Text Visualization in Social Science

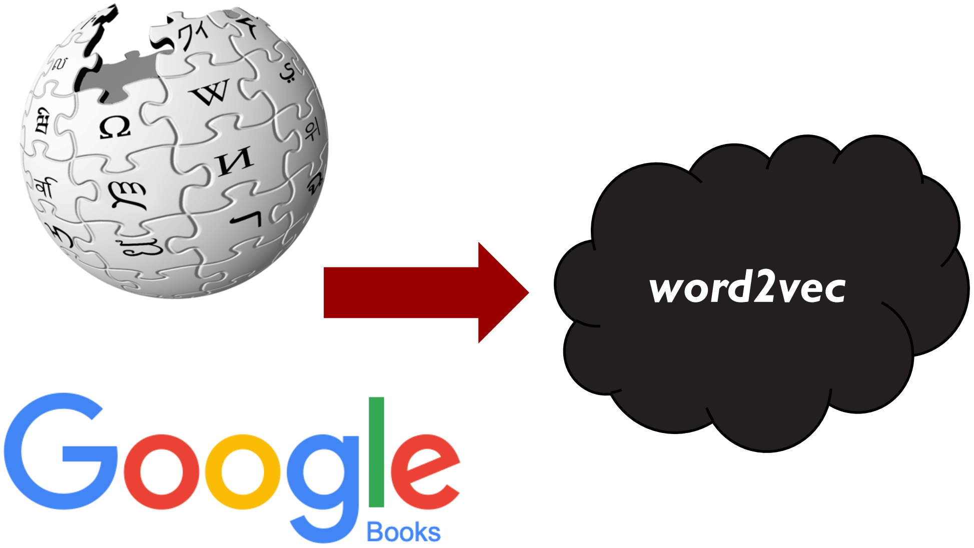
Yongjun Zhang, Ph.D.
Dept of Sociology and IACS
https://yongjunzhang.com

Today's Agenda

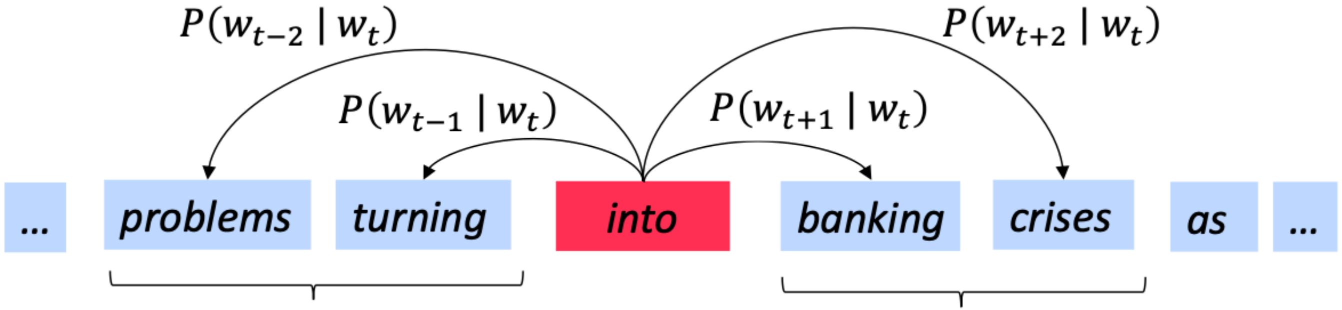
1. Guest Speaker Dr. Dustin Stoltz (6:00-6:50PM)
2. More on Word Embedding and Text Visualization (7:00-7:50 PM)
3. Lab Tutorial on Text Visualization (8:00-8:50PM)

Dr. Dustin Stoltz
Ass. Prof. of Sociology
@Lehigh University

More on Word Embeddings



	D1	D2	D3	D4
W1	.02	.03	.5	.45
W2				
W3				
W4				



V= c("problems", "turning", "into", "banking", "crises")

Training Samples

Window size=2

problems	turning	into	banking	crises	as
problems	turning	into	banking	crises	as

(into, turning)
(into, problems)
(into, banking)
(into, crises)

(banking, turning)
(banking, into)
(banking, crises)
(banking, as)

