Neural Machine Translation

John DeNero
UC Berkeley

Decoding for Phrase-Based Machine Translation

Search state:

- The most recent n-1 target words (for n-gram language model)
- Coverage of source words (to ensure each word translated once)
- Most recent source position translated (for reordering)

Node score:

- Translation, language model, and reordering (distortion) scores
- Optimistic estimate of future translation & LM scores

Search strategy:

- Build target sentence left-to-right (to score language model)
- Each new state added by translating one untranslated phrase
- Extend a partial translation only if it's among the top K ways to translate N source words.

(Koehn Slides)

Neural Sequence-to-Sequence Models

Conditional Sequence Generation

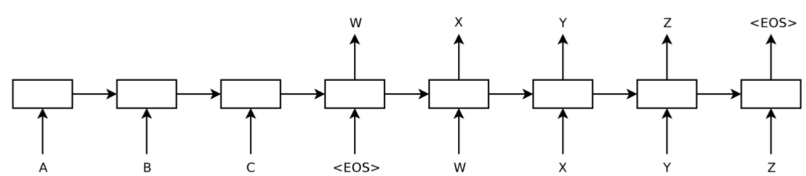
P(e|f) could just be estimated from a sequence model P(f, e)

<f> das Haus ist klein </f> the house is small </e>

Run an RNN over the whole sequence, which first computes P(f), then computes P(e, f).

Encoder-Decoder: Use different parameters or architectures encoding f and predicting e.

"Sequence to sequence" learning (Sutskever et al., 2014)



(Sutskever et al., 2014) Sequence to sequence learning with neural networks.

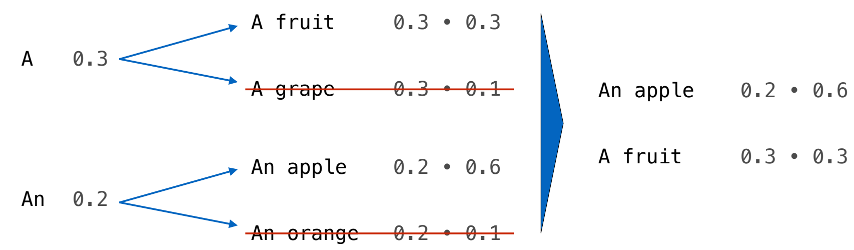
Neural Decoding

Search Strategies for Neural Machine Translation

For each target position, each word in the vocabulary is scored. (Alternatively, a restricted list of vocabulary items can be selected based on the source sentence, but quality can degrade.)

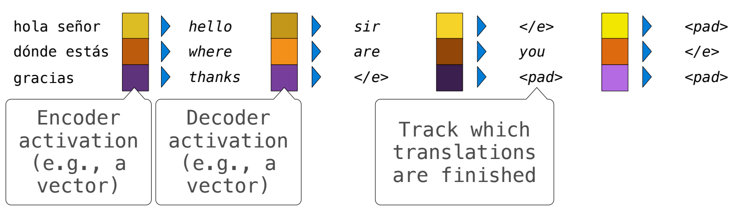
Greedy decoding: Extend a single hypothesis (partial translation) with the next word that has highest probability.

Beam search: Extend multiple hypotheses, then prune.

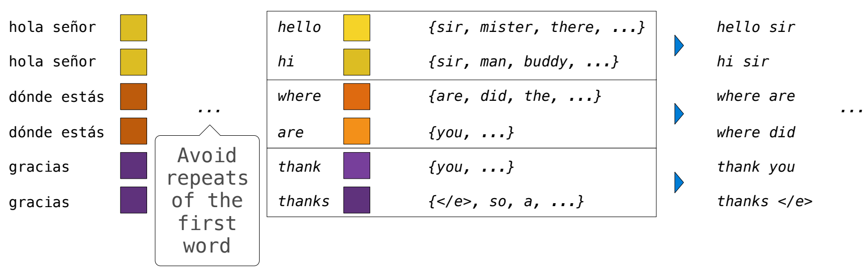


Implementing Beam Search for Batch Decoding

Greedy search:



Beam search (beam width of 2):



