Distributional Semantics

Sean Simpson
(some slides from Marine Carpuat, Nathan Schneider)
(last updated by Tatsuya Aoyama, March 2024)

ENLP Lecture 13
March 11, 2025

Topics

- **Lexical Semantics**
  - → Word Similarity
  - → Distributional Hypothesis
  - → Vector Representations
  - → Evaluation
- **Document Semantics**
  - → Topic Modeling

Lexical Semantics

Semantic similarity: Intuition

- **Identify word closest to target:**

- **Accidental**
  - → Abominate
  - → Meander
  - → Inadvertent
  - → inhibit

Semantic similarity: Intuition

- Identify word closest to target:

- Accidental
  - Abominate
  - Meander
  - Inadvertent
  - inhibit

Semantic similarity: Intuition

- Identify word closest to target:

- **FedEx**
  - ➔ car
  - ➔ UPS
  - ➔ rotate
  - ➔ Navy

Semantic similarity: Intuition

- Identify word closest to target:

- FedEx
  - car
  - UPS
  - rotate
  - Navy

Semantic similarity: Intuition

- **Identify word closest to target:**

- **Millennial**
  - ➔ octopus
  - ➔ fork
  - ➔ water
  - ➔ avocado

Semantic similarity: Intuition

- Identify word closest to target:

- Millennial
  - octopus
  - fork
  - water
  - avocado

Semantic Similarity

What drives semantic similarity?

- **Meaning**
  - → The two concepts are close in terms of meaning
  - → e.g. ‘inadvertent’ and ‘accidental’
- **World Knowledge**
  - → The two concepts have similar properties, often occur together, or occur in similar contexts
  - → e.g. ‘spinach’ and ‘kale,’ or ‘UPS’ and ‘FedEx’
- **Psychology**
  - → The two concepts fit together within an over-arching psychological schema or framework
  - → e.g. ‘money’ and ‘bank’, or ‘millennial’ and ‘avocado’

Semantic Similarity

What drives semantic similarity?

- **Meaning**
  - → The two
  - → e.g. 'inac
- **World Kı**
  - → The two ccur in similar c
  - → e.g. 'spir
- **Psychol**
  - → The two schema or framework
  - → e.g. 'money' and 'bank', or 'millennial' and 'avocado'

Automatic computation of semantic similarity

Why would such a thing be useful?

- **Semantic similarity gives us a way to generalize beyond word identities**
- **Lots of practical applications**
  - Information retrieval
  - Machine translation
  - Ontological hierarchies
  - Etc.

Beyond one-hot vectors

So far in this course, most of our statistical models have treated words as discrete categories.

- No explicit relationship between "cat" and "feline" in our LMs, classifiers, HMMs
- Equivalently, each word type in the vocabulary can be represented as an integer or as a one-hot vector
  - "cat" = [0 0 0 0 0 1 0 0 0 …]
  - "feline" = [0 0 0 0 0 0 0 1 0 …]
  - They are orthogonal; dot product is 0
  - Length is size of the vocabulary

Distributional Hypothesis

Idea: Similar linguistic objects have similar contents (for documents, paragraphs, sentences) or contexts (for words)

➔ “Differences of meaning correlates with differences of distribution” (Harris, 1970)

➔

➔ “You shall know a word by the company it keeps!” (Firth, 1957)

Example

- ➔ He handed her a glass of bardiwac
- ➔ Beef dishes are made to complement the bardiwac
- ➔ Nigel staggered to his feet, face flushed from too much bardiwac.
- ➔ Malbec, one of the lesser-known bardiwac grapes, responds well to Australia’s sunshine
- ➔ I dined off bread and cheese and this excellent bardiwac
- ➔ The drinks were delicious: bold bardiwac as well as light, sweet Rhenish.

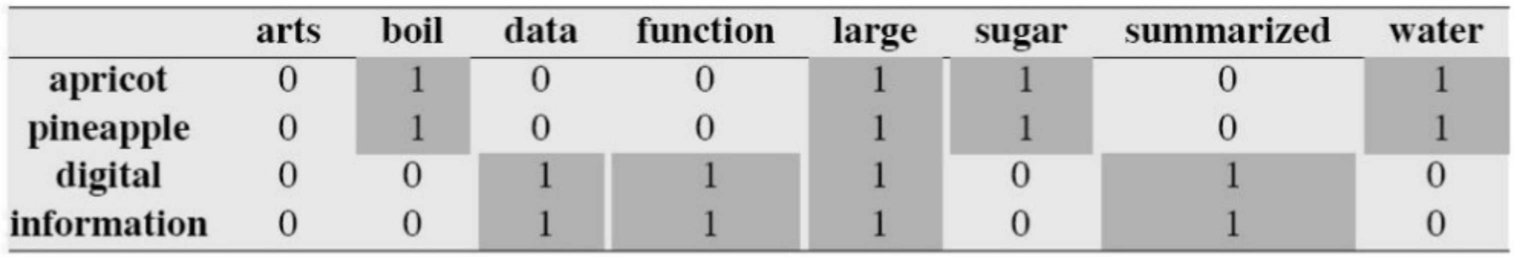
Word Vectors

- A word type may be represented as a vector of features indicating the contexts in which it occurs in a corpus.

