The Attention Mechanism & Encoder-Decoder Variants

CMSC 470
Marine Carpuat
Introduction to Neural Machine Translation

- Neural language models review

- Sequence to sequence models for MT
  - Encoder-Decoder
  - Sampling and search (greedy vs beam search)
  - Practical tricks

- Attention mechanism

- Sequence to sequence models for other NLP tasks
The Encoder models the input/source sentence $F=(f_1, \ldots, f_{|F|})$.

The decoder hidden state is initialized with the last hidden state of the encoder.

The Decoder models the output/target sentence $E=(e_1, \ldots, e_{|E|})$. 
$m_{t}^{(f)} = M_{:,f_{t}}^{(f)}$

$h_{t}^{(f)} = \begin{cases} 
\text{RNN}^{(f)}(m_{t}^{(f)}, h_{t-1}^{(f)}) & t \geq 1, \\
0 & \text{otherwise.}
\end{cases}$

$m_{t}^{(e)} = M_{:,c_{t-1}}^{(e)}$

$h_{t}^{(e)} = \begin{cases} 
\text{RNN}^{(e)}(m_{t}^{(e)}, h_{t-1}^{(e)}) & t \geq 1, \\
h_{t}^{(f)} |_{F} & \text{otherwise.}
\end{cases}$

$p_{t}^{(e)} = \text{softmax}(W_{hs}h_{t}^{(e)} + b_{s})$
Problem with previous encoder-decoder model

• This approach doesn’t quite work...
  • Lots of data + lots of tricks needed to get translations that are not horrible

• Why?
  • Long-distance dependencies remain a problem
  • A single vector represents the entire source sentence
    • No matter its length

• The attention mechanism helps address this issue
  • An example of incorporating inductive bias in model architecture
Attention model intuition

• Encode each word in source sentence into a vector

• When decoding, perform a linear combination of these vectors, weighted by “attention weights”

• Use this combination when predicting next word

[Bahdanau et al. 2015]
Attention model

Source word representations

- We can use representations from bidirectional RNN encoder

\[
\begin{align*}
\overrightarrow{h}_j^{(f)} &= \text{RNN}(\text{embed}(f_j), \overrightarrow{h}_{j-1}^{(f)}) \\
\overleftarrow{h}_j^{(f)} &= \text{RNN}(\text{embed}(f_j), \overleftarrow{h}_{j+1}^{(f)}).
\end{align*}
\]

- And concatenate them in a matrix

\[
\begin{align*}
\overrightarrow{h}_j^{(f)} &= \left[ \overrightarrow{h}_j^{(f)} ; \overleftarrow{h}_j^{(f)} \right].
\end{align*}
\]

\[
H^{(f)} = \text{concat}_\text{col}(h_1^{(f)}, \ldots, h_{|F|}^{(f)}).
\]
Attention model: at each decoding time step $t$, create a source context vector $c_t$

- Attention vector $\alpha_t$:
  - Entries between 0 and 1
  - Interpreted as weight given to each source word when generating output at time step $t$

$$c_t = H^{(f)} \alpha_t.$$  

- Used to combine source representations into a context vector $c_t$
Attention model

\[ h_t^{(e)} = \text{enc}([\text{embed}(e_{t-1}); c_{t-1}], h_{t-1}) \].

\[ a_{t,j} = \text{attn\_score}(h_j^{(f)}, h_t^{(e)}). \]

\[ \alpha_t = \text{softmax}(a_t). \]

\[ p_t^{(e)} = \text{softmax}(W_{hs}[h_t^{(e)}; c_t] + b_s). \]

The context vector is concatenated with the decoder hidden state to generate the next target word.

Figure 28: A computation graph for attention.
Attention model
Various ways of calculating attention score

• Dot product

\[
\text{attn\_score}(h_j^{(f)}, h_t^{(e)}) := h_j^{(f)\top} h_t^{(e)}.
\]

• Bilinear function

\[
\text{attn\_score}(h_j^{(f)}, h_t^{(e)}) := h_j^{(f)\top} W_a h_t^{(e)}.
\]

• Multi-layer perceptron (original formulation in Bahdanau et al.)

\[
\text{attn\_score}(h_t^{(e)}, h_j^{(f)}) := w_{a2}^\top \tanh(W_{a1}[h_t^{(e)}; h_j^{(f)}]).
\]
Attention model
Illustrating attention weights
Advantages of attention

• Helps illustrate/interpret translation decisions

• Can help insert translations for out-of-vocabulary words
  • By copying or look up in external dictionary

• Can incorporate linguistically motivated priors in model
Attention extensions

Bidirectional constraints (Cohn et al. 2015)

- Intuition: attention should be similar in forward and backward translation directions

- Method: train so that we get a bonus based on the trace of matrix product for training in both directions

\[ \text{tr}(A_{X \rightarrow Y} A_{Y \rightarrow X}^T) \]
Attention extensions
An active area of research

• Attend to multiple sentences (Zoph et al. 2015)

• Attend to a sentence and an image (Huang et al. 2016)
A few more tricks: addressing length bias

• Default models tend to generate short sentences
• Solutions:
  • Prior probability on sentence length
    \[ \hat{E} = \arg\max_{E} \log P(|E| \mid |F|) + \log P(E \mid F). \]
  • Normalize by sentence length
    \[ \hat{E} = \arg\max_{E} \log P(E \mid F)/|E|. \]
Issue with Neural MT: it only works well in high-resource settings

Ongoing research

- Learn from other sources of supervision than pairs (E,F)
  - Monolingual text
  - Multiple languages

- Incorporate linguistic knowledge
  - As additional embeddings
  - As prior on network structure or parameters
  - To make better use of training data

[Koehn & Knowles 2017]
State-of-the-art neural MT models are very powerful, but still make many errors

https://www.youtube.com/watch?v=3-rfBsWmo0M
Neural Machine Translation
What you should know

• How to formulate machine translation as a sequence-to-sequence transformation task
• How to model $P(E|F)$ using RNN encoder-decoder models, with and without attention
• Algorithms for producing translations
  • Ancestral sampling, greedy search, beam search
• How to train models
  • Computation graph, batch vs. online vs. minibatch training
• Examples of weaknesses of neural MT models and how to address them
  • Bidirectional encoder, length bias
• Determine whether a NLP task can be addressed with neural sequence-to-sequence models