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

Noah Smith
© 2022

University of Washington
nasmith@cs.washington.edu

Winter 2022

Readings: Eisenstein (2019) 18
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.
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.
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 $Y$:

\[
\text{source} \rightarrow Y \rightarrow \text{channel} \rightarrow X
\]
Aperitif: Noisy Channel Models

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

```
source $\rightarrow$ Y $\rightarrow$ channel $\rightarrow$ 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 $Y$:

\[
\text{source} \rightarrow Y \rightarrow \text{channel} \rightarrow X
\]

- $Y$ 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 $Y$:

$$
\text{source} \rightarrow Y \rightarrow \text{channel} \rightarrow X
$$

- $Y$ 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 $X = x$.

$$
y^* = \arg\max_y p(y \mid x)
= \arg\max_y \frac{p(x \mid y) \cdot p(y)}{p(x)}
= \arg\max_y \underbrace{p(x \mid y)}_{\text{channel model}} \cdot \underbrace{p(y)}_{\text{source model}}
$$
Successful speech recognition requires generating a word sequence that is:

- Faithful to the acoustic input
- Fluent

If we’re mapping acoustics $a$ to word sequences $w$, then:

$$w^* = \arg\max_w \text{Faithfulness}(w; a) + \text{Fluency}(w)$$

Language models can provide a “fluency” score.
Successful speech recognition requires generating a word sequence that is:

- Faithful to the acoustic input
- Fluent

If we’re mapping acoustics \( a \) to word sequences \( w \), then:

\[
\begin{align*}
\mathbf{w}^* &= \arg\max_w \text{Faithfulness}(w; a) + \text{Fluency}(w) \\
&= \arg\max_w \log p(a \mid w) + \log p(w) \\
&= \log p(\mathbf{a} \mid \mathbf{w}) + \log p(\mathbf{w})
\end{align*}
\]

Language models can provide a “fluency” score.
Let $f$ and $e$ be two sequences in French and English, respectively.

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

In a noisy channel machine translation system, we could use this together with source/language model $p(E)$ to “decode” $f$ into an English translation.
Reflection

Where might we find parallel data?
IBM Model 1
(Brown et al., 1993)

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

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

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

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

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

$$= \prod_{i=1}^{m} \frac{1}{\ell + 1} \cdot \theta_{f_i \mid e_{a_i}} = \left( \frac{1}{\ell + 1} \right)^m \prod_{i=1}^{m} \theta_{f_i \mid e_{a_i}}$$
Mr President, Noah's ark was filled not with production factors, but with living creatures.

Noahs Arche war nicht voller Produktionsfaktoren, sondern Geschöpfe.

\[ a = \langle 4, \ldots \rangle \]

\[ p(f, a \mid e, m; \theta) = \frac{1}{17 + 1} \cdot \theta_{\text{Noah's}} \]
Example: $f$ 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.

\[
a = \langle 4, 5, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{17 + 1} \cdot \theta_{\text{Noahs}|\text{Noah's}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Arche}|\text{ark}}
\]
Example: $f$ 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.

\[
a = \langle 4, 5, 6, \ldots \rangle
\]

\[
p(f, a | e, m; \theta) = \frac{1}{17 + 1} \cdot \theta_{\text{Noahs}|\text{Noah's}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Arche}|\text{ark}} \\
\quad \quad \cdot \frac{1}{17 + 1} \cdot \theta_{\text{war}|\text{was}}
\]
Example: $f$ 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.

\[
a = \langle 4, 5, 6, 8, \ldots \rangle
\]

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

\[
\cdot \frac{1}{17 + 1} \cdot \theta_{\text{war|was}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{nicht|not}}
\]
Example: $f$ 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.

\[
a = \langle 4, 5, 6, 8, 7, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{17 + 1} \cdot \theta_{\text{Noahs}\mid \text{Noah's}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Arche}\mid \text{ark}} \\
\frac{1}{17 + 1} \cdot \theta_{\text{war}\mid \text{was}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{nicht}\mid \text{not}} \\
\frac{1}{17 + 1} \cdot \theta_{\text{voller}\mid \text{filled}}
\]
Mr President, Noah's ark was filled not with production factors, but with living creatures.

Noah's Arche war nicht voller Produktionsfaktoren, sondern Geschöpfe.

