Word Embeddings

Mikhail Yurushkin
BroutonLab

October 1, 2020

Vector representations

Vector representation

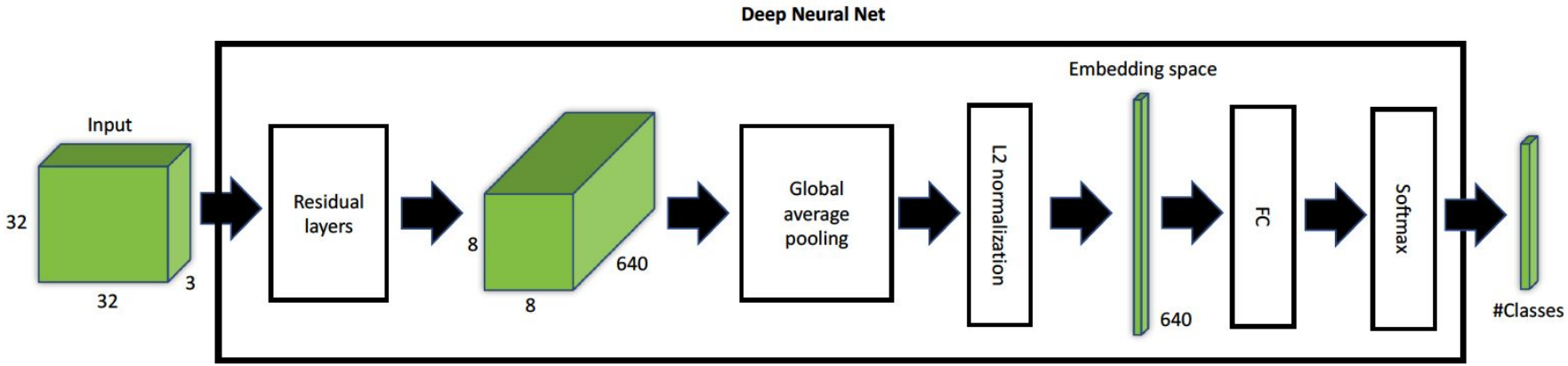
= embedding

= vector of fixed size

- typically gained in unsupervised way
- used in transfer learning
- speedup training of neural networks

