Natural Language Processing
Machine translation

Sofia Serrano
sofias6@cs.washington.edu

Credit to Noah Smith for slides

Announcements

- A2 is due today at 11:59pm
  - Careful with the number of late days you have left!
    - If you used three late days on A1, you have two late days left for the quarter, so the latest you can turn in A2 (using both remaining late days) is **Sunday 2/19 at 11:59pm**.
    - Otherwise, the latest you can turn in A2 (using three late days) is **Monday 2/20 at 11:59pm.**
  - Make sure to commit/push your **.preds files** by the deadline, in addition to your code and writeup
  - **Don't forget to push your a2-final tags!** Check gitlab if in doubt that it went through.
- No lecture or office hours on Monday (Presidents' Day)
- Quiz 6 goes out on Canvas on Wednesday 2/22 at the end of lecture
  - Available until Friday 2/24 at 2:20pm; you'll have 15 minutes to complete it once you start.
  - Remember that you can use your notes during the quiz
  - Will cover material from Wednesday's lecture and today's lecture (so, transformers and machine translation)

Finishing up transformers

ELMo (Peters et al. 2018)

The big takeaway:

if you train a **language model**,

then just replace the model's output layer and use the parameters from the original language model to adjust your word embeddings in the model's first few layers,

those new word embeddings can help you perform **really** well on a whole range of NLP tasks, especially if you **finetune** the parameters (i.e., train them to perform your actual task of interest).

The BERT training objective (Devlin et al. 2018)

Very similar idea to ELMo, but

- used the transformer architecture (unlike ELMo)
- used **masked language modeling** as its pretraining objective instead

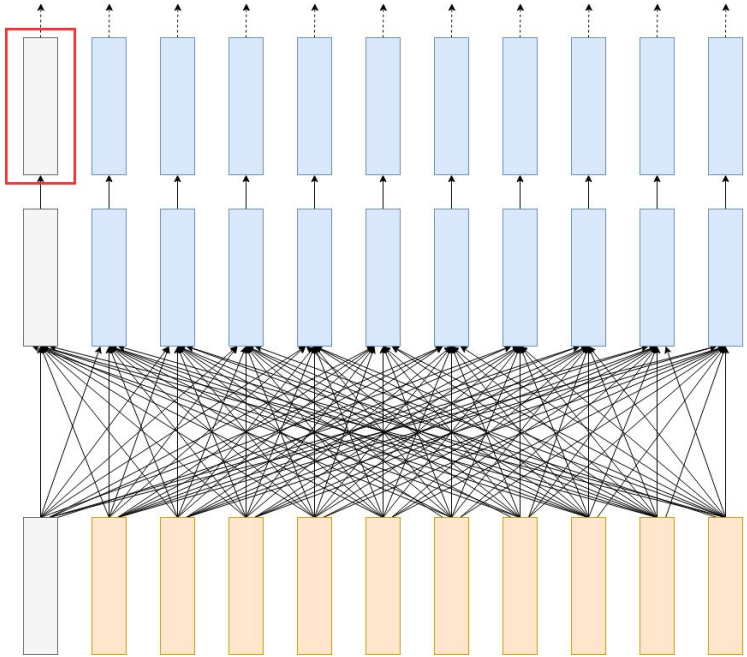
The quick brown _____
The quick brown fox _____

The _____ brown fox jumps _____ the...
The quick _____ _____ jumps over the...

Side note: the reason for this little red box

Want a good representation of a sentence?

It's common to use BERT (or RoBERTa or something) to encode the sentence, and then just take the first representation (corresponding to the special [CLS] token) from the final layer in the transformer as a representation of the sentence as a whole.



Lots of different strategies proposed for sampling text from powerful language models

For example, Nucleus Sampling (which we won't talk about today due to time).

The way in which you sample from a language model's output probability distribution can have a big effect on the kind of text you get!

We'll briefly go over the top-K sampling algorithm as an example.

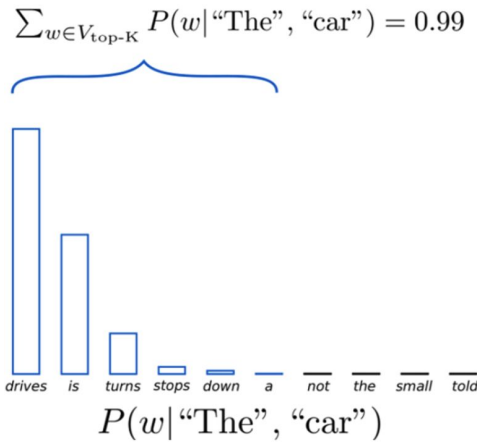
The top-K sampling algorithm

We will represent $P(\cdot\,|W_{1..i})$ by $p = (p_1, p_2, \ldots, p_{|V|})$, where the elements is sorted that $p_1 \geq p_2 \geq p_3 \ldots \geq p_{|V|}$.

Top-K sampling transforms p to $\hat{p}$ by:

$$\widehat{p_i} = \frac{p_i \cdot 1\{i \leq K\}}{Z}$$

And we sample W_{i+1} from $\hat{p}$.



