Vector Semantics and Embeddings

Goals

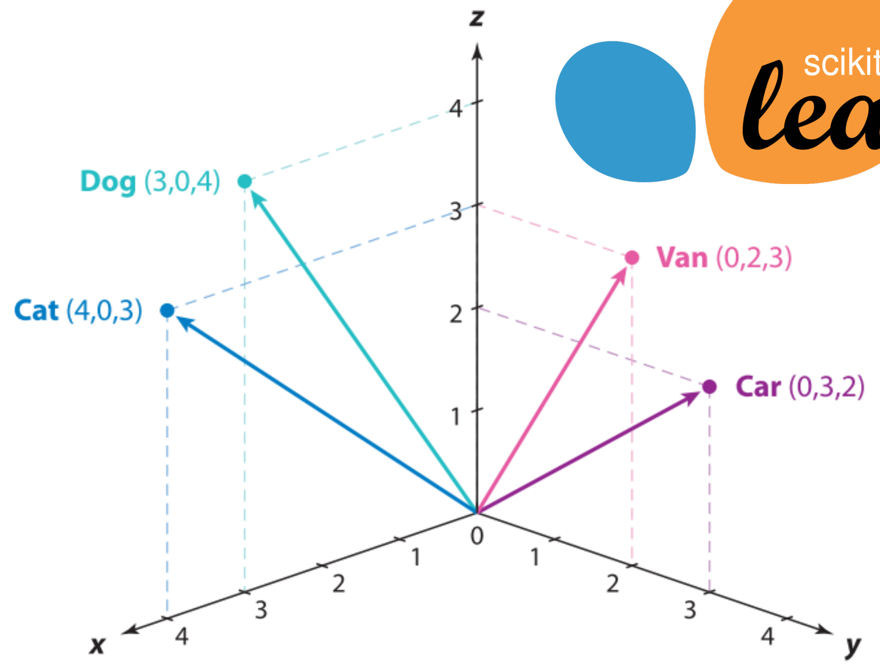
1. Consolidating the knowledge on word vectors' background and objectives
2. Appreciating the importance of producing dense real-value word vectors
3. Familiarising with the most popular word embeddings algorithms

