Image: Trawling for Babel fish. Concept and juxtaposition: Raeid Saqur.

Statistical + **Neural machine translation**

CSC401/2511 – Natural Language Computing – Fall 2024

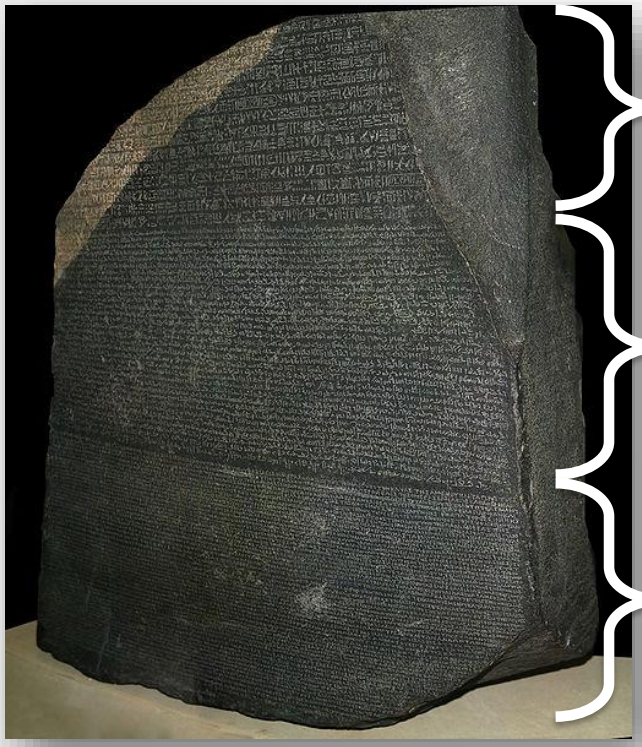
Gerald Penn

Lecture 6

University of Toronto

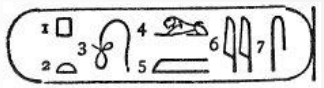
The Rosetta Stone

- The **Rosetta Stone** dates from 196 BCE.
  - It was re-discovered by French soldiers during Napoleon's invasion of Egypt in 1799 CE.

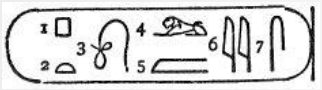


- It contains three **parallel** texts in different languages.
- Demotic had been partly deciphered.
- *For 20+ years after Rosetta's discovery, Egyptian hieroglyphics largely remained a mystery.*

Deciphering Rosetta

- During 1822–1832, **Jean-François Champollion** worked on the Rosetta stone. He noticed:
  1. The circled Egyptian symbols, e.g.  appeared in roughly the same positions as words like '*Ptolemy*' in Greek.
  2. The number of Egyptian hieroglyph tokens was **much larger** than the number of Greek words → Egyptian seemed to have been partially phonographic.
  3. Cleopatra's cartouche was written 

Aside – deciphering Rosetta

- So if  was *'Ptolemy'* and  was *'Cleopatra'* and the symbols corresponded to sounds – can we match up the symbols?

P	T	O	L	M	E	S		
C	L	E	O	P	A	T	R	A

- This approach demonstrated the value of working from **parallel texts** to decipher an unknown language:
  - *There are several examples of decipherment having been achieved without aligning unknown words in bitexts.*

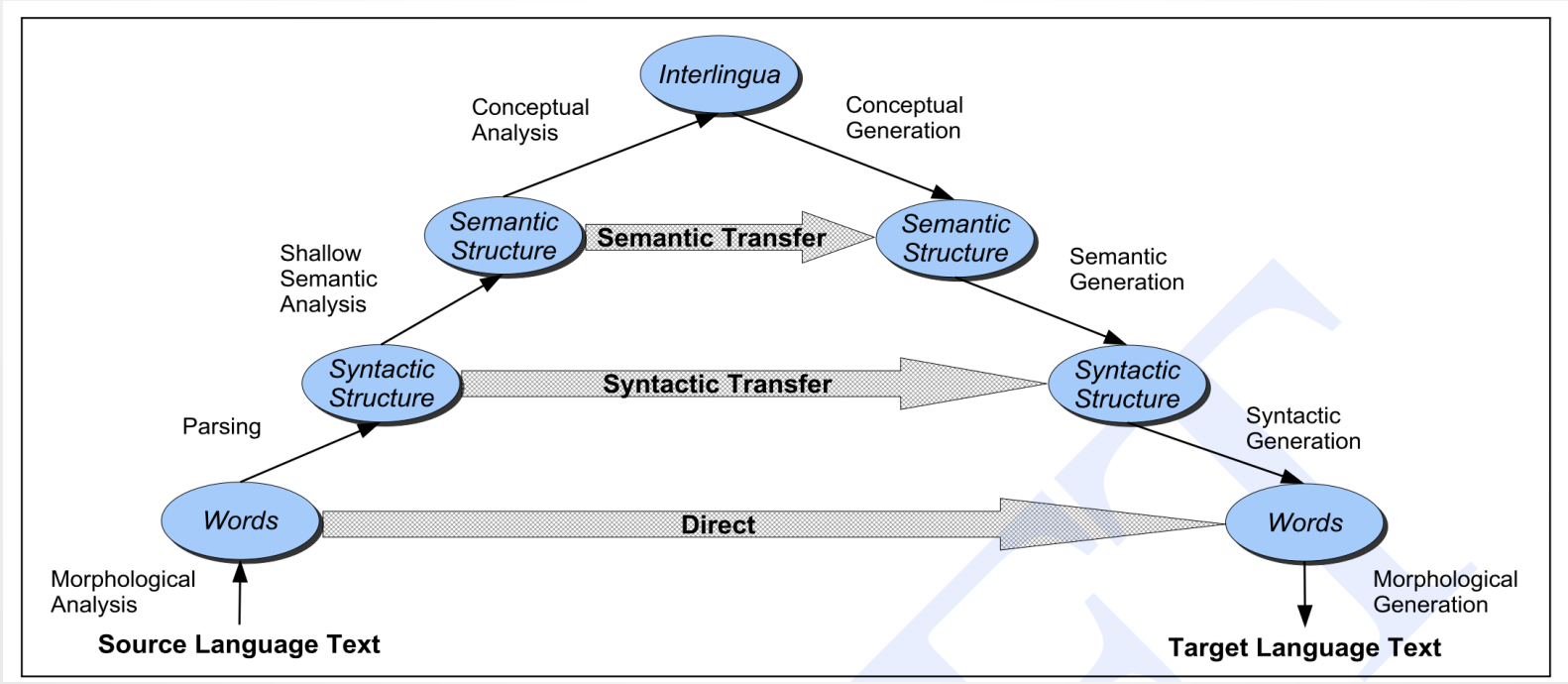
Circa 2016

- Where's my statistical machine translation (SMT)?
  - ABC's speech recognizer transcribes French as though it were English in the Prime Minister's bilingual remarks:

"…Nazi innings…
…recourse to ice packs
…I'd love the log trucks"

The Vauquois triangle (1968)

- High-level classes of methodologies:
  - "Direct" Translation
  - Syntactic Transfer
  - Semantic Transfer
  - Interlingua



“Direct” translation

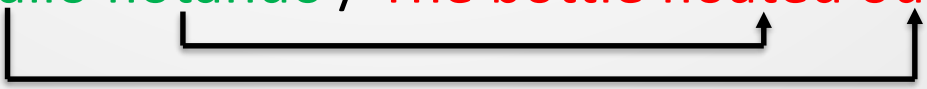
- A bilingual dictionary that aligns words across languages can be helpful, but only for certain cases.

¿	*Dónde*	*está*	*la*	*biblioteca*	*?*
	Where	*is*	*the*	*library*	*?*
	Où	*est*	*la*	*bibliothèque*	*?*

Mi	*nombre*	*es*	*T-bone*
My	*name*	*is*	*T-bone*
Mon	*nom*	*est*	*T-bone*

Difficulties in MT: typology

- Different **morphology** → difficult **mappings**, *e.g.*
  - Many (*polysynthetic*) vs one (*isolating*) roots per word
    - *polysynthetic*: e.g., Yupik
    - *isolating*: e.g., Cantonese
  - Many (*fusional*) vs few (*agglutinative*) *features* per morpheme
    - *fusional*: e.g., Russian
    - *agglutinative*: e.g., Turkish
- Different **head-position effects in syntax**, *e.g.*
  - SVO vs. SOV vs. VSO (e.g. English vs. Japanese vs. Arabic)
    - He listens to music / kare ha ongaku wo kiku
      - He: Subject; listens: Verb; to music: Object
      - kare: Subject; ha ongaku: Object; wo kiku: Verb
  - Satellite vs. nuclear-framed (e.g. Spanish vs. English)
    - La botella salió flotando / The bottle floated out

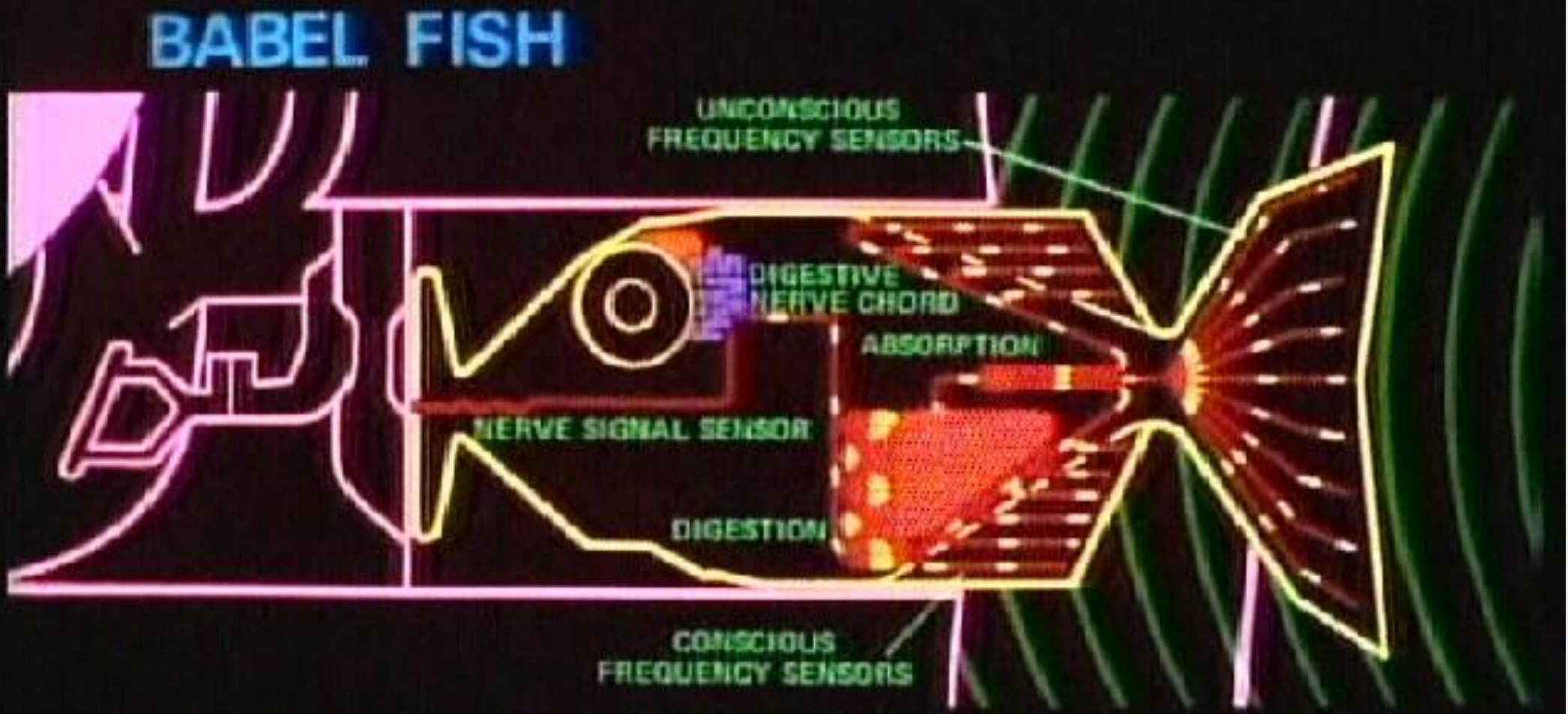


Difficulties in MT: ambiguity

- **Ambiguity** makes it hard to pick one translation
  - Lexical: many-to-many word mappings

