Embeddings Learned By Gradient Descent

Benjamin Roth; Folien von Hinrich Schütze

Center for Information and Language Processing, LMU Munich

GoogleNews embeddings

- Embedding basis for `cis.lmu.de/schuetze/e`
- These were computed with word2vec skipgram.
- 3,000,000 words and phrases
- 300-dimensional embeddings
- Trained on 100 billion word Google news dataset

word2vec parameters

- -train <file>
  input corpus
- -output <file>
  word vectors are saved in output file
- -size <int>
  dimensionality of word vectors
- -window <int>
  skip length between words
- -sample <float>
  threshold for downsampling frequent words
- -hs <int>
  use hierarchical softmax (0/1)
- -negative <int>
  number of negative samples
- -threads <int>
  number of threads

word2vec parameters

- -iter <int>
  number of training iterations
- -min-count <int>
  discard words that occur less often than this
- -alpha <float>
  starting learning rate
- -classes <int>
  output word classes rather than word vectors
- -debug <int>
  set the debug mode
- -binary <int>
  use binary or plain text output format (0/1)

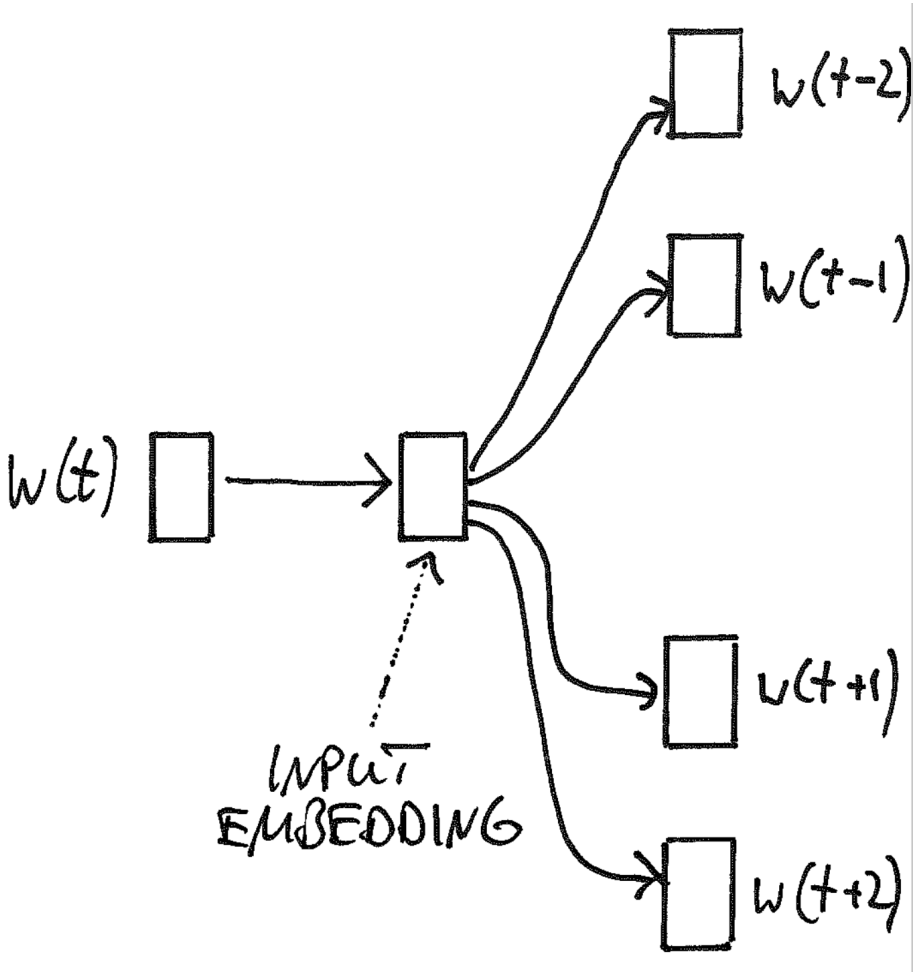
word2vec parameters

- -save-vocab <file>
  save the vocabulary
- -read-vocab <file>
  read the vocabulary from file (don't take it from corpus)
- -cbow <int>
  use cbow or not (0/1)

