Natural Language Processing (CSE 517 & 447): Sequence-to-Sequence Translation

Noah Smith

University of Washington
nasmith@cs.washington.edu

Winter 2022

Readings: Eisenstein (2019) 18

Machine Translation

The driving application motivating this lecture is automatic translation between natural languages, known as "machine translation" (MT).

The sequence-to-sequence (sometimes abbreviated "seq2seq") family of approaches was developed for MT, and we'll focus on that use case.

Today, it's applied to many problems in NLP. Out of the box, it's usually not *the best* thing you can do, but it's an easy starting point.

MT Evaluation

Intuition: good translations are **fluent** in the target language and **faithful** to the original meaning.

Bleu score (Papineni et al., 2002):

- Compare to a human-generated reference translation
- Or, better: multiple references
- Weighted average of n-gram precision (across different n)

There are some alternatives; most papers that use them report Bleu, too.

Better: human evaluations that compare output to reference.

Warren Weaver to Norbert Wiener, 1947

One naturally wonders if the problem of translation could be conceivably treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.'

Aperitif: Noisy Channel Models

A pattern for modeling a pair of random variables, X and Y:

$$\boxed{\text{source}} \longrightarrow Y \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

Aperitif: Noisy Channel Models

A pattern for modeling a pair of random variables, X and Y:

$$\boxed{\text{source}} \longrightarrow Y \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

- Y is the plaintext, the true message, the missing information, the output

Aperitif: Noisy Channel Models

A pattern for modeling a pair of random variables, X and Y:

$$\boxed{\text{source}} \longrightarrow Y \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

- Y is the plaintext, the true message, the missing information, the output
- X is the ciphertext, the garbled message, the observable evidence, the input

Aperitif: Noisy Channel Models

A pattern for modeling a pair of random variables, X and Y:

$$\boxed{\text{source}} \longrightarrow Y \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

- Y is the plaintext, the true message, the missing information, the output
- X is the ciphertext, the garbled message, the observable evidence, the input
- Decoding: select y given $X = x$.

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

Review from LM lecture: Speech Recognition

Successful speech recognition requires generating a word sequence that is:

- Faithful to the acoustic input
- Fluent

If we're mapping acoustics $\boldsymbol{a}$ to word sequences $\boldsymbol{w}$, then:

$$\boldsymbol{w}^* = \operatorname*{argmax}_{\boldsymbol{w}} \mathrm{Faithfulness}(\boldsymbol{w}; \boldsymbol{a}) + \mathrm{Fluency}(\boldsymbol{w})$$

Language models can provide a "fluency" score.

Review from LM lecture: Speech Recognition

Successful speech recognition requires generating a word sequence that is:

- Faithful to the acoustic input
- Fluent

If we're mapping acoustics $\boldsymbol{a}$ to word sequences $\boldsymbol{w}$, then:

$$\begin{aligned}
\boldsymbol{w}^* &= \operatorname*{argmax}_{\boldsymbol{w}} \mathrm{Faithfulness}(\boldsymbol{w}; \boldsymbol{a}) + \mathrm{Fluency}(\boldsymbol{w}) \\
&= \operatorname*{argmax}_{\boldsymbol{w}} \underbrace{\log p(\boldsymbol{a} \mid \boldsymbol{w})}_{\text{channel model}} + \underbrace{\log p(\boldsymbol{w})}_{\text{source model}}
\end{aligned}$$

Language models can provide a "fluency" score.

Bitext/Parallel Text

Let $\boldsymbol{f}$ and $\boldsymbol{e}$ be two sequences in French and English, respectively.

If we have enough such examples, we could estimate a conditional distribution $p(\boldsymbol{F} \mid \boldsymbol{E})$, known as the translation model.

In a noisy channel machine translation system, we could use this together with source/language model $p(\boldsymbol{E})$ to "decode" $\boldsymbol{f}$ into an English translation.

Reflection

Where might we find parallel data?

IBM Model 1

(Brown et al., 1993)

Let ℓ and m be the (known) lengths of $\boldsymbol{e}$ and $\boldsymbol{f}$.

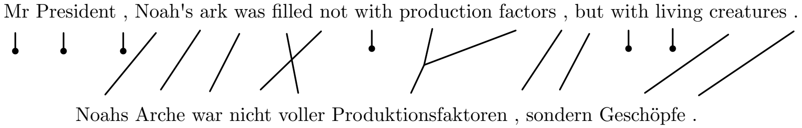
Latent variable $\boldsymbol{a} = \langle a_1, \ldots, a_m \rangle$, each a_i ranging over $\{0, \ldots, \ell\}$ (positions in $\boldsymbol{e}$).

- $a_4 = 3$ 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.

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

$$
\begin{aligned}
p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) &= \prod_{i=1}^{m} p(a_i \mid i, \ell, m) \cdot p(f_i \mid e_{a_i}; \boldsymbol{\theta}) \\
&= \prod_{i=1}^{m} \frac{1}{\ell + 1} \cdot \theta_{f_i \mid e_{a_i}} = \left(\frac{1}{\ell + 1} \right)^m \prod_{i=1}^{m} \theta_{f_i \mid e_{a_i}}
\end{aligned}
$$

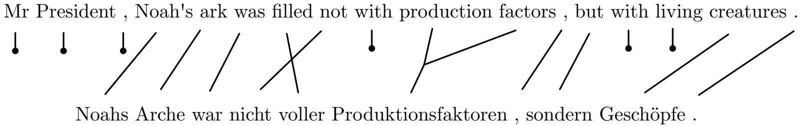
Example: $\boldsymbol{f}$ is German



$$\boldsymbol{a} = \langle 4, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{17+1} \cdot \theta_{\textsf{Noahs}|\textsf{Noah's}}$$

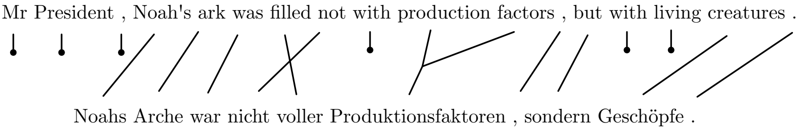
Example: $\boldsymbol{f}$ is German



$$\boldsymbol{a} = \langle 4, 5, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{17+1} \cdot \theta_{\mathsf{Noahs|Noah's}} \cdot \frac{1}{17+1} \cdot \theta_{\mathsf{Arche|ark}}$$

Example: $\boldsymbol{f}$ is German



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

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{17+1} \cdot \theta_{\text{Noahs}|\text{Noah's}} \cdot \frac{1}{17+1} \cdot \theta_{\text{Arche}|\text{ark}} \cdot \frac{1}{17+1} \cdot \theta_{\text{war}|\text{was}}$$

Example: $\boldsymbol{f}$ is German

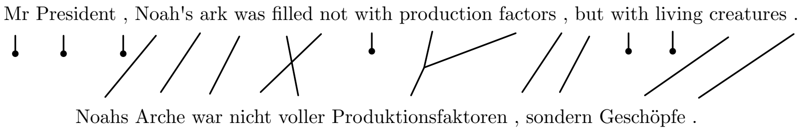
Mr President , Noah's ark was filled not with production factors , but with living creatures .

