Distributed Representations of Words & Phrases and their Compositionality

Tomas Mikolov Ilya Sutskever Kai Chen Greg Corrado Jeffrey Dean
(Google)

NIPS '13

Noah Lee (noahlee357@korea.ac.kr)
2021.09.18

01 Problem Statement

How can we improve both **quality** of word representation & **training speed** from Word2Vec?

Motivation

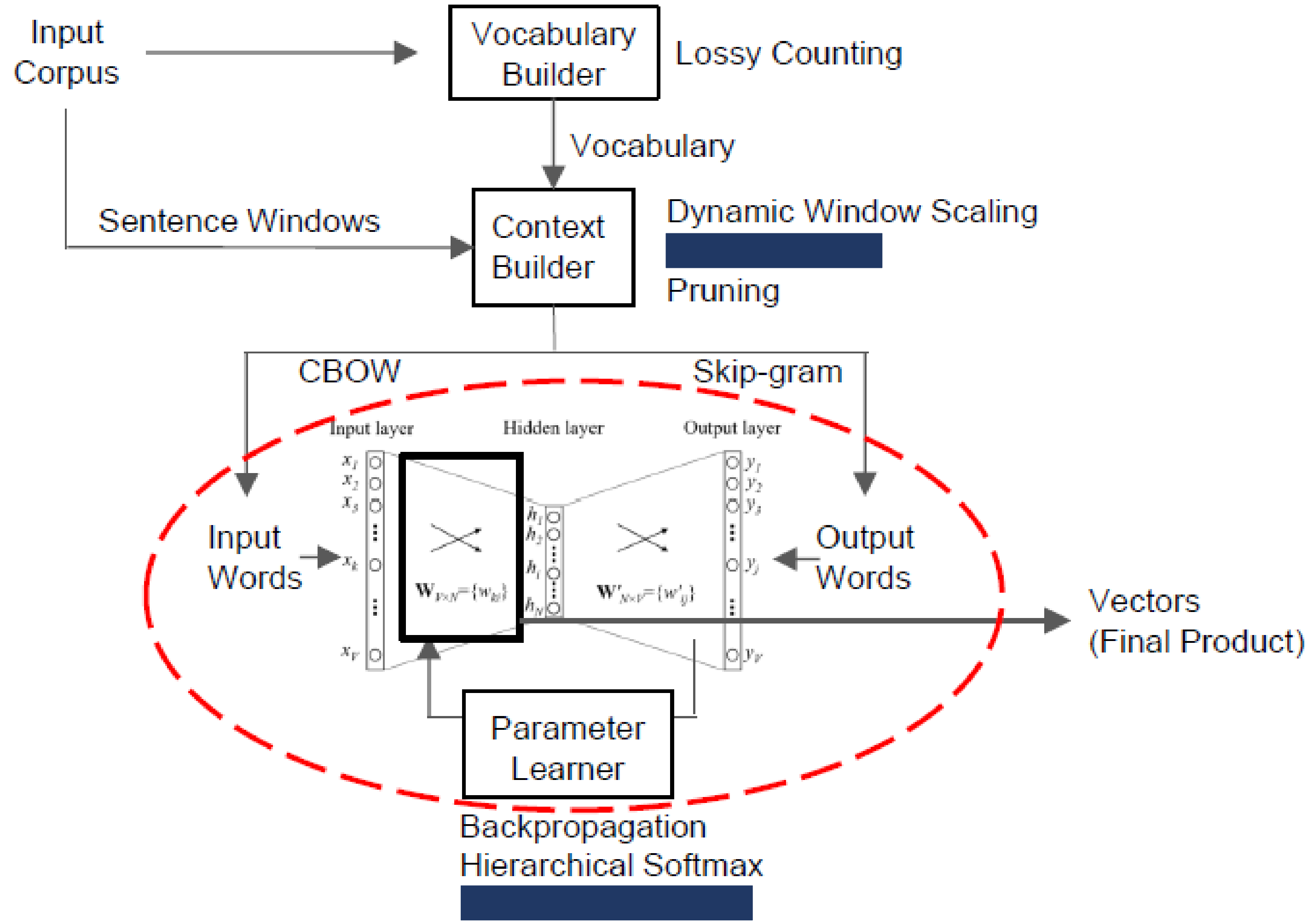
Word2Vec*

Sparse Representation		Continuous Representation
E.g. [1,0,0,0]		E.g. [0.6, 0.7]

As to capture semantic & syntactic relations…

*Efficient Estimation of Word Representations in Vector Space , Mikolov et al., '13