Overview

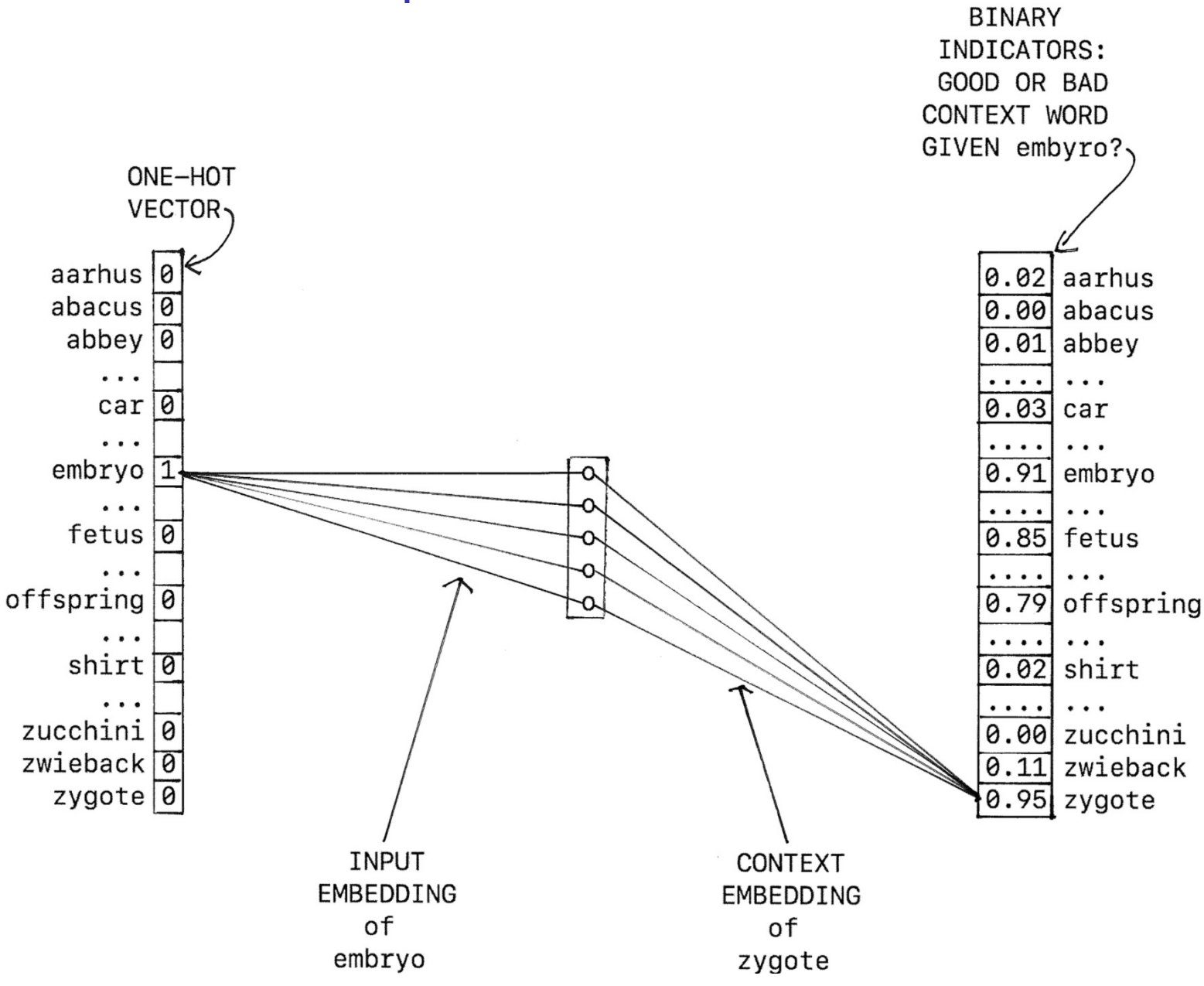
word2vec skipgram

predict, based on input word, a context word



word2vec skipgram

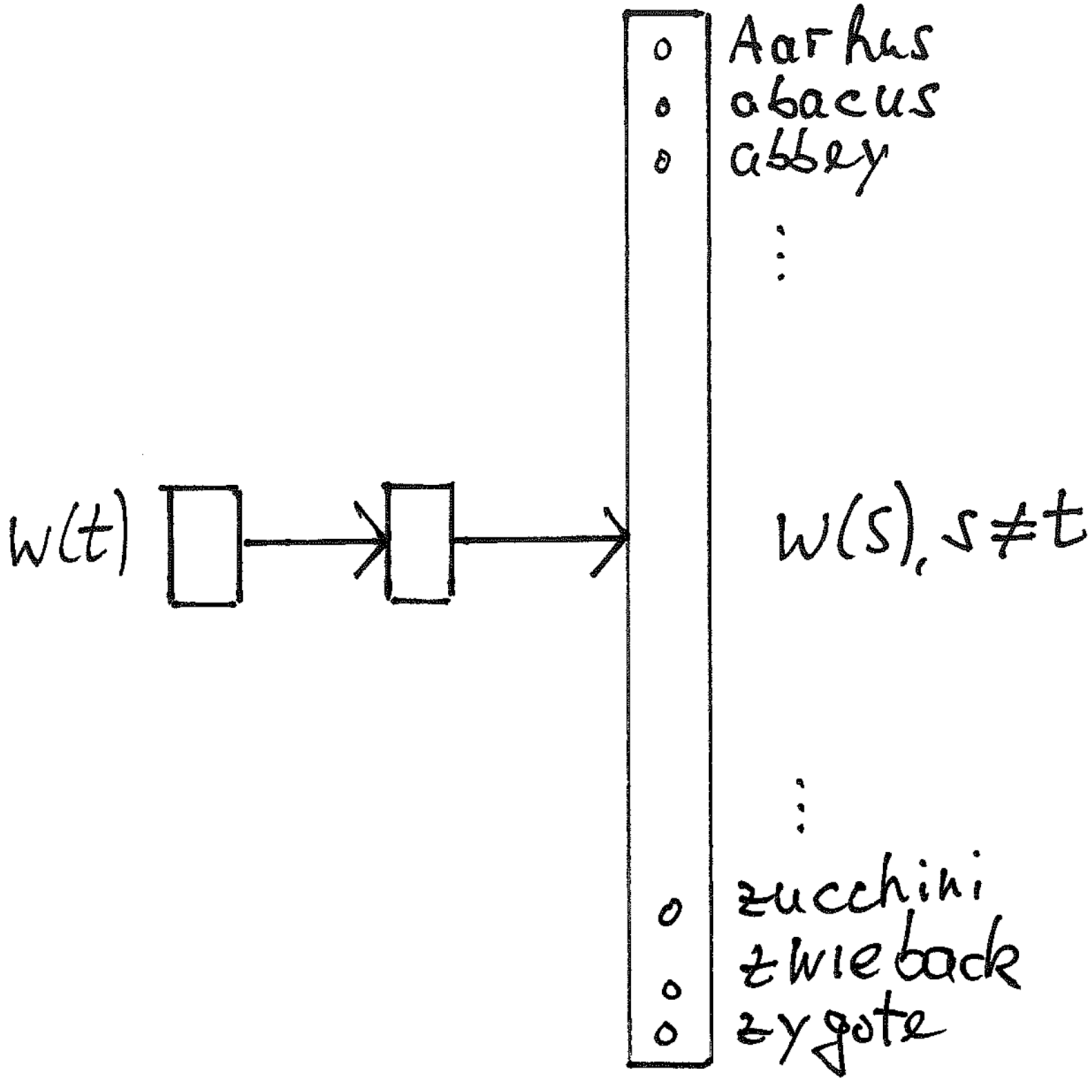
predict, based on input word, a context word