← from
https://huggingface.co/blog/how-to-generate
(Optional reading)

Slide by Tianxing He

Examples from the GPT2 model

- Prompt: *MIT is a private research university in Cambridge, Massachusetts. It is one of the best universities in the U.S.*,

- GPT2 with naive sampling: but the teaching of traditional African-American studies and African-American literacy continued. Soon thereafter, MIT was renamed The International Comparative University by Lord (then), …
- GPT2 with topk40 sampling: and the home of most of the top international universities in the world. Our alumni are internationally renown, but our mission is unique. We are the only university in the world where there is a chance to take on the challenge of making an impact, …
- topk40 another sample: with a reputation for innovation and open and flexible public systems. Its principal research area deals with autonomous vehicles, robotics and artificial intelligence. To date, MIT has published 40 peer-reviewed papers on this topic, …

- Message: sampling algorithms provide a sweet quality-diversity trade-off.
- (which is the key difference to decoding e.g., beam-search)
- Tianxing did not cherry-pick these examples!

Slide by Tianxing He

References (optional reading)

On recency bias in LSTMs, and comparing LSTMs to the transformer:

- Michał Daniluk et al., “Frustratingly Short Attention Spans in Neural Language Modeling.” 2017. https://arxiv.org/abs/1702.04521.
- Jared Kaplan et al., “Scaling Laws for Neural Language Models.” 2020. https://arxiv.org/abs/2001.08361.
- Urvashi Khandelwal et al., “Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context.” 2018. https://arxiv.org/abs/1805.04623.

Sinno Jialin Pan and Qiang Yang, “A Survey on Transfer Learning.” 2010. 10.1109/TKDE.2009.191.

Noah A. Smith, “Contextual Word Representations: A Contextual Introduction.” 2019. https://arxiv.org/abs/1902.06006.

Machine translation

What is machine translation?

Given a sentence in source language S,

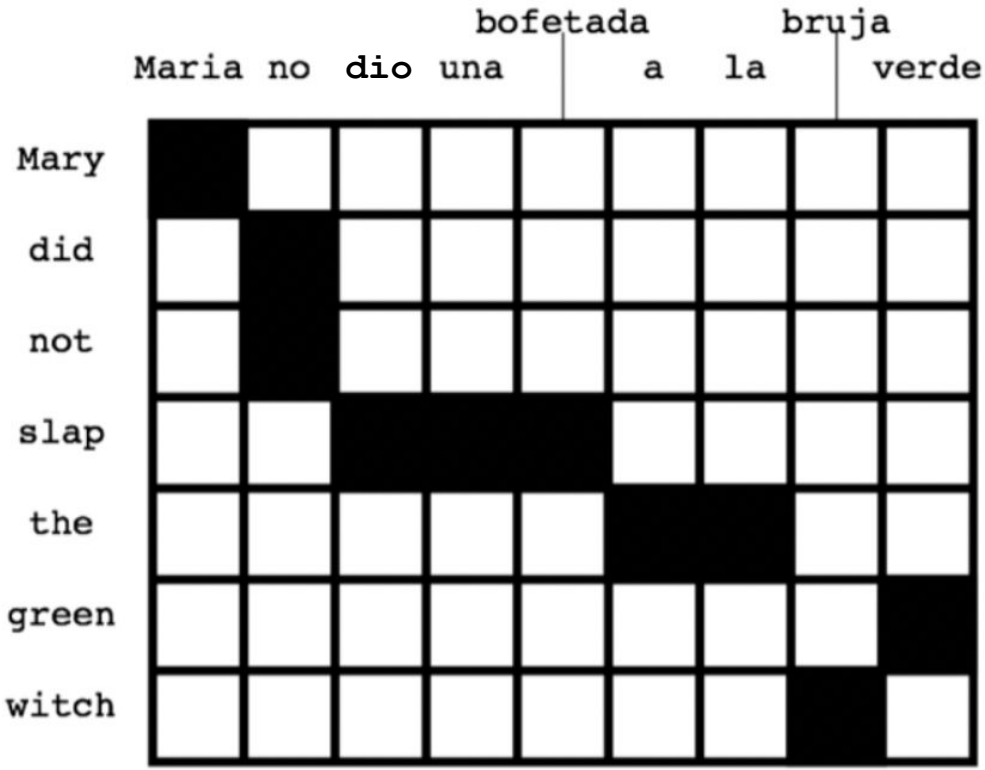
Maria did not slap the green witch.

produce a sentence in target language T.

Maria no dio una bofetada a la bruja verde.

How do we break this problem down into smaller parts?

Word alignments



The problem: those word-word alignments aren't given.

Early statistical machine translation models focused heavily on these alignments and how to get around not having them explicitly annotated in the data.

IBM models 1-6: we'll give a brief description of IBM model 1

IBM Model 1 (Brown et al., 1993)

Let ℓ and m be the (known) lengths of $\boldsymbol{e}$ and $\boldsymbol{f}$.
Latent variable $\boldsymbol{a} = \langle a_1, \ldots, a_m \rangle$, each a_i ranging over $\{0, \ldots, \ell\}$ (positions in $\boldsymbol{e}$).

- $a_4 = 3$ means that f_4 is "aligned" to e_3.
- $a_6 = 0$ means that f_6 is "aligned" to a special NULL symbol, e_0.