Noahs Arche war nicht voller Produktionsfaktoren , sondern Geschöpfe .

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

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

Example: $\boldsymbol{f}$ is German

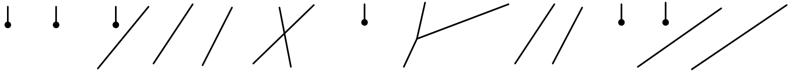


$$\boldsymbol{a} = \langle 4, 5, 6, 8, 7, \ldots \rangle$$

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

Example: $\boldsymbol{f}$ is German

Mr President , Noah's ark was filled not with production factors , but with living creatures .

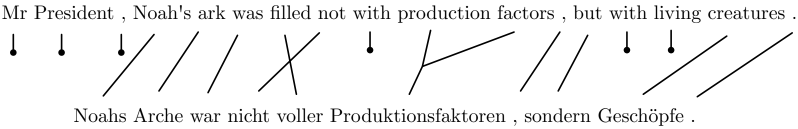


Noahs Arche war nicht voller Produktionsfaktoren , sondern Geschöpfe .

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

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

Example: $\boldsymbol{f}$ is German

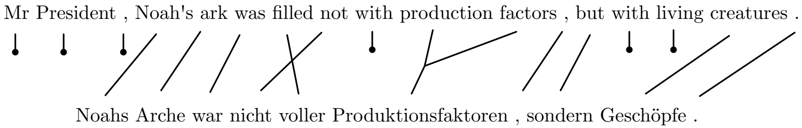


$$\boldsymbol{a} = \langle 4, 5, 6, 8, 7, ?, \ldots \rangle$$

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

Problem: This alignment isn't possible with IBM model 1! Each f_i is aligned to at most *one* e_{a_i}!

Example: $\boldsymbol{f}$ is English

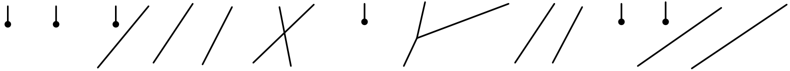


$$\boldsymbol{a} = \langle 0, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\text{NULL}}$$

Example: $\boldsymbol{f}$ is English

Mr President , Noah's ark was filled not with production factors , but with living creatures .

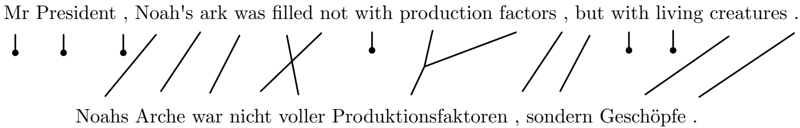


Noahs Arche war nicht voller Produktionsfaktoren , sondern Geschöpfe .

$$\boldsymbol{a} = \langle 0, 0, 0, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{President}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{,}|\textsc{null}}$$

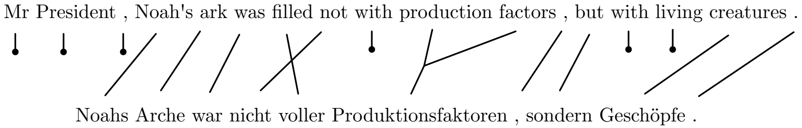
Example: $\boldsymbol{f}$ is English



$$\boldsymbol{a} = \langle 0, 0, 0, 1, \ldots \rangle$$

$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\text{NULL}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{President}|\text{NULL}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{,}|\text{NULL}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{Noah's}|\mathsf{Noahs}}$$

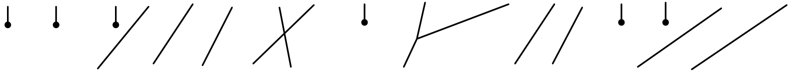
Example: $\boldsymbol{f}$ is English



$$
\begin{aligned}
\boldsymbol{a} &= \langle 0, 0, 0, 1, 2, \ldots \rangle \\
p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) &= \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{President}|\textsc{null}} \\
&\cdot \frac{1}{10+1} \cdot \theta_{,|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{Noah's}|\mathsf{Noahs}} \\
&\cdot \frac{1}{10+1} \cdot \theta_{\mathsf{ark}|\mathsf{Arche}}
\end{aligned}
$$

Example: $\boldsymbol{f}$ is English

Mr President , Noah's ark was filled not with production factors , but with living creatures .



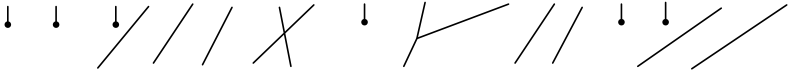
Noahs Arche war nicht voller Produktionsfaktoren , sondern Geschöpfe .

$$\boldsymbol{a} = \langle 0, 0, 0, 1, 2, 3, \ldots \rangle$$

$$\begin{aligned} p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = & \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{President}|\textsc{null}} \\ & \cdot \frac{1}{10+1} \cdot \theta_{,|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{Noah's}|\mathsf{Noahs}} \\ & \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{ark}|\mathsf{Arche}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{was}|\mathsf{war}} \end{aligned}$$

Example: $\boldsymbol{f}$ is English

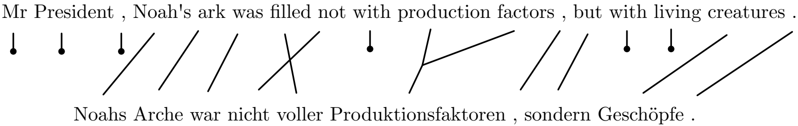
Mr President , Noah's ark was filled not with production factors , but with living creatures .