Three versions of word2vec skipgram

- All three share skipgram objective (previous slide): predict, based on input word, a context word
- 1. Matrix factorization (SVD) of PPMI matrix
  - Tuesday's lecture
- 2. skipgram negative sampling (SGNS) using GD
  - Today's topic
  - Levy&Goldberg show rough equivalence: SGNS $\approx$ SVD-of-PPMI-matrix
  - No rigorous proof?
- 3. hierarchical softmax (skipgram HS)
  - skipgram HS vs. SGNS: different objectives

skipgram softmax



skipgram softmax: objective

$$\text{argmax}_\theta \sum_{(w,c) \in D} \log \frac{\exp(\vec{v}_w \cdot \vec{v}_c)}{\sum_{c' \in V} \exp(\vec{v}_w \cdot \vec{v}_{c'})}$$

(hierarchical softmax is hierarchical version of this)

Three versions of skipgram: Learning algorithms

w2v skipgram SGNS (original)	gradient descent
w2v skipgram SGNS (Levy&Goldberg)	SVD
w2v skipgram hierarchical softmax	gradient descent

skipgram negative sampling (SGNS): objective

skipgram negative sampling (SGNS): objective (not!)

$$\text{argmax}_\theta [\sum_{(w,c) \in D} (\vec{v}_w \cdot \vec{v}_c) + \beta \sum_{(w,c) \in V \times V} (-\vec{v}_w \cdot \vec{v}_c)]$$

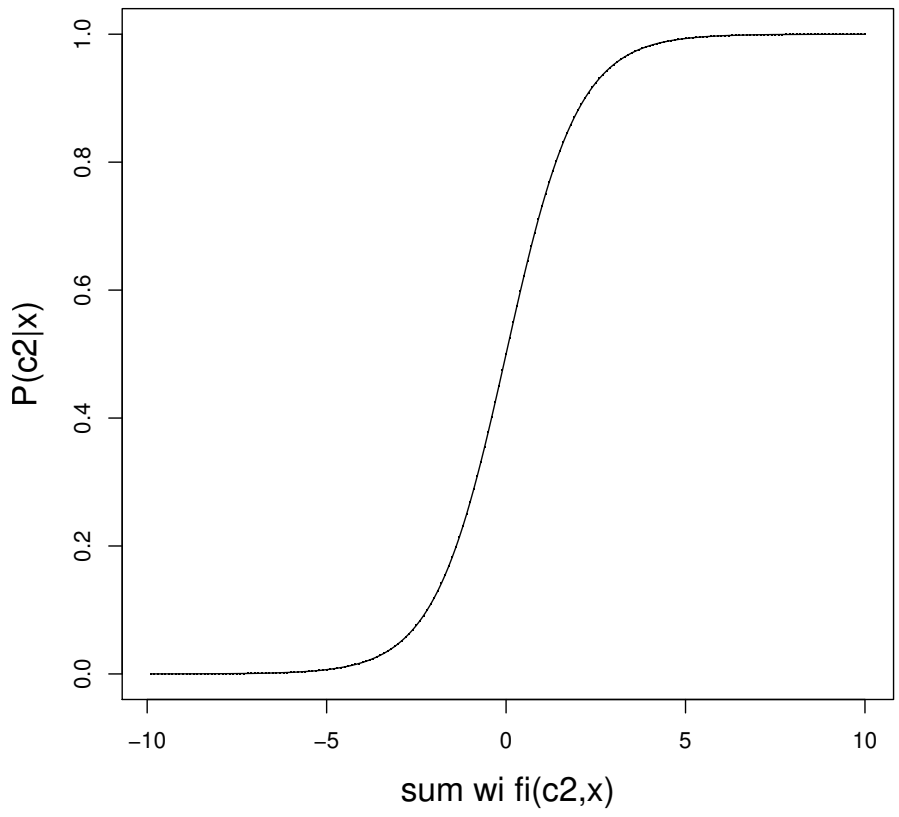
- Training set D: set of word-context pairs (w, c)
- We learn an embedding $\vec{v}_w$ for each w.
- We learn an embedding $\vec{v}_c$ for each c.
- Note that each word has two embeddings:
  an input embedding and a context embedding
- We generally only use the input embedding.
- make dot product of "true" pairs as big as possible
- dot product of "false" pairs as small as possible

skipgram negative sampling (SGNS): objective

$$\text{argmax}_\theta [\sum_{(w,c) \in D} \log \sigma(\vec{v}_w \cdot \vec{v}_c) + \beta \sum_{(w,c) \in V \times V} \log \sigma(-\vec{v}_w \cdot \vec{v}_c)]$$

- $\sigma(x) = 1/(1 + e^{-x})$
- Training set D: set of word-context pairs (w, c)
- We learn an embedding $\vec{v}_w$ for each w.
- We learn an embedding $\vec{v}_c$ for each c.
- Note that each word has two embeddings:
  an input embedding and a context embedding
- We generally only use the input embedding.
- make dot product of "true" pairs as big as possible
- dot product of "false" pairs as small as possible

σ: Logistic = Sigmoid



Housing prices in Portland

input variable **x** size (feet^2)	output variable **y** price ($) in 1000s
2104	460
1416	232
1534	315
852	178

We will use m for the number of training examples.