\[
\mathbf{a} = \langle 4, 5, 6, 8, 7, ?, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{17 + 1} \cdot \theta_{\text{Noahs|Noah's}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Arche|ark}} \\
\quad \cdot \frac{1}{17 + 1} \cdot \theta_{\text{war|was}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{nicht|not}} \\
\quad \cdot \frac{1}{17 + 1} \cdot \theta_{\text{voller|filled}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Productionsfaktoren|?}}
\]
Example: $f$ 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.

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

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

$$\cdot \frac{1}{17 + 1} \cdot \theta_{\text{war}\mid \text{was}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{nicht}\mid \text{not}}$$

$$\cdot \frac{1}{17 + 1} \cdot \theta_{\text{voller}\mid \text{filled}} \cdot \frac{1}{17 + 1} \cdot \theta_{\text{Productionsfaktoren}\mid ?}$$

**Problem:** This alignment isn’t possible with IBM model 1! Each $f_i$ is aligned to at most *one* $e_{a_i}$!
Example: \( f \) 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.

\[
a = \langle 0, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{\text{Mr}|\text{NULL}}
\]
Example: $f$ 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.

\[
a = \langle 0, 0, 0, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{\text{Mr}|\text{NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{President}|\text{NULL}} \\
\cdot \frac{1}{10 + 1} \cdot \theta_{\text{,}|\text{NULL}}
\]
Example: \( f \) 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.

\[ a = \langle 0, 0, 0, 1, \ldots \rangle \]

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

\[
\cdot \frac{1}{10 + 1} \cdot \theta_{\text{,|NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{Noah's|Noahs}}
\]
Example: \( f \) is English

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

\[
a = \langle 0, 0, 0, 1, 2, \ldots \rangle
\]

\[
p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{\text{Mr}|\text{NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{President}|\text{NULL}} \\
\quad \cdot \frac{1}{10 + 1} \cdot \theta_{|\text{NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{Noah’s}|\text{Noahs}} \\
\quad \cdot \frac{1}{10 + 1} \cdot \theta_{\text{ark}|\text{Arche}}
\]
Mr President, Noah's ark was filled not with production factors, but with living creatures.

Noahs Arche war nicht voller Produktionsfaktoren, sondern Geschöpfe.

\[ a = \langle 0, 0, 0, 1, 2, 3, \ldots \rangle \]

\[
p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{Mr|NULL} \cdot \frac{1}{10 + 1} \cdot \theta_{President|NULL} \cdot \frac{1}{10 + 1} \cdot \theta_{Noah's|Noahs} \cdot \frac{1}{10 + 1} \cdot \theta_{was|war}
\]
Mr President, Noah's ark was filled not with production factors, but with living creatures.

Noah's Arche war nicht voller Produktionsfaktoren, sondern Geschöpfe.

\[ a = \langle 0, 0, 0, 1, 2, 3, 5, \ldots \rangle \]

\[ p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{\text{Mr} \mid \text{NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{President} \mid \text{NULL}} \]

\[ \cdot \frac{1}{10 + 1} \cdot \theta_{\text{null}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{Noah's} \mid \text{Noahs}} \]

\[ \cdot \frac{1}{10 + 1} \cdot \theta_{\text{ark} \mid \text{Arche}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{was} \mid \text{war}} \]

\[ \cdot \frac{1}{10 + 1} \cdot \theta_{\text{filled} \mid \text{voller}} \]
Mr President, Noah's ark was filled not with production factors, but with living creatures.

Noahs Arche war nicht voller Produktionsfaktoren, sondern Geschöpfe.

\[ a = \langle 0, 0, 0, 1, 2, 3, 5, 4, \ldots \rangle \]

\[
p(f, a \mid e, m; \theta) = \frac{1}{10 + 1} \cdot \theta_{\text{Mr|NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{President|NULL}} \\
\quad \cdot \frac{1}{10 + 1} \cdot \theta_{\text{|NULL}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{Noah's|Noahs}} \\
\quad \cdot \frac{1}{10 + 1} \cdot \theta_{\text{ark|Arche}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{was|war}} \\
\quad \cdot \frac{1}{10 + 1} \cdot \theta_{\text{filled|voller}} \cdot \frac{1}{10 + 1} \cdot \theta_{\text{not|nicht}}
\]
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: $\langle e_1^{(1)}, f^{(1)}, a^{(1)} \rangle, \ldots, \langle e^{(N)}, f^{(N)}, a^{(N)} \rangle$.

