Deep Learning

8.2 Word Embeddings (CBOW)

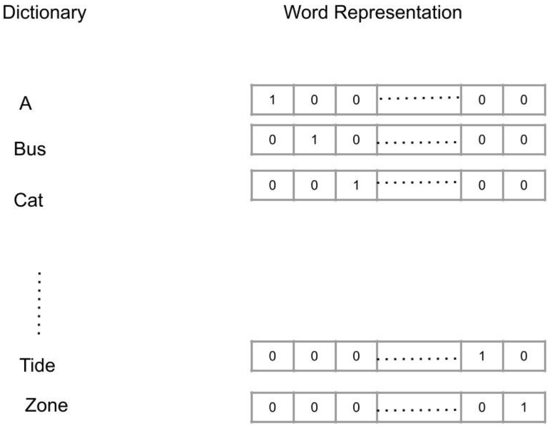
Dr. Konda Reddy Mopuri
kmopuri@iittp.ac.in
Dept. of CSE, IIT Tirupati

Word Embeddings

1. Text processing with NNs require to encoding into vectors

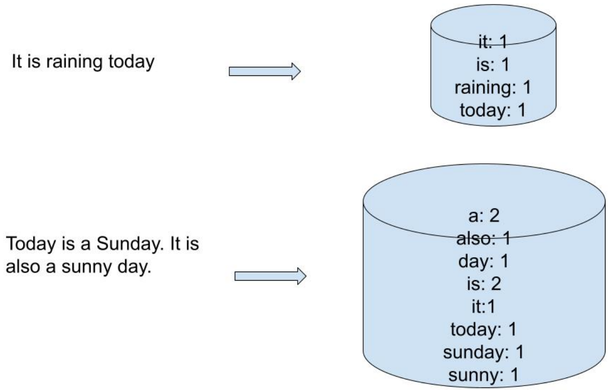
One-hot encoding

1. One-hot encoding: N words encoded as binary vectors of length N



Bag of Words (BoW)

1. Bag of Words: Collection and frequency of words



Drawbacks

1. Space inefficient

Drawbacks

1. Space inefficient
2. Word order is lost

Drawbacks

1. Space inefficient
2. Word order is lost
3. Doesn't capture language structure

Word Embeddings: idea

1. Learn embeddings from the words into vectors: $W(\text{word})$

Word Embeddings: idea

1. Learn embeddings from the words into vectors: $W(\text{word})$
2. Expecting that similar words fall nearby in the space

Word Embeddings

1. What is the dimension of the embedding?

Word Embeddings

1. What is the dimension of the embedding?
2. Trade-off: greater capacity vs. efficiency

Word Embeddings

1. Finding W: as a part of a prediction task that involves neighboring words

Word Embeddings: word2vec

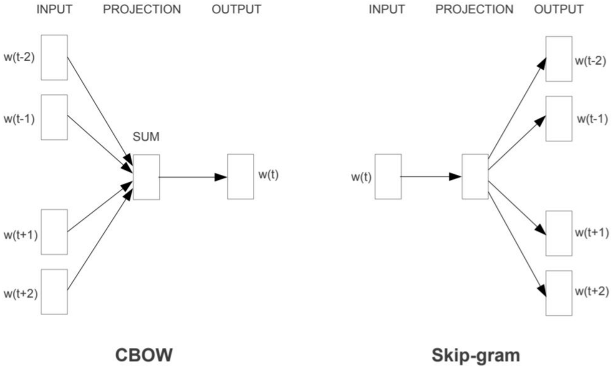
1. T Mikolov et al. (2013)

Word Embeddings: word2vec

1. T Mikolov et al. (2013)
2. Predict words from the context

Word Embeddings: word2vec

1. T Mikolov et al. (2013)
2. Predict words from the context
3. Two versions: Continuous Bag of Words (CBoW) and Skip-gram