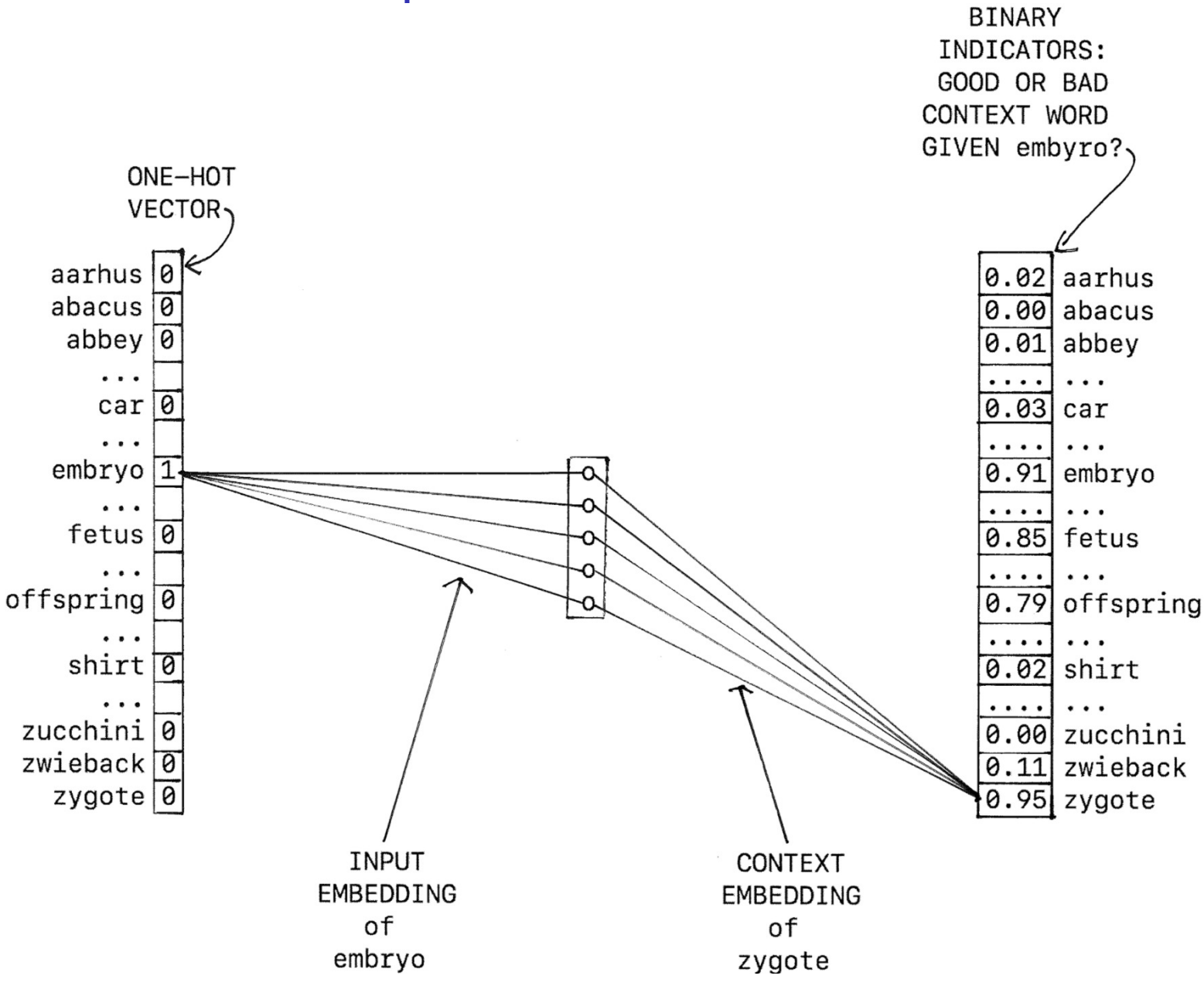
Housing prices in Portland

input variable **x** size (feet^2)	output variable **y** price ($) in 1000s
2104	460
1416	232
1534	315
852	178

We will use m for the number of training examples.

word2vec skipgram

predict, based on input word, a context word



Setup to learn housing price predictor using GD
Next: Setup for word2vec skipgram

- Hypothesis:

$$h_\theta = \theta_0 + \theta_1 x$$

- Parameters:

$$\theta = (\theta_0, \theta_1)$$

- Cost function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

- Objective: $\text{minimize}_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Parameters

house prices: $\theta = (\theta_0, \theta_1)$

dimensionality of embeddings: d, size of vocabulary: n, word embeddings θ, context embeddings η
word2vec skipgram:
$\theta_{11}, \theta_{12}, \ldots, \theta_{1d}$
$\theta_{21}, \theta_{22}, \ldots, \theta_{2d}$
$\ldots$
$\theta_{n1}, \theta_{n2}, \ldots, \theta_{nd}$
$\eta_{11}, \eta_{12}, \ldots, \eta_{1d}$
$\eta_{21}, \eta_{22}, \ldots, \eta_{2d}$
$\ldots$
$\eta_{n1}, \eta_{n2}, \ldots, \eta_{nd}$

Hypothesis

house prices: $h_\theta = \theta_0 + \theta_1 x$

word2vec skipgram:

$$h_{\theta,\eta}(w, c) = sim(\theta(w), \eta(c))$$

Read: *"The similarity of word vector θ and context vector η for word-context-pair w, c"*

This similarity should be ~ 1 if the word-context-pair has been observed in the corpus, and it should be ~ 0 if it is a random combination of two words in the vocabulary.

