Word Embeddings

Paul Schmitt
Research topics in natural language processing
TU Wien, Austria

Table of Contents

- Vector Semantics
- Types of Embeddings
- Cosine Similarity
- PPMI (Sparse)
- Word2Vec (Dense)
- Evaluation methods
- Beyond Word2Vec
- Temporal Word Embeddings

Vector Semantics

you shall know a word by the company it keeps

Firth (1957)

Vector Semantics

- Foundational idea in linguistics
- Studies how the meanings of words can be represented using vectors in mathematical spaces
- Reflected in modern computational linguistics techniques like Word2Vec

Key Ideas

- Represent a word as a point in a multidimensional space that is derived from the distributions of word neighbours
- Vectors for representing words are called (word-)embeddings
- Vector semantic models can be learned automatically from text without supervision
  - Generally based on a co-occurrence matrix

Goals

- Create abstract representations of words
- Analyze relationships between words in a mathematical or visual way
- Learning representations of data to make it easier to extract useful information when building classifiers or other predictors

  ⍰ offers enormous power to NLP applications

Sparse Embeddings

- PPMI, Count Vectors (Bag-of-Words)
- High dimensionality
- Memory inefficient
- Common in traditional NLP

Dense Embeddings

- Word2Vec, GloVe, BERT
- Much smaller dimension than the vocabulary size (e.g., 300 dimensions for Word2Vec).
- Computationally efficient
- Typically generated using neural networks on large corpora.
- Dominant in modern NLP

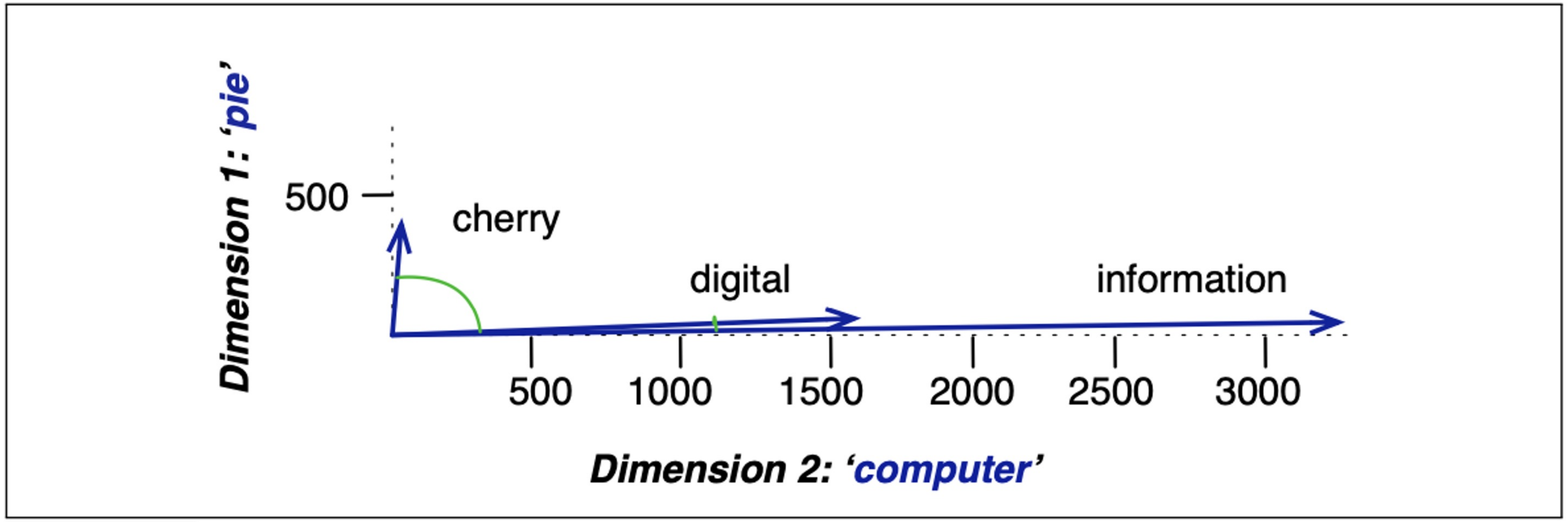
Cosine Similarity

- Measurement of similarity between vectors (words)
- Based on the normalized dot product
- Range: 0 (no similarity) – 1 (perfect similarity)

