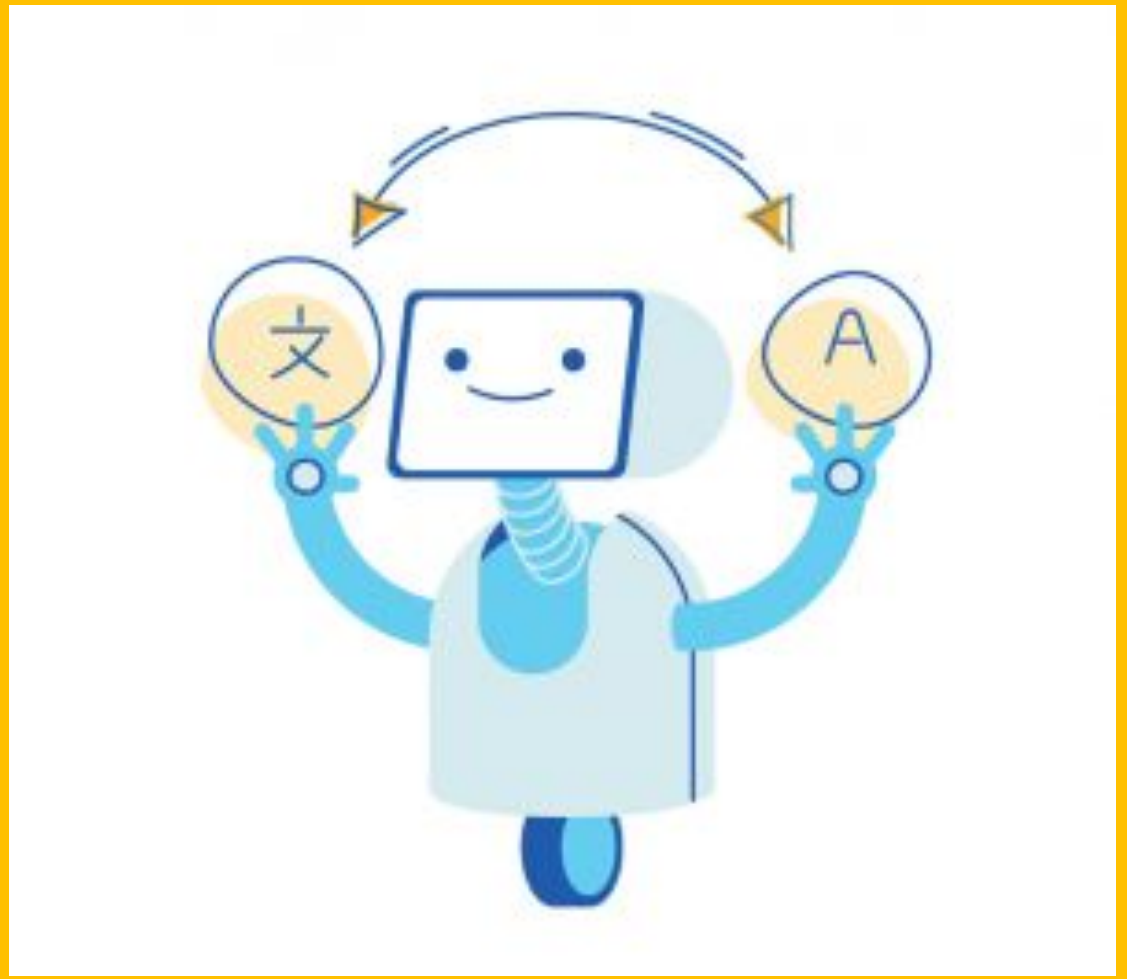
Natural Processing Language

Machine Translation

VÕ PHAN ANH QUÂN **MSSV: 2014285**

TRẦN CÔNG MINH QUÂN **MSSV: 2012528**



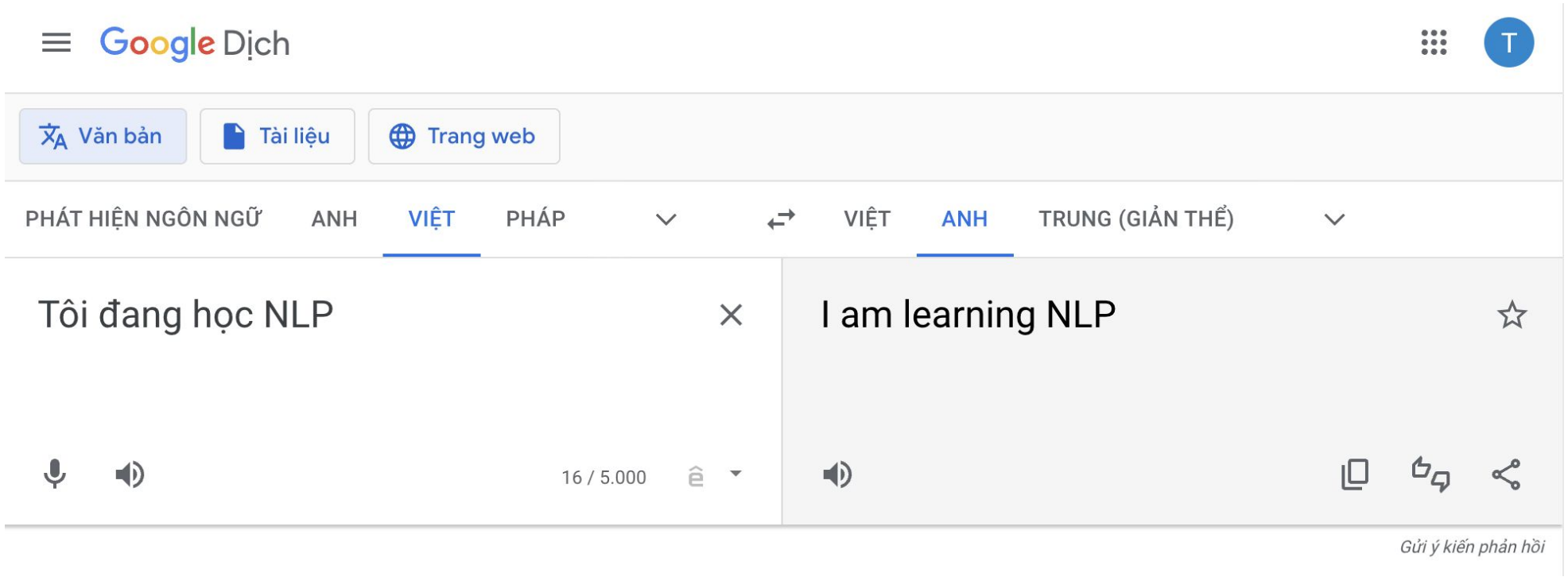
CONTENT

1 - Introduction

- Goal: Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**

1 – Introduction

❖ Goal: Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**



1 – Introduction

❖ Goal: Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**

- Can be formulated as an optimization problem:

$$\hat{w}^{(t)} = \underset{w^{(t)}}{\operatorname{argmax}}\, \theta(w^{(s)}, w^{(t)})$$

Where θ is a scoring function over source and target sentences

- Requires two components:
  - Learning algorithm to compute parameters of θ
  - Decoding algorithm for computing the best translation $\hat{w}^{(t)}$

1 – Introduction

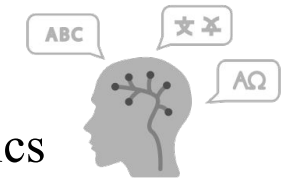
❖ Goal: Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**

Automatic Speech Recognition (ASR)

translation of spoken language into text

Natural Language Understanding (NLU)

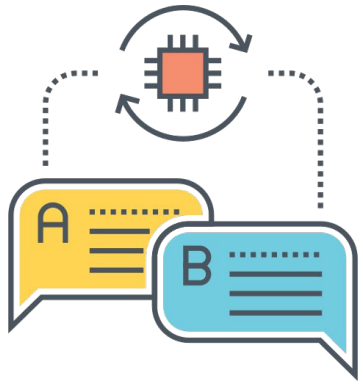
a computer's ability to understand language



- ❑ Syntax
- ❑ Semantics
- ❑ Phonology
- ❑ Pragmatics
- ❑ Morphology

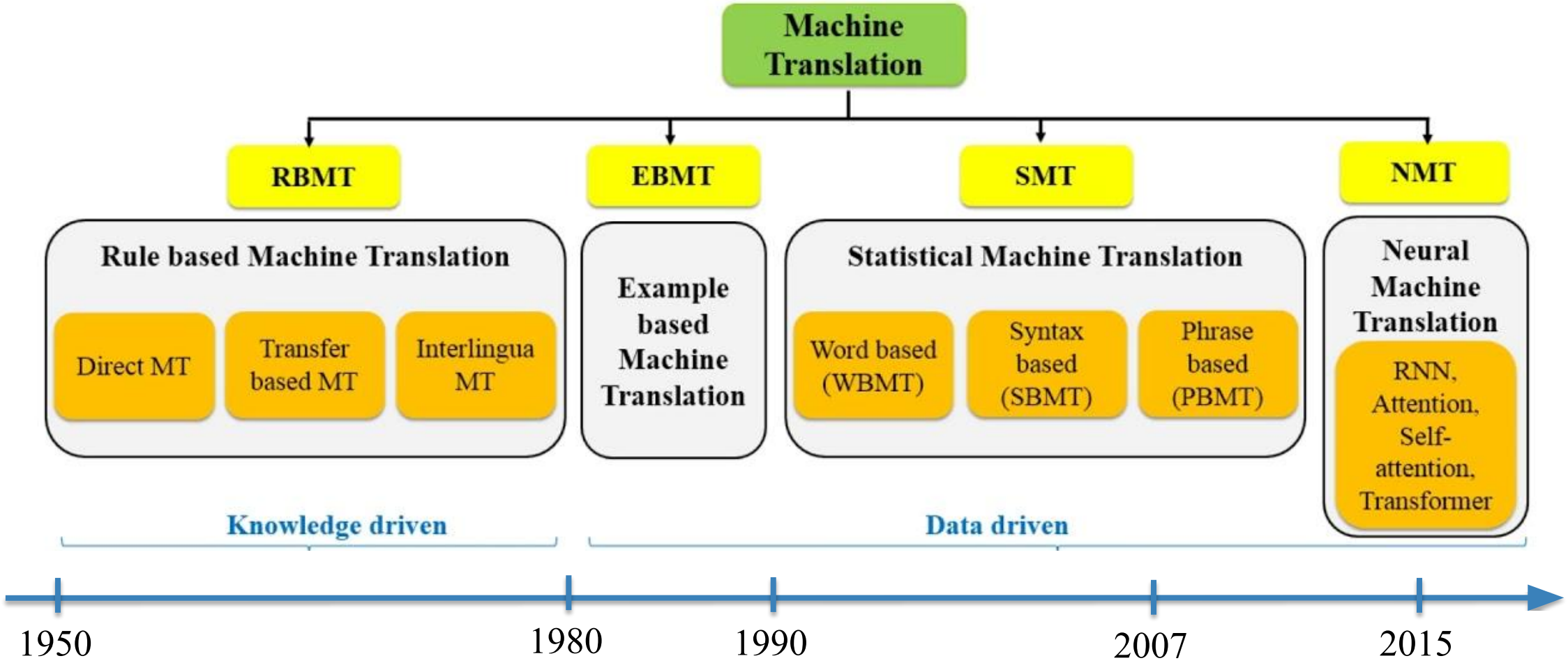
Natural Language Generation (NLG)

generate natural language by a computer



1 – Introduction

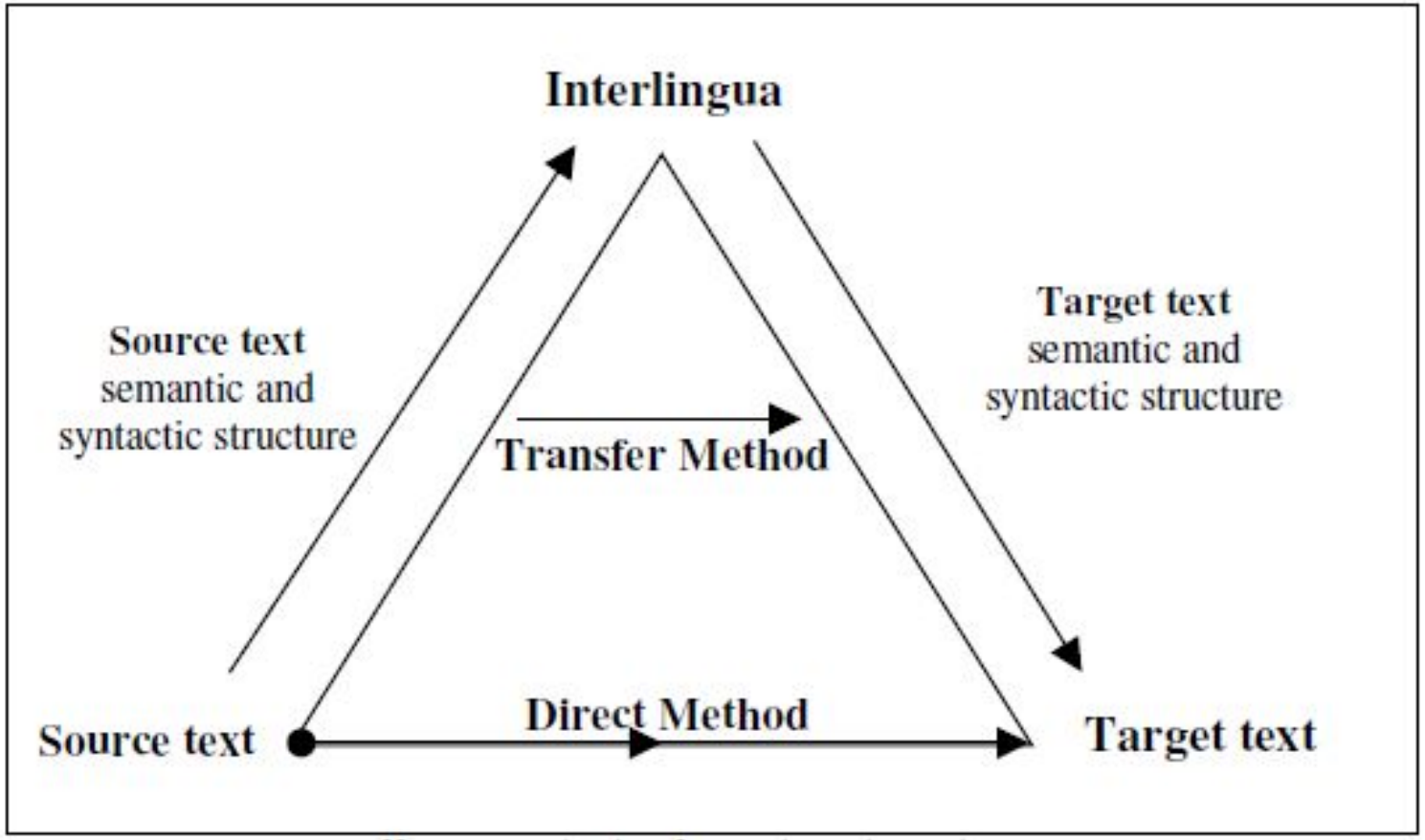
History of Machine Translation



https://arxiv.org/pdf/2002.07526.pdf

2 - Rule-based Machine Translation

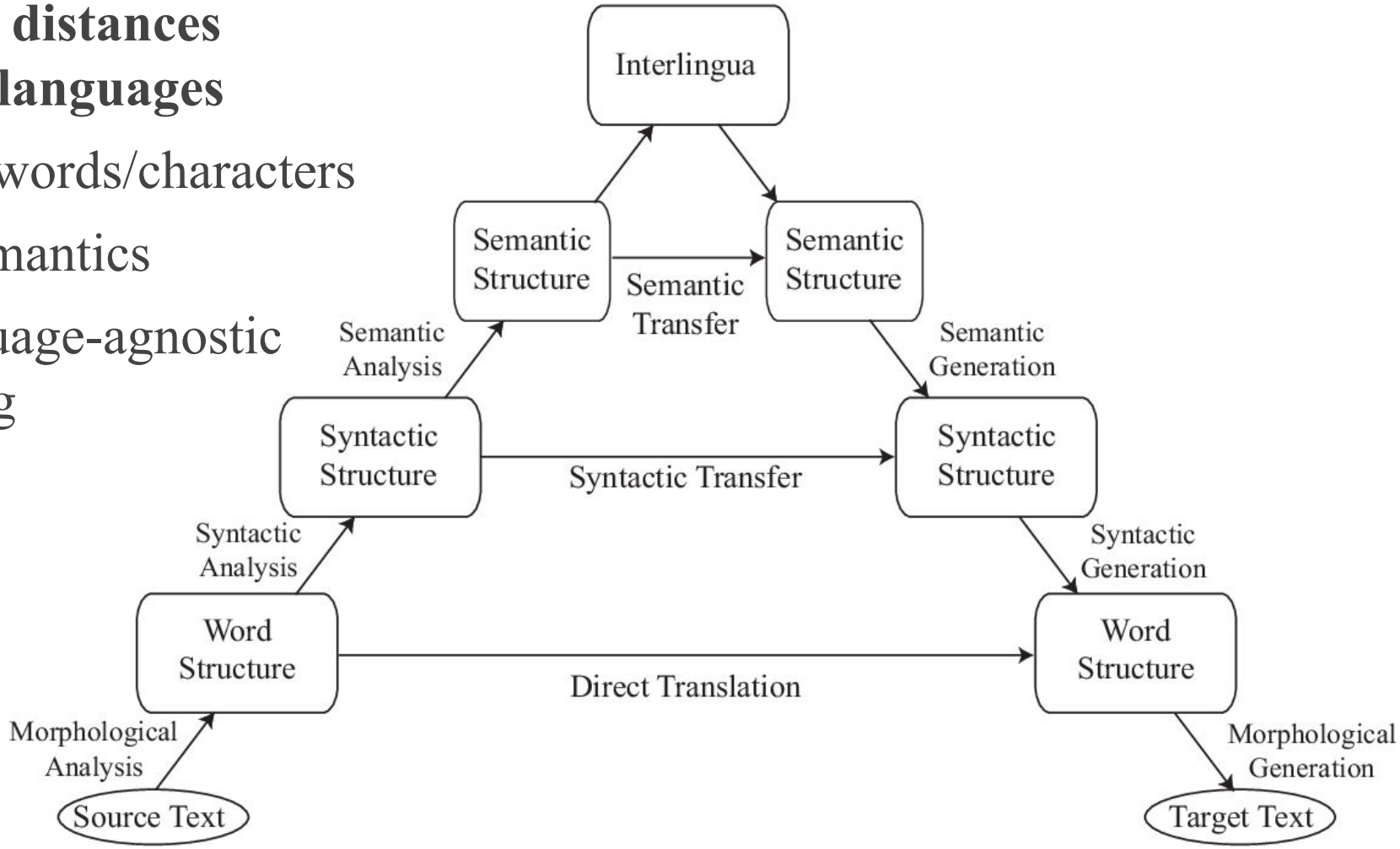
❖ A classical approach of machine translation (MT) systems based on **linguistic information** that includes **morphological**, **syntactic**, and **semantic** of both the ***source language*** (SL) and ***target language*** (TL).