Noahs Arche war nicht voller Produktionsfaktoren , sondern Geschöpfe .

$$
\begin{aligned}
\boldsymbol{a} &= \langle 0, 0, 0, 1, 2, 3, 5, \ldots \rangle \\
p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) &= \frac{1}{10+1} \cdot \theta_{\mathsf{Mr}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{President}|\textsc{null}} \\
&\quad \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{,}|\textsc{null}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{Noah's}|\mathsf{Noahs}} \\
&\quad \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{ark}|\mathsf{Arche}} \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{was}|\mathsf{war}} \\
&\quad \cdot \frac{1}{10+1} \cdot \theta_{\mathsf{filled}|\mathsf{voller}}
\end{aligned}
$$

Example: $\boldsymbol{f}$ is English



$$\boldsymbol{a} = \langle 0, 0, 0, 1, 2, 3, 5, 4, \ldots \rangle$$

$$\begin{aligned} p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) = & \frac{1}{10+1} \cdot \theta_{\text{Mr}|\text{NULL}} \cdot \frac{1}{10+1} \cdot \theta_{\text{President}|\text{NULL}} \\ & \cdot \frac{1}{10+1} \cdot \theta_{,|\text{NULL}} \cdot \frac{1}{10+1} \cdot \theta_{\text{Noah's}|\text{Noahs}} \\ & \cdot \frac{1}{10+1} \cdot \theta_{\text{ark}|\text{Arche}} \cdot \frac{1}{10+1} \cdot \theta_{\text{was}|\text{war}} \\ & \cdot \frac{1}{10+1} \cdot \theta_{\text{filled}|\text{voller}} \cdot \frac{1}{10+1} \cdot \theta_{\text{not}|\text{nicht}} \end{aligned}$$

Reflection

This is a problem of **incomplete data**: at training time, we see e and f, but not a. Have we seen anything like this before?

Expectation Maximization

Review from vector embeddings lecture!

Many ways to understand it. Today, we'll stick with a simple one.

Start with arbitrary (e.g., random) parameter values. Alternate between two steps:

- E step: calculate the posterior distribution over each word's assignment to an other-language word (today) or a topic (in PLSA).
- M step: treat the posteriors as soft counts, and re-estimate the model.

Doing this is a kind of hill-climbing on the likelihood of the *observed* data.

"Complete Data" IBM Model 1

Let the training data consist of N word-aligned sentence pairs: $\langle \boldsymbol{e}_1^{(1)}, \boldsymbol{f}^{(1)}, \boldsymbol{a}^{(1)} \rangle, \ldots, \langle \boldsymbol{e}^{(N)}, \boldsymbol{f}^{(N)}, \boldsymbol{a}^{(N)} \rangle$.

Define:

$$q_{n,i}(j) = \begin{cases} 1 & \text{if } a_i^{(n)} = j \\ 0 & \text{otherwise} \end{cases}$$

Maximum likelihood estimate for $\theta_{f|e}$:

$$\hat{\theta}_{f|e} = \frac{\text{count}(e, f)}{\text{count}(e)} = \frac{\displaystyle\sum_{n=1}^{N} \sum_{i:f_i^{(n)}=f} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}{\displaystyle\sum_{n=1}^{N} \sum_{i=1}^{m^{(n)}} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}$$

MLE with "Soft" Counts for IBM Model 1

Let the training data consist of N "softly" aligned sentence pairs, $\langle \boldsymbol{e}_1^{(1)}, \boldsymbol{f}^{(1)}, \rangle, \ldots, \langle \boldsymbol{e}^{(N)}, \boldsymbol{f}^{(N)} \rangle$.

Now, let $q_{n,i}(j)$ be "soft," interpreted as:

$$q_{n,i}(j) = p(a_i^{(n)} = j; \boldsymbol{\theta})$$

Maximum likelihood estimate for $\theta_{f|e}$:

$$\hat{\theta}_{f|e} = \frac{\displaystyle\sum_{n=1}^{N} \sum_{i:f_i^{(n)}=f} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}{\displaystyle\sum_{n=1}^{N} \sum_{i=1}^{m^{(n)}} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}$$

Expectation Maximization Algorithm for IBM Model 1

1. Initialize $\boldsymbol{\theta}$ to some arbitrary values.
2. E step: use current $\boldsymbol{\theta}$ to estimate expected ("soft") counts.

$$q_{n,i}(j) \leftarrow \theta_{f_i^{(n)} | e_j^{(n)}} \Bigg/ \sum_{j'=1}^{\ell^{(n)}} \theta_{f_i^{(n)} | e_{j'}^{(n)}}$$

3. M step: carry out "soft" MLE.

$$\hat{\theta}_{f|e} \leftarrow \frac{\displaystyle\sum_{n=1}^{N} \sum_{i : f_i^{(n)} = f} \sum_{j : e_j^{(n)} = e} q_{n,i}(j)}{\displaystyle\sum_{n=1}^{N} \sum_{i=1}^{m^{(n)}} \sum_{j : e_j^{(n)} = e} q_{n,i}(j)}$$

4. Go to 2 until converged.

Expectation Maximization

- Originally introduced in the 1960s for estimating HMMs when the states really are "hidden."
- Can be applied to any generative model with hidden variables (we saw it for PLSA earlier in the class).
- Greedily attempts to maximize probability of the observable data, marginalizing over latent variables. For IBM model 1, that means:

$$\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} p(\boldsymbol{f}^{(n)} \mid \boldsymbol{e}^{(n)}; \boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{\boldsymbol{a}} p(\boldsymbol{f}^{(n)}, \boldsymbol{a} \mid \boldsymbol{e}^{(n)}; \boldsymbol{\theta})$$

- Usually converges only to a *local* optimum of the above, which is in general not convex.
- Strangely, for IBM model 1 (and very few other models), it *is* convex!

IBM Model 2

(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 $\boldsymbol{e}$).

- E.g., $a_4 = 3$ means that f_4 is "aligned" to e_3.

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

$$\begin{aligned} p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}, m; \boldsymbol{\theta}) &= \prod_{i=1}^{m} p(a_i \mid i, \ell, m; \boldsymbol{\theta}) \cdot p(f_i \mid e_{a_i}; \boldsymbol{\theta}) \\ &= \theta^{\text{distortion}}_{a_i \mid i, \ell, m} \cdot \theta^{\text{translation}}_{f_i \mid e_{a_i}} \end{aligned}$$

Variations

- Dyer et al. (2013) introduced a new parameterization:

$$\theta^{\text{distortion}}_{j|i,\ell,m} \propto \exp -\lambda \left| \frac{i}{m} - \frac{j}{\ell} \right|$$

  (This is called `fast_align`.)

- IBM models 3–5 (Brown et al., 1993) introduced increasingly more powerful ideas, such as "fertility" and "distortion."

Some History

Obstacles for noisy channel MT:

- Proprietary implementation; open-source implementation of IBM model didn't come until 1999 (Al-Onaizan et al., 1999)!
- No decoding algorithm was offered; even for simple models exact decoding is NP-complete (Knight, 1999).
- No automatic evaluation until the Bleu score (Papineni et al., 2002).