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

Cosine Similarity

	pie	data	computer
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325

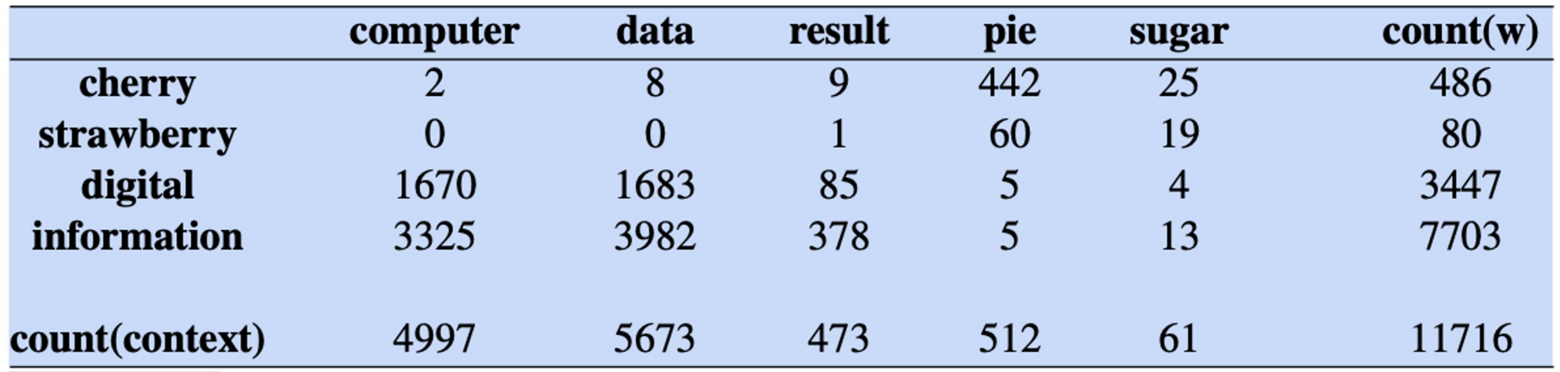


(P)PMI

$$\text{PPMI}(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$

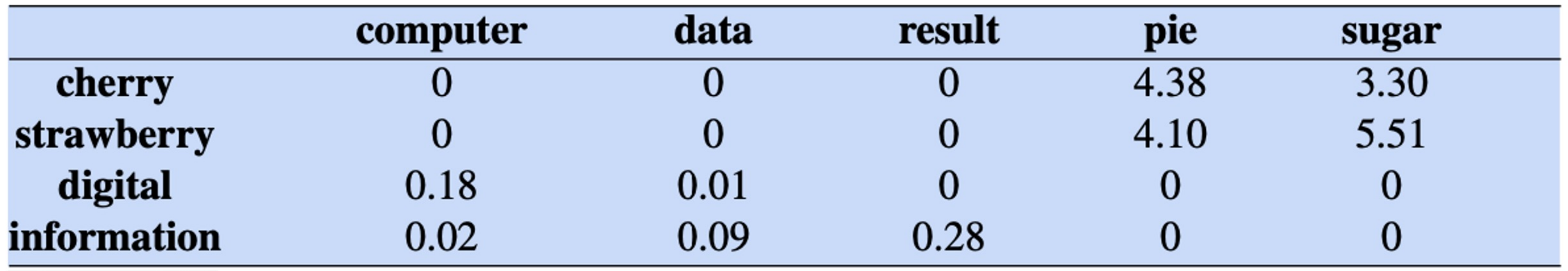
- Positive Pointwise Mutual Information (PPMI)
- Can be computed from a co-occurrence matrix
- Quantifies the degree to which two events co-occur compared to what would be expected if they were independent.

