Lecture 9: Sequence-to-Sequence Models

André Martins, Francisco Melo, Mário Figueiredo

Deep Learning Course, Winter 2022-2023

Today's Roadmap

Lecture 8 focused on sequence tagging and sequence generation. Today we focus on sequence-to-sequence models.

- Machine translation
- Sequence vector representation
- Encoder-decoder architecture
- Sequence matrix representation
- Attention mechanism
- Encoder-decoder with attention

Sequence-to-Sequence

Sequence-to-sequence models map a source sequence (of arbitrary length) into a target sequence (also of arbitrary length)

This differs from sequence tagging, where the two sequences are of the same length

Example: Machine Translation

Goal: translate a **source sentence** x in one language into a **target sentence** y in another language.

Example (Portuguese to English):

x: *"A ilha de Utopia tem 200 milhas de diâmetro na parte central."*

$$\downarrow$$

y: *"The island of Utopia is two hundred miles across in the middle part."*

Outline

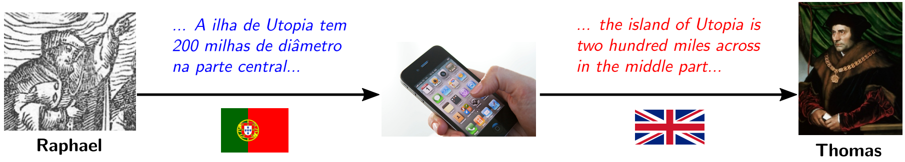
1950s: Early Machine Translation

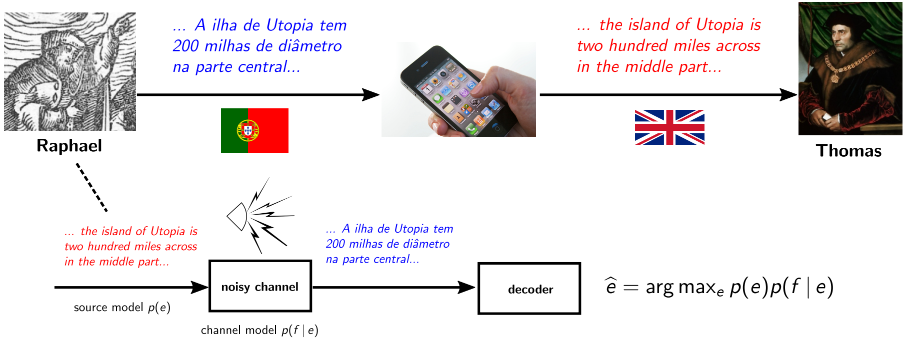
(Source: `https://youtu.be/K-HfpsHPmvw`)

- MT research began in early 1950s
- Mostly Russian-English (motivated by the Cold War!)
- Systems were mostly rule-based, using a bilingual dictionary

Noisy Channel Model (Shannon and Weaver, 1949)

"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.' "


... A ilha de Utopia tem
200 milhas de diâmetro
na parte central...
... the island of Utopia is
two hundred miles across
in the middle part...
Raphael
Thomas



A very simple model: builds a generative story that works "backwards" (flips source and target, via Bayes law)

Yet, the dominant paradigm in MT for several decades (until 2014)

In 2014: neural machine translation (later)

1990s-2010s: Statistical Machine Translation

Goal: find the "best" (most probable) English sentence y, given the Russian sentence x

$$\widehat{y} = \arg\max_{y} \mathbb{P}(y \mid x)$$

1990s-2010s: Statistical Machine Translation

Goal: find the "best" (most probable) English sentence y, given the Russian sentence x

$$\widehat{y} = \arg\max_{y} \mathbb{P}(y \mid x)$$

Key idea: use Bayes' law to break this down into two components:

$$\widehat{y} = \arg\max_{y} \frac{\mathbb{P}(x \mid y)\, \mathbb{P}(y)}{\mathbb{P}(x)} = \arg\max_{y} \mathbb{P}(x \mid y)\, \mathbb{P}(y)$$

1990s-2010s: Statistical Machine Translation

Goal: find the "best" (most probable) English sentence y, given the Russian sentence x

$$\widehat{y} = \arg\max_{y} \mathbb{P}(y \mid x)$$

Key idea: use Bayes' law to break this down into two components:

$$\widehat{y} = \arg\max_{y} \frac{\mathbb{P}(x \mid y)\, \mathbb{P}(y)}{\mathbb{P}(x)} = \arg\max_{y} \mathbb{P}(x \mid y)\, \mathbb{P}(y)$$

- **Translation model:** models how words/phrases are translated (learnt from **parallel** data)
- **Language model:** models how to generate fluent English (learn from **monolingual** data)

How to Learn the Language Model?

- Need large amounts of monolingual data
  (easy to get for most languages).

How to Learn the Language Model?

- Need large amounts of monolingual data
  (easy to get for most languages).
- How to learn a language model from these data?

How to Learn the Language Model?

- Need large amounts of monolingual data
  (easy to get for most languages).
- How to learn a language model from these data?
- We covered language models in previous lectures:
  - Markov models (maybe with *smoothing*)
  - Neural language models
  - ...

How to Learn the Language Model?

- Need large amounts of monolingual data
  (easy to get for most languages).

- How to learn a language model from these data?

- We covered language models in previous lectures:

  - Markov models (maybe with *smoothing*)

  - Neural language models

  - ...

- Pick your favorite!

How to Learn the Translation Model?

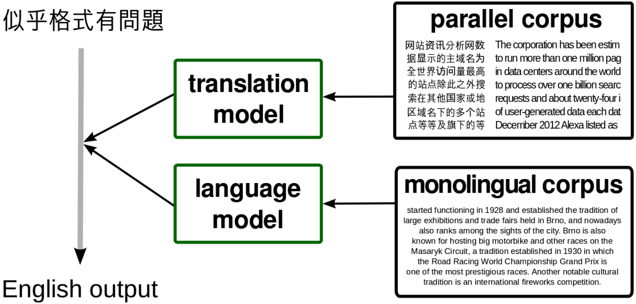
Need large amounts of parallel data!

(*e.g.*, pairs of human-translated Chinese-English sentence pairs.)