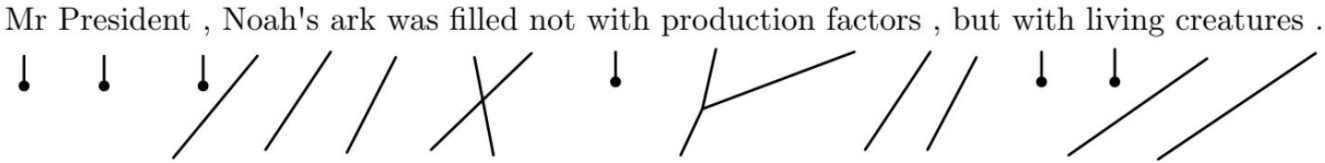
$$p(\boldsymbol{f} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \sum_{a_1=0}^{\ell} \sum_{a_2=0}^{\ell} \cdots \sum_{a_m=0}^{\ell} p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta})$$

$$= \sum_{\boldsymbol{a} \in \{0,\ldots,\ell\}^m} p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta})$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \prod_{i=1}^{m} p(a_i \mid i, \ell, m) \cdot p(f_i \mid e_{a_i}; \boldsymbol{\theta})$$

$$= \prod_{i=1}^{m} \frac{1}{\ell + 1} \cdot \theta_{f_i | e_{a_i}} = \left(\frac{1}{\ell + 1} \right)^m \prod_{i=1}^{m} \theta_{f_i | e_{a_i}}$$

Example: *f* is German



$$\boldsymbol{a} = \langle 4, 5, 6, 8, 7, ?, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{17+1} \cdot \theta_{\mathsf{Noahs|Noah's}} \cdot \frac{1}{17+1} \cdot \theta_{\mathsf{Arche|ark}}$$
$$\cdot \frac{1}{17+1} \cdot \theta_{\mathsf{war|was}} \cdot \frac{1}{17+1} \cdot \theta_{\mathsf{nicht|not}}$$
$$\cdot \frac{1}{17+1} \cdot \theta_{\mathsf{voller|filled}} \cdot \frac{1}{17+1} \cdot \theta_{\mathsf{Productionsfactoren|?}}$$

Seems like this could easily produce disfluent text?

Yeah, this was a real problem.

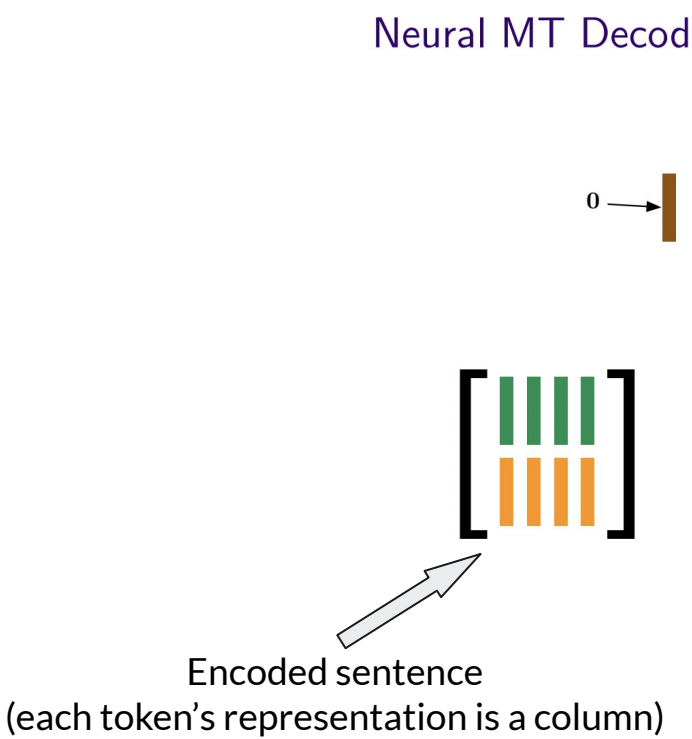
In practice, systems like this were often used in a **noisy channel** setup:

$$
\begin{aligned}
y^* &= \operatorname*{argmax}_y p(y \mid x) \\
&= \operatorname*{argmax}_y \frac{p(x \mid y) \cdot p(y)}{p(x)} \\
&= \operatorname*{argmax}_y \underbrace{p(x \mid y)}_{\text{channel model}} \cdot \underbrace{p(y)}_{\text{source model}}
\end{aligned}
$$

(In other words, they incorporated a language model to try and encourage fluency separately.)