Modern NLP

Modern NLP: key features

Background

- Distributional Hypothesis (DH)
- ML

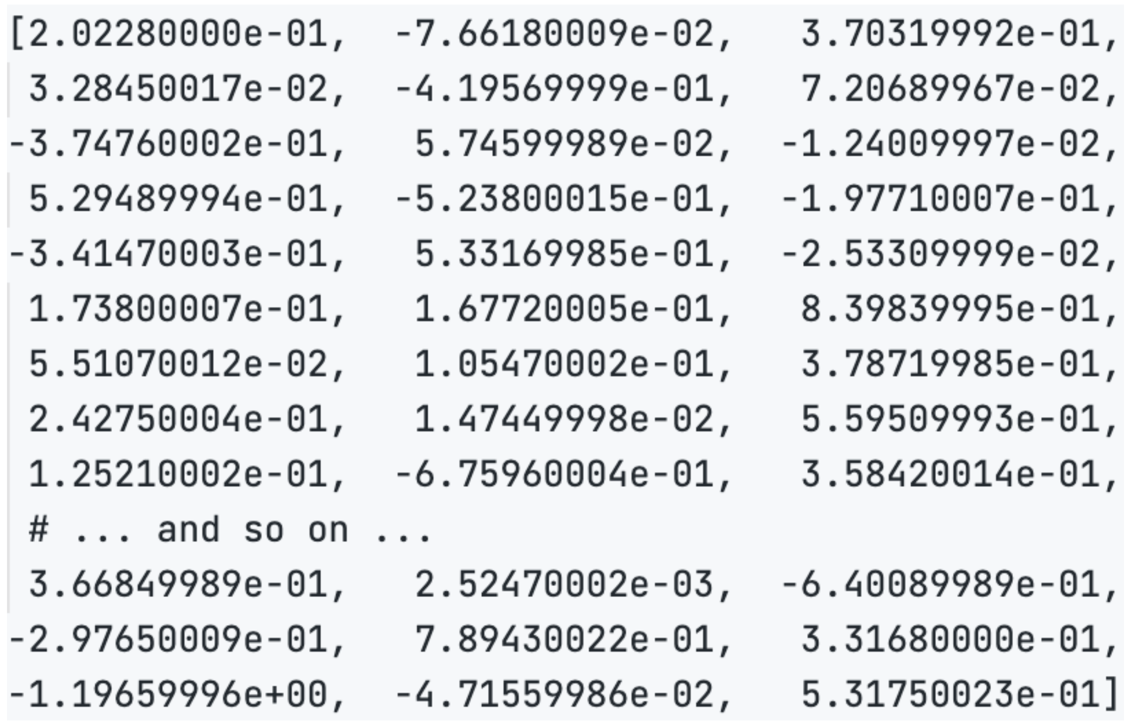


Objective

Building a dense real-value vector for each word, such that semantically associated words — i.e., words appearing in similar linguistic contexts — are associated with similar vectors.

Why do we want dense real-value vectors?

- Convenience: in ML, you don't want to deal with thousands of features.
- Effectiveness: they may do better at capturing synonymy/semantic similarity:
    - ‘car’ and ‘automobile are synonyms, but they are distinct dimensions.
    - a word with ‘car’ as a neighbor and a word with ‘automobile’ as a neighbor should be similar, but are not.

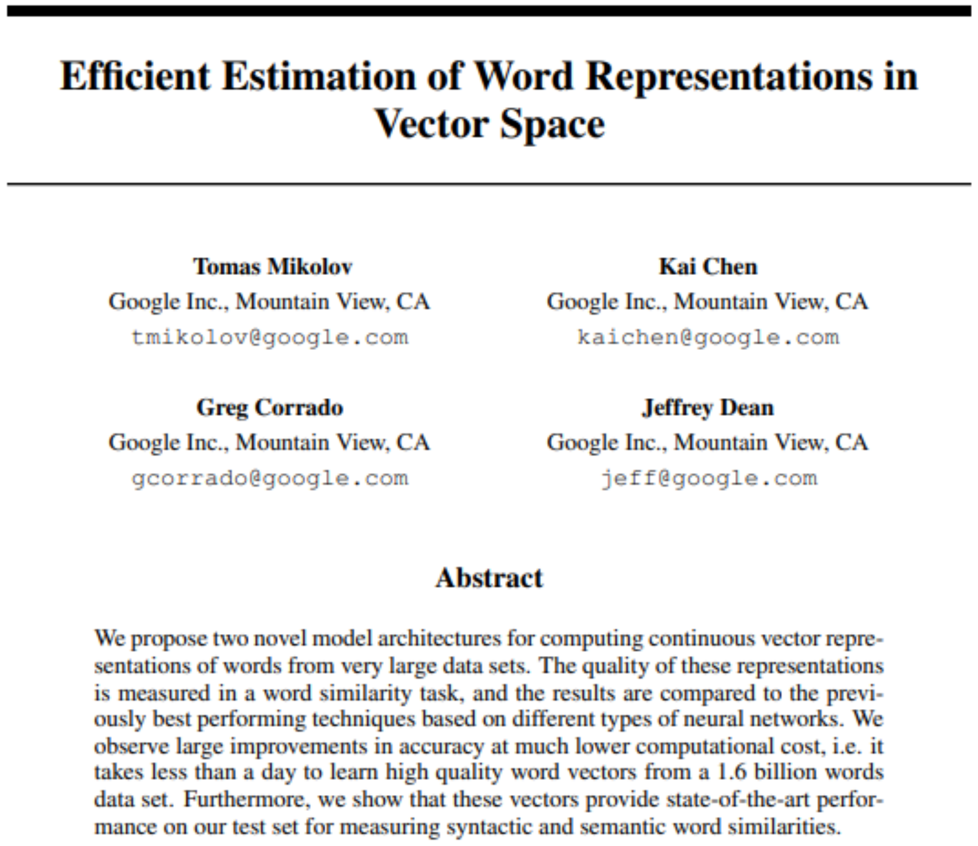


```
[2.02280000e-01,  -7.66180009e-02,   3.70319992e-01,
 3.28450017e-02,  -4.19569999e-01,   7.20689967e-02,
-3.74760002e-01,   5.74599989e-02,  -1.24009997e-02,
 5.29489994e-01,  -5.23800015e-01,  -1.97710007e-01,
-3.41470003e-01,   5.33169985e-01,  -2.53309999e-02,
 1.73800007e-01,   1.67720005e-01,   8.39839995e-01,
 5.51070012e-02,   1.05470002e-01,   3.78719985e-01,
 2.42750004e-01,   1.47449998e-02,   5.59509993e-01,
 1.25210002e-01,  -6.75960004e-01,   3.58420014e-01,
 # ... and so on ...
 3.66849989e-01,   2.52470002e-03,  -6.40089989e-01,
-2.97650009e-01,   7.89430022e-01,   3.31680000e-01,
-1.19659996e+00,  -4.71559986e-02,   5.31750023e-01]
```