(P)PMI



	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

Figure 6.10 Co-occurrence counts for four words in 5 contexts in the Wikipedia corpus, together with the marginals, pretending for the purpose of this calculation that no other words/contexts matter.



	computer	data	result	pie	sugar
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

Figure 6.12 The PPMI matrix showing the association between words and context words, computed from the counts in Fig. 6.11. Note that most of the 0 PPMI values are ones that had a negative PMI; for example PMI(*cherry,computer*) = -6.7, meaning that *cherry* and *computer* co-occur on Wikipedia less often than we would expect by chance, and with PPMI we replace negative values by zero.

Word Embeddings

It turns out that dense vectors work better in every NLP task than sparse vectors.

Word2Vec

- Developed by Mikolov et al. at Google
- 2 algorithms in a software package called word2vec
- Convert words to dense vector representations that capture semantic and syntactic relationships.
- “Is word c likely to show up near apricot?”

  ⍰ Self Supervision

Word2Vec – Training Process

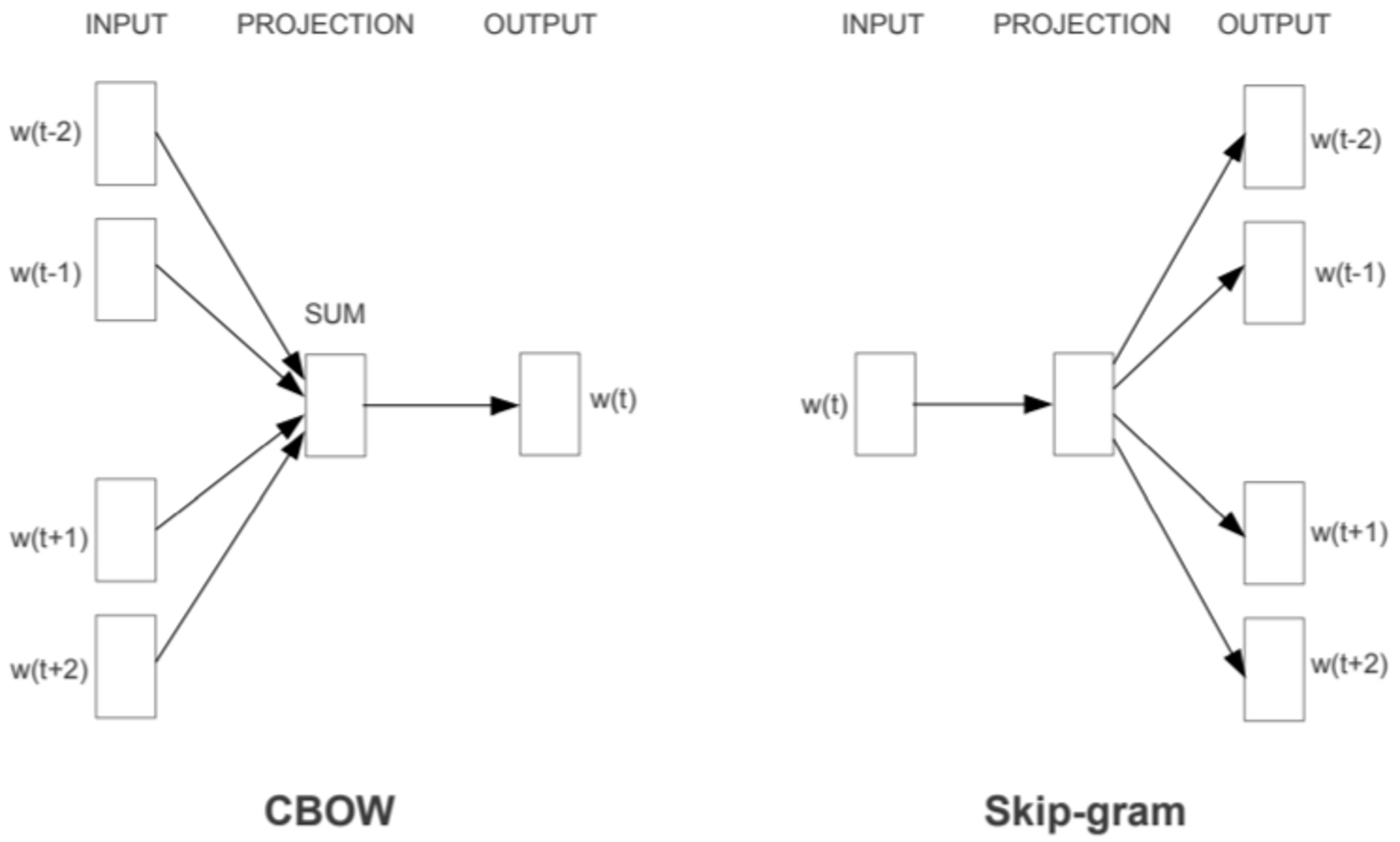
- Start with random vectors for each word in the vocabulary
- Use a sliding window to capture a word and its surrounding context
- Predict the center word from context (CBOW) or context from the center word (Skip-Gram)
- Iteratively adjust weights to minimize the difference between predicted and actual word contexts
- Harvest weights from the NN and use them as the embeddings