Cost function

house prices:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

word2vec skipgram: It's a reward function!

$$[\sum_{(w,c) \in D} \log \sigma(\vec{v}_w \cdot \vec{v}_c) + \beta \sum_{(w,c) \in V \times V} \log \sigma(-\vec{v}_w \cdot \vec{v}_c)]$$

$$J(\theta, \eta) = [\sum_{(w,c) \in D} \log \sigma(\theta(w) \cdot \eta(c)) + \beta \sum_{(w,c) \in V \times V} \log \sigma(-\theta(w) \cdot \eta(c))]$$

Objective

house prices: gradient descent

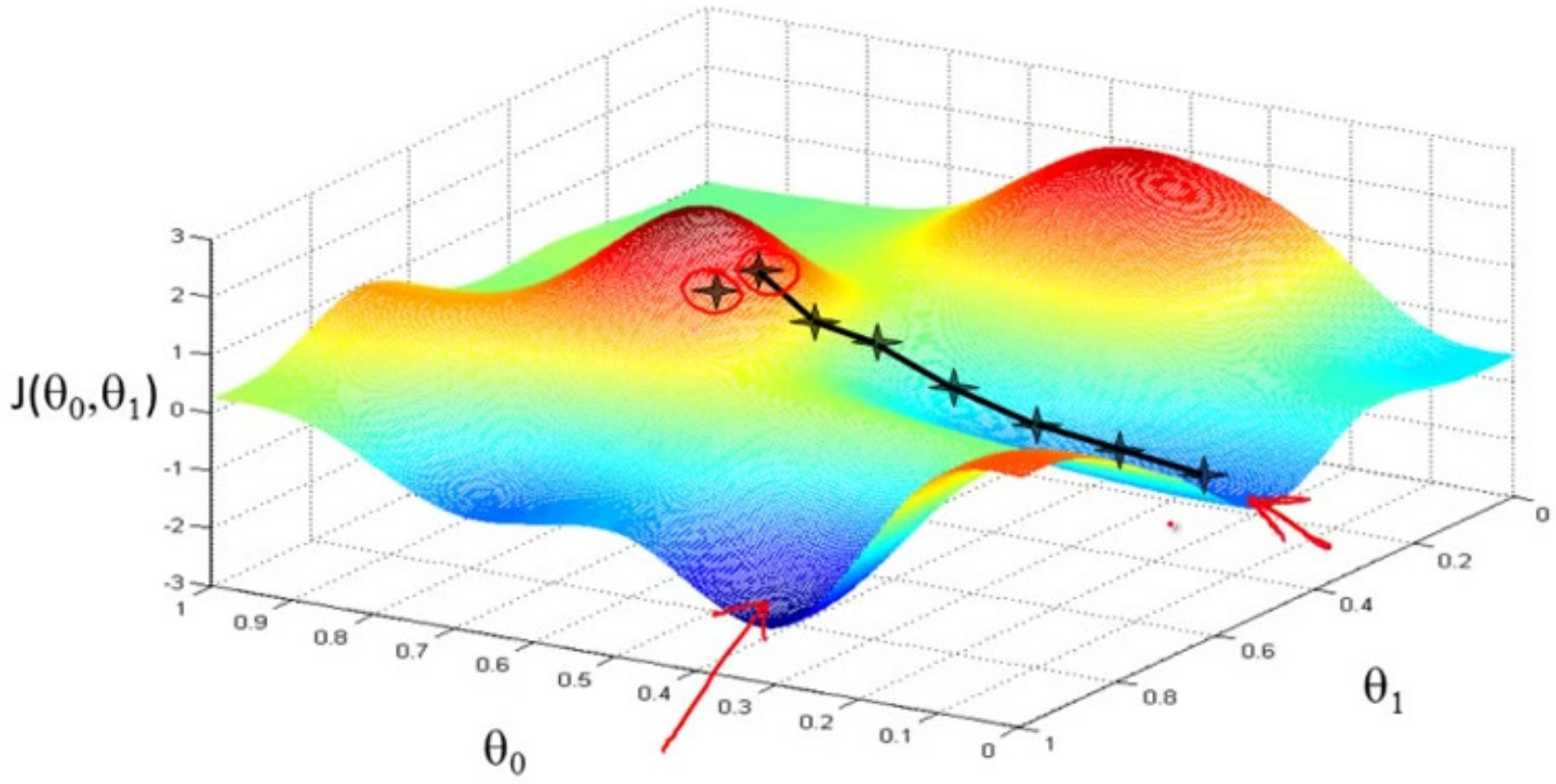
$$\text{minimize}_{\theta_0,\theta_1} J(\theta_0, \theta_1)$$

word2vec skipgram: gradient ascent

$$\text{maximize}_{\theta,\eta} J(\theta, \eta)$$

equivalent formulation for gradient descent:

$$\text{minimize}_{\theta,\eta} -J(\theta, \eta)$$


J(θ0,θ1)
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1

Exercise

- What is the maximum value that the objective can take in word2vec skipgram? (focus on first term, below)
- Are we likely to find parameters for which we reach the maximum? (focus on first term, below)
- (Recall: $\sigma(x) = 1/(1 + e^{-x})$)
- Why?