Examples of word embeddings

Word2vec

- It is the pioneer in the field of word embeddings.
- Various wrappers allow to train this algorithm in a few lines of code (e.g., Gensim).
- Still very popular among researchers and industry practitioners.



Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

Variations of word2vec

GloVe

Distinctive feature: it emphasises co-occurrences of words in the whole text corpus instead of ‘local’ linguistic contexts.

fastText

Distinctive feature: It considers the aggregation of sub-words, which are eventually associated with separate vectors. This is very important for morphologically rich languages such as German (e.g., ‘street sweeper’ is a single word in German, ‘Stráßenkehrer’)

Families of word embeddings

Static embeddings

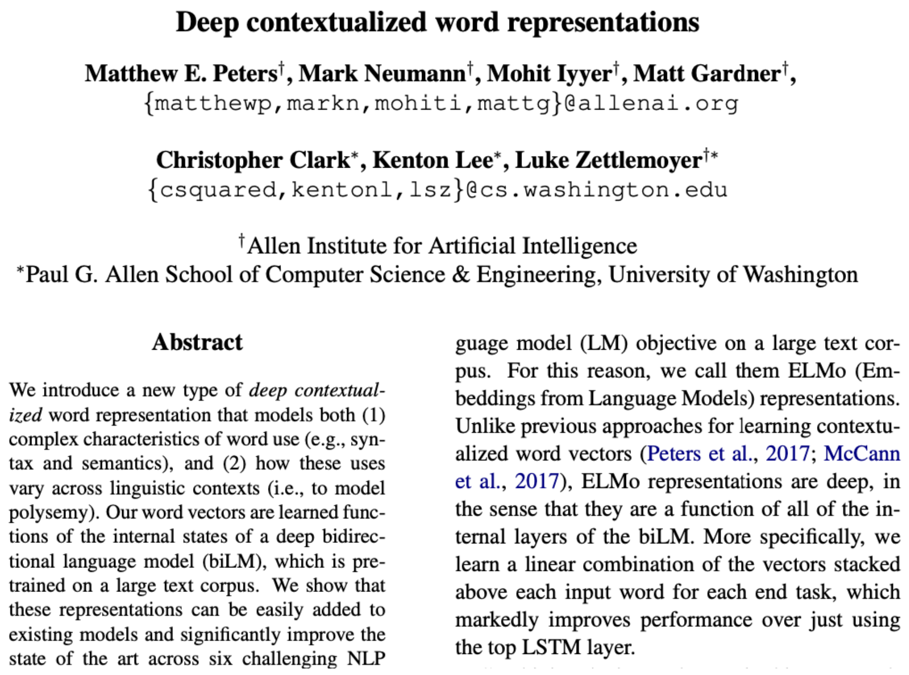
- Each word is in 1-to-1 correspondence with a vector.
- word2vec, fastText, and GloVe are examples of static embedding algorithms.

