VL Deep Learning for Natural Language Processing

Word Embeddings

Aftab Anjum

AG Information Profiling and Retrieval

Text Represenation

In traditional NLP, we regard words as discrete symbols:
hotel, conference, motel — a localist representation

one 1, the rest 0's

Words can be represented by one-hot vectors:

hotel = [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

motel = [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]

Vector dimension = number of words in vocabulary (e.g., 500,000)

How to compute similarity of two words?

Text Represenation

- The vectors we get from one-hotc encoding are sparse (most are 0's) & long (vocabulary size)

- Alternative: we want to represent words as short (50-300 dimensional) & dense (real-valued) vectors

Why do we need a dense vectors?

- Semantic Similarity
- Dimensionality Reduction
- Continuous Representation
- Generalization
- Transfer Learning

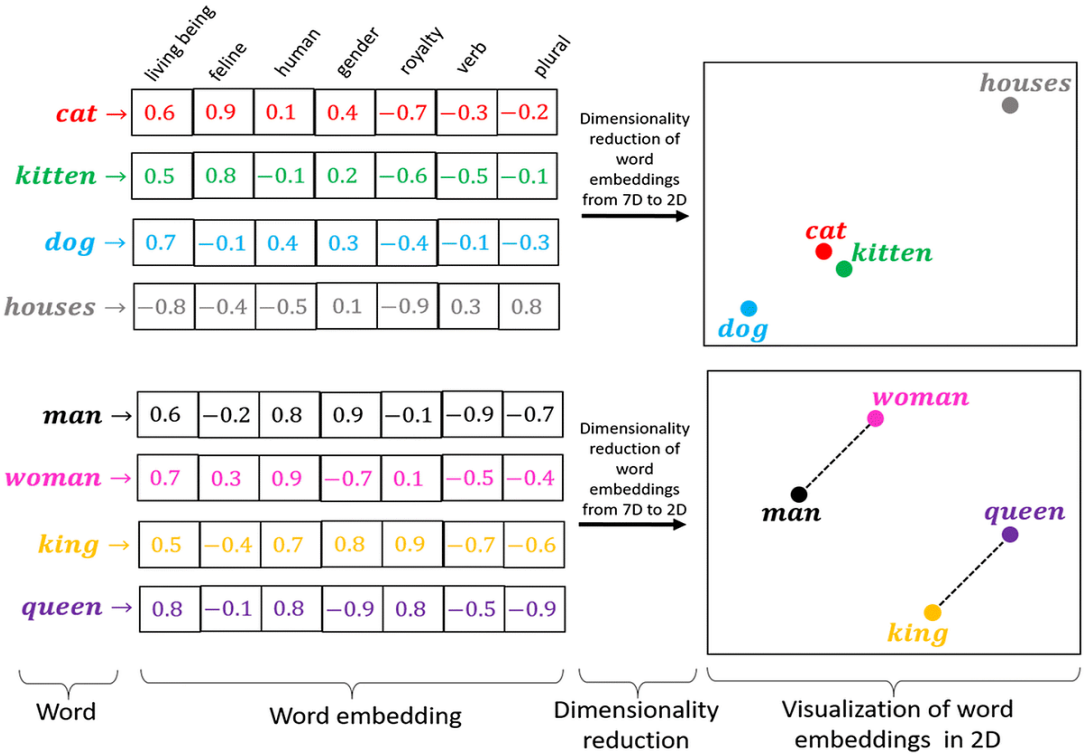
Word Embedding

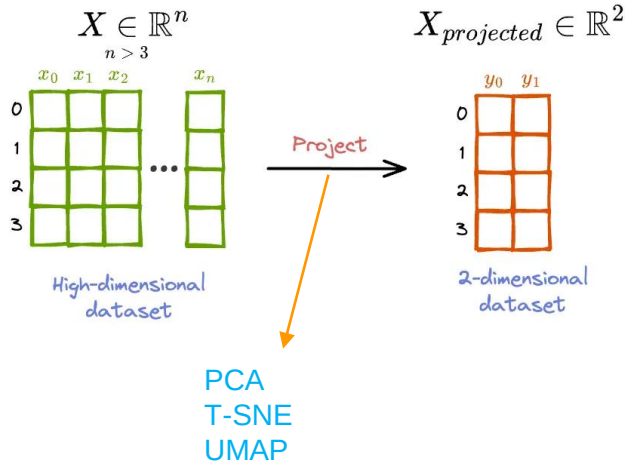
- Deep neural networks need tensors as input
  - One-hot-encoding
    - Bag-of-words
    - Sparsely populated
  - Embedding
    - Dimensionality reduction
    - Densly populated

