Neural machine translation

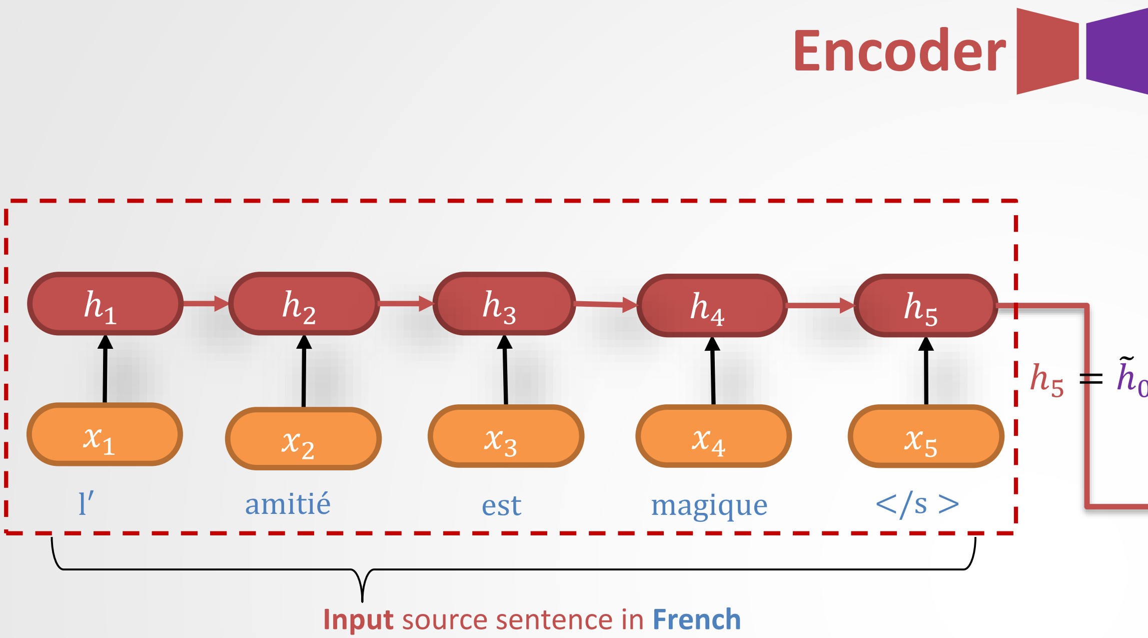
CSC401/2511 – Natural Language Computing
Spring 2026

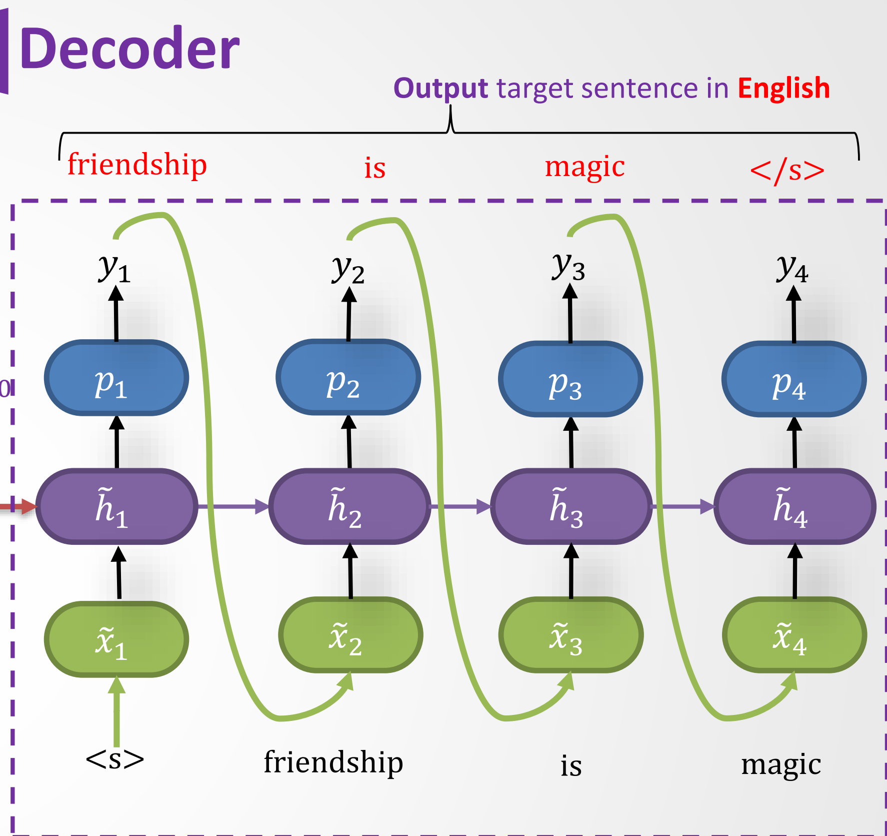
Ken Shi and Gerald Penn

Lecture 8

University of Toronto

NMT: the seq2seq model





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

- NMT directly calculates $y^* = \text{argmax}_y P(y|x)$
- I.e. with our formulation:

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

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

Decoder is predicting the next word of the target sentence y

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

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

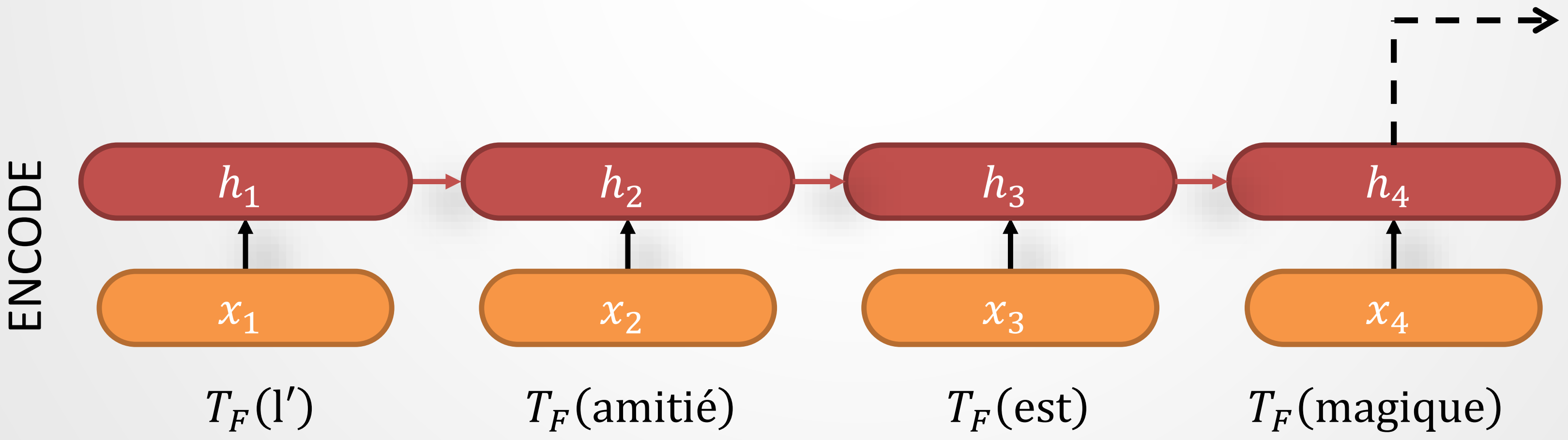
UNIVERSITY OF TORONTO

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 $p_t = f\left(\tilde{h}_t\right)$
$y_{1:T}$	Sampled output token from decoder $y_t \sim P(y_t \| p_t)$
$c_{1:T}$	Attention context $c_t = Attend\left(\tilde{h}_t, h_{1:S}\right) = \sum_s \alpha_{t,s} h_s$
$e_{1:T,1:S}$	Score function output $e_{t,s} = score\left(\tilde{h}_t, h_s\right)$
$\alpha_{1:T,1:S}$	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 $x = (x_1, x_2, \ldots)$
  - $x_s = T_F(F_s)$ a source word embedding
- Outputs last hidden state of RNN
- Note $h_S = f(F_{1:S})$ conditions on entire source