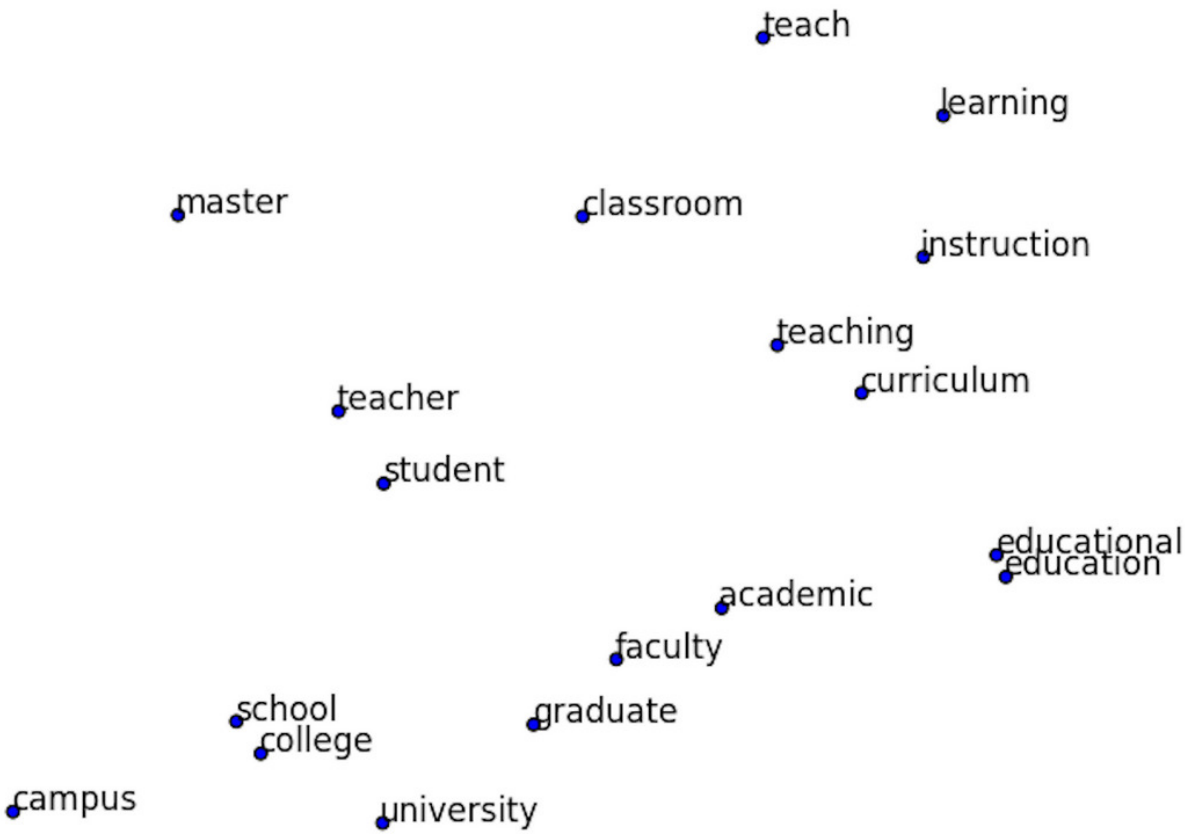
$$J(\theta, \eta) = [\sum_{(w,c) \in D} \log \sigma(\theta(w) \cdot \eta(c)) + \beta \sum_{(w,c) \in V \times V} \log \sigma(-\theta(w) \cdot \eta(c))]$$

house prices	word2vec skipgram
θ_0, θ_1	$2\|V\|d$ parameters: θ, η
$h_\theta(x) = \theta_0 + \theta_1 x$	$h_{\theta,\eta}(w, c) = sim(\theta(w), \eta(c))$
$J(\theta) =$	$J(\theta, \eta) =$
$1/(2m) \sum (h_\theta(x^{(i)}) - y^{(i)})^2$	$\sum_{(w,c)\in D} \log \sigma(\theta(w) \cdot \eta(c))$
	$+\beta \sum_{(w,c)\in V\times V} \log \sigma(-\theta(w) \cdot \eta(c))$
$\text{argmin}_\theta J(\theta)$	$\text{argmax}_{\theta,\eta} J(\theta, \eta)$