② Exponentiation makes anything positive

$$P(o|c) = \frac{\exp(\boxed{u_o^T v_c})}{\boxed{\sum_{w \in V} \exp(u_w^T v_c)}}$$

① Dot product compares similarity of *o* and *c*.
$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$
Larger dot product = larger probability

③ Normalize over entire vocabulary to give probability distribution

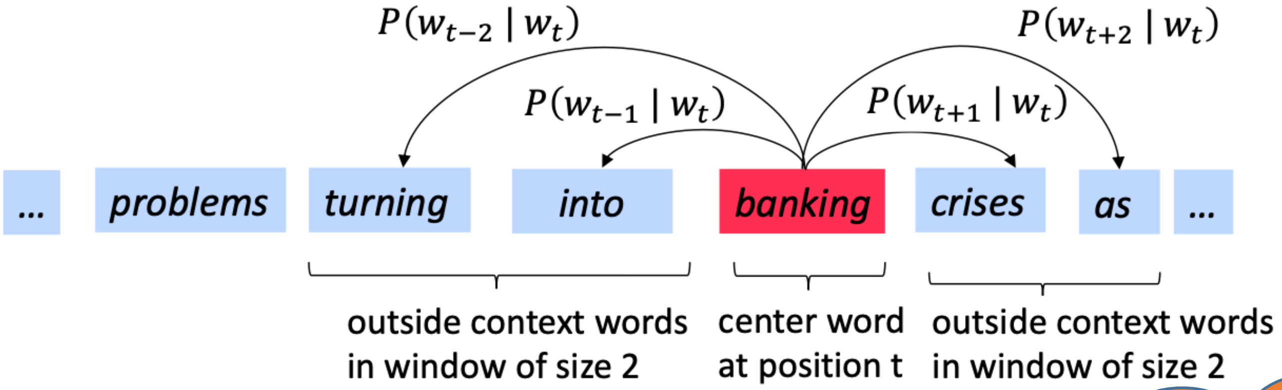
- This is an example of the **softmax function** $\mathbb{R}^n \rightarrow (0,1)^n$

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)} = p_i$$

Open region

- The softmax function maps arbitrary values x_i to a probability distribution p_i
  - "max" because amplifies probability of largest x_i
  - "soft" because still assigns some probability to smaller x_i
  - Frequently used in Deep Learning

Manning, CS224n, 2020



For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_j.

Likelihood = $L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} \mid w_t; \theta)$

θ is all variables to be optimized

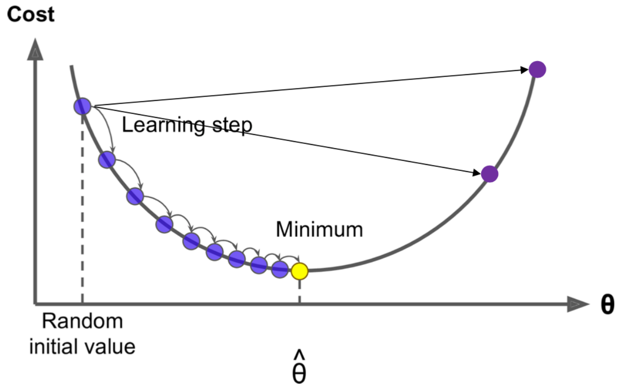
sometimes called *cost* or *loss* function

The objective function $J(\theta)$ is the (average) negative log likelihood:

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log P(w_{t+j} \mid w_t; \theta)$$

Can anyone tell me why the cost function is –1/TlogL(theta)??

- We have a cost function $J(\theta)$ we want to minimize
- **Gradient Descent** is an algorithm to minimize $J(\theta)$
- Idea: for current value of θ, calculate gradient of $J(\theta)$, then take **small** step in the direction of negative gradient. Repeat.