$$\vec{w} = (f_1, f_2, f_3, \ldots f_N)$$

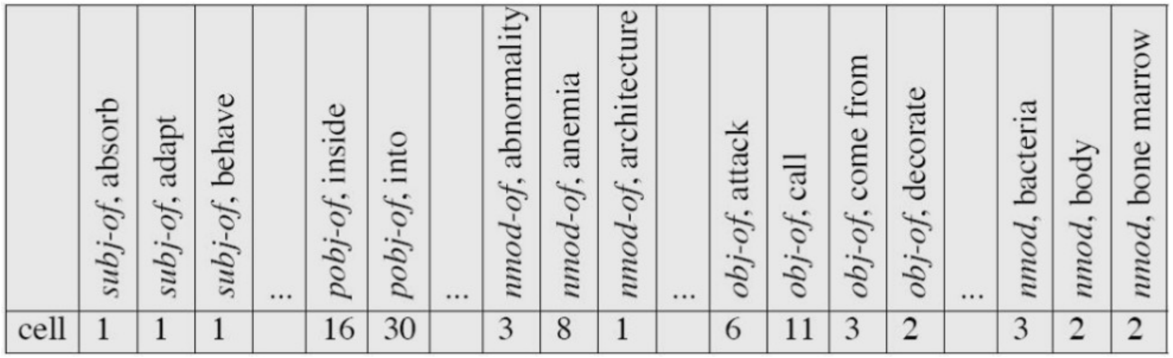
Context Features

Word Co-occurrence within a window:



	arts	boil	data	function	large	sugar	summarized	water
apricot	0	1	0	0	1	1	0	1
pineapple	0	1	0	0	1	1	0	1
digital	0	0	1	1	1	0	1	0
information	0	0	1	1	1	0	1	0

Grammatical Relations:



	subj-of, absorb	*subj-of*, adapt	*subj-of*, behave	…	*pobj-of*, inside	*pobj-of*, into	…	*nmod-of*, abnormality	*nmod-of*, anemia	*nmod-of*, architecture	…	*obj-of*, attack	*obj-of*, call	*obj-of*, come from	*obj-of*, decorate	…	*nmod*, bacteria	*nmod*, body	*nmod*, bone marrow
cell	1	1	1		16	30		3	8	1		6	11	3	2		3	2	2

Context Features

Feature Values:

➔Boolean

➔Raw Counts

➔ Weighting Scheme (e.g. tf-idf)

➔Association Values

Association Value: Pointwise Mutual Information

- Measures how often a target word *w* and a feature *f* occur together compared to what we would expect if the two were independent

$$\text{association}_{\text{PMI}}(w, f) = \log_2 \frac{P(w, f)}{P(w)P(f)}$$

➔ PMI ranges from *-inf* to *+inf*, but negative values are generally unreliable (Jurafsky & Martin, 2017:275).

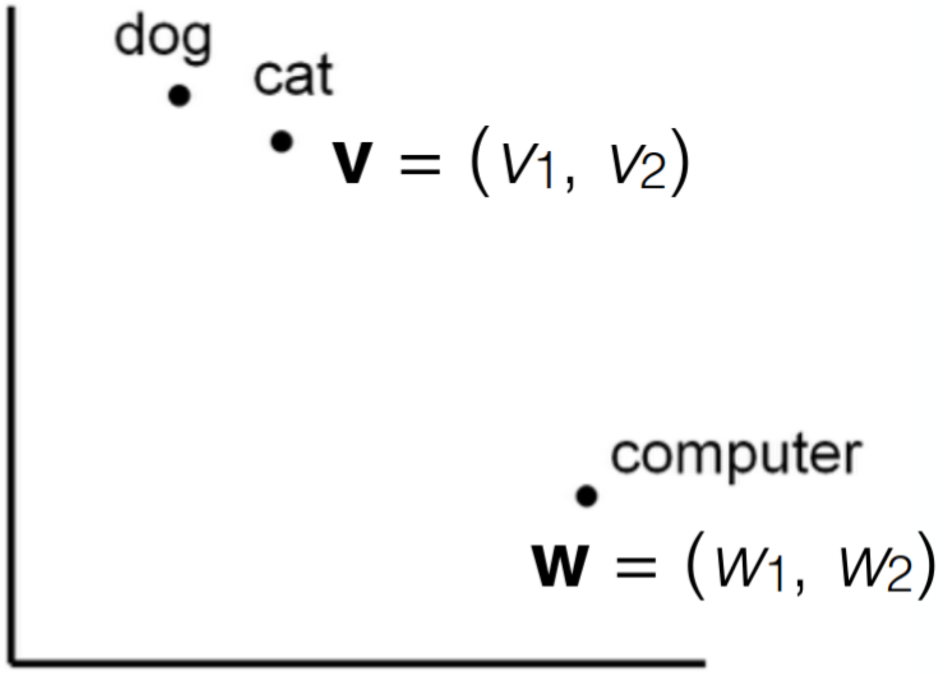
➔ Use positive PMI and clip at zero.