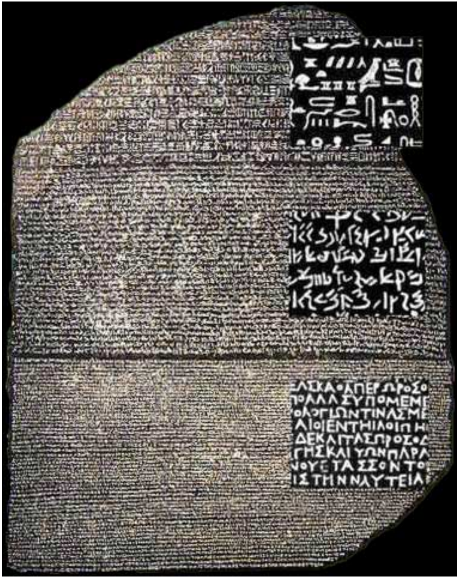
How to Learn the Translation Model?

Need large amounts of parallel data!

(*e.g.*, pairs of human-translated Chinese-English sentence pairs.)



Rosetta Stone



- (Re-)discovered in 1799 near Alexandria
- Parallel corpora: ancient Egyptian, demotic Egyptian, ancient Greek

Europarl

- Proceedings from the European parliament sessions, translated into all EU official languages
- Around 1M parallel sentences for some language pairs
- Other corpora: Hansard, MultiUN, News Commentary, Wikipedia, OpenSubtitles, Paracrawl, ...

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- Assume we have enough parallel training data.
- Break it down further: consider instead

$$\mathbb{P}(x, \boldsymbol{a} \mid y),$$

where $\boldsymbol{a}$ are word alignments, i.e., word-level correspondences between sentence x and sentence y

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- Assume we have enough parallel training data.
- Break it down further: consider instead

$$\mathbb{P}(x, \boldsymbol{a} \mid y),$$

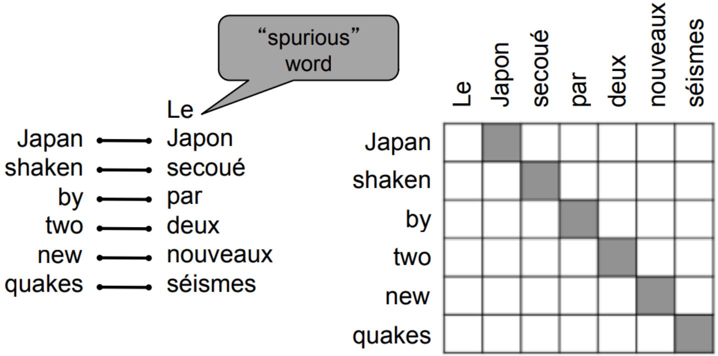
  where $\boldsymbol{a}$ are word alignments, i.e., word-level correspondences between sentence x and sentence y

- Word alignments are generally a latent/missing variable at training time, and need to be marginalized over at test time,

$$\mathbb{P}(x \mid y) = \sum_{\boldsymbol{a}} \mathbb{P}(x, \boldsymbol{a} \mid y),$$

Word Alignments

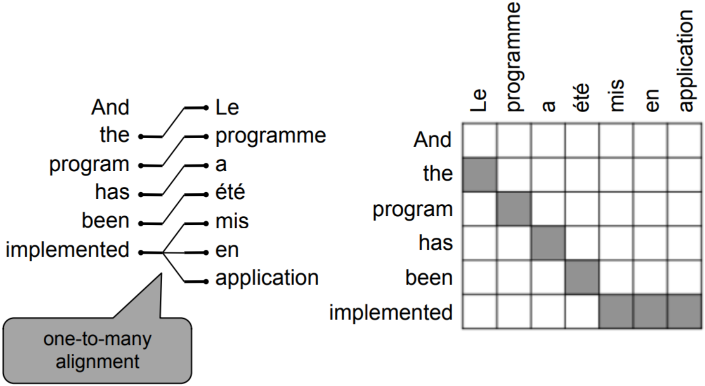
Example for English-French:



Some words may be unaligned (no counterpart in the other language)!

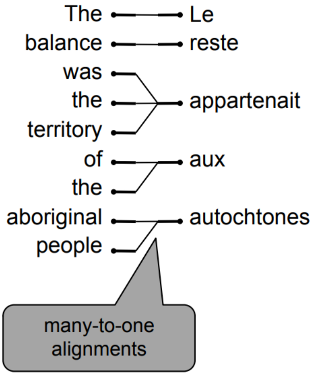
Word Alignments

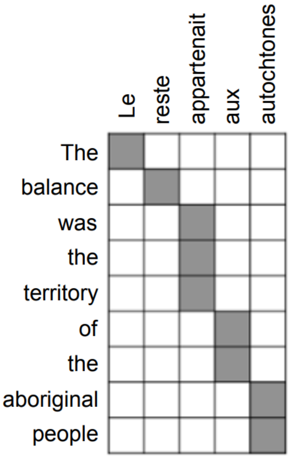
Alignment can be one-to-many (word fertility):



Word Alignments

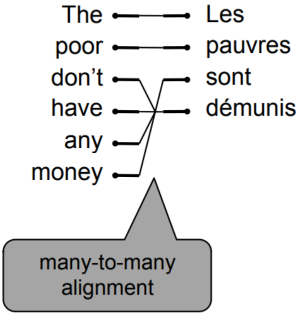
Alignment can be many-to-one:

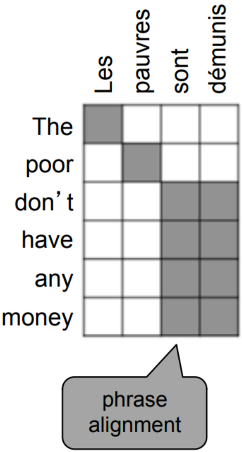




Word Alignments

Alignment can be many-to-many (phrase-level): phrase-based MT:





1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- ...assuming we have enough parallel training data.

