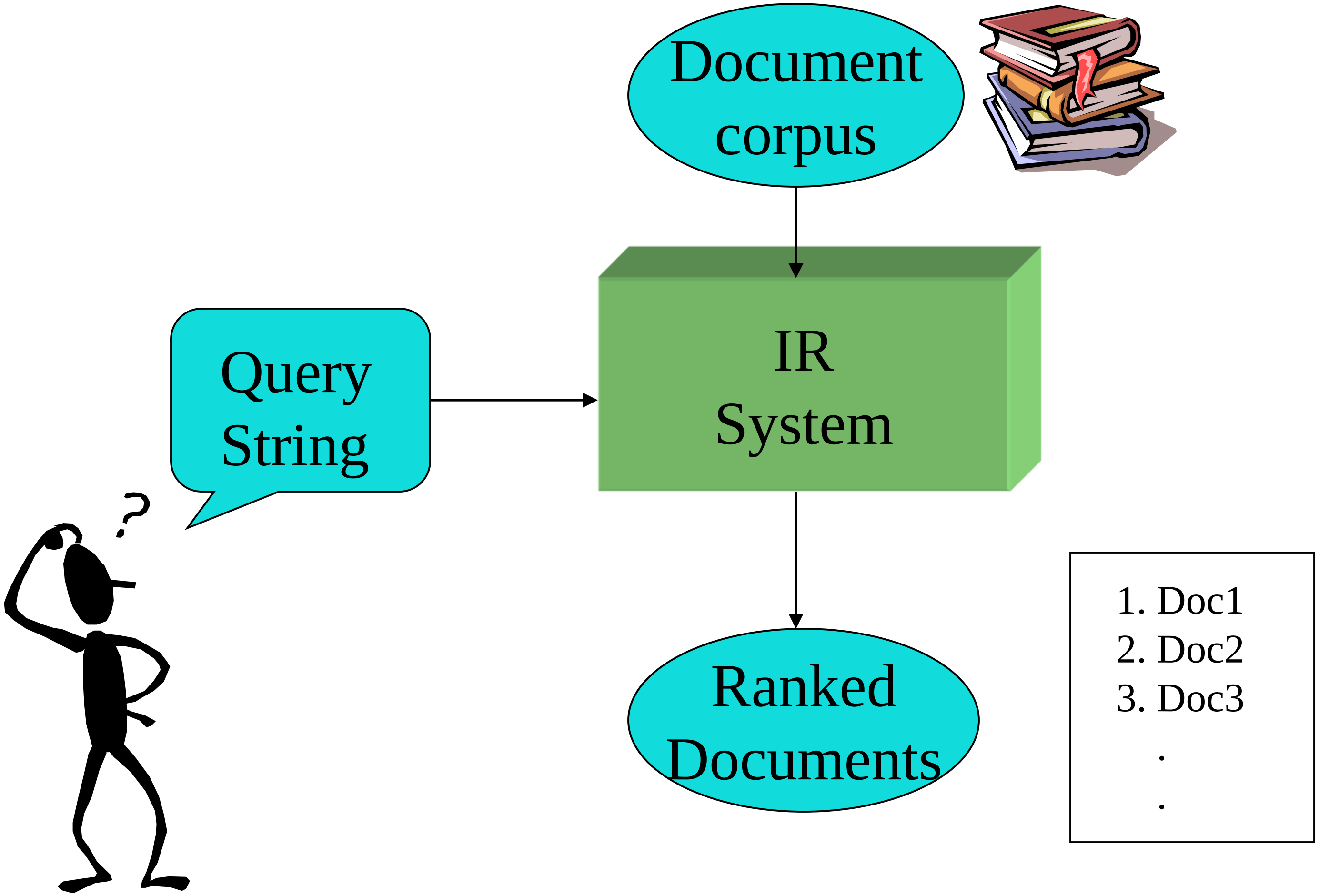
Vector-Space (Distributional) Lexical Semantics

Information Retrieval System



The Vector-Space Model

- Assume *t* distinct terms remain after preprocessing; call them index terms or the vocabulary.
- These "orthogonal" terms form a vector space.

  Dimension = t = |vocabulary|

- Each term, *i*, in a document or query, *j*, is given a real-valued weight, w_{ij}.
- Both documents and queries are expressed as *t*-dimensional vectors:

  $d_j = (w_{1j}, w_{2j}, \ldots, w_{tj})$

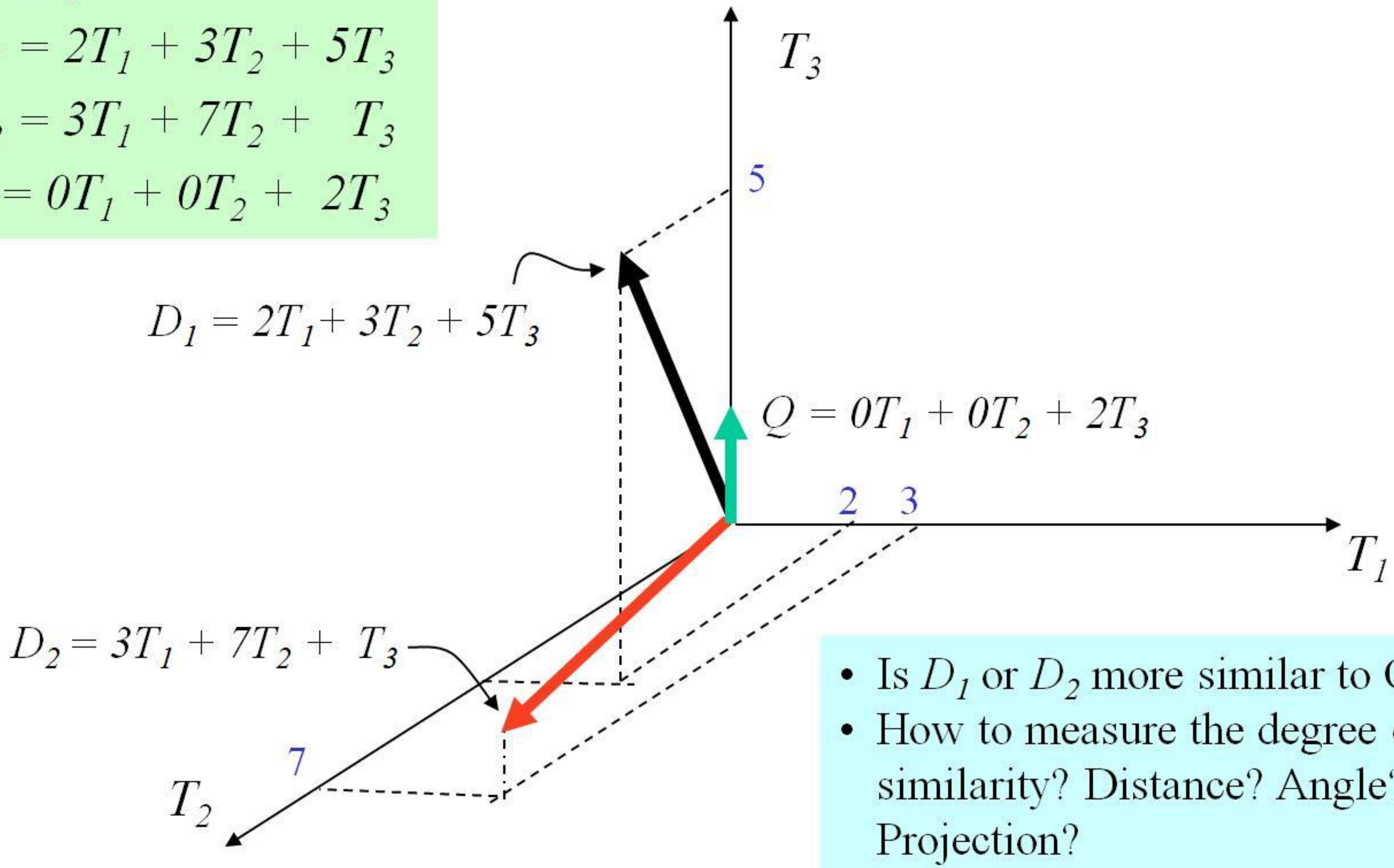
Graphic Representation

Example:

$D_1 = 2T_1 + 3T_2 + 5T_3$

$D_2 = 3T_1 + 7T_2 + T_3$

$Q = 0T_1 + 0T_2 + 2T_3$



- Is D_1 or D_2 more similar to Q?
- How to measure the degree of similarity? Distance? Angle? Projection?

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

  f_{ij} = frequency of term i in document j

- May want to normalize *term frequency* (*tf*) by dividing by the frequency of the most common term in the document:

  $tf_{ij} = f_{ij} / max_i\{f_{ij}\}$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

  df_i = document frequency of term i

  = number of documents containing term i

  idf_i = inverse document frequency of term i,

  $= \log_2 (N / df_i)$

  (N: total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to *tf*.

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N/ df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

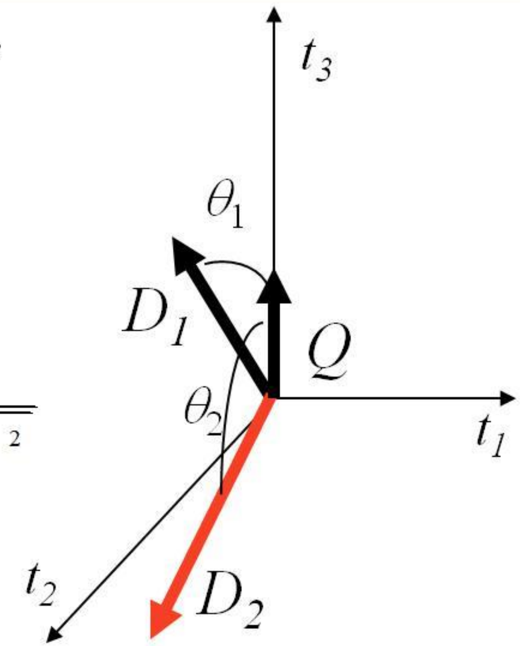
Similarity Measure

- A similarity measure is a function that computes the *degree of similarity* between two vectors.

- Using a similarity measure between the query and each document:
  - It is possible to rank the retrieved documents in the order of presumed relevance.
  - It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.

Cosine Similarity Measure

- Cosine similarity measures the cosine of the angle between two vectors.
- Inner product normalized by the vector lengths.



$$\text{CosSim}(\boldsymbol{d}_j, \boldsymbol{q}) = \frac{\vec{d}_j \cdot \vec{q}}{\left|\vec{d}_j\right| \cdot \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$

$D_1 = 2T_1 + 3T_2 + 5T_3$ CosSim(D_1 , Q) = $10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$

$D_2 = 3T_1 + 7T_2 + 1T_3$ CosSim(D_2 , Q) = $2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$