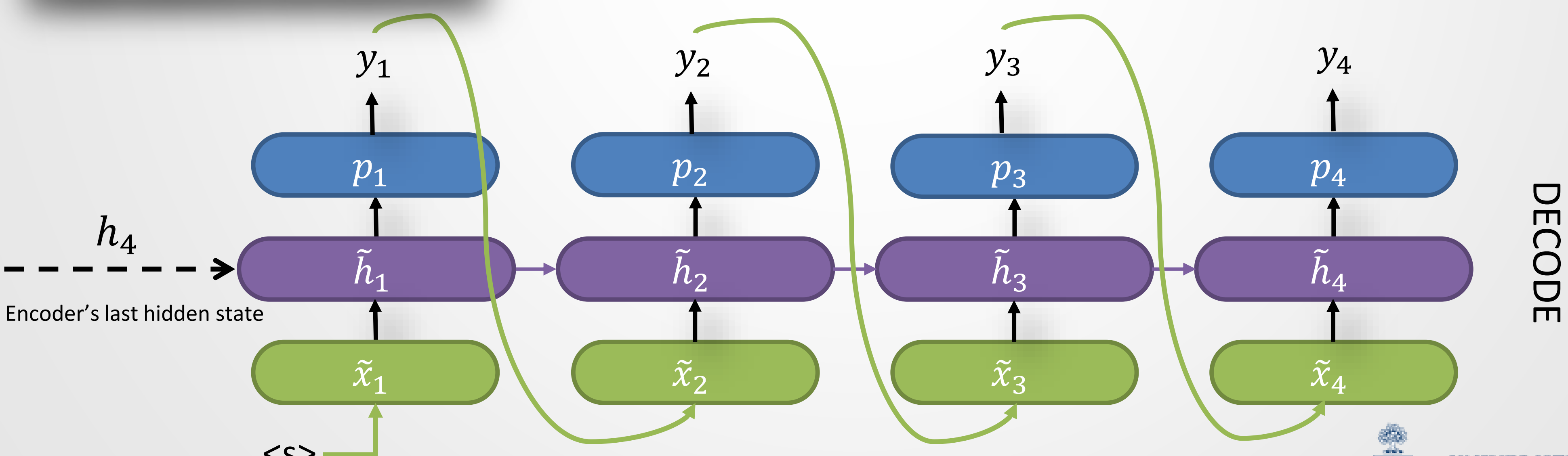
Source sentence (French): *L' amitié est magique*

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** $\tilde{x}_t = T_E(\boldsymbol{y_{t-1}})$
- $p_t = f(\tilde{h}_t) = f\left(g(\tilde{x}_t, \tilde{h}_{t-1})\right)$ is **deterministic**
- Base case: $\tilde{x}_1 = T_E(\text{<s>}),\ \tilde{h}_0 = h_S$
- $P(y_{1:T}|F_{1:S}) = \prod_t P(y_t|y_{<t}, F_{1:S}) \rightarrow$ **auto-regressive**

