Neural sequence-to-sequence models for machine translation

CMSC 470
Marine Carpuat
Machine Translation

• Translation system
  • Input: source sentence $F$
  • Output: target sentence $E$
  • Can be viewed as a function

\[
\hat{E} = \text{mt}(F)
\]

• Modern machine translation systems

\[
\hat{E} = \arg\max_{E} P(E \mid F; \theta)
\]

• 3 problems

• Modeling
  • how to define $P(.)$?

• Training/Learning
  • how to estimate parameters from parallel corpora?

• Search
  • How to solve argmax efficiently?
Neural Machine Translation

• Neural language models review

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

• Sequence to sequence models for other NLP tasks
A feedforward neural 3-gram model

\[ m = \text{concat}(M_{e_t-2}, M_{e_t-1}) \]
\[ h = \tanh(W_{mh}m + b_h) \]
\[ s = W_{hs}h + b_s \]
\[ p = \text{softmax}(s) \]
A recurrent language model

\[ m_t = M_{., e_{t-1}} \]

\[ h_t = \begin{cases} \tanh(W_{mh} m_t + W_{hh} h_{t-1} + b_h) & t \geq 1 \\ 0 & \text{otherwise.} \end{cases} \]

\[ p_t = \text{softmax}(W_{hs} h_t + b_s). \]
A recurrent language model

\[ m_t = M_c e_{t-1} \]
\[ h_t = \text{RNN}(m_t, h_{t-1}) \]
\[ p_t = \text{softmax}(W_{hs} h_t + b_s). \]
Neural Machine Translation

• Neural language models review

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

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

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

The Decoder models the output/target sentence $E=(e_1, ..., e_{|E|})$. 

$P(E|F)$ as an encoder-decoder model
$P(E|F)$ as an encoder-decoder model

$$m^{(f)}_t = M_{:,f_t}$$

$$h^{(f)}_t = \begin{cases} \text{RNN}^{(f)}(m^{(f)}_t, h^{(f)}_{t-1}) & t \geq 1, \\ 0 & \text{otherwise.} \end{cases}$$

$$m^{(e)}_t = M_{:,e_{t-1}}$$

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

$$p^{(e)}_t = \text{softmax}(W_{hs}h^{(e)}_t + b_s)$$
Generating Output

• We have a model $P(E|F)$, how can we generate translations?

• 2 methods

  • **Sampling**: generate a random sentence according to probability distribution

  • **Argmax**: generate sentence with highest probability
Ancestral Sampling

While $e_{j-1}! = <\text{s}>$

$$e_j \sim P(e_j|F,e_1,\ldots,e_{j-1})$$

• Randomly generate words one by one
• Until end of sentence symbol
• Done!
Greedy search

While \( e_{j-1}! = </s> \)

\[ e_j = \arg\max P(e_j \mid F, e_1, \ldots, e_{j-1}) \]

• One by one, pick single highest probability word

• Problems
  • Often generates easy words first
  • Often prefers multiple common words to rare words
Greedy Search Example

Consider this complete search graph for a model with vocabulary \{a,b, </s>\}

What sequence does greedy search produces?
What is the best scoring sequence in the search space?
Greedy Search Example

Consider this complete search graph for a model with vocabulary \{a,b,\textless/s\}\}

Here greedy search fails to discover the best scoring output!
Beam Search

Idea: consider $b$ top hypotheses at each time step

At each time step:
- Expand the $b$ hypotheses for all words in the vocabulary
- Prune down to the top $b$ hypotheses
- Move to next step

Example beam search with $b = 2$
Neural Machine Translation

• Neural language models review

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

• Sequence to sequence models for other NLP tasks