Word2Vec - Architectures

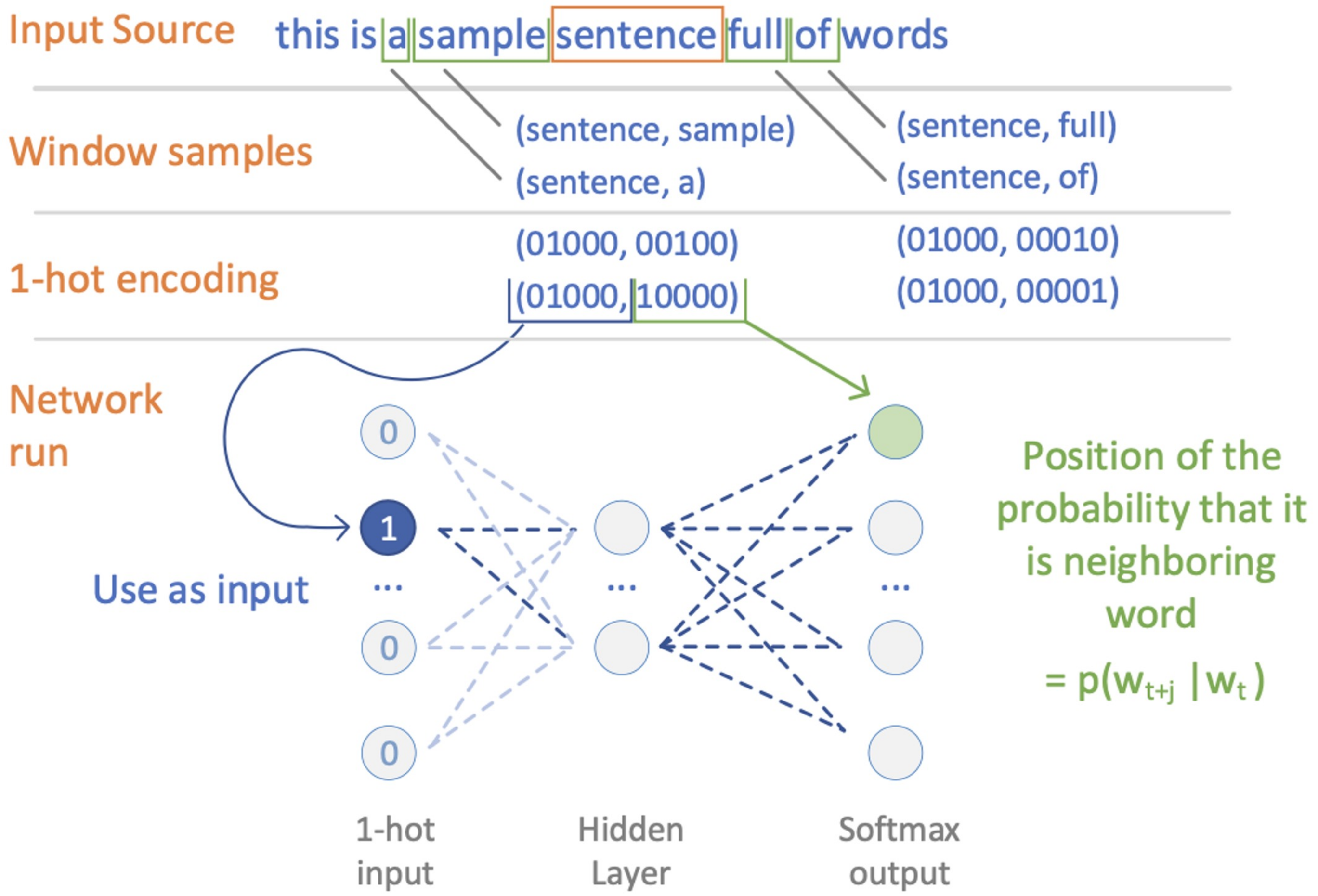
- Skip-Gram
  - Predict surrounding context words from a target word
  - Input: Target word - Outputs: Context words
  - performs better for larger datasets and rare words