Some History

Obstacles for noisy channel MT:

- Proprietary implementation; open-source implementation of IBM model didn't come until 1999 (Al-Onaizan et al., 1999)!
- No decoding algorithm was offered; even for simple models exact decoding is NP-complete (Knight, 1999).
- No automatic evaluation until the Bleu score (Papineni et al., 2002).

By the early 2000s, it was becoming clear that modeling translation "word-by-word" was missing out on powerful contextual cues. There were two solutions in friendly competition:

- Phrase-based translation: work with chunks of words instead of words.
- Syntax-based translation: use parse trees of input, output, or both.

From Alignment to (Phrase-Based) Translation

Obtaining word alignments in a parallel corpus is a common first step in building a machine translation system.

1. Infer alignments between the words, using the IBM models.
2. Extract and score **phrase pairs**.
3. Estimate a global scoring function to optimize (a proxy for) translation quality.
4. Decode French sentences into English ones.

The noisy channel pattern isn't taken quite so seriously when we build real systems, but we still have notions of faithfulness and fluency, and **language models** are really, really important for the latter.

Phrases?

Phrase-based translation uses automatically-induced subsequences/chunks of words.

Examples of Phrases

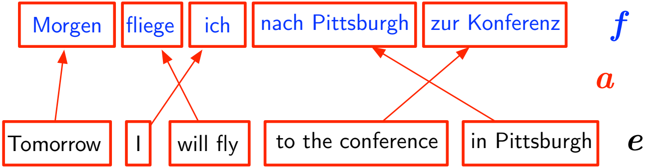
Courtesy of Chris Dyer.

German	English	$p(\bar{\boldsymbol{f}} \mid \bar{\boldsymbol{e}})$
das Thema	the issue	0.41
	the point	0.72
	the subject	0.47
	the thema	0.99
es gibt	there is	0.96
	there are	0.72
morgen	tomorrow	0.90
fliege ich	will I fly	0.63
	will fly	0.17
	I will fly	0.13

Phrase-Based Translation Model

Originated by Koehn et al. (2003).

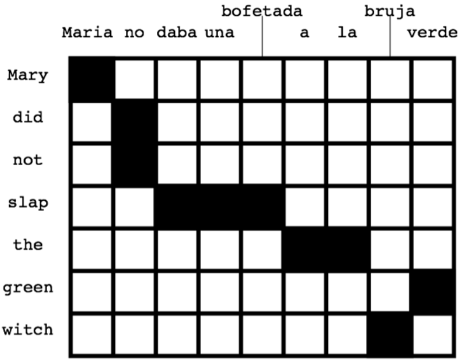
R.v. $\boldsymbol{A}$ captures segmentation of sentences into phrases, alignment between them, and reordering.



$$p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e}) = p(\boldsymbol{a} \mid \boldsymbol{e}) \cdot \prod_{i=1}^{|\boldsymbol{a}|} p(\bar{\boldsymbol{f}}_i \mid \bar{\boldsymbol{e}}_i)$$

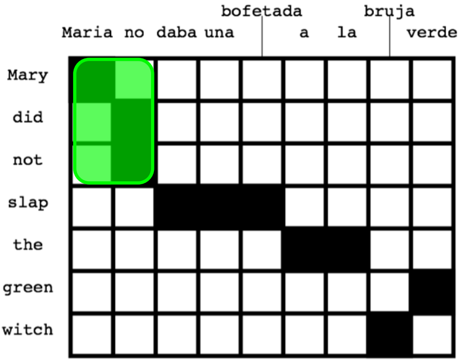
Extracting Phrases

After inferring word alignments, apply heuristics.



Extracting Phrases

After inferring word alignments, apply heuristics.

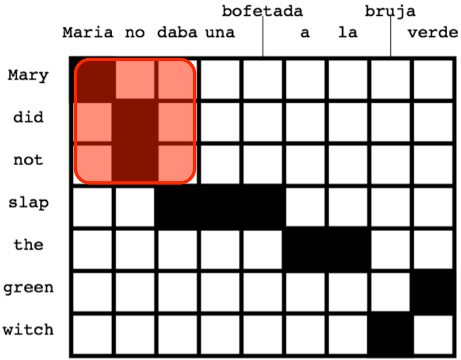


Extracting Phrases

After inferring word alignments, apply heuristics.

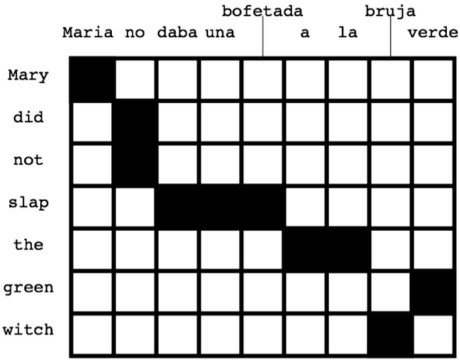
Extracting Phrases

After inferring word alignments, apply heuristics.



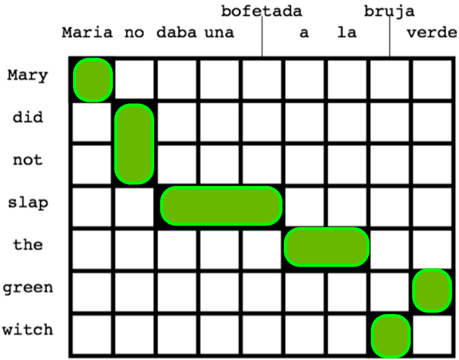
Extracting Phrases

After inferring word alignments, apply heuristics.



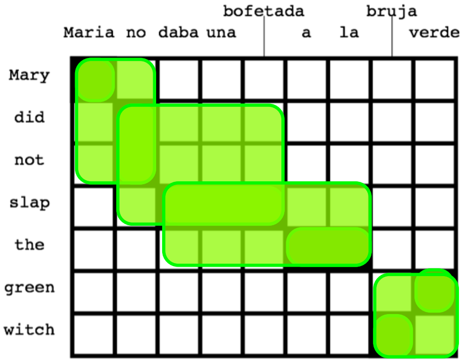
Extracting Phrases

After inferring word alignments, apply heuristics.



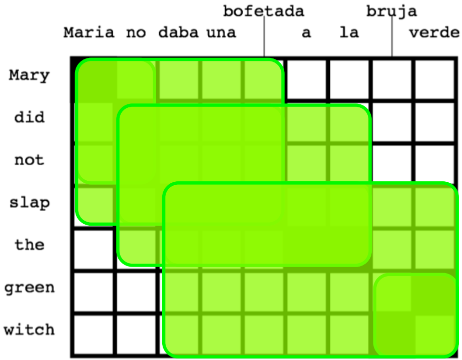
Extracting Phrases

After inferring word alignments, apply heuristics.