Contextual embeddings

- A word’s embedding is not fixed — instead, it depends on the linguistic context in which the word appears.
- Hence, in order to get a vector, analysts must pass a focal word along with a sequence of leading and trailing words.
- Contextual embeddings are particularly helpful for polysemous words, whose meanings are contingent (consider for example the word ‘point’).

Examples of contextual word embeddings

ELMo



Deep contextualized word representations

Matthew E. Peters[†], Mark Neumann[†], Mohit Iyyer[†], Matt Gardner[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], Kenton Lee[*], Luke Zettlemoyer[†*]
{csquared,kentonl,lsz}@cs.washington.edu

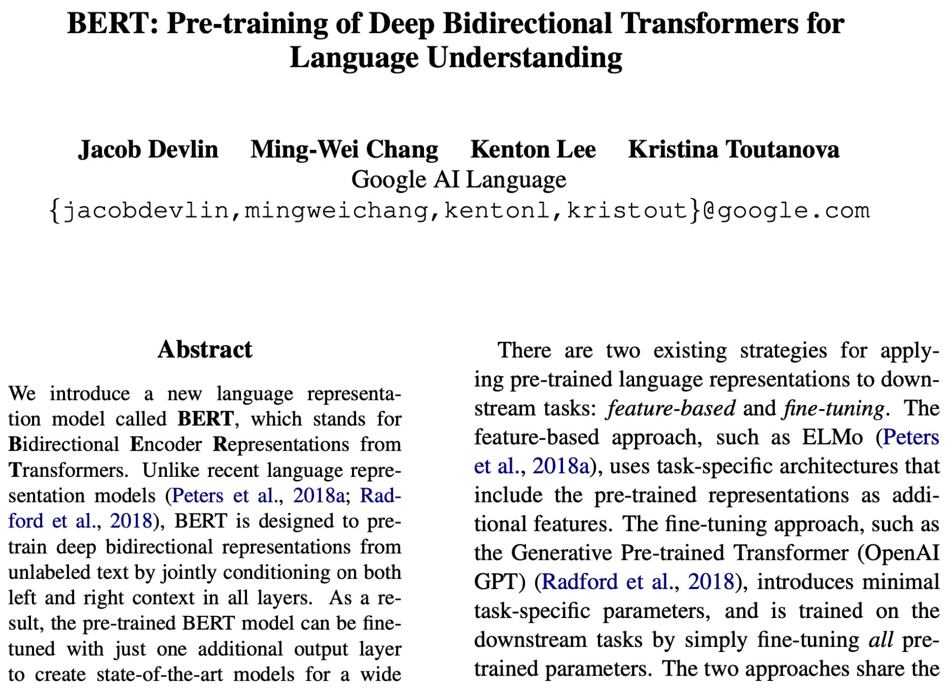
[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. We show that these representations can be easily added to existing models and significantly improve the state of the art across six challenging NLP

guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked above each input word for each end task, which markedly improves performance over just using the top LSTM layer.

BERT



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide

There are two existing strategies for applying pre-trained language representations to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning *all* pre-trained parameters. The two approaches share the

Wrap-up

Main points

- Word embeddings build on DH and ML to produce dense real value vectors.
- word2vec is the pioneer in the field of word embeddings.
- fastText and GloVe are incremental innovations concerning word2Vec.
- Contextual embeddings such as ELMo and BERT can be considered radical innovations since they associate different vectors with the same linguistic context changes.