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- ...assuming we have enough parallel training data.
- Break it down further: consider instead

$$\mathbb{P}(x, \boldsymbol{a} \mid y).$$

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- ...assuming we have enough parallel training data.
- Break it down further: consider instead

$$\mathbb{P}(x, \boldsymbol{a} \mid y).$$

- Learn $\mathbb{P}(x, \boldsymbol{a} \mid y)$ as a combination of several factors:
  - Probability of particular words aligning (co-occurrence, relative position, etc.)
  - Probability of words having a particular fertility
  - ...

1990s: IBM Models for Statistical MT

- How to learn the translation model $\mathbb{P}(x \mid y)$?
- ...assuming we have enough parallel training data.
- Break it down further: consider instead

$$\mathbb{P}(x, \boldsymbol{a} \mid y).$$

- Learn $\mathbb{P}(x, \boldsymbol{a} \mid y)$ as a combination of several factors:
  - Probability of particular words aligning (co-occurrence, relative position, etc.)
  - Probability of words having a particular fertility
  - ...
- This has lead to IBM models 1, 2, 3, 4, ...

1990s: IBM Models for Statistical MT

- To search the best translation, we need to solve

$$\widehat{y} = \arg\max_{y} \sum_{\boldsymbol{a}} \mathbb{P}(x, \boldsymbol{a} \mid y)\, \mathbb{P}(y),$$

combining the translation and language models.

1990s: IBM Models for Statistical MT

- To search the best translation, we need to solve

$$\widehat{y} = \arg\max_{y} \sum_{\boldsymbol{a}} \mathbb{P}(x, \boldsymbol{a} \mid y)\, \mathbb{P}(y),$$

combining the translation and language models.

- Enumerating all possible hypothesis and alignments is intractable.

1990s: IBM Models for Statistical MT

- To search the best translation, we need to solve

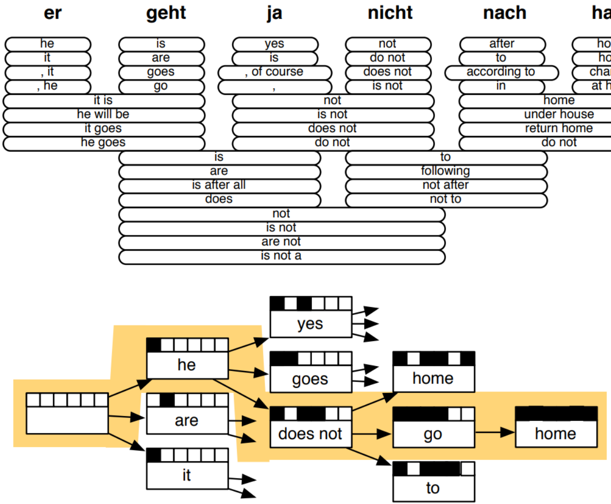
$$\widehat{y} = \arg\max_{y} \sum_{\boldsymbol{a}} \mathbb{P}(x, \boldsymbol{a} \mid y)\, \mathbb{P}(y),$$

  combining the translation and language models.

- Enumerating all possible hypothesis and alignments is intractable.

- Typical approach: heuristic search to gradually build the translation, discarding hypotheses that are too low probability.

Searching for the Best Translation



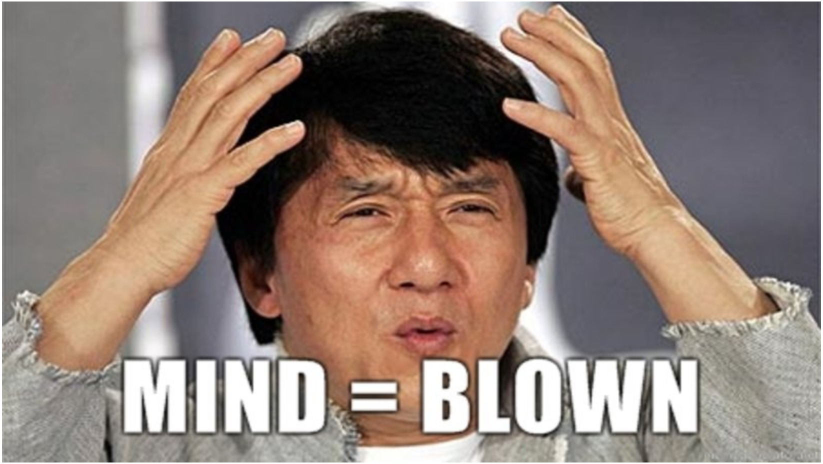
(Slide credit: `https://web.stanford.edu/class/cs224n/lectures/lecture10.pdf`)

Summarizing: Statistical Machine Translation

We only glimpsed the tip of the iceberg: SMT is (was?) a huge research field.

- The best systems are extremely complex;
- Usually a big pipeline, with many separately-designed subcomponents (translation and language model are only two examples);
- Lots of feature engineering;
- System design is very language-dependent;
- Requires compiling and maintaining resources (e.g., phrase tables);
- Models are storage/memory hungry;
- Lots of human effort to maintain.

2014: Neural Machine Translation



Outline

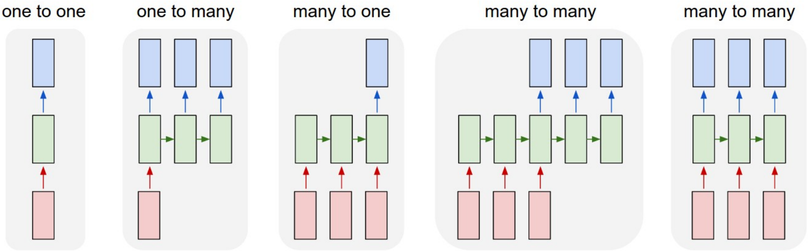
What is Neural Machine Translation (NMT)?

- NMT = MT with a single neural network
- End-to-end training with parallel data (no more complex pipelines!)
- The underlying architecture is an encoder-decoder (also called a **sequence-to-sequence model**)
- In fact, NMT is also statistical; however, historically, "statistical MT" refers to non-neural models, and NMT to NN-based models.