Define:

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

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

$$\hat{\theta}_{f|e} = \frac{\text{count}(e, f)}{\text{count}(e)} = \frac{\sum_{n=1}^{N} \sum_{i : f_{i}^{(n)} = f} \sum_{j : e_{j}^{(n)} = e} q_{n,i}(j)}{\sum_{n=1}^{N} \sum_{i=1}^{m(n)} \sum_{j : e_{j}^{(n)} = e} q_{n,i}(j)}$$
MLE with “Soft” Counts for IBM Model 1

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

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

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

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

$$\hat{\theta}_{f|e} = \frac{\sum_{n=1}^{N} \sum_{i:f_i^{(n)}=f} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}{\sum_{n=1}^{N} \sum_{i=1}^{m^{(n)}} \sum_{j:e_j^{(n)}=e} q_{n,i}(j)}$$
1. Initialize $\theta$ to some arbitrary values.
2. E step: use current $\theta$ 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.

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

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:

\[
\max_{\theta} \prod_{n=1}^{N} p(f^{(n)} | e^{(n)}; \theta) = \max_{\theta} \prod_{n=1}^{N} \sum_{a} p(f^{(n)}, a | e^{(n)}; \theta)
\]

- 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!
Let \( \ell \) and \( m \) be the (known) lengths of \( e \) and \( f \).

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

\begin{itemize}
  \item E.g., \( a_4 = 3 \) means that \( f_4 \) is “aligned” to \( e_3 \).
\end{itemize}

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

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

\[
= \theta_{\text{distortion}}^{a_i \mid i, \ell, m} \cdot \theta_{\text{translation}}^{f_i \mid e_{a_i}}
\]
Variations

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

\[ \theta_{j|i,\ell,m}^{\text{distortion}} \propto \exp(-\lambda \left| \frac{i}{m} - \frac{j}{\ell} \right|) \]

(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    | $p(f | \bar{e})$ |
|--------------|------------|-----------------|
| 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. $A$ captures segmentation of sentences into phrases, alignment between them, and reordering.

$$p(f, a \mid e) = p(a \mid e) \cdot \prod_{i=1}^{|a|} p(f_i \mid e_i)$$
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.

Mary did not slap the green witch

Maria no daba una bofetada a la bruja verde
Extracting Phrases

After inferring word alignments, apply heuristics.
Extracting Phrases

After inferring word alignments, apply heuristics.
Scoring Whole Translations

\[
\text{score}(e, a; f) = \log p(e) + \log p(f, a | e)
\]

Remarks:
- Segmentation, alignment, reordering are all predicted as well (not marginalized).
- This does not factor nicely.
Scoring Whole Translations

\[
\text{score}(e, a; f) = \log p(e) + \log p(f, a \mid e) + \log p(e, a \mid f)
\]

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

$$\text{score}(e, a; f) = \beta_{l.m.} \log p(e) + \beta_{t.m.} \log p(f, a \mid e) + \beta_{r.t.m.} \log p(e, a \mid f)$$

**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

Maria no dio una bofetada a la bruja verde

Mary not give a slap to the witch green

did not slap by hag bawdy

no slap to the green witch

did not give the the witch
Maria no dio una bofetada a la bruja verda

Mary did not give a slap to the witch green

did not slap by hag bawdy

no slap to the green witch

did not give the the witch
Decoding: Example

Maria no dio una bofetada a la bruja verde

Mary did not give a slap to the witch green

did not

slap

to the

witch

green

no

slap

to the

green witch

did not give

the

the witch
Beam Search for Sequential Classifiers
Review from conditional random fields lecture.

Input: $x$ (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 $\hat{y}_{1:i-1}$ scored by $H_{i-1}$:
  - For each $y \in L$, place new hypothesis $\hat{y}_{1:i} \rightarrow H_{i-1}(\hat{y}_{1:i}) + score(x, i, \hat{y}_{1:i-1}, y)$ 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: \( \langle \circ \circ \ldots \circ, \text{""} \rangle \) with score 0
\[ |f| \]

Goal state: \( \langle \bullet \bullet \ldots \bullet, e^* \rangle \) with (approximately) the highest score
\[ |f| \]

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

Maria no dio una bofetada a la bruja verde

Mary did not give a slap to the green witch

Did not slap by hag bawdy

No slap to the green witch

Did not give the witch
Decoding Example

\[
\langle \cdot \circ \circ \circ \circ \circ \circ \circ \circ \circ \circ \circ , \text{“Mary”} \rangle, \quad \log p_{l.m.}(\text{Mary}) + \log p_{t.m.}(\text{Maria} | \text{Mary})
\]
Decoding Example

(Maria did not)

\log p_{l.m.}(\text{Mary did not}) + \log p_{t.m.}(\text{Maria} | \text{Mary})

+ \log p_{t.m.}(\text{no} | \text{did not})
Decoding Example

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

\[ p(E = e \mid f) = p(E = e \mid \text{encode}(f)) \]
\[ = \prod_{j=1}^{\ell} p(e_j \mid e_0, \ldots, e_{j-1}, \text{encode}(f)) \]

The encoding of the source sentence is a deterministic function of the words in that sentence.
