Sequence-to-sequence models with attention

The attending RNN generates a query describing what it wants to focus on.

Each item is dot producted with the query to produce a score, describing how well it matches the query. The scores are fed into a softmax to create the attention distribution.

Many slides adapted from J. Johnson
Outline

• Vanilla seq2seq with RNNs
• Seq2seq with RNNs and attention
• Image captioning with attention
• Convolutional seq2seq with attention
Sequence-to-sequence with RNNs

Encoder: \( h_t = f_W(x_t, h_{t-1}) \)

Decoder: \( s_t = g_U(y_{t-1}, h_{t-1}, c) \)

From final hidden state predict:

- Initial decoder state \( s_0 \)
- Context vector \( c \) (often \( c = h_T \))

\[ \text{[START]} \rightarrow y_0 \rightarrow y_1 \rightarrow y_2 \rightarrow y_3 \rightarrow [\text{STOP}] \]

\[ \text{we} \rightarrow \text{are} \rightarrow \text{eating} \rightarrow \text{bread} \]
Sequence-to-sequence with RNNs

I. Sutskever, O. Vinyals, Q. Le, *Sequence to Sequence Learning with Neural Networks*, NeurIPS 2014

Sequence-to-sequence with RNNs

Encoder: \( h_t = f_W(x_t, h_{t-1}) \)

Decoder: \( s_t = g_U(y_{t-1}, h_{t-1}, c) \)

From final hidden state predict:

Initial decoder state \( s_0 \)

Context vector \( c \) (often \( c = h_T \))

Problem: Input sequence bottlenecked through fixed-sized vector

Idea: use new context vector at each step of decoder!

A. Sutskever, O. Vinyals, Q. Le, *Sequence to sequence learning with neural networks*, NeurIPS 2014
Sequence-to-sequence with RNNs and attention

- Intuition: translation requires *alignment*
Sequence-to-sequence with RNNs and attention

- At each timestep of decoder, context vector “looks at” different parts of the input sequence
Sequence-to-sequence with RNNs and attention

- Compute context vector as
  \[ c_t = \sum_i a_{t,i} h_i \]

- Compute scalar alignment scores
  \[ e_{t,i} = f_{\text{att}}(s_{t-1}, h_i) \]

- Normalize to get attention weights \( a_{t,i} \)

- Intuition: Context vector “attends” to the relevant part of the input sequence “estamos” = “we are” so maybe \( a_{11} = a_{12} = 0.45, \ a_{13} = a_{14} = 0.05 \)

Sequence-to-sequence with RNNs and attention

\[ \begin{align*}
    h_1 &\rightarrow a_{21} & e_{21} &\rightarrow x_1 \\
    h_2 &\rightarrow a_{22} & e_{22} &\rightarrow x_2 \\
    h_3 &\rightarrow a_{23} & e_{23} &\rightarrow x_3 \\
    h_4 &\rightarrow a_{24} & e_{24} &\rightarrow x_4 \\
\end{align*} \]

softmax

\[ \begin{align*}
    \text{estamos} &\rightarrow y_1 \\
    \text{comiendo} &\rightarrow y_2 \\
\end{align*} \]

Use \( c_2 \) to compute \( s_2, y_2 \)

Repeat: Use \( s_1 \) to compute new context vector \( c_2 \)

\[ \begin{align*}
    [\text{START}] &\rightarrow s_0 \\
    s_0 &\rightarrow s_1 \\
    s_1 &\rightarrow s_2 \\
\end{align*} \]
Sequence-to-sequence with RNNs and attention

$h_1 \rightarrow h_2 \rightarrow h_3 \rightarrow h_4 \rightarrow s_0$

softmax

$e_{31} \rightarrow e_{32} \rightarrow e_{33} \rightarrow e_{34}$

$a_{31} \rightarrow a_{32} \rightarrow a_{33} \rightarrow a_{34}$

$\times \times \times \times$

$[START]$ \rightarrow $s_0$ \rightarrow $s_1$ \rightarrow $s_2$ \rightarrow $s_3$