- Word2vec
- GloVe
- Fast Text
- ELMo
- BERT

- Embedding Layer can learn representation **task-specific**
  - First layer in a DNN learns dimensionality reduction / representation
- **Pretrained** Embeddings can be used
  - First layer maps input words to pretrained word vectors

Featurized Representation: Word Embedding





https://www.researchgate.net/figure/Word-embeddings-map-words-in-a-corpus-of-text-to-vector-space-Linear-combinations-of_fig6_340825443

Word2vec

- Input: a large text corpora, V, d
- V: a pre-defined vocabulary
- d: dimension of word vectors (e.g. 300)

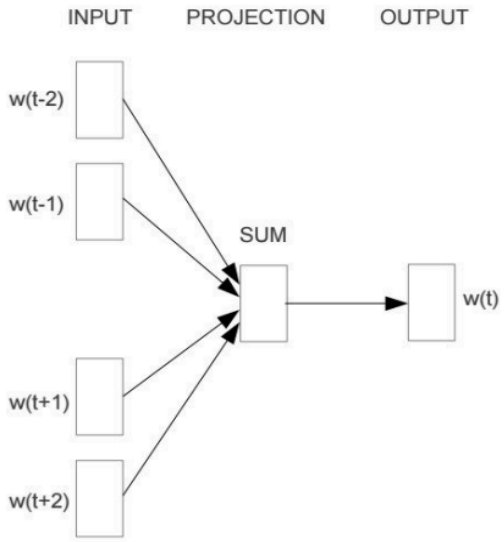
- Text corpora:
  - Wikipedia + Gigaword 5: 6B
  - Twitter: 27B
  - Common Crawl: 840B

- Embedding

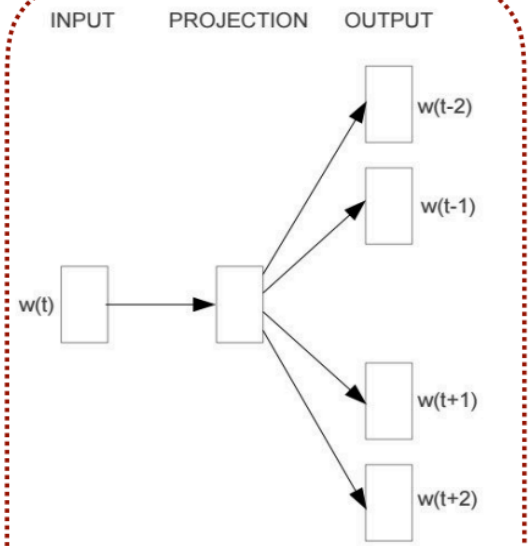
$$v_{\text{cat}} = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix} \quad v_{\text{dog}} = \begin{pmatrix} -0.124 \\ 0.430 \\ -0.200 \\ 0.329 \end{pmatrix}$$

$$v_{\text{the}} = \begin{pmatrix} 0.234 \\ 0.266 \\ 0.239 \\ -0.199 \end{pmatrix} \quad v_{\text{language}} = \begin{pmatrix} 0.290 \\ -0.441 \\ 0.762 \\ 0.982 \end{pmatrix}$$