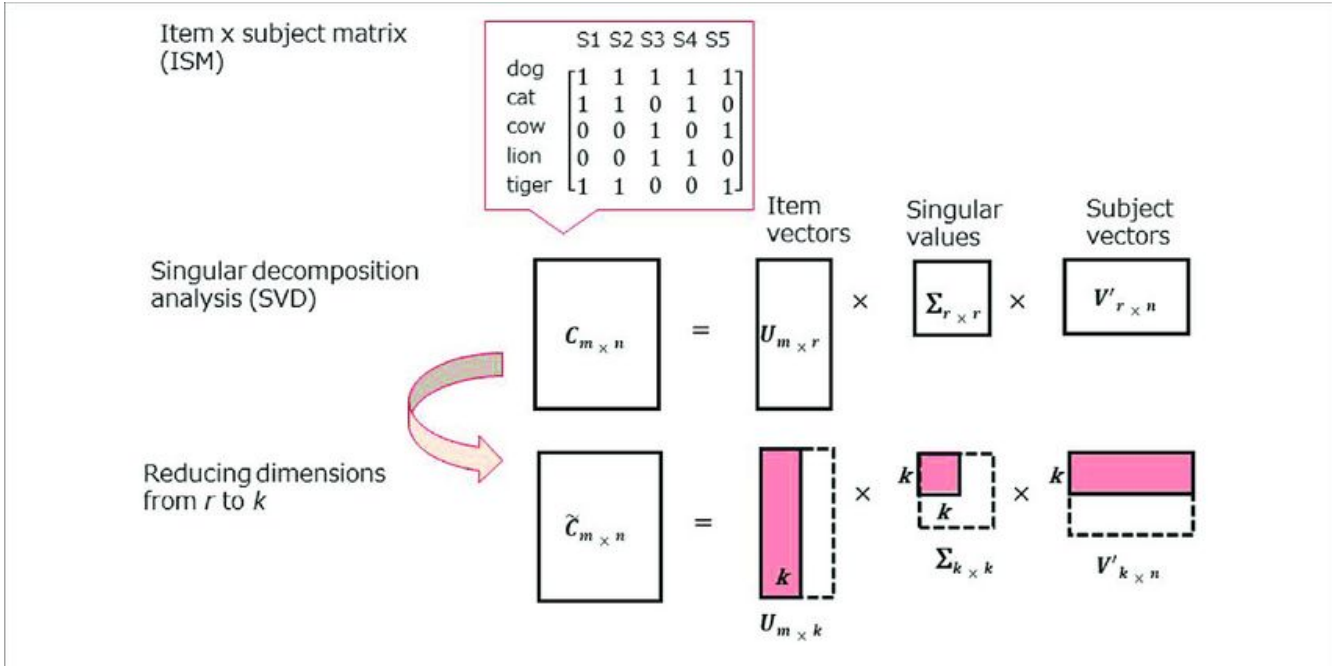
Classical example: one-hot encoding

Usage in Computer Vision



Classical approach: SVD

Not scalable :(



Word2Vec

Paper: **Efficient Estimation of Word Representations in Vector Space.** 2013, Jeffrey Dean et al.

- set of methods with the same idea
- C-BOW, Skip-Gram

X = vector(”**biggest**”)−vector(”**big**”) + vector(”**small**”)

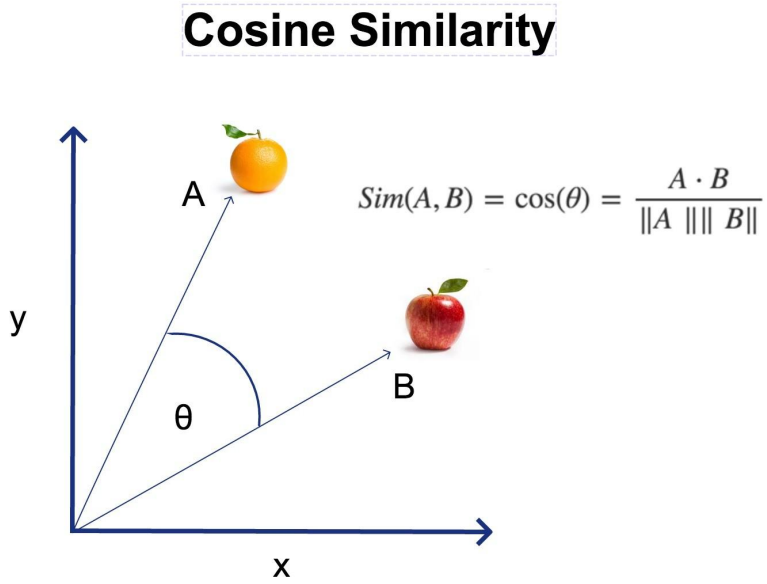
Result: **smallest**

Measuring similarity of vectors

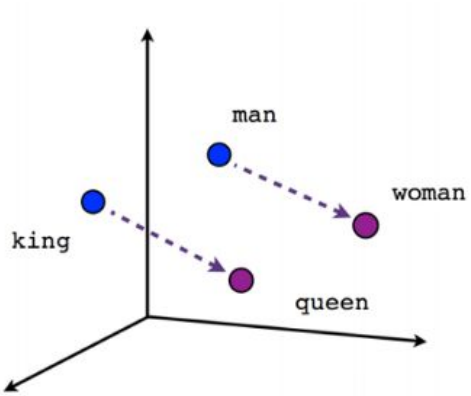
Vectors can be compared in different ways.

Most popular ways:

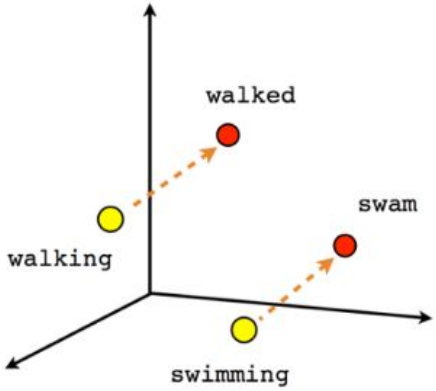
1. L2 distance
2. Cosine similarity



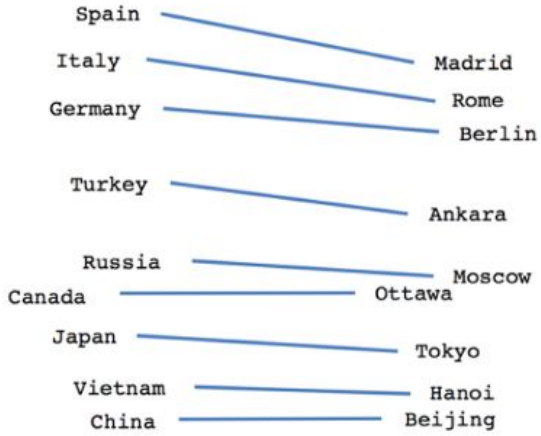
Semantic similarity



Male-Female



Verb tense



Country-Capital

Other semantic similarity examples

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Applications of word embeddings

- Search of typos or synonyms in search queries
- Can be used as a features in most of NLP tasks:
  - Named entity recognition
  - Word tagging
  - End2end machine translation
  - Sentiment analysis
  - Document classification
  - etc.