$y_1 \rightarrow y_2 \rightarrow y_3$

$c_1 \rightarrow y_0 \rightarrow c_2 \rightarrow y_1 \rightarrow c_3 \rightarrow y_2$

we \rightarrow are \rightarrow eating \rightarrow bread

estamos \rightarrow comiendo \rightarrow pan
Sequence-to-sequence with RNNs and attention
Sequence-to-sequence with RNNs and attention

\[
\begin{align*}
&x_1 \rightarrow h_1 \\
&x_2 \rightarrow h_2 \\
&x_3 \rightarrow h_3 \\
&x_4 \rightarrow h_4 \\
&\text{we} \rightarrow \text{are} \rightarrow \text{eating} \rightarrow \text{bread}
\end{align*}
\]
Sequence-to-sequence with RNNs and attention

- Visualizing attention weights:

Same word order in source and target languages

Flipped word order

Verb conjugation is different

Quantitative evaluation

![Graph showing BLEU scores vs sentence length for different models and attention settings.]

- With attention (trained with sentence length \( \leq 50 \))
- With attention (trained with sentence length \( \leq 30 \))
- No attention

Google Neural Machine Translation (GNMT)

Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi
yonghui,schuster,zhifengc,qvl,mnorouzi@google.com

Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey,
Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser,
Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens,
George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa,
Alex Rudnicky, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean

Y. Wu et al., Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, arXiv 2016

https://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html
Google Neural Machine Translation (GNMT)

Y. Wu et al., Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, arXiv 2016
Google Neural Machine Translation (GNMT)

- **Standard training objective**: maximize log-likelihood of ground truth output given input:
  \[ \sum_i \log P_W(Y_i^*|X_i) \]
  - Not related to task-specific reward function (e.g., BLEU score)
  - Does not encourage “better” incorrect sentences to get better likelihood

- **Refinement objective**: expectation of rewards over possible predicted sentences \( Y \):
  \[ \sum_i \sum_Y P_W(Y|X_i) R(Y,Y_i^*) \]
  - Use variant of BLEU score to compute reward
  - Reward is not differentiable -- need reinforcement learning to train (initialize with ML-trained model)
Google Neural Machine Translation (GNMT)

- Human evaluation results on production data (500 randomly sampled sentences from Wikipedia and news websites)

<table>
<thead>
<tr>
<th>Language Pair</th>
<th>PBMT</th>
<th>GNMT</th>
<th>Human</th>
<th>Relative Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>English → Spanish</td>
<td>4.885</td>
<td>5.428</td>
<td>5.550</td>
<td>87%</td>
</tr>
<tr>
<td>English → French</td>
<td>4.932</td>
<td>5.295</td>
<td>5.496</td>
<td>64%</td>
</tr>
<tr>
<td>English → Chinese</td>
<td>4.035</td>
<td>4.594</td>
<td>4.987</td>
<td>58%</td>
</tr>
<tr>
<td>Spanish → English</td>
<td>4.872</td>
<td>5.187</td>
<td>5.372</td>
<td>63%</td>
</tr>
<tr>
<td>French → English</td>
<td>5.046</td>
<td>5.343</td>
<td>5.404</td>
<td>83%</td>
</tr>
<tr>
<td>Chinese → English</td>
<td>3.694</td>
<td>4.263</td>
<td>4.636</td>
<td>60%</td>
</tr>
</tbody>
</table>

**Side-by-side scores**: range from 0 (“completely nonsense translation”) to 6 (“perfect translation”), produced by human raters fluent in both languages

**PBMT**: Translation by phrase-based statistical translation system used by Google

**GNMT**: Translation by GNMT system

**Human**: Translation by humans fluent in both languages
Outline

• Vanilla seq2seq with RNNs
• Seq2seq with RNNs and attention
• Image captioning with attention
Generalizing attention

\[ c_t = \sum_i a_{t,i} h_i \]

- The decoder doesn’t use the fact that the \( h_i \) form an ordered sequence – it just treats them as an unordered set
- Can use similar architecture given any set of input hidden vectors \( \{h_i\} \)!
Image captioning with RNNs and attention

• Idea: pay attention to different parts of the image when generating different words
• Automatically learn this grounding of words to image regions without direct supervision

K. Xu et al., Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, ICML 2015
Image captioning with RNNs and attention

\[ e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j}) \]

Use CNN to extract a grid of features

K. Xu et al., *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*, ICML 2015
Image captioning with RNNs and attention

\[ e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j}) \]