  - Syntactic: same token sequence, different structure
    - Rick hit the Morty [with the stick]$_{PP}$ / Rick golpeó el Morty con el palo
    - Rick hit the Morty [with the stick]$_{PP}$ / Rick golpeó el Morty que tenia el palo
  - Semantic: same structure, different meanings
    - I'll pick you up / {Je vais te chercher, Je vais te ramasser}
  - Pragmatic: different contexts, different interpretations
    - Poetry vs technical report



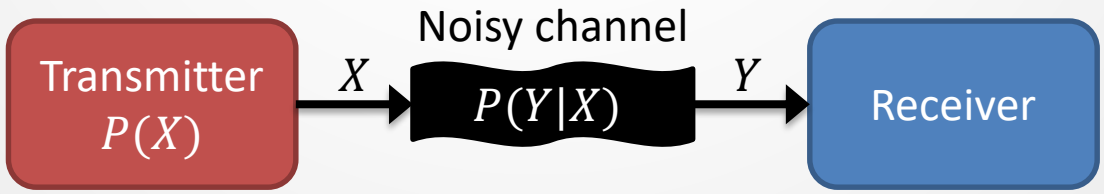
THE NOISY CHANNEL

The noisy channel model

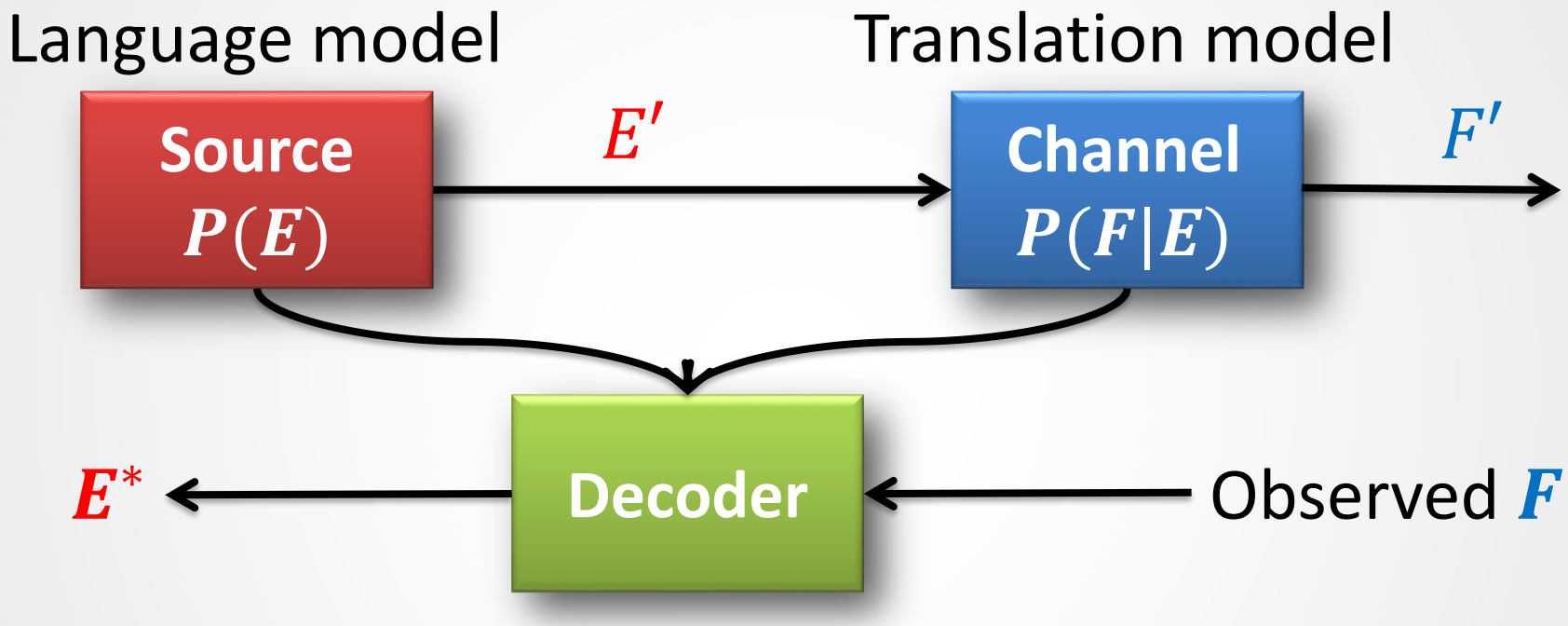
- Imagine that you're given a French sentence, F, and you want to convert it to the best corresponding English sentence, E^*
  - i.e., $$E^* = \underset{E}{\operatorname{argmax}}\, P(E|F)$$
- Use Bayes's Rule:

$$E^* = \operatorname{argmax}_E \frac{P(F|E)P(E)}{P(F)}$$

- $P(F)$ doesn't change argmax



The noisy channel



$$E^* = \underset{E}{\operatorname{argmax}}\, P(F|E)P(E)$$

How to use the noisy channel

- How does this work?

$$E^* = \underset{E}{\operatorname{argmax}} \underbrace{P(F|E)}_{\text{Translation model}} \overbrace{P(E)}^{\text{Language model}}$$

- $P(E)$ is a **language model** (e.g., *N*-gram) and encodes knowledge of word order.
- $P(F|E)$ is a **word- (or phrase-)level translation model** that encodes only knowledge on an *unordered* basis.

- **Combining** these models can give us **fluency** and **consistency**, respectively.

How to use the noisy channel

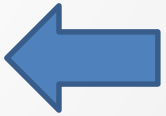
- Example from Koehn and Knight using only conditional likelihoods of Spanish words given English words.

- *Que hambre tengo yo*

  →

What hunger have I	$P(S\|E) = 1.4E^{-5}$
Hungry I am so	$P(S\|E) = 1.0E^{-6}$
I am so hungry	$P(S\|E) = 1.0E^{-6}$
Have I that hunger	$P(S\|E) = 2.0E^{-5}$

  …



Best translation using only the translation model

How to use the noisy channel

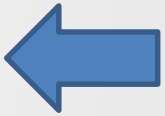
- … and with the English language model

- *Que hambre tengo yo*

  →

  What hunger have I $P(S|E)P(E) = 1.4E^{-5} \times 1.0E^{-6}$

  Hungry I am so $P(S|E)P(E) = 1.0E^{-6} \times 1.4E^{-6}$

  I am so hungry $P(S|E)P(E) = 1.0E^{-6} \times 1.0E^{-4}$ 

  Have I that hunger $P(S|E)P(E) = 2.0E^{-5} \times 9.8E^{-7}$

  …

How to learn $P(F|E)$?

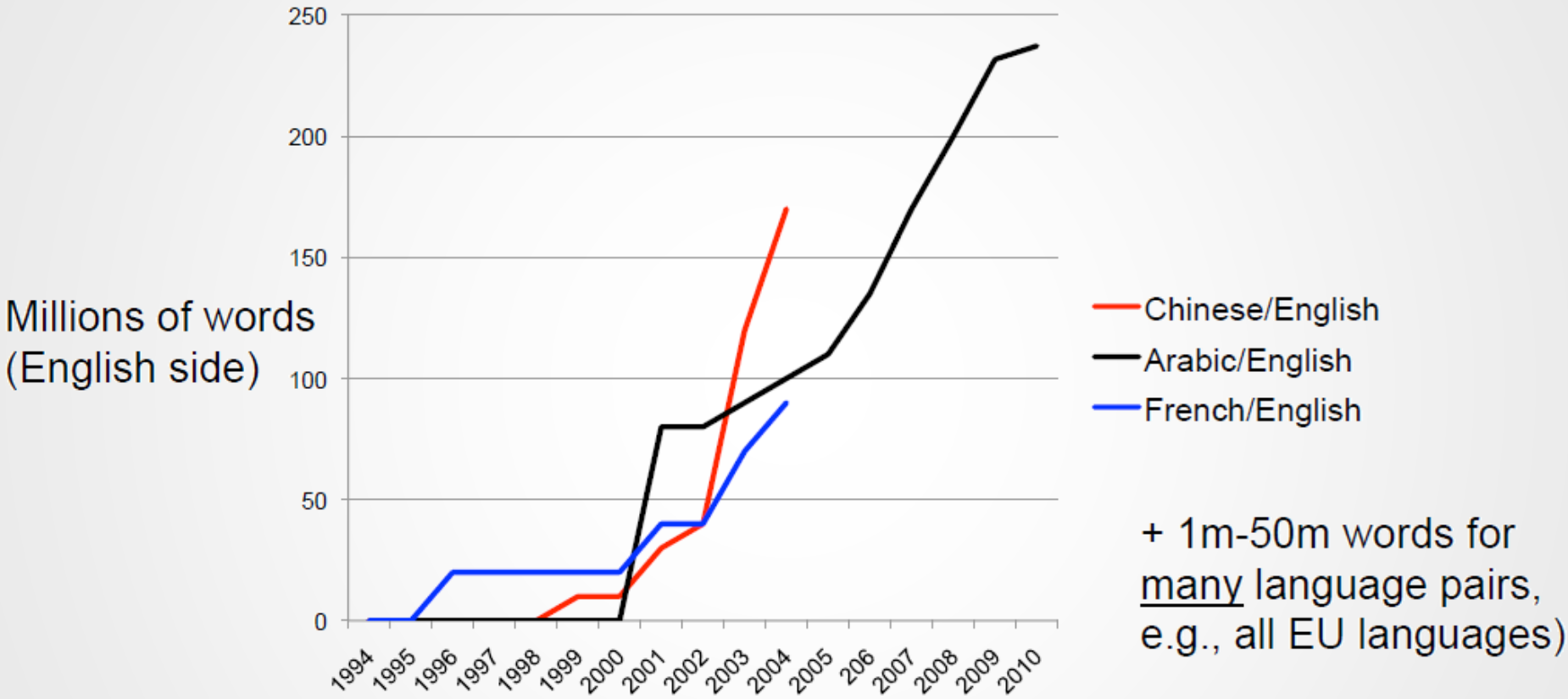
- Solution: collect statistics on vast parallel texts

… **citizen** of Canada has the **right** to vote in an election of members of the House of Commons or of a legislative assembly and to be qualified for membership …

… **citoyen** canadien a le **droit** de vote et est éligible aux élections législatives fédérales ou provinciales …

e.g., the *Canadian Hansards*: bilingual Parliamentary proceedings

Bilingual data



Source: Chris Manning's lecture slide

- Data from Linguistic Data Consortium (LDC) at University of Pennsylvania.

