NMLM

NLP

Jeff Abrahamson

6 septembre 2020

Disclaimer

We have 13 minutes for 6 years.

This will simplify quite a lot. (Maybe too much.)

All the papers are on github.

Overview

What's changed:

- (Much) better performance
- Polysemy
- Pre-trained models

Overview

- $\leqslant 2014$
- 2014: word2vec and improvements
- 2018: ELMo
- 2019: BERT
- 2018–2020: GPT

Overview

Pre-trained models:

- Context-free (word2vec, FastText, GloVe)
- Contextual
  - Unidirectional (ELMo (I think))
  - Bidirectional (BERT, GPT-2,3 (I think))

Before 2014

TF-IDF (classic example)

- 1-hot encoding to get high-dimensional vectors
- TF-IDF to scale (weight)
- PCA or t-SNE etc. (if we want lower dimensionality)

Before 2014

TF-IDF (classic example)

- 1-hot encoding to get high-dimensional vectors
- TF-IDF to scale (weight)
- PCA or t-SNE etc. (if we want lower dimensionality)

What do we have?

- A space of likely significant words.
- Likely no good semantic knowledge.

Word2vec

- NN, but not deep: think auto-encoder or RBL
- Train on context: similar context $\Rightarrow$ similar embedding
- Want: distortion $\approx$ meaning

Word2vec

- NN, but not deep: think auto-encoder or RBL
- Train on context: similar context $\Rightarrow$ similar embedding
- Want: distortion $\approx$ meaning

Mikolov: trade complexity for efficiency $\Rightarrow$ learn bigger datasets

Word2vec

- NN, but not deep: think auto-encoder or RBL
- Train on context: similar context $\Rightarrow$ similar embedding
- Want: distortion $\approx$ meaning

Mikolov: trade complexity for efficiency $\Rightarrow$ learn bigger datasets

Uses: search, translation, ...

Word2vec

Encode words at learning time.

$$(\mathbb{R}^N \to \mathbb{R}^n \quad \text{for } N \approx 5 \times 10^5 \text{ and } n \approx 300)$$

Problem: no polysemy

Word2vec

How does it work?

It's learning, so we're optimising something. What?

Word2vec

How does it work?

It's learning, so we're optimising something. What?

similarity of context

Word2vec

How does it work?

It's learning, so we're optimising something. What?

cosine distance for similar contexts

Word2vec

How does it work?

It's learning, so we're optimising something. What?

cosine distance for similar contexts

Look at words around the target word. Think of as an n-gram. We want words with similar context to project close together, words with different context (ideally) not to.

Word2vec

Interesting properties:

- Magnitude is related to importance
  - Too common rarely learns large magnitude
  - Too rare rarely grows large
  - So Goldilocks property

Word2vec

Interesting properties:

- Magnitude is related to importance
  - Too common rarely learns large magnitude
  - Too rare rarely grows large
  - So Goldilocks property
- Learns stop words. Because their contexts are uncorrelated, they optimise near zero.

Word2vec

How does it work?

It's not a single algorithm:

- CBOW (continuous bag-of-words) – context predicts word
- SGNS (skipgram negative sampling) – context predicts context

Word2vec

How does it work?

- SGNS factorises a word-context PMI matrix

PMI = pointwise mutual information

$$pmi(x; y) = \log\left(\frac{\mathbf{Pr}(x, y)}{\mathbf{Pr}(x)\,\mathbf{Pr}(y)}\right) = \log\left(\frac{\mathbf{Pr}(x \mid y)}{\mathbf{Pr}(x)}\right) = \log\left(\frac{\mathbf{Pr}(y \mid x)}{\mathbf{Pr}(y)}\right)$$

Word2vec

How does it work?

Output is a softmax: cost is proportional to number of classes (50K words).

Word2vec

How does it work?

Output is a softmax: cost is proportional to number of classes (50K words).
Approximate the softmax to reduce cost.

Word2vec

The famous “man : woman as king : queen” in maths: we’re maximising two similarities and minimising a dissimilarity.

??

Word2vec

Word2vec

Improvements:

1. FastText (Facebook)
2. GloVe (“Global Vectors”)
3. ULMFit (“Universal Language Model Fine Tuning”)

Better performance, similar properties.

What NLP last quarter century, just a lot better.

ELMo

- Encode words before and after (not just at learning time)
  - No: dictionary: map word to vector
  - Yes: on the fly, run context through deep network to produce new context
- Captures linguistic context: polysemy
- Models word use: captures both syntactic and semantic information
- Most tasks better than word2vec and relatives (question answering, named entity exceptions, sentiment analysis, . . .)

ELMo

- Bidirectional LSTM + residual connectors between 1st and second layer
- Character embedding rather than word: helps with out-of-vocabular words
- Convolutional filters instead of n-gram features

Up to here, similar to Jozfowicz

Jozfowicz

ELMo

- Bidirectional LSTM + residual connectors between 1st and second layer
- Character embedding rather than word: helps with out-of-vocabular words
- Convolutional filters instead of n-gram features

- Learn different representations for different tasks
- Transfer learning

In CV, it's standard to learn ImageNet and transfer to problem at hand.

ELMo shows we can do this for NLP problems.

BERT

- Builds on ELMo
- Semi-supervised, encoder stack of transformer architecture
- Encoder-decoder: use self-attention to encode and attention to decode

Questions?