Neural machine translation (NMT)

A neural approach: seq2seq

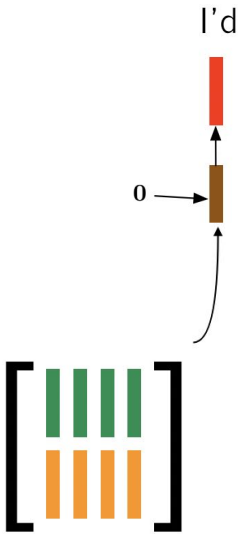


A neural approach: seq2seq

Neural MT Decoder

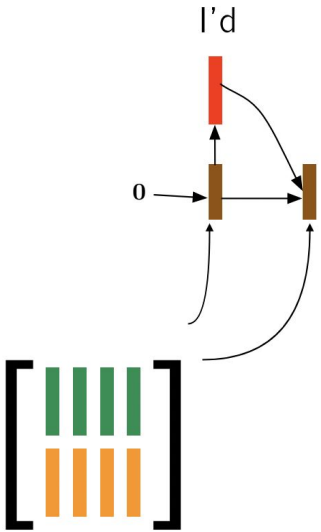
A neural approach: seq2seq

Neural MT Decoder



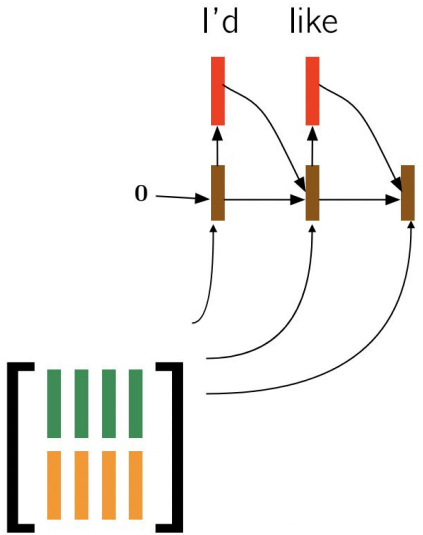
A neural approach: seq2seq

Neural MT Decoder



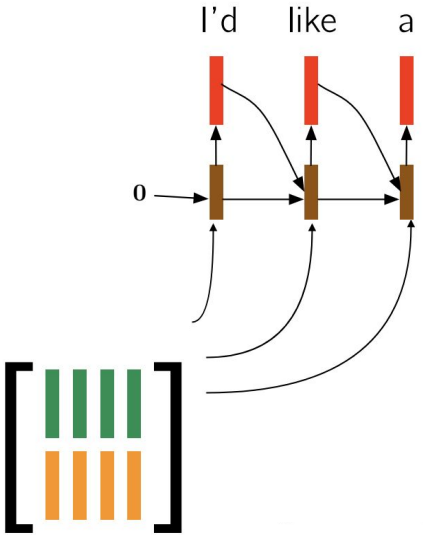
A neural approach: seq2seq

Neural MT Decoder



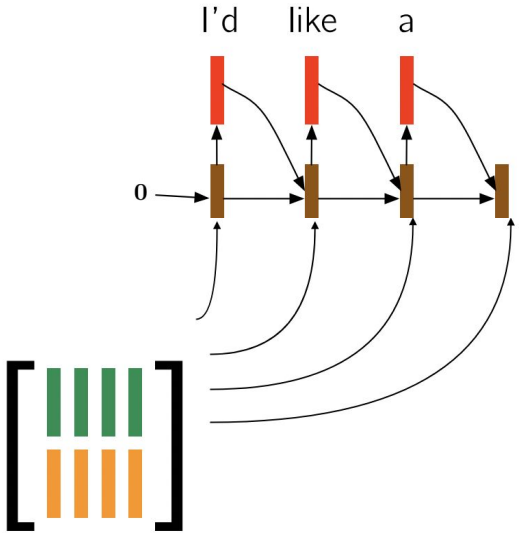
A neural approach: seq2seq

Neural MT Decoder



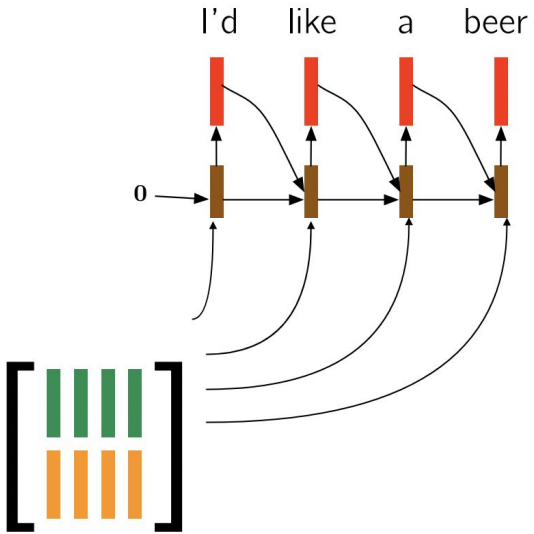
A neural approach: seq2seq

Neural MT Decoder