2 - Rule-based Machine Translation

Hierarchy of concepts and distances between them in different languages

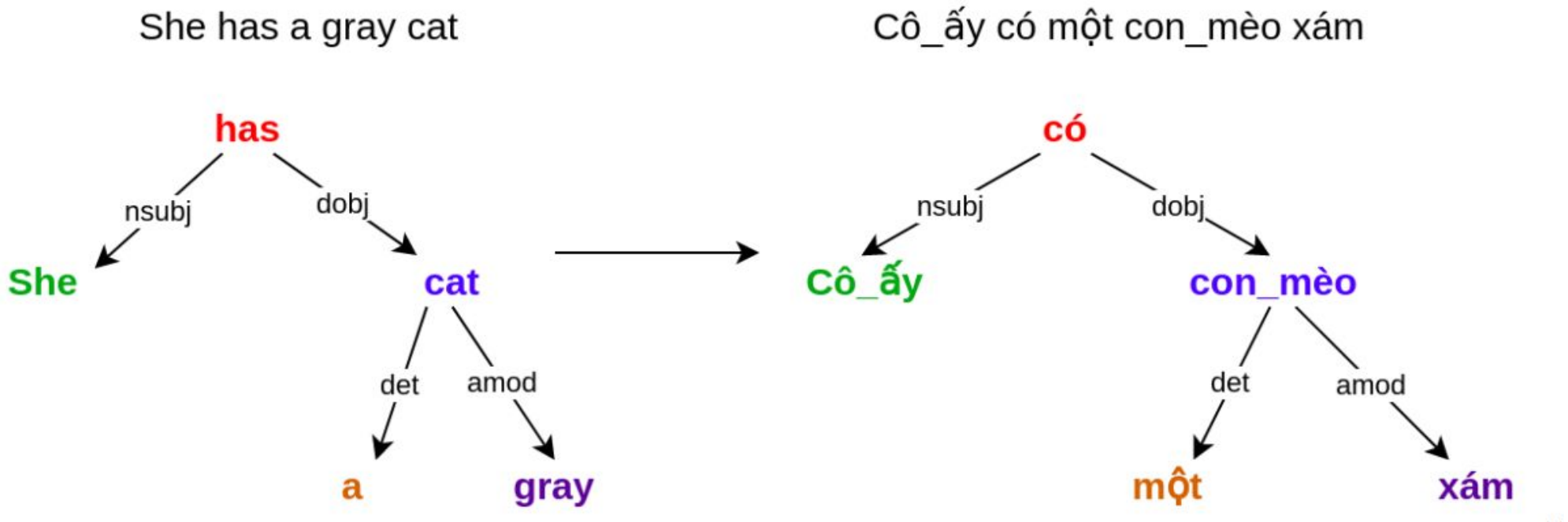
- Lowest level: individual words/characters
- Higher levels: syntax, semantics
- Interlingua: generic language-agnostic representation of meaning



2 - Rule-based Machine Translation

Syntactic Translation Problems

- Reordering of phrases



2 - Rule-based Machine Translation

Semantic Translation Problems

- Words are ambiguous, variations:

"Con ngựa đá con ngựa đá"

"Ba tôi đang đi làm"

"Năm nay tôi vào lớp ba"

- Single words may be replaced with multi-word phrases

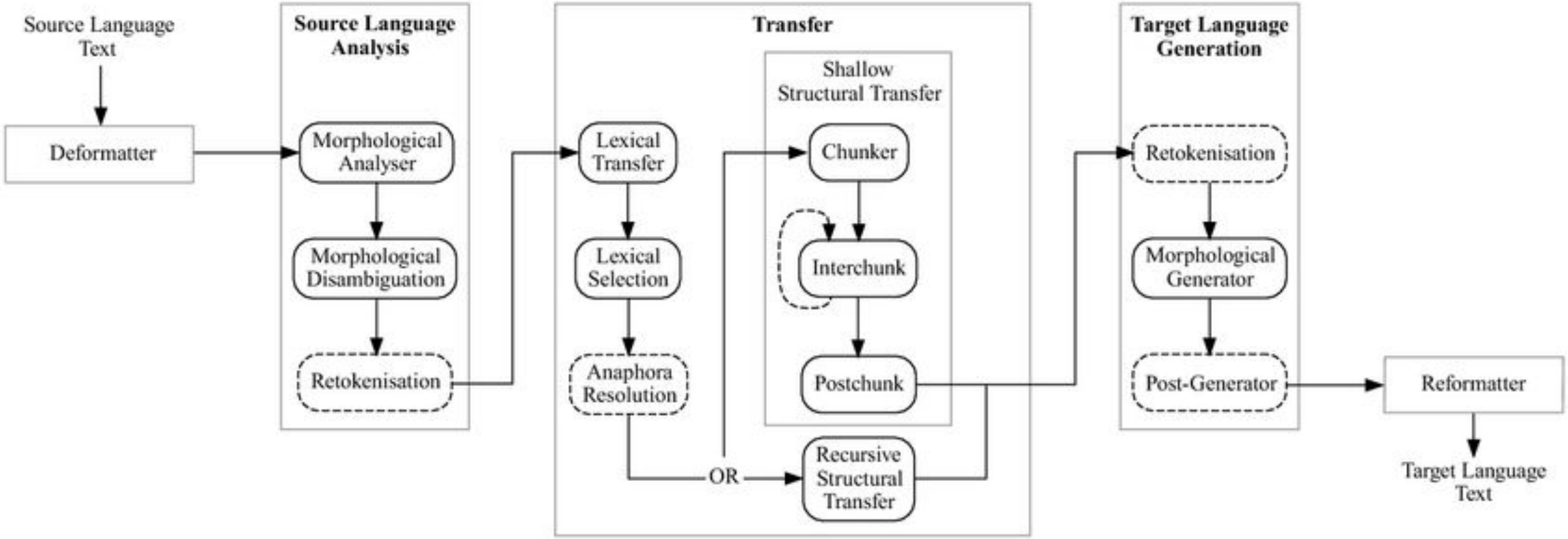
"rose" <=> "hoa hồng"

- What is the best translation?

Tôi đang học NLP	I am studying NLP
	I am learning NLP

2 - Rule-based Machine Translation

Transfer-based Machine Translation



https://www.researchgate.net/publication/355393125_Recent_advances_in_Apertium_a_freeopen-source_rule-based_machine_translation_platform_for_low-resource_languages

2 - Rule-based Machine Translation

- Disadvantages
  - Computational inefficiency in determining the adaptive rule of one sentence

    → **hardly** achieves satisfactory performance in implementation.
  - Grammar rules are also hard to be organized, too many syntax rules in one language (especially language with more relaxed grammar rules)
  - Easily ignore the need of context information in the translation process

Sentence 1: “The pen is in the box.”

Sentence 2: “The box is in the pen.”

3 – Statistical Machine Translation

❖ 3.1 Lexical Translation

How to translate a word → look up in a dictionary

nhà – house, home, domicile, family, dynasty

Multiple translation

- some more frequent than others, e.g: house, home
- Special cases: dynasty

In this presentation, we translate Vietnamese into English

3 – Statistical Machine Translation

❖ 3.2 Estimate Translation Probabilities

Translations of *nhà*	Count
house	8000
home	1600
domicile	200
family	150
dynasty	50

Collect statistics

$$p_f(e) = \begin{cases} 0.8 & \text{if } e = house \\ 0.16 & \text{if } e = home \\ 0.02 & \text{if } e = domicile \\ 0.015 & \text{if } e = family \\ 0.005 & \text{if } e = dynasty \end{cases}$$

Maximum likelihood estimation

$$p_f : e \rightarrow p_f(e)$$

nhà

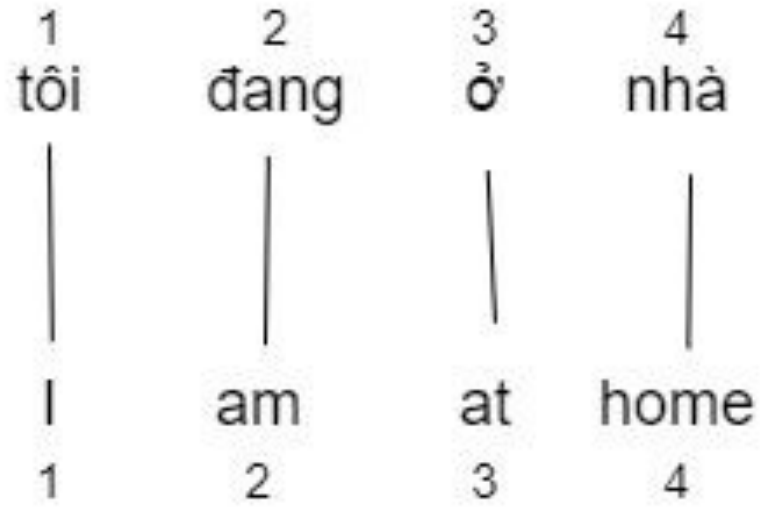
e	$t(e\|f)$
house	0.8
home	0.16
domicile	0.02
family	0.015
dynasty	0.005

T-table

$t(e|f)$

3 – Statistical Machine Translation

❖ 3.3 Alignment



In a **parallel text** (or when we translate), we **align words** in one language with the words in the other

- Word positions are numbered 1–4

3 – Statistical Machine Translation

❖ 3.3 Alignment

- Formalizing alignment with an **alignment function**
- Mapping an English **target word** at position *i* to a Vietnamese **source word** at position *j* with a function

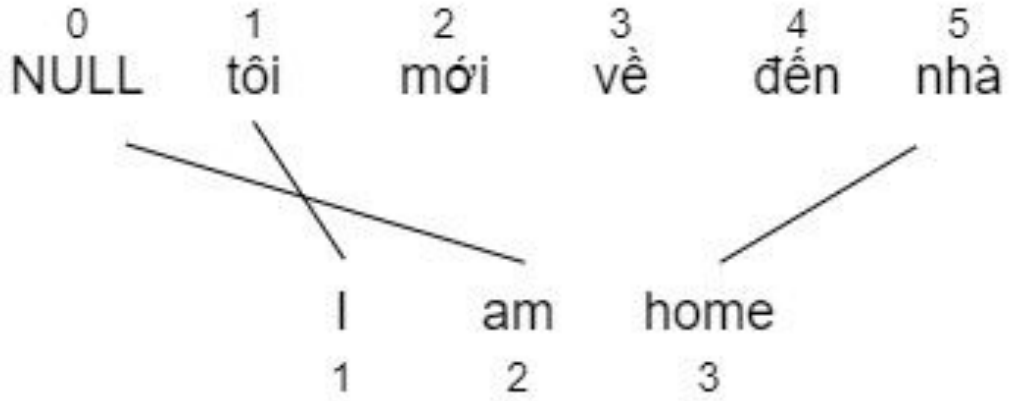
$$a : i \rightarrow j$$

- E.g.

$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4\}$$

3 – Statistical Machine Translation

❖ 3.4 Dropping and inserting words



- Words may be dropped or added when translated
- Map English word to special token **NULL**
- E.g.

$$a : \{1 \rightarrow 1,\ 2 \rightarrow 0,\ 3 \rightarrow 5\}$$

3 – Statistical Machine Translation

❖ 3.5 IBM Model 1

IBM Model 1 only uses **lexical translation**

Translation probability

- for a foreign sentence **f** = $(f_1, ..., f_{lf})$ of length l_f
- to an English sentence **e** = $(e_1, ..., e_{le})$ of length l_e
- with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{\varepsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \qquad (2.3)$$

3 – Statistical Machine Translation

❖ 3.6 Learning Lexical Translation Models

• We would like to estimate the lexical translation probabilities *t(e|f)* from a parallel corpus

• ... but we do not have the alignments

• Chicken and egg problem

– if we had the alignments,

→ we could estimate the parameters of our model

– if we had the parameters,

→ we could estimate the alignments

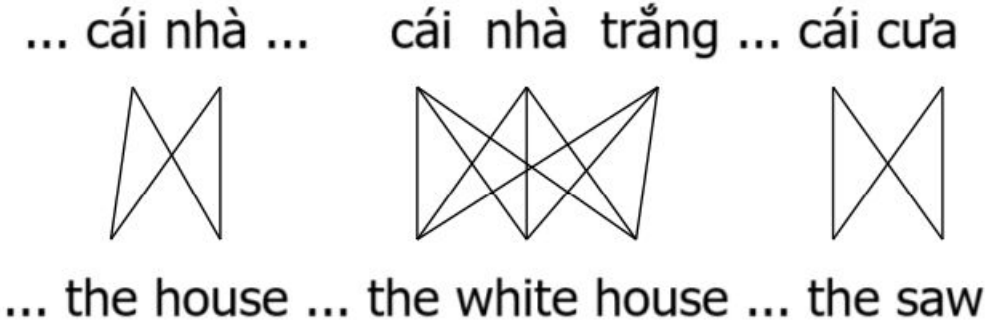
3 – Statistical Machine Translation

❖ 3.7 Expectation Maximization Algorithm

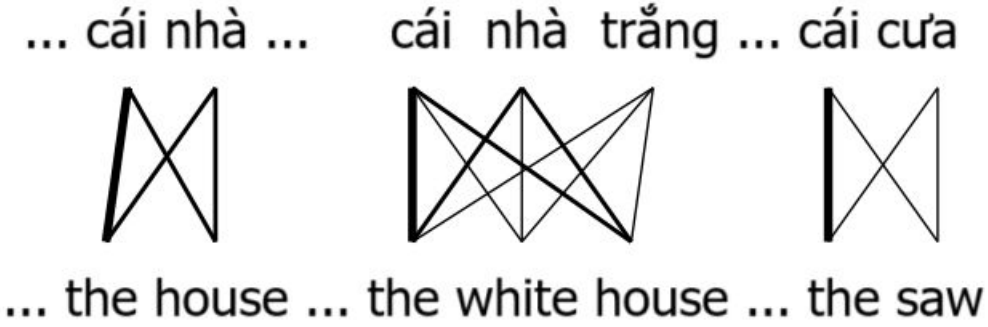
- Incomplete data
  – if we had complete data, would could estimate model
  – if we had model, we could fill in the gaps in the data
- Expectation Maximization (EM) in a nutshell
  1. initialize model parameters (e.g. uniform)
  2. assign probabilities to the missing data
  3. estimate model parameters from completed data
  4. iterate steps 2–3 until convergence

3 – Statistical Machine Translation

❖ 3.7 Expectation Maximization Algorithm



- Initial step: all alignments equally likely
- Model learns that, e.g., *cái* is often aligned with *the*



- After one iteration
- Alignments, e.g., between *cái* and *the* are more likely

3 – Statistical Machine Translation

❖ 3.7 Expectation Maximization Algorithm

- After another iteration
- It becomes apparent that alignments, e.g., between *cưa* and *saw* are more likely (pigeon hole principle)