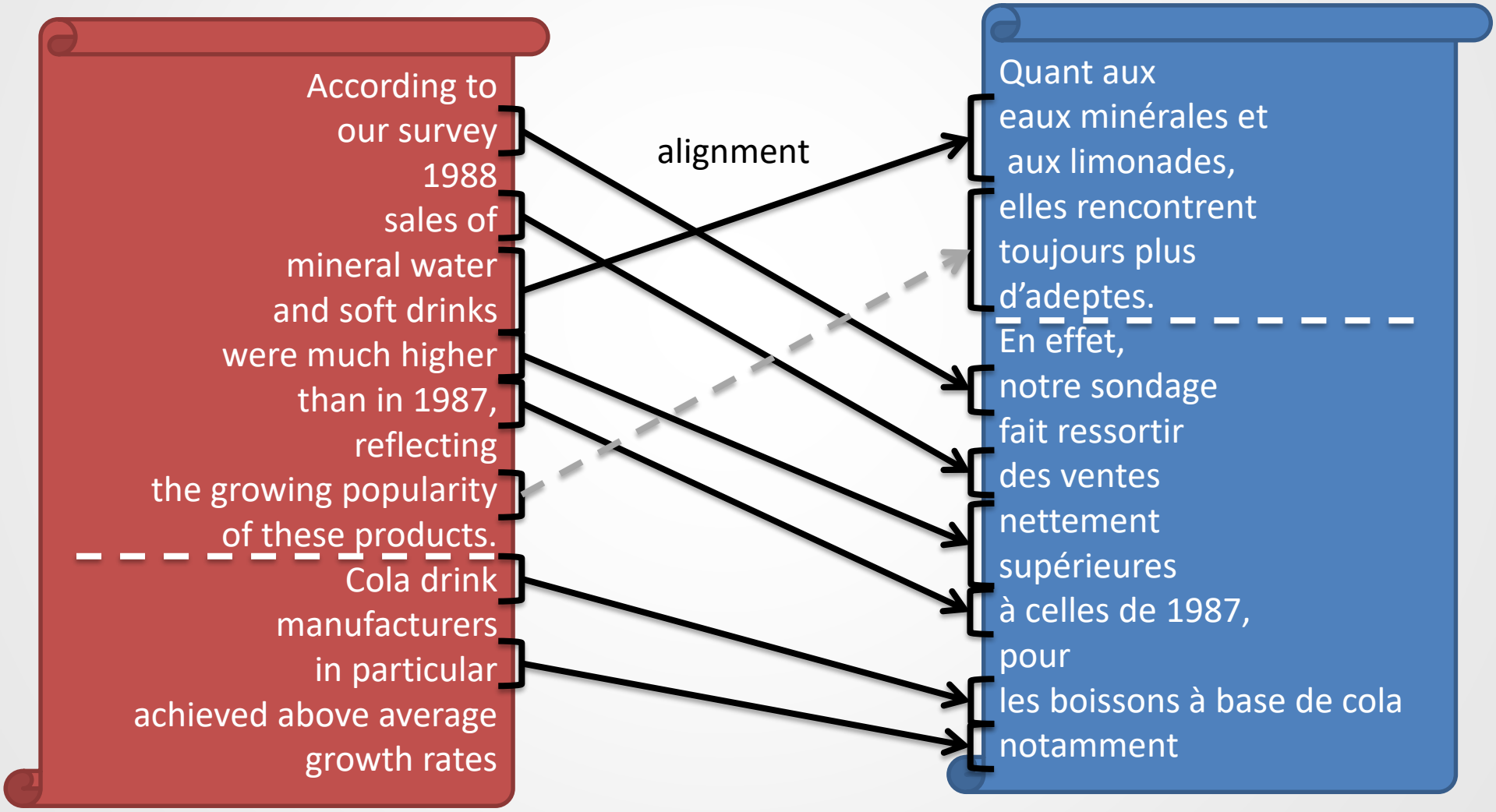
Alignments

- Alignments at different granularities
  - Word, phrase, sentence, document
- SMT makes alignments explicit
  - One block of source text entirely responsible for a block (conditional independence) of target text
- Letting A index pairs of aligned blocks in bitext

$$P(F|E) = \sum_A P(F, A|E) = \sum_A P(A|E) \prod_i P(A(Ei)|E_i, A)$$

Alignment

- In practice, words and phrases can be out of order.



From Manning & Schütze

Alignment

- Also in practice, we're usually not given the alignment.

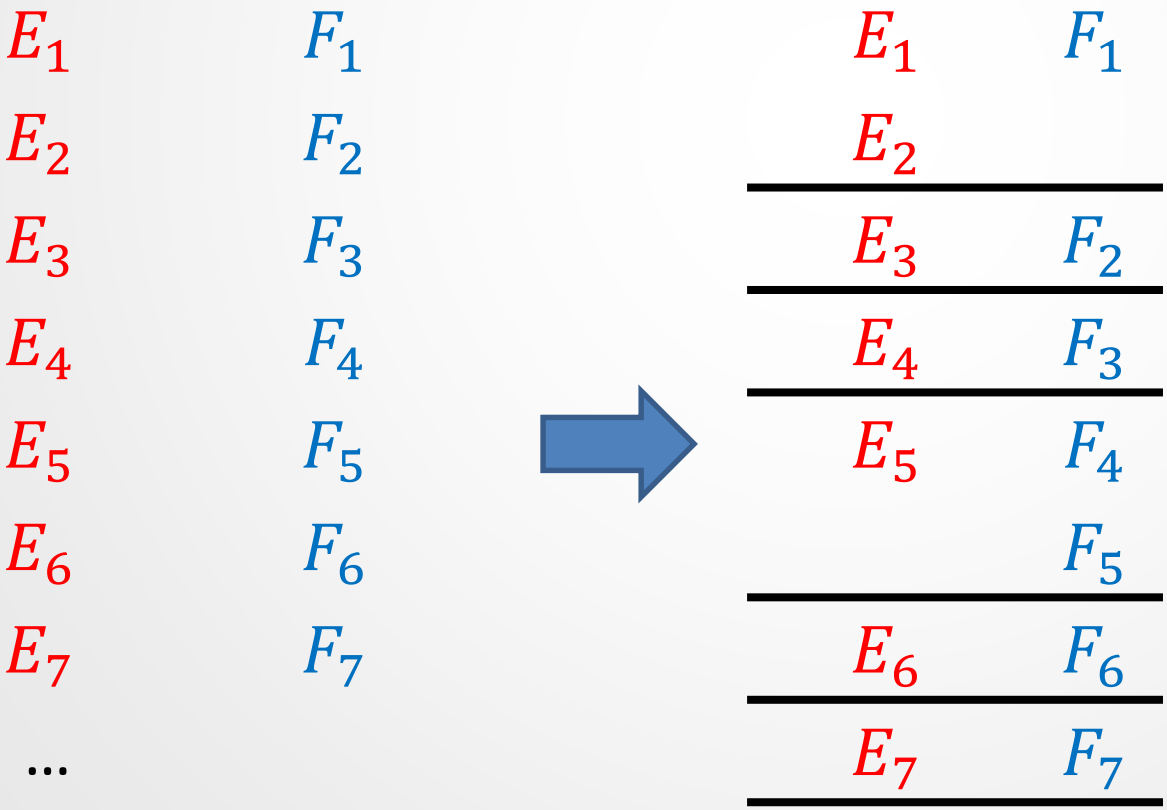
According to
our survey
1988
sales of
mineral water
and soft drinks
were much higher
than in 1987,
reflecting
the growing popularity
of these products.
Cola drink
manufacturers
in particular
achieved above average
growth rates

Quant aux
eaux minérales et
aux limonades,
elles rencontrent
toujours plus
d'adeptes.
En effet,
notre sondage
fait ressortir
des ventes
nettement
supérieures
à celles de 1987,
pour
les boissons à base de cola
notamment

From Manning & Schütze

Sentence alignment

- Sentences can also be **unaligned** across translations.
  - E.g., *He was happy.*$_{E1}$ *He had bacon.*$_{E2}$ →
    Il était heureux parce qu'il avait du bacon.$_{F1}$



Recalling $\prod_i P(F_{A_{i,1}}|E_{A_{i,2}})$:
$A_1 = (\{1\},\{1,2\})$
$A_2 = (\{2\},\{3\})$
$A_3 = (\{3\},\{4\})$
$A_4 = (\{4,5\},\{5\})$
Etc…

Sentence alignment

- Sentences can also be **unaligned** across translations.
  - E.g.,
    …il présente une toux qui ressemble à un cri de phoque ou un chien qui aboie. → *…The gentleman is mistaken.*

Word alignment models

- Make a simplifying assumption that every word in F maps to one E (i.e. $A_i = (\{i\}, \{j\}) \mapsto j$)
- E.g. IBM-1: $P(F|A,E) \propto \prod_i P\left(F_i \middle| E_{A_i}\right)$

$$\frac{Count\left(F_i, E_{A_i}\right)}{Count\left(E_{A_i}\right)}$$

- Trained via Expectation Maximization (see HMM lecture)

	Maria	no	dió	una	bofetada	a	la	bruja	verde
Mary	A_1								
did						A_6			
not		A_2							
slap			A_3	A_4	A_5				
the							A_7		
green									A_9
witch								A_8	

A **word alignment matrix** From J&M 2nd Ed.

Problems with word alignments

- What if some E_j isn't aligned anywhere?
- Need more flexible context!

$P(E|F)$

(For English to Spanish)

NP

NEURAL MACHINE TRANSL-ATION

SMT - Summary

- 1989-2014 SMT: huge research field
- So far, we only discussed the high-level ideas (e.g. *alignment*), omitting lots of details and caveats
- Best systems were extremely complex with many separately designed sub-components
- Lots of human effort & optimization for specific language pairs (e.g. SBMT for Arabic-English, PBMT for Chinese-English)
- Rule-based, hand-designed components never really were replaced in their entirety (e.g., headedness of NPs)

NMT – the breakout of Deep Learning in NLP

- Although there had been significant advances in neural language modelling and neural acoustic modelling beforehand, the NLP community remained resistant to embracing neural methods until a wildly successful attempt at neural MT in 2014. [1,2]
- NMT systems trained by a small group of engineers in a few months outperforms a SOTA heavily engineered SMT system.
- NMT remains an important rationalizer for neural methods in NLP – it was one of the first showcase tasks for attention mechanisms.

[1] Sutskever, Ilya, et al. "Sequence to sequence learning with neural networks." *NeurIPS* (2014).

[2] Bahdanau, Dzmitry, et al. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

What is NMT?

- Machine translation with neural networks
- *Usually* drops noisy channel: $E^* = \text{argmax}_E P(E|F)$
  - Some NMT researchers (e.g. "Simple and effective noisy channel modeling for neural machine translation," 2019. Yee *et al.*) use an objective inspired by the noisy channel
- No (explicit) alignments – often not even sentence-aligned
- Outperforms "SMT" by a large margin on poorly resourced language pairs.

