

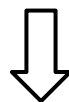
# Word Embedding

EECS 16ML

# Problem in Text Learning

- Computers can't read
- Messy structure
- Too much information (semantics, grammar, etc.)

Our goal: represent text in a way that computers can read but at the same time retaining rich information included in text



**Use numbers to capture information in text.**  
But how?

# Bag of Words

- Count-based
- Measure frequency of word occurrences
- Documents are “close” in the vector space if both used similar words for similar times

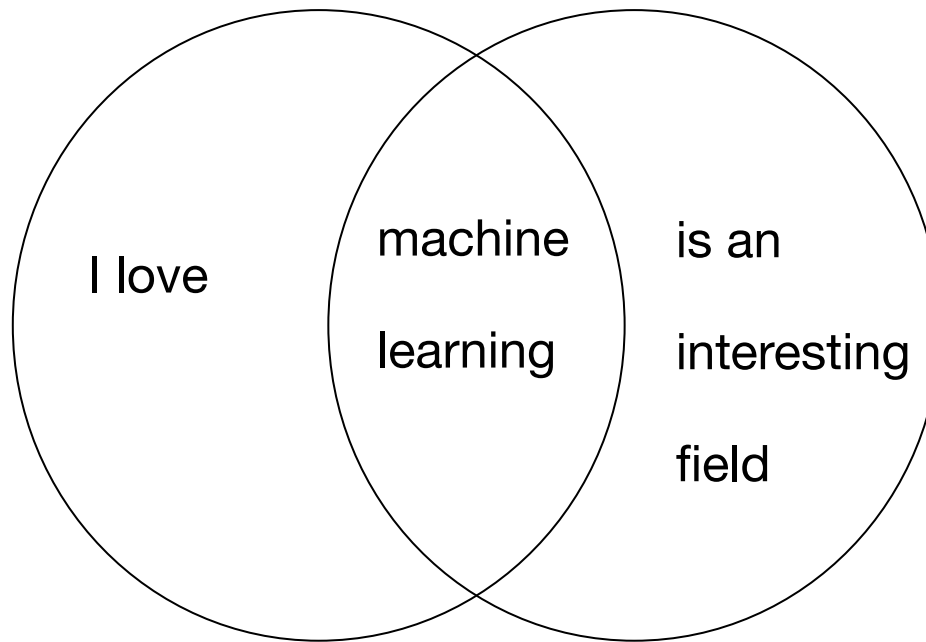
# Bag of Words

Consider three sentences:

- a. I love machine learning!
- b. Machine learning is an interesting field.
- c. I love learning machine learning!

	I	love	machine	learning	is	an	interesting	field
a	1	1	1	1	0	0	0	0
b	0	0	1	1	1	1	1	1
c	1	1	1	2	0	0	0	0

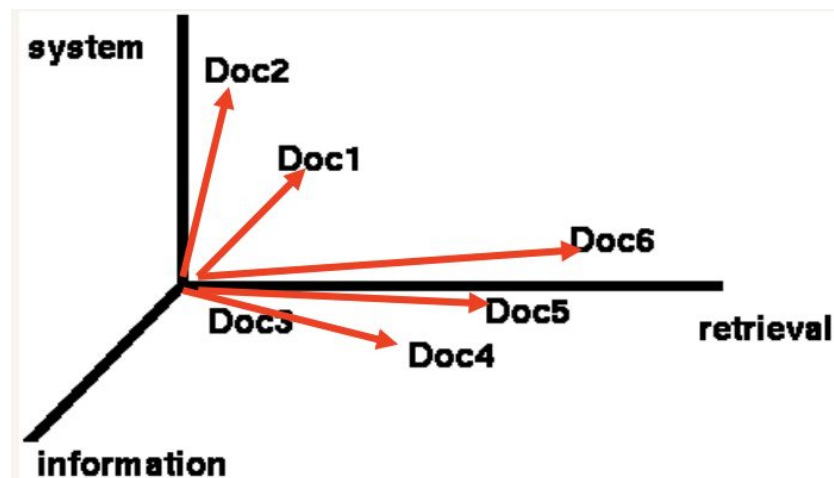
# Are the vectors representative?