- Convergence
- Inherent hidden structure revealed by EM

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM

- EM Algorithm consists of two steps
- **Expectation-Step**: Apply model to the data → compute **probability of alignments**
  – parts of the model are hidden (here: alignments)
  – using the model, assign probabilities to possible values
- **Maximization-Step**: Estimate model from data → compute **count collection**
  – take assign values as fact
  – collect counts (weighted by probabilities)
  – estimate model from counts
- **Iterate** these steps until **convergence**

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Expectation Step

- We need to compute $p(a|\mathbf{e},\mathbf{f})$
- Applying the chain rule:

$$p(a|\mathbf{e},\mathbf{f}) = \frac{p(\mathbf{e},a|\mathbf{f})}{p(\mathbf{e}|\mathbf{f})} \tag{2.4}$$

- We already have the formula for $p(\mathbf{e}, a|\mathbf{f})$

$$p(\mathbf{e},a|\mathbf{f}) = \frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \tag{2.3}$$

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Expectation Step

- We need to compute $p(\mathbf{e}|\mathbf{f})$

$$
\begin{aligned}
p(\mathbf{e}|\mathbf{f}) &= \sum_{a} p(\mathbf{e}, a|\mathbf{f}) \\
&= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e}, a|\mathbf{f}) \\
&= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\
&= \frac{\varepsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\
&= \frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i) \qquad (2.5)
\end{aligned}
$$

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Expectation Step

- Combine what we have:

$$
\begin{aligned}
p(a|\mathbf{e},\mathbf{f}) &= \frac{p(\mathbf{e},a|\mathbf{f})}{p(\mathbf{e}|\mathbf{f})} \\
&= \frac{\frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\varepsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)} \\
&= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)} \qquad (2.6)
\end{aligned}
$$

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Maximization Step

- Now we have to collect counts
- Evidence from a sentence pair **e**, **f** that word *e* is a translation of word *f*:

$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_a p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)}) \qquad (2.7)$$

- With the same simplication as before:

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{j=1}^{l_f} \delta(f,f_i) \qquad (2.8)$$

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Maximization Step

After collecting these counts over a corpus, we can estimate the model:

$$t(e|f;\mathbf{e},\mathbf{f}) = \frac{\sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f})}{\sum_{e}\sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f})} \qquad (2.9)$$

3 – Statistical Machine Translation

❖ 3.8 IBM Model 1 and EM: Pseudo-code

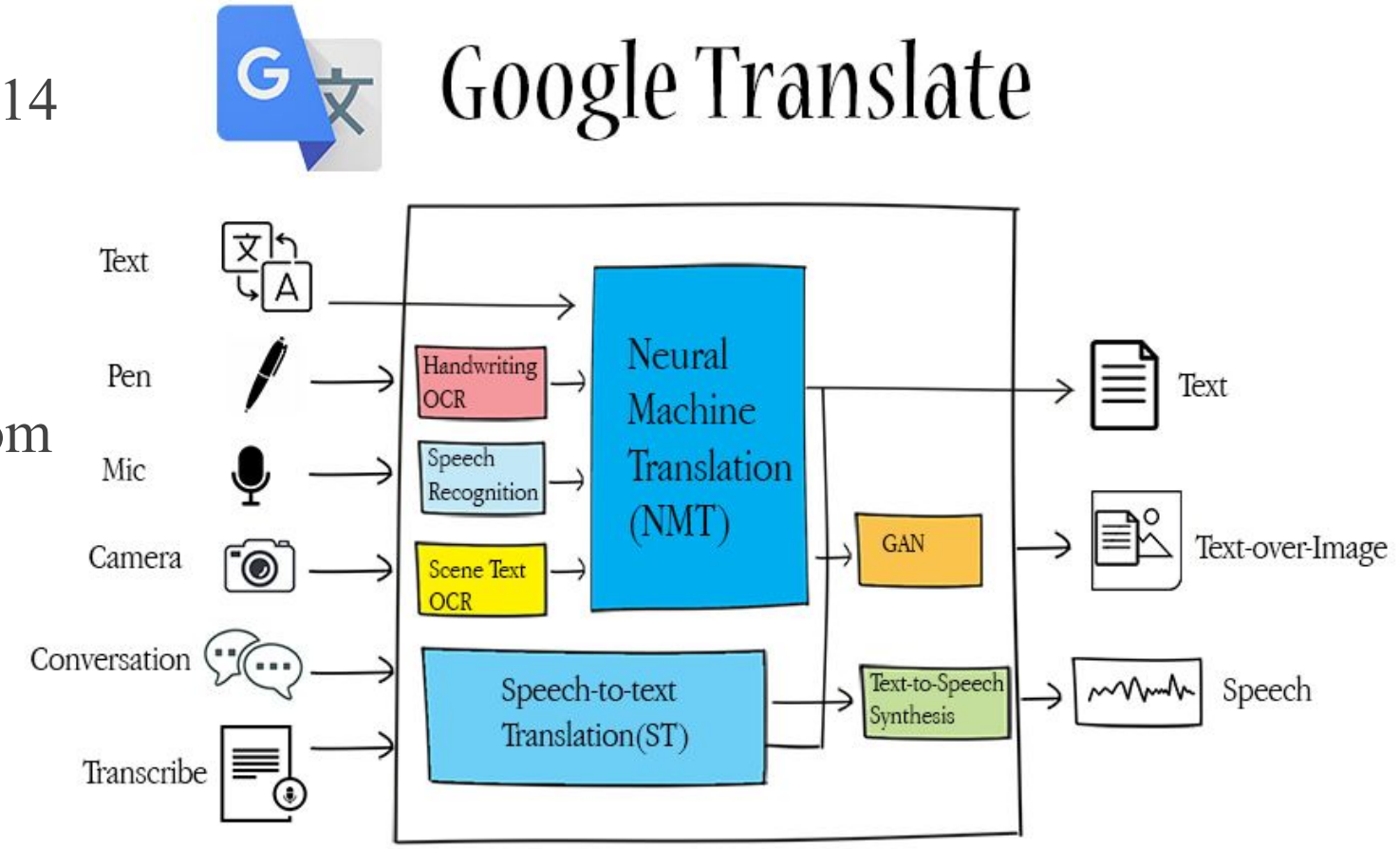
Input: set of sentence pairs (**e**, **f**)
Output: translation prob. $t(e|f)$

initialize $t(e|f)$ uniformly
while not converged **do**
  // initialize
  count$(e|f) = 0$ **for all** e, f
  total$(f) = 0$ **for all** f
  for all sentence pairs (**e**,**f**) **do**
    // compute normalization
    for all words e in **e** **do**
      s-total$(e) = 0$
      for all words f in **f** **do**
        s-total(e) += $t(e|f)$
      end for
    end for
    // collect counts
    for all words e in **e** **do**
      for all words f in **f** **do**
        count$(e|f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
        total(f) += $\frac{t(e|f)}{\text{s-total}(e)}$
      end for
    end for
  end for
  // estimate probabilities
  for all foreign words f **do**
    for all English words e **do**
      $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
    end for
  end for
end while

4 - Neural Machine Translation

❖ 4.1. NMT - The biggest success story of NLP Deep Learning

- Neural Machine Translation went from a fringe research activity in 2014 to the leading standard method in 2016.
- 2014: First seq2seq paper published
- 2016: Google Translate switches from SMT to NMT



4 - Neural Machine Translation

❖ 4.2. Text Preprocessing

To prepare the text data for the model building we perform text preprocessing. It is the very first step of NLP projects. Some of the preprocessing steps are:

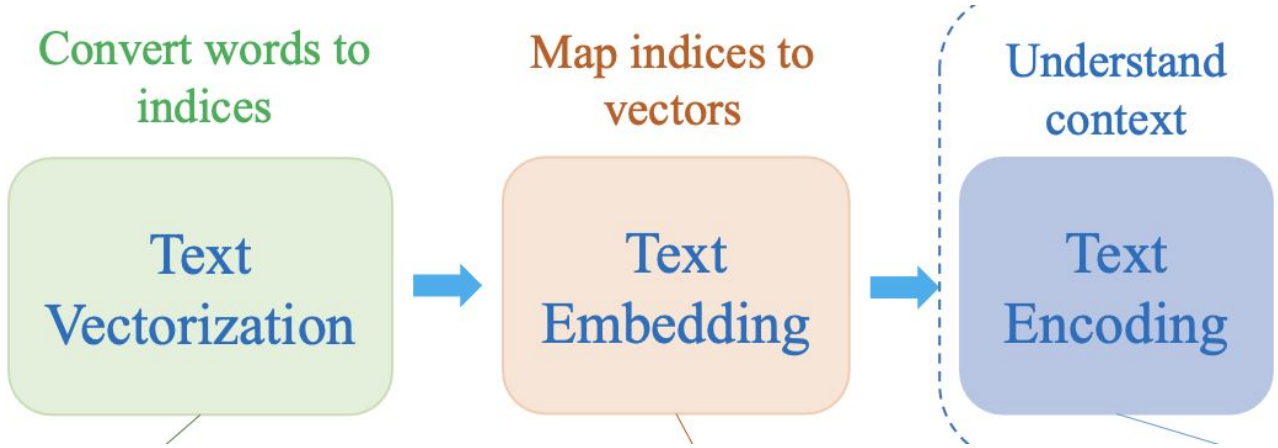
- ❖ Removing punctuations like . , ! $( ) * % @
- ❖ Removing URLs
- ❖ Removing Stop words
- ❖ Lower casing
- ❖ Tokenization
- ❖ Stemming
- ❖ Lemmatization

4 - Neural Machine Translation

❖ 4.2. Text Preprocessing

Text Vectorization

Sample 1: “We are learning AI”
Sample 2: “AI is a CS topic”



Step 1: Build vocabulary from corpus

index	0	1	2	3	4	5	6	7
word	pad	[UNK]	ai	we	topic	learning	is	cs

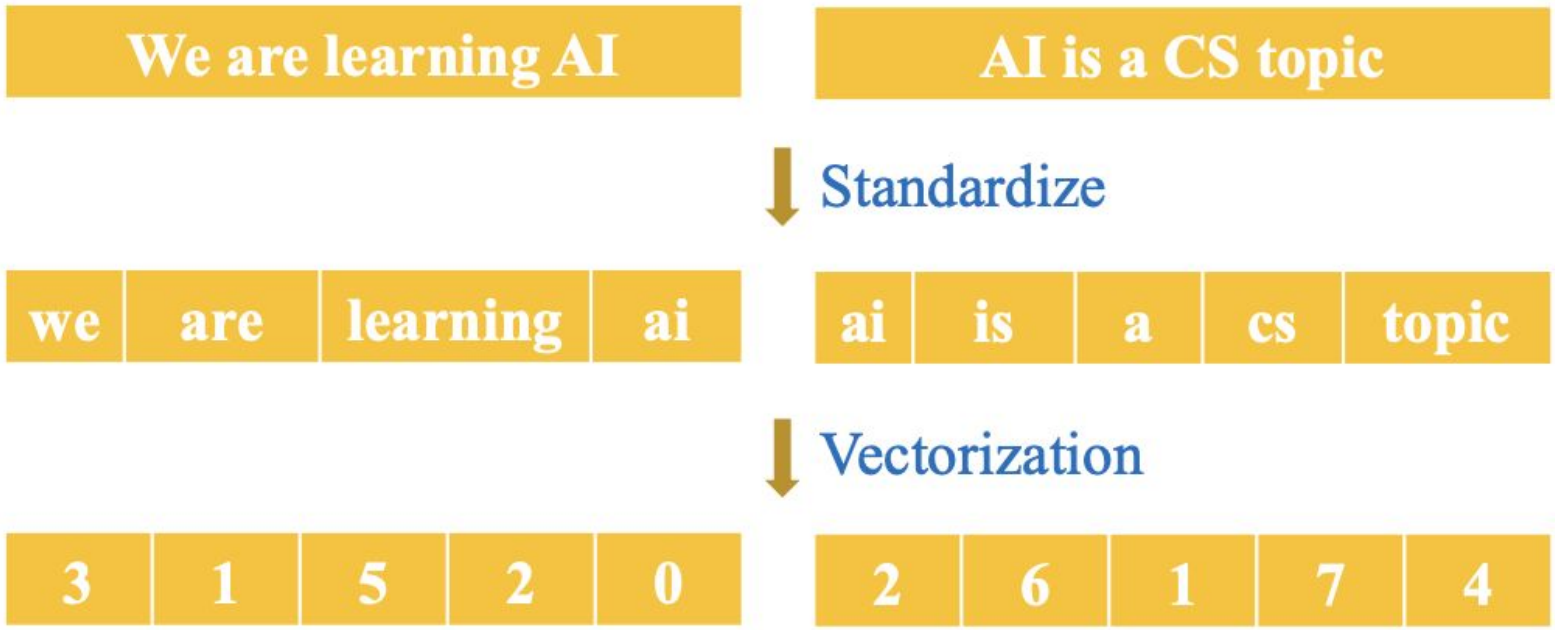
4 - Neural Machine Translation

❖ 4.2. Text Preprocessing

Text Vectorization

index	word
0	pad
1	[UNK]
2	AI
3	We
4	Topic
5	Learning
6	is
7	cs

Sample 1: “We are learning AI”
Sample 2: “AI is a CS topic”

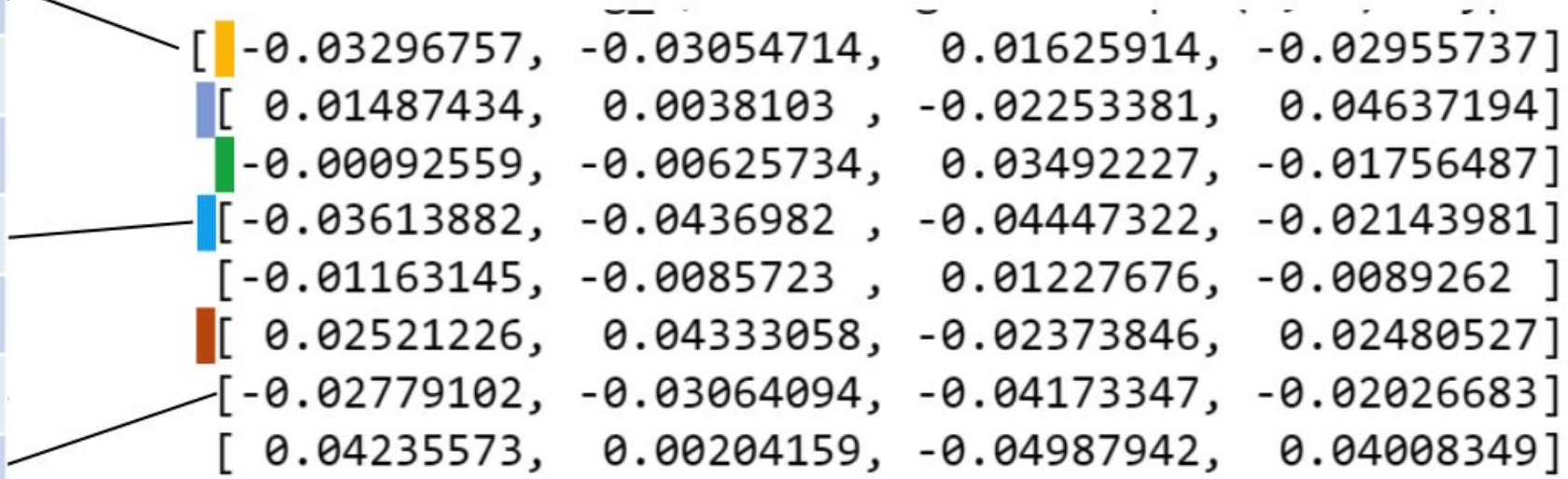


4 - Neural Machine Translation

❖ 4.2. Text Preprocessing

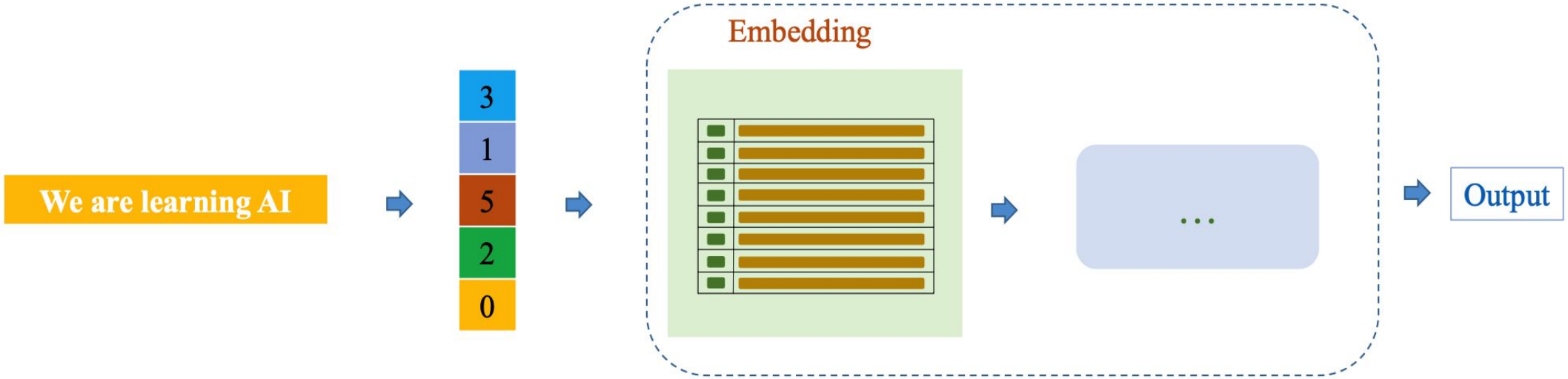
Text Embedding

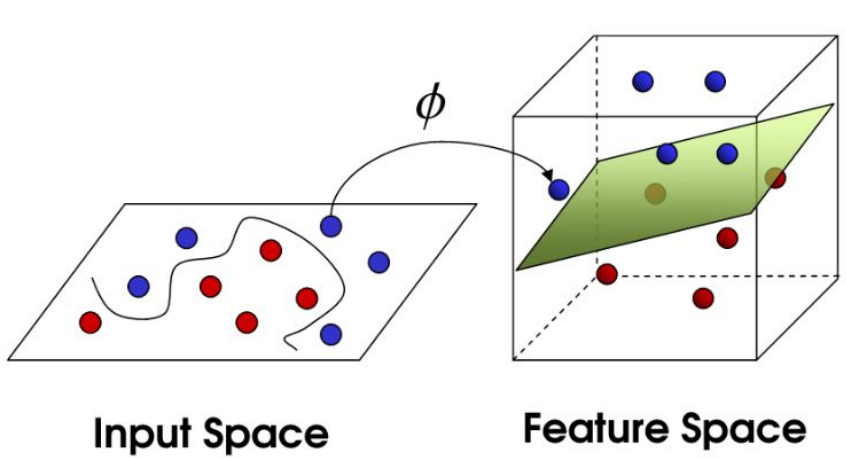
index	word
0	pad
1	[UNK]
2	AI
3	We
4	Topic
5	Learning
6	is
7	cs