Note: Our objectives may not be convex like this

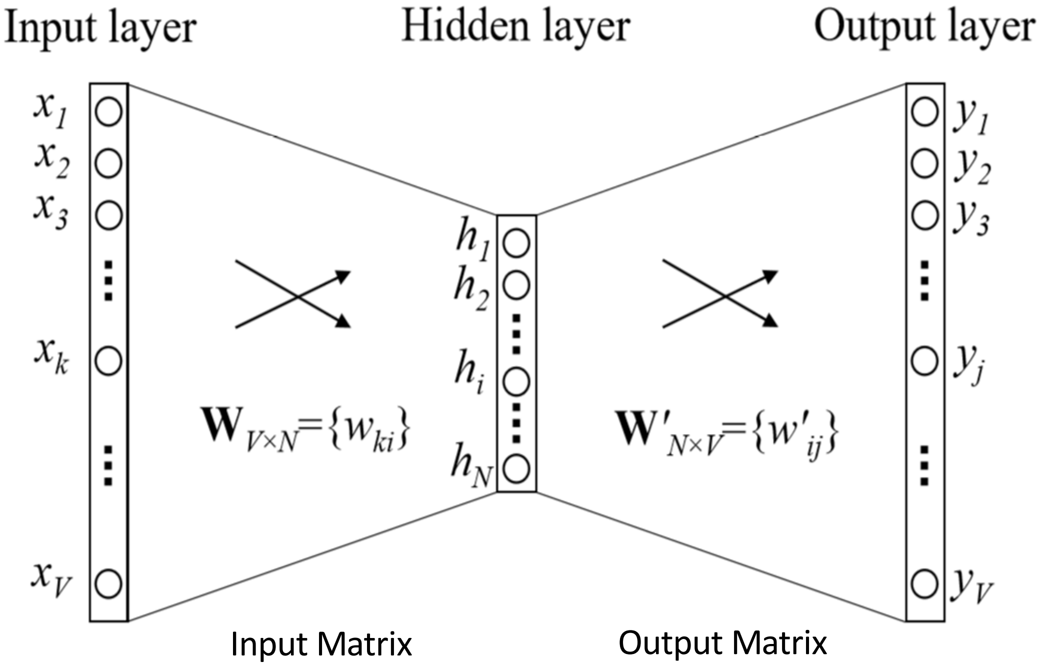
- Update equation (in matrix notation):

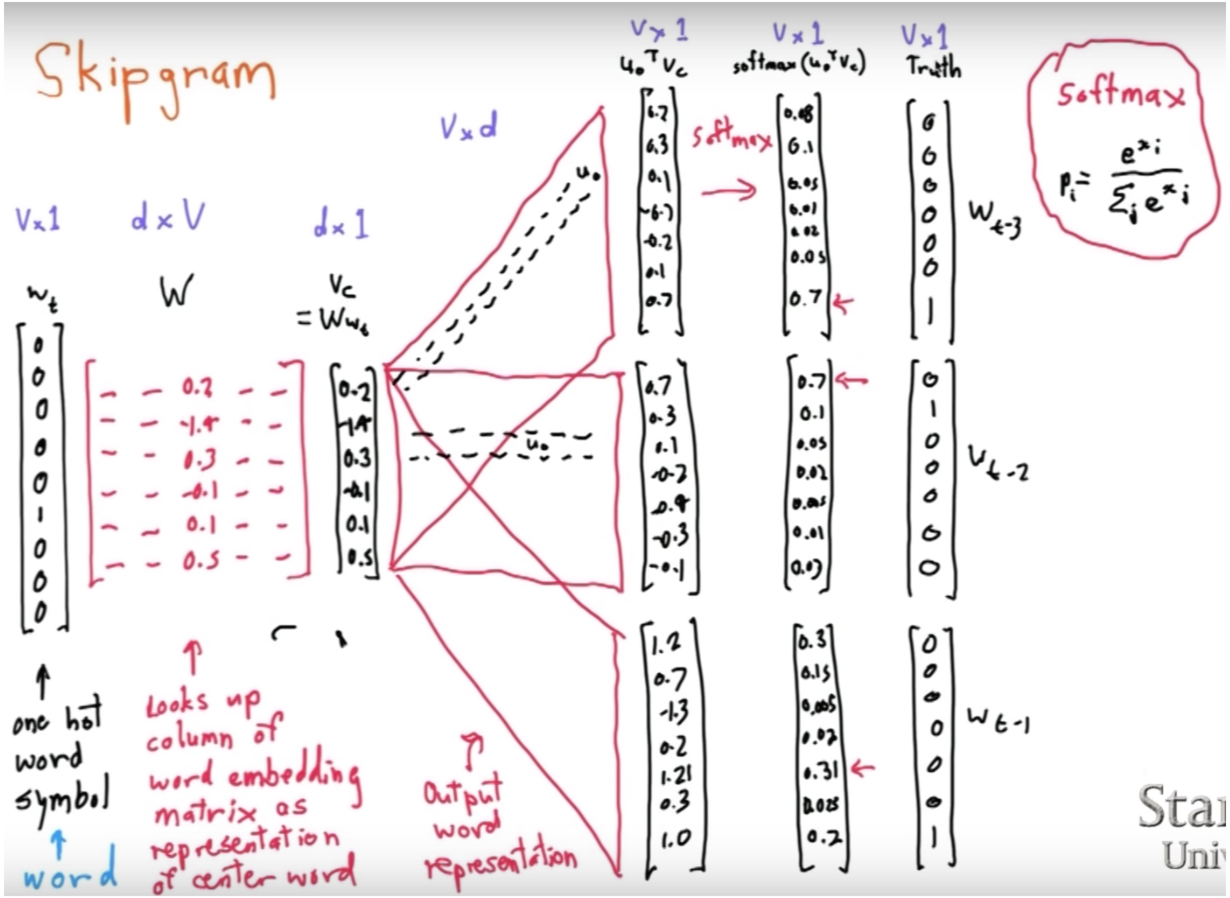
$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$$