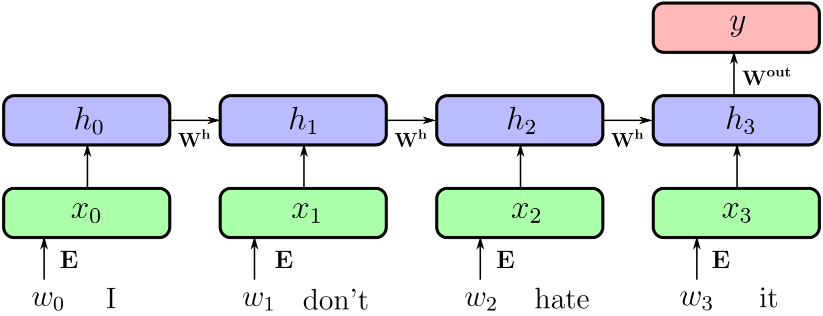
Outline

Recap: Recurrent Neural Networks

Lecture 9 covered RNNs and showed they can have several uses...

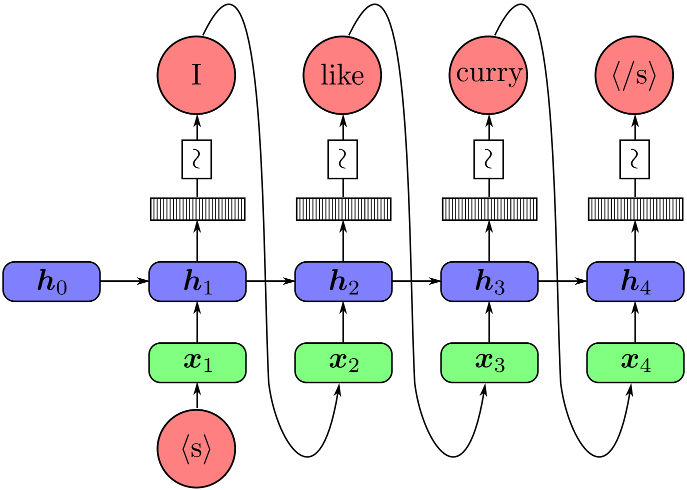


Recap: RNNs for Pooled Classification



(Slide credit: Ollion & Grisel)

Recap: Auto-Regressive RNNs for Sequence Generation



Sequence-to-Sequence Learning (Cho et al., 2014; Sutskever et al., 2014)

- Can we put the two things together?

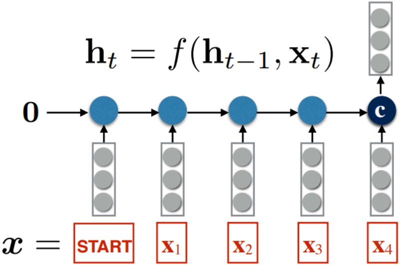
Sequence-to-Sequence Learning (Cho et al., 2014; Sutskever et al., 2014)

- Can we put the two things together?

- Yes!

  1. Encoder RNN encodes source sentence, generating a vector state
  2. Decoder RNN generates target sentence conditioned on vector state.

Encode a Sequence as a Vector



(Slide credit: Chris Dyer)

What is a vector representation of a sequence $\boldsymbol{x}$?

$$\boldsymbol{c} = \text{RNN}(\boldsymbol{x})$$

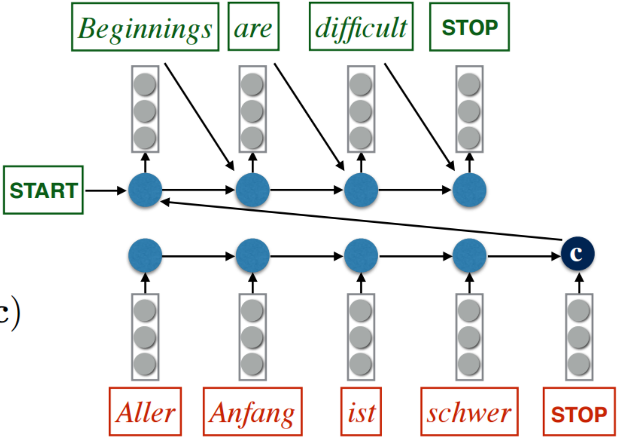
What is the probability of a sequence $\boldsymbol{y} \mid \boldsymbol{x}$?

$$\boldsymbol{y} \mid \boldsymbol{x} \sim \text{RNNLM}(\boldsymbol{c})$$

Encoder-Decoder Architecture

$$\mathbf{c} = \mathrm{RNN}(\boldsymbol{x})$$

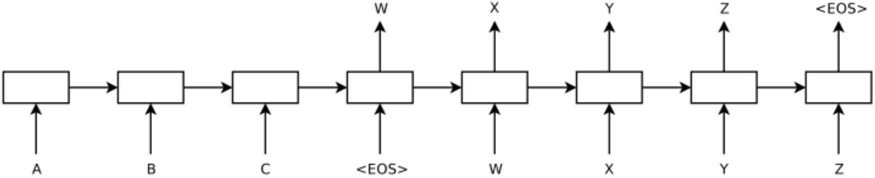
$$\boldsymbol{y} \mid \mathbf{c} \sim \mathrm{RNNLM}(\mathbf{c})$$



(Slide credit: Chris Dyer)

Encoder-Decoder Architecture

Another way of depicting it (from Sutskever et al. (2014)):



Some Problems

- If the source sentence is long, the encoder may forget the initial words and the translation will be degraded
  - Cheap solution: reverse the source sentence.

Some Problems

- If the source sentence is long, the encoder may forget the initial words and the translation will be degraded
  - Cheap solution: reverse the source sentence.

- The decoder does greedy search—this leads to error propagation
  - Solution: beam search.