- Continuous Bag of Words (CBOW)
  - Predict a target word from its surrounding context
  - Inputs: Context words - Output: Target word
  - faster and more data-efficient for smaller datasets

Word2Vec


INPUT
PROJECTION
OUTPUT
w(t-2)
w(t-1)
SUM
w(t+1)
w(t+2)
w(t)
CBOW
INPUT
PROJECTION
OUTPUT
w(t)
w(t-2)
w(t-1)
w(t+1)
w(t+2)
Skip-gram

Word2Vec – Training Process



Evaluation of Embeddings

- Internal
  - E.g. Analogy-Tasks (King – Man + Woman = Queen)
  - Similarity: correlation between similarity scores of algorithm and human

- External
  - Measure performance on ML Tasks that use the embeddings as inputs

Beyond Word2Vec

- Unsupervised prediction of context in a sequence is a very generalizable idea

  ⍰ Doc2Vec, Sentence-BERT (SBERT)

- Temporal word embeddings (historical semantics)

- Endless minor adjustments and improvements

More at https://github.com/MaxwellRebo/awesome-2vec

Beyond Word2Vec

- FastText
- GloVe (Global Vectors for Word Representation)
- ELMo (Embeddings from Language Models)
- BERT (Bidirectional Encoder Representations from Transformers)

Temporal Word Embeddings

- Dynamic Nature of Language
  - Language evolves over time, changing meanings and usages
  - Traditional embeddings don't capture these shifts
- Diachronic Word Embeddings
  - Train embeddings on different time slices of data
  - Allows tracking semantic changes over periods
- Alignment is not trivial !

