1-gram = Number of correct predicted 1-grams / Number of total predicted 1-grams

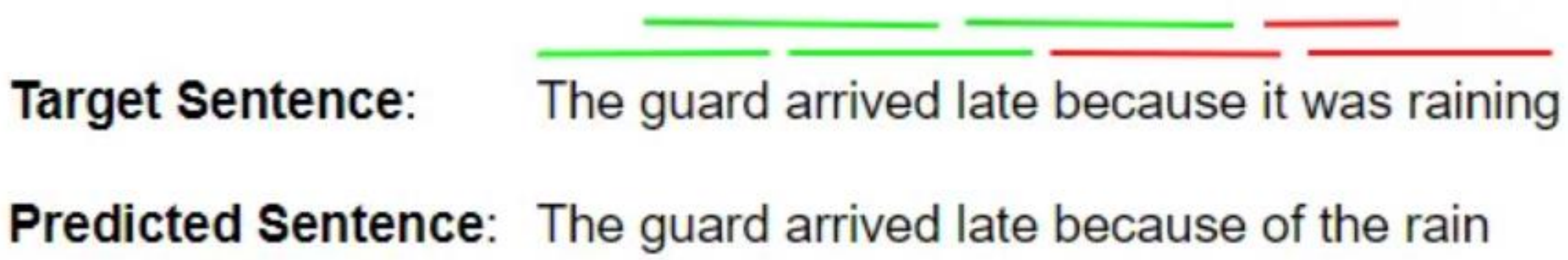
Precision(Image by Author)

So, Precision 1-gram (p_1) = 5 / 8

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Precision 2-gram

Precision 2-gram = Number of correct predicted 2-grams / Number of total predicted 2-grams



Precision 2-gram (Image by Author)

So, Precision 2-gram (p_2) = 4 / 7

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Precision 3-gram

Similarly, Precision 3-gram (p_3) = 3 / 6

Target Sentence: The guard arrived late because it was raining

Predicted Sentence: The guard arrived late because of the rain

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Precision 4-gram

And, Precision 4-gram (p_4) = 2 / 5

Target Sentence: The guard arrived late because it was raining

Predicted Sentence: The guard arrived late because of the rain

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Geometric Average Precision (GAP)

$$\begin{aligned} Geometric\ Average\ Precision\ (N) &= exp(\sum_{n=1}^{N} w_n \log p_n) \\ &= \prod_{n=1}^{N} p_n^{w_n} \\ &= (p_1)^{\frac{1}{4}} \cdot (p_2)^{\frac{1}{4}} \cdot (p_3)^{\frac{1}{4}} \cdot (p_4)^{\frac{1}{4}} \end{aligned}$$

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Brevity Penalty

- Sometimes, for longer sentences the candidates might be very small and missing important information relative to the reference.

Reference: "Transformers make everything quick and efficient through parallel computation of self-attention heads“

Candidate: "Transformers make everything quick and efficient"

- Here, we are missing information because of the short prediction. But, the GAP is high (1.0) as the additional words that are in the reference but not in the candidate are not being considered.

https://medium.com/@priyankads/evaluation-metrics-in-natural-language-processing-bleu-dc3cfa8faaa5

Brevity Penalty

- The Brevity Penalty penalizes sentences that are too short

$$Brevity\ Penalty = \begin{cases} 1, & \text{if } c > r \\ e^{(1-r/c)}, & \text{if } c <= r \end{cases}$$

- c is *predicted length = number of words in the predicted sentence* and
- r is *target length = number of words in the target sentence*

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

Brevity Penalty

Reference: "Transformers make everything quick and efficient through parallel computation of self-attention heads“

Candidate: "Transformers make everything quick and efficient"

BP = exp(1 - (6/12)) = 0.37

https://medium.com/@priyankads/evaluation-metrics-in-natural-language-processing-bleu-dc3cfa8faaa5

BLUE score

BLUE = Brevity Penalty * Geometric Average Precision

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

https://medium.com/@priyankads/evaluation-metrics-in-natural-language-processing-bleu-dc3cfa8faaa5