Visualization

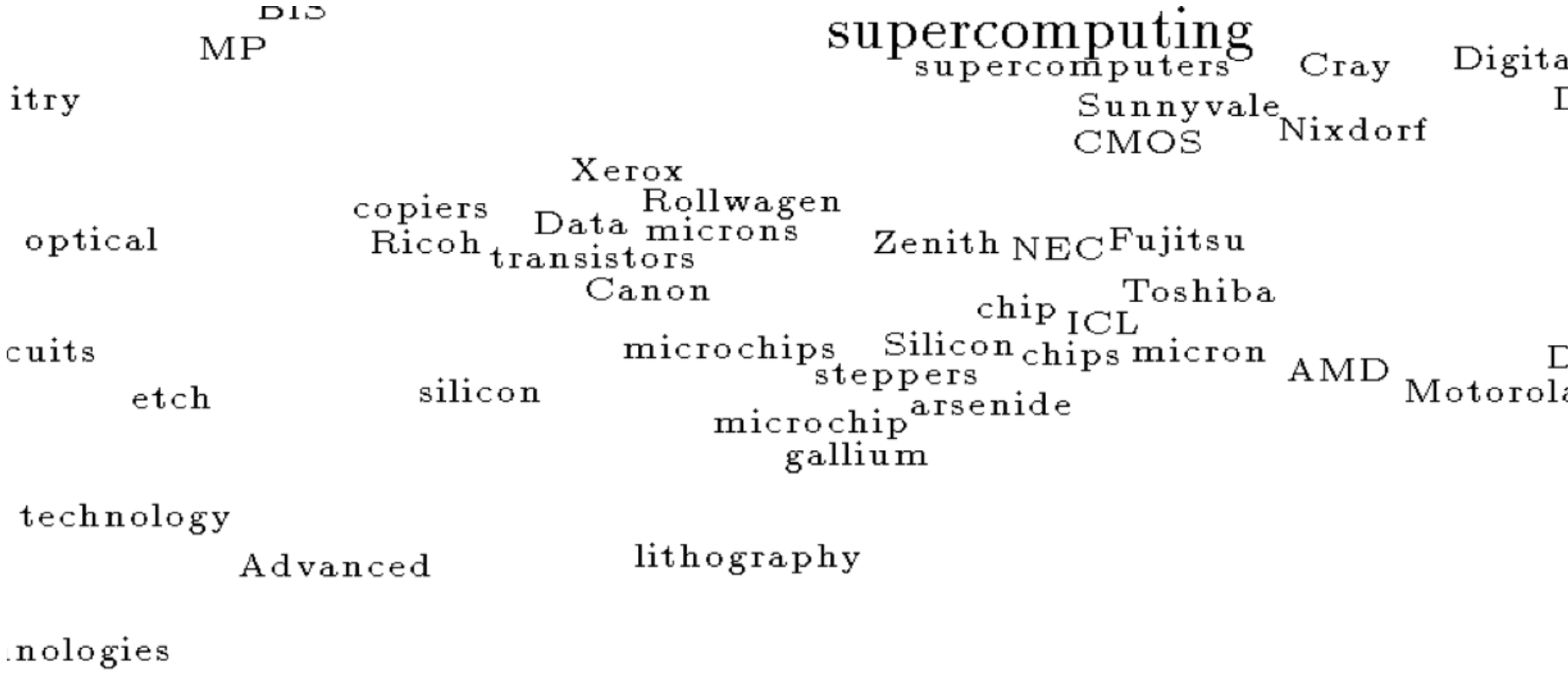
- How to understand / analyze embeddings?
- Frequently used: two-dimensional projections
- Methods / software
  - Traditional:
    multidimensional scaling, PCA
  - t-SNE
    `https://lvdmaaten.github.io/tsne/`
  - gensim
    `https://radimrehurek.com/gensim/`
  - Pretty much all methods are implemented in R:
    `https://www.r-project.org`
- Important: The two dimensions are typically not interpretable.