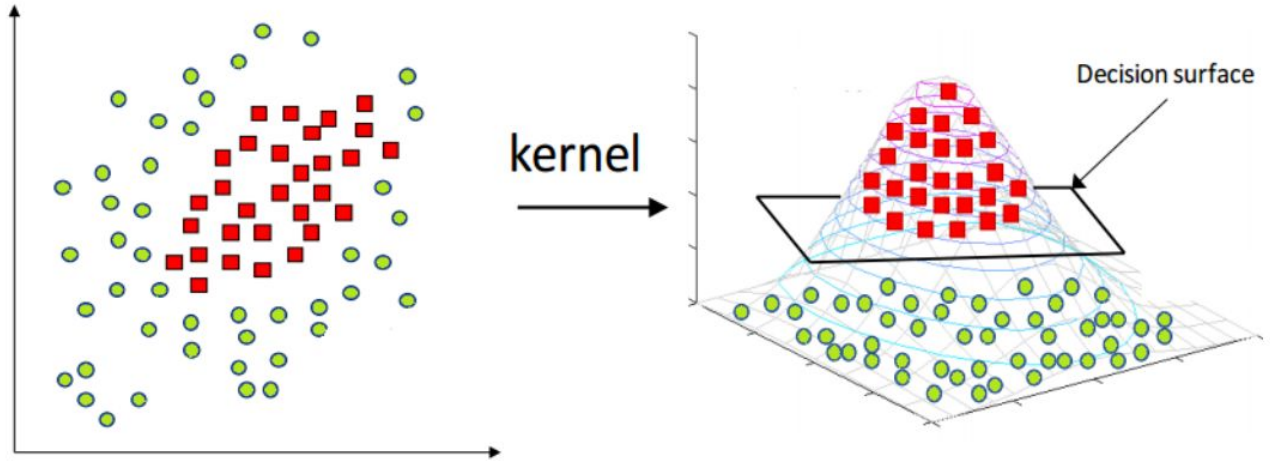


4 - Neural Machine Translation

❖ 4.2. Text Preprocessing

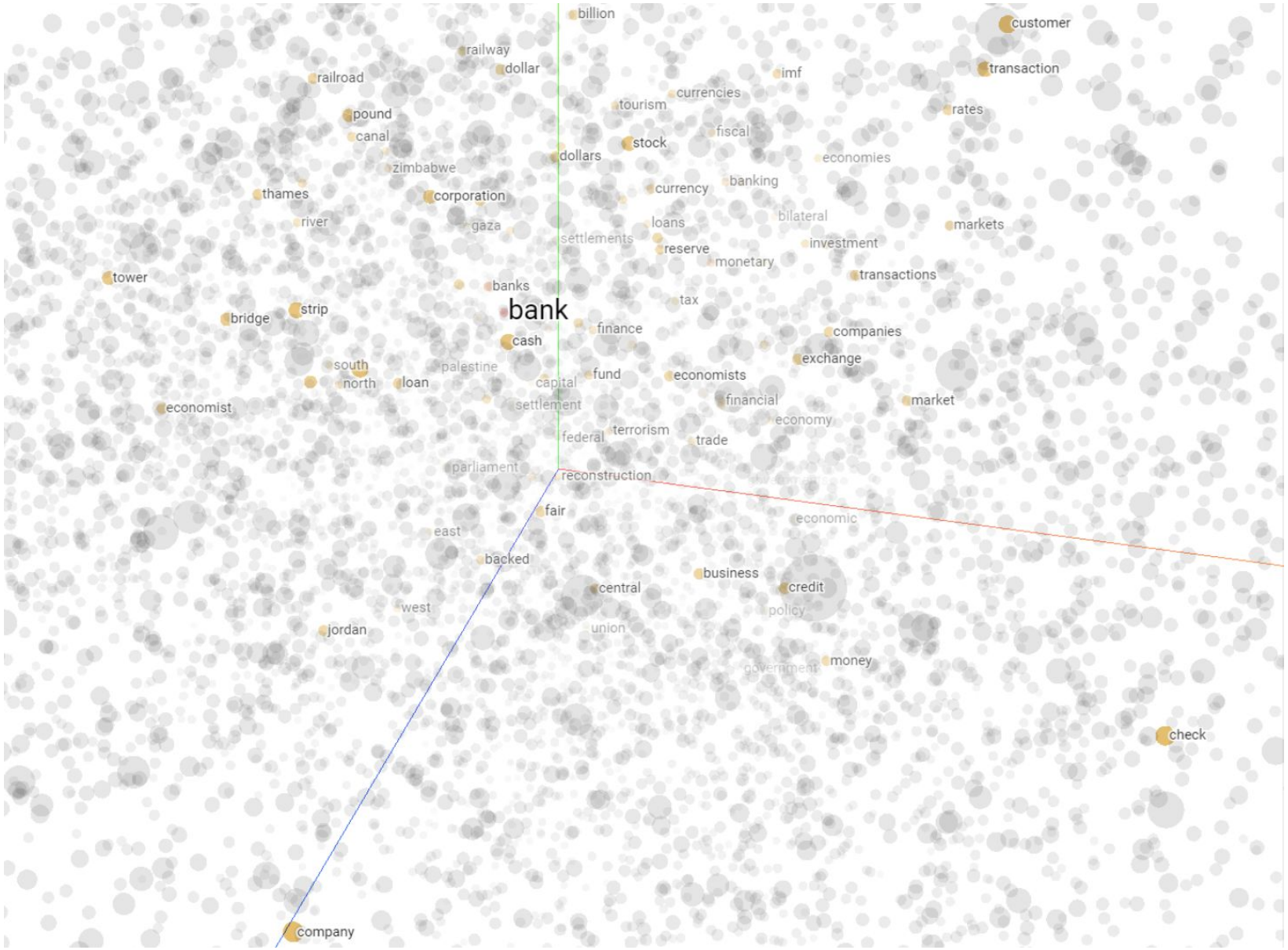






4 - Neural Machine Translation

Embedding visualization



4 - Neural Machine Translation

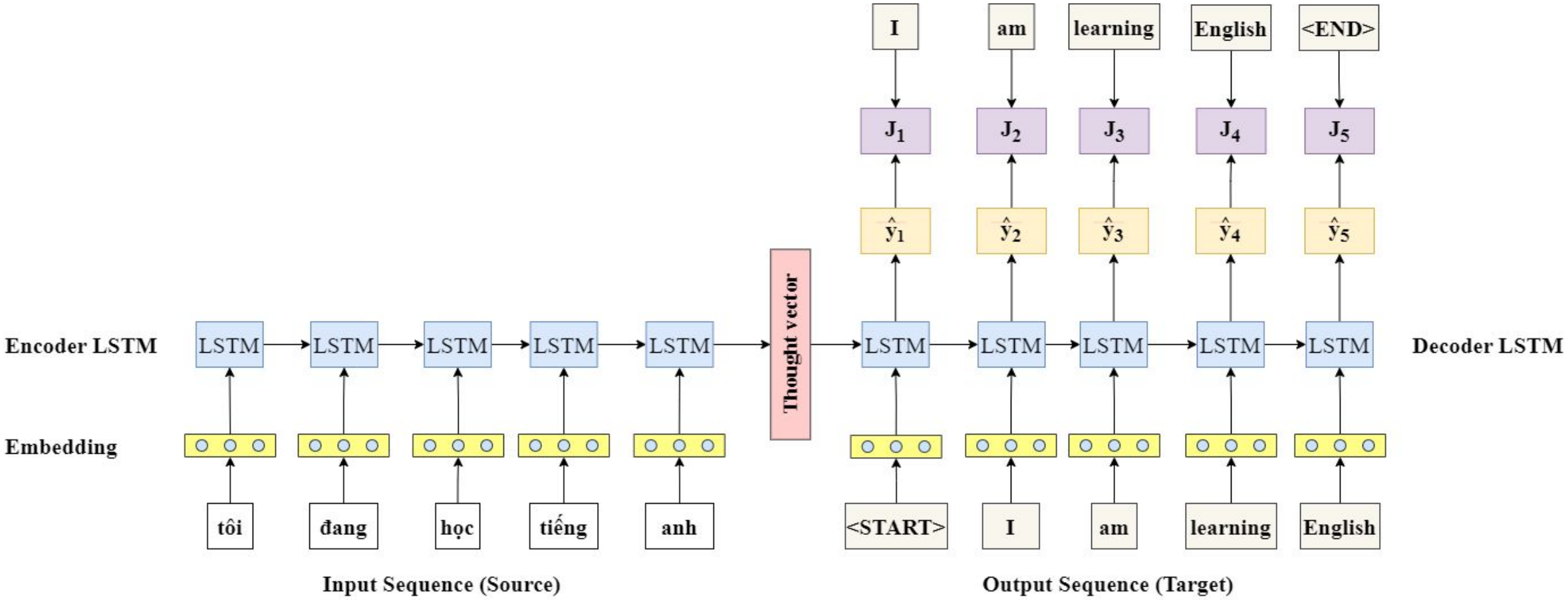
❖ 4.3. Sequence-to-Sequence Model

- A single neural network is used to translate from source to target
- Architecture: Encoder-Decoder
- Encoder: Convert source sentence (input) into a vector/matrix (State)
- Decoder: Convert encoding into a sentence in target language (output)

Thought Vector
Capture all information of input sentence

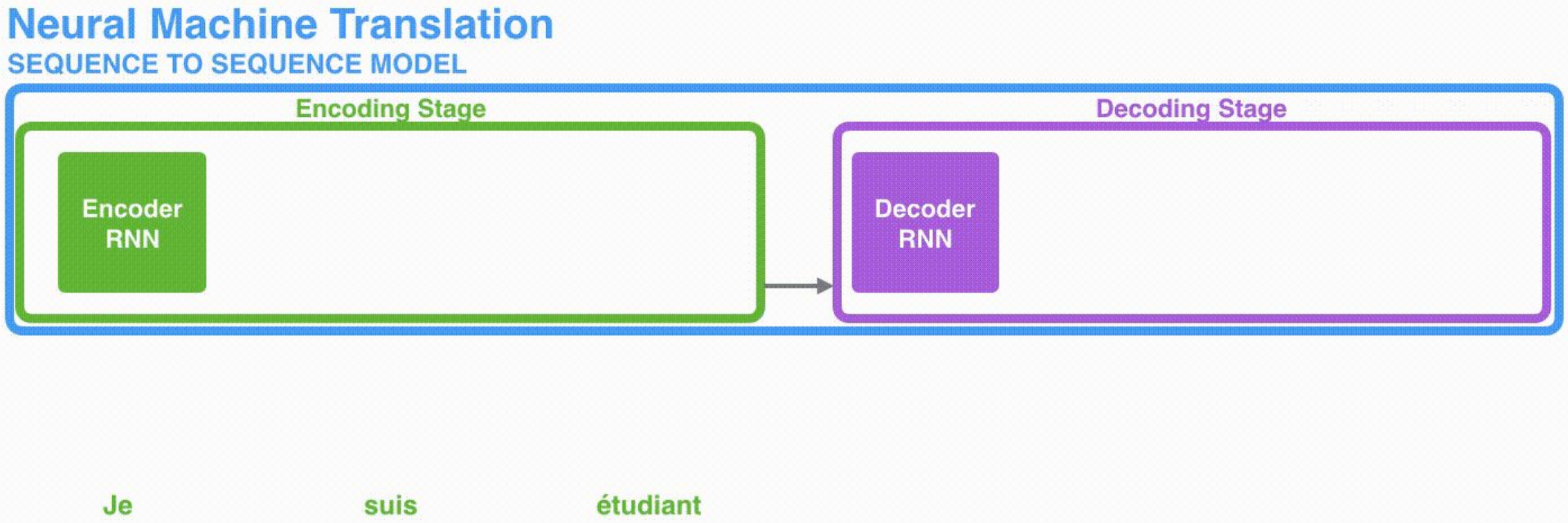
4 - Neural Machine Translation

❖ 4.3. Sequence-to-Sequence Model



4 - Neural Machine Translation

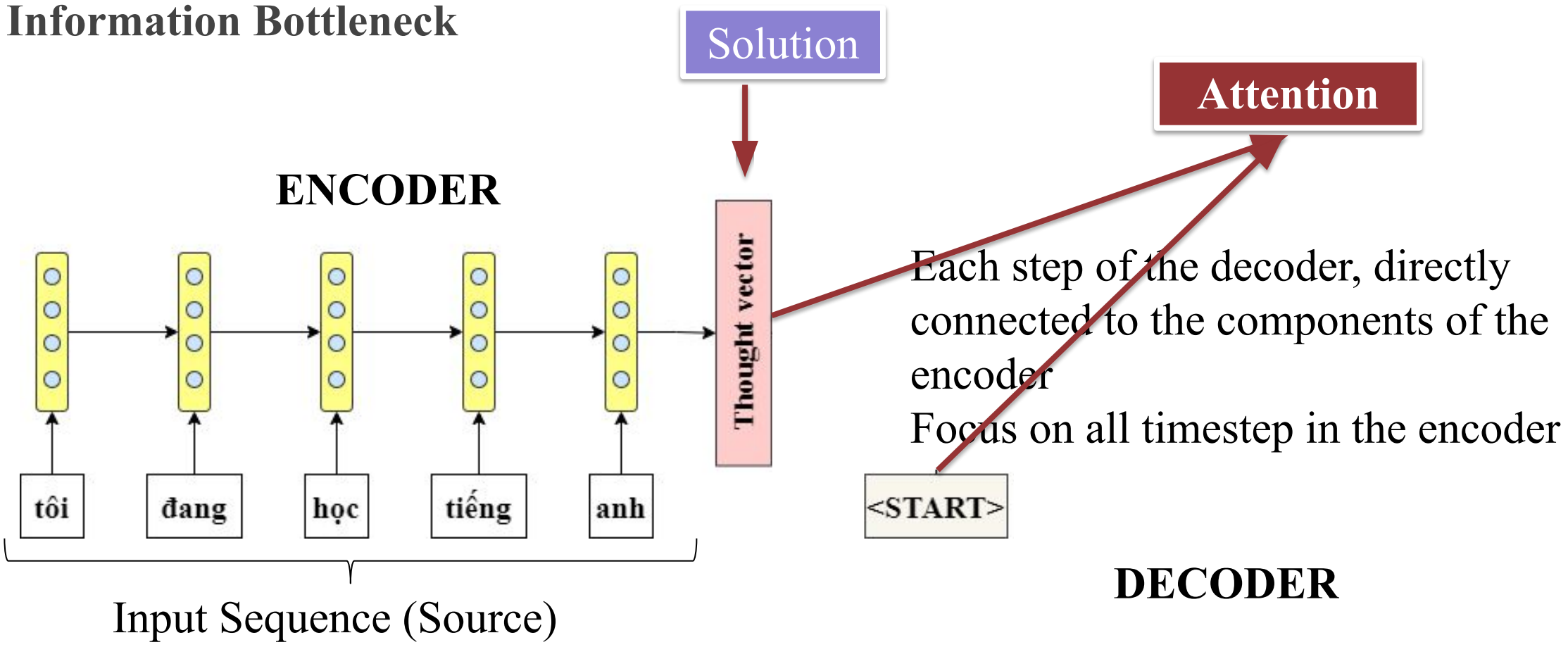
❖ 4.3. Sequence-to-Sequence Model



Visualization + Explanation: https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