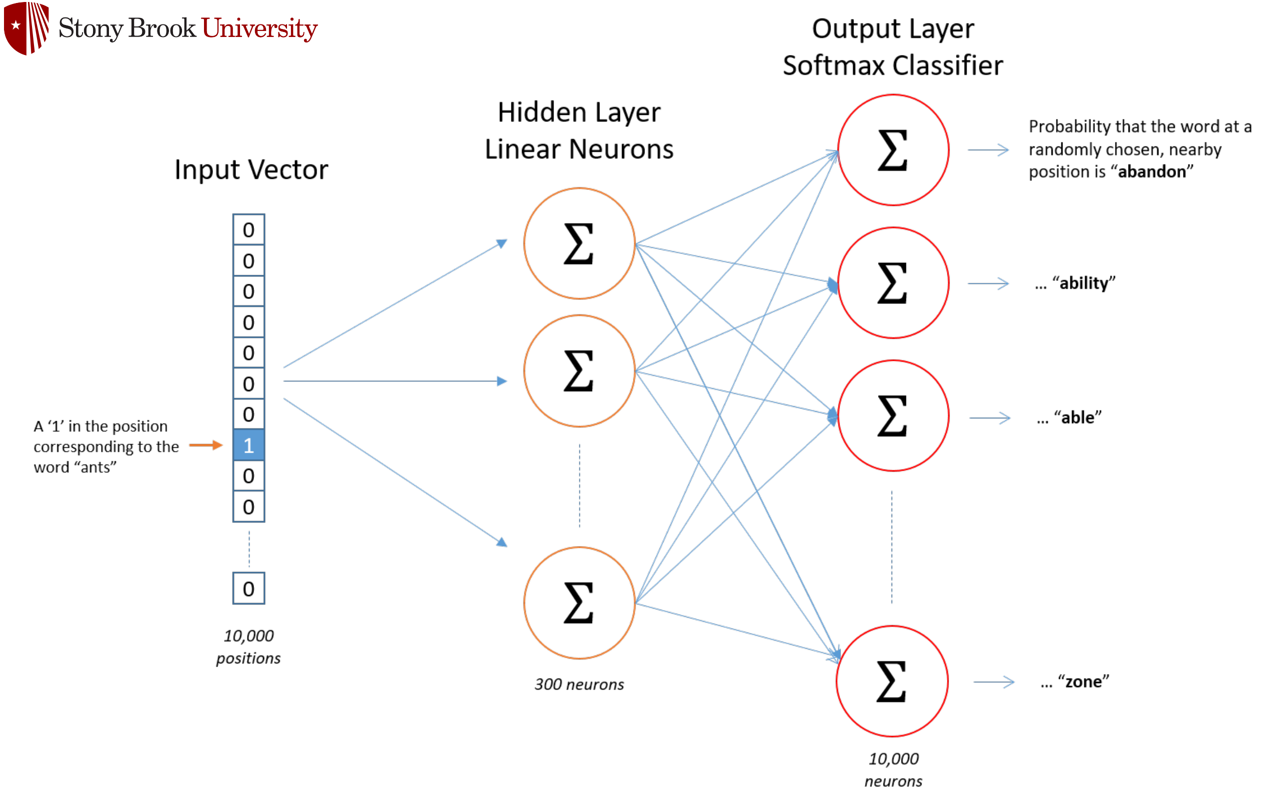
α = *step size* or *learning rate*

- Update equation (for a single parameter):