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



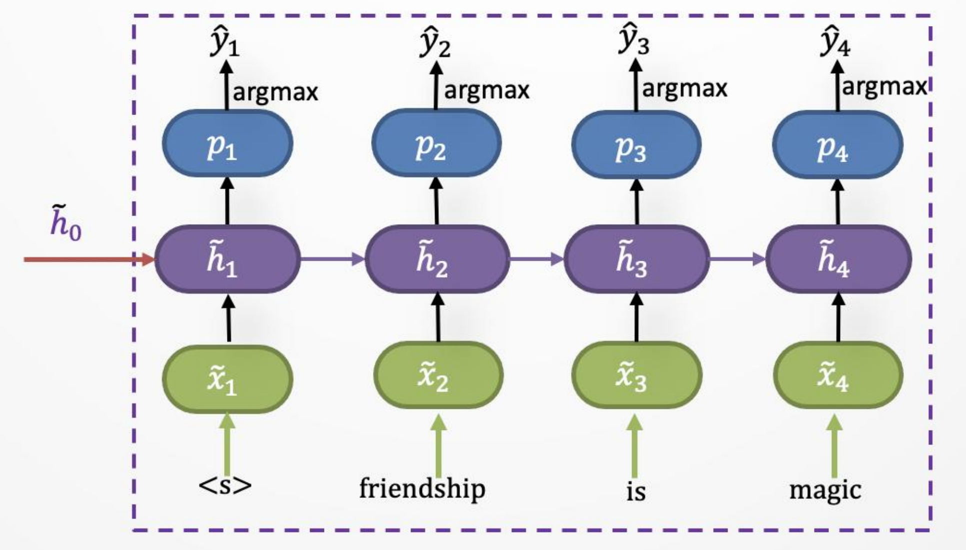
NMT: Training an MT system

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

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

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

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



UNIVERSITY OF TORONTO

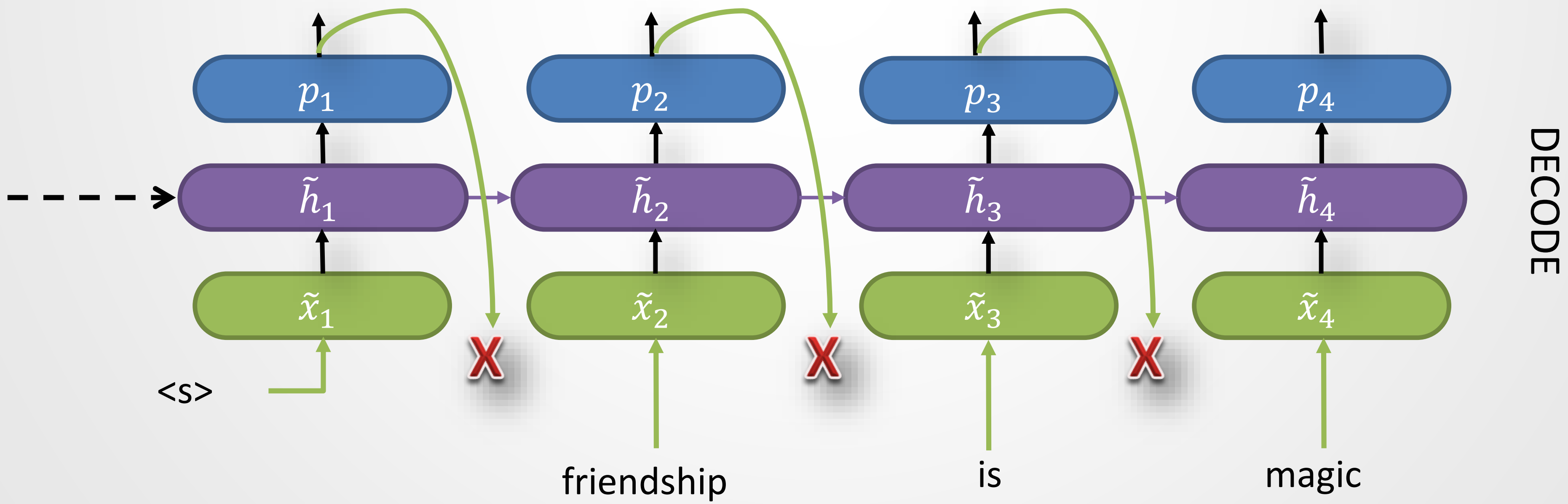
Teacher forcing

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 $\tilde{x}_t = T(\underbrace{y_{t-1}}_{\text{Predicted output}})$ with $\tilde{x}_t = T(\underbrace{E_{t-1}}_{\text{target or ground truth}})$
- Caveat: since $y_{t-1} \neq E_{t-1}$ in general, causes **exposure bias**

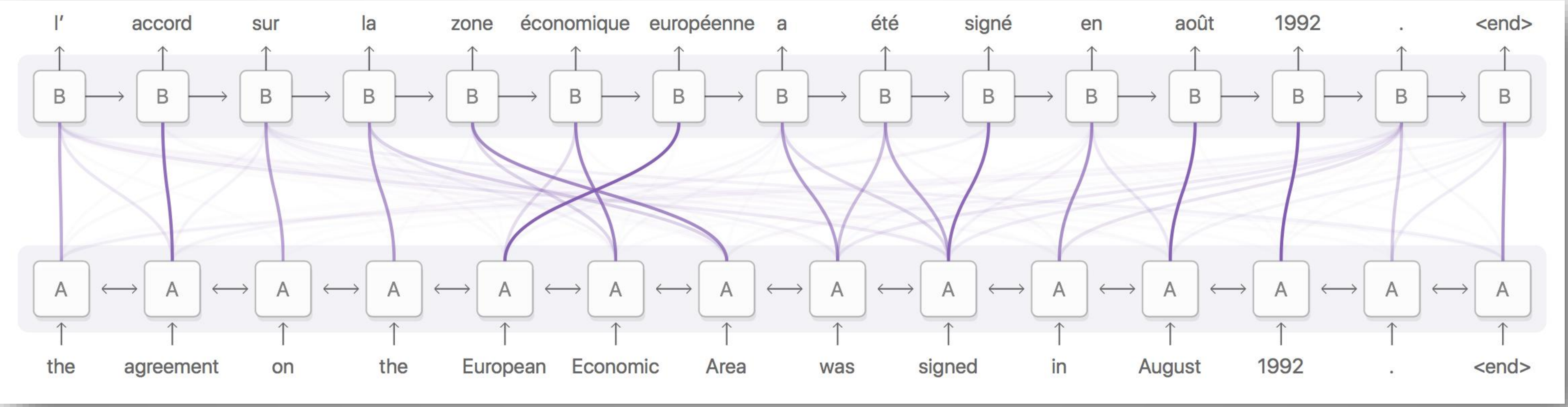
$$\mathcal{L} = \quad -\log P(\text{friendship}|\cdots) - \log P(\text{is}|\cdots) \; - \log P(\text{magic}|\cdots) - \log P(</s>|\cdots)$$



UNIVERSITY OF TORONTO

Attention mechanisms

- 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).