4 - Neural Machine Translation

❖ 4.3. Sequence-to-Sequence Model



4 - Neural Machine Translation

❖ 4.4. Transformer

- Architecture:

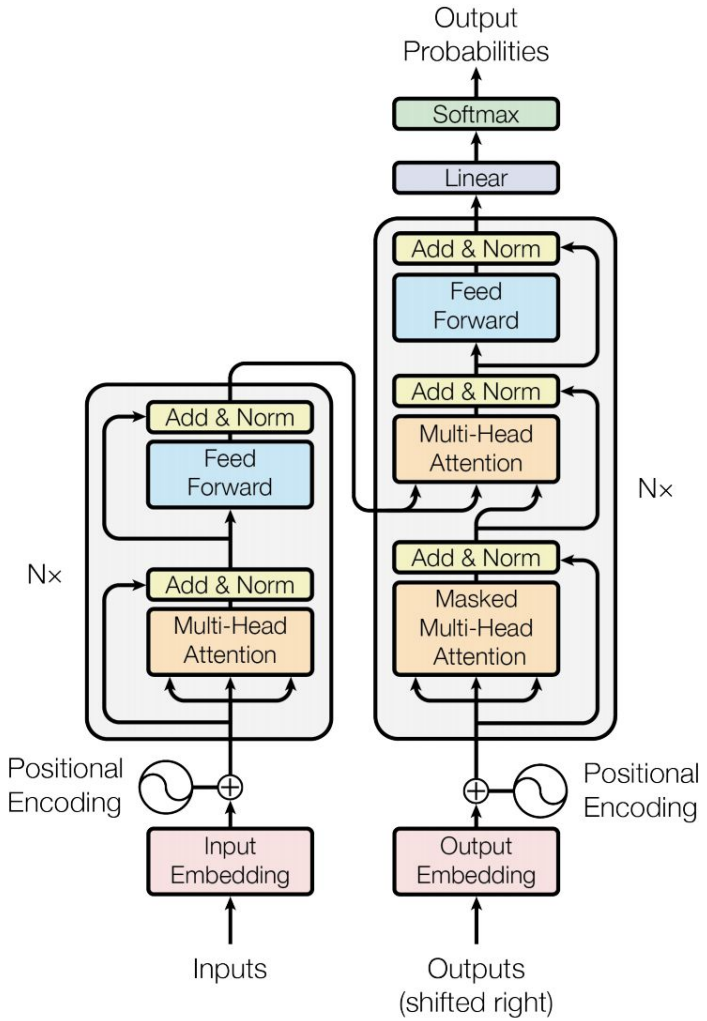
  Token Embedding

  Positional Embedding

  N x Encoder Layer:

  N x Decoder Layer

- Core technique: Self-Attention



4 - Neural Machine Translation

❖ 4.4. Transformer

Input Embedding

Positional Encoding: Sinusoid

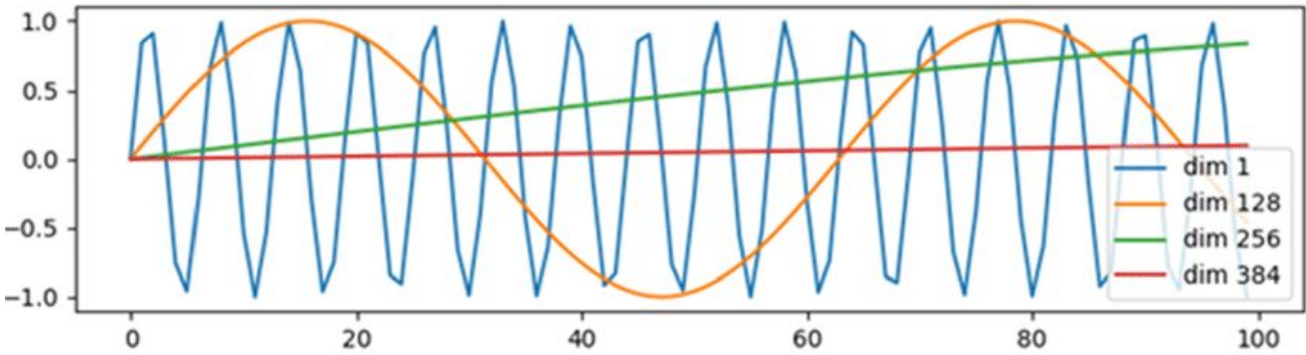
tôi
đang
học
tiếng
anh

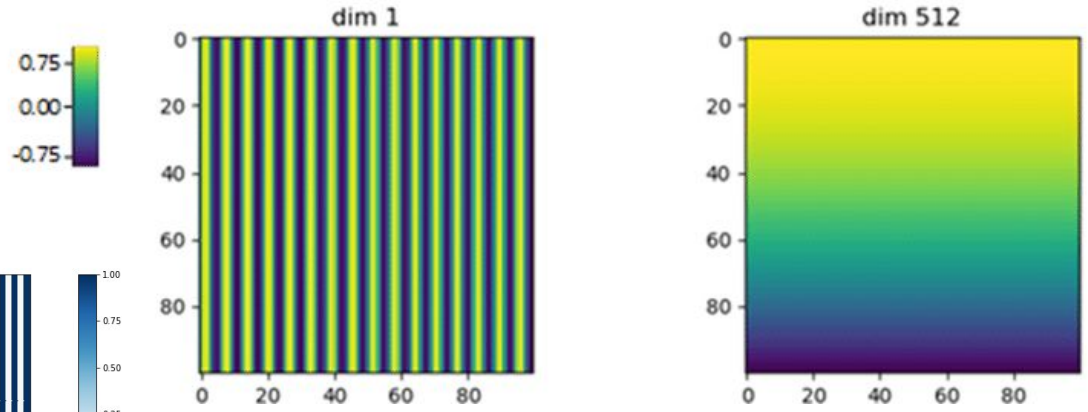
0.1	0.2	0.3	0.4	0.5
0.3	0.2	0.1	0.8	0.6
0.1	0.4	0.3	0.1	0.5
0.3	0.2	0.3	0.3	0.1
0.1	0.3	0.5	0.4	0.5

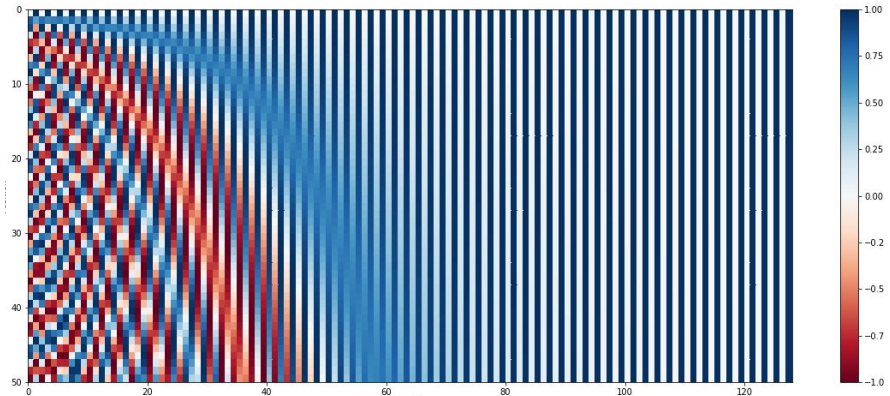
0	1	0	1	0
0.84	0.54	0.16	0.99	0.025
0.91	-0.416	0.311	0.95	0.05
0.141	-0.99	0.46	0.89	0.07
-0.76	-0.65	0.59	0.81	0.1

Final Embedding

0.1	1.2	0.3	1.4	0.5
1.14	0.74	0.26	1.79	0.625
1.01	-0.016	0.611	1.05	0.55
0.44	-0.79	0.76	1.12	0.176
-0.65	-0.35	1.09	1.21	0.6

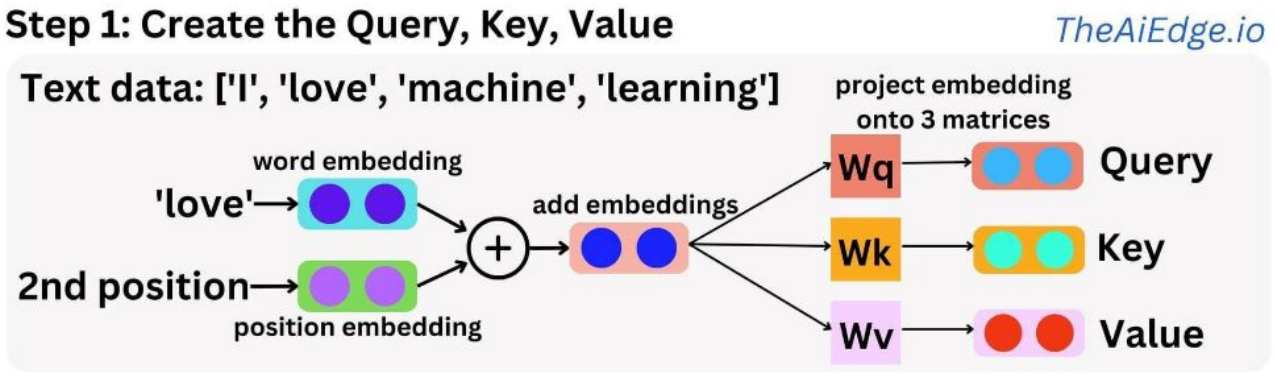


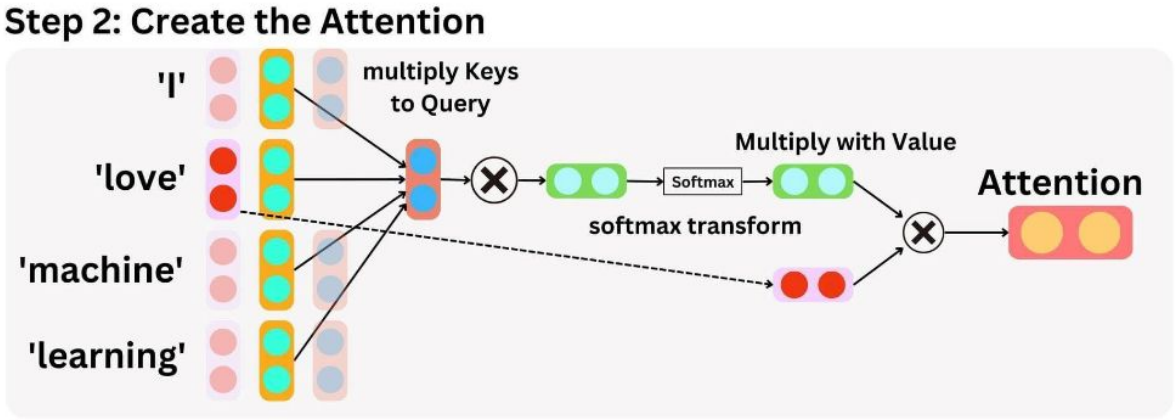


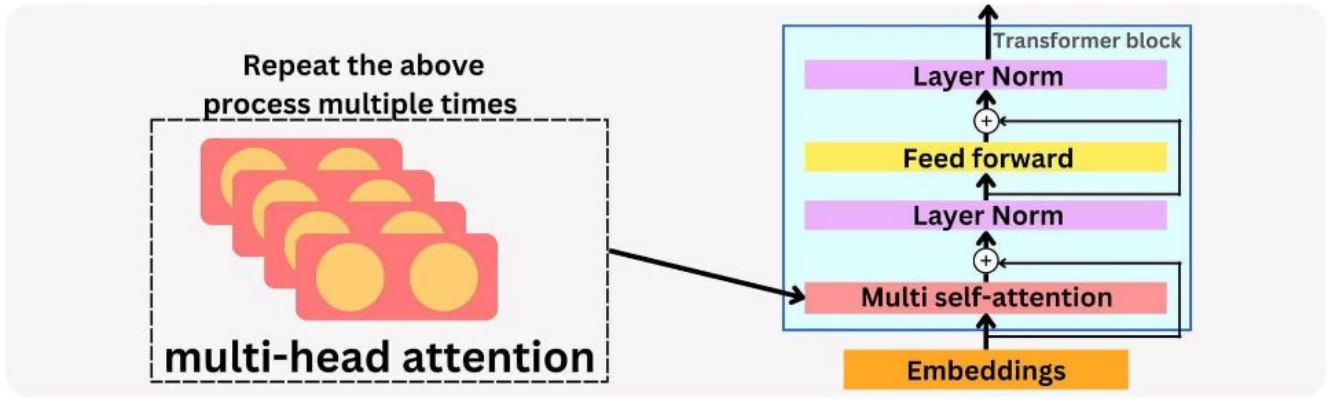


4 - Neural Machine Translation

❖ 4.4. Transformer

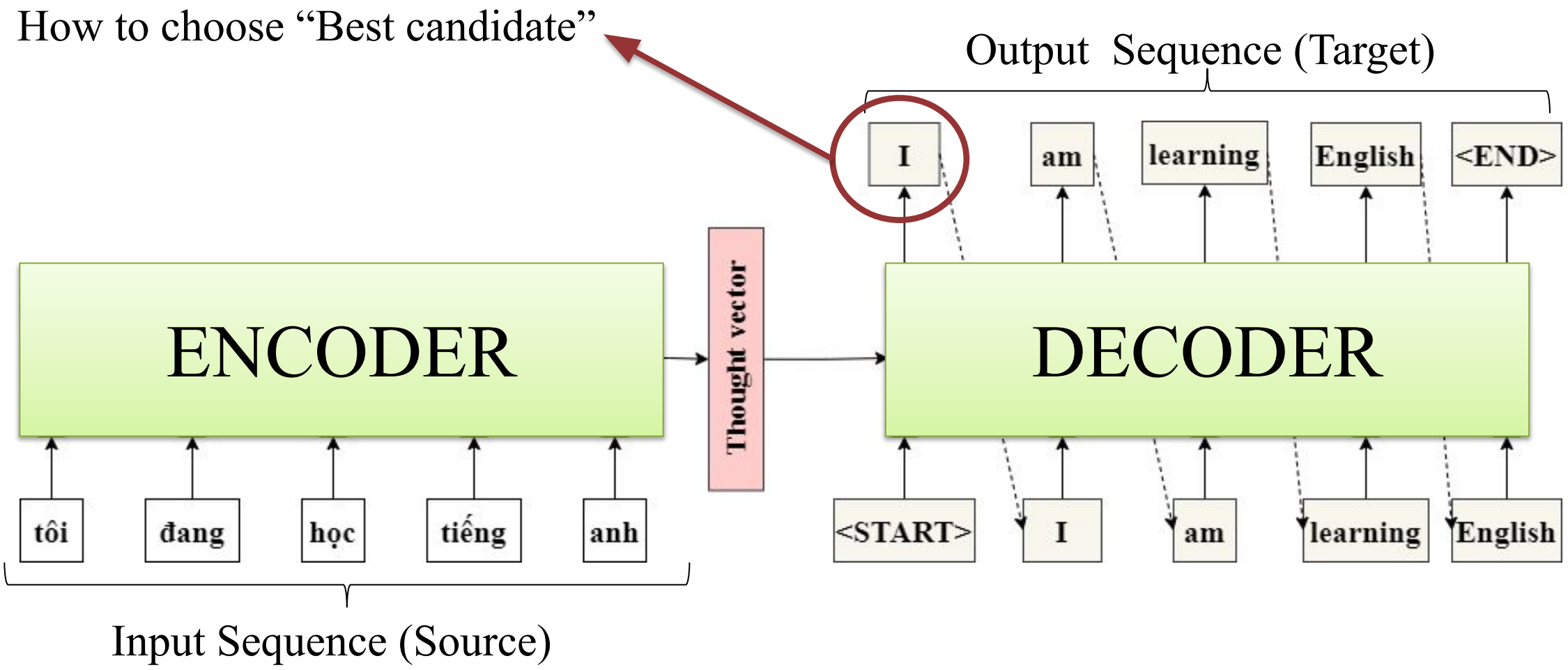






4 - Neural Machine Translation

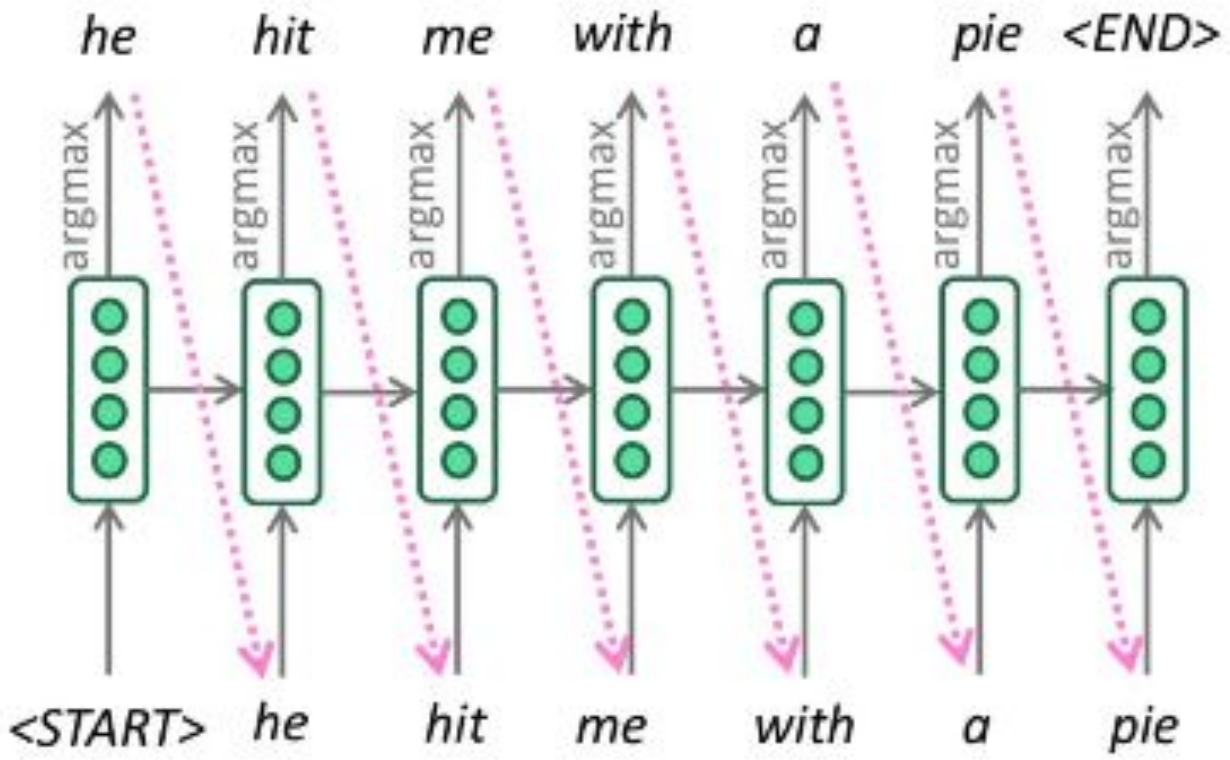
❖ 4.5. Decoding in NMT



4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Greedy Search: Compute argmax at every step of decoder to generate token