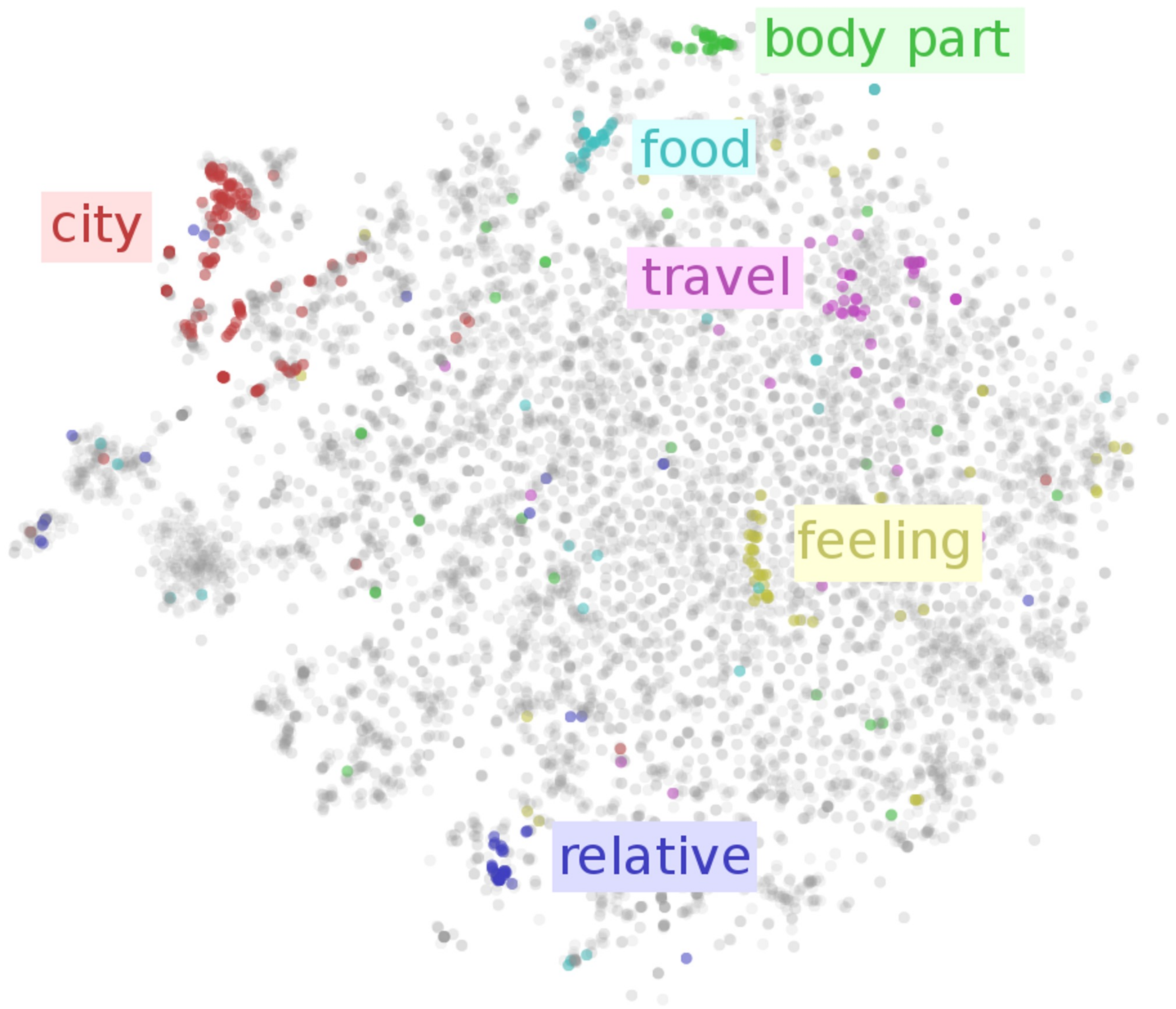
Temporal Word Embeddings

- Applications
  - Detecting semantic drifts, e.g., the changing meaning of the word "gay" over the 20th century.
  - Analyzing cultural and societal shifts based on language changes
- Challenges
  - Need vast amounts of time-stamped data
  - Aligning and comparing embeddings across time can be complex

Summary

- Words can be defined by their usage in a language
- Represent words as vectors in multidimensional space
- Traditional approach (sparse embeddings) is followed by modern approaches (dense embeddings)
- Temporal word embeddings can be used to study the evolution of language

Thank You for your Attention!


body part
food
city
travel
feeling
relative