Word2vec

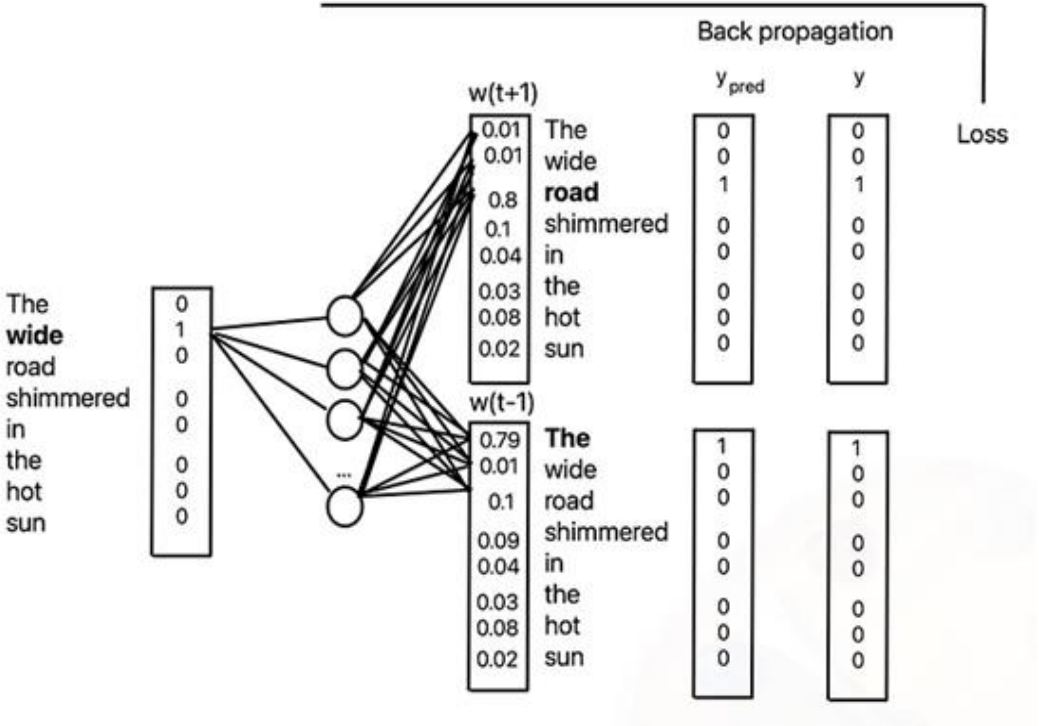


Continuous Bag of Words (CBOW)



Skip-grams

Word2vec: Skip-grams



https://www.researchgate.net/figure/2D-PCA-projection-of-word-embeddings-Five-different-word-clusters-are-shown_fig2_332892222

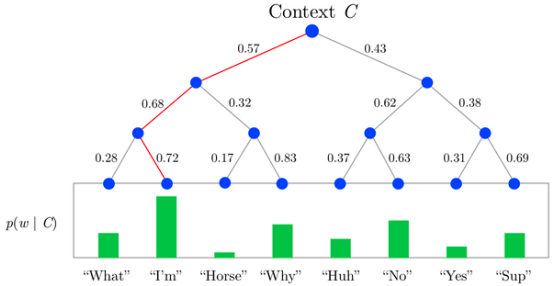
Negative Sampling

Input Words	Output words	Targets
wide	the	1
wide	mark	0
wide	data	0
wide	road	1

Pick randomly from vocabulary

Taco
Mark
Solar
Earth
System
Data
Key
.
.
.

Table



Hierarchical Softmax

Word Embedding: Application

What you think, where we can used word embedding?

- Text classification
- Name entity recognition
- Machine translation
- Question answering
- Information retrieval

What are the limitation of word embedding

- Out-of-Vocabulary (OOV) Words
- Biased
- Limited Contextual Information

Exercise

- Implement the fasttext and word2vec idea of n-gram representations
- Compare the results
  - fasttext vs. Word2vec
  - do some analogy and visualization
  - compare the results with fine-tuned and pre-trained embedding

Pretrained Word Vectors vs. Newly Learned

- How about a compromise?
  - Load pretrained word vectors and then continue training with current data?
- Problem:
  - Words that occur in the training set move around in the embedding space; words that do not occur in the training set but maybe in the test set stay where they are.

Literature

- Evaluation methods for unsupervised word embeddings
- Linear Algebraic Structure of Word Senses, with Applications to Polysemy
- On the Dimensionality of Word Embedding
- Debiasing Word Embeddings
- Dynamic Word Embeddings