Beam Search Criteria to Compensate for Bad Models

NMT models often prefer translations that are too short.

$$s(e) = \sum_{i=1}^{m} \log P(e_i | e_{1:i}, f)$$

"For more than 50% of the sentences, the model in fact assigns its global best score to the empty translation" (Stahlberg & Byrne, 2019)

Alternatives for scoring items on the beam:

Length normalization: $s(e)/m$

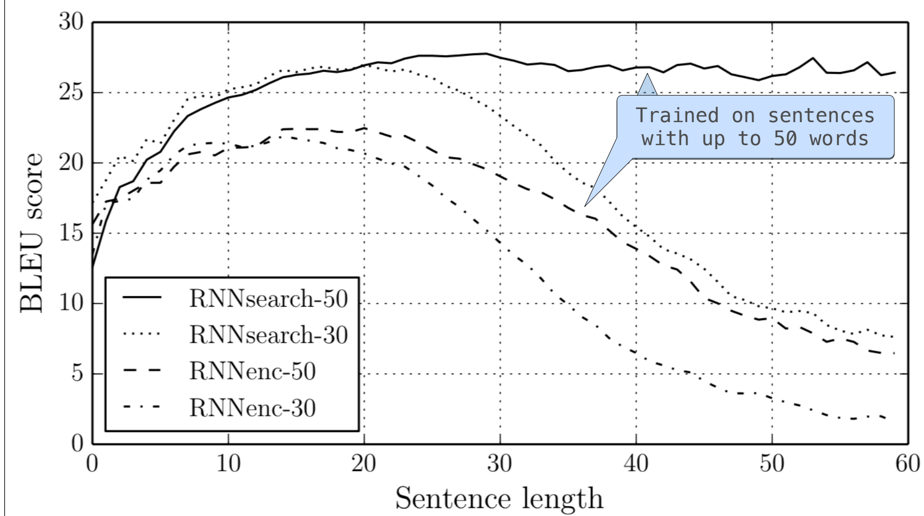
Google's correction (2016): $\dfrac{s(e)}{\frac{(5+m)^\alpha}{(5+1)^\alpha}}$

Word reward: $s(e) + \gamma m$

(Stahlberg & Byrne, 2019) On NMT Search Errors and Model Errors: Cat Got Your Tongue?
(Murray & Chiang, 2018) Correcting Length Bias in Neural Machine Translation

Attention

Impact of Attention on Long Sequence Generation



(Badhanau et al., 2015) Neural Machine Translation by Jointly Learning to Align and Translate

Conditional Gated Recurrent Unit with Attention

$$\mathbf{s}_j = \mathrm{cGRU}_{\mathrm{att}}\left(\mathbf{s}_{j-1}, y_{j-1}, \mathrm{C}\right)$$

Architecture for the top research system in WMT16 and WMT17 (Univ. Edinburgh)

$$\mathbf{s}'_j = (1 - \mathbf{z}'_j) \odot \underline{\mathbf{s}}'_j + \mathbf{z}'_j \odot \mathbf{s}_{j-1}$$

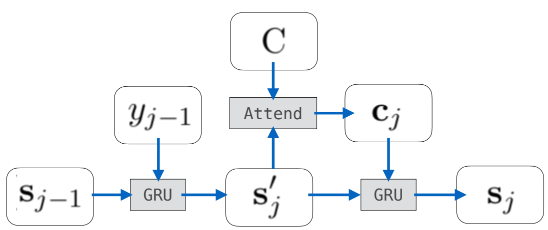
$$\underline{\mathbf{s}}'_j = \tanh\left(\mathbf{W}'\mathbf{E}[y_{j-1}] + \mathbf{r}'_j \odot (\mathbf{U}'\mathbf{s}_{j-1})\right),$$

$$\mathbf{r}'_j = \sigma\left(\mathbf{W}'_r\mathbf{E}[y_{j-1}] + \mathbf{U}'_r\mathbf{s}_{j-1}\right),$$

Reset gate masks the previous state's projection within the nonlinear forward step

$$\mathbf{z}'_j = \sigma\left(\mathbf{W}'_z\mathbf{E}[y_{j-1}] + \mathbf{U}'_z\mathbf{s}_{j-1}\right),$$