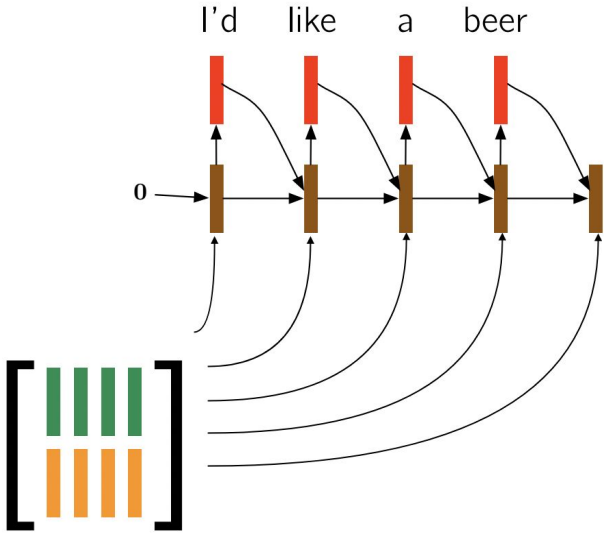
A neural approach: seq2seq

Neural MT Decoder



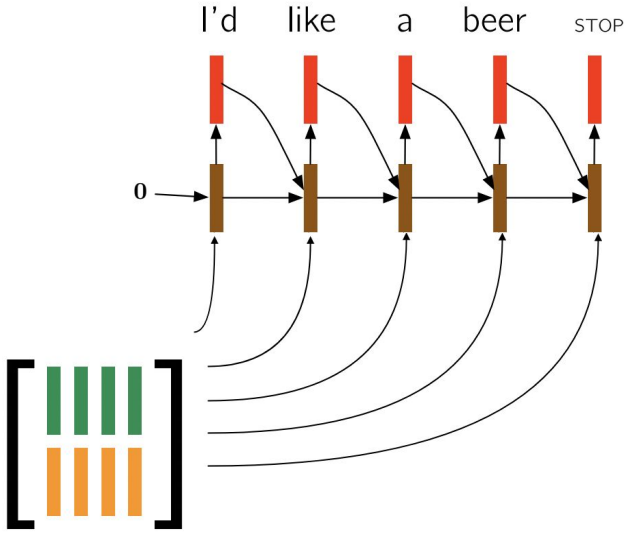
A neural approach: seq2seq

Neural MT Decoder



A neural approach: seq2seq

Neural MT Decoder



Notice: output sequence length does NOT have to match input sequence length!

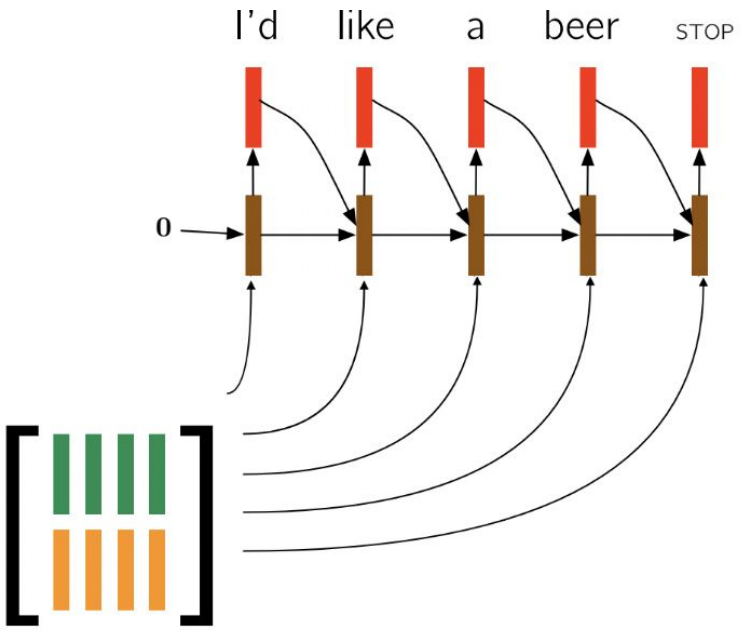
How do we turn our encoded sentence into a fixed-length vector at each timestep??

Our decoder needs the encoded input sentence as a fixed-length vector!

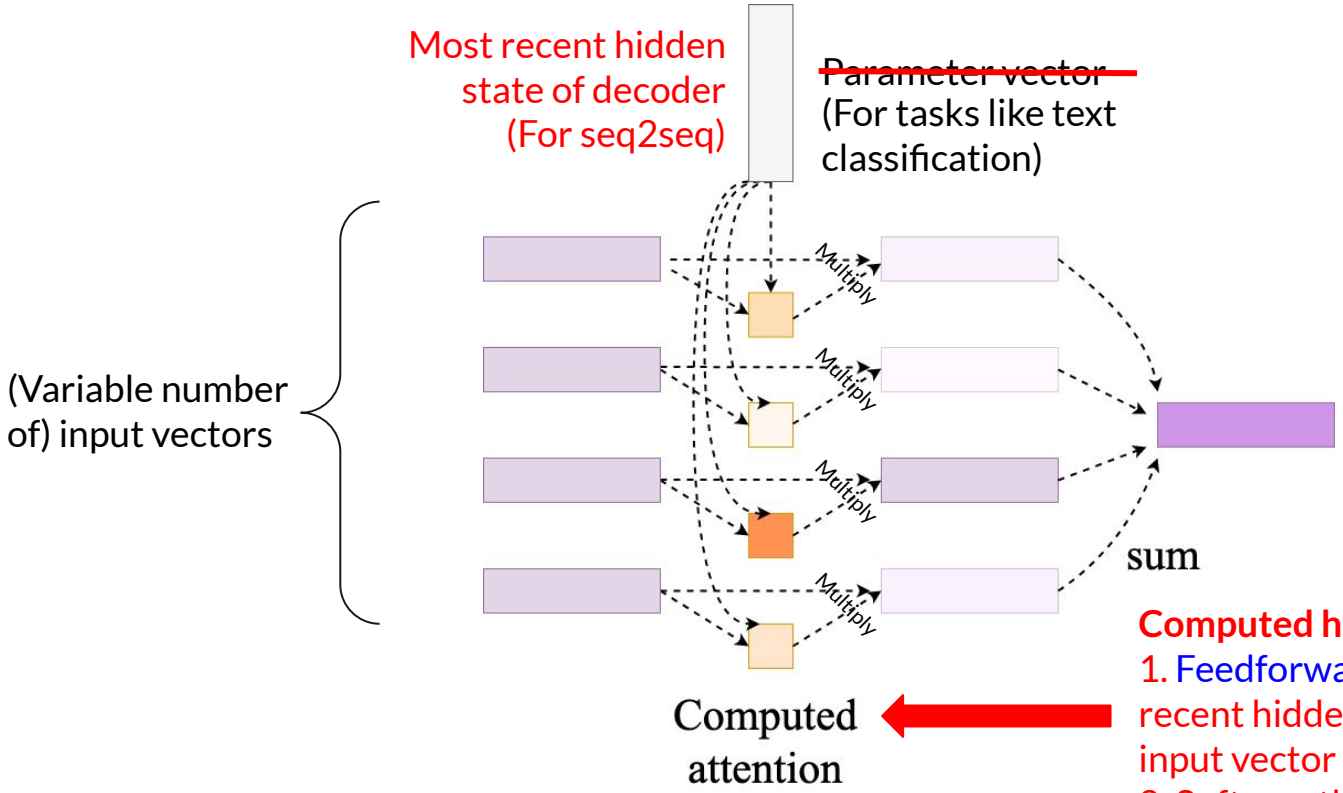
We *could* run an RNN over the input sentence and just take the final state…