Extracting Phrases

After inferring word alignments, apply heuristics.



Scoring Whole Translations

$$\text{score}(\boldsymbol{e}, \boldsymbol{a}; \boldsymbol{f}) = \underbrace{\log p(\boldsymbol{e})}_{\text{language model}} + \underbrace{\log p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e})}_{\text{translation model}}$$

Remarks:

- Segmentation, alignment, reordering are all predicted as well (not marginalized).
- This does not factor nicely.

Scoring Whole Translations

$$\text{score}(\boldsymbol{e}, \boldsymbol{a}; \boldsymbol{f}) = \underbrace{\log p(\boldsymbol{e})}_{\text{language model}} + \underbrace{\log p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e})}_{\text{translation model}}$$

$$+ \underbrace{\log p(\boldsymbol{e}, \boldsymbol{a} \mid \boldsymbol{f})}_{\text{reverse t.m.}}$$

Remarks:

- Segmentation, alignment, reordering are all predicted as well (not marginalized).
- This does not factor nicely.
- I am simplifying!
  - Reverse translation model typically included.

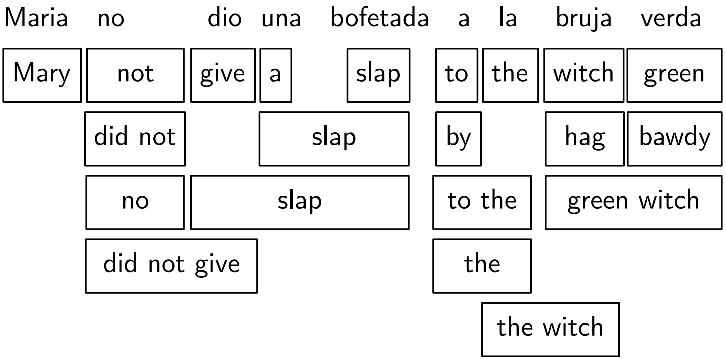
Scoring Whole Translations

$$\text{score}(\boldsymbol{e}, \boldsymbol{a}; \boldsymbol{f}) = \beta_{\text{l.m.}} \underbrace{\log p(\boldsymbol{e})}_{\text{language model}} + \beta_{\text{t.m.}} \underbrace{\log p(\boldsymbol{f}, \boldsymbol{a} \mid \boldsymbol{e})}_{\text{translation model}}$$
$$+ \beta_{\text{r.t.m.}} \underbrace{\log p(\boldsymbol{e}, \boldsymbol{a} \mid \boldsymbol{f})}_{\text{reverse t.m.}}$$

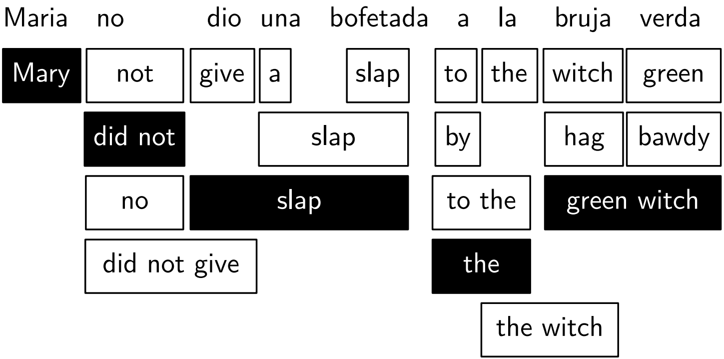
Remarks:

- Segmentation, alignment, reordering are all predicted as well (not marginalized).
- This does not factor nicely.
- I am simplifying!
  - Reverse translation model typically included.
  - Each log-probability is treated as a "feature" and weights are optimized for Bleu performance.

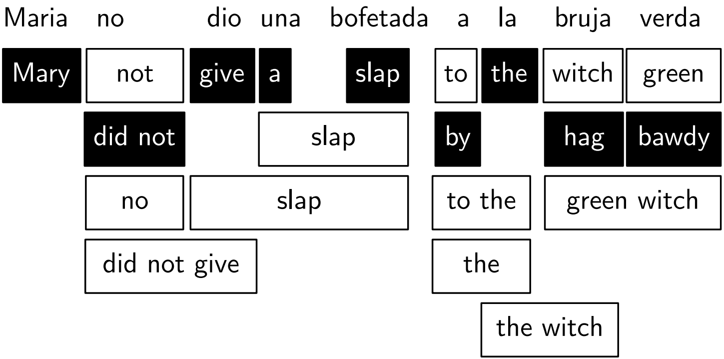
Decoding: Example



Decoding: Example



Decoding: Example



Beam Search for Sequential Classifiers

Review from conditional random fields lecture.

Input: $\boldsymbol{x}$ (length n), a sequential classifier's scoring function score, and beam width k

Let H_0 score hypotheses at position 0, defining only $H_0(\langle\rangle) = 0$.
For $i \in \{1, \ldots, n\}$:

- Empty C.
- For each hypothesis $\hat{\boldsymbol{y}}_{1:i-1}$ scored by H_{i-1}:
  - For each $y \in \mathcal{L}$, place new hypothesis $\hat{\boldsymbol{y}}_{1:i}y \rightarrow H_{i-1}(\hat{\boldsymbol{y}}_{1:i}) + \mathrm{score}(\boldsymbol{x}, i, \hat{\boldsymbol{y}}_{1:i-1}, y)$ into C.
- Let H_i be the k-best scored elements of C.

Output: best scored element of H_n.

Decoding in Phrase-Based MT

Adapted from Koehn et al. (2006).

Initial state: $\langle \underbrace{\circ \circ \ldots \circ}_{|\boldsymbol{f}|}, \text{``''} \rangle$ with score 0

Goal state: $\langle \underbrace{\bullet \bullet \ldots \bullet}_{|\boldsymbol{f}|}, \boldsymbol{e}^* \rangle$ with (approximately) the highest score

Reaching a new state:

- Find an uncovered span of $\boldsymbol{f}$ for which a phrasal translation exists in the input $(\bar{\boldsymbol{f}}, \bar{\boldsymbol{e}})$
- New state appends $\bar{\boldsymbol{e}}$ to the output and "covers" $\bar{\boldsymbol{f}}$.
- Score of new state includes additional language model, translation model components for the global score.