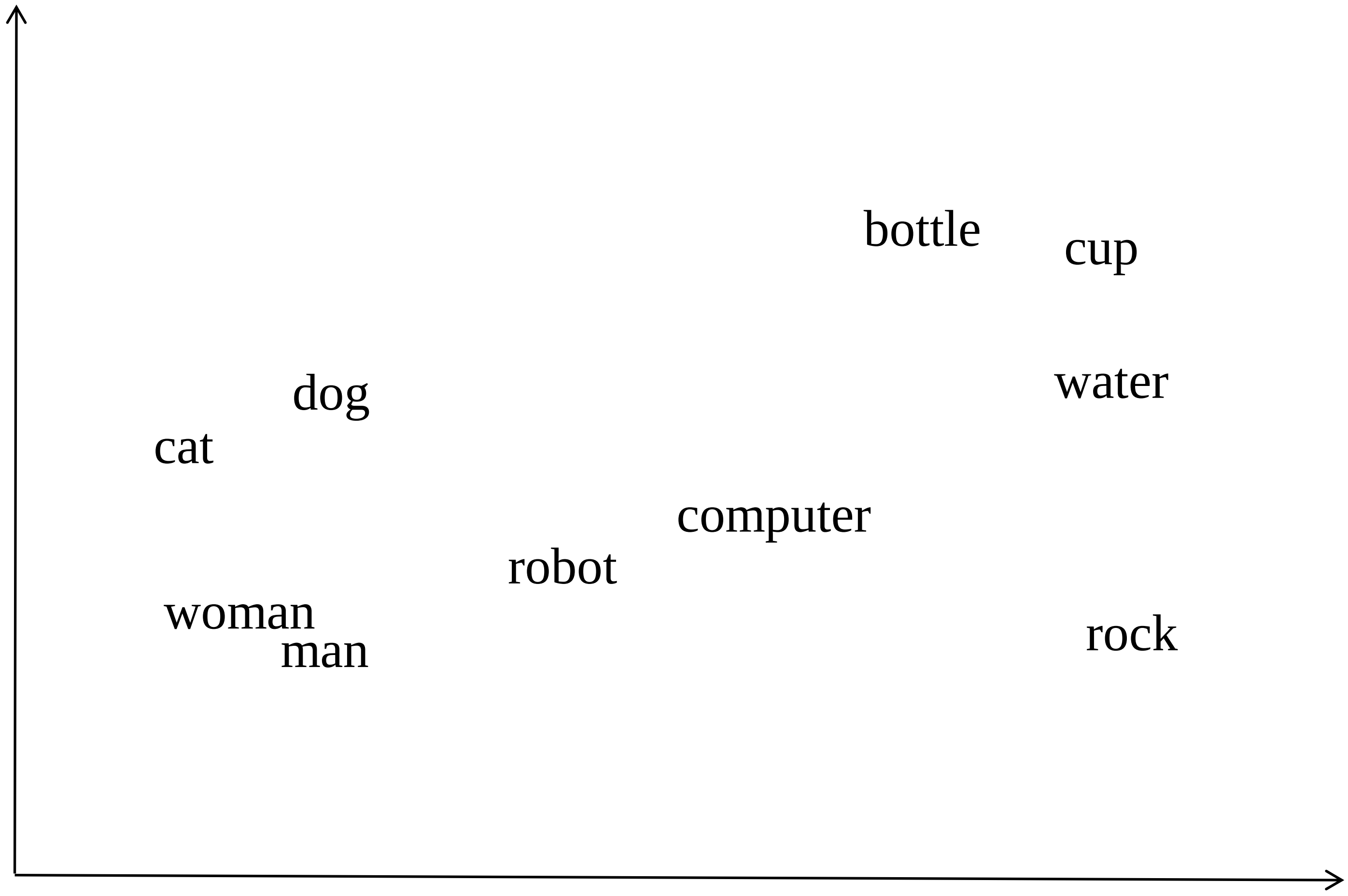
$Q = 0T_1 + 0T_2 + 2T_3$

D_1 is 6 times better than D_2 using cosine similarity but only 5 times better using inner product.

Vector-Space (Distributional) Lexical Semantics

- Represent word meanings as points (vectors) in a (high-dimensional) Euclidian space.
- Dimensions encode aspects of the context in which the word appears (e.g. how often it co-occurs with another specific word).
  - "You will know a word by the company that it keeps." (J.R. Firth, 1957)
- Semantic similarity defined as distance between points in this semantic space.

Sample Lexical Vector Space



Simple Word Vectors

- For a given target word, *w*, create a bag-of-words "document" of all of the words that co-occur with the target word in a large corpus.
  - Window of *k* words on either side.
  - All words in the sentence, paragraph, or document.
- For each word, create a (tf-idf weighted) vector from the "document" for that word.
- Compute semantic relatedness of words as the cosine similarity of their vectors.

Other Contextual Features

- Use syntax to move beyond simple bag-of-words features.
- Produced typed (edge-labeled) dependency parses for each sentence in a large corpus.
- For each target word, produce features for it having specific dependency links to specific other words (e.g. subj=dog, obj=food, mod=red)

Other Feature Weights

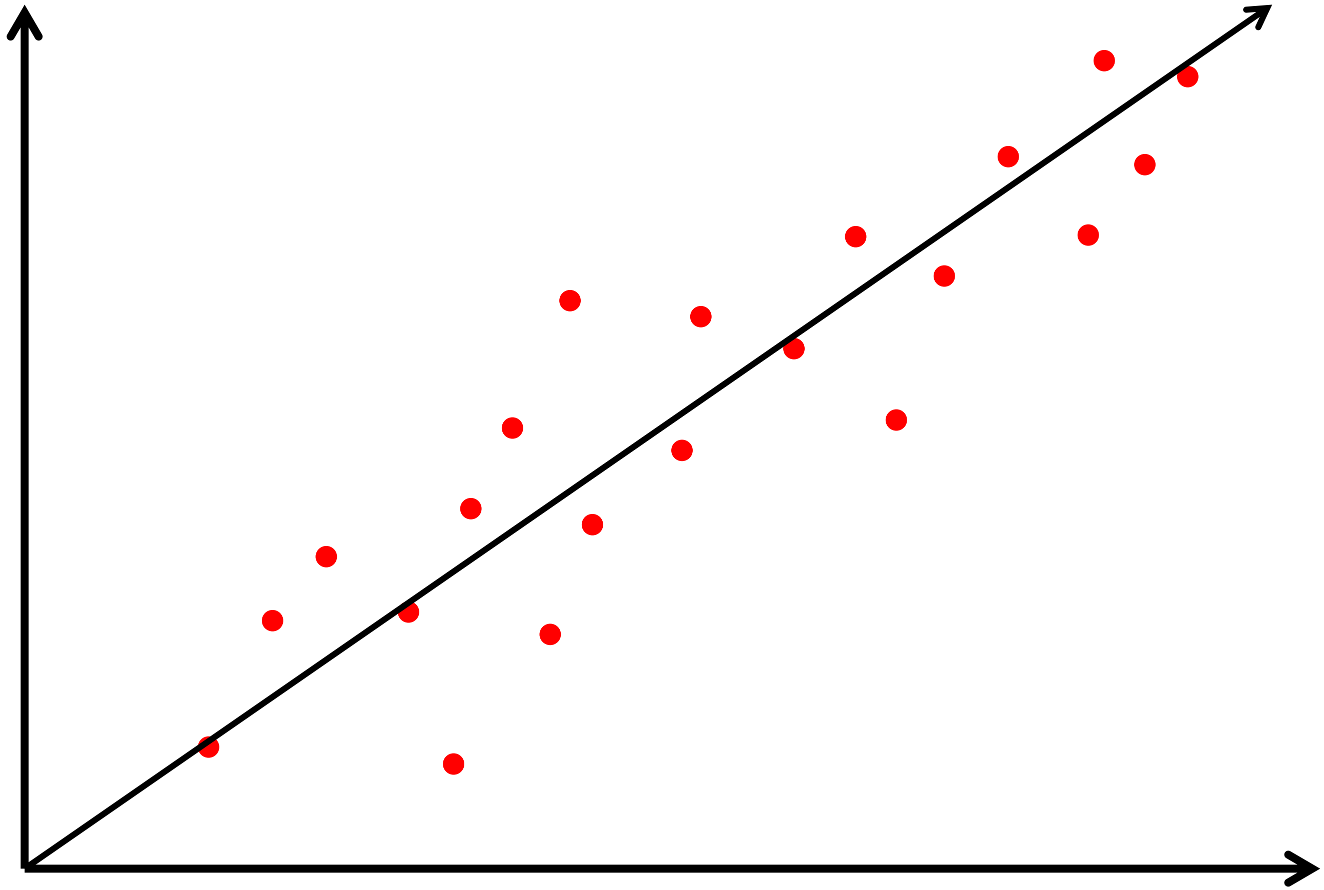
- Replace TF-IDF with other feature weights.
- *Pointwise mutual information* (PMI) between target word, *w*, and the given feature, *f*:

$$PMI(f,w) = log \frac{P(f,w)}{P(f)p(w)}$$

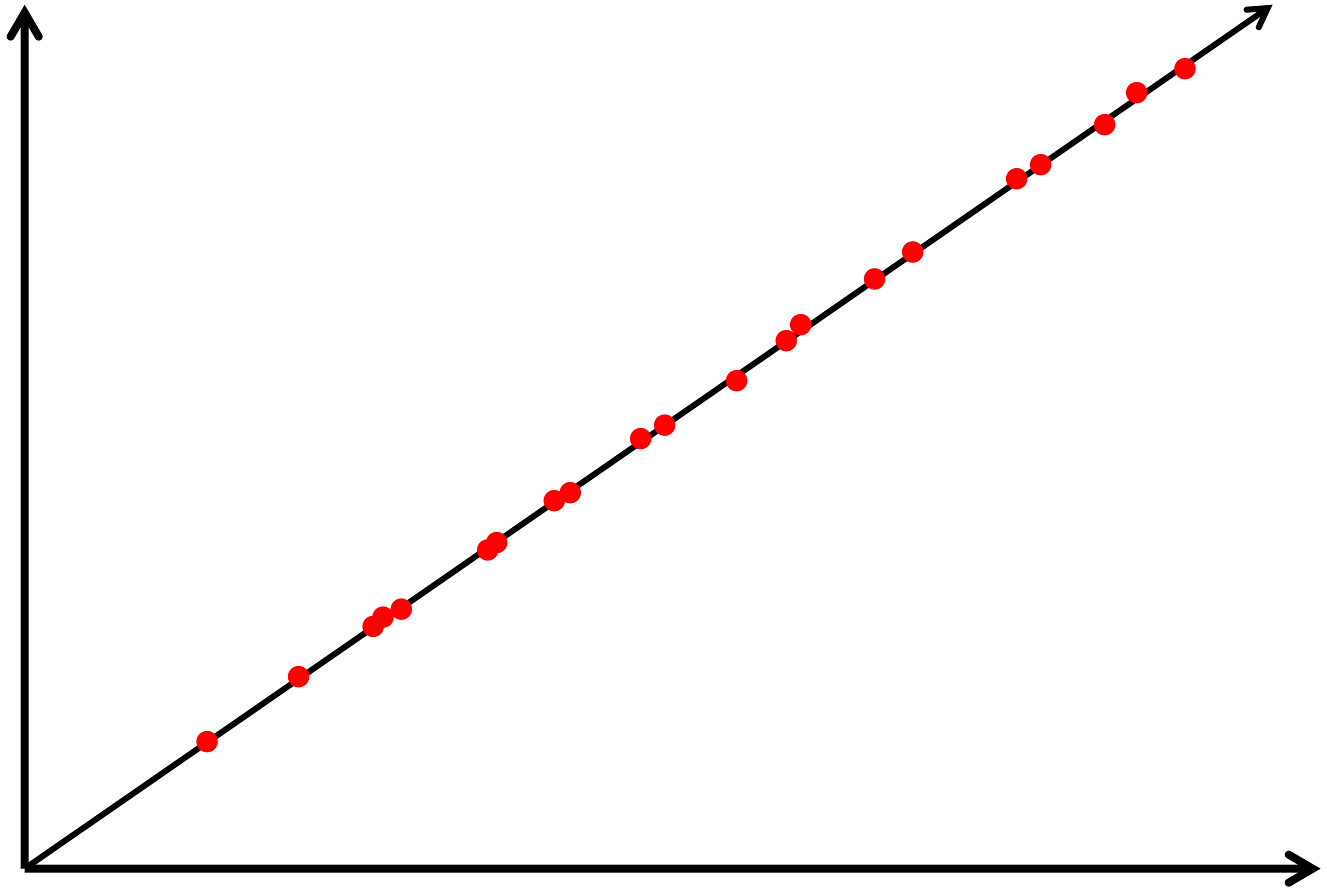
Dimensionality Reduction

- Word-based features result in extremely high-dimensional spaces that can easily result in over-fitting.
- Reduce the dimensionality of the space by using various mathematical techniques to create a smaller set of *k* new dimensions that most account for the variance in the data.
  - Singular Value Decomposition (SVD) used in Latent Semantic Analysis (LSA)
  - Principle Component Analysis (PCA)