Computing Similarity

Semantic similarity boils down to computing some measure of spatial similarity between context vectors in vector space.

Words in a vector space

- **In 2 dimensions:**
  - V = ‘cat’
  - W = ‘computer’

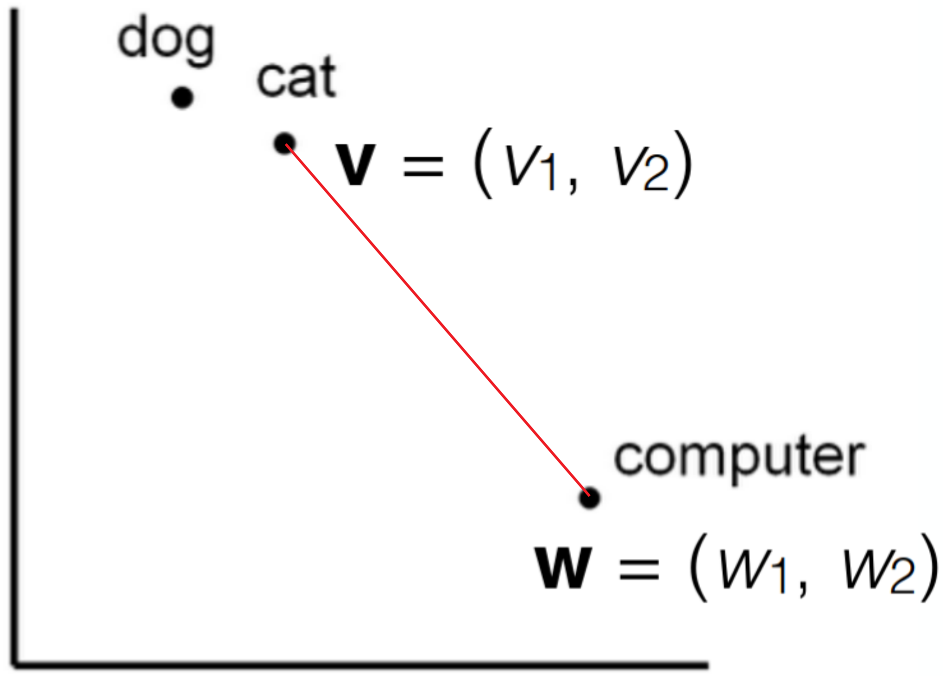


Euclidean Distance

- **Formula:**

$$\sqrt{\Sigma_i (v_i - w_i)^2}$$

→ Can be oversensitive to extreme values

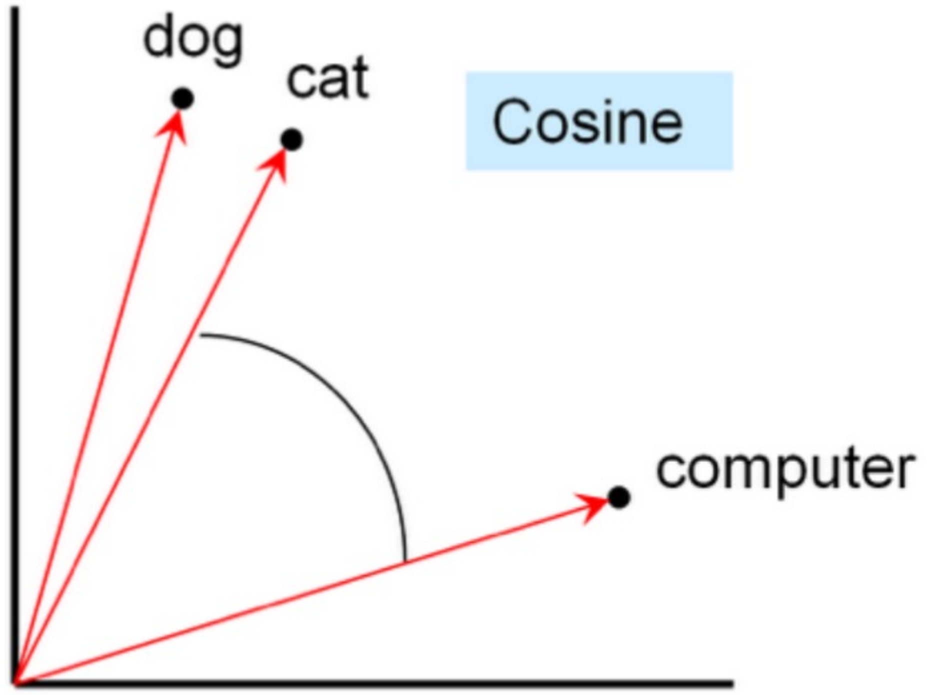


Cosine Similarity

- **Formula:**

$$\text{sim}_{\text{cosine}}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