… but then the representation of the encoded input sentence would be the same at each timestep of the output, when different parts of the output might need emphasis on different parts of the input.

What does this ring a bell from?



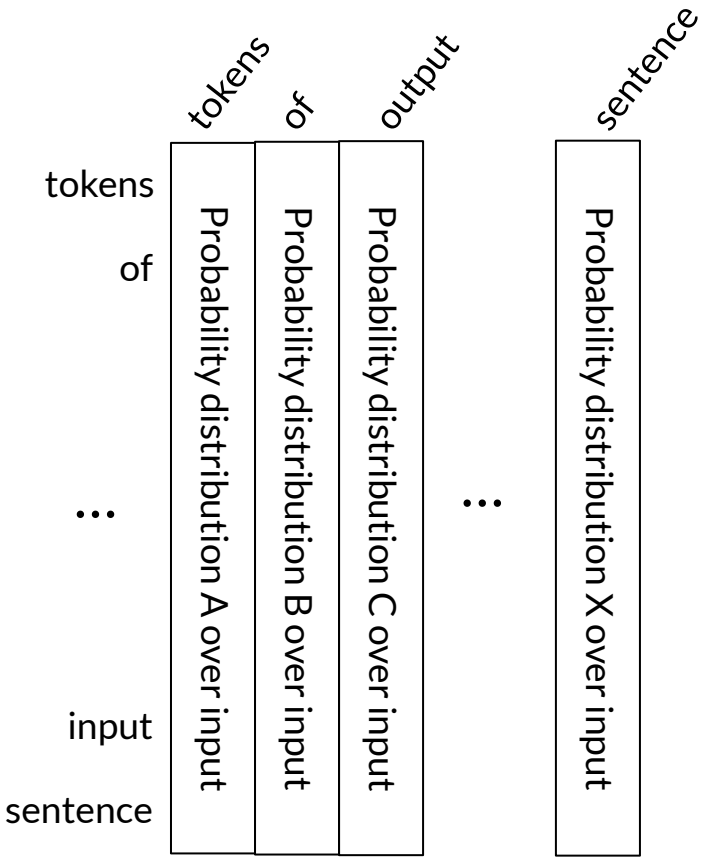
Enter the attention mechanism! (Bahdanau et al. 2014)



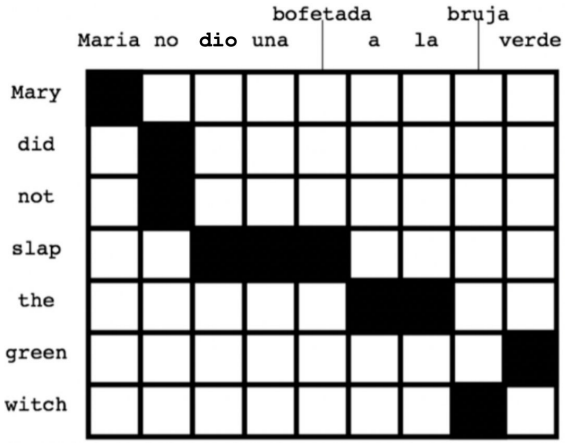
Computed how?

1. Feedforward network applied to most recent hidden state of decoder and each input vector
2. Softmax the set of resulting scalars.

So attention gives us…



which is basically a (soft) version of…



→ soft alignments!