Sample Dimensionality Reduction

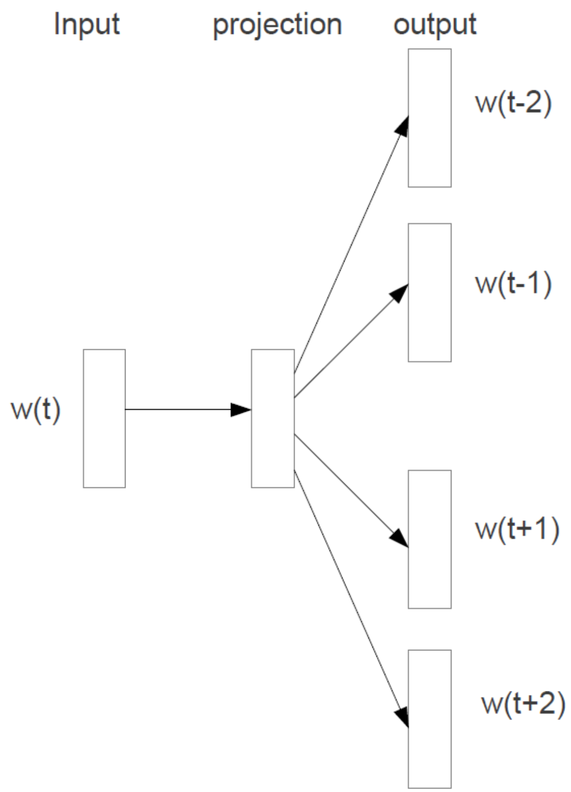


Sample Dimensionality Reduction



Neural Word2Vec
(Mikolov et al., 2013)

- Learn an "embedding" of words that supports effective prediction of surrounding "skip gram" of words.



Skip-Gram Word2Vec Network Architecture

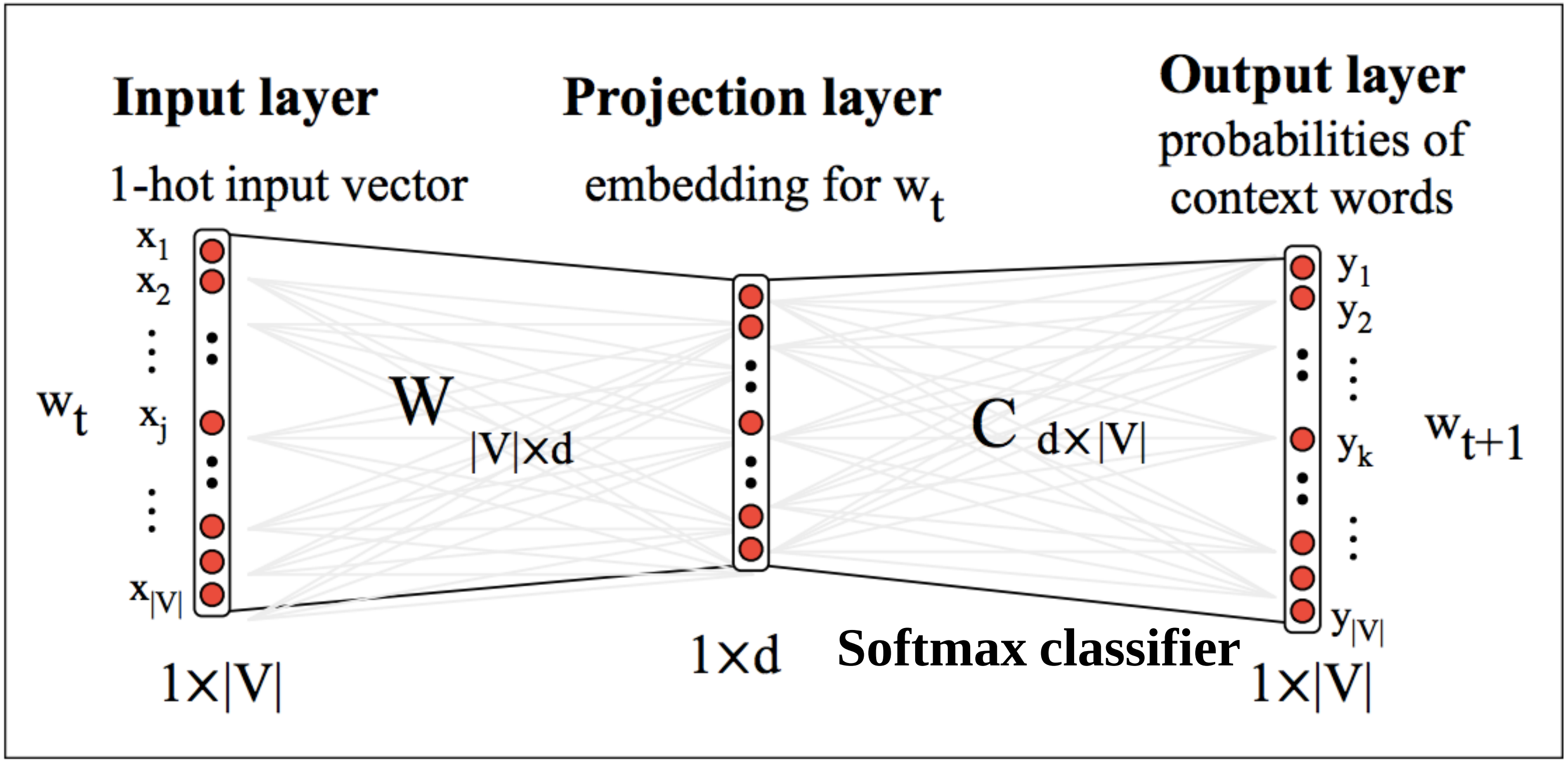


Figure 16.5 The skip-gram model viewed as a network (Mikolov et al. 2013, Mikolov et al. 2013a).

Word2Vec Math

- Softmax classifier predicts surrounding words from a word embedding.

$$\log p(o|c) = \log \frac{\exp\left(u_o^T v_c\right)}{\sum_{w=1}^{W} \exp\left(u_w^T v_c\right)}$$

- Train to maximize the probability of skip-gram predictions.