Update gate mixes the output of the forward step with the previous state



(Firat and Cho, 2016) DL4MT-Tutorial: Conditional Gated Recurrent Unit with Attention Mechanism

Conditional Gated Recurrent Unit with Attention

$$\mathbf{s}_j = \mathrm{cGRU}_{\mathrm{att}}\left(\mathbf{s}_{j-1}, y_{j-1}, \mathrm{C}\right)$$

$$\mathbf{s}'_j = (1 - \mathbf{z}'_j) \odot \underline{\mathbf{s}}'_j + \mathbf{z}'_j \odot \mathbf{s}_{j-1}$$

$\mathbf{E}[y_{j-1}]$

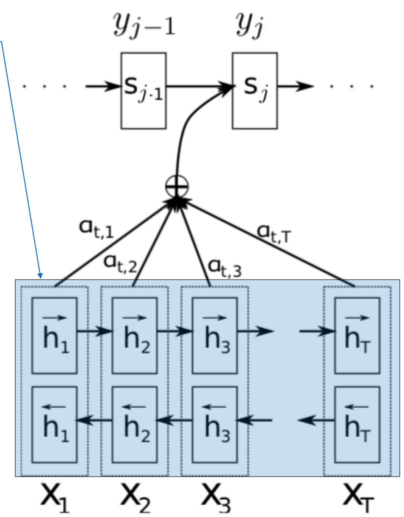
$$\mathbf{c}_j = \mathrm{ATT}\left(\mathrm{C}, \mathbf{s}'_j\right) = \sum_i^{T_x} \alpha_{ij}\mathbf{h}_i,$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{Tx}\exp(e_{kj})},$$

$$e_{ij} = \mathbf{v}_a^{\intercal}\tanh\left(\mathbf{U}_a\mathbf{s}'_j + \mathbf{W}_a\mathbf{h}_i\right)$$

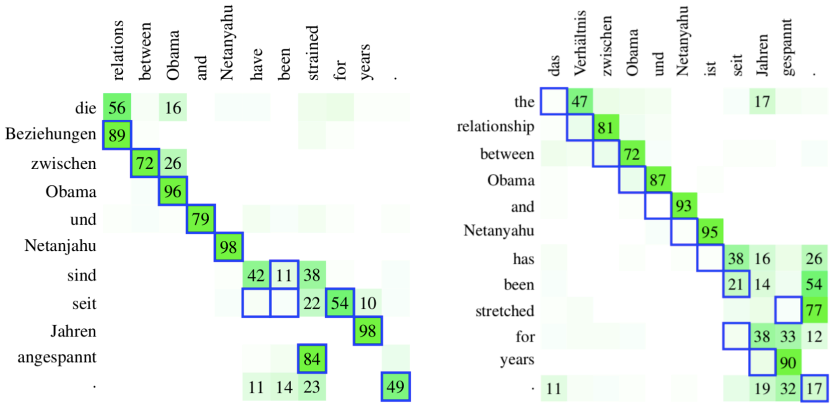
$$\mathbf{s}_j = (1 - \mathbf{z}_j) \odot \underline{\mathbf{s}}_j + \mathbf{z}_j \odot \mathbf{s}'_j$$

$\mathbf{c}_j$



Attention Activations

Attention activations above 0.1

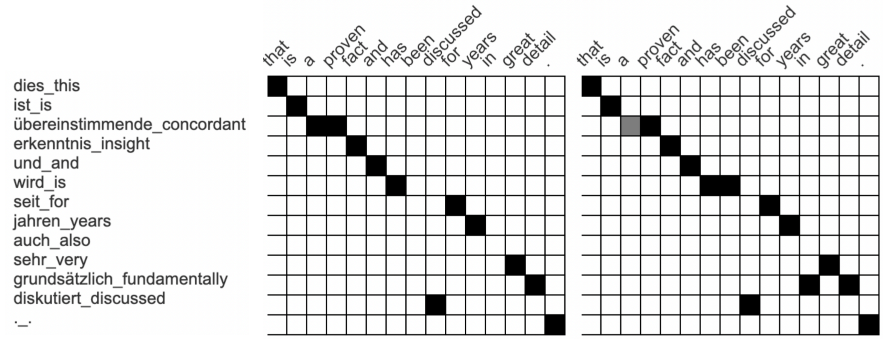


English–German   German–English

(Koehn & Knowles 2017) Six Challenges for Neural Machine Translation

Better Alignments from Attention Activations

Ideas:
(1) Find attention activations that would have led to correct word choice.
(2) Choose target words conditioned only on source context.
(3) Find attention activations that are good for both e->f and f->e.