Need some notion of the importance of words

# Another problem

- Terms are axes of the space
- Documents are vectors in this space
- High dimensional vectors
  - contain many 0s
  - very sparse vector



# TF-IDF

- Stands for “term frequency - inverse document frequency”
- Based on two scores
  - Term frequency
  - Inverse document frequency

# Term Frequency

- A measure of how frequently a word appears in a document
- Each document and term would have its own TF score

$$tf_{t,d} = \frac{n_{t,d}}{\text{Number of terms in the document}}$$

- Review: This movie is very scary and is long.
  - $TF(\text{'scary'}) = 1/7$ ,  $TF(\text{'is'}) = 2/7$
  - $TF(\text{'good'}) = 0/7$
- Rare terms are more informative than frequent terms
  - like word “scary” can show the attitude of this reviewer on this movie.
- How to give a high weight on rare terms?



# Inverse Document Frequency

- A measure of how important a term is to a document.
- Terms that occur in many documents are weighted less and vice versa
  - Like word “the” should have low IDF score
- We add logarithmic transformation to dampen the effect of idf
  - If there are 100 million documents and the number of docs with the given word is 10, then IDF is  $100M/10 = 10M$ , which is much larger than TF score.

$$idf_t = \log \frac{\text{number of documents}}{\text{number of documents with term 't'}}$$

# Example of IDF

Review 1: This course is very boring.

Review 2: This course is really intense.

Review 3: This course is not boring and meaningful.

- $IDF(course) = \log(3/3) = 0$
- $IDF(boring) = \log(3/2) = 0.18$
- $IDF(meaningful) = \log(3/1) = 0.48$

We see that words like “is”, “this” are reduced to 0 and have little importance; while words like “meaningful”, “intense” are words with more importance and thus have a higher value.

# Put it Together

- The tf-idf score of word  $t$  in document  $d$  is

$$(tf\_idf)_{t,d} = tf_{t,d} * idf_t$$

# Example of TF-IDF

Review 1: This course is very boring.

Review 2: This course is really intense.

Review 3: This course is not boring and meaningful.

- $\text{TF-IDF}(\text{course}, \text{review2}) = 1/5 * \log(3/3) = 0$
- $\text{TF-IDF}(\text{boring}, \text{review1}) = 1/5 * \log(3/2) = 0.035$
- $\text{TF-IDF}(\text{meaningful}, \text{review3}) = 1/7 * \log(3/1) = 0.068$ 
  - unique identifier for review3

TF-IDF also gives larger values for unique words and is high when both IDF and TF scores are high i.e the word is rare in all the documents but frequent in a single document.

# Problem

- TF-IDF is based on the bag-of-words (BoW) model
  - Does not capture position in text
  - Does not capture semantics
  - Does not capture co-occurrences in different documents
- Need more advanced methods

# Co-Occurrence Matrix

- Count number of co-occurrences of words
- Word representations in vector form by applying PCA/SVD
- Consider again these three sentences:
  - a. I love machine learning!
  - b. Machine learning is an interesting field.
  - c. I love learning machine learning!

# Co-Occurrence Matrix

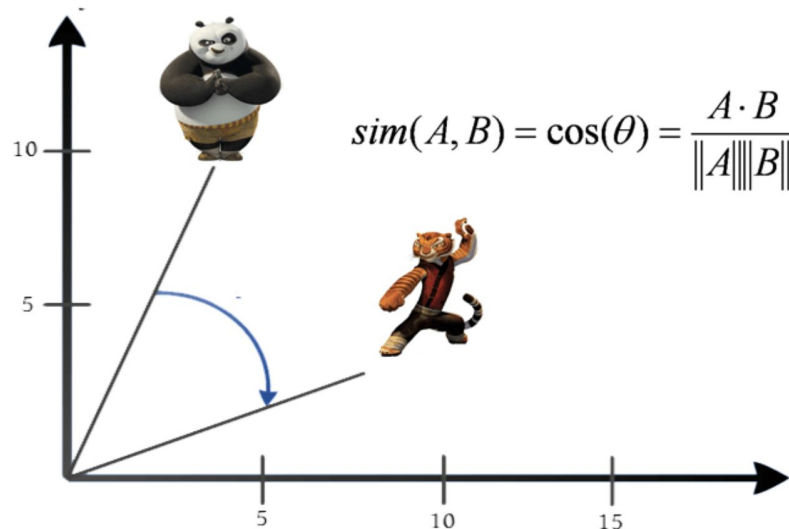
Co-occurrence matrix with window size = 2:

	I	love	machine	learning	is	an	interesting	field
I	0	2	1	1	0	0	0	0
love	2	0	2	2	0	0	0	0
machine	1	2	0	3	1	0	0	0
learning	1	2	3	1	1	1	0	0
is	0	0	1	1	0	1	1	0
an	0	0	0	1	1	0	1	1
interesting	0	0	0	0	1	1	0	1
field	0	0	0	0	0	1	1	0

# Word2Vec

- Distributed representations
- Predictive model, not count-based anymore
- Maximize the similarity of word representation of the context words to the center word

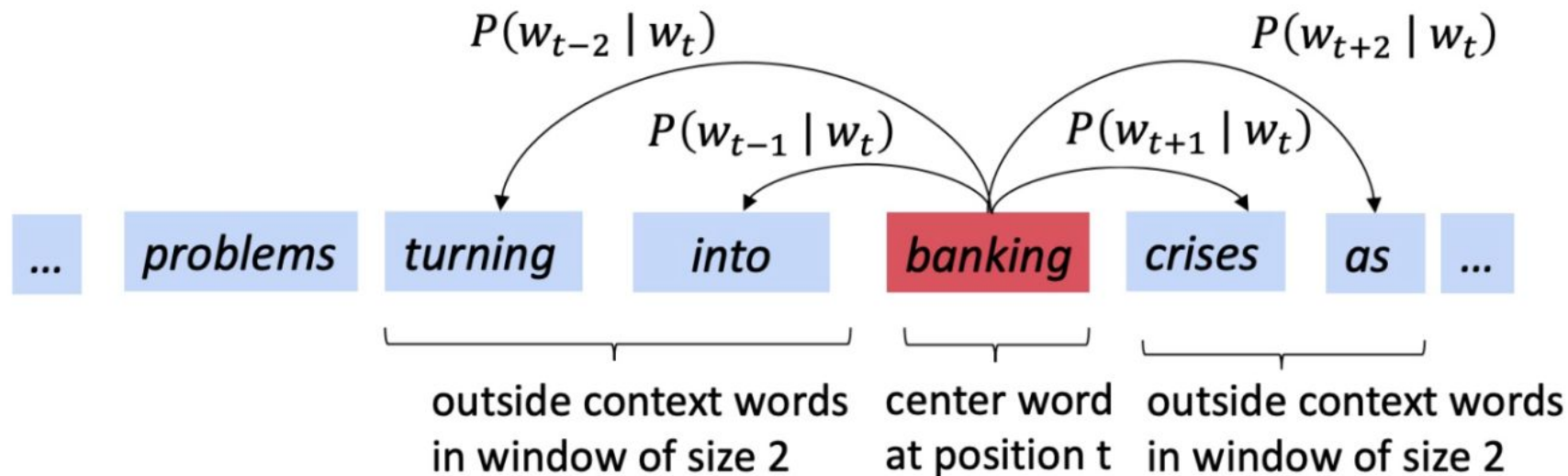
## Cosine Similarity



[Google Images](#)