**Neural MT Source-Sentence Encoder**

\[ F \text{ is a } d \times m \text{ matrix encoding the source sentence } f \text{ (length } m).\]

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.

\[ s_t = g_{\text{recurr}}(e_{t-1}, \overbrace{F_a}^{\text{"context"}}, s_{t-1}) \]

\[ y_t = g_{\text{output}}(s_t) \]

\[ p(E_t = v | e_1, \ldots, e_{t-1}, f) = [y_t]_v \]

(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 \(g_{\text{recurr}}\).
Neural MT Decoder
Neural MT Decoder
Neural MT Decoder

I'd like a beer STOP
$a_1^\top$
I'd like a beer.
Neural MT Decoder

I'd like a beer STOP

\[
\begin{bmatrix}
0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\circ \circ \circ \circ \\
\circ \circ \circ \circ \\
\hat{a}_1^\top & \hat{a}_2^\top
\end{bmatrix}
\]
Neural MT Decoder

I’d like a beer

[ ]

[ ]

[ ]

0

\[
\begin{bmatrix}
\text{[● ● ● ● ●]} & \text{[● ● ● ●]} & \text{[● ● ● ● ●]}
\end{bmatrix}
\]

\[
a_1^T \quad a_2^T \quad a_3^T
\]
Neural MT Decoder

I’d like a beer STOP

[ ] [ ] [ ]

a1 a2 a3 ⊤ ⊤ ⊤
Neural MT Decoder

I’d like a beer STOP

0

a_1 a_2 a_3 a_4
Neural MT Decoder

I’d like a beer

\[
\begin{bmatrix}
\bullet & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ \\
\bullet & \circ & \circ & \circ
\end{bmatrix}
\]

\[
a_1^T a_2^T a_3^T a_4^T
\]
Neural MT Decoder

I'd like a beer

\[
\begin{bmatrix}
\end{bmatrix}
\]

\[
a_1^T a_2^T a_3^T a_4^T a_5^T
\]

[ ] [ ] [ ] [ ] [ ]
Neural MT Decoder

I’d like a beer

\[
\begin{bmatrix}
\begin{array}{cccccc}
\odot & \odot & \odot & \odot & \odot & \odot \\
\odot & \odot & \odot & \odot & \odot & \odot \\
\odot & \odot & \odot & \odot & \odot & \odot \\
\odot & \odot & \odot & \odot & \odot & \odot \\
\odot & \odot & \odot & \odot & \odot & \odot \\
\odot & \odot & \odot & \odot & \odot & \odot \\
\end{array}
\end{bmatrix}
\]

\[
\begin{bmatrix}
a_1^T \\
a_2^T \\
a_3^T \\
a_4^T \\
a_5^T \\
\end{bmatrix}
\]
Neural MT Decoder

I'd like a beer

0

\[
\begin{bmatrix}
\vdots
\end{bmatrix}
\]

\[
\begin{bmatrix}
\vdots
\end{bmatrix}
\]

\[
\begin{bmatrix}
\vdots
\end{bmatrix}
\]

a_1^T a_2^T a_3^T a_4^T a_5^T
Computing “Attention”

Let $V_{s_{t-1}}$ be the “expected” input embedding for timestep $t$.
(Parameters: $V$.)

Attention is $a_t = \text{softmax} \left( F^\top V_{s_{t-1}} \right)$.

Context is $F_{a_t}$, 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

\[
\log p(e \mid \text{encode}(f)) = \sum_{i=1}^{m} \log p(e_i \mid e_{0:i-1}, \text{encode}(f))
\]

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

Decoding typically uses beam search.
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 $p(e) \times p(f \mid e)$ vs. “direct” model $p(e \mid f)$
- Alignment as a discrete random variable vs. attention as a deterministic, differentiable function
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 (Stanovský et al., 2019).
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 III


