Transformer Sequence Models and Sequence Applications
(Machine Translation, Speech Recognition)

CSE392 - Spring 2019
Special Topic in CS
Most NLP Tasks. E.g.
- Sequence Tasks
  - Language Modeling
  - Machine Translation
  - Speech Recognition

- Transformer Networks
  - Transformers
  - BERT
Multi-level bidirectional RNN (LSTM or GRU)
Multi-level bidirectional RNN (LSTM or GRU)

Each node has a forward -> and backward <- hidden state: Can represent as a concatenation of both.

(Eisenstein, 2018)
Multi-level bidirectional RNN (LSTM or GRU)

Average of top layer is an embedding (average of concatenated vectors)

(Eisenstein, 2018)
Multi-level bidirectional RNN (LSTM or GRU)

Sometimes just use left-most and right-most hidden state instead

(Eisenstein, 2018)
Encoder

A representation of input.

(Eisenstein, 2018)
Encoder-Decoder

Representing input and converting to output

(Eisenstein, 2018)
Encoder-Decoder

(Eisenstein, 2018)
Encoder-Decoder

A representation of input.
Encoder-Decoder

A representation of input.

essentially a language model conditioned on the final state from the encoder.
Encoder-Decoder

When applied to new data...

Embedding lookup

essentially a language model conditioned on the final state from the encoder.
Encoder-Decoder

A representation of input.
Encoder-Decoder

"seq2seq" model

Language 1: (e.g. Chinese)

Language 2: (e.g. English)

Softmax
Encoder-Decoder

Challenge:

- Long distance dependency when translating:
Encoder-Decoder

Challenge:

- Long distance dependency when translating:

![Diagram showing long distance dependency in Encoder-Decoder model]
Encoder-Decoder

Challenge:

- Long distance dependency when translating:

The ball was kicked by kayla.

Kayla kicked the ball.
Encoder-Decoder

Challenge:

- Long distance dependency when translating:

The ball was kicked by kayla.

Kayla kicked the ball.

A lot of responsibility put fixed-size hidden state passed from encoder to decoder.
Long Distance / Out of order dependencies

A lot of responsibility put fixed-size hidden state passed from encoder to decoder
Long Distance / Out of order dependencies
Attention

Analogy: random access memory
Attention

attention layer

\[ s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow \ldots \]

\[ \ldots \rightarrow b_{n-1} \rightarrow b_n \rightarrow b_{n+1} \rightarrow \ldots \]

\[ x_{m-1}^{(s)} \rightarrow x_m^{(s)} \rightarrow x_{m+1}^{(s)} \]

\[ y^{(0)} \rightarrow y^{(1)} \rightarrow y^{(2)} \rightarrow y^{(3)} \rightarrow \ldots \]

Softmax

\[ c_{bi} \]

\[ i: \text{current token of output} \]
\[ N: \text{tokens of input} \]

\[ c_{hi} = \sum_{n=1}^{\mid s \mid} \alpha_{h_i \rightarrow s_n} s_n \]
Attention

\[ c_{hi} = \sum_{n=1}^{\left|s\right|} \alpha_{h_i \rightarrow s_n} s_n \]
$Z$ is the vector to be attended to (the value in memory). It is typically hidden states of the input (i.e. $s_n$) but can be anything.

$$c_{hi} = \sum_{n=1}^{|s|} \alpha_{h_i \rightarrow s_n} z_n$$
Attention

\[ c_{hi} = \sum_{n=1}^{\left| s \right|} \alpha_{hi \rightarrow s_n} s_n \]
Attention

\[ \alpha_{h_i \rightarrow s} = \text{softmax}(\psi(h_i, s)) \]

\[ c_{h_i} = \sum_{n=1}^{\left|s\right|} \alpha_{h_i \rightarrow s_n} s_n \]
Attention

\[
\psi(h_i, s) = v^T \tanh(W_h h_i + W_s s) \]

\[
\alpha_{h_i \rightarrow s} = \text{softmax}(\psi(h_i, s))
\]

\[
c_{h_i} = \sum_{n=1}^{\mid s \mid} \alpha_{h_i \rightarrow s_n} s_n
\]
Attention

A useful abstraction is to make the vector attended to (the “value vector”, \( Z \)) separate than the “key vector” (s).

