CS224n: NLP with deep learning
Week 7

5 May

Yukun J
@ nyu shanghai

Lecture 13: Contextual word Embeddings

- up till now, we have one representation of words:
  i.e. Word2Vec, GloVe, fastText
  pre-trained word vectors. in most cases, use of pre-trained word vectors helps, because we can train & test on more data for more words, the performance more robust.
- 2 problems: ① always the same representation regardless context
  - want fine-grained word sense disambiguation

  ② words have different aspects, semantics, syntactic connotations.
- In our LSTM,
  models are producing context-specific word representation at each position.
- use different layers to represent different aspects of word meanings.
  — weighting of LSTM
  - 1 layer for lower-level syntax
  - 1 layer for high-level semantic

Transformer models

- Motivation: - we want parallelization,
  but RNNs are inherently sequential
  - need attention for long length
- Dot-Product attention

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$

output is weighted sum of values

need normalizations

· Multi-head attention

Bert (Bidirectional Encoder Representation from Transformer)

- Mask out k% of the input words, and then predict the masked words.
  - 15%, 1 out of 7.

Bert complication: next sentence prediction

- simply learn a classifier built on top layer for each task that you fine tune for.

Lecture 14: Transformers and Self-attentation for generative models

we want model hierarchy.
No explicit method for long / short dependence range

CNN? long-distance dependencies require many layers.
Use Attention for representation.

· Text generation: self-attention: constant 'path length' between any two positions

· Self-similarity · images
· music

· Probabilistic image generation

· Non-local mean: de-blurring
we can combine locality with selfattention

Music generation using relative attention
↓
Multihead + convolution?

· Convolutions and Translational Equivariance