➔ Typically better than Euclidean distance for vector space semantic



Vector Sparseness

- Co-occurrence based context vectors tend to very long and very sparse.
  - ➔ len(word_vec) == len(vocab)

- short (dim. of around 50-300) and dense context vectors are usually preferable.
  - ➔ Easier to include as features in machine learning systems
  - ➔ Fewer parameters = better generalization & less over-fitting
  - ➔ Better at capturing synonymy

Dense Vectors

2 Main methods of producing short, dense vectors:

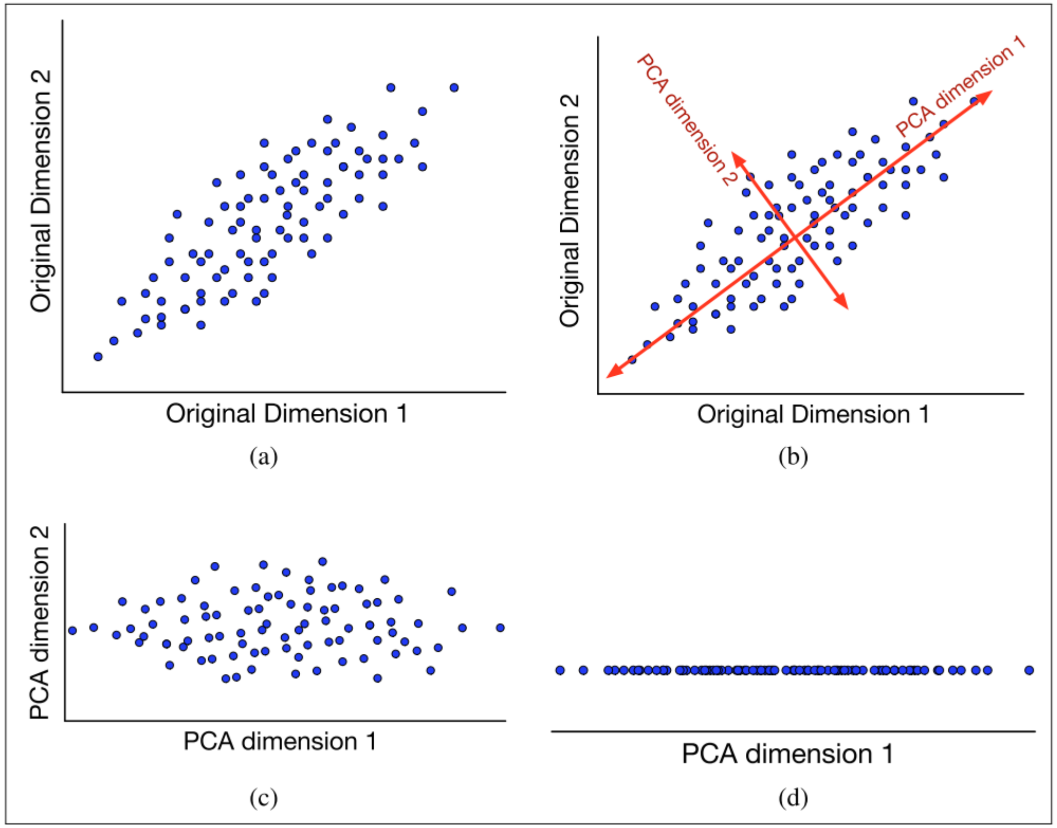
(1) Dimensionality reduction

(2) Neural Language Models

Dimensionality Reduction

Methods:

- ➔Principal Component Analysis (PCA)
- ➔t-Distributed Stochastic Neighbor Embedding (t-SNE)
- ➔Latent Semantic Analysis (LSA)
- ➔...



Neural Network Embeddings

Idea: Train a neural network to predict context words based on current ‘target’ word.

- Similar input words → similar context word prediction
- Similar input words → similar corresponding rows in the weight matrix of the trained network.

We don’t actually care about context word prediction!

- Rows in the trained weight matrix become our context vectors (aka word vectors, aka word embeddings)