Solving the alignment problem

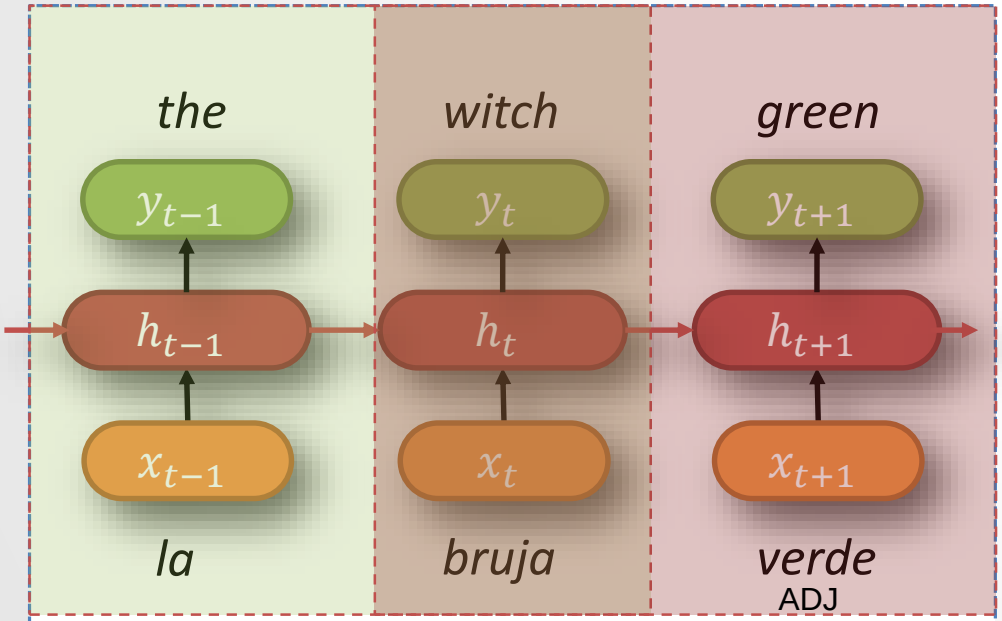
- Recall that source and target words (or, sentences) are not always one-to-one
- SMT solution is to marginalize explicit alignments
  - $E^* = \text{argmax}_E \sum_A P(F, A|E)P(E)$
- NMT uses "sequence-to-sequence (seq2seq)" encoder/decoder architectures
  - An **encoder** produces a representation of F
  - A **decoder** interprets that representation and generates an output sequence E

Seq2seq motivation

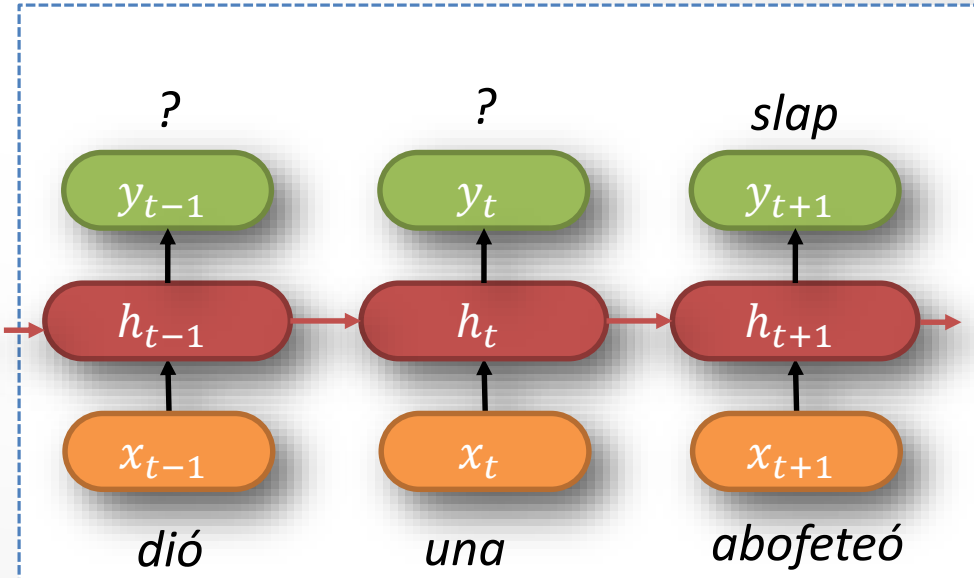
In the absence of any hierarchical structure (such as from a parser), there aren't a lot of options that are invariant to the choice of language pair.

Why not train an RNN to output a translated token from source token?

*"Mary no **dió una abofeteó** a la bruja verde." -> "Mary did not **slap** the green witch."*

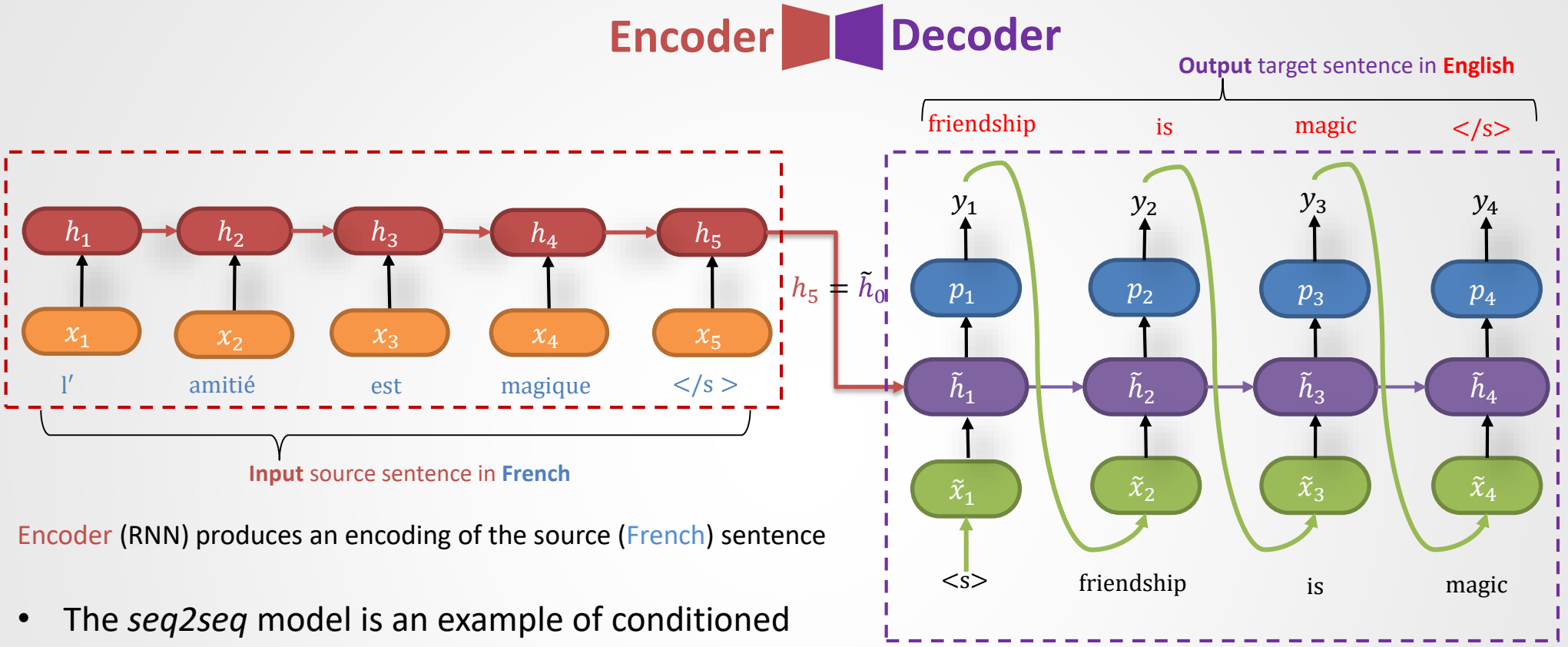


Different morphology: Adj, NN order not same



- len(src) == len(tgt) is an extreme restriction:
- Mapping is not always 1:1 (e.g. many:1)

NMT: the seq2seq model



Encoder (RNN) produces an encoding of the source (French) sentence

Decoder (RNN) generates target sentence (in English), conditioned on the encoding

- The *seq2seq* model is an example of conditioned language model (LM)
- Many variants exists. The classical (vanilla) seq2seq model outlined here
- NMT directly calculates $y^* = \text{argmax}_y P(y|x)$
- I.e. with our formulation:

$$E^* = \text{argmax}_E P(E|F)$$

Decoder is predicting the next word of the target sentence y

Prediction is **conditioned** on the source sentence **x**

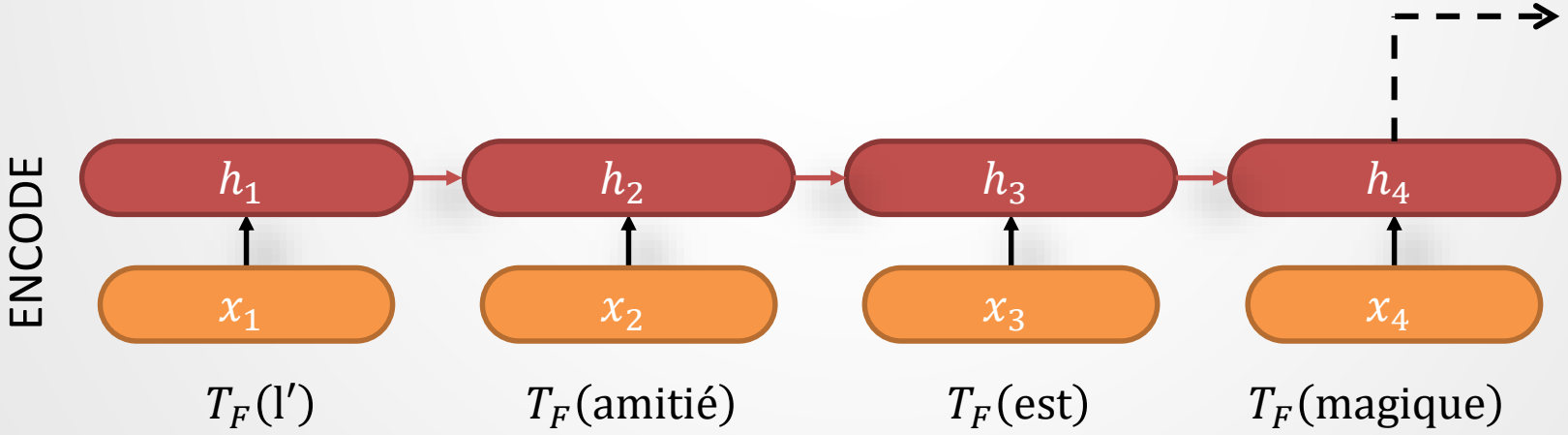
$$P(y|x) = P(y_1|x)P(y_2|y_1, x) \dots P(y_T|y_1, \dots y_{(T-1)}, x)$$

Notation

Term	Meaning
$F_{1:S}$	Source sequence (translating from)
$E_{1:T}$	Target sequence (translating to)
$x_{1:S}$	Input to encoder RNN (i.e. source embeddings $x_s = T_F(F_s)$)
$h_{1:S}^{(\ell,n)}$	Encoder hidden states (w/ optional layer index ℓ or head n)
$\tilde{x}_{1:T}$	Input to decoder RNN
$\tilde{h}_{1:T}^{(\ell,n)}$	Decoder hidden states (w/ optional layer index ℓ or head n)
$p_{1:T}$	Decoder output token distribution parameterization $p_t = f(\tilde{h}_t)$
$y_{1:T}$	Sampled output token from decoder $y_t \sim P(y_t \mid p_t)$
$c_{1:T}$	Attention context $c_t = Attend(\tilde{h}_t, h_{1:S}) = \sum_s \alpha_{t,s} h_s$
$e_{1:T,1:S}$	Score function output $e_{t,s} = score(\tilde{h}_t, h_s)$
$\alpha_{1:T,1:S}$	Attention weights $\alpha_{t,s} = \exp e_{t,s} / \sum_{s'} \exp e_{t,s'}$
$\tilde{z}_{1:T}^{(\ell)}$	Transformer decoder intermediate hidden states (after self-attention)

Encoder

- Encoder given source text $x = (x_1, x_2, \dots)$
  - $x_s = T_F(F_s)$ a source word embedding
- Outputs last hidden state of RNN
- Note $h_S = f(F_{1:S})$ conditions on entire source