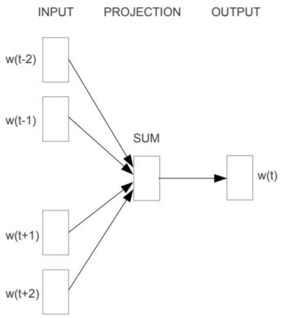


Word Embeddings: CBoW

1. Considers the embeddings of ‘n’ words before and ‘n’ words after the target word

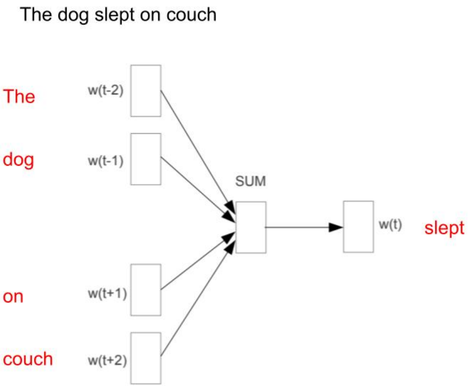
Word Embeddings: CBoW

1. Considers the embeddings of ‘n’ words before and ‘n’ words after the target word
2. Adds them (order is lost) for predicting the target word



CBOW

Word Embeddings: CBoW



Word Embeddings: CBoW

1. Size of the vocabulary $= V$

Word Embeddings: CBoW

1. Size of the vocabulary $= V$
2. Dimension of the embeddings $= N$

Word Embeddings: CBoW

1. Size of the vocabulary $= V$
2. Dimension of the embeddings $= N$
3. Input layer will have the weight matrix $W_{N \times V}$

Word Embeddings: CBoW

1. Size of the vocabulary $= V$
2. Dimension of the embeddings $= N$
3. Input layer will have the weight matrix $W_{N \times V}$
4. Projects the words in to N dimensional space

Word Embeddings: CBoW

1. Size of the vocabulary $= V$
2. Dimension of the embeddings $= N$
3. Input layer will have the weight matrix $W_{N \times V}$
4. Projects the words in to N dimensional space
5. Projections of all the $(2n)$ words in context are summed (result is an Nd vector)

Word Embeddings: CBoW

1. Next layer has a weight matrix $W'_{V \times N}$

Word Embeddings: CBoW

1. Next layer has a weight matrix $W'_{V \times N}$
2. Projects the accumulated embeddings onto the vocabulary

Word Embeddings: CBoW

1. Next layer has a weight matrix $W'_{V \times N}$
2. Projects the accumulated embeddings onto the vocabulary
3. That is, V- way classification $\rightarrow$ (after a softmax) maximizes the probability for the target word

Word Embeddings: CBoW

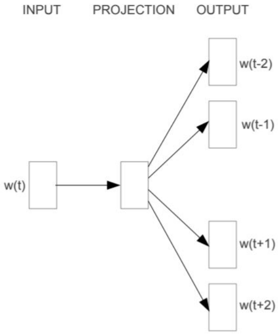
1. $W_{N \times V}$ or $W'_{V \times N}$ can be considered as the word embeddings

Word Embeddings: CBoW

1. $W_{N \times V}$ or $W'_{V \times N}$ can be considered as the word embeddings
2. Or, take the average of both the representations

Word Embeddings: Skipgram

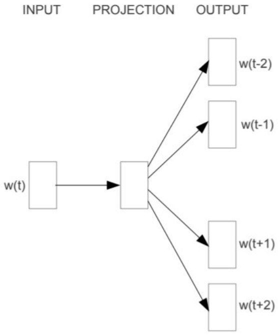
1. Predicts surrounding words given current word



Skip-gram

Word Embeddings: Skipgram

1. Predicts surrounding words given current word
2. Pick a word in the context randomly, and predict that word



Skip-gram

Word Embeddings: interesting results

1. W(Paris) - W(France) $+$ W(Italy) $=$ W(Rome)

Word Embeddings: interesting results

1. W(Paris) - W(France) + W(Italy) = W(Rome)
2. W(Man) - W(Woman) + W(King) = W(Queen)

Word Embeddings: Applications

1. Key for the success of many NLP tasks such as PoS tagging, parsing, semantic role labeling, etc.

Word Embeddings: Applications

1. Key for the success of many NLP tasks such as PoS tagging, parsing, semantic role labeling, etc.
2. Can serve projecting multi-modal data (e.g. multiple languages, images and text, etc.)

References

1. Mikolov, Tomas; et al. (2013). "Efficient Estimation of Word Representations in Vector Space". arXiv:1301.3781