Corpus BLUE

```python
import evaluate
bleu = evaluate.load('bleu')
predictions = ["Transformers Transformers are fast plus efficient",
               "Good Morning", "I am waiting for new Transformers"]
references = [
              ["HuggingFace Transformers are quick, efficient and awesome",
               "Transformers are awesome because they are fast to execute"],
              ["Good Morning Transformers", "Morning Transformers"],
              ["People are eagerly waiting for new Transformer models",
               "People are very excited about new Transformers"]

]
results = bleu.compute(predictions=predictions, references=references,
          max_order = 2)
print(results)
```

```
{'bleu': 0.5037930378757725,
'precisions': [0.7142857142857143, 0.5454545454545454],
'brevity_penalty': 0.8071177470053892, 'length_ratio': 0.8235294117647058,
'translation_length': 14, 'reference_length': 17}
```

Corpus BLUE

```
predictions = ["Transformers Transformers are fast plus efficient",
               "Good Morning", "I am waiting for new Transformers"]
references = [
              ["HuggingFace Transformers are quick, efficient and awesome",
               "Transformers are awesome because they are fast to execute"],
              ["Good Morning Transformers", "Morning Transformers"],
              ["People are eagerly waiting for new Transformer models",
               "People are very excited about new Transformers"]
```

- To calculate BP, the total candidate length is 14(c) and the effective reference length is 17(r).
- The effective reference length is calculated by summing up the n-grams in the references that are closer to the candidate. Here, for Reference-1 first sentence is selected with 8 tokens, Reference-2 and 3 have 2 and 7 tokens in their second sentences.

https://medium.com/@priyankads/evaluation-metrics-in-natural-language-processing-bleu-dc3cfa8faaa5

Corpus BLUE

```
{'bleu': 0.5037930378757725,
'precisions': [0.7142857142857143, 0.5454545454545454],
'brevity_penalty': 0.8071177470053892, 'length_ratio': 0.8235294117647058,
'translation_length': 14, 'reference_length': 17}
```

```
Total MP = (0.7142857142857143)^0.5 * (0.5454545454545454)^0.5
Total MP = 0.6241878

Since c < r,

BP = exp(1 - (17/14)) =  0.8071177

BLEU = BP * Total MP = 0.8071177 * 0.6241878 = 0.503793

Rounded BLEU score = 0.5
```

https://medium.com/@priyankads/evaluation-metrics-in-natural-language-processing-bleu-dc3cfa8faaa5

Advantages of BLEU score

- ✓ It is quick to calculate and easy to understand.
- ✓ BLEU score has been seen to have a high correlation with the human judgement of the prediction quality.
- ✓ Importantly, it is language-independent making it straightforward to apply to your NLP models.

Disadvantages of BLEU score

- Synonyms of the n-grams are not considered until and unless they are present as one of the references. This is because the meaning of the n-grams is not being taken into account. For example, “Transformers are quick and efficient” and “Transformers have fast execution time” have a similar meaning but the BLEU-1 score is only 0.2.

- It looks only for exact word matches. Sometimes a variant of the same word can be used eg. “rain” and “raining”, but Bleu Score counts that as an error.