C-BOW:

- Idea: Using context words, we can predict center word

  i.e. Probability("It is (?) to finish" → "time")

  context words (window_size=2)

- Present word as distributed vector of probability → Low dimension
- Goal: Train weight-matrix($\boldsymbol{W}$) satisfies below

$$\text{argmax}_W \{Minimize(|\boldsymbol{time} - softmax(pr(\boldsymbol{time}|\boldsymbol{it}, \boldsymbol{is}, \boldsymbol{to}, \boldsymbol{finish}))|; W)\}$$

  * Softmax(): K-dim vector of $x \in \mathbb{R}$ → K-dim vector that has $(0,1) \in \mathbb{R}$

- Loss-function (using cross-entropy method)

$$E = -\log p(w_t | w_{t-C} .. w_{t+C})$$

C-BOW:

$$\mathbf{h} = W^T\mathbf{x}$$
$$\mathbf{u} = W'^T\mathbf{h} = W'^T W^T\mathbf{x}$$
$$\mathbf{y} = \mathbb{S}\text{oftmax}(\mathbf{u}) = \mathbb{S}\text{oftmax}(W'^T W^T\mathbf{x})$$

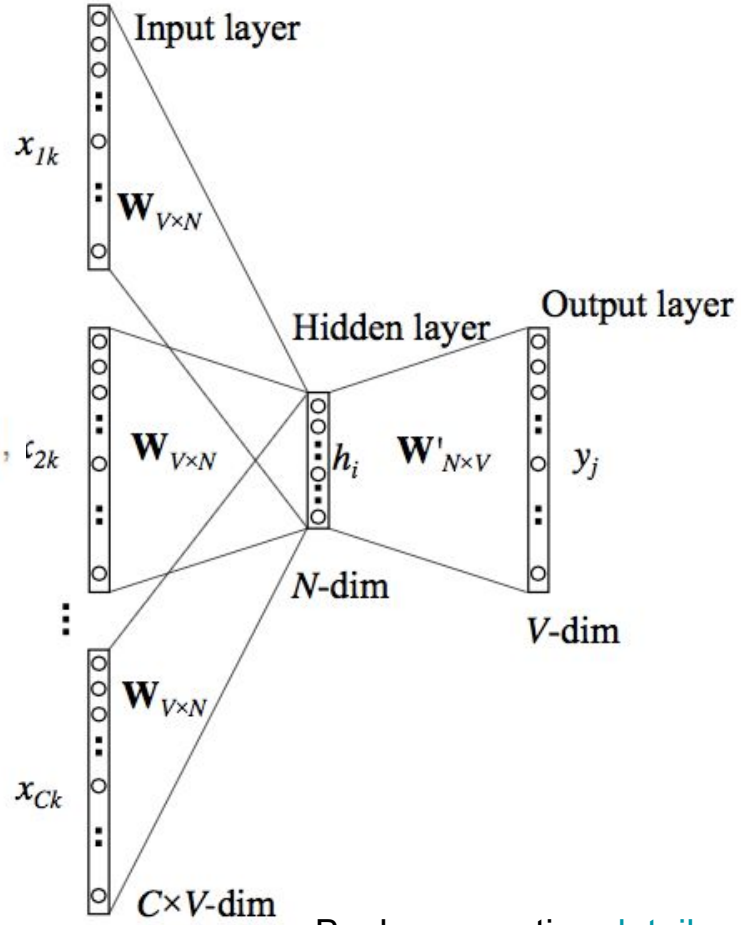
$$\mathcal{L} = -\log \mathbb{P}(w_t|w_c) = -\log y_{j^*} = -\log[\mathbb{S}\text{oftmax}(u_{j^*})] = -\log\left(\frac{\exp u_{j^*}}{\sum_i \exp u_i}\right),$$

$$\boxed{\mathcal{L} = -u_{j^*} + \log \sum_i \exp(u_i).}$$

$$\mathbf{h} = \frac{1}{C}W^T\sum_{c=1}^{C}\mathbf{x}^{(c)} = W^T\overline{\mathbf{x}}$$

$$\mathbf{u} = W'^T\mathbf{h} = \frac{1}{C}\sum_{c=1}^{C}W'^T W^T\mathbf{x}^{(c)} = W'^T W^T\overline{\mathbf{x}}$$