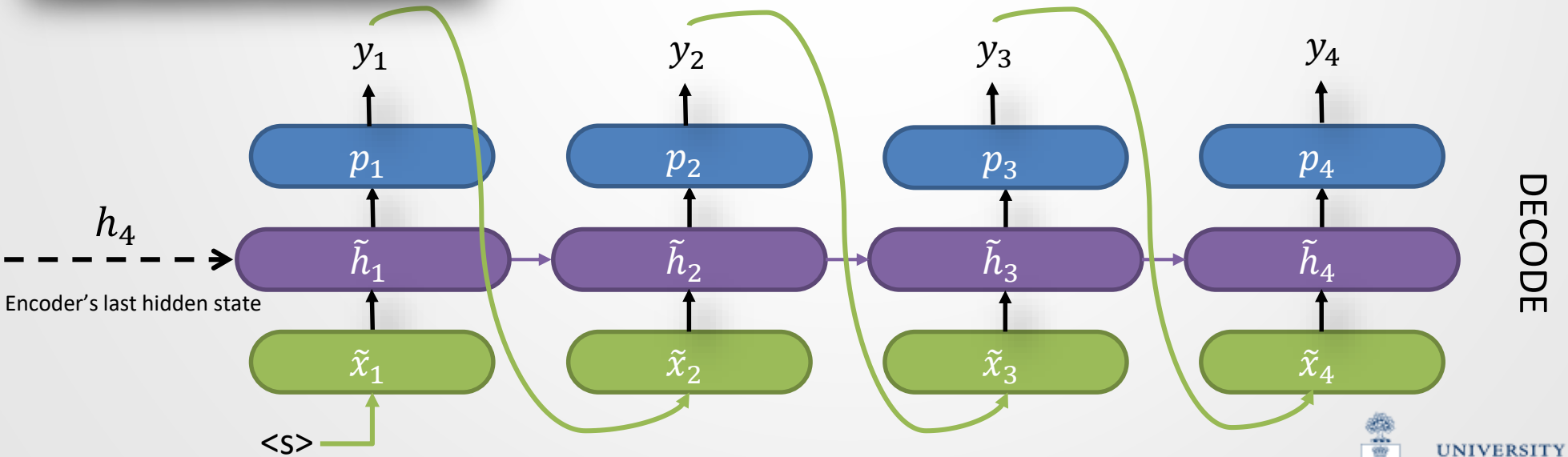
Source sentence (French): *L' amitié est magique*

Target sentence (English): *Friendship is magic* [Ground truth output]

Decoder

- **Sample** a target sentence word by word $y_t \sim P(y_t|p_t)$
- Set input to be embedding of **previously generated word** $\tilde{x}_t = T_E(y_{t-1})$
- $p_t = f(\tilde{h}_t) = f\left(g(\tilde{x}_t, \tilde{h}_{t-1})\right)$ is **deterministic**
- Base case: $\tilde{x}_1 = T_E(<s>),\ \tilde{h}_0 = h_S$
- $P(y_{1:T}|F_{1:S}) = \prod_t P(y_t|y_{<t}, F_{1:S})$ → **auto-regressive**

N.B.: Implicit $y_0 = <s>, P(y_0) = 1$



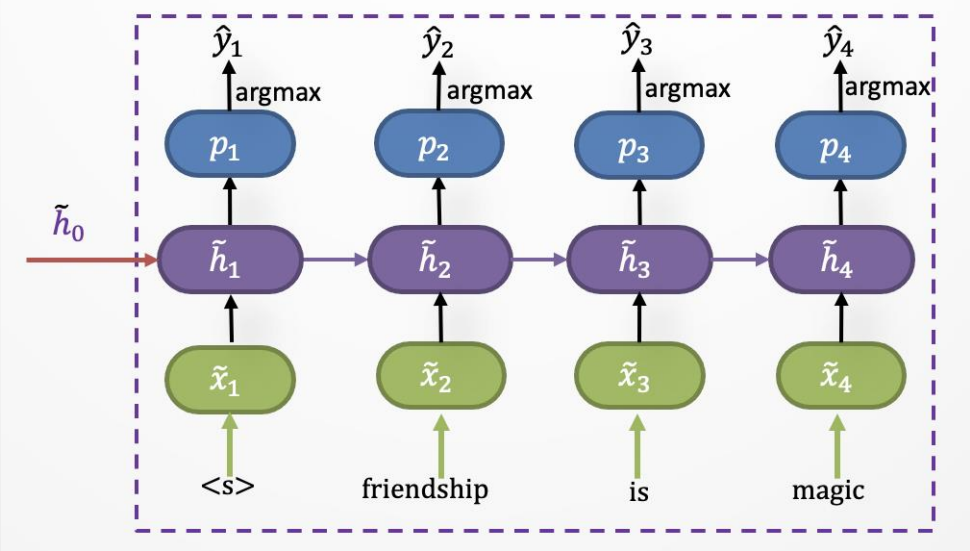
NMT: Training a MT system

- Train towards maximum likelihood estimate (MLE) against **one** translation E
- Auto-regression simplifies independence

MLE: $\theta^* = \text{argmin}_\theta \mathcal{L}(\theta|E,F)$

$$\mathcal{L}(\theta|E,F) = -\log P_\theta(y = E|F)$$
$$= -\sum_t \log P_\theta(y_t = E_t|E_{<t}, F_{1:S})$$

$$\mathcal{L} = -\log P(\text{friendship}|\cdots) - \log P(\text{is}|\cdots) - \log P(\text{magic}|\cdots) - \log P(\text{</s>}|\cdots)$$



Teacher forcing

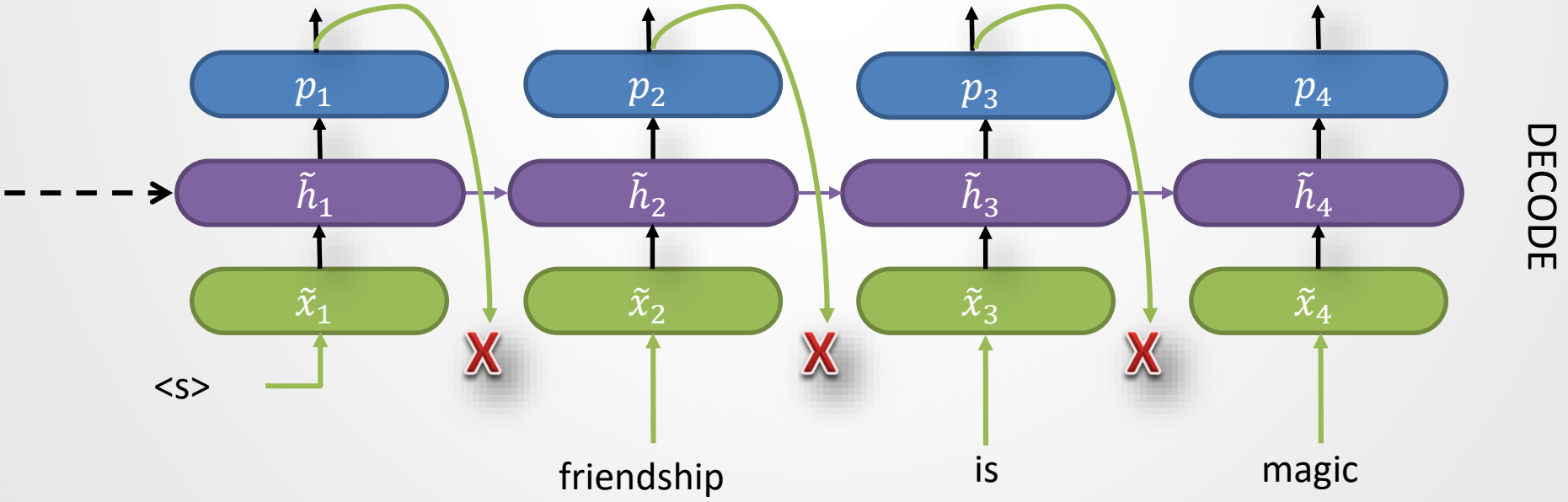
Core Idea

Remove feed-forward **recurrence** from the previous output to the hidden units at a time step and **replace** with ground-truth values for faster training

- Teacher forcing = maximum likelihood estimate (MLE)
- Replace $\tilde{x}_t = T(y_{t-1})$ with $\tilde{x}_t = T(E_{t-1})$

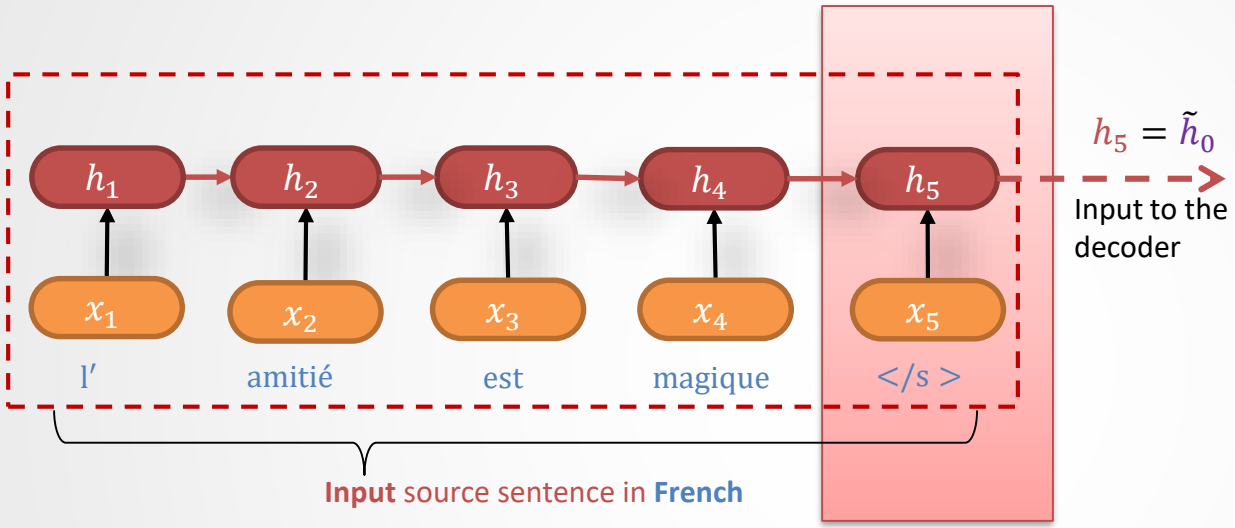
  Predicted output; target or ground truth
- Caveat: since $y_{t-1} \neq E_{t-1}$ in general, causes **exposure bias**

$$\mathcal{L} = -\log P(\text{friendship}|\cdots) - \log P(\text{is}|\cdots) - \log P(\text{magic}|\cdots) - \log P(</\text{s}>|\cdots)$$



Attention mechanisms - I

- The information bottleneck problem with vanilla *RNN* model



The encoder RNN output h_5 has to encode information from all preceding time steps.

Creates a bottleneck at h_5, due to the *vanishing gradient problem* for longer sequences

- **Solution**: sequence to sequence with **attention mechanism**[2]

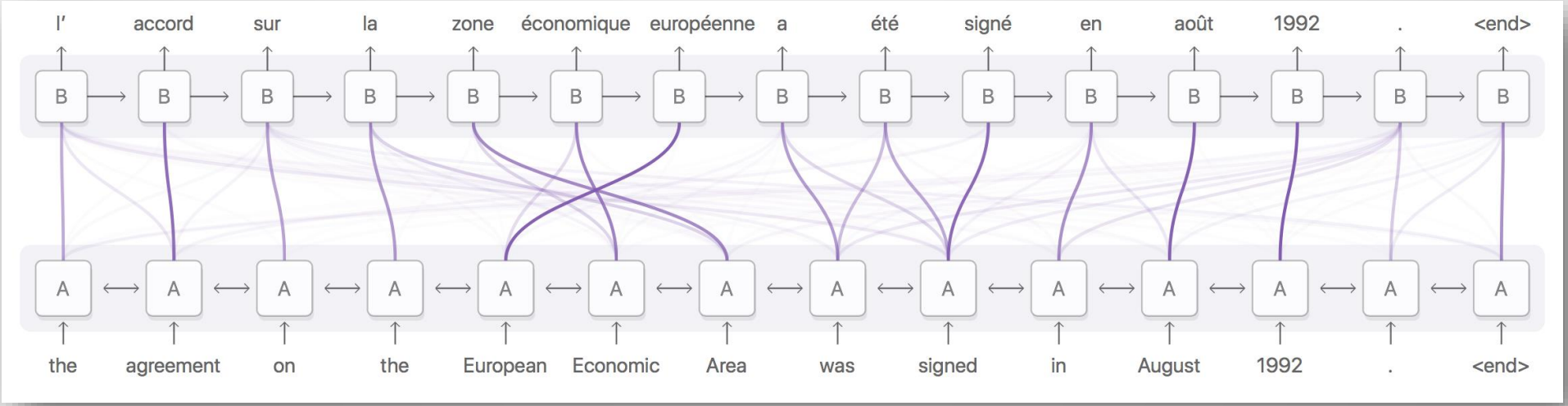
Core Idea

Use **direct connection** to the encoder states and **focus** on selective, **relevant parts** of the **source sequence** at every step of the decoder

[2] Bahdanau, Dzmitry, et al. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Attention mechanisms - II

- Allow decoder to "**attend**" (or, *query*) to certain areas of input (*values*) when making decisions. (Warning: correlation ≠ causation!) [1,2]
- Combines input from sequence dimension $h_{1:S}$ in a context-dependent way



Imagery from the excellent https://distill.pub/2016/augmented-rnns/#attentional-interfaces .