## Context window



Source: Stanford 224N

# Word2Vec

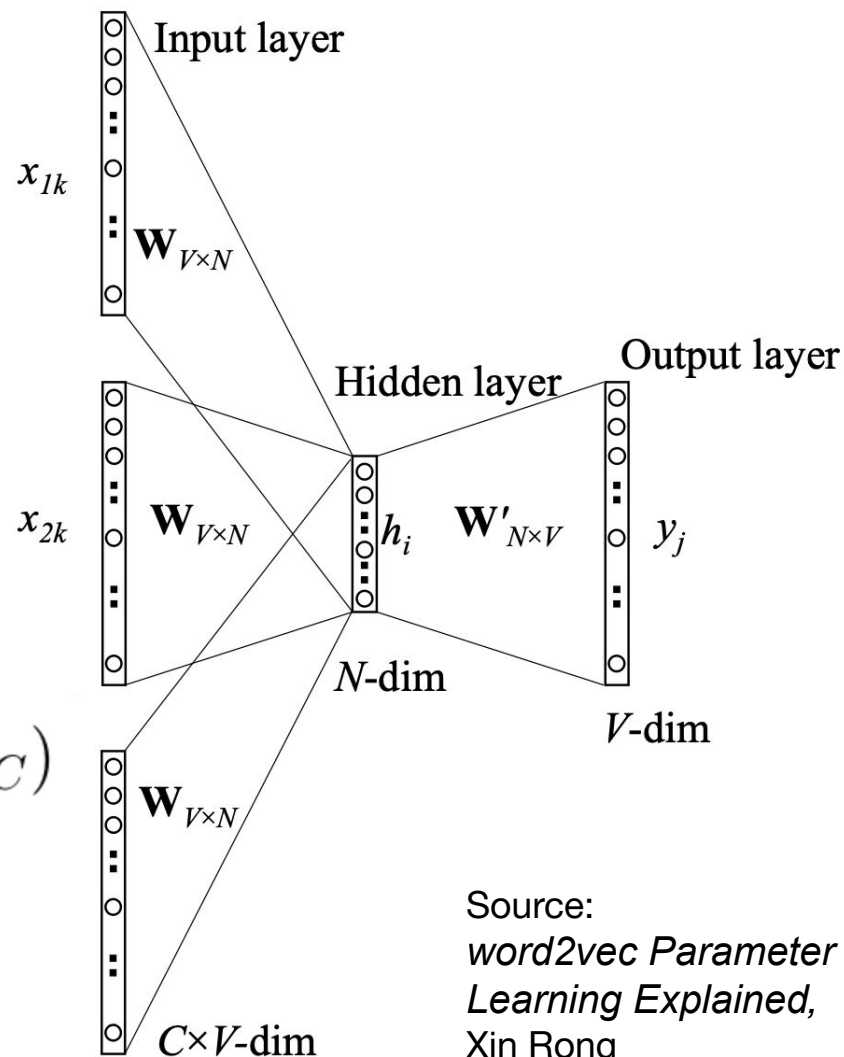
## Continuous Bag of Words

- Input: context word(s)
- Predict: center word

## Objective:

- Minimize

$$\begin{aligned}
 E &= -\log p(W_O | W_{I,1}, \dots, W_{I,C}) \\
 &= -\log \frac{e^{u_{j^*}}}{\sum_{j'=1}^V e^{u_{j'}}}
 \end{aligned}$$



# Word2Vec

## Skip Gram

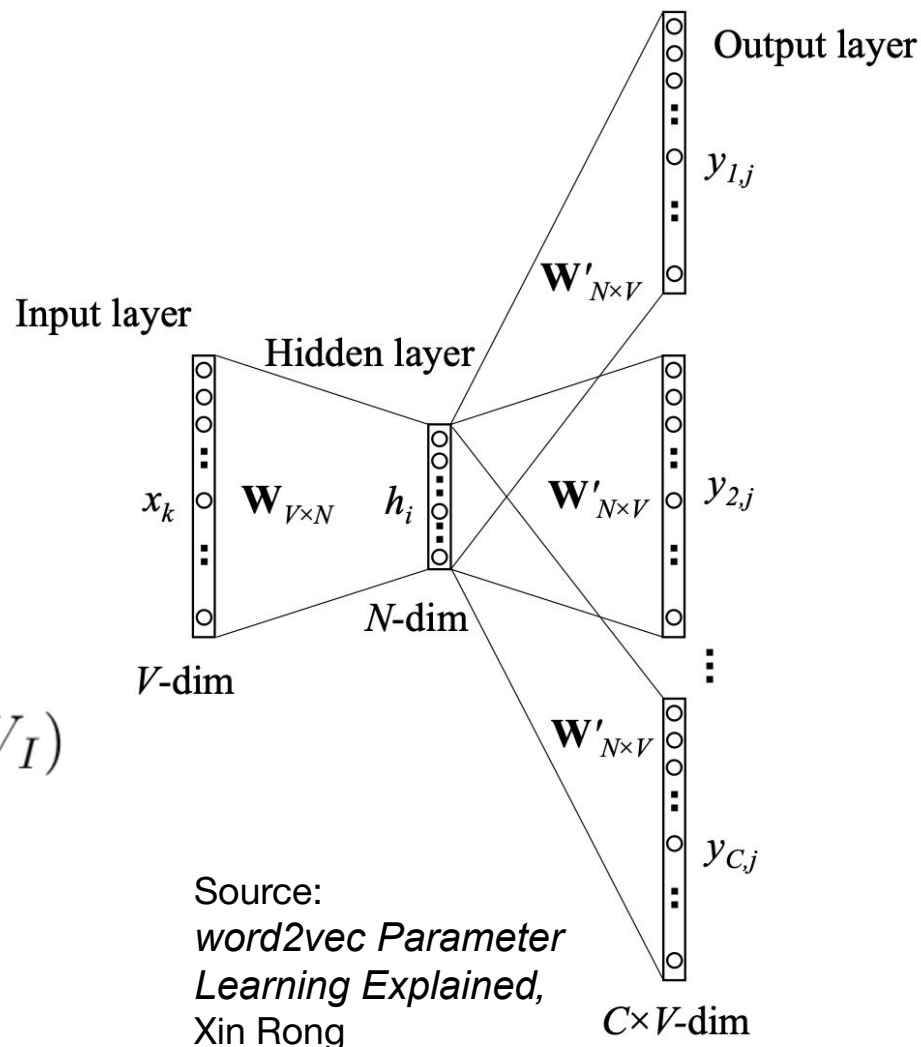
- Input: center word
- Predict: context word(s)