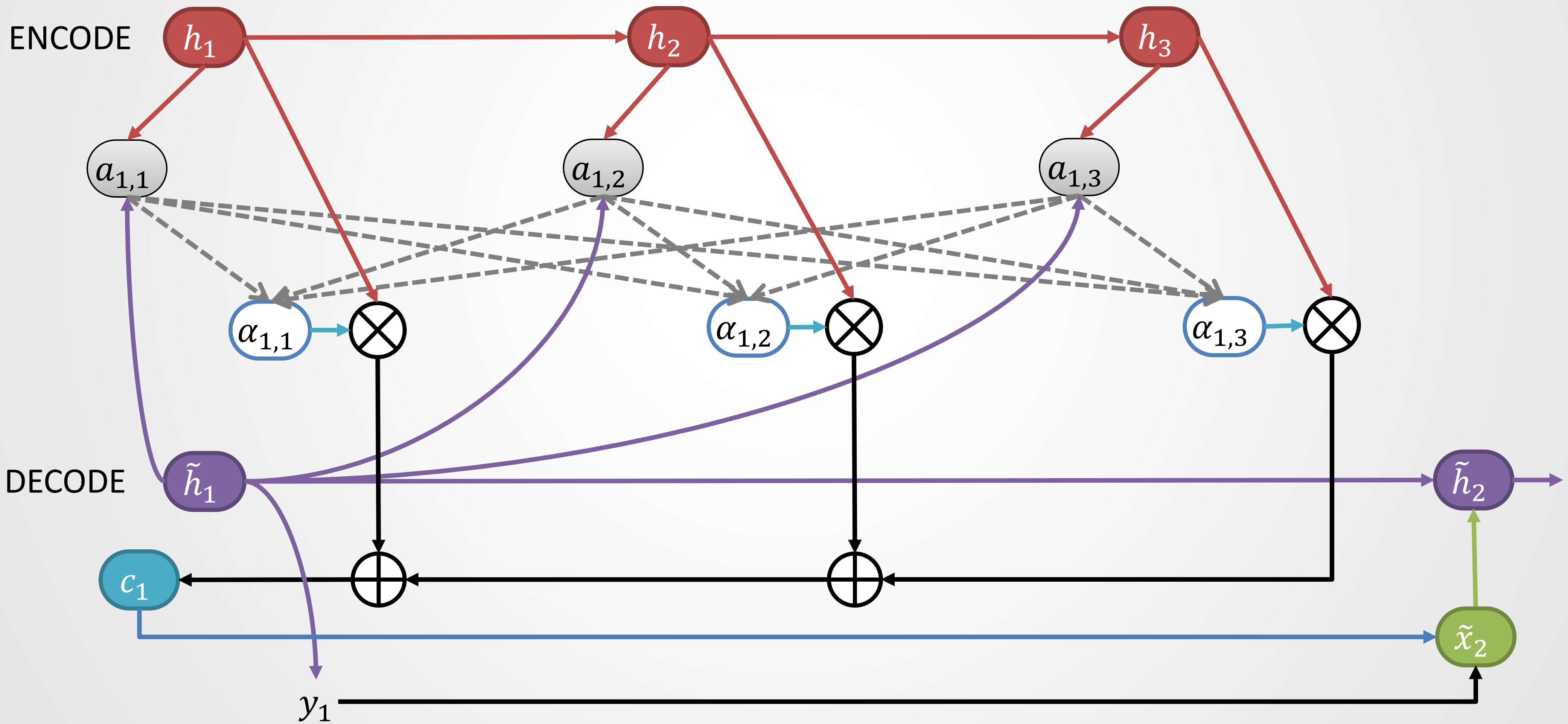
Score function variants

- 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

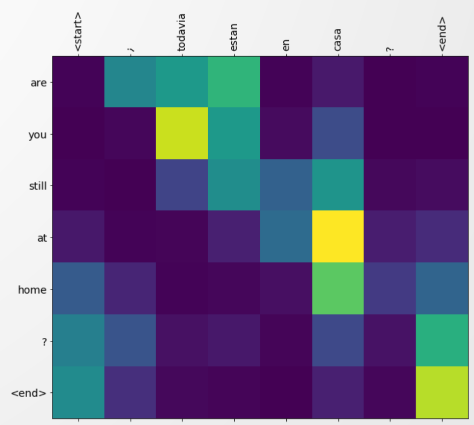
$a_{t,s} = score(\tilde{h}_t, h_s)$ $\alpha_{t,s} = softmax(a_{t,1:S}, s)$ $c_t = \sum_s \alpha_{t,s} h_s$ $\tilde{x}_t = [c_{t-1}; T_E(y_{t-1})] \in \mathbb{R}^{2d}$