Beam Search

Ideally we want to find the target sentence y that maximizes

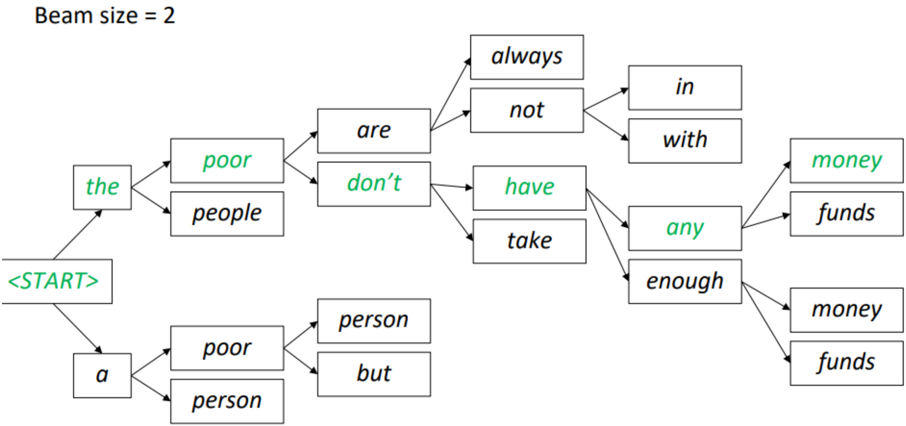
$$\hat{y} = \arg\max_{y} \mathbb{P}(y \mid x) = \arg\max_{y_1,\ldots,y_L} \mathbb{P}(y \mid x) = \prod_{i=1}^{L} \mathbb{P}(y_i \mid y_{1:i-1}, x)$$

Enumerating all y is intractable!

Beam Search:

- Approximate search strategy
- At each decoder step, keep track of the k most probable partial translations; discard the rest
- k is the beam size
- For $k = 1$, we have greedy search.

Beam Search



(Slide credits: Chris Manning)

Beam Search

- A little better than strictly greedy search, but still greedy
- Linear runtime as a function of beam size k: speed/accuracy trade-off
- In practice: beam sizes $\sim$ 4–12

Some Additional Tricks

From Sutskever et al. (2014):

- Deep LSTMs
- Reversing the source sentence

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Some Additional Tricks

From Sutskever et al. (2014):

- Deep LSTMs
- Reversing the source sentence

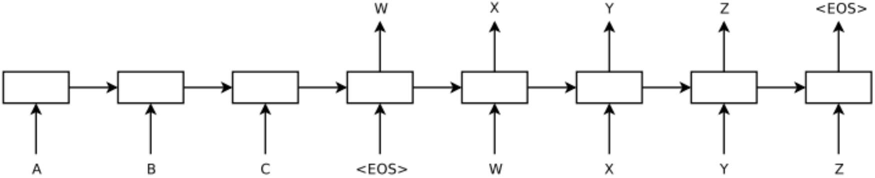
Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

At run time:

- Beam search;
- Ensembling: combine N independently trained models and obtaining a "consensus" (always helps!).

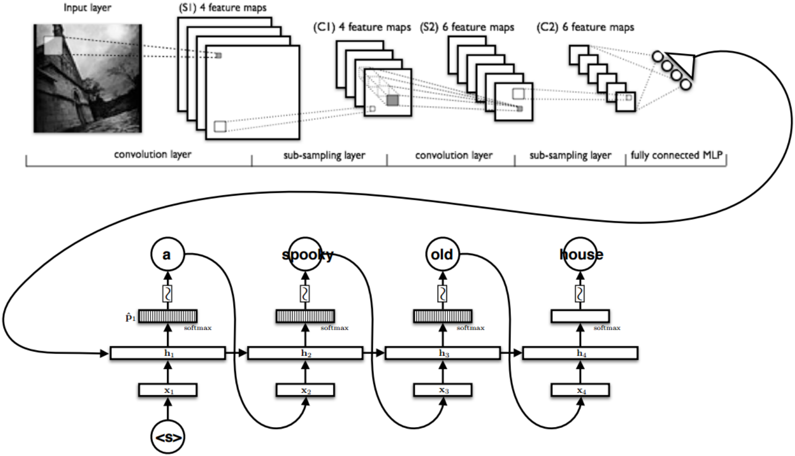
End-to-End Neural Machine Translation

- Previous statistical machine translation models were complicated pipelines (word alignments $\rightarrow$ phrase table extraction $\rightarrow$ language model $\rightarrow$ decoding a phrase lattice);
- Alternative: end-to-end NMT uses a simple encoder-decoder;
- Doesn't quite work yet, but gets close to top performance



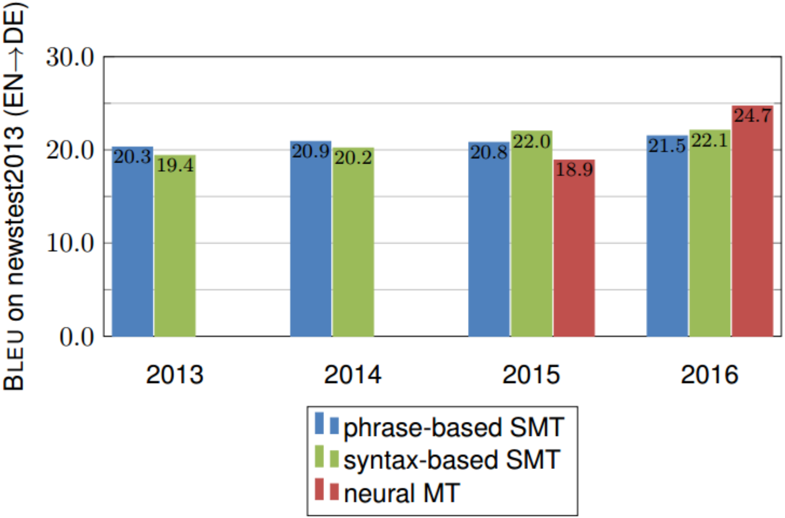
Caption Generation

Works for image inputs too!



(Slide credit: Chris Dyer)

Progress in Machine Translation



Slide credit: Rico Sennrich

NMT: A Success Story

- Neural MT went from a fringe research activity in 2014 to the leading standard method in 2016
  - 2014: First seq2seq paper published
  - 2016: Google Translate switches from SMT to NMT
- This is amazing!
- SMT systems, built by hundreds of engineers over many years, outperformed by NMT trained by a handful of engineers in a few months.

So Is Machine Translation Solved?

No!

Many difficulties and open research fronts remain:

- Out-of-vocabulary words
- Domain mismatch between train and test data
- Low-resource language pairs
- Specific domains (medical, scientific, legal, ...)
- Maintaining context over longer text (coming next!)