Disadvantages of BLEU score

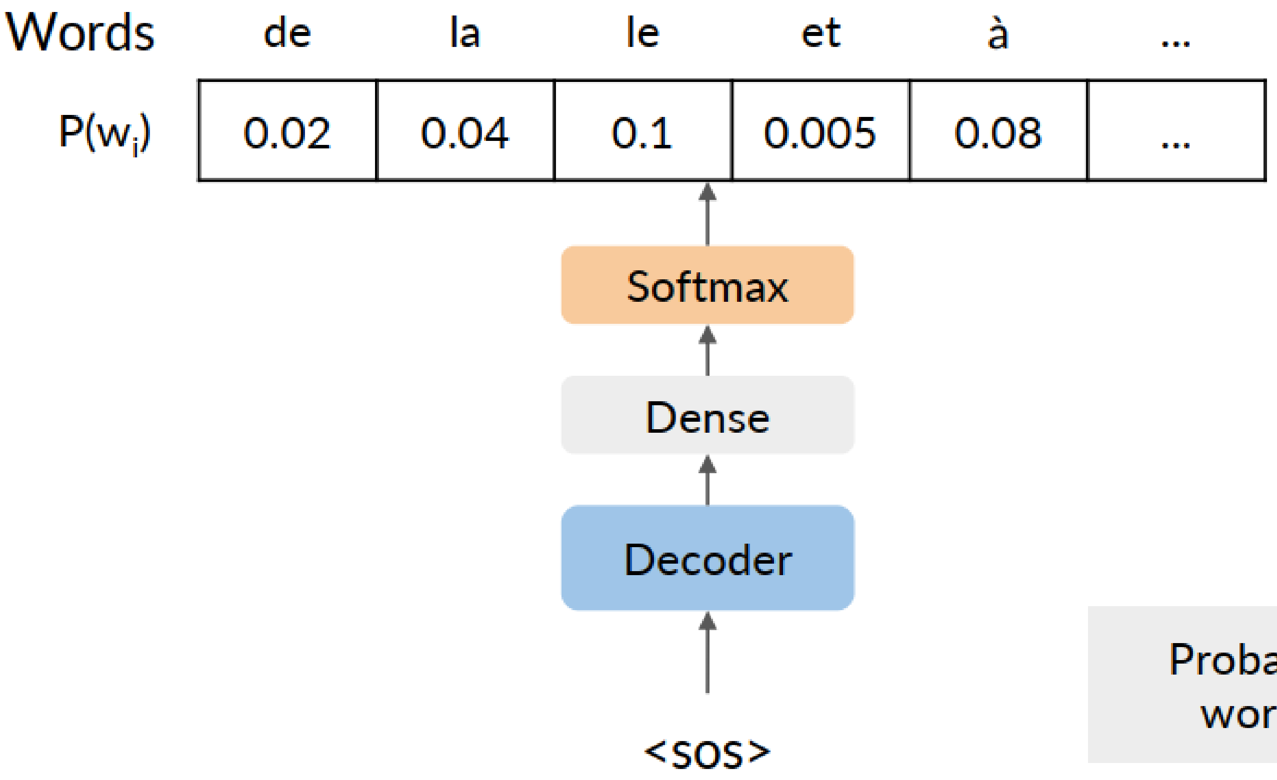
- The problem of word order cannot be solved using higher-order n-grams alone.

  - For example, the candidate "quick and efficient Transformers are" with reference **"Transformers are quick and efficient"** gives a high BLEU-2 score (0.87) but the translation cannot be considered right.

Different Decoding Methods for Text Generation

- Greedy Search
- Beam Search

Seq2Seq model



Probability distribution over words in target language

Figure from deeplearning.ai

Greedy Search Decoding

Selects the most probable word at each step

But the best word at each step may not be the best for longer sequences...

Can be fine for shorter sequences, but limited by inability to look further down the sequence