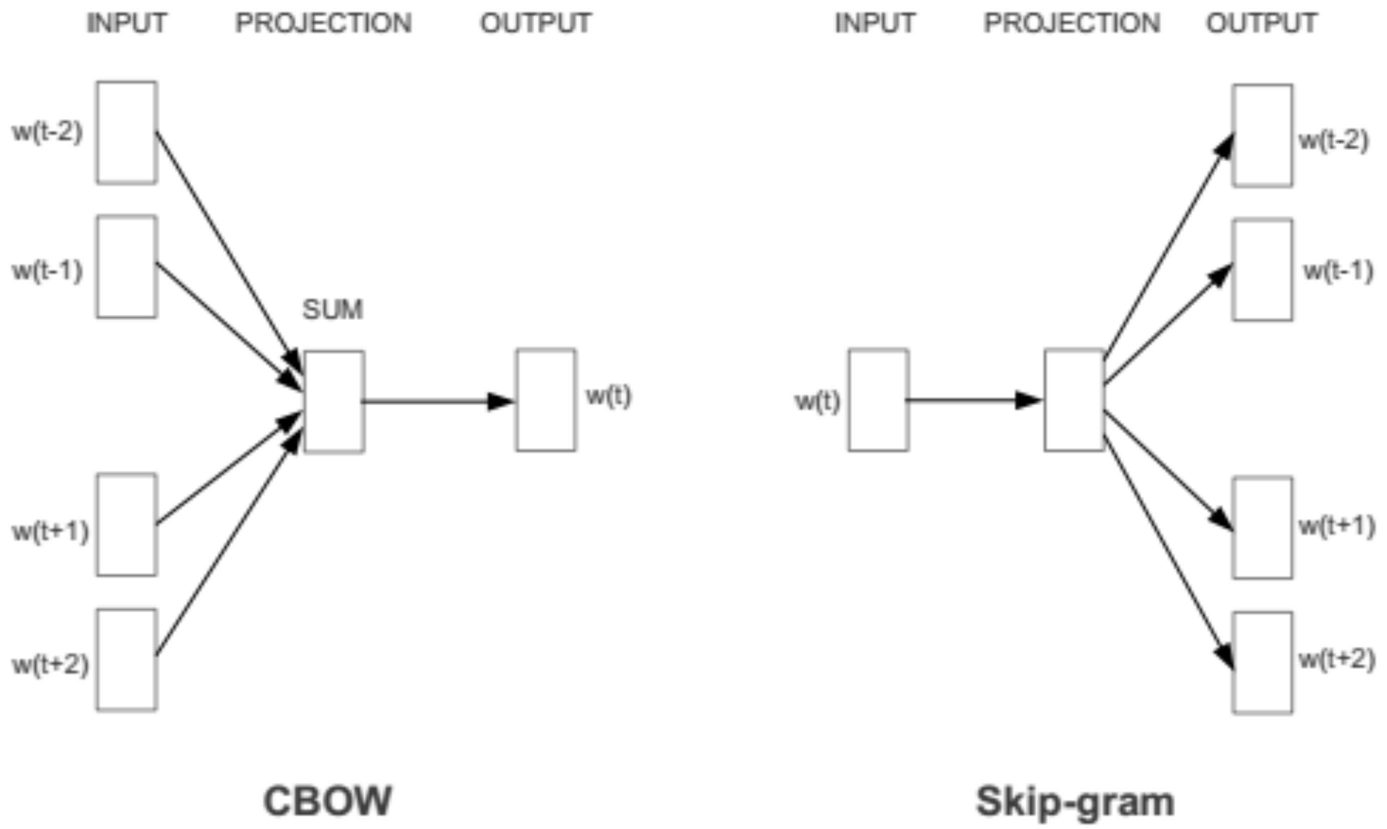
Neural Network Embeddings

- This idea marked a transition from count-based methods to prediction-based methods for obtaining word embeddings, and prediction-based methods are shown to be better (Baroni et al., 2014, *Don't count, predict!*)

- This means that we can use *any* language models' weights to represent a word, but let's see the first of its kind, **word2vec**

Neural Network Embeddings

Most popular family of methods: word2vec (Mikolov et al. 2013, Mikolov et al. 2013a)



Neural LM architectures: which to use?

- CBOW and Skip-Gram typically produce similar embeddings, but:
  - CBOW is several times faster to train, better accuracy for frequent words
  - Skip-Gram works well with small amounts of training data, and does well with representing rare words

- Mikolov: “Best practice is to try a few experiments and see what works the best for you”

- https://groups.google.com/forum/#!searchin/word2vec-toolkit/c-bow/word2vec-toolkit/NLvYXU99cAM/ESld8LcDxlAJ

Properties of dense word embeddings

Dense word embeddings encode:

- ➔ Semantic Relationships
- ➔ Syntactic Relationships

Can probe relations between words using vector arithmetic:

- ➔ king – male + female = ?
- ➔ walked – walk + fly = ?

Type-based vector to token-based vector

Now we’ve got one vector for each word type… what’s next?

- Did you hear the sound?
- He arrived home safe and sound.

… do you see the problem?

- As mentioned earlier, we can use *any* language model’s weights to represent a word