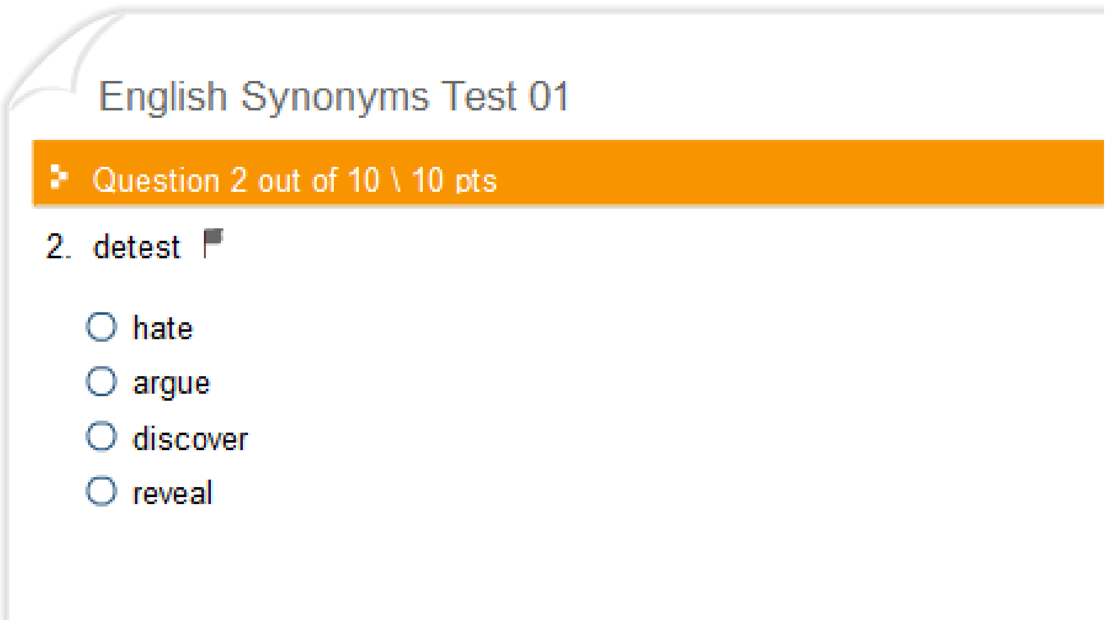
$$J(\theta) = \frac{1}{T} \sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{t+j} | w_t)$$

Evaluation of Vector-Space Lexical Semantics

- Have humans rate the semantic similarity of a large set of word pairs.
  – (dog, canine): 10; (dog, cat): 7; (dog, carrot): 3; (dog, knife): 1
- Compute vector-space similarity of each pair.
- Compute correlation coefficient (Pearson or Spearman) between human and machine ratings.

TOEFL Synonymy Test

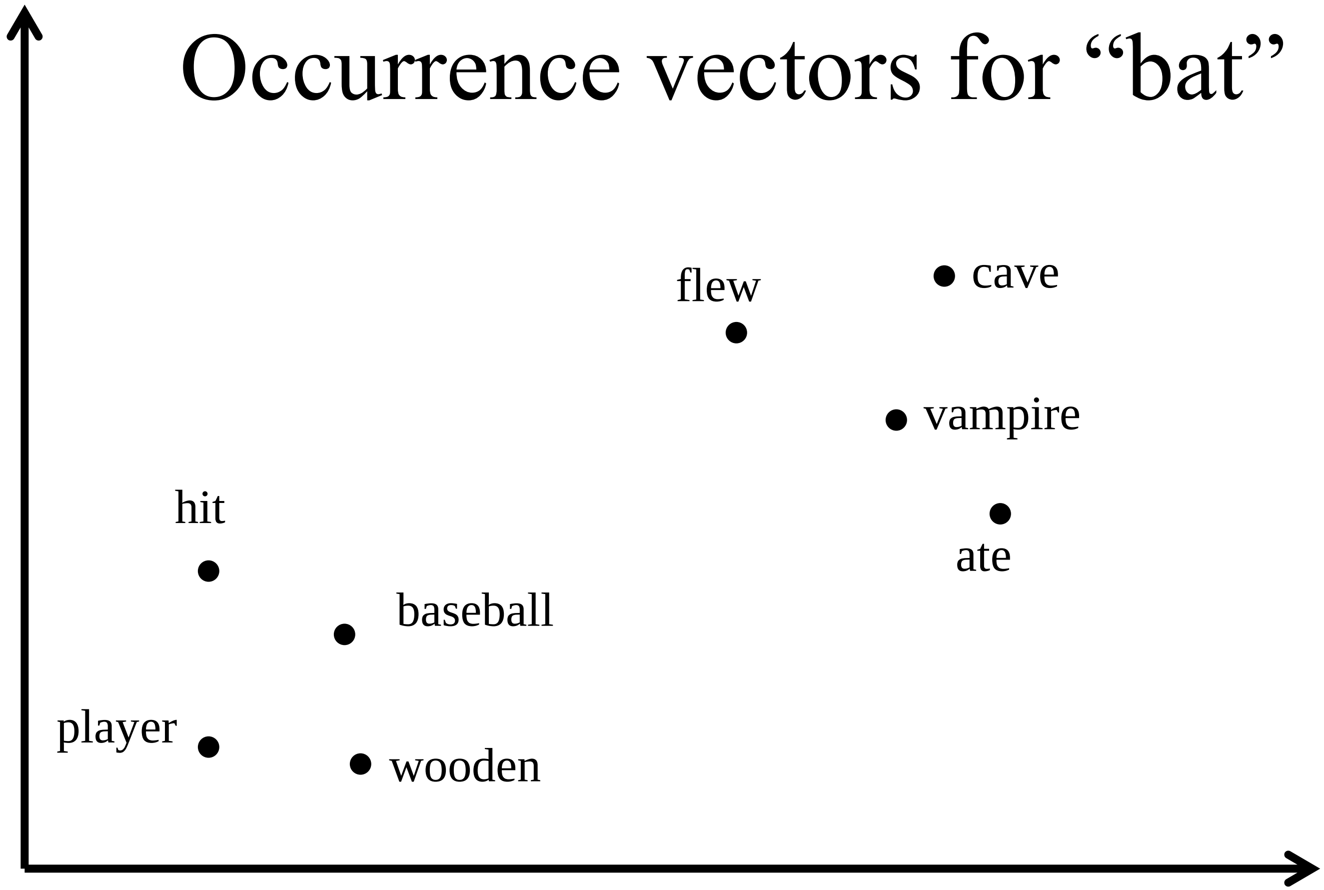
- LSA shown to be able to pass TOEFL synonymy test.