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial}{\partial \theta_j^{old}} J(\theta)$$

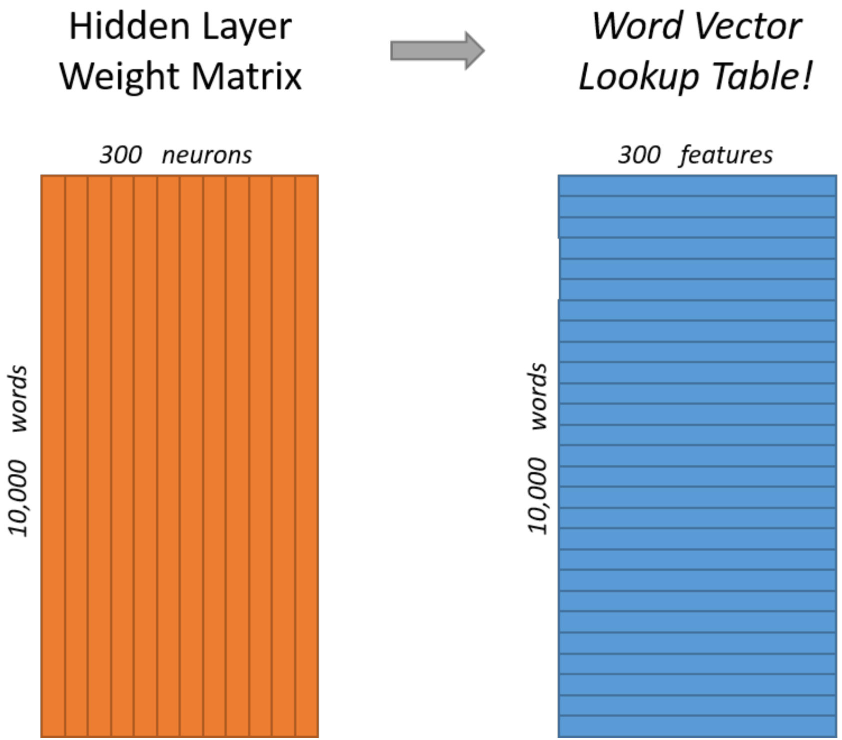

Input layer
Hidden layer
Output layer
x_1
x_2
x_3
x_k
x_V
h_1
h_2
h_i
h_N
y_1
y_2
y_3
y_j
y_V
$\mathbf{W}_{V \times N} = \{w_{ki}\}$
$\mathbf{W}'_{N \times V} = \{w'_{ij}\}$
Input Matrix
Output Matrix


Skipgram
V x 1
d x V
d x 1
V x d
V x 1
V x 1
V x 1
$u_o^T v_c$
softmax($u_o^T v_c$)
Truth
w_t
W
$v_c = W w_t$
u_o
softmax
w_{t-3}
w_{t-2}
w_{t-1}
Softmax
$p_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$
one hot word symbol
word
Looks up column of word embedding matrix as representation of center word
Output word representation



When training this network on word pairs, the input is a one-hot vector representing the input word and the training output is also a one-hot vector representing the output word. But when you evaluate the trained network on an input word, the output vector will actually be a probability distribution (i.e., a bunch of floating point values, not a one-hot vector).