Alignment scores

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(e_{1,1})</td>
<td>(e_{1,2})</td>
<td>(e_{1,3})</td>
</tr>
<tr>
<td>(e_{1,2})</td>
<td>(e_{1,2})</td>
<td>(e_{1,3})</td>
</tr>
<tr>
<td>(e_{1,3})</td>
<td>(e_{1,2})</td>
<td>(e_{1,3})</td>
</tr>
</tbody>
</table>

Attention weights

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_{1,11})</td>
<td>(a_{1,12})</td>
<td>(a_{1,13})</td>
</tr>
<tr>
<td>(a_{1,21})</td>
<td>(a_{1,22})</td>
<td>(a_{1,23})</td>
</tr>
<tr>
<td>(a_{1,31})</td>
<td>(a_{1,32})</td>
<td>(a_{1,33})</td>
</tr>
</tbody>
</table>

\[ c_t = \sum_i a_{t,i,j} h_i \]

CNN

Cat

\[ y_1 \]

\[ y_0 \]
Image captioning with RNNs and attention

$$e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j})$$

Use a CNN to compute a grid of features for an image

<table>
<thead>
<tr>
<th>Alignment scores</th>
<th>Attention weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e_{2,1,1}$</td>
<td>$a_{2,1,1}$</td>
</tr>
<tr>
<td>$e_{2,1,2}$</td>
<td>$a_{2,1,2}$</td>
</tr>
<tr>
<td>$e_{2,1,3}$</td>
<td>$a_{2,1,3}$</td>
</tr>
<tr>
<td>$e_{2,2,1}$</td>
<td>$a_{2,2,1}$</td>
</tr>
<tr>
<td>$e_{2,2,2}$</td>
<td>$a_{2,2,2}$</td>
</tr>
<tr>
<td>$e_{2,2,3}$</td>
<td>$a_{2,2,3}$</td>
</tr>
<tr>
<td>$e_{2,3,1}$</td>
<td>$a_{2,3,1}$</td>
</tr>
<tr>
<td>$e_{2,3,2}$</td>
<td>$a_{2,3,2}$</td>
</tr>
<tr>
<td>$e_{2,3,3}$</td>
<td>$a_{2,3,3}$</td>
</tr>
</tbody>
</table>

$$s_0$$

$$c_t = \sum_i a_{t,i,j} h_i$$

CNN

[START] cat

$y_1$

$y_2$

$y_0$

$y_1$

$c_1$

$y_0$

$y_1$
e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j})

c_t = \sum_i a_{t,i,j} h_i

Each time step of decoder uses a different context vector that looks at different parts of the input image.
Example results

- **Good captions**

  - 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.
Example results

- **Mistakes**

  - A large white **bird** standing in a forest.
  - A woman holding a **clock** in her hand.
  - A man wearing a hat and a hat on a **skateboard**.
  - A person is standing on a beach with a **surfboard**.
  - A woman is sitting at a table with a large **pizza**.
  - A man is talking on his cell phone while another man watches.
Quantitative results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>BLEU-1</th>
<th>BLEU-2</th>
<th>BLEU-3</th>
<th>BLEU-4</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flickr8k</td>
<td>Google NIC</td>
<td>63</td>
<td>41</td>
<td>27</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Soft-Attention</td>
<td>67</td>
<td>44.8</td>
<td>29.9</td>
<td>19.5</td>
<td>18.93</td>
</tr>
<tr>
<td></td>
<td>Hard-Attention</td>
<td>67</td>
<td>45.7</td>
<td>31.4</td>
<td>21.3</td>
<td>20.30</td>
</tr>
<tr>
<td>Flickr30k</td>
<td>Google NIC</td>
<td>66.3</td>
<td>42.3</td>
<td>27.7</td>
<td>18.3</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Soft-Attention</td>
<td>66.7</td>
<td>43.4</td>
<td>28.8</td>
<td>19.1</td>
<td>18.49</td>
</tr>
<tr>
<td></td>
<td>Hard-Attention</td>
<td>66.9</td>
<td>43.9</td>
<td>29.6</td>
<td>19.9</td>
<td>18.46</td>
</tr>
<tr>
<td>COCO</td>
<td>Google NIC</td>
<td>66.6</td>
<td>46.1</td>
<td>32.9</td>
<td>24.6</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Soft-Attention</td>
<td>70.7</td>
<td>49.2</td>
<td>34.4</td>
<td>24.3</td>
<td>23.90</td>
</tr>
<tr>
<td></td>
<td>Hard-Attention</td>
<td>71.8</td>
<td>50.4</td>
<td>35.7</td>
<td>25.0</td>
<td>23.04</td>
</tr>
</tbody>
</table>
“Show, attend, and tell” (Xu et al, ICML 2015)
Look at image, attend to image regions, produce question