Score function:

\[
\psi(h_i, s) = v^T \tanh(W_h h_i + W_s s)
\]

\[
\alpha_{h_i \rightarrow s} = \text{softmax}(\psi(h_i, s))
\]

\[
c_{h_i} = \sum_{n=1}^{\mid s \mid} \alpha_{h_i \rightarrow s_n} z_n
\]
A useful abstraction is to make the vector attended to (the “value vector”, $Z$) separate than the “key vector” ($s$).
Attention

Alternative Scoring Functions

\[ \psi_{\text{add}}(h_i, s) = v^T \tanh(W_h h_i + W_s s) \]

\[ \psi_{\text{dp}}(h_i, s) = s^T h_i \]

\[ \psi_{\text{mult}}(h_i, s) = s^T W h_i \]
Attention

context vector

attention weights

0.5
0.3
0.1
0.1

I am a student

Je suis étudiant

("synced", 2017)
Attention

\[ c_{hi} = \sum_{n=1}^{|s|} \alpha_{h_{i \rightarrow s_n}} s_n \]

context vector

attention weights

0.5 0.3 0.1 0.1

attention vector

\[ \alpha_{h_{i \rightarrow s}} = \text{softmax}(\psi(h_i, s)) \]

("synced", 2017)
Attention

\[ c_{hi} = \sum_{n=1}^{s} \alpha_{hi \rightarrow s_n} s_n \]

(Bahdanau et al., 2015)

\[ \alpha_{hi \rightarrow s} = \text{softmax}(\psi(h_i, s)) \]

("synced", 2017)
Attention

$$c_{h_i} = \sum_{n=1}^{s} \alpha_{h_i \rightarrow s_n}s_n$$

(Bahdanau et al., 2015)

$$\alpha_{h_i \rightarrow s} = \text{softmax}(\psi(h_i, s))$$

(“synced”, 2017)
Machine Translation

Why?

- $40\text{billion/}year\ industry
- A center piece of many genres of science fiction
- A fairly “universal” problem:
  - Language understanding
  - Language generation
- Societal benefits of inter-cultural communication

\textit{THE BABEL FISH IS SMALL, YELLOW, LEECHLIKE, AND PROBABLY THE ODDEST THING IN THE UNIVERSE.
IT FEEDS ON BRAIN WAVE ENERGY, ABSORBING ALL...}
Machine Translation

Why?

- $40 billion/year industry
- A center piece of many genres of science fiction
- A fairly “universal” problem:
  - Language understanding
  - Language generation
- Societal benefits of inter-cultural communication

(Douglas Adams)
Machine Translation

Why Neural Network Approach works? (Manning, 2018)

- Joint end-to-end training: learning all parameters at once.
- Exploiting distributed representations (embeddings)
- Exploiting variable-length context
- High quality generation from deep decoders - stronger language models (even when wrong, make sense)
Machine Translation

As an optimization problem (Eisenstein, 2018):

$$\hat{w}^{(t)} = \arg\max_{w^{(t)}} \Psi(w^{(s)}, w^{(t)})$$
Attention

\[ c_{hi} = \sum_{n=1}^{|s|} \alpha_{h_i \rightarrow s_n} s_n \]

```
context vector
```

```
attention weights
```

\[ \alpha_{h_i \rightarrow s} = \text{softmax}(\psi(h_i, s)) \]

```
(i: synched, 2017)
```

```
I am a student <s>
```

```
Je suis étudiant <s>
```

```
```

Attention

Analogy: random access memory
Do we even need all these RNNs?
(Vaswani et al., 2017: Attention is all you need)
A useful abstraction is to make the vector attended to (the “value vector”, $Z$) separate than the “key vector” ($s$).
A useful abstraction is to make the vector attended to (the “value vector”, $Z$) separate than the “key vector” ($s$). (Eisenstein, 2018)
The Transformer: “Attention-only” models

Attention as weighting a value based on a query and key:

(Eisenstein, 2018)
The Transformer: “Attention-only” models

$z^{(i)}$

$\alpha_{m}^{(i)}$

$\psi_{\alpha}^{(i)}(m, \cdot)$

$h^{(i-1)}$

$\psi_{\alpha}$

Query

Key

Value

Output

Activation

(Eisenstein, 2018)
**The Transformer**: “Attention-only” models

\[ z^{(i)} \]
\[ \alpha^{(i)}_{m \rightarrow} \]
\[ \psi^{(i)}_{\alpha}(m, \cdot) \]
\[ h^{(i-1)} \]

\[ v \]
\[ k \]
\[ o \]
\[ m - 1 \]
\[ m \]
\[ m + 1 \]

Output

Query \[ \psi_{\alpha} \]

Key

Value

activation

(Eisenstein, 2018)