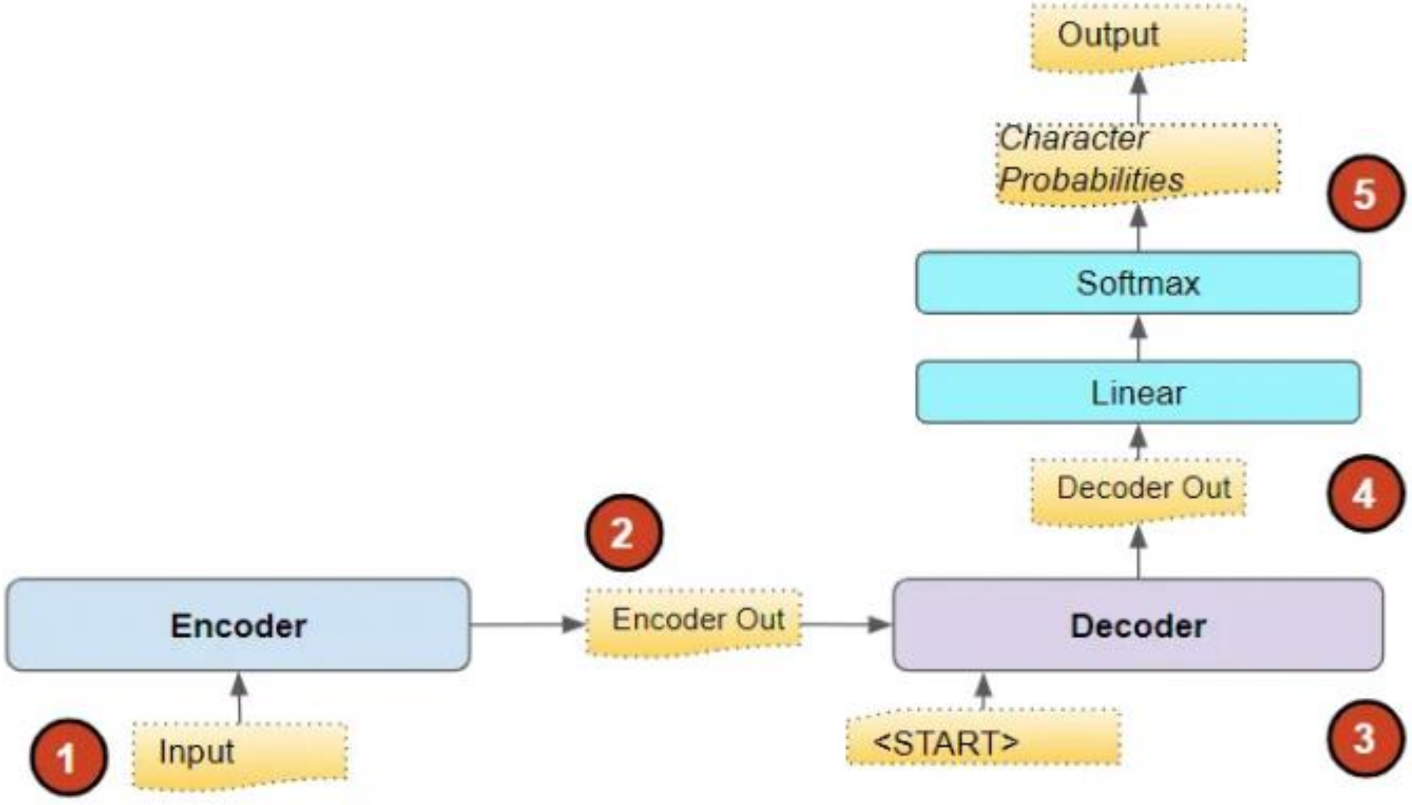
J'ai faim.

I am hungry.

I am, am, am, am...

Figure from deeplearning.ai

Greedy Search Decoding



Sequence-to-Sequence Model for Machine Translation (Image by Author)

https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24

Greedy Search Decoding

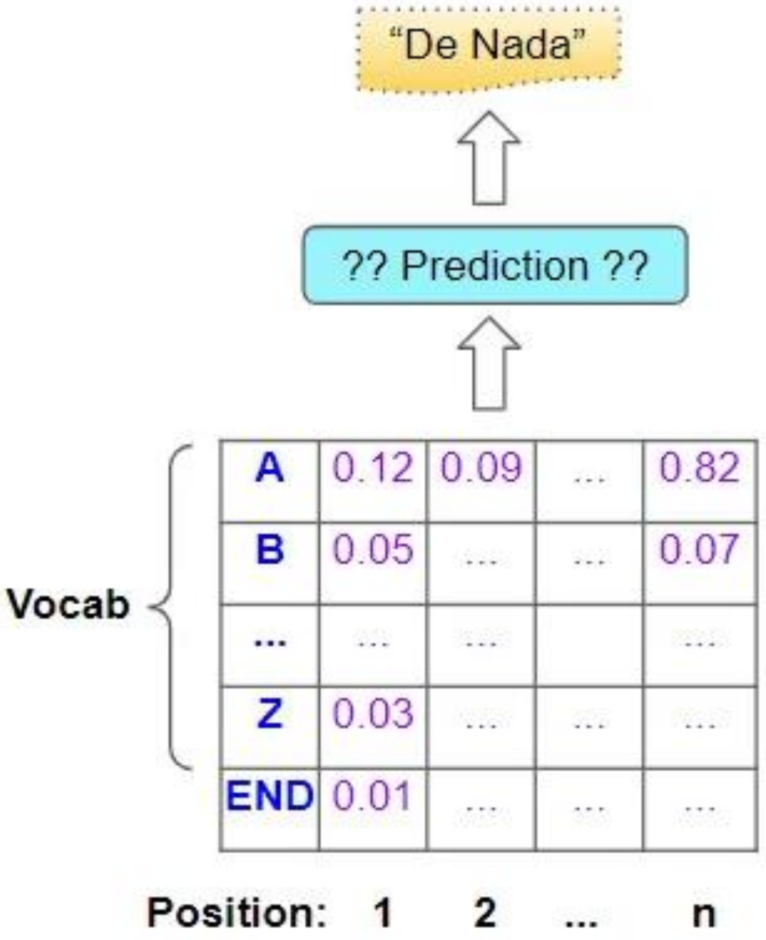
Position:	1	2	...	n
A	0.12	0.09	...	0.82
B	0.05	...	...	0.07
...	...	...		...
Z	0.03	...	...	...
END	0.01	...	...	...