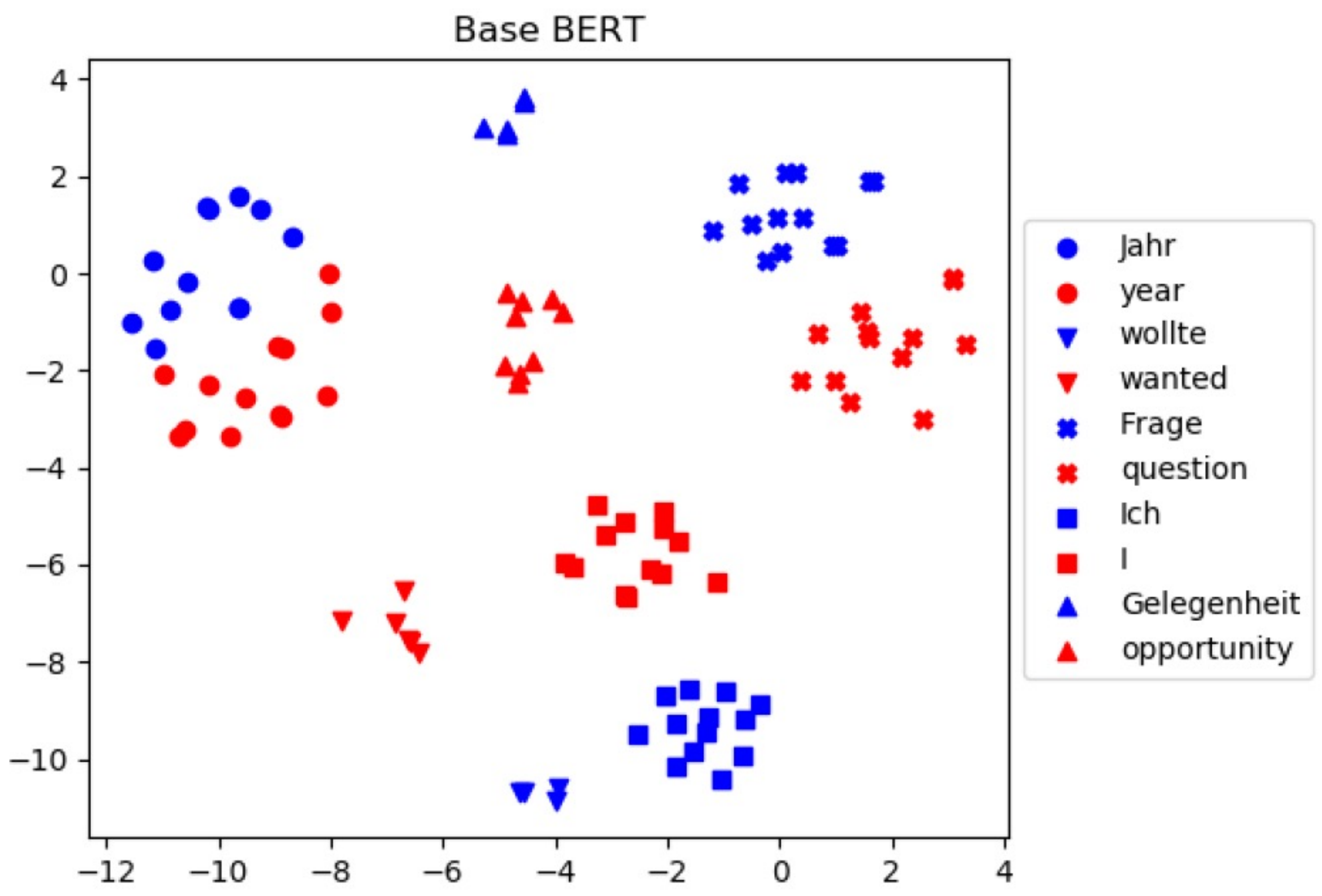
Type-based vector to token-based vector

Language models have evolved to capture this difference

- ELMo (Peters et al., 2017)
- BERT (Devlin et al., 2018)
- . . . (more on neural architecture later in this course!)

- These models generate *contextualized* (or dynamic, as opposed to static) word embeddings (CWEs).
- Some models are even *multilingual*!

Type-based vector to token-based vector



Cao et al. (2020)

- t-SNE-ed snapshot of some words from Europarl corpus in multilingual BERT's embedding space
- Each word *type* occurs multiple times because this is contextualized (*token*-based word embeddings)

Train your own word embeddings:

TensorFlow: https://www.tensorflow.org/tutorials/word2vec

Gensim: https://rare-technologies.com/word2vec-tutorial/

FastText: https://github.com/facebookresearch/fastText

Pretrained Word embeddings:

Word2Vec: https://code.google.com/archive/p/word2vec/

➔ Trained on 100 billion tokens from Google News corpus

GloVe: https://nlp.stanford.edu/projects/glove/

➔6B wikipedia, 42-840B tokens Common Crawl, 27B tokens Twitter

LexVec: https://github.com/alexandres/lexvec

➔ 58B tokens Common Crawl, 7B tokens Wikipedia + NewsCrawl

Nowadays… https://huggingface.co/models

➔500k+ models including ELMo, BERT, GPTs, and you can also train them from scratch with enough GPU!

Word embeddings: Evaluation

How to judge the quality of embeddings?

- **‘Relatedness’ scores for given word pairs**
  - ➔ Compare model’s relatedness scores to human relatedness scores
- **Analogy tests**
  - ➔ Find x such that x : y best resembles a sample relationship a : b
- **Categorization**
  - ➔ Recover a known clustering of words into different categories.