(b) Bidir. Optimization   (c) Gold Alignments

(Zenkel et al., 2020) End-to-End Neural Word Alignment Outperforms GIZA++

Transformer Architecture

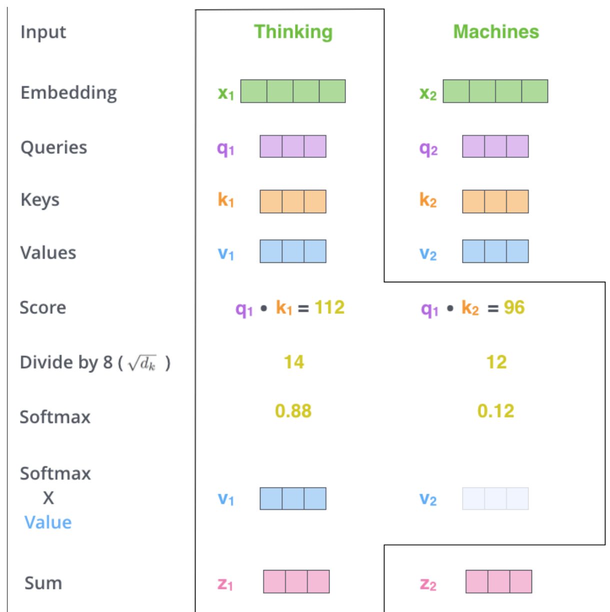
Transformer

In lieu of an RNN, use attention.

High throughput & expressivity: compute queries, keys and values as (different) linear transformations of the input.

Attention weights are queries • keys; outputs are sums of weighted values.

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

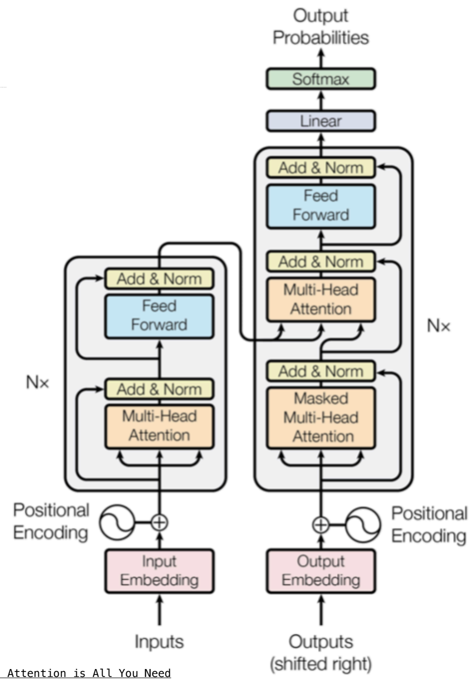


(Vaswani et al., 2017) Attention is All You Need
Figure: http://jalammar.github.io/illustrated-transformer/

Transformer Architecture

- Layer normalization ("Add & Norm" cells) helps with RNN+attention architectures as well.
- Positional encodings can be learned or based on a formula that makes it easy to represent distance.

	EN-DE
ByteNet [18]	23.75
Deep-Att + PosUnk [39]	
GNMT + RL [38]	24.6
ConvS2S [9]	25.16
MoE [32]	26.03
Deep-Att + PosUnk Ensemble [39]	
GNMT + RL Ensemble [38]	26.30
ConvS2S Ensemble [9]	26.36
Transformer (base model)	27.3
Transformer (big)	**28.4**



(Vaswani et al., 2017) Attention is All You Need

Some Transformer Concerns

Problem: Bag-of-words representation of the input.
Remedy: Position embeddings are added to the word embeddings.