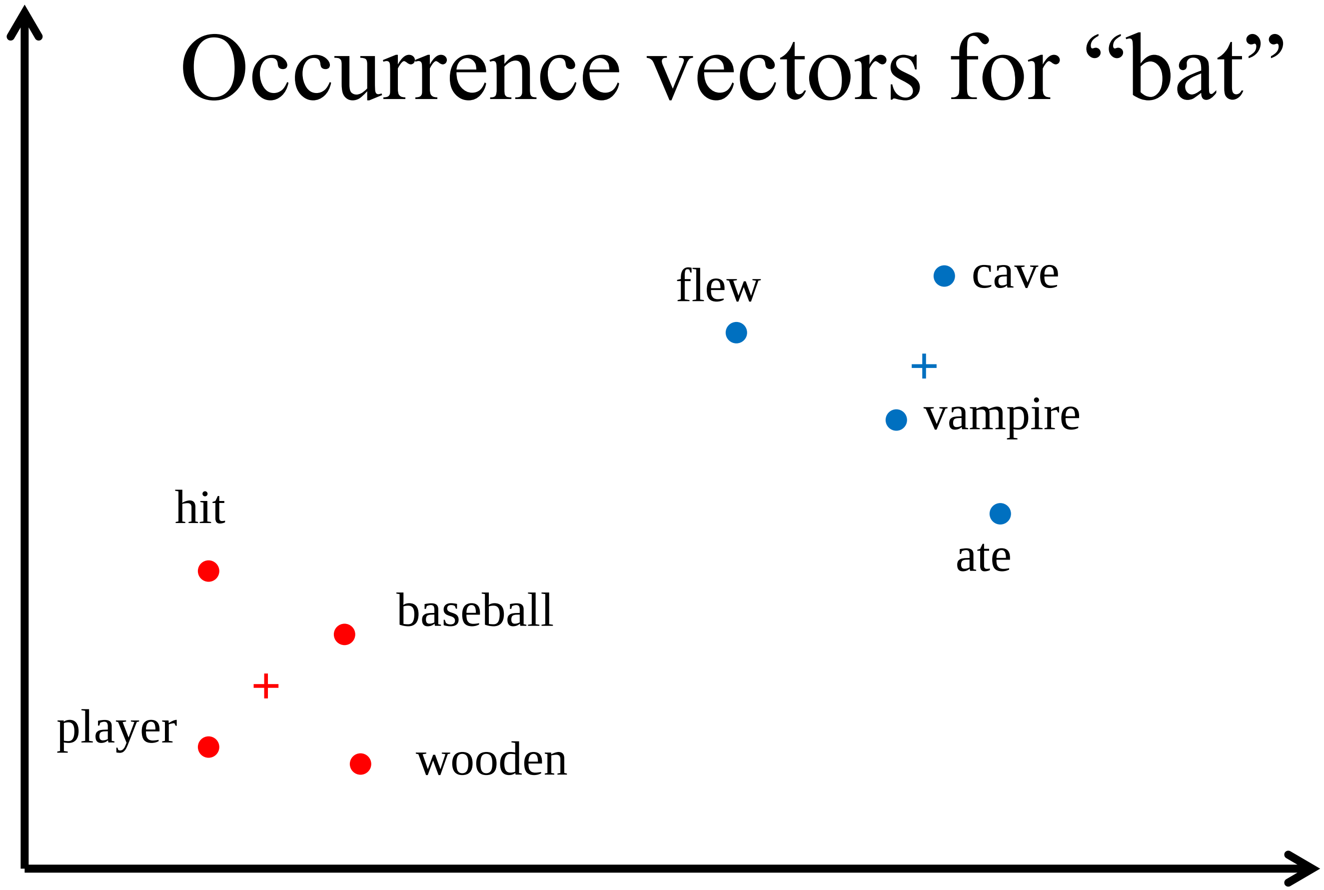
Vector-Space Word Sense Induction (WSI)

- Create a context-vector for ***each individual occurrence*** of the target word, *w*.
- Cluster these vectors into *k* groups.
- Assume each group represents a “sense” of the word and compute a vector for this sense by taking the mean of each cluster.

Sample Word Sense Induction

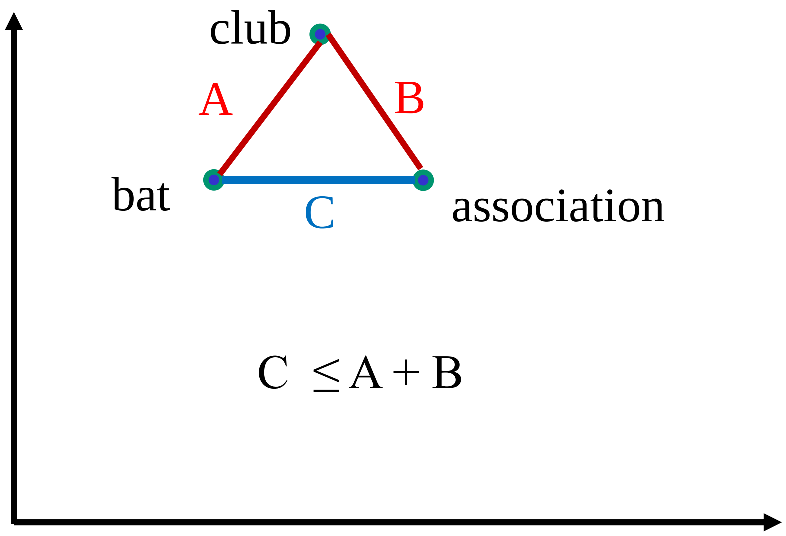


Sample Word Sense Induction



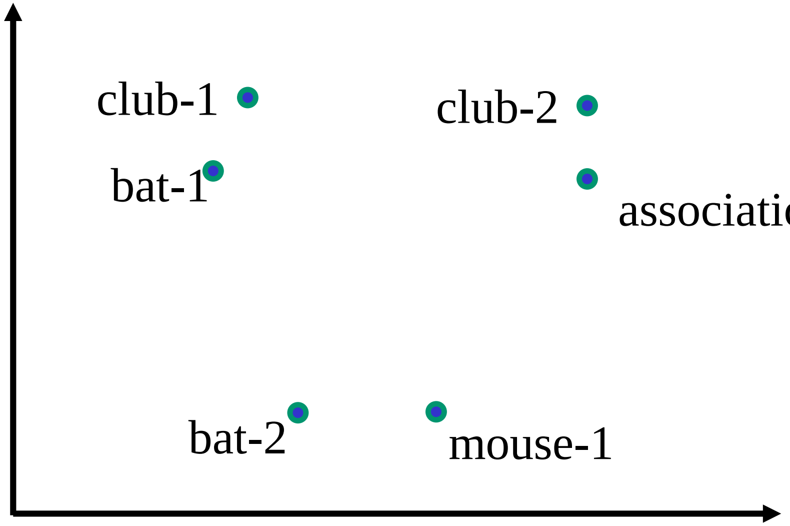
Word Sense and Vector Semantics

- Having one vector per word ignores the impact of homonymous senses.
- Similarity of ambiguous words violates the triangle inequality.