Vocab

Probabilities for each character in the vocabulary, for each position in the output sequence

https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24

Greedy Search Decoding

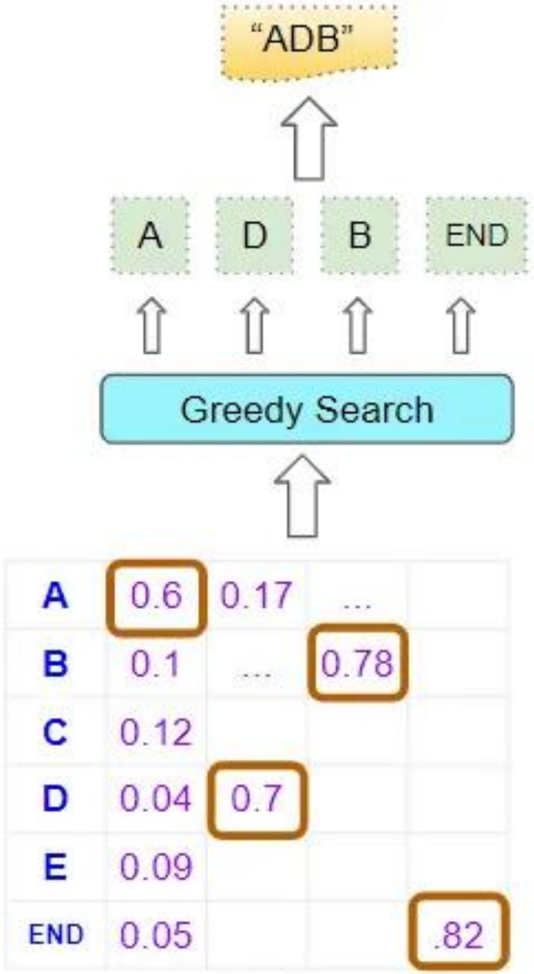


The model predicts an output sentence based on the probabilities

How does it do that?

https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24

Greedy Search Decoding



✓ Simply take the word that has the highest probability at each position and predict that

https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24

Beam Search Decoding

Most probable translation **is not** the one with the most probable word at each step

$\big\downarrow$ Solution

Calculate probability of multiple possible sequences

$\big\downarrow$

Beam search

Figure from deeplearning.ai

Beam Search Decoding

Probability of multiple possible sequences at each step

Beam width B determines number of sequences you keep

Until all B most probable sequences end with <EOS>

Beam search with **B=1** is **greedy decoding.**

Figure from deeplearning.ai

Beam Search Example

B = 2

P(w_1 | “<sos>”)