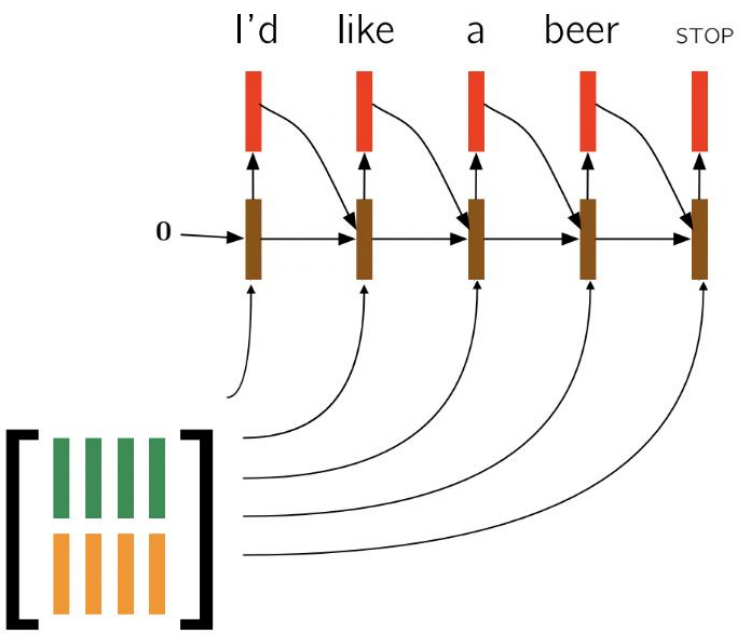
One more practical detail about decoding

It's tempting to approach picking the best path through the output space like we did for sequence labeling: with Viterbi.

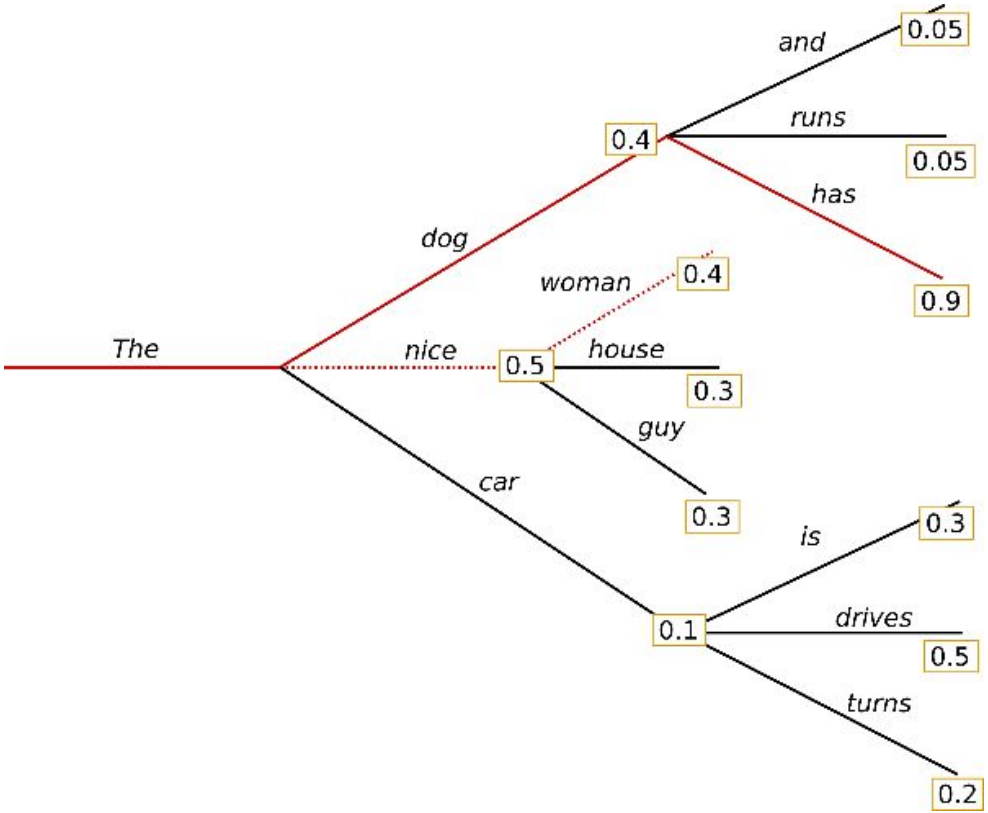
Two problems:

1. We don't know the length of the output sequence ahead of time → table of infinite length 😬
2. Size of output vocabulary is ENORMOUS compared to size of typical tagsets

What do we do if not Viterbi??