Reflection

Consider how decoding with phrase-based MT (slide 57), which might not always move left-to-right across the input, differs from the sequential classification case (slide 56). How might you modify the beam search algorithm to allow the kind of exploration we need to decode with the models described here?

Decoding Example

$$\langle \circ\circ\circ\circ\circ\circ\circ\circ\circ, \text{``''} \rangle, 0$$

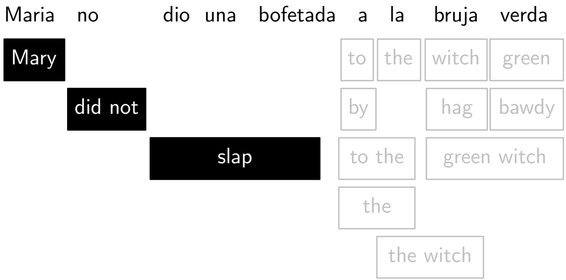
Decoding Example

$$\langle \bullet\circ\circ\circ\circ\circ\circ\circ\circ, \text{“Mary”} \rangle,\ \log p_{\text{l.m.}}(\text{Mary}) + \log p_{\text{t.m.}}(\text{Maria} \mid \text{Mary})$$

Decoding Example

$$\langle \bullet \bullet \circ \circ \circ \circ \circ \circ \circ, \text{“Mary did not”} \rangle,$$
$$\log p_{\text{l.m.}}(\text{Mary did not}) + \log p_{\text{t.m.}}(\text{Maria} \mid \text{Mary})$$
$$+ \log p_{\text{t.m.}}(\text{no} \mid \text{did not})$$

Decoding Example



$$\langle \bullet\bullet\bullet\bullet\bullet\circ\circ\circ\circ, \text{“Mary did not slap”}\rangle,$$
$$\log p_{\text{l.m.}}(\text{Mary did not slap}) + \log p_{\text{t.m.}}(\text{Maria} \mid \text{Mary})$$
$$+ \log p_{\text{t.m.}}(\text{no} \mid \text{did not}) + \log p_{\text{t.m.}}(\text{dio una bofetada} \mid \text{slap})$$

Machine Translation: Remarks

Sometimes phrases are organized hierarchically (Chiang, 2007).

Extensive research on syntax-based machine translation (Galley et al., 2004), but requires considerable engineering to match phrase-based systems.

Some good pre-neural overviews: Lopez (2008); Koehn (2009)

The Main Dish

Neural Machine Translation

Original idea proposed by Forcada and Ñeco (1997); resurgence in interest starting around 2013.

Strong starting point for current work: Bahdanau et al. (2014). (My exposition is borrowed with gratitude from a lecture by Chris Dyer.)

This approach eliminates (hard) alignment and phrases.

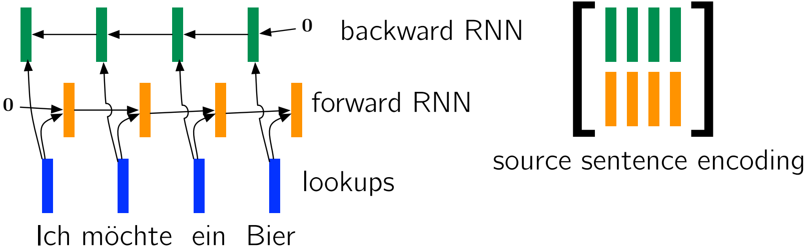
Take care: here, the terminology “encoder” and “decoder” are used differently than in the noisy-channel pattern.

High-Level Model

$$
\begin{aligned}
p(\boldsymbol{E} = \boldsymbol{e} \mid \boldsymbol{f}) &= p(\boldsymbol{E} = \boldsymbol{e} \mid \mathsf{encode}(\boldsymbol{f})) \\
&= \prod_{j=1}^{\ell} p(e_j \mid e_0, \ldots, e_{j-1}, \mathsf{encode}(\boldsymbol{f}))
\end{aligned}
$$

The encoding of the source sentence is a *deterministic* function of the words in that sentence.

Neural MT Source-Sentence Encoder



$\mathbf{F}$ is a $d \times m$ matrix encoding the source sentence $\boldsymbol{f}$ (length m). Originally, RNNs (depicted here) were used; now transformers are more popular (Vaswani et al., 2017).

Decoder: Contextual Language Model

Two inputs, the previous word and the source sentence context.

$$\mathbf{s}_t = g_{\text{recurrent}}(\mathbf{e}_{e_{t-1}}, \underbrace{\mathbf{F}\mathbf{a}_t}_{\text{“context”}}, \mathbf{s}_{t-1})$$

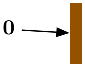
$$\mathbf{y}_t = g_{\text{output}}(\mathbf{s}_t)$$

$$p(E_t = v \mid e_1, \ldots, e_{t-1}, \boldsymbol{f}) = [\mathbf{y}_t]_v$$

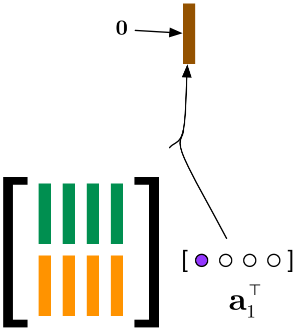
(The forms of the two component gs are suppressed; just remember that they (i) have parameters and (ii) are differentiable with respect to those parameters.)

The neural language model we discussed earlier (Mikolov et al., 2010) didn't have the context as an input to $g_{\text{recurrent}}$.