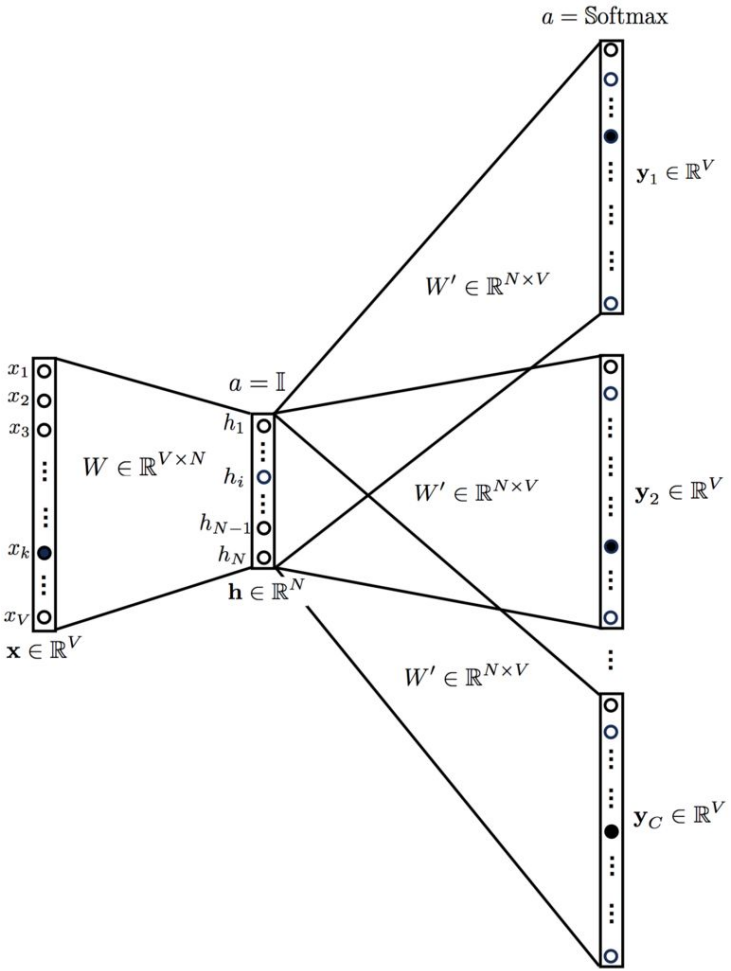
$$\mathbf{y} = \mathbb{S}\text{oftmax}(\mathbf{u}) = \mathbb{S}\text{oftmax}\left(W'^T W^T\overline{\mathbf{x}}\right)$$



Backpropagation details

Skip-Gram:

Inversed C-BOW

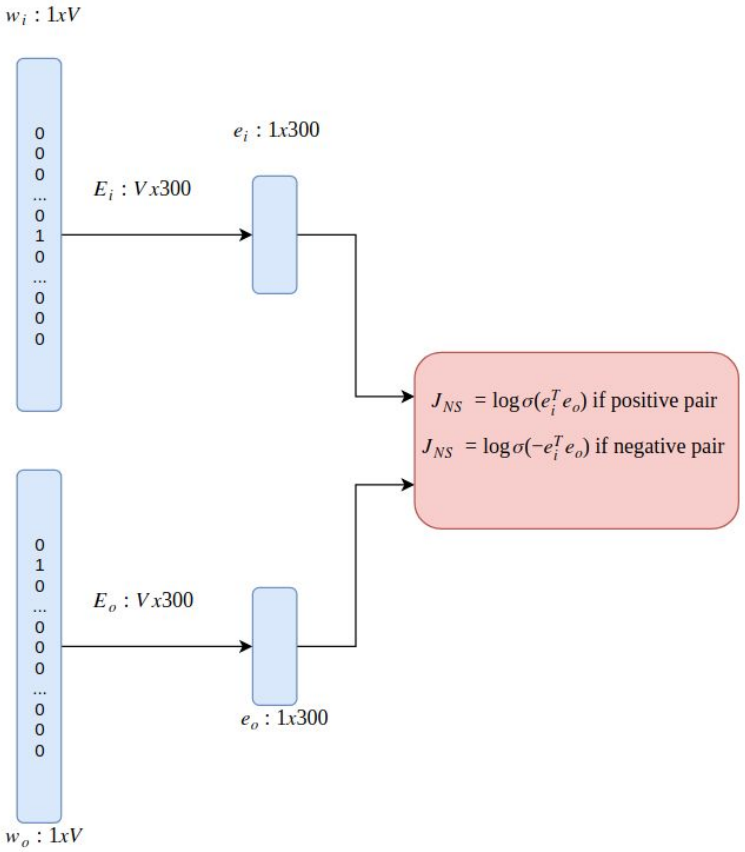


Negative sampling

- Difficult to calculate big softmax?
- let's solve binary classification problem!