Multi-Prototype Vector Space Models (Reisinger & Mooney, 2010)

- Do WSI and create a multiple sense-specific vectors for ambiguous words.
- Similarity of two words is the maximum similarity of sense vectors of each.



Vector-Space Word Meaning in Context

- Compute a semantic vector for an individual occurrence of a word based on its context.
- Combine a standard vector for a word with vectors representing the immediate context.

Example Using Dependency Context

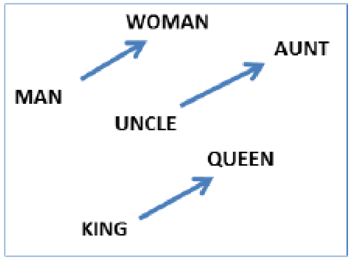
- Compute vector for nsubj-boss by summing contextual vectors for all word occurrences that have “boss” as a subject.
- Compute vector for dobj-secretary by summing contextual vectors for all word occurrences that have “secretary” as a direct object.
- Compute “in context” vector for “fire” in “boss fired secretary” by adding nsubj-boss and dobj-secretary vectors to the general vector for “fire”

Compositional Vector Semantics

- Compute vector meanings of phrases and sentences by combining (composing) the vector meanings of its words.
- Simplest approach is to use vector addition or component-wise multiplication to combine word vectors.
- Evaluate on human judgements of sentence-level semantic similarity (*semantic textual similarity*, STS, SemEval competition).

Other Vector Semantics Computations

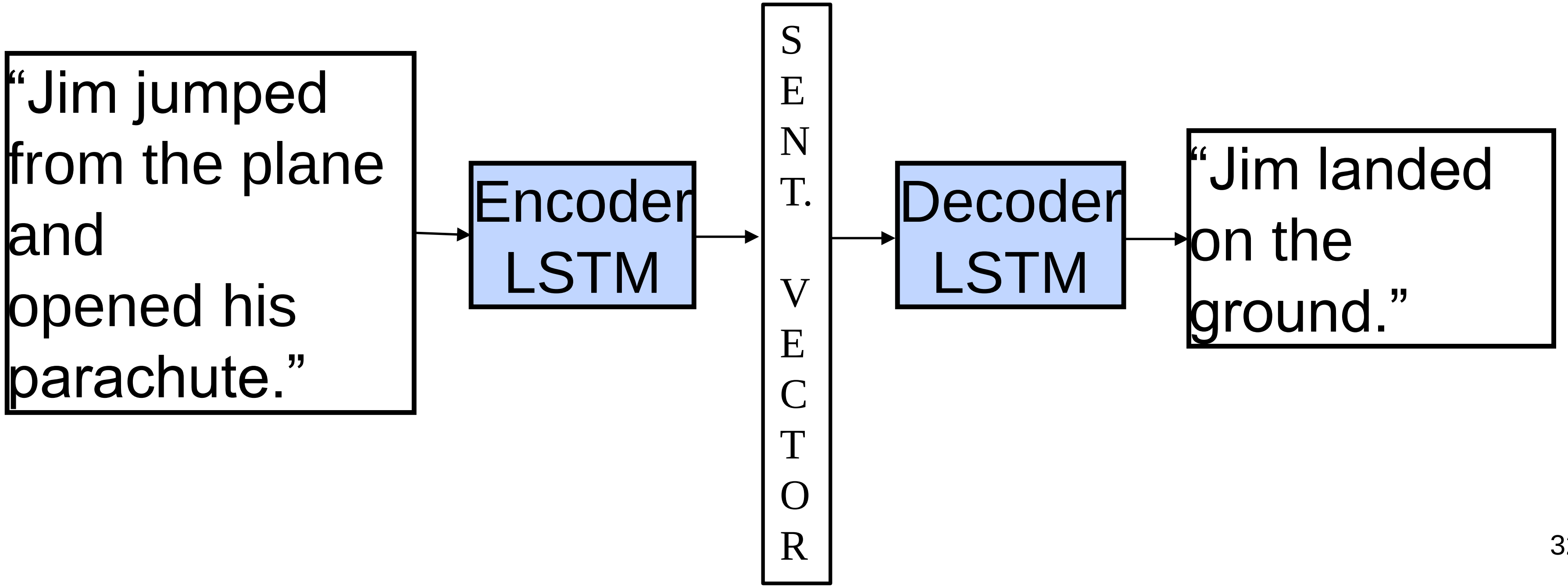
- Compute meanings of words by mathematically combining meanings of other words (Mikolov, et al., 2013)
  - $\overrightarrow{king} = \overrightarrow{queen} - \overrightarrow{female} + \overrightarrow{male}$



- Evaluate on solving word analogies
  - King is to queen as uncle is to ______?

Sentence-Level Neural Language Models

- "Skip-Thought Vectors" (Kiros et al., NIPS 2015)
  - Use LSTMs to encode whole sentences into lower-dimensional vectors.
  - Vectors trained to predict previous and next sentences.



Conclusions

- A word's meaning can be represented as a vector that encodes distributional information about the contexts in which the word tends to occur.
- Lexical semantic similarity can be judged by comparing vectors (e.g. cosine similarity).
- Vector-based word senses can be automatically induced by clustering contexts.
- Contextualized vectors for word meaning can be constructed by combining lexical and contextual vectors.