2D projection of embeddings

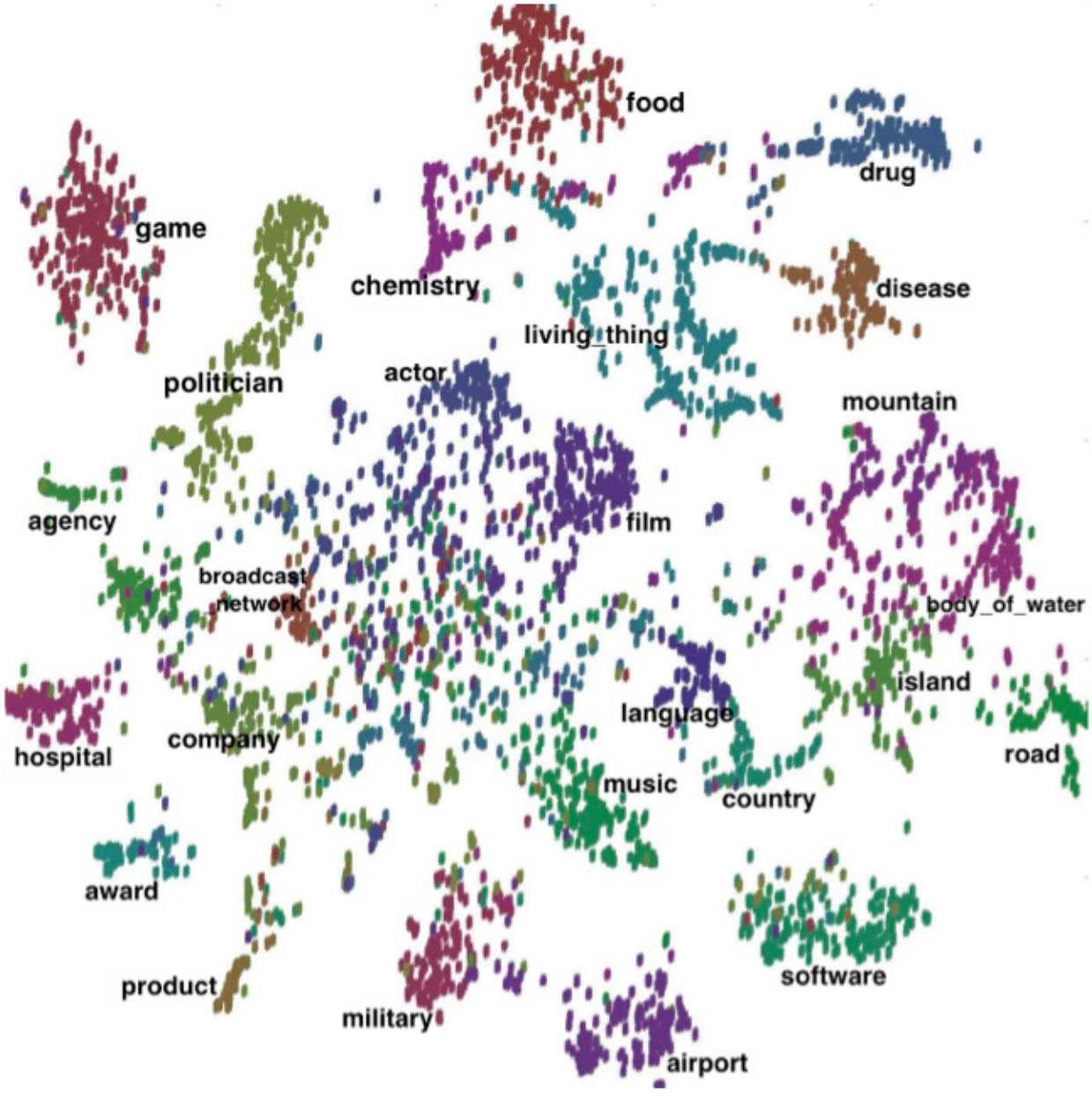


2D projection of embeddings

2D projection of embeddings



2D projection of entity embeddings



FastText

- FastText is an extension of skipgram word2vec.
- It also computes embeddings for character ngrams.
- A word's embedding is a weighted sum of its character ngram embeddings.
- Parameters: minimum ngram length: 3, maximum ngram length: 6
- The embedding of "dendrite" will be the sum of the following ngrams: @dendrite@ @de den end ndr dri rit ite te@ @den dend endr ndri drit rite ite@ @dend dendr endri ndrit drite rite@ @dendr dendri endrit ndrite drite@

FastText

- Example 1: embedding for character ngram “dendrit” $\rightarrow$ “dentrite” and “dentritic” are similar
- Example 2: embedding for character ngram “tech-” $\rightarrow$ “tech-rich” and “tech-heavy” are similar

Three frequently used embedding learners

- word2vec
  `https://code.google.com/archive/p/word2vec/`
- FastText
  `https://research.fb.com/projects/fasttext/`
- gensim
  `https://radimrehurek.com/gensim/`

```
fasttext skipgram -dim 50 -input tinycorpus.txt
-output tiny

cat ftvoc.txt | fasttext print-vectors tiny.bin >
ftvoc.vec
```

Letter n-gram generalization can be good

word2vec

1.000 automobile 779 mid-size 770 armored 763 seaplane 754 bus 754 jet 751 submarine 750 aerial 744 improvised 741 anti-aircraft

FastText

1.000 automobile 976 automobiles 929 Automobile 858 manufacturing 853 motorcycles 849 Manufacturing 848 motorcycle 841 automotive 814 manufacturer 811 manufacture

Letter n-gram generalization can be bad

word2vec

1.000 Steelers 884 Expos 865 Cubs 848 Broncos 831 Dinneen 831 Dolphins 827 Pirates 826 Copley 818 Dodgers 814 Raiders

FastText

1.000 Steelers 893 49ers 883 Steele 876 Rodgers 857 Colts 852 Oilers 851 Dodgers 849 Chalmers 849 Raiders 844 Coach

Letter n-gram generalization: no-brainer for unknowns

word2vec

("video-conferences" did not occur in corpus)

FastText

1.000 video-conferences 942 conferences 872 conference 870
Conferences 823 inferences 806 Questions 805 sponsorship 800
References 797 participates 796 affiliations

FastText skipgram parameters

- -input <path>
  training file path
- -output <path>
  output file path
- -lr <float>
  learning rate
- -lrUpdateRate <int>
  rate of updates for the learning rate
- -dim <int>
  dimensionality of word embeddings
- -ws <int>
  size of the context window
- -epoch <int>
  number of epochs

FastText skipgram parameters

- -minCount <int>
  minimal number of word occurences
- -neg <int>
  number of negatives sampled
- -wordNgrams <int>
  max length of word ngram
- -loss <string>
  loss function $\in$ { ns, hs, softmax }
- -bucket <int>
  number of buckets
- -minn <int>
  min length of char ngram
- -maxn <int>
  max length of char ngram

FastText skipgram parameters

- -threads <int>
  number of threads
- -t <float>
  sampling threshold
- -label <string>
  labels prefix
- -verbose <int>
  verbosity level

Takeaway: Three versions of word2vec skipgram

- Matrix factorization (SVD) of PPMI matrix
- skipgram negative sampling (SGNS) using GD
- hierarchical softmax

Takeaway: Embeddings learned via gradient descent

- Cost (actually reward) function is negative sampling:
  Make dot product of "true" pairs as big as possible and of "false" pairs as small as possible
- Number of parameters: $2d|V|$
- Gradient ascent

Takeaway: Visualization

- 2D or 3D visualization of embeddings
- 2D/3D visualization of high-dimensional spaces is often misleading.

Takeaway: FastText

- Learns embeddings for character ngrams
- Can handle out-of-vocabulary (OOV) words
- Sometimes you gain ("automobile"), sometimes you lose ("Steelers").