Document features

- So far: Features in word-vectors can be: context counts, PMI scores, weights from neural LMs…
- Can also be features of the docs in which the words occur.
- Document occurrence features are useful for topical/thematic similarity

Document-Term Matrix

	D1	D2	D3	D4
W1	23	17	0	0
W2	102	0	14	24
W3	14	57	0	2
W4	0	0	18	38

Term Frequency – Inverse Document Frequency (tf-idf)

- Common in IR tasks
- Popular method to weight term-document matrices in general

Tf: relative frequency of term in document

→ tf(t,d) = f(t,d)

Idf: inverse of the proportion of docs containing the term

→ N / n_t (N = total # of docs, n_t = # of docs term t appeared in)

Document-Term Matrix

	D1	D2	D3	D4
W1	23	17	0	0
W2	102	0	14	24
W3	14	57	0	2
W4	0	0	18	38

Tf-idf weighted Document-Term Matrix

	D1	D2	D3	D4
W1	.12	.16	0	0
W2	.21	0	.13	.11
W3	.03	.22	0	.01
W4	0	0	.39	.41

Tf-idf weighted Document-Term Matrix

Word Vectors

	D1	D2	D3	D4
W1	.12	.16	0	0
W2	.21	0	.13	.11
W3	.03	.22	0	.01
W4	0	0	.39	.41

Tf-idf weighted Document-Term Matrix

Document Vectors

	D1	D2	D3	D4
W1	.12	.16	0	0
W2	.21	0	.13	.11
W3	.03	.22	0	.01
W4	0	0	.39	.41

Topic Models

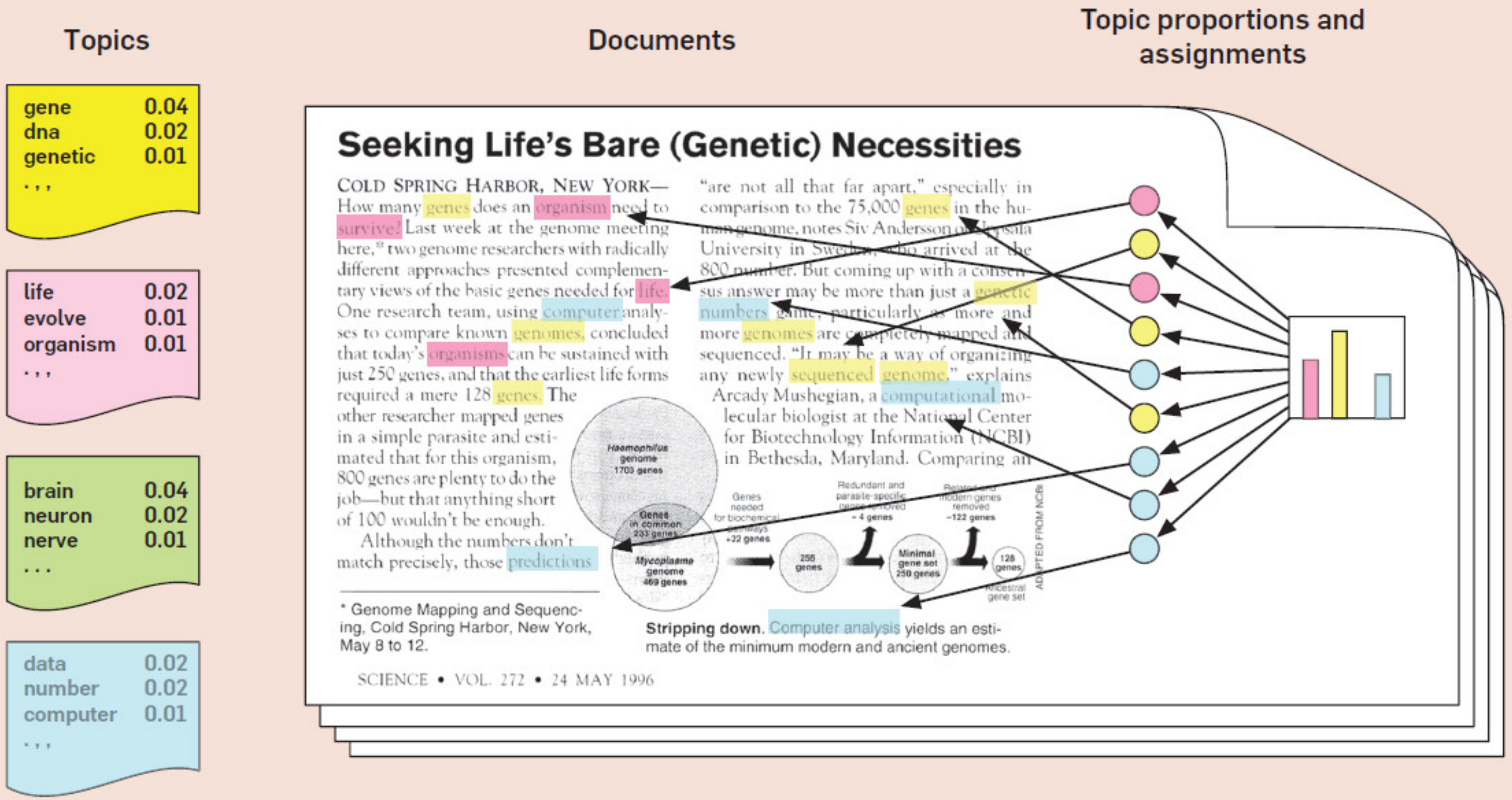
Latent Dirichlet Allocation (LDA) and variants known as topic models.