UNIVERSITY OF TORONTO

Attention advantages for NMT

- 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



UNIVERSITY OF TORONTO

Runtime complexity

- Assume $S \approx T$

Model	Complexity	Reason
Without attention	$\boldsymbol{O}(\boldsymbol{T})$	Encoder, then decoder
With attention	$O(T^2)$	Decoder attends to all encoder states
Transformer	$O(T^2)$	Everyone attends to everyone else

- Parallelization caveats:
  - Quick to train, slow during decoding
  - Auto-regressive stacked RNN much slower than non-auto-regressive stacked RNNs
  - More details in CSC 413/2516

Decoding in NMT

Exhaustive search decoding

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

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

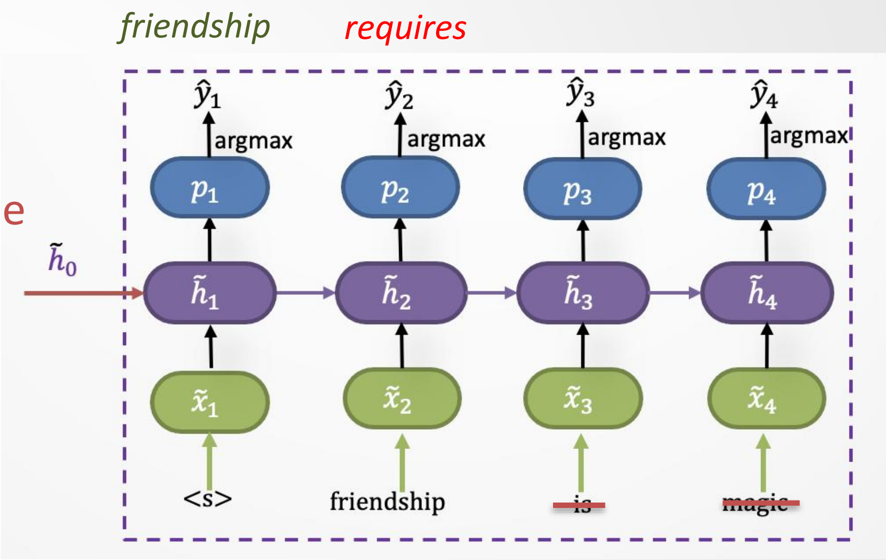
- At each decoder time step *t,* with vocab size *V* :
  - there are *V* possibilities for the decoded token e^t
  - we are tracking V^t possible *partial translations*
- The $O(V^T)$ runtime complexity is infeasible

Greedy Decoding

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

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

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

UNIVERSITY OF TORONTO

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, $5 \le K \le 10$ 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*)

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

$V = \{\text{H}, \text{A}, </s>\}$, 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*)

$V = \{\mathrm{H}, \mathrm{A}, </\mathrm{s}>\}$, 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 <u>different timesteps</u> for each candidate hypothesis
  - Mark a hypothesis as **complete** <u>when </s> is produced</u>
  - 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*)

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

UNIVERSITY OF TORONTO

Beam search example (*t=5*)

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

k	$b_{5,1}^{(k)}$	$P\left(b_5^{(k)}\right)$
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

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

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

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

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

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

$t \leftarrow t+1$

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

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

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

UNIVERSITY OF TORONTO

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

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

- **<u>Candidate 1</u>**: *It is a guide to action which ensures that the military always obeys the commands of the party*
- **<u>Candidate 2</u>**: *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*

  - **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*

Bigram precision, p_2

$p_2 = 10/17$

$p_2 = 1/13$

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**:

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

($r_i < c_i$)

($r_i \geq c_i$)

UNIVERSITY OF TORONTO

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:

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

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

UNIVERSITY OF TORONTO

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

Also assume BLEU order $n = 2$

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

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*.

  - BLEU does not allow for sufficient opportunities to **celebrate** the high degree of fluency in NMT translations.

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