Problem: During generation, can't attend to future words.
Remedy: Masked training that zeroes attention to future words.

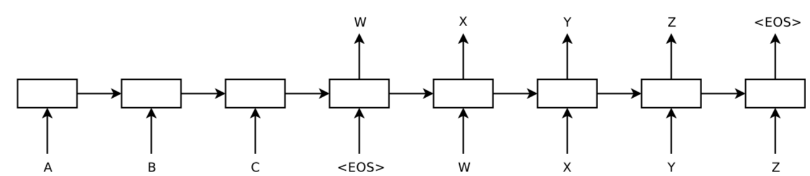
Problem: Deep networks needed to integrated lots of context.
Remedies: Residual connections and multi-head attention.

Problem: Optimization is hard.
Remedies: Large mini-batch sizes and layer normalization.

Training and Inference

Training Loss Function

Teacher forcing: During training, only use the predictions of the model for the loss, not the input.



Label smoothing: Update toward a distribution in which
- 0.9 probability is assigned to the observed word, and
- 0.1 probability is divided uniformly among all other words.

Sequence-level loss has been explored, but (so far) abandoned.

Training Data

Subwords

The sequence of symbols that are embedded should be common enough that an embedding can be estimated robustly for each, and all symbols have been observed during training.

Solution 1: Symbols are words with rare words replaced by UNK.

- Replacing UNK in the output is a new problem (like alignment).
- UNK in the input loses all information that might have been relevant from the rare input word (e.g., tense, length, POS).

Solution 2: Symbols are subwords.

- Byte-Pair Encoding is the most common approach.
- Other techniques that find common subwords aren't reliably much better (but are somewhat more complicated).
- Training on many sampled subword decompositions can improve out-of-domain translations.

(Sennrich et al., 2016) Neural Machine Translation of Rare Words with Subword Units
(Kudo, 2018) Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates

BPE Example

system	sentence
source	health research institutes
reference	Gesundheitsforschungsinstitute
word-level (with back-off)	Forschungsinstitute
character bigrams	Fo|rs|ch|un|gs|in|st|it|ut|io|ne|n
BPE	Gesundheits|forsch|ungsin|stitute

Example from Rico Sennrich

Initialize: Split each word into symbols that are individual characters

Repeat: Convert the most frequent symbol bigram into a new symbol

```
vocab = {'l o w </w>' : 5,
         'l o w e r </w>' : 2,
         'n e w e s t </w>': 6,
         'w i d e s t </w>': 3}
```

```
('e', 's') appears 9 times and is now 'es'
('es', 't') appears 9 times and is now 'est'
('est', '</w>') appears 9 times and is now 'est</w>'
('l', 'o') appears 7 times and is now 'lo'
('lo', 'w') appears 7 times and is now 'low'
('n', 'e') appears 6 times and is now 'ne'
('ne', 'w') appears 6 times and is now 'new'
('new', 'est</w>') appears 6 times and is now 'newest</w>'
('low', '</w>') appears 5 times and is now 'low</w>'
('w', 'i') appears 3 times and is now 'wi'
```

```
{'low</w>': 5, 'low e r </w>': 2, 'newest</w>': 6, 'wi d est</w>': 3}
```

(Sennrich et al., 2016) Neural Machine Translation of Rare Words with Subword Units

Back Translations

Synthesize an en-de parallel corpus by using a de-en system to translate monolingual de sentences.

- Better generating systems don't seem to matter much.
- Can help even if the de sentences are already in an existing en-de parallel corpus!

system	EN→DE		DE→EN	
	dev	test	dev	test
baseline	22.4	26.8	26.4	28.5
+synthetic	25.8	31.6	29.9	36.2
+ensemble	27.5	33.1	31.5	37.5
+r2l reranking	**28.1**	**34.2**	**32.1**	**38.6**

Table 2: English↔German translation results (BLEU) on dev (newstest2015) and test (newstest2016). Submitted system in bold.

(Sennrich et al., 2015) Improving Neural Machine Translation Models with Monolingual Data
(Sennrich et al., 2016) Edinburgh Neural Machine Translation Systems for WMT 16