[1] Jain, Sarthak, and Byron C. Wallace. "Attention is not explanation." *arXiv preprint arXiv:1902.10186* (2019)

[2] Wiegreffe, Sarah, and Yuval Pinter. "Attention is not not explanation." *arXiv preprint arXiv:1908.04626* (2019)

Attention mechanisms

- Input to decoder a weighted sum of **all** encoder states
- Weights determined **dynamically by decoder's previous hidden state**
- $\tilde{x}_t = [c_{t-1}; T_E(y_{t-1})]$
  - 1. Attention scores $a_{t,1:S} = score(\tilde{h}_t, h_{1:S})$
  - 2. Weights $\alpha_{t,s} = softmax(a_{t,1:S}, s) = {\exp a_{t,s}}/{\sum_{s'} \exp a_{t,s'}}$
  - 3. Context vector $c_t = Attend(\tilde{h}_t, h_{1:S}) = \sum_s \alpha_{t,s} h_s$
- Score function, usually $score(a, b) = |a|^{-1/2}\langle a, b\rangle$ (scaled **dot-product** attention).

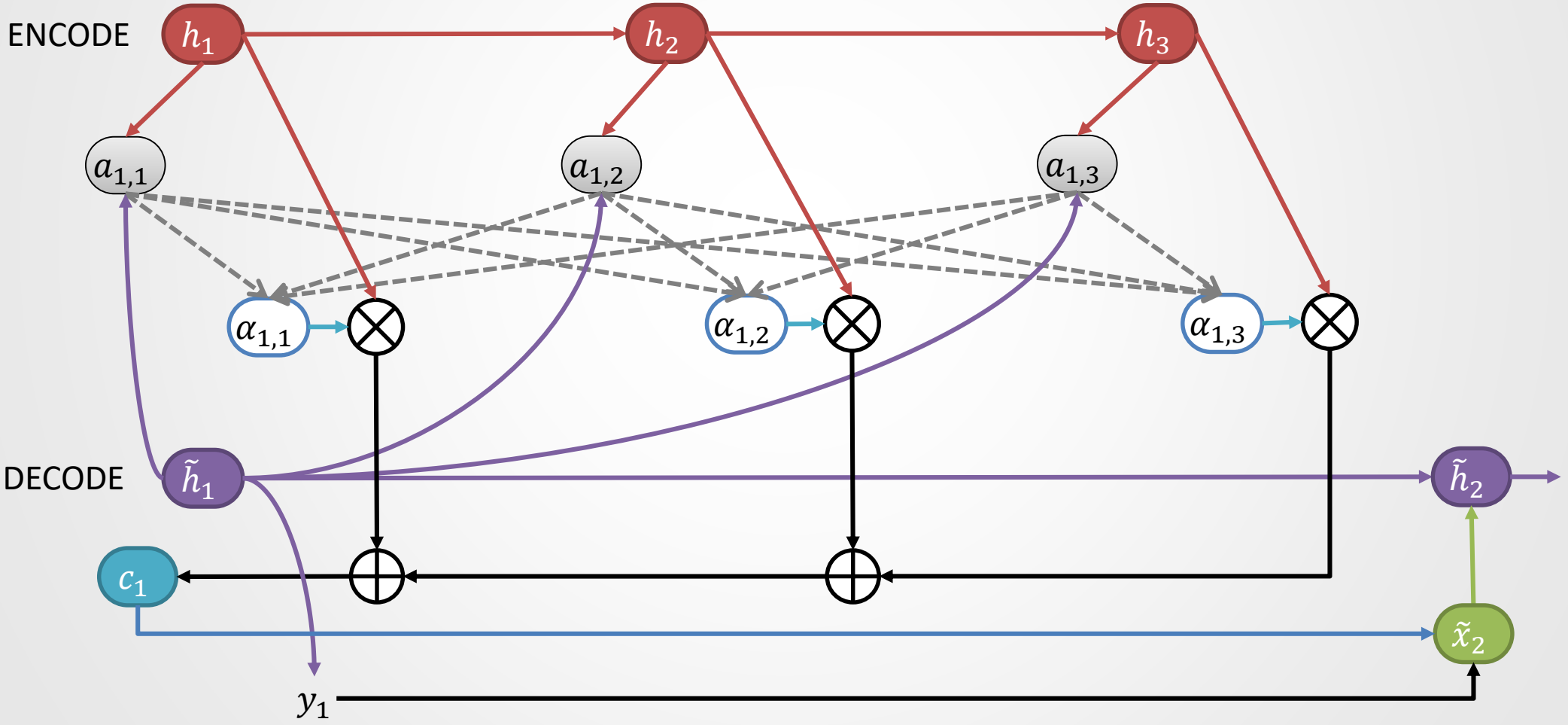
Score function variants

- Attention scores $a_{t,1:S} = score(\tilde{h}_t, h_{1:S})$
- Many variants of the score function for calculating attention scores between decoder's $\tilde{h}_t$ and encoder's $h_{1:S}$
- Basic dot-product attention $a_{t,s} = \tilde{h}_t^T . h_s \in \mathbb{R}$
  - Assumption: $\tilde{h}_{(t)}, h_{(s)} \in \mathbb{R}^d$
- Multiplicative (bilinear) attention $a_{t,s} = \tilde{h}_t^T . \boldsymbol{W} . h_s \in \mathbb{R}$
  - Assumption: $\tilde{h}_{(t)} \in \mathbb{R}^{d_1}, h_{(s)} \in \mathbb{R}^{d_2}$, $W \in \mathbb{R}^{d_1 \times d_2}$ is a weight matrix

Mind Map: the decoder hidden state at time t, $\tilde{h}_t$, is a **query** that attends to all the encoder hidden states, $h_{1:S}$, the **values**!

Attention example

$$a_{t,s} = score(\tilde{h}_t, h_s) \quad \alpha_{t,s} = softmax(a_{t,1:S}, s) \quad c_t = \sum_s \alpha_{t,s} h_s \quad \tilde{x}_t = [c_{t-1}; T_E(y_{t-1})] \in \mathbb{R}^{2d}$$



Multi-headed attention (in *seq2seq*)

Core Idea

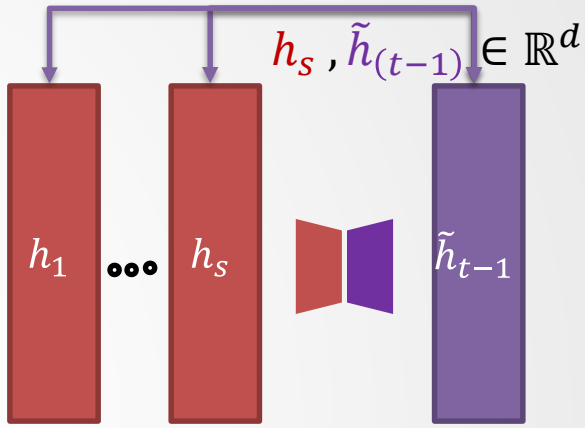
We want to "attend to different things" for a given time step → use **multi-headed attention**

1. Split *N* heads (with $W^{(n)}, \widetilde{W}^{(n)} \in \mathbb{R}^{(d \times \frac{d}{N})}$)

$$\underbrace{\tilde{h}_{t-1}^{(n)}}_{\in \mathbb{R}^{\frac{d}{N}}} = \underbrace{\tilde{h}_{t-1}^{T}}_{\in \mathbb{R}^{d}} \underbrace{\widetilde{W}^{(n)}}_{\in \mathbb{R}^{(d \times \frac{d}{N})}}$$

$$\underbrace{h_{s}^{(n)}}_{\in \mathbb{R}^{\frac{d}{N}}} = \underbrace{h_{s}^{T}}_{\in \mathbb{R}^{d}} \underbrace{W^{(n)}}_{\in \mathbb{R}^{(d \times \frac{d}{N})}}$$

Think of the W, $\widetilde{W}$ as transformation matrices projecting hidden states h, $\tilde{h}$ to a more compact dimension $\in \mathbb{R}^{\frac{d}{N}}$

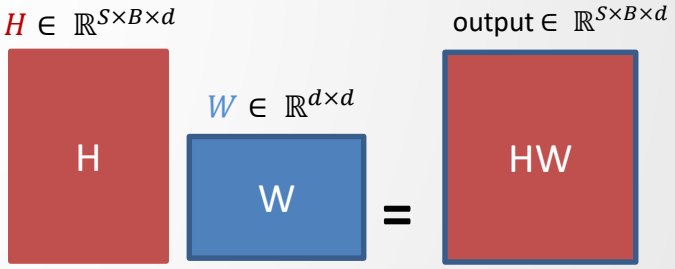


2. Use attention: $c_{t-1}^{(n)} = Att\left(\tilde{h}_{t-1}^{(n)}, h_{1:S}^{(n)}\right)$

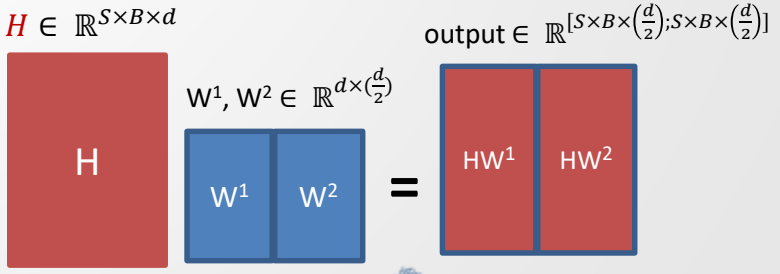
3. Combine for result:

$$\tilde{x}_t = \left[\underbrace{Qc_{t-1}^{(1:N)}}_{\in \mathbb{R}^{?}}; \underbrace{T_E(y_{t-1})}_{\in \mathbb{R}^{d}}\right]$$

here **Q** is a parameter matrix for transforming the concatenated multi-head context vectors $c_{t-1}^{(1:N)}$



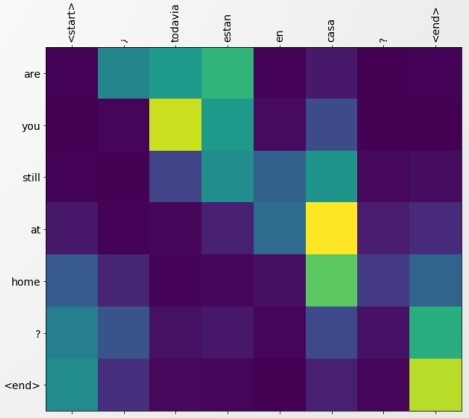
Single-head attention



Multi-head attention (N=2)

Attention advantages

- Improves NMT performance significantly (reply to RNN)
- Appears to solve the bottleneck problem
  - Allows the decoder to look at the source sentence directly, circumventing the bottleneck
- Helps with the long-horizon (vanishing gradient) problem – by providing shortcut to distant states
- Makes the model (somewhat) interpretable
  - We can examine the attention distribution to see what the decoder was focusing on
- We get soft alignment for free
  - Compare w/ the '*word alignment*' matrix from SMT
  - This was also often soft
  - Comes from only sentence-aligned input
  - There had already been a number of unsupervised alignment methods proposed for SMT