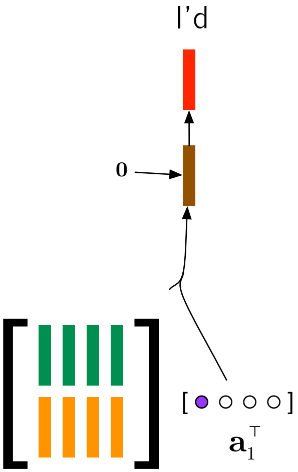
Neural MT Decoder



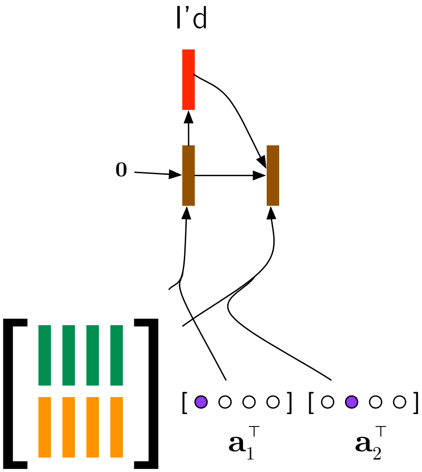
Neural MT Decoder



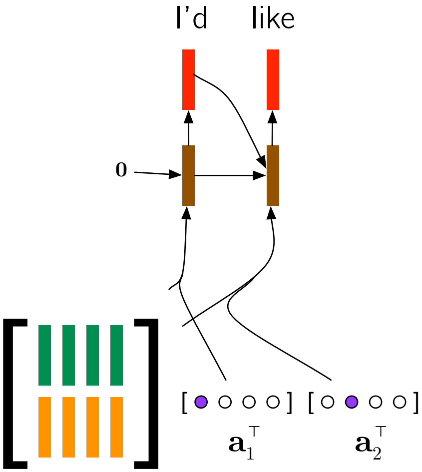
Neural MT Decoder



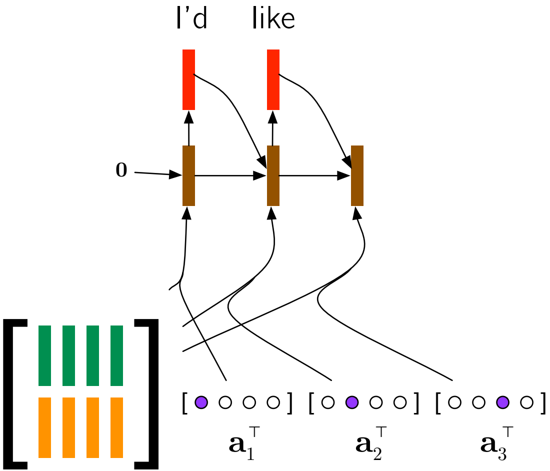
Neural MT Decoder



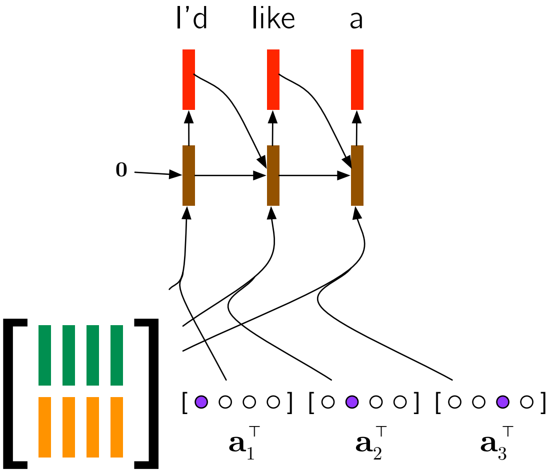
Neural MT Decoder



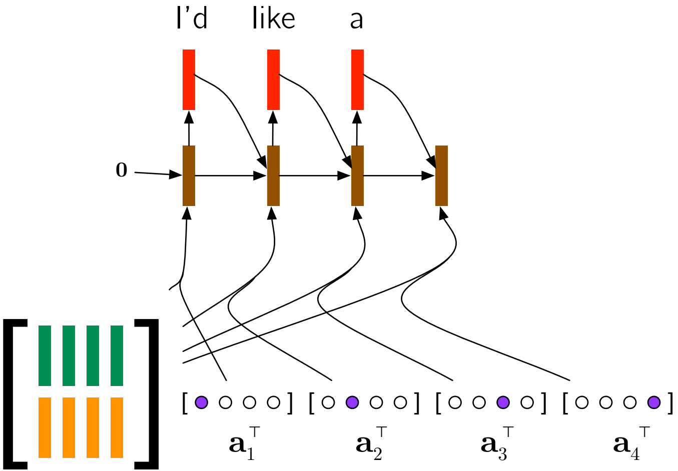
Neural MT Decoder



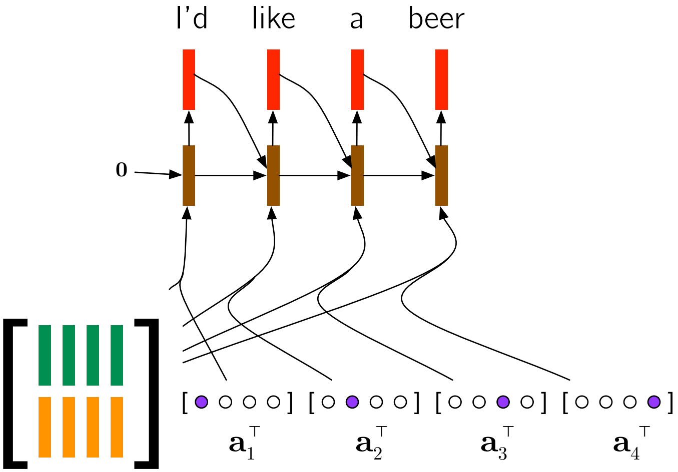
Neural MT Decoder



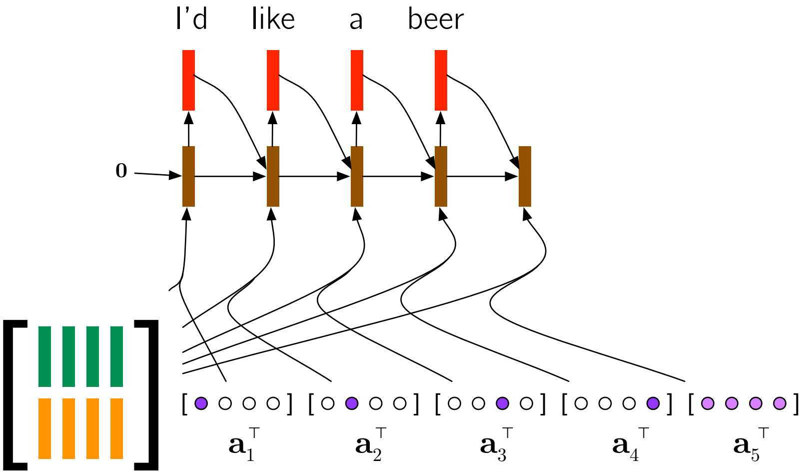
Neural MT Decoder



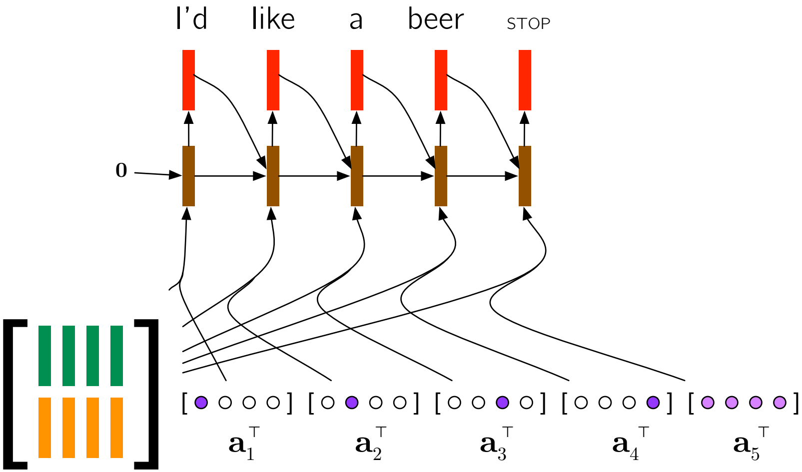
Neural MT Decoder



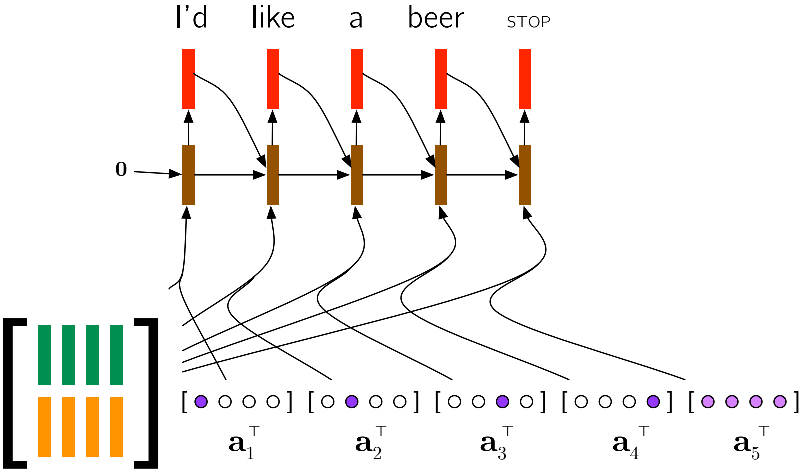
Neural MT Decoder

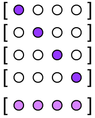


Neural MT Decoder



Neural MT Decoder





Computing "Attention"

Let $\mathbf{V}\mathbf{s}_{t-1}$ be the "expected" input embedding for timestep t. (Parameters: $\mathbf{V}$.)

Attention is $\mathbf{a}_t = \text{softmax}\left(\mathbf{F}^\top \mathbf{V}\mathbf{s}_{t-1}\right)$.

Context is $\mathbf{F}\mathbf{a}_t$, i.e., a weighted sum of the source words' in-context representations.

With transformers, there's also attention over the previously decoded target-language words.

Learning and Decoding

$$\log p(\boldsymbol{e} \mid \textsf{encode}(\boldsymbol{f})) = \sum_{i=1}^{m} \log p(e_i \mid \boldsymbol{e}_{0:i-1}, \textsf{encode}(\boldsymbol{f}))$$

is differentiable with respect to all parameters of the neural network, allowing "end-to-end" training.

Decoding typically uses beam search.

Remarks

We covered two approaches to machine translation:

- Phrase-based statistical MT following Koehn et al. (2003), including probabilistic noisy-channel models for alignment (a key preprocessing step; Brown et al., 1993), and
- Neural MT with attention, following Bahdanau et al. (2014).

Note two key differences:

- Noisy channel $p(\boldsymbol{e}) \times p(\boldsymbol{f} \mid \boldsymbol{e})$ vs. "direct" model $p(\boldsymbol{e} \mid \boldsymbol{f})$
- Alignment as a discrete random variable vs. attention as a deterministic, differentiable function

Additional Notes

We didn't talk about tokenization; current systems split words into smaller units (Sennrich et al., 2016b; Wu et al., 2016) for better generalization to unseen words.

Neural MT is the strongest approach today, at least when you have enough data.

When monolingual target-language data is plentiful, we'd like to use it! Some recent neural models try (Sennrich et al., 2016a; Xia et al., 2016; Yu et al., 2017).

Limitations

MT is now widely deployed commercially and works well, for some language pairs and some genres of usage. Expect degradation on any language variety that looks different from the training data. All MT models pick up cultural biases (Stanovsky et al., 2019).

Digestif: Sequence-to-Sequence Everything?