<sos>

I	0.5
am	0.4
hungry	0.1
<eos>	0.0

Figure from deeplearning.ai

Beam Search Example

B = 2

<sos>

$P(w_1 \mid \text{"<sos>"})$

I	0.5
am	0.4

$P(w_2 \mid \text{"I"})$

I	0.1
am	0.5
hungry	0.3
<eos>	0.1

$P(w_2 \mid \text{"I"})P(\text{"I"})$

I I	0.05
I am	0.25
I hungry	0.15
I <eos>	0.05

$P(w_2 \mid \text{"am"})$

I	0.7
am	0.05
hungry	0.2
<eos>	0.05

$P(w_2 \mid \text{"am"})P(\text{"am"})$

am I	0.28
am am	0.02
am hungry	0.08
am <eos>	0.02

Calculate the conditional probability of all other words given the two sequences that you have kept so far I and am.

Figure from deeplearning.ai

Beam Search Example

B = 2

<sos>

$P(w_1 \mid$ “<sos>”$)$

I	0.5
am	0.4

$P(w_2 \mid$ “I”$)$

I	0.1
am	0.5
hungry	0.3
<eos>	0.1

$P(w_2 \mid$ “I”$)P($“I”$)$

I I	0.05
I am	0.25
I hungry	0.15
I <eos>	0.05

$P(w_3 \mid$ “I am”$)$

I	0.1
am	0.05
hungry	0.6
<eos>	0.3

$P(w_2 \mid$ “am”$)$

I	0.7
am	0.05
hungry	0.2
<eos>	0.05

$P(w_2 \mid$ “am”$)P($“am”$)$

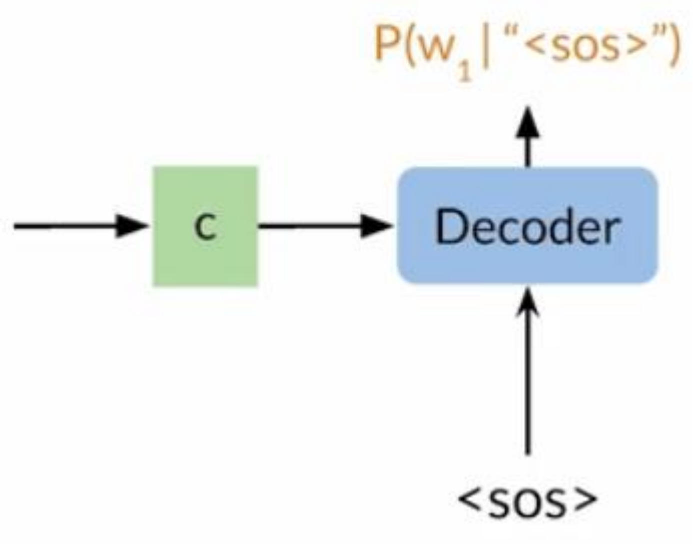
am I	0.28
am am	0.02
am hungry	0.08
am <eos>	0.02