Limitations

A possible conceptual problem:

- Sentences have unbounded lengths
- Vectors have finite capacity

"You can't cram the meaning of a whole %&$# sentence into a single $&# vector!" (Ray Mooney)

A possible practical problem:

- Distance between "translations" and their sources are large—can LSTMs help?

Outline

1. Statistical Machine Translation

2. Neural Machine Translation

    Encoder-Decoder Architecture

    Encoder-Decoder with Attention

3. Conclusions

Encode Sentences as Matrices, Not Vectors

Problem with the fixed-size vector model:

- Sentences are of different sizes but vectors are of the same size
- Bottleneck problem: a single vector represents the whole source sentence!

Encode Sentences as Matrices, Not Vectors

Problem with the fixed-size vector model:

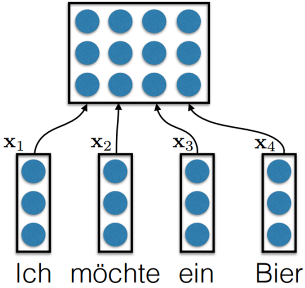
- Sentences are of different sizes but vectors are of the same size
- Bottleneck problem: a single vector represents the whole source sentence!

Solution: use matrices instead!

- Fixed number of rows, but number of columns depends on the number of words
- Then, before generating each word in the decoder, use an attention mechanism to condition on the relevant source words only

How to Encode a Sentence as a Matrix?

First shot: define the sentence words' vectors as the columns



(Image credit: Chris Dyer)

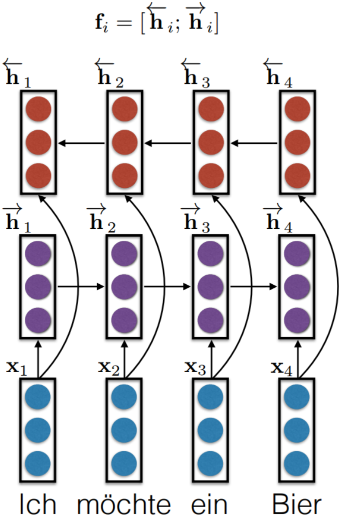
- Not very effective: word vectors carry no context information

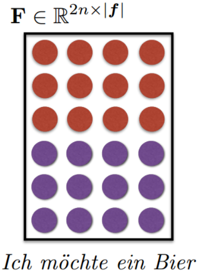
How to Encode a Sentence as a Matrix?

Other strategies:

- Convolutional NNs (Kalchbrenner et al., 2014) capture context;
- **Typical choice:** bidirectional LSTMs (Bahdanau et al., 2015);
- Later: transformer networks (Vaswani et al., 2017).

Bidirectional LSTM Encoder





(Slide credit: Chris Dyer)

Generation from Matrices

- We now have a matrix $\boldsymbol{F}$ representing the input.
- How to generate from it?
- **Answer:** use attention! (Bahdanau et al., 2015)
- Attention is the neural counterpart of word alignments.

Generation from Matrices with Attention

- Generate the output sentence word by word using an RNN;

Generation from Matrices with Attention

- Generate the output sentence word by word using an RNN;
- At each output position t, the RNN receives two inputs:
  - a fixed-size vector embedding of the previous output symbol y_{t-1}
  - a fixed-size vector encoding a "view" of the input matrix $\boldsymbol{F}$, via a weighted sum of its columns (i.e., words): $\boldsymbol{F}\boldsymbol{a}_t$

Generation from Matrices with Attention

- Generate the output sentence word by word using an RNN;
- At each output position t, the RNN receives two inputs:
  - a fixed-size vector embedding of the previous output symbol y_{t-1}
  - a fixed-size vector encoding a "view" of the input matrix $\boldsymbol{F}$, via a weighted sum of its columns (i.e., words): $\boldsymbol{F}\boldsymbol{a}_t$
- The input columns weighting at each time-step ($\boldsymbol{a}_t$) is called the attention distribution.

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

1. Compute "similarity" with each of the source words:

$$z_{t,i} = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b}), \quad \text{for } i = 1, ..., L$$

   where $\boldsymbol{h}_i$ is the i-th column of $\boldsymbol{F}$ (representation of the i-th source word) and $\boldsymbol{v}$, $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{b}$ are model parameters

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

1. Compute "similarity" with each of the source words:

$$z_{t,i} = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b}), \quad \text{for } i = 1, ..., L$$

   where $\boldsymbol{h}_i$ is the i-th column of $\boldsymbol{F}$ (representation of the i-th source word) and $\boldsymbol{v}$, $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{b}$ are model parameters

2. From $\boldsymbol{z}_t = (z_{t,1}, \ldots, z_{t,L})$ compute attention $\boldsymbol{a}_t = \text{softmax}(\boldsymbol{z}_t)$

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

1. Compute "similarity" with each of the source words:

$$z_{t,i} = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b}), \quad \text{for } i = 1, ..., L$$

   where $\boldsymbol{h}_i$ is the i-th column of $\boldsymbol{F}$ (representation of the i-th source word) and $\boldsymbol{v}$, $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{b}$ are model parameters

2. From $\boldsymbol{z}_t = (z_{t,1}, \ldots, z_{t,L})$ compute attention $\boldsymbol{a}_t = \text{softmax}(\boldsymbol{z}_t)$

3. Use attention to compute input conditioning state $\boldsymbol{c}_t = \boldsymbol{F}\boldsymbol{a}_t$

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

1. Compute "similarity" with each of the source words:

$$z_{t,i} = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b}), \quad \text{for } i = 1, ..., L$$

   where $\boldsymbol{h}_i$ is the i-th column of $\boldsymbol{F}$ (representation of the i-th source word) and $\boldsymbol{v}$, $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{b}$ are model parameters

2. From $\boldsymbol{z}_t = (z_{t,1}, \ldots, z_{t,L})$ compute attention $\boldsymbol{a}_t = \text{softmax}(\boldsymbol{z}_t)$
3. Use attention to compute input conditioning state $\boldsymbol{c}_t = \boldsymbol{F}\boldsymbol{a}_t$
4. Update RNN state $\boldsymbol{s_t}$ based on $\boldsymbol{s}_{t-1}, y_{t-1}, \boldsymbol{c}_t$

Attention Mechanism (Bahdanau et al., 2015)

Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots$ be the states produced by the decoder RNN

When predicting the t-th target word:

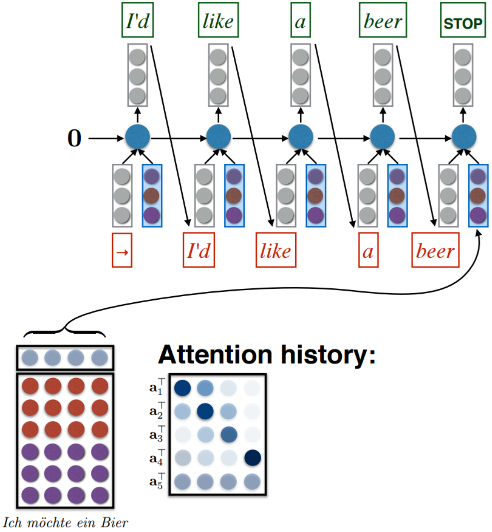
1. Compute "similarity" with each of the source words:

$$z_{t,i} = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b}), \quad \text{for } i = 1, ..., L$$

   where $\boldsymbol{h}_i$ is the i-th column of $\boldsymbol{F}$ (representation of the i-th source word) and $\boldsymbol{v}$, $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{b}$ are model parameters

2. From $\boldsymbol{z}_t = (z_{t,1}, \ldots, z_{t,L})$ compute attention $\boldsymbol{a}_t = \mathrm{softmax}(\boldsymbol{z}_t)$
3. Use attention to compute input conditioning state $\boldsymbol{c}_t = \boldsymbol{F}\boldsymbol{a}_t$
4. Update RNN state $\boldsymbol{s_t}$ based on $\boldsymbol{s}_{t-1}, y_{t-1}, \boldsymbol{c}_t$
5. Predict $y_t \sim p(y_t \mid \boldsymbol{s}_t)$

Encoder-Decoder with Attention



(Slide credit: Chris Dyer)

Putting It All Together

obtain input matrix $\boldsymbol{F}$ with a bidirectional LSTM
$t = 0$, $y_0 = \text{START}$ (the start symbol)
$\boldsymbol{s}_0 = \boldsymbol{w}$ (learned initial state)
repeat
  $t = t + 1$
  $\boldsymbol{z}_t = \boldsymbol{v}^T \boldsymbol{g}(\boldsymbol{W}\boldsymbol{F} + \boldsymbol{U}\boldsymbol{s}_{t-1} + \boldsymbol{b})$
  compute attention $\boldsymbol{a}_t = \text{softmax}(\boldsymbol{z}_t)$
  compute input conditioning state $\boldsymbol{c}_t = \boldsymbol{F}\boldsymbol{a}_t$
  $\boldsymbol{s}_t = \text{RNN}(\boldsymbol{s}_{t-1}, [\boldsymbol{E}(y_{t-1}), \boldsymbol{c}_t])$
  $y_t \sim \text{softmax}(\boldsymbol{P}\boldsymbol{s}_t + \boldsymbol{b})$
until $y_t \neq \text{STOP}$

Notice that each y_t is conditioned (depends on) $y_{t-1}, \ldots, y_1$ and $\boldsymbol{x}$.

Attention Mechanisms

- Attention is closely related to pooling (used in CNNs and other architectures); it's a form of weighted pooling;

Attention Mechanisms

- Attention is closely related to pooling (used in CNNs and other architectures); it's a form of weighted pooling;
- Attention in MT plays a similar role as alignment, but in a "soft" version, instead of "hard" alignment

Attention Mechanisms

- Attention is closely related to pooling (used in CNNs and other architectures); it's a form of weighted pooling;
- Attention in MT plays a similar role as alignment, but in a "soft" version, instead of "hard" alignment
- Bahdanau et al. (2015)'s model has no bias in favor of diagonals, short jumps, fertility, etc.

Attention Mechanisms

- Attention is closely related to pooling (used in CNNs and other architectures); it's a form of weighted pooling;
- Attention in MT plays a similar role as alignment, but in a "soft" version, instead of "hard" alignment
- Bahdanau et al. (2015)'s model has no bias in favor of diagonals, short jumps, fertility, etc.
- Some following work adds some "structural" biases (Luong et al., 2015; Cohn et al., 2016)

Attention Mechanisms

- Attention is closely related to pooling (used in CNNs and other architectures); it's a form of weighted pooling;
- Attention in MT plays a similar role as alignment, but in a "soft" version, instead of "hard" alignment
- Bahdanau et al. (2015)'s model has no bias in favor of diagonals, short jumps, fertility, etc.
- Some following work adds some "structural" biases (Luong et al., 2015; Cohn et al., 2016)
- Other works constrains the amount of attention each word can receive (based on its fertility): Malaviya et al. (2018).

Attention is Great!

- Attention significantly improves NMT performance!

Attention is Great!

- Attention significantly improves NMT performance!
- It is very useful to allow the decoder to focus on parts of the source;

Attention is Great!

- Attention significantly improves NMT performance!
- It is very useful to allow the decoder to focus on parts of the source;
- Attention solves the bottleneck problem, by allowing the decoder to look directly at the source:

Attention is Great!

- Attention significantly improves NMT performance!
- It is very useful to allow the decoder to focus on parts of the source;
- Attention solves the bottleneck problem, by allowing the decoder to look directly at the source:
- Attention helps with vanishing gradient problem (provides shortcut to faraway states);

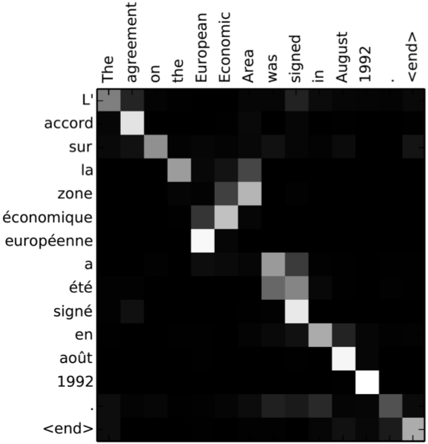
Attention is Great!