Some have recently proposed the MT-derived sequence-to-sequence paradigm as a way to tackle a much broader range of NLP problems, including summarization, question answering, and even non-traditionally sequential tasks like classification (Raffel et al., 2020; Lewis et al., 2020).

This view also extends to pretraining, as you might expect.

References I

Yaser Al-Onaizan, Jan Cuřin, Michael Jahr, Kevin Knight, John Lafferty, I. Dan Melamed, Noah A. Smith, Franz-Josef Och, David Purdy, and David Yarowsky. Statistical machine translation. CLSP Research Notes 42, Johns Hopkins University, 1999.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*, 2014. URL `https://arxiv.org/abs/1409.0473`.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

David Chiang. Hierarchical phrase-based translation. *computational Linguistics*, 33(2): 201–228, 2007.

Chris Dyer, Victor Chahuneau, and Noah A Smith. A simple, fast, and effective reparameterization of IBM Model 2. In *Proc. of NAACL*, 2013.

Jacob Eisenstein. *Introduction to Natural Language Processing*. MIT Press, 2019.

Mikel L. Forcada and Ramón P. Ñeco. Recursive hetero-associative memories for translation. In *International Work-Conference on Artificial Neural Networks*, 1997.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *Proc. of NAACL*, 2004.

Kevin Knight. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615, 1999.

References II

Philipp Koehn. *Statistical Machine Translation*. Cambridge University Press, 2009.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proc. of NAACL*, 2003.

Philipp Koehn, Marcello Federico, Wade Shen, Nicola Bertoldi, Ondrej Bojar, Chris Callison-Burch, Brooke Cowan, Chris Dyer, Hieu Hoang, and Richard Zens. Open source toolkit for statistical machine translation: Factored translation models and confusion network decoding, 2006. Final report of the 2006 JHU summer workshop.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, 2020.

Adam Lopez. Statistical machine translation. *ACM Computing Surveys*, 40(3):8, 2008.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proc. of Interspeech*, 2010. URL `http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf`.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, 2002.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

References III

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proc. of ACL*, 2016a. URL `http://www.aclweb.org/anthology/P16-1009`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proc. of ACL*, 2016b.

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. Evaluating gender bias in machine translation. In *Proceedings of ACL*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation, 2016. arXiv:1609.08144.

Yingce Xia, Di He, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *NeurIPS*, 2016.

Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomas Kocisky. The neural noisy channel. In *Proc. of ICLR*, 2017.