http://mccormickml.com/



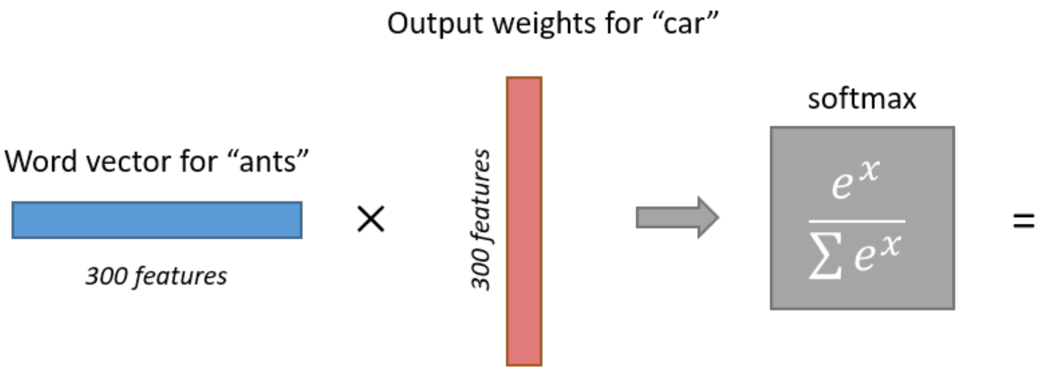
http://mccormickml.com/

Input Vector

Input Weight Matrix

Word Vector

$$[0 \quad 0 \quad 0 \quad \boxed{1} \quad 0] \times \begin{bmatrix} 17 & 24 & 1 \\ 23 & 5 & 7 \\ 4 & 6 & 13 \\ \boxed{10} & \boxed{12} & \boxed{19} \\ 11 & 18 & 25 \end{bmatrix} = [10 \quad 12 \quad 19]$$



Probability that if you randomly pick a word nearby “ants”, that it is “car”

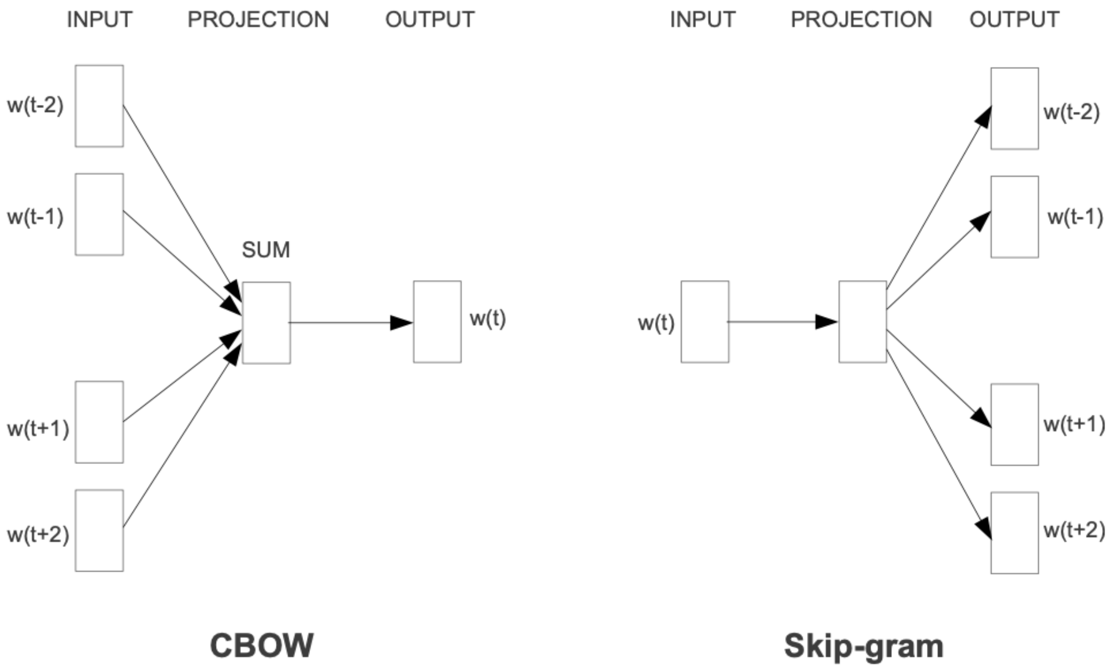


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

Some Pre-trained Models

Google Word2Vec

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

Improvements and pre-trained models for word2vec:

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

https://fasttext.cc/ (by Facebook)

More on Text Visualization

Thank you!

Yongjun Zhang, Ph.D
Assistant Professor
Dept of Sociology and
Institute for Advanced Computational Science
Stony Brook University
Yongjun.Zhang@stonybrook.edu
https:yongjunzhang.com