Transformers
Instructor: Yoav Artzi
Overview

• Motivation

• Transformers and self-attention
Encoders

- RNN: map each token vector to a new context-aware token using an autoregressive sequential process.
- CNN: similar outcome, but with local context using filters.
- Attention can be an alternative method to generate context-dependent embeddings.
LSTM/CNN Context

• What context do we want token embeddings to take into account?

The ballerina is very excited that she will dance in the show.

• What words need to be used as context here?
  • Pronouns context should be the antecedents (i.e., what they refer to)
  • Ambiguous words should consider local context
  • Words should look at syntactic parents/children

• Problem: RNNs (i.e., LSTMs) and CNNs fail to do this
LSTM/CNN Context

- Want:
  - LSTMs/CNNs: tend to be local
  - To appropriately contextualize, need to pass information over long distances for each word

The ballerina is very excited that she will dance in the show.

The ballerina is very excited that she will dance in the show.

- To appropriately contextualize, need to pass information over long distances for each word
Self-attention

- Each word is a *query* to form attention over all tokens
- This generates a context-dependent representation of each token: a weighted sum of all tokens
- The attention weights dynamically mix how much is taken from each token
- Can run this process iteratively, at each step computing self-attention on the output of the previous level

[Vaswani et al. 2017]
Self-attention

- Each word is a query to form attention over all tokens

- This generates a context-dependent representation of each token: a weighted sum of all tokens

- The attention weights dynamically mix how much is taken from each token

- Can run this process iteratively, at each step computing self-attention on the output of the previous level

[Vaswani et al. 2017]
Self-attention

$k$: level number

$X$: input vectors

$X = x_1, \ldots, x_n$

$x^1_i = x_i$

$\bar{\alpha}_{i,j}^k = x_i^{k-1} \cdot x_j^{k-1}$

$\alpha_i^k = \text{softmax}(\bar{\alpha}_{i,1}^k, \ldots, \bar{\alpha}_{i,n}^k)$

$x_i^k = \sum_{i=1}^{n} \alpha_{i,j}^k x_j^{k-1}$

[Vaswani et al. 2017]
Multiple Attention Heads

- Multiple attention heads can learn to attend in different ways.

- Why multiple heads? Softmax operations often end up peaky, making it hard to put weight on multiple items.

- Requires additional parameters to compute different attention values and transform vectors.

- Analogous to multiple convolutional filters.
Multiple Attention Heads

- \( k \): level number
- \( L \): number of heads
- \( X \): input vectors

\[
X = x_1, \ldots, x_n
\]

\[
x^1_i = x_i
\]

\[
\bar{\alpha}^{k,l}_{i,j} = x^{k-1}_i W^{k,l} x^{k-1}_j
\]

\[
\alpha^{k,l}_i = \text{softmax}(\bar{\alpha}^{k,l}_{i,1}, \ldots, \bar{\alpha}^{k,l}_{i,n})
\]

\[
x^{k,l}_i = \sum_{i=1}^{n} \alpha^{k,l}_{i,j} x^{k-1}_j
\]

\[
x^{k}_i = V^k [x^{k,1}_i; \ldots; x^{k,L}_i]
\]

The movie was great
What Can Self-attention do?

- Attend to nearby related terms
- But just the same to far semantically related terms

The ballerina is very excited that she will dance in the show.
• This is the basic building block of an architecture called Transformers

• There are many details to get it to work, see Vaswani et al. 2017, later work, and available implementations

• Significant improvements for many tasks, including machine translation (Vaswani et al. 2017) and context-dependent pre-trained embeddings (BERT; Devlin et al. 2018)