$$\mathcal{L} = \sum_{(w,s)\in D_1} \log \sigma(v_w^T v_s) + \sum_{(w,s)\in D_2} \log \sigma(-v_w^T v_s),$$

$$D_1 = \{(w,s) : s \in c(w)\}, D_2 = \{(w,s) : s \notin c(w)\}$$



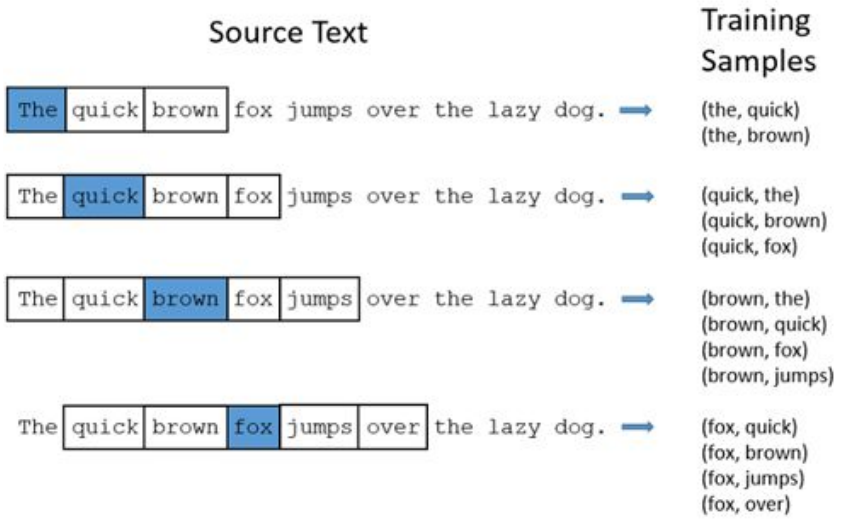
Negative sampling

: Center Word
: Context Word

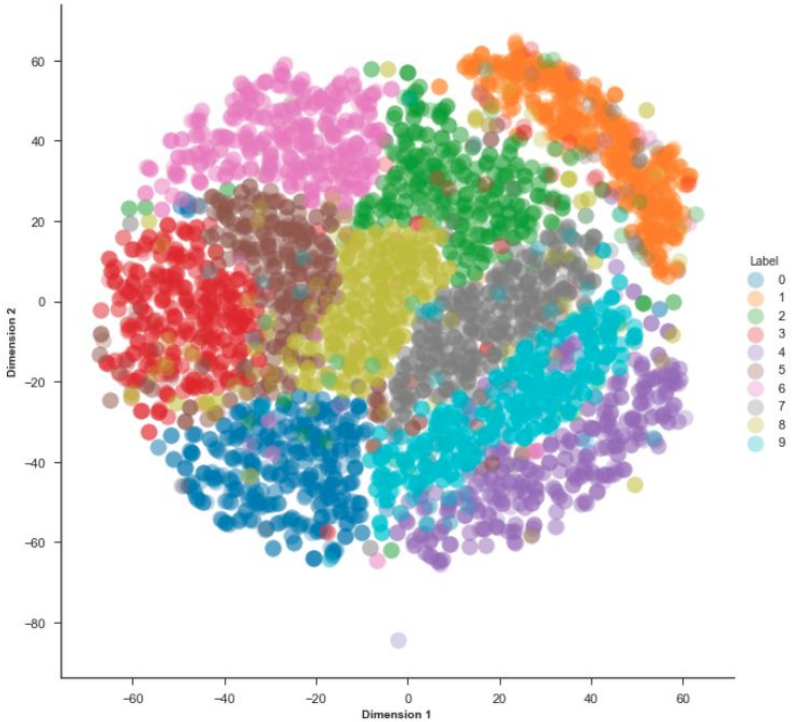
c=0 The cute cat jumps over the lazy dog.

c=1 The cute cat jumps over the lazy dog.

c=2 The cute cat jumps over the lazy dog.



How to check your embeddings are good?



2-D t-SNE

- Unsupervised
- clusterization
- non-linear technique primarily used for data exploration and visualizing high-dimensional data.