➔Learned on large document collection (unsupervised)

➔Latent probabilistic **clustering** of words that tend to occur in the same document. Each ‘**topic**’ cluster is a distribution over words.

➔Generative Model: Each document is a sparse mixture of topics. Each word in the doc is chosen by sampling a topic from the doc-specific topic distribution, then sampling a word from that topic.

Topic Models



gene	0.04
dna	0.02
genetic	0.01
. , ,	

life	0.02
evolve	0.01
organism	0.01
. , ,	

brain	0.04
neuron	0.02
nerve	0.01
. . .	

data	0.02
number	0.02
computer	0.01
. , ,	

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

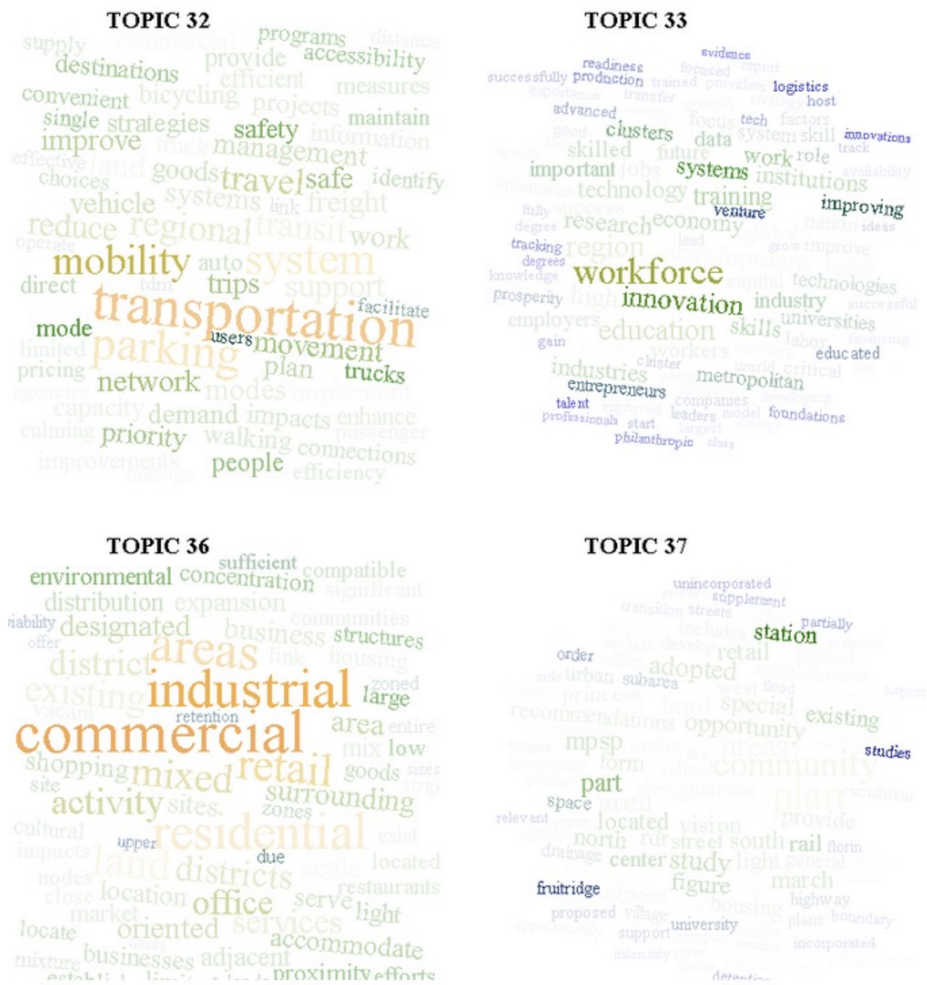
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

https://cacm.acm.org/magazines/2012/4/147361-probabilistic-topic-models/fulltext

Visualizing Topics



https://dhs.stanford.edu/algorithmic-literacy/using-word-clouds-for-topic-modeling-results/

Neural Language Models (again!)

- At word level, we saw count-based -> prediction-based
- At document level, we saw count-based (tf-idf)… so what about prediction-based?
- Document-level prediction tasks are not as straightforward as word/sentence level prediction tasks (as opposed to binary prediction in word2vec or NWP/NSP in modern LMs).

Neural Language Models (again!)

- Use word representations
    - Concatenate CWEs of tokens in a document
    - Max/average-pooling
    - …
- Document/span specific models
    - doc2vec (Le& Mikolov, 2014)
        - Document embeddings trained to predict next word in the document
    - SpanBERT (Joshi et al., 2019)
    - Longformer (Beltagy et al., 2020)
    - BigBird (Zaheer et al., 2020)
    - HiPool (Li et al., 2023)

Questions?