4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Beam Search

- A middle ground
- At every step, keep track of the k most probable partial translations (hypotheses)
- Score of each hypothesis = log probability:

$$\sum_{t=1}^{j} \log P(y_t | t_1, \dots y_{t-1}, x_1, \dots, x_n)$$

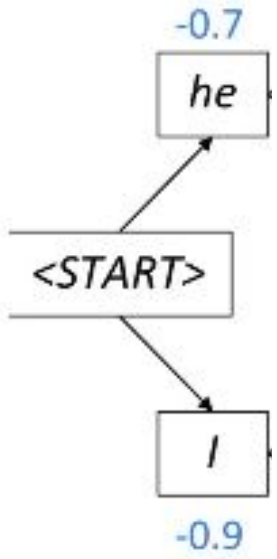
- Not guaranteed to be optimal
- More efficient than exhaustive search

4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Beam Search

- Beam size = 2

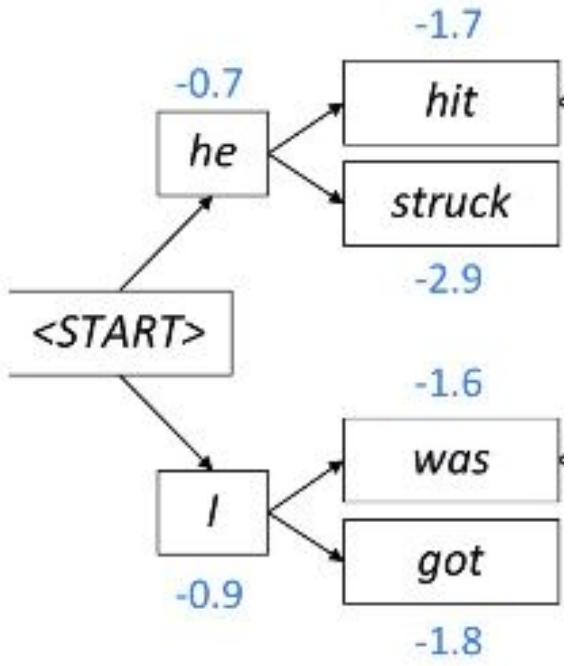


4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Beam Search

- Beam size = 2

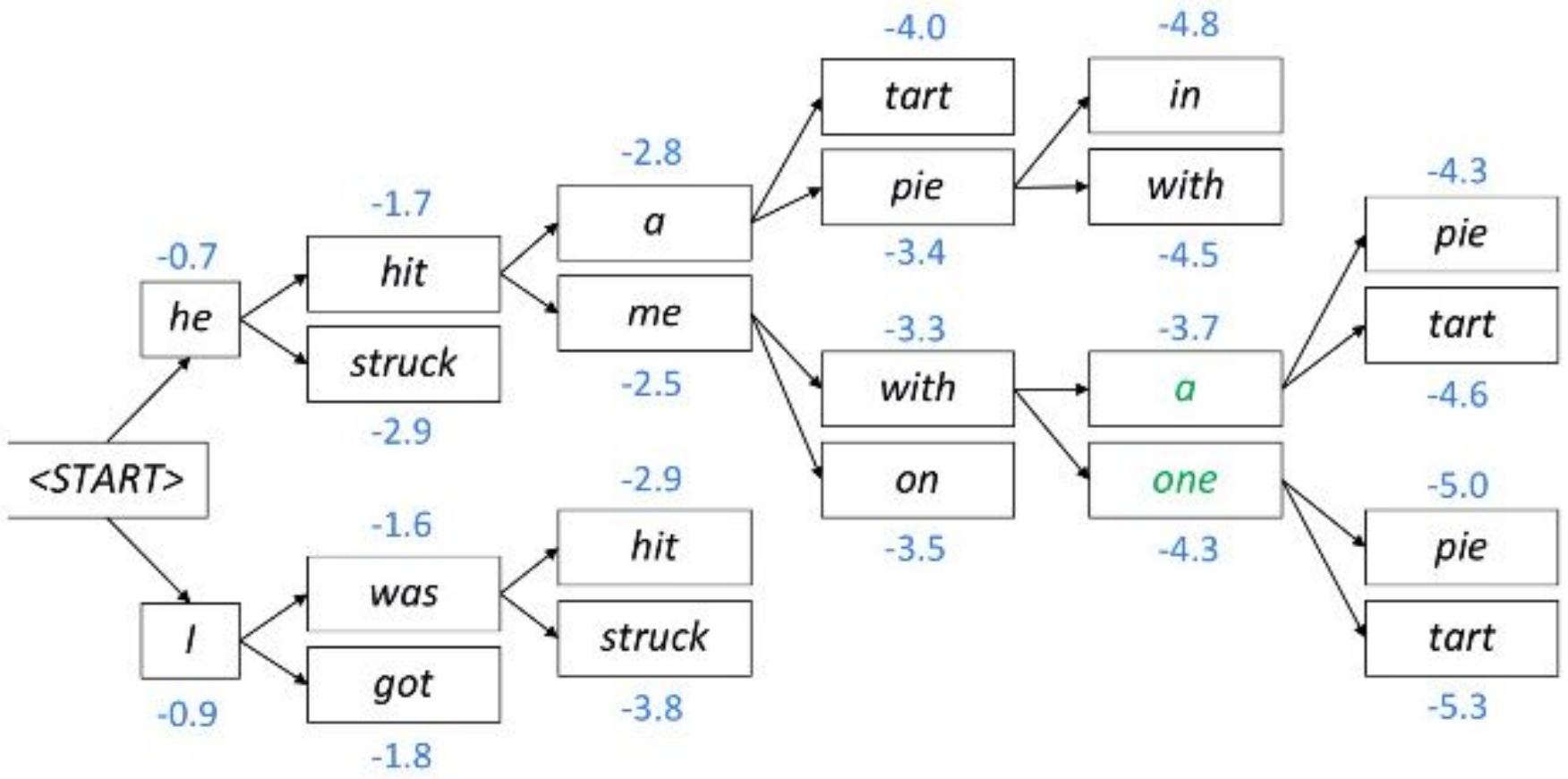


4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Beam Search

- Bea



4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

Beam Search

- Different hypotheses may produce <end> token at different timesteps
- Continue beam search until:

  All k hypotheses produce <end> token

  Hit max decoding limit T
- Select top hypotheses using the normalized likelihood score

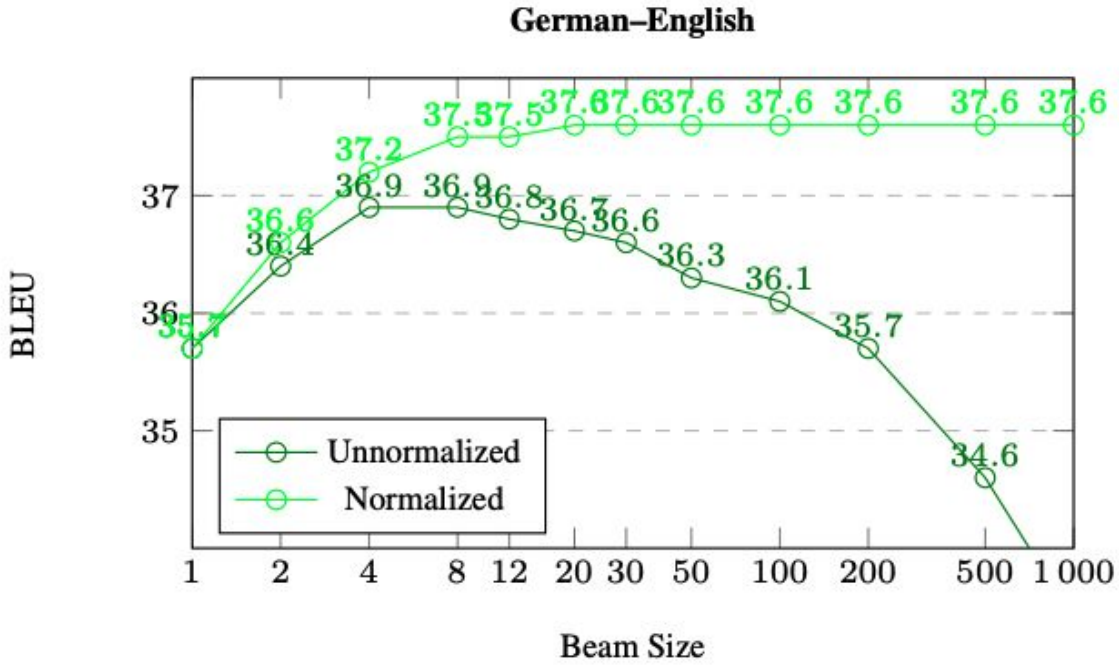
$$\frac{1}{T}\sum_{t=1}^{j}\log P(y_t|t_1,\dots y_{t-1},x_1,\dots,x_n)$$

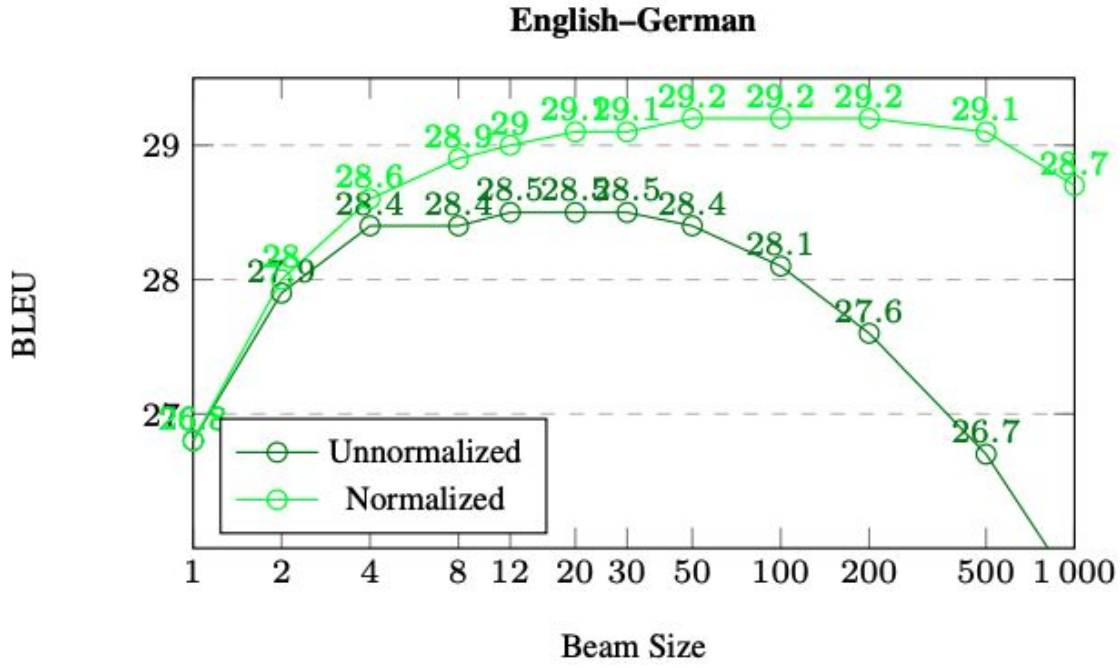
Otherwise shorter hypothesis have higher scores