Beam search

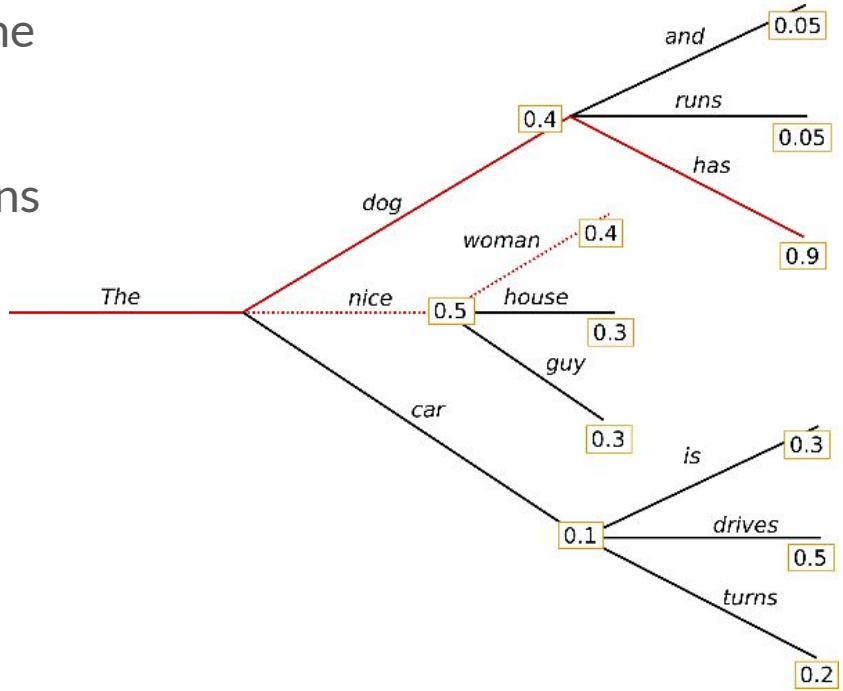


From Huggingface

Beam search

Greedy selection of *n* best paths up through time step t.

For timestep t+1, we only consider continuations from any one of those *n* best-looking paths.



From Huggingface

What's our loss function?

Just cross-entropy loss at each output timestep for correct output word (either pretending we've gotten all previous tokens in the output correct, or not– see this pytorch NMT tutorial for details).

Where might we find this kind of parallel data?

Note that this seq2seq approach can apply to more tasks than NMT!

Evaluating a machine translation system

Adequacy, fluency

In the past, these were usually separately computed elements of a score.

They’re still relevant today, but these days, they’re sort of both rolled into…

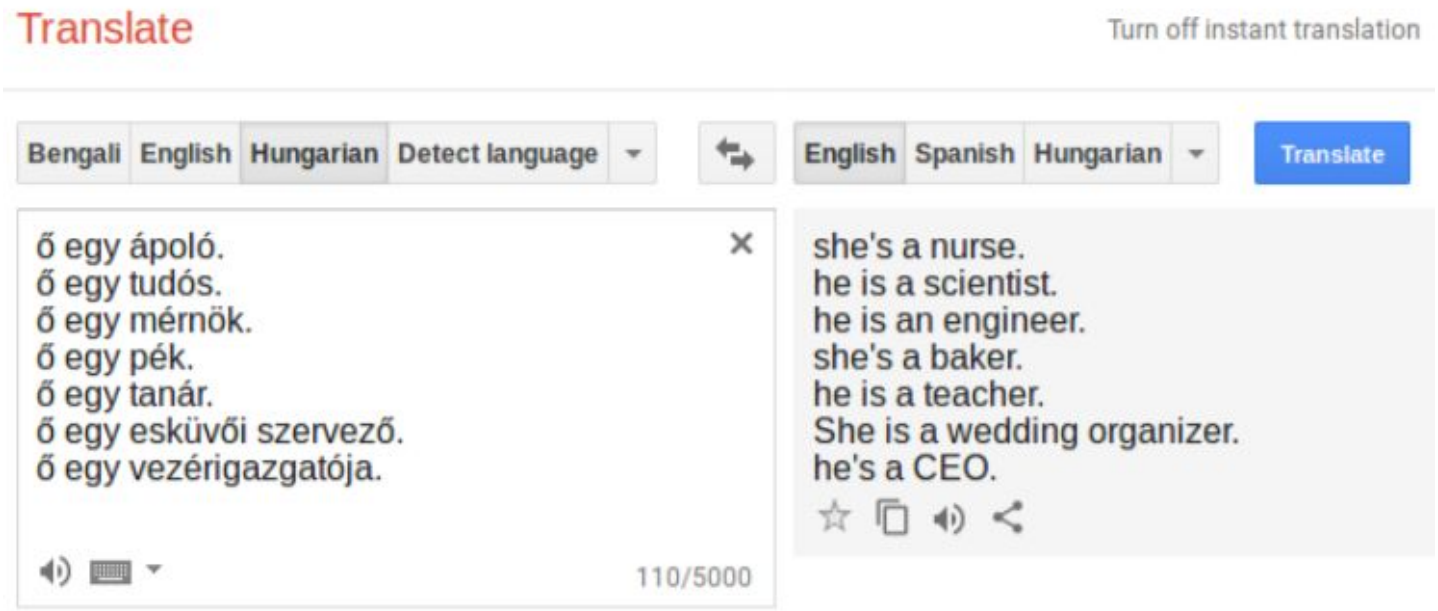
These days: BLEU score, mostly.

For each instance in your test set, have a set of human-written reference translations.

$$p_n = \frac{\text{number of } n\text{-grams appearing in both reference and hypothesis translations}}{\text{number of } n\text{-grams appearing in the hypothesis translation}}$$

Once you've got reference translations for a test set, pretty cheap to compute.

Evaluating bias in MT systems



See, for example, [Stanovsky et al. 2019]