“Ask, attend, and answer” (Xu and Saenko, ECCV 2016)
“Show, ask, attend, and answer” (Kazemi and Elqursh, 2017)
Read text of question, attend to image regions, produce answer

“Listen, attend, and spell” (Chan et al, ICASSP 2016)
Process raw audio, attend to audio regions while producing text

“Listen, attend, and walk” (Mei et al, AAAI 2016)
Process text, attend to text regions, output navigation commands

“Show, attend, and interact” (Qureshi et al, ICRA 2017)
Process image, attend to image regions, output robot control commands

“Show, attend, and read” (Li et al, AAAI 2019)
Process image, attend to image regions, output text

Source: J. Johnson
Outline

• Vanilla seq2seq with RNNs
• Seq2seq with RNNs and attention
• Image captioning with attention
• Convolutional seq2seq with attention
Recurrent vs. convolutional sequence models

- **Recurrent models:**
  - Treat input as ordered sequence (inherently sequential processing)
  - Build up context using the hidden vector

- **Convolutional models:**
  - Treat input as a grid indexed by time and feature dimension
  - Build up context using multiple layers of convolutions
  - Processing can be parallel at training time, but convolutions must be *causal*
WaveNet

- Goal: generate raw audio
  - Represented as sequence of 16-bit integer values (can be quantized to 256 discrete levels), 16K samples per second
- Applications: text-to-speech, music generation
  - Also works for speech recognition

Figure 1: A second of generated speech.

WaveNet

• Training time: compute predictions of all timesteps in parallel (conditioned on ground truth)
WaveNet

- Test time: feed each predicted sample back into the model to make prediction at next timestep
WaveNet: Results

- Text-to-speech with different speaker identities:

- Generated sample of classical piano music:

https://deepmind.com/blog/article/wavenet-generative-model-raw-audio
Temporal convolutional networks (TCNs)

- TCNs can be competitive with RNNs for a variety of sequence modeling tasks

Convolutional seq2seq with attention

**WMT’14 English-German**

<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Luong et al. (2015) LSTM (Word 50K)</td>
<td>20.9</td>
</tr>
<tr>
<td>Kalchbrenner et al. (2016) ByteNet (Char)</td>
<td>23.75</td>
</tr>
<tr>
<td>Wu et al. (2016) GNMT (Word 80K)</td>
<td>23.12</td>
</tr>
<tr>
<td>Wu et al. (2016) GNMT (Word pieces)</td>
<td>24.61</td>
</tr>
<tr>
<td>ConvS2S (BPE 40K)</td>
<td>25.16</td>
</tr>
</tbody>
</table>

**WMT’14 English-French**

<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wu et al. (2016) GNMT (Word 80K)</td>
<td>37.90</td>
</tr>
<tr>
<td>Wu et al. (2016) GNMT (Word pieces)</td>
<td>38.95</td>
</tr>
<tr>
<td>Wu et al. (2016) GNMT (Word pieces) + RL</td>
<td>39.92</td>
</tr>
<tr>
<td>ConvS2S (BPE 40K)</td>
<td>40.51</td>
</tr>
</tbody>
</table>

J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. Dauphin, *Convolutional sequence to sequence learning*, ICML 2017
Different ways of processing sequences

**RNN**

- Works on ordered sequences
- Pros: Good at long sequences: After one RNN layer, $h_T$ "sees" the whole sequence
- Cons: Not parallelizable: need to compute hidden states sequentially

**1D convolutional network**

- Works on multidimensional grids
- Pros: Good at long sequences: After one self-attention layer, each output "sees" all inputs!
- Pro: Highly parallel: Each output can be computed in parallel
- Con: Very memory-intensive

**Transformer**

- Works on sets of vectors
- Pro: Good at long sequences: after one self-attention layer, each output "sees" all inputs!
- Pro: Highly parallel: Each output can be computed in parallel
- Con: Very memory-intensive