4 - Neural Machine Translation

❖ 4.5. Decoding in NMT

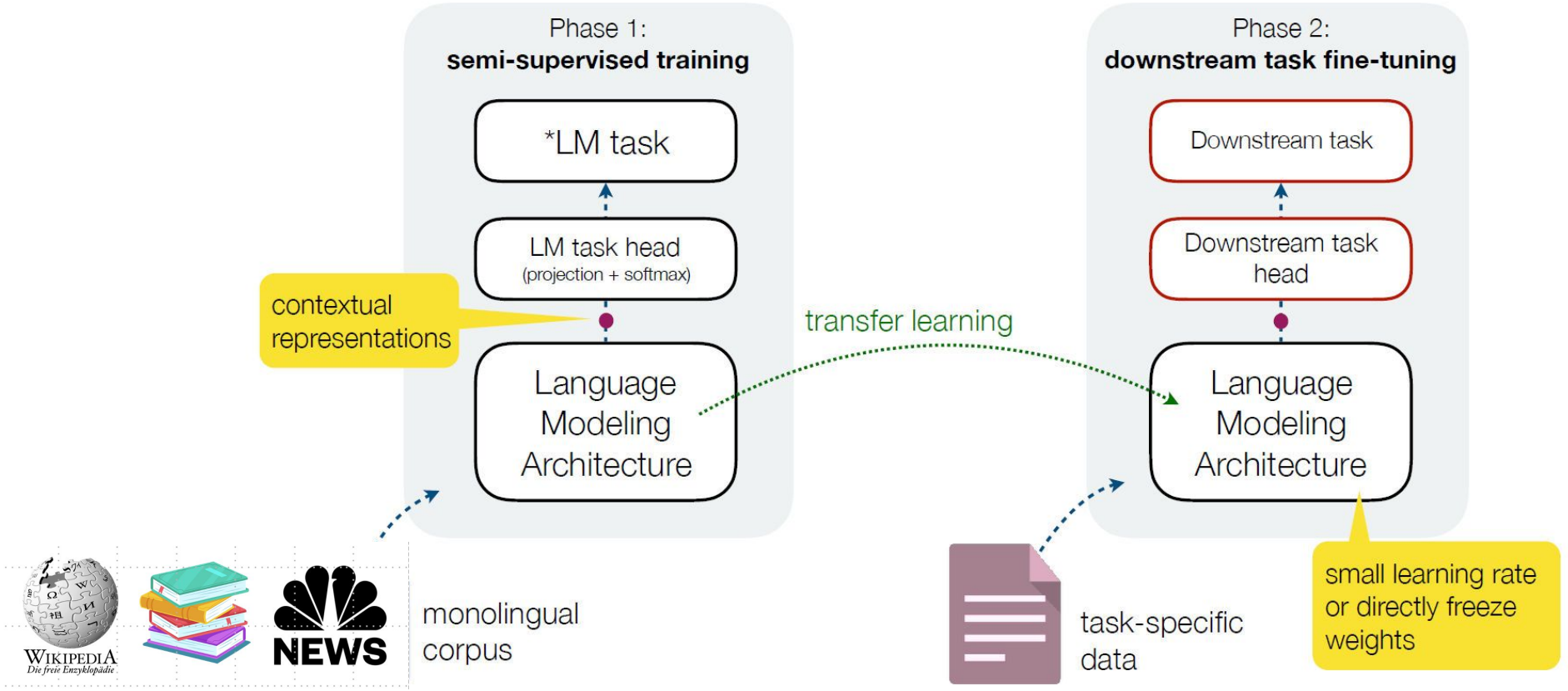
Beam Search





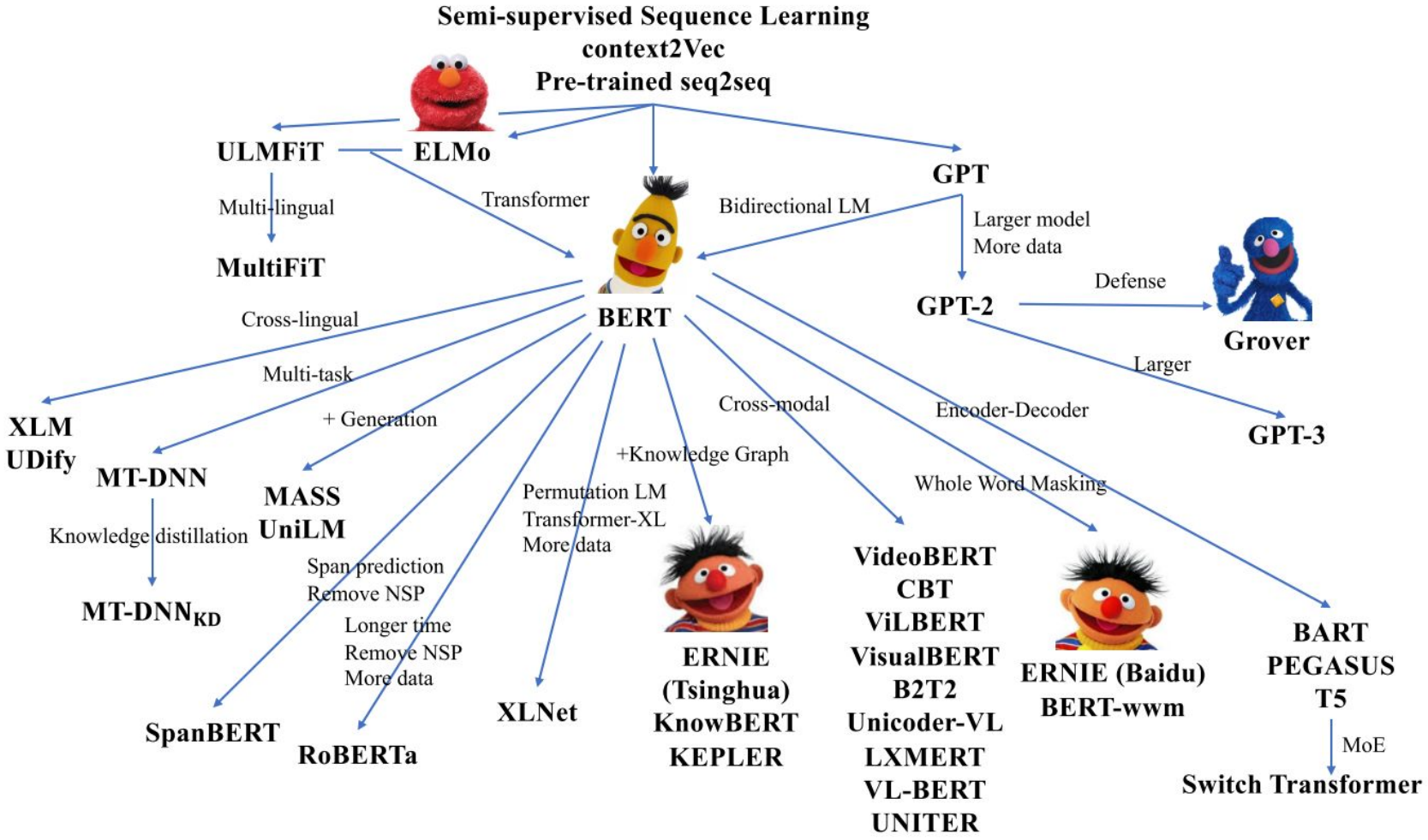
5 - Pretrained Language Models

❖ 5.1 Pre-Trained Language Model



5 - Pretrained Language Models

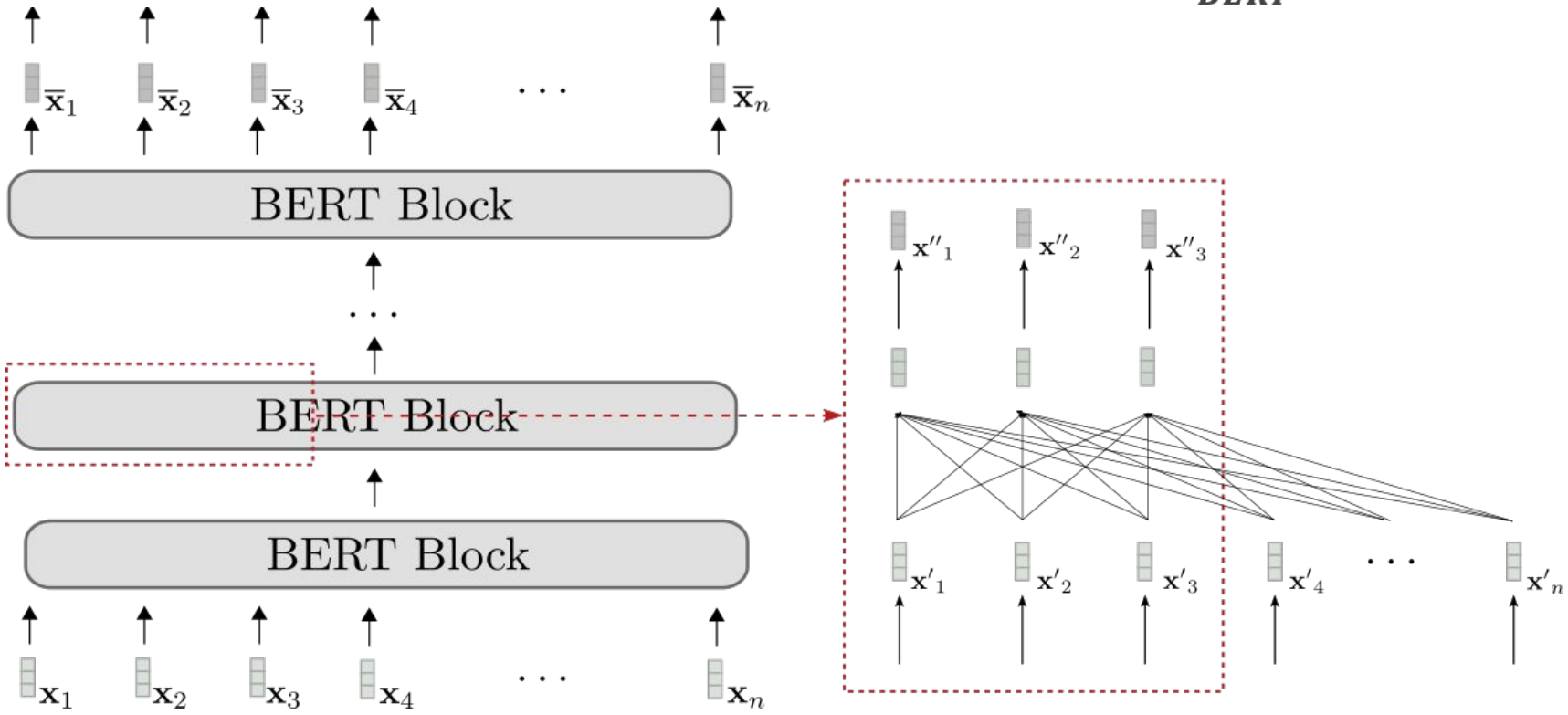
❖ 5.1 Pre-Trained Language Model



5 - Pretrained Language Models

❖ 5.2. BERT

- BERT: An encoder-only model
- Maps an input sequence to a contextualized sequence: $f_{\theta_{BERT}}: X_{1:n} \rightarrow \bar{X}_{1:n}$

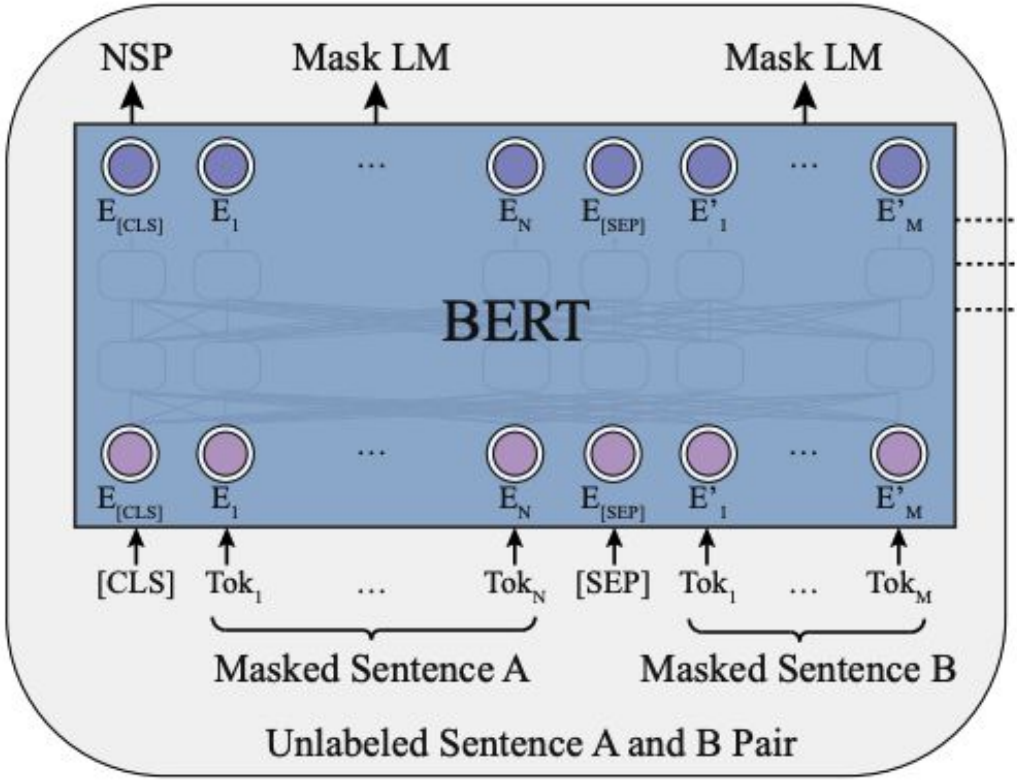


5 - Pretrained Language Models

❖ 5.2. BERT

Objective Functions:

- ❖ Masked LM (15% token):
  - ❑ 80%: replace with [MASK]
  - ❑ 10%: replace with a random word
  - ❑ 10%: keep unchanged
- ❖ Next Sentence Prediction (NSP)
  - ❑ Classification Task
  - ❑ 2 Labels: IsNext and NotNext
  - ❑ Use [SEP] token

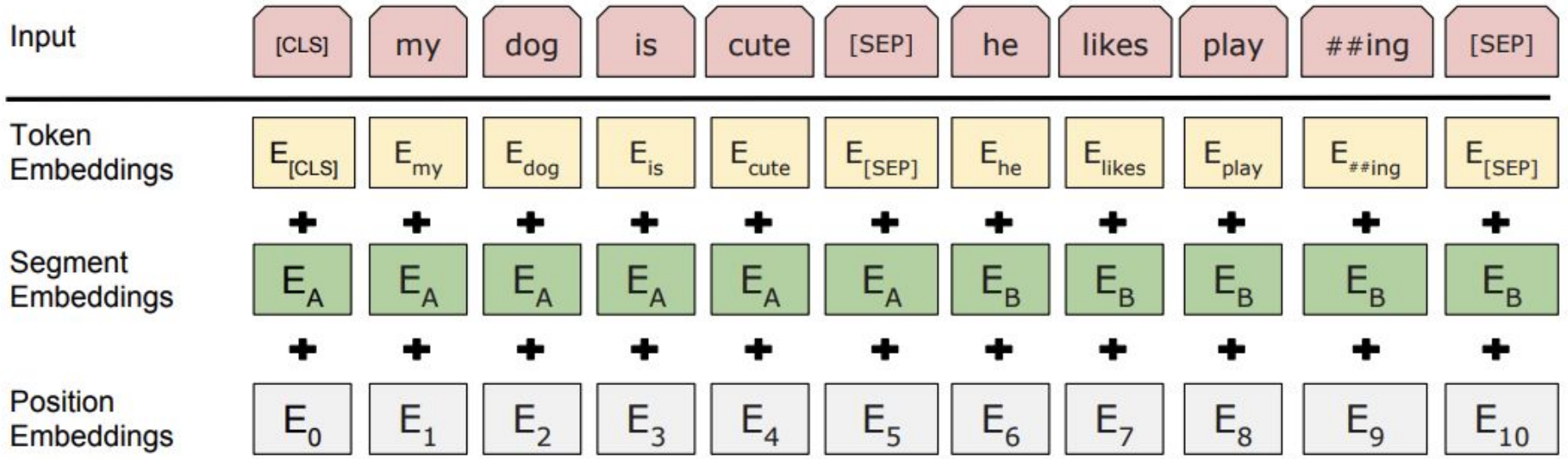


Pre-training

5 - Pretrained Language Models

❖ 5.2. BERT

❖ BERT input representation

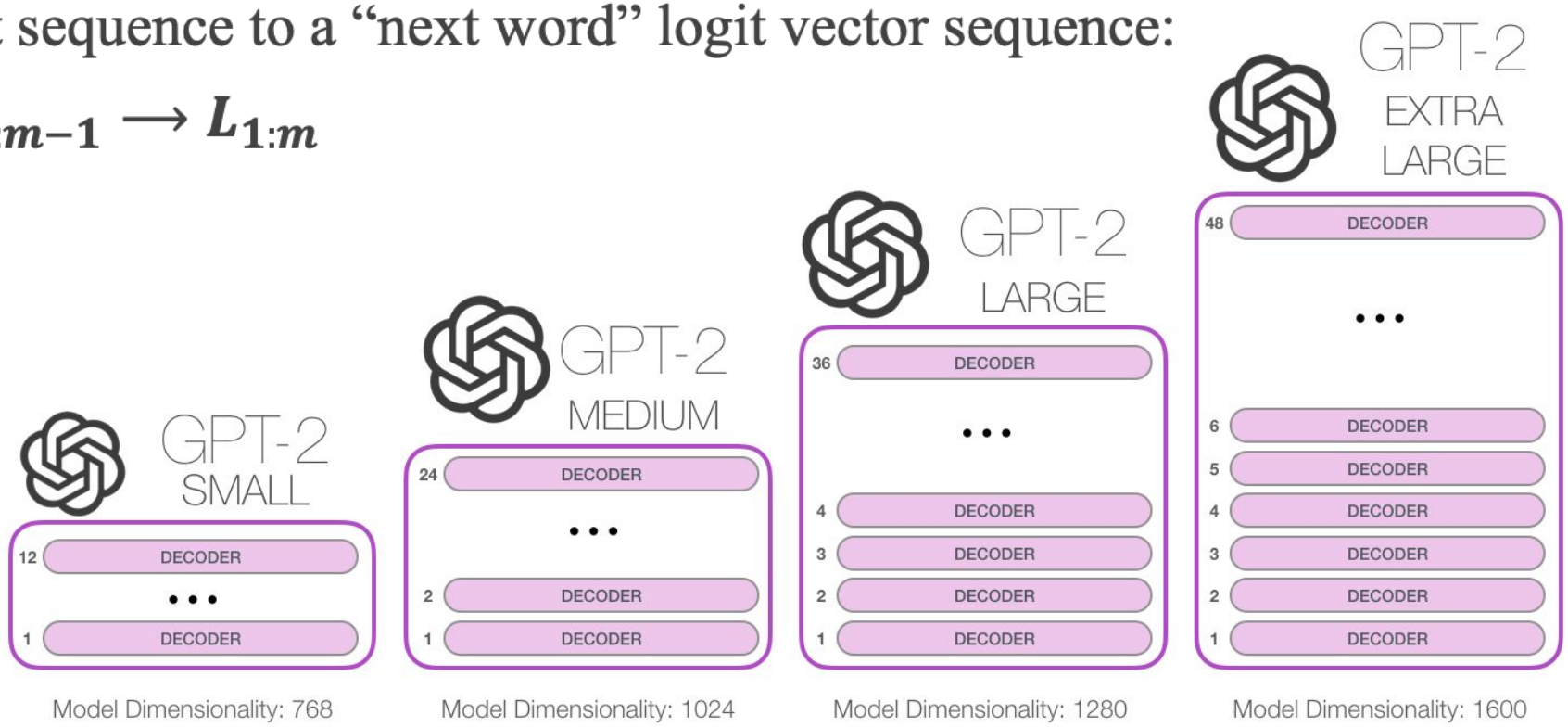


5 - Pretrained Language Models

❖ 5.3. GPT (GPT2)

- GPT2: A decoder-only model, use uni-directional (causal) self-attention
- Maps an input sequence to a "next word" logit vector sequence:

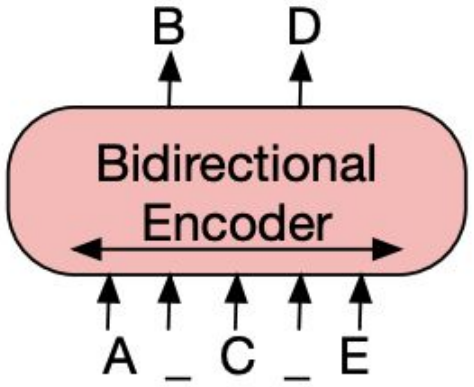
$$f_{\theta_{GPY2}}: X_{0:m-1} \longrightarrow L_{1:m}$$



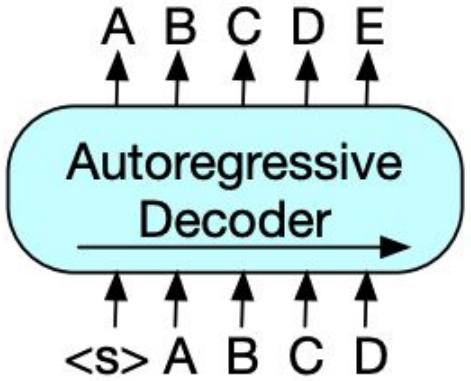
5 - Pretrained Language Models

❖ 5.4. BART

- **BERT and GPT2:** a great catalyst for NLU
- But, less successful for sequence-to-sequence tasks: machine translation, text summarization,…



Missing tokens are predicted independently, so BERT cannot easily be used for generation

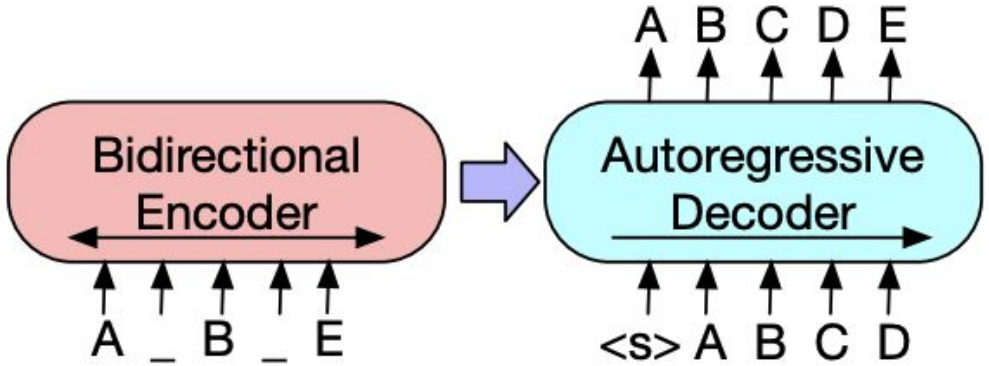


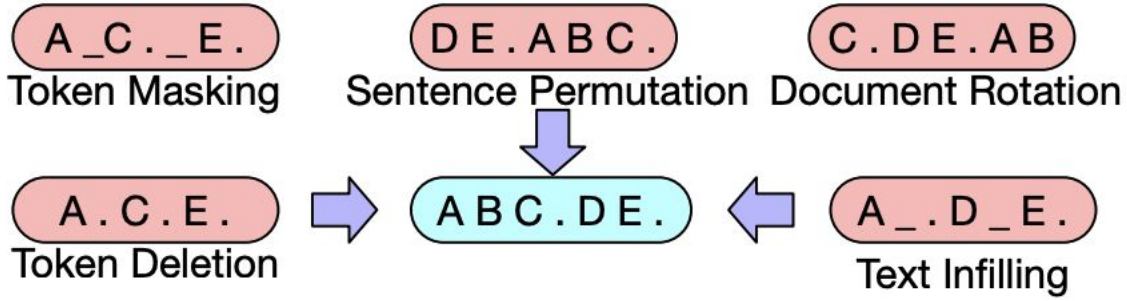
Tokens can only condition on leftward context, so it cannot learn bidirectional interactions

5 - Pretrained Language Models

❖ 5.4. BART

- **BART:** Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation and Comprehension





5 - Pretrained Language Models

❖ 5.4. BART

GLUE (Wang et al., 2019) *(dev set, single model, single-task finetuning)*

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B
`roberta.large`	90.2	94.7	92.2	86.6	96.4	90.9	68.0	92.4
`bart.large`	89.9	94.9	92.5	87.0	96.6	90.4	62.8	91.2

SQuAD (Rajpurkar et al., 2018) *(dev set, no additional data used)*

Model	SQuAD 1.1 EM/F1	SQuAD 2.0 EM/F1
`roberta.large`	88.9/94.6	86.5/89.4
`bart.large`	88.8/94.6	86.1/89.2

CNN/Daily Mail *(test set, no additional data used)*

Model	R1	R2	RL
`BERTSUMEXTABS`	42.13	19.60	39.18
`bart.large`	44.16	21.28	40.90

5 - Pretrained Language Models

❖ 5.4. BART

- **mBART50:** Multilingual Translation with Extensible Multilingual Pretraining

Data size	Languages
10M+	German, Czech, French, Japanese, Spanish, Russian, Polish, Chinese
1M - 10M	Finnish, Latvian, Lithuanian, Hindi, Estonian
100k to 1M	Tamil, Romanian, Pashto, Sinhala, Malayalam, Dutch, Nepali, Italian, Arabic, Korean, Hebrew, Turkish, Khmer, Farsi, Vietnamese, Croatian, Ukrainian
10K to 100K	Thai, Indonesian, Swedish, Portuguese, Xhosa, Afrikaans, Kazakh, Urdu, Macedonian, Telugu, Slovenian, Burmese, Georgia
10K-	Marathi, Gujarati, Mongolian, Azerbaijani, Bengali

5 - Pretrained Language Models

❖ 5.4. BART

❖ **PhoMT dataset**

Model	Validation set				Test set					
	En-to-Vi		Vi-to-En		En-to-Vi			Vi-to-En		
	TER↓	BLEU↑	TER↓	BLEU↑	TER↓	BLEU↑	Human↑	TER↓	BLEU↑	Human↑
Google Translate	45.86	40.10	44.69	36.89	46.52	39.86	23/100	45.86	35.76	10/100
Bing Translator	45.36	40.82	45.32	36.61	46.04	40.37	14/100	46.09	35.74	15/100
Transformer-base	42.77	43.01	43.42	38.26	43.79	42.12	13/100	44.28	37.19	13/100
Transformer-big	42.13	43.75	43.08	39.04	43.04	42.94	18/100	44.06	37.83	28/100
mBART	**41.56**	**44.32**	**41.44**	**40.88**	**42.57**	**43.46**	**32/100**	**42.54**	**39.78**	**34/100**

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

- Two main criteria:
  - Adequacy: Translation $w^{(t)}$ should adequately reflect the linguistic content of $w^{(s)}$
  - Fluency: Translation $w^{(t)}$ should be fluent text in the target language
- Different translations of "A Vinay le gusta Python"

	Adequate?	Fluent?
To Vinay it like Python	yes	no
Vinay debugs memory leaks	no	yes
Vinay likes Python	yes	yes

Adequancy	
5	All meaning
4	Most meaning
3	Much meaning
2	Little meaning
1	None

Fluency	
5	Flawless English
4	Good English
3	Non-native English
2	Disfluent English
1	Incomprehensible

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

- Manual evaluation is most accurate, but expensive
- Automated evaluation metrics:
  - Compare system hypothesis with reference translations
  - BLEU Score (BiLingual Evaluation Understudy): Modified n-gram Precision
    - N-gram overlap between machine translation output and reference translation
    - Compute precision for n-grams of size 1 to 4
    - Add brevity penalty (foo too short translations)
    - Typically computed over the entire corpus, not single sentence
  - SacreBLEU Score (A Call for Clarity in Reporting BLEU Scores)

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

BLEU

Precision 1-gram: 5/8

Target Sentence: The guard arrived late because ~~it was raining~~

Predicted Sentence: The guard arrived late because of the rain

Precision 2-gram: 4/7

Target Sentence: The guard arrived late because it was raining

Predicted Sentence: The guard arrived late because of the rain

Precision 3-gram: 3/6

Precision 4-gram: 2/5

$$\text{BLEU} = \min\left(1, \frac{\text{output} - \text{length}}{\text{reference} - \text{length}}\right)\left(\prod_{i=1}^{4} \text{precision}_1\right)^{1/4}$$

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

Precision and Recall of words

System A	A	officials	~~responsibility~~	~~of~~	airport	~~safety~~	
Reference	A	officials	are	responsible	for	airport	security

- Precision:

$$\frac{\text{correct}}{\text{output} - \text{length}} = \frac{3}{6} = 50\%$$

- Recall:

$$\frac{\text{correct}}{\text{reference} - \text{length}} = \frac{3}{7} = 43\%$$

- F-measure:

$$\frac{P \times R}{(P + R)/2} = \frac{0.5 \times 0.43}{(0.5 + 0.43)/2} = 46\%$$

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

Precision and Recall of words

- Flaw: no penalty for reordering

System A	A	officials	~~responsibility~~	~~of~~	airport	~~safety~~	
Reference	A	officials	are	responsible	for	airport	security
System B	airport	security	A	officials	are	responsible	

Metric	System A	System B
Precision	50%	100%
Recall	43%	100%
F-measure	46%	100%

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

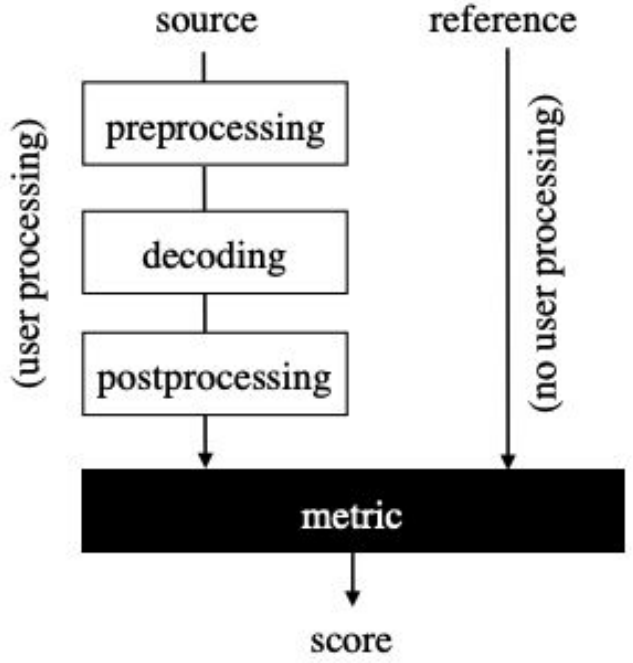
$$Brevity\ Penalty = \begin{cases} 1, & \text{if } c > r \\ e^{(1-r/c)}, & \text{if } c <= r \end{cases}$$

c là *predicted length* = số lượng từ có trong predicted sentence

r là *target length* = số lượng từ có trong target sentence

BLEU

$$\text{logBLEU} = \min\left(1 - \frac{r}{c}, 0\right) + \sum_{n=1}^{N} w_n \log p_n$$



r: reference-length, c: output (candidate)-length

n: n-gram (1,2,3,4), w_n: weight of n-gram

uniform weights $w_n = 1/n$

p_n: precision n-gram

SacreBLEU (A Call for Clarity in Reporting BLEU)

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

BLEU

System A	A	officials	~~responsibility~~	~~of~~	~~airport~~	~~safety~~	
Reference	A	officials	are	responsible	for	airport	security
System B	airport	security	A	officials	are	responsible	

2 -gram

Metric	System A	System B
Precision (1 gram)	3/6	6/6
Precision (2 gram)	1/5	4/5
Precision (3 gram)	0/4	2/4
Precision (4 gram)	0/3	1/4
Brevity penalty	6/7	6/7
BLEU	0	0.52

6 - Metrics and Performance

❖ 6.1 Evaluate metrics

BLEU

BLEU Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

The following color gradient can be used as a general scale interpretation of the BLEU score:

6 - Metrics and Performance

❖ 6.2. Compare Models

❖ **Dataset: IWSLT'15 English-Vietnamese - Link Code and Model**

Training: 133 317 Validation: 1 553 Test: 1 268

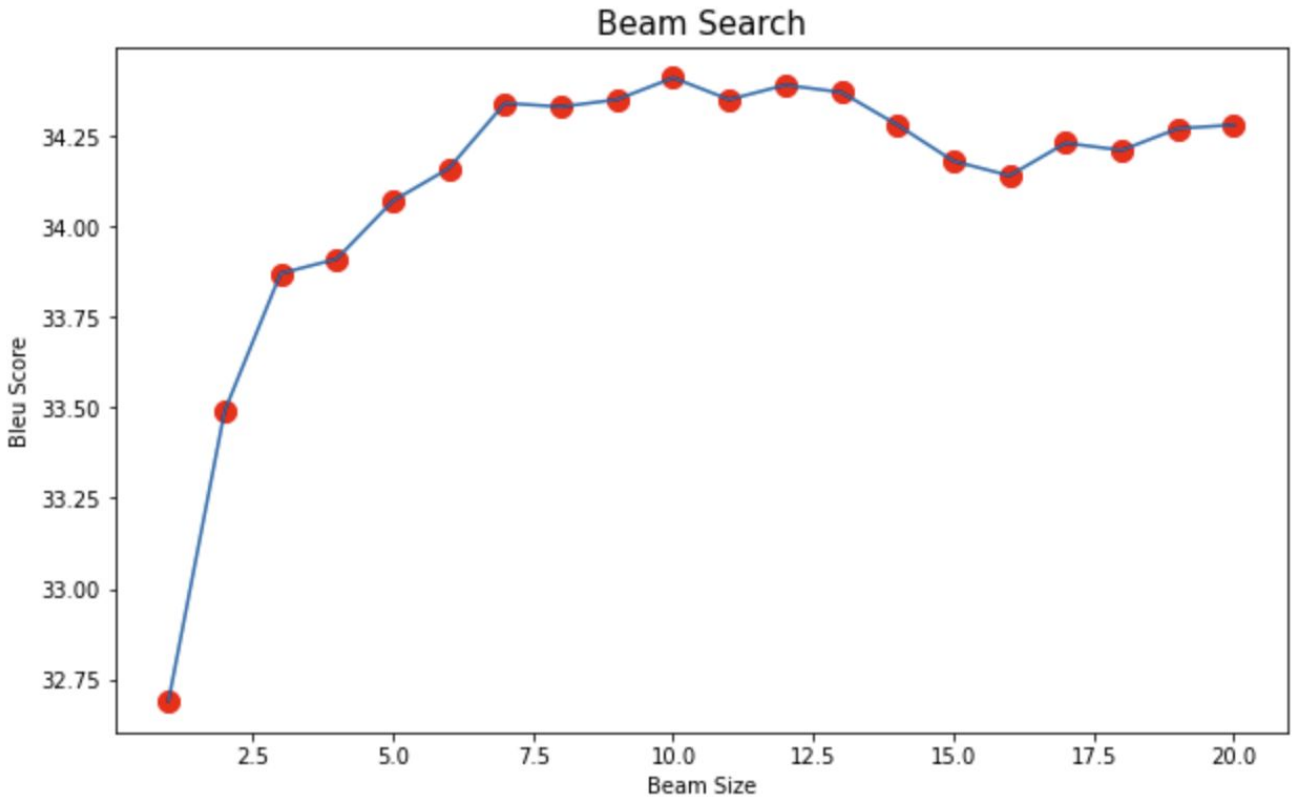
Experiment	Model	ScareBLEU
#1	Standard Transformer (Greedy Search)	24.66 55.9/30.3/18.5/11.8
#2	BERT-to-BERT (Greedy Search)	23.06 48.2/28.0/17.9/11.7
#3	BERT-to-BERT (Beam Search, Size=5)	23.72 49.5/28.7/18.4/12.1
#4	BERT-to-GPT2 (Greedy Search)	23.56 49.1/28.5/18.4/12.0
#5	BERT-to-GPT2 (Beam Search, Size=5)	23.46 48.6/28.3/18.3/12.1
#6	mBART50 (Greedy Search)	32.69 **64.3**/41.3/29.0/20.9
#7	mBART50 (Beam Search, Size=5)	**34.07** 62.8/**40.9/29.1/21.2**

6 - Metrics and Performance

6.2. Compare Models

- **Model: mBART50**

Experiment	Beam Size	ScareBLEU
#1	1 (Greedy Search)	32.69 **64.3**/**41.3**/29.0/20.9
#2	2	33.49 63.7/41.1/29.1/21.1
#3	4	33.91 63.0/40.8/29.1/21.2
#4	6	34.16 62.8/40.8/29.1/21.3
#5	8	34.33 62.8/41.0/**29.3**/21.4
#6	10	**34.41** 62.7/41.0/**29.3**/**21.5**
#7	12	34.39 62.7/41.0/**29.3**/21.4
#8	14	34.28 62.5/40.7/29.1/21.3
#9	16	34.14 62.4/40.6/28.9/21.2
#10	18	34.21 62.3/40.6/29.0/21.2



7 - Our Machine Translation System

7 - Our Machine Translation System

7 - Our Machine Translation System