$P(w_3 \mid$ “Am I”$)$

I	0.15
am	0.15
hungry	0.5
<eos>	0.2

Figure from deeplearning.ai

Beam Search Example



Select B most probable words

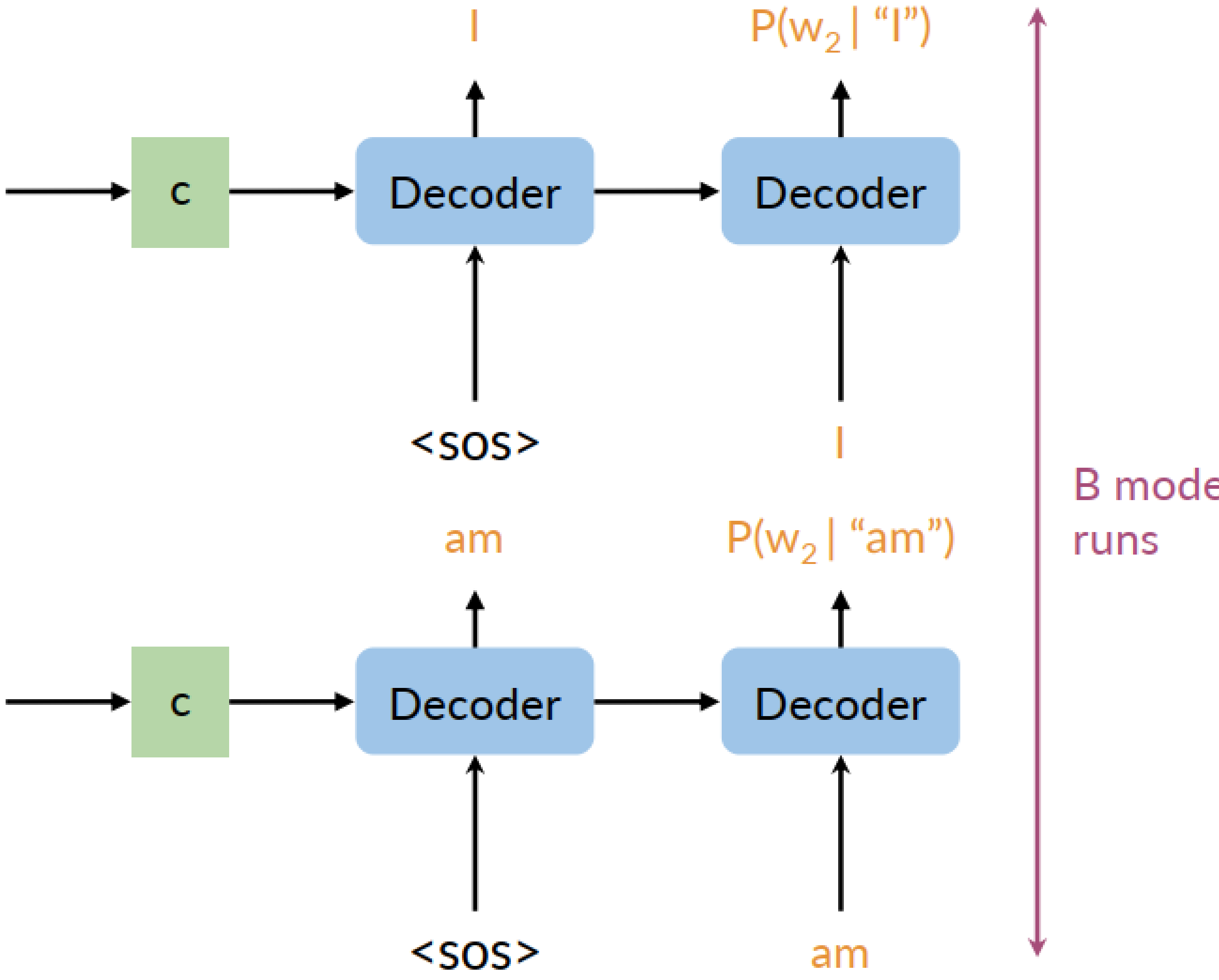


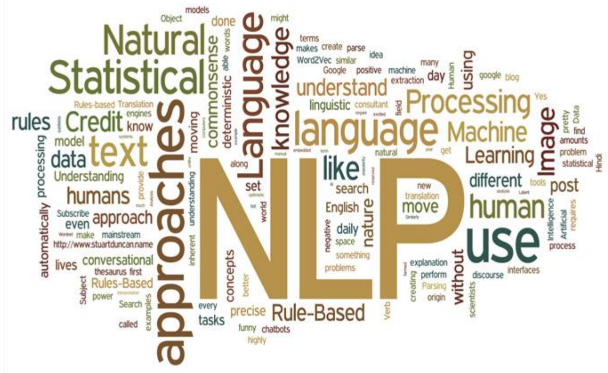
Figure from deeplearning.ai

Beam Search Decoding

Beam Search makes two improvements over Greedy Search:

- ✓ With Greedy Search, we took just the single best word at each position. In contrast, Beam Search expands this and takes the best “B” words.
- ✓ With Greedy Search, we considered each position in isolation. Once we had identified the best word for that position, we did not examine what came before it (i.e. in the previous position), or after it. In contrast, Beam Search picks the “B” best *sequences* so far and considers the probabilities of the combination of all of the preceding words along with the word in the current position.

https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24



با تشکر از توجه شما