Representing long texts for NLU

CS224U Spring 2019
Learning representations so far

- At the word or sentence level
- **Word similarity**: \((word1, word2) \rightarrow distance\)
- **Sentiment**: sentence \(\rightarrow \{\text{positive, neutral, negative}\}\)
- **NLI**: \((word1, word2) \rightarrow \{\text{entails, not entails}\}\)
  - Recall combining multiple words with `vector_combo_func`
Learning representations so far

- Fixed-dimensional representations useful for lots of downstream tasks
- Once we have an embedding, we can perform classification, clustering, etc.
Goal
How can we apply NLU methods to long texts? (*Think news articles, scientific papers, books, transcripts, etc.*)
Sample tasks

- Document classification
- Document similarity/clustering
- Reading comprehension (e.g. NewsQA)
- Summarization
Methods
Vector representations of words

- We’ve seen lots of methods for this
  - One-hot, PPMI, LSA, word2vec, GloVe, BERT
- How can we get from word vectors to paragraph/document vectors?
Good baseline methods

- Bag of word vectors (sum, mean, max-pool)
  - What are some drawbacks?
Good baseline methods

- Bag of word vectors (sum, mean, max-pool)
  - Loses sentence structure
- Combine using structure of parse trees [1]

Good baseline methods

- Bag of word vectors (sum, mean, max-pool)
  - Loses sentence structure
- Combine using structure of parse trees [1]
  - Relies on accurate parsing, does not work as well beyond single sentences

RNNs as document encoders

- Train an RNN as an autoencoder, or for your downstream task
- Use the output at the last timestep as a document embedding
- Length limitations: loses context information after many timesteps
Doc2vec [2]

Continuous Bag of Words algorithm (word2vec [3])

Paragraph Vector - Distributed Memory

Doc2vec [2]

- Simultaneously learn a word vector for every word and document vector for every document
- Unsupervised training
- To get the vector for a new document, fix word matrix $W$, augment document matrix $D$, and train for few epochs
  - Careful: can yield different vectors for the same input!

Resources

- Doc2vec
  - The *gensim* package provides an easy-to-use API
- General document embedding
  - The *flair* library allows for using and combining various embedding types (so far only supports pooling and RNN document embedders)
TODO: LM deep learning

- Transformer
- BERT -- call out limitations
- ELMo
- Etc
- Other encoder-decoder type approaches
Sentence encoders (to include?)

- Skip-thought (sentences)
- InferSent
- Google Universal Sentence Encoder (USE)