Recitation 6: Pretraining

Subword Embeddings, MLM, and Fine Tuning

Abby Bertics & Evan Hernandez

MIT 6.864 Spring 2021

Outline

1. Subword Embeddings
2. Masked Language Modeling (MLM)
3. Pretraining and Fine Tuning
4. Colab Demo: Fine Tuning BERT

Motivation

berry, cranberry, strawberry, blueberry, blackberry, raspberry, gooseberry, boysenberry, lingonberry, huckleberry, chokeberry, elderberry, mullberry, …

Motivation

act:

Motivation

morphology

- verb inflection
- prefixes (un, de, pre, etc.)
- suffixes (ness, ly, ify, etc.)

Motivation

English is generally morphologically WEAK:

- I/you/we/they eat, he eats

Other languages have way more forms:

- ex: Russian prefixes + cases + conjugations (>100 words that have to do with reading): *читать, читаю, читаешь, читает, читаем, читаете, читают, вчитываться, вычитание, дочитать, дочитывать, дочитала, дочитал, дочитали, отчитывать, перечитать, перечитаю, итд.....*
- Agglutinative languages

Subword Embedding Methods

Goal: Chunk word into subwords (that hopefully represent meaningful morphemes)

Solutions:

- fastText
- byte pair encoding
- WordPiece Models

fastText

Steps:

- we have a word, w=
- generate all subwords of length 3 to 6:
- The main word vector $\boldsymbol{u}_w$ for the word w is:

$$\boldsymbol{u}_w = \sum_{g \in G_w} \boldsymbol{z}_g$$

Byte Pair Encoding

Allows for variable-length subwords in a fixed-size vocabulary

Greedy approach; iteratively merges the most frequent pair of consecutive symbols

BERT Tokenizer

- uses ∼30,000 token WordPiece model (variation on BPE)

ex:

```
text = "I love embeddings!"
marked_text = "[CLS] " + text + " [SEP]"

# Tokenize our sentence with the BERT tokenizer.
tokenized_text = tokenizer.tokenize(marked_text)

# Print out the tokens.
print (tokenized_text)

['[CLS]', 'i', 'love', 'em',
    '##bed', '##ding', '##s', '!', '[SEP]']
```

Subword Embedding Summary

- Practically:
  - allows us to handle large vocabularies
  - reduces memory and computation
  - addresses out-of-vocabulary issue
- Philosophically:
  - words are not the smallest unit of meaningl!!

Outline

1. Subword Embeddings
2. Masked Language Modeling (MLM)
3. Pretraining and Fine Tuning
4. Colab Demo: Fine Tuning BERT

Which is easiest?

”An elephant’s brain is three times...”

”...than that of a human.”

”An elephant’s brain is three times......than that of a human.”

Motivation

Idea: mask tokens in a sentence and predict them

kind of like how you use context to infer the meaning of an unknown word (ex: A summer *zephyr* gently stirred her hair.)

MLM vs LM

Language Model	LSTM	$P(w_i \mid w_1, ..., w_{i-1})$
Masked Language Model	BERT	$P(w_i \mid w_{j \neq i})$*

*this is assuming one word masked at a time for simplicity

LM as a Generative Model

Intuitive, word-by-word sentence building.

At each time step, i, sample a word from the LM distribution, $P(w_i|w_1, ..., w_{i-1})$

$$P(\text{sent}) = \prod_{w_i \in \text{sent}} P(w_i|w_1, ..., w_{i-1})$$

autoregressive LMs implicitly contain a distribution on sentence lengths

MLM as a Generative Model

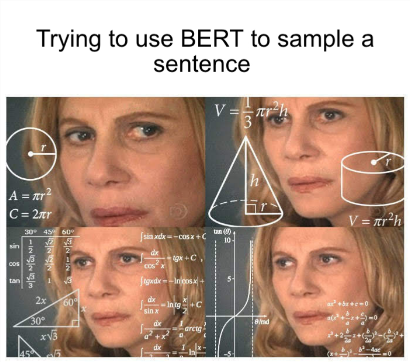

Figure 1: Confusion

Why this doesn't work

- learns a distribution over sentences *of a given length* so you could technically compare with a chain rule type method
    - ex: ”I eat cow” vs. ”I eat pig”
- how would you compare probabilities of sequences with different lengths?
- how would you generate a sentence?

Outline

1. Subword Embeddings
2. Masked Language Modeling (MLM)
3. Pretraining and Fine Tuning
4. Colab Demo: Fine Tuning BERT

Pretraining

- Learn unsupervised representations
- Train (M)LM on HUGE data
- ex:
    - word2vec
    - GloVe
    - fastText
    - BERT

(VERY LOOSELY) getting general ”language knowledge”

Fine Tuning

- $\sim$Transfer learning
- Focus on specific downstream tasks
- ex:
    - Classification problems (e.g. sentiment analysis)
    - Sequence labeling (e.g. POS tagging)
    - Natural language inference
    - Grammaticality judgements
    - Question Answering

(VERY LOOSELY) using general language knowledge to ”solve specific problems”

Example: Natural Language Inference

What is the relationship between these two sentences?

*This bird is a **cardinal**.* | *The bird is **red all over**.*

Example: Natural Language Inference

What is the relationship between these two sentences?

*This bird is a **cardinal**.*

*The bird is **red all over**.*

The left sentence **entails** ($\Rightarrow$) the right.

Example: Natural Language Inference

What is the relationship between these two sentences?

This bird is a ***cardinal****.* | *This is a* ***bird of paradise****.*

Example: Natural Language Inference

What is the relationship between these two sentences?

This bird is a ***cardinal****.*	*This is a* ***bird of paradise****.*
	

The left sentence **contradicts** ($\perp$) the right.

Example: Natural Language Inference

What is the relationship between these two sentences?

*This bird is a **cardinal**.* | *The bird is **perched on a branch**.*

Example: Natural Language Inference

What is the relationship between these two sentences?

*This bird is a **cardinal**.*

*The bird is **perched on a branch**.*

Neither of these sentences entail nor contradict the other!

Example: Natural Language Inference

- These are **natural language inference** (NLI) problems.
- Goal: Given a **hypothesis** (left sentence) and a **premise** (right sentence), predict whether the hypothesis entails, contradicts, or shares no relationship with the premise.

Example: Natural Language Inference

- NLI is hard! It requires strong language proficiency and a lot of prior knowledge about the world.
- Datasets for tasks like NLI are also (relatively) small—models trained on these datasets alone will see orders of magnitude fewer tokens than a language model trained on unlabeled text.
- Pretraining and fine tuning allow us leverage pretrained language models to bridge the gap.

Outline

1. Subword Embeddings
2. Masked Language Modeling (MLM)
3. Pretraining and Fine Tuning
4. Colab Demo: Fine Tuning BERT

Useful Material

- All the pretrained transformers you could ever want: *https://github.com/huggingface/transformers*
- Playground featuring a real world NLI model: *https://huggingface.co/roberta-large-mnli*