## Objective:

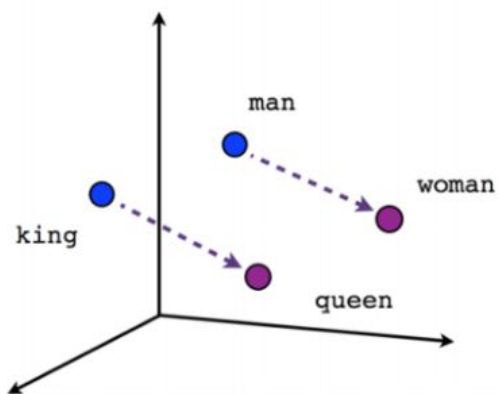
- Minimize

$$E = -\log p(W_{O,1}, \dots, W_{O,C} | W_I)$$

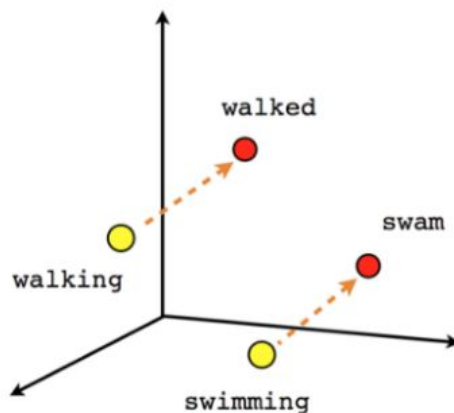
$$= -\log \prod_{c=1}^C \frac{e^{u_{jc}^*}}{\sum_{j'=1}^V e^{u_{j'}}}$$



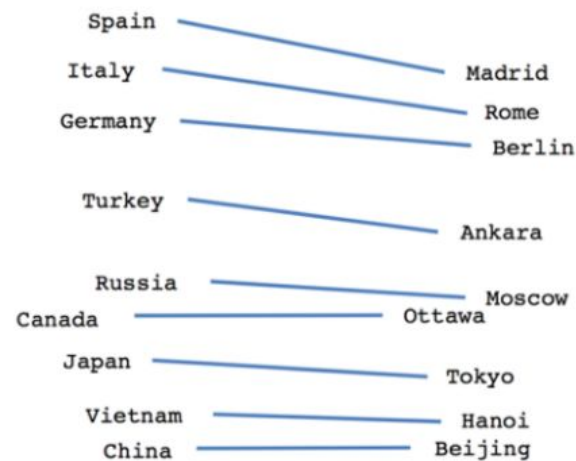
# Word2Vec



Male-Female



Verb tense



Country-Capital

<https://medium.com/data-science-group-iitr/word-embedding-2d05d270b285>

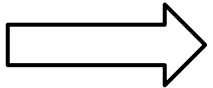
- Word vector representation
- Combine both count-based and predictive approaches
- “Global statistics”
- Pre-trained vectors available (various dim. and corpus)

Input: Co-occurrence matrix  $G$  (normalized, log-smoothed)

Objective:

• Minimize

$$L = \sum_{i=1}^V \sum_{j=1}^V f(G_{ij})(w_i^T w_j + bias - G_{ij})^2$$


 $w_i^T w_j + bias \approx G_{ij}$