A Brief History of Deep Learning for NLP

- Not Deep
- Deep (“no hand-crafted features”)
- Fat
- End-to-End
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep (log-likelihood models, sparse higher-order models)
- Deep (“no hand-crafted features”)
- Fat
- End-to-End
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep
- Deep
- Fat, e.g. RNNs, LSTMs (“horizontally deep”)
- End-to-End
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep
- Deep (deep layers remote from input)
- Fat, e.g. LSTMs (“horizontally deep”)
- End-to-End
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep
- Deep
- Fat
- End-to-End (input/output only – “neural” means it works)
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep (invariably bested by hybrid models)
- Deep
- Fat
- End-to-End (input/output only – “neural” means it works)
- Language Models über alles

A Brief History of Deep Learning for NLP

- Not Deep
- Deep
- Fat
- End-to-End
- Language Models (few-shot learning, prompting)

A Brief History of Deep Learning for NLP

- Not Deep (transformers/EDs use devices that bear a strong resemblance to log-likelihood and sparse higher-order models)
- Deep (“fine tuning”)
- Fat
- End-to-End
- Language Models (few-shot learning, prompting)

Decoding in NMT

Exhaustive search decoding

- Computationally intractable
- Maximize the probability of length T translation E_T

$$P(E|F_S) = (P(e_1|F_S)P(e_2|y_1, F_S), \dots, P(e_T|y_1, y_2 \dots, y_{T-1}, F_S)$$

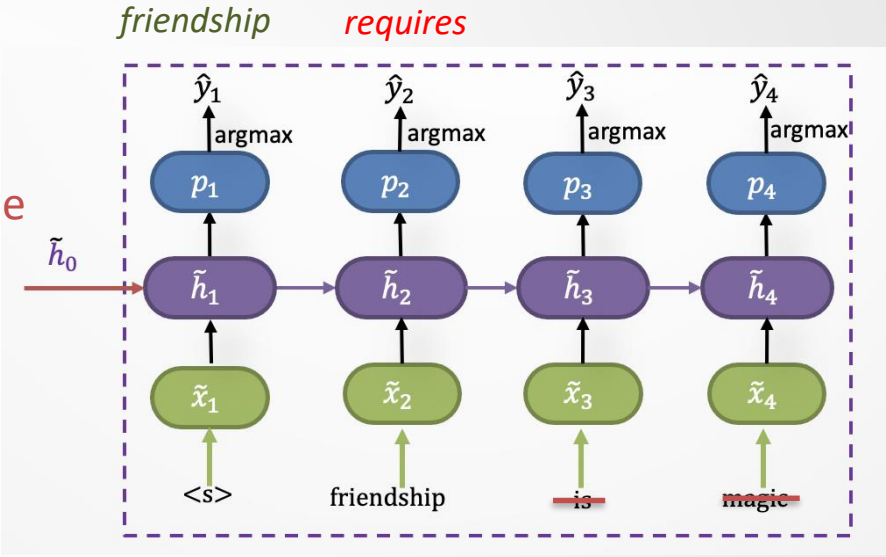
- At each decoder time step *t,* with vocab size *V* :
  - there are *V* possibilities for the decoded token e^t
  - we are tracking V^t possible *partial translations*
- The $O(V^T)$ runtime complexity is infeasible

Greedy Decoding

- Core idea: take the **most probable** word on each step

$$y_t = \text{argmax}_i\left(p_{t,i}\right)$$

Input: L' amitié est magique



- Problem: Can't recover from a prior bad choice (no '*undo*')

- Sub-optimal in an auto-regressive setup:
  - $\tilde{h}_t$ continuous, depends on y_{t-1}
  - Dynamic-programming solution over a discrete, finite state space (e.g. *Viterbi search* - HMM lecture) would have been better

Beam search: top-*K* greedy

- **Core idea**: track the K top choices (most probable) of partial translations (or, hypotheses) at each step of decoding
- K is also called the '*beam width*' or '*beam size*'
  - Where, $5 \leq K \leq 10$ usually in practice
- The score of a hypothesis $(y_1, \dots, y_t)$ is its log probability:

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

  - We search and track the top k hypotheses based on the score
  - Scores are all negative, and higher is better
- Beam search does *not* guarantee finding the optimal solution
- However, much more efficient and practical than exhaustive search

Beam search example (*t=1*)

$V = \{\text{H}, \text{A}, </\text{s}>\}$, K=2

$b_{t,0}^{(k)}$: *k*-th path hidden state
$b_{t,1}^{(k)}$: *k*-th path sequence
$b_t^{(k \to v)}$: *k*-th path extended with token *v*

k	$b_{0,1}^{(k)}$	$P\left(b_0^{(k)}\right)$
1	[<s>]	1
2	[<s>]	0

k	$b_{0,1}^{(k \to v)}$	$P\left(b_0^{(k \to v)}\right)$
1*	[<s>,H]	1x0.1=0.1
1*	[<s>,A]	1x0.9=0.9
1*	[<s>,</s>]	1x0=0
2	[<s>,H]	0x0.1=0
2	[<s>,A]	0x0.9=0
2	[<s>,</s>]	0x0=0

k	$b_{1,1}^{(k)}$	$P\left(b_1^{(k)}\right)$
1	[<s>,A]	0.9
2	[<s>,H]	0.1

*Note $\forall k. \sum_v P\left(b_t^{(k \to v)}\right) = 1$

Beam search example (*t=2*)

$V = \{\mathrm{H}, \mathrm{A}, </\mathrm{s}>\}$, K=2

k	$b_{1,1}^{(k)}$	$P\left(b_1^{(k)}\right)$
1	[<s>,A]	0.9
2	[<s>,H]	0.1

k	$b_{1,1}^{(k\to v)}$	$P\left(b_1^{(k\to v)}\right)$
1	[<s>,A,H]	0.9x0.5=0.45
1	[<s>,A,A]	0.9x0.3=0.27
1	[<s>,A,</s>]	0.9x0.2=0.18
2	[<s>,H,H]	0.1x0.9=0.09
2	[<s>,H,A]	0.1x0.0=0
2	[<s>,H,</s>]	0.1x0.1=0.01

k	$b_{2,1}^{(k)}$	$P\left(b_2^{(k)}\right)$
1	[<s>,A,H]	0.45
2	[<s>,A,A]	0.27

Problem 1: concentrated mass on a prefix creates near identical hypotheses

Beam search example (*t=3*)

$V = \{\text{H}, \text{A}, </\text{s}>\}$, K=2

k	$b_{2,1}^{(k)}$	$P\left(b_2^{(k)}\right)$
1	[<s>,A,H]	0.45
2	[<s>,A,A]	0.27

k	$b_{2,1}^{(k \to v)}$	$P\left(b_2^{(k \to v)}\right)$
1	[<s>,A,H,H]	0.45x0.5=**0.225**
1	[<s>,A,H,A]	0.45x0.3=0.135
1	[<s>,A,H,</s>]	0.45x0.2=0.09
2	[<s>,A,A,H]	0.27x0.2=0.054
2	[<s>,A,A,A]	0.27x0.2=0.054
2	[<s>,A,A,</s>]	0.27x0.6=**0.162**

EOS token generated

k	$b_{3,1}^{(k)}$	$P\left(b_3^{(k)}\right)$
1	[<s>,A,H,H]	0.225
2	[<s>,A,A,</s>]	0.162

A complete hypothesis

Beam search: stopping criterion

- **Continue** decoding greedily **until** the model produces an end of sequence (</s>) token
- But '</s>' can be produced at different timesteps for each candidate hypothesis
  - Mark a hypothesis as **complete** when </s> is produced
  - The probability of a completed hypothesis **does not decrease**
  - Place it aside and continue exploring other hypotheses paths
- Usually we continue beam search until:
  - A pre-defined cutoff timestep *T* is reached
  - A pre-defined cutoff of completed hypotheses *n* has been reached

Beam search example (*t=4*)

$V = \{\text{H}, \text{A}, </\text{s}>\}$, K=2

k	$b_{3,1}^{(k)}$	$P\left(b_3^{(k)}\right)$
1	[<s>,A,H,H]	0.225
2	[<s>,A,A,</s>]	0.162

k	$b_{3,1}^{(k\to v)}$	$P\left(b_3^{(k\to v)}\right)$
1	[<s>,A,H,H,H]	0.225x0.9=**0.214**
1	[<s>,A,H,H,A]	0.225x0.05=0.01
1	[<s>,A,H,H,</s>]	0.18x0=0
2*	[<s>,A,A,</s>]	0.162x0=0
2*	[<s>,A,A,</s>]	0.162x0=0
2*	[<s>,A,A,</s>]	0.162x1=0.162

k	$b_{4,1}^{(k)}$	$P\left(b_4^{(k)}\right)$
1	[<s>,A,H,H,H]	0.214
2	[<s>,A,A,</s>]	0.162

*Since k=2 is finished

Beam search example (*t=5*)

$V = \{\text{H}, \text{A}, </\text{s}>\}$, K=2

k	$b_{4,1}^{(k)}$	$P\left(b_4^{(k)}\right)$
1	[<s>,A,H,H,H]	0.214
2	[<s>,A,A,</s>]	0.162

k	$b_{4,1}^{(k \to v)}$	$P\left(b_4^{(k \to v)}\right)$
1	[<s>,A,H.H,H,H]	0.214x0.7=0.150
1	[<s>,A,H,H,H,A]	0.214x0.3=0.064
1	[<s>,A,H,H,H,</s>]	0.171x0=0
2	[<s>,A,A,</s>]	0.162x0=0
2	[<s>,A,A,</s>]	0.162x0=0
2	[<s>,A,A,</s>]	0.162x1=0.162

Winner!

	$b_{5,1}^{(\kappa)}$	$P\left(b_5^{(\kappa)}\right)$
1	[<s>,A,A,</s>]	0.162
2	[<s>,A,H,H,H,H]	0.150

Problem 2: finished path probability doesn't decrease → preference for shorter paths

Solution: Normalize hypotheses score by length (1/t)

Beam search: top-*K* greedy

$b_{t,0}^{(k)}$: *k*-th path hidden state
$b_{t,1}^{(k)}$: *k*-th path sequence
$b_t^{(k\to v)}$: *k*-th path extended with token *v*

Given vocab V, decoder σ, beam width K

$\forall k \in [1,K]. b_{0,0}^{(k)} \leftarrow \tilde{h}_0, \mathrm{b}_{0,1}^{(k)} \leftarrow [<\mathrm{s}>], \log \mathrm{P}\left(b_0^{(k)}\right) \leftarrow -\mathbb{I}_{k\neq 1}\infty$

$f \leftarrow \emptyset$ # finished path indices

While $1 \notin f$:

$\forall k \in [1,K]. \tilde{h}_{t+1}^{(k)} \leftarrow \sigma\left(b_{t,0}^{(k)}, last\left(b_{t,1}^{(k)}\right)\right)$ # $last(x)$ gets last token in x

$\forall v \in V, k \in [1,K] \backslash f. b_{t,0}^{(k\to v)} \leftarrow \tilde{h}_{t+1}^{(k)}, b_{t,1}^{(k\to v)} \leftarrow \left[b_{t,1}^{(k)}, v\right]$

Calculate hypothesis score $\log P\left(b_t^{(k\to v)}\right) \leftarrow \log P(y_{t+1} = v|\tilde{h}_{t+1}^{(k)}) + \log P\left(b_t^{(k)}\right)$

$\forall v \in V, k \in f. b_t^{(k\to v)} \leftarrow b_t^{(k)}, \log P\left(b_t^{(k\to v)}\right) \leftarrow \log P\left(b_t^{(k)}\right) - \mathbb{I}_{v\neq <\mathrm{/s}>}\infty$