- Attention significantly improves NMT performance!
- It is very useful to allow the decoder to focus on parts of the source;
- Attention solves the bottleneck problem, by allowing the decoder to look directly at the source:
- Attention helps with vanishing gradient problem (provides shortcut to faraway states);
- Attention allows for some interpretability (shows what the the decoder was focusing on when producing each output symbol)

Attention is Great!

- Attention significantly improves NMT performance!
- It is very useful to allow the decoder to focus on parts of the source;
- Attention solves the bottleneck problem, by allowing the decoder to look directly at the source:
- Attention helps with vanishing gradient problem (provides shortcut to faraway states);
- Attention allows for some interpretability (shows what the the decoder was focusing on when producing each output symbol)
- This is good because we never explicitly trained a word aligner; the network learns it by itself!

Attention Map



Dzmitry Bahdanau, KyungHuyn Cho, and Yoshua Bengio. ***Neural Machine Translation by Jointly Learning to Translate and Align****. ICLR'15.*

Example: Machine Translation

Some positive examples where NMT has impressive performance:

Source	When asked about this, an official of the American administration replied: "The United States is not conducting electronic surveillance aimed at offices of the World Bank and IMF in Washington."	
PBMT	Interrogé à ce sujet, un responsable de l'administration américaine a répondu : "Les Etats-Unis n'est pas effectuer une surveillance électronique destiné aux bureaux de la Banque mondiale et du FMI à Washington".	3.0
GNMT	Interrogé à ce sujet, un fonctionnaire de l'administration américaine a répondu: "Les États-Unis n'effectuent pas de surveillance électronique à l'intention des bureaux de la Banque mondiale et du FMI à Washington".	6.0
Human	Interrogé sur le sujet, un responsable de l'administration américaine a répondu: "les Etats-Unis ne mènent pas de surveillance électronique visant les sièges de la Banque mondiale et du FMI à Washington".	6.0
Source	Martin told CNN that he asked Daley whether his then-boss knew about the potential shuffle.	
PBMT	Martin a déclaré à CNN qu'il a demandé Daley si son patron de l'époque connaissaient le potentiel remaniement ministériel.	2.0
GNMT	Martin a dit à CNN qu'il avait demandé à Daley si son patron d'alors était au courant du remaniement potentiel.	6.0
Human	Martin a dit sur CNN qu'il avait demandé à Daley si son patron d'alors était au courant du remaniement éventuel.	5.0

(From Wu et al. (2016))

Example: Machine Translation

... But also some negative examples:

- Dropping source words (lack of attention)
- Repeated source words (too much attention)

Source:	1922 in Wien geboren, studierte Mang während und nach dem Zweiten Weltkrieg Architektur an der Technischen Hochschule in Wien bei Friedrich Lehmann.
Human:	Born in Vienna in 1922, Meng studied architecture at the Technical University in Vienna under Friedrich Lehmann *during and after the second World War.*
NMT:	*Born in Vienna in 1922, Mang studied architecture at the Technical College in Vienna with Friedrich Lehmann.
Source:	Es ist schon komisch, wie dies immer wieder zu dieser Jahreszeit auftaucht.
Human:	It's funny how this always comes up at *this time* of year.
NMT:	*It's funny how **this time** to come back to **this time** of year.

Example: Caption Generation

Attention over images:

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

(Slide credit to Yoshua Bengio)

A More Extreme Example...

(Slide credit to Dhruv Batra)

Attention and Memories

Attention is used in several problems, sometimes under different names:

- image caption generation (Xu et al., 2015)
- speech recognition (Chorowski et al., 2015)
- memory networks for reading comprehension (Sukhbaatar et al., 2015; Hermann et al., 2015)
- neural Turing machines and other "differentiable computers" (Graves et al., 2014; Grefenstette et al., 2015)

Other Attentions

- Can we have more interpretable attention? Closer to hard alignments?

Other Attentions

- Can we have more interpretable attention? Closer to hard alignments?
- Can we upper bound how much attention a word receives? This may prevent a common problem in neural MT, repetitions

Other Attentions

- Can we have more interpretable attention? Closer to hard alignments?
- Can we upper bound how much attention a word receives? This may prevent a common problem in neural MT, repetitions
- Sparse attention via sparsemax (Martins and Astudillo, 2016)

Other Attentions

- Can we have more interpretable attention? Closer to hard alignments?
- Can we upper bound how much attention a word receives? This may prevent a common problem in neural MT, repetitions
- Sparse attention via sparsemax (Martins and Astudillo, 2016)
- Constrained attention with constrained softmax/sparsemax (Malaviya et al., 2018)

Outline

Conclusions

- Machine translation is a key problem in AI since the 1950s
- Neural machine translation with sequence-to-sequence models was a breakthrough
- Representing a full sentence with a single vector is a bottleneck
- Attention mechanisms allow focusing on different parts of the input and solve the bottleneck problem
- Other applications beyond MT: speech recognition, image captioning, etc.

Thank you!

Questions?

References I

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In *Proc. of Empirical Methods in Natural Language Processing*.

Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. (2015). Attention-based Models for Speech Recognition. In *Advances in Neural Information Processing Systems*, pages 577–585.

Cohn, T., Hoang, C. D. V., Vymolova, E., Yao, K., Dyer, C., and Haffari, G. (2016). Incorporating structural alignment biases into an attentional neural translation model. *arXiv preprint arXiv:1601.01085*.

Graves, A., Wayne, G., and Danihelka, I. (2014). Neural Turing Machines. *arXiv preprint arXiv:1410.5401*.

Grefenstette, E., Hermann, K. M., Suleyman, M., and Blunsom, P. (2015). Learning to Transduce with Unbounded Memory. In *Advances in Neural Information Processing Systems*, pages 1819–1827.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Malaviya, C., Ferreira, P., and Martins, A. F. T. (2018). Sparse and constrained attention for neural machine translation. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*.

Martins, A. F. T. and Astudillo, R. (2016). From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *Proc. of the International Conference on Machine Learning*.

Shannon, C. E. and Weaver, W. (1949). The mathematical theory of communication. *Urbana: University of Illinois Press*, 29.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). End-to-End Memory Networks. In *Advances in Neural Information Processing Systems*, pages 2431–2439.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

References II

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *International Conference on Machine Learning*.