$\forall k \in [1,K]. b_{t+1}^{(k)} \leftarrow \mathrm{argmax}^k_{b_t^{(k'\to v)}} \log P\left(b_t^{(k'\to v)}\right)$ # k-th max $b_t^{(k'\to v)}$

$f \leftarrow \{k \in [1,K] | last\left(b_{t+1}^{(k)}\right) = <\mathrm{/s}>\}$

$t \leftarrow t+1$

Return $b_{t,1}^{(1)}$

*Other completion criteria exist (e.g. $t \leq T$, finish some # of paths)

Beam search: top-*K* greedy

In lecture annotations

$b_{t,0}^{(k)}$: *k*-th path hidden state
$b_{t,1}^{(k)}$: *k*-th path sequence
$b_t^{(k\to v)}$: *k*-th path extended with token *v*

Initialization

Given vocab V, decoder σ, beam width K

$\forall k \in [1,K]. b_{0,0}^{(k)} \leftarrow \tilde{h}_0, b_{0,1}^{(k)} \leftarrow [\text{<s>}], \log P\left(b_0^{(k)}\right) \leftarrow -\mathbb{I}_{k\neq 1}\infty$

$f \leftarrow \emptyset$ # finished path indices

While $1 \notin f$: End search when the most probable of the K prefixes end with </s>

$\forall k \in [1,K]. \tilde{h}_{t+1}^{(k)} \leftarrow \sigma\left(b_{t,0}^{(k)}, last\left(b_{t,1}^{(k)}\right)\right)$ # $last(x)$ gets last token in x

$\forall v \in V, k \in [1,K]\backslash f. b_{t,0}^{(k\to v)} \leftarrow \tilde{h}_{t+1}^{(k)}, b_{t,1}^{(k\to v)} \leftarrow \left[b_{t,1}^{(k)}, v\right]$

K paths excluding the finished ones

Calculate hypothesis score $\log P\left(b_t^{(k\to v)}\right) \leftarrow \log P(y_{t+1} = v|\tilde{h}_{t+1}^{(k)}) + \log P\left(b_t^{(k)}\right)$

$\forall v \in V, k \in f. b_t^{(k\to v)} \leftarrow b_t^{(k)}, \log P\left(b_t^{(k\to v)}\right) \leftarrow \log P\left(b_t^{(k)}\right) - \mathbb{I}_{v\neq\text{</s>}}\infty$

Pick top-K (sorted) $\forall k \in [1,K]. b_{t+1}^{(k)} \leftarrow \text{argmax}^k_{b_t^{(k'\to v)}} \log P\left(b_t^{(k'\to v)}\right)$ # *k-th max* $b_t^{(k'\to v)}$

$f \leftarrow \{k \in [1,K] | last\left(b_{t+1}^{(k)}\right) = \text{</s>}\}$ Write as finished path if </s> generated

$t \leftarrow t+1$ Go to next time-step

Return $b_{t,1}^{(1)}$ Return the most probable (index 1) finished path sequence

Sub-word Tokenization

- Out-of-vocabulary (OOV) words can be handled by breaking up words into parts
  - "Abwasser+behandlungs+anlage" → "water sewage plant"
    ["incorporation" (German)]
- Sub-word units are built out of combining characters (like phrases?)
- Popular (sub-word tokenization) approaches include
  - Byte Pair Encoding (BPE): "Neural machine translation of rare words with subword units," 2016. Sennrich *et al.* Used in GPT-2, BERT-based PLMs
  - Wordpieces: "Google's neural machine translation system: bridging the gap between human and machine translation," 2016. Wu *et al.*

Aside – advanced NMT

- Modifications to beam search
  - "Diverse beam search," 2018. Vijayakumar *et al.*
- Exposure bias
  - "Optimal completion distillation," 2018. Sabour *et al.*
- Back translation
  - "Improving neural machine translation models with monolingual data," 2016. Senrich *et al*.
- **N**on-**a**utoregressive neural machine **t**ranslation, 2018. Gu *et al*.
- Unsupervised neural machine translation, 2018. Artetxe *et al*.
- + *Optional readings* listed on course webpage

Automatic evaluation

- We want an **automatic** and effective method to **objectively** rank competing translations.
  - **Word Error Rate (WER)** measures the number of erroneous word **insertions**, **deletions**, **substitutions** in a translation.
    - E.g., **Reference**: *how to recognize speech*
      Translation: *how understand a speech*
    - Works for Automatic Speech Recognition (ASR)
  - **Problem**: There are many possible valid translations.
    (There's no need for an exact match)

Challenges of human evaluation

- **Human judges**: expensive, slow, non-reproducible (different judges – different biases).

- Multiple valid translations, e.g.:
  - **Source**: *Il s'agit d'un guide qui assure que l'armée sera toujours fidèle au Parti*
  - **T1**: *It is a guide to action that ensures that the military will forever heed Party commands*
  - **T2**: *It is the guiding principle which guarantees the military forces always being under command of the Party*

BLEU evaluation

- **BLEU (BiLingual Evaluation Understudy)** is an automatic and popular method for evaluating MT.
  - It uses **multiple** human **reference** translations, and looks for local matches, allowing for phrase movement.
  - **Candidate:** *n.* a translation produced by a machine.
- There are a few parts to a **BLEU score**…

[1]Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th ACL*. 2002. [link]

Example of BLEU evaluation

- **Reference 1**: *It is a guide to action that ensures that the military will forever heed Party commands*
- **Reference 2**: *It is the guiding principle which guarantees the military forces always being under command of the Party*
- **Reference 3**: *It is the practical guide for the army always to heed the directions of the party*

- **Candidate 1**: *It is a guide to action which ensures that the military always obeys the commands of the party*
- **Candidate 2**: *It is to insure the troops forever hearing the activity guidebook that party direct*

BLEU: Unigram precision

- The **unigram precision** of a candidate is

$$\frac{C}{N}$$

where N is the number of words in the **candidate**
and C is the number of words in the **candidate** which are in **at least one reference**.

- e.g., **Candidate 1**: *It is a guide to action which ensures that the military always obeys the commands of the party*
  - **Unigram precision** $= \frac{17}{18}$

    (*obeys* appears in none of the three references).

BLEU: Modified unigram precision

- **Reference 1**: *The lunatic is on the grass*
- **Reference 2**: *There is a lunatic upon the grass*
- **Candidate**: *The the the the the the the*
  - Unigram precision $= \frac{7}{7} = 1$ 

- **Capped unigram precision:**

  A candidate word type w can only be correct a **maximum** of $cap(w)$ times.
  - e.g., with $\boldsymbol{cap(the) = 2}$, the above gives

    $$p_1 = \frac{2}{7}$$

BLEU: Generalizing to *N*-grams

- Generalizes to higher-order *N*-grams.
  - **Reference 1**: ***It is*** *a guide to action that ensures that the military will forever heed Party commands*
  - **Reference 2**: ***It is*** *the guiding principle which guarantees the military forces always being under command of the Party*
  - **Reference 3**: ***It is*** *the practical guide for the army always to heed the directions of the party*

Bigram precision, p_2

- **Candidate 1**: ***It is*** *a guide to action which ensures that the military always obeys the commands of the party*

$p_2 = 10/17$

- **Candidate 2**: ***It is*** *to insure the troops forever hearing the activity guidebook that party direct*

$p_2 = 1/13$

BLEU: Precision is not enough

- **Reference 1**: *It is a guide to action that ensures that the military will forever heed Party commands*
- **Reference 2**: *It is the guiding principle which guarantees the military forces always being under command **of the** Party*
- **Reference 3**: *It is the practical guide for the army always to heed the directions **of the** party*

- **Candidate 1**: ***of the***

Unigram precision, $p_1 = \frac{2}{2} = 1$ Bigram precision, $p_2 = \frac{1}{1} = 1$

BLEU: Brevity

- Solution: Penalize brevity.
- **Step 1**: for each candidate,
  find the reference **most similar in length**.
- **Step 2**: $\boldsymbol{c_i}$ is the length of the i^{th} candidate, and
  $\boldsymbol{r_i}$ is the nearest length among the references,

$$brevity_i = \frac{r_i}{c_i}$$

Bigger = too brief

- **Step 3**: multiply precision by the (0..1) **brevity penalty**:

$$BP_i = \begin{cases} 1 & \text{if } brevity_i < 1 \\ e^{1-brevity_i} & \text{if } brevity_i \geq 1 \end{cases}$$

$(r_i < c_i)$

$(r_i \geq c_i)$

BLEU: Final score

- On slide 96, $r_1 = 16, r_2 = 17, r_3 = 16$, and $c_1 = 18$ and $c_2 = 14$,

$$brevity_1 = \frac{17}{18} \qquad BP_1 = 1$$

$$brevity_2 = \frac{16}{14} \qquad BP_2 = e^{1-\left(\frac{8}{7}\right)} = 0.8669$$

- **Final score** of candidate C:

$$BLEU_C = BP_C \times (p_1 p_2 \dots p_n)^{1/n}$$

where p_n is the n-gram precision. (You can set n empirically)

Example: Final BLEU score

- **Reference 1:** *I am afraid Dave*
  Reference 2: *I am scared Dave*
  Reference 3: *I have fear David*
  Candidate: *I fear David*

Assume $cap(\cdot) = 2$ for all *N*-grams

- $brevity = \frac{4}{3} \geq 1$ so $BP = e^{1-\left(\frac{4}{3}\right)}$

Also assume BLEU order $n = 2$

- $p_1 = \frac{1+1+1}{3} = 1$
- $p_2 = \frac{1}{2}$
- $BLEU = BP(p_1 p_2)^{\frac{1}{2}} = e^{1-\left(\frac{4}{3}\right)} \left(\frac{1}{2}\right)^{\frac{1}{2}} \approx 0.5067$

Aside – Corpus-level BLEU

- To calculate BLEU over M source sentences (assuming one candidate per source)…
- $BLEU \neq \frac{1}{M}\sum_{m=1}^{M} BLEU_m$
- Sum statistics over *all* sources
  - m indexes m-th source sentence, dropping candidate index i $(ablative)$
  - $p^i_n = \frac{\sum_{m=1,m\#i}^{M} capped\_true\_ngram\_count_m}{\sum_{m=1,m\#i}^{M} N_m}$
  - $r = \sum_{m=1}^{M} r_m$
  - $c = \sum_{m=1}^{M} c_m$
  - $brevity = r/c$
- **We won't ask you to calculate it this way**

BLEU: summary

- BLEU is a **geometric mean** over n-gram precisions.
  - These precisions are **capped** to avoid strange cases.
    - E.g., the translation "*the the the the*" is not favoured.
  - This geometric mean is **weighted** (*brevity penalty*) so as not to favour unrealistically short translations, e.g., "*the*"
- Initially, evaluations showed that BLEU predicted human judgements very well, but:
  - People started **optimizing** MT systems to **maximize** BLEU. Correlations between BLEU and humans **decreased**.

BLEU is not *construct valid*.

NMT - Advantages

NMT has many advantages over SMT:

- Overall better performance
- Simpler design (though still very large):
  - A single neural network can be trained end-to-end (but it's probably a mistake to do so)
  - Where there are components, you can jointly optimize/train
- Significantly less effort necessary in some respects:
  - Same method for all language pairs
  - No feature engineering for specific requirements

NMT - Disadvantages

NMT has disadvantages compared to SMT:

- Less interpretable
- Harder to debug (though SMT wasn't easy)
- Significantly fewer opportunities for a little more effort:
  - Can't specify rules or guidelines for translation
  - More prone to various biases

NMT – Research questions

- Morphological errors
- Biases in training data
- Low-resource languages
- Common-sense translations
- Contextual, multi-modally grounded reasoning
  - Instruction following by AI agents (EAI agents, robots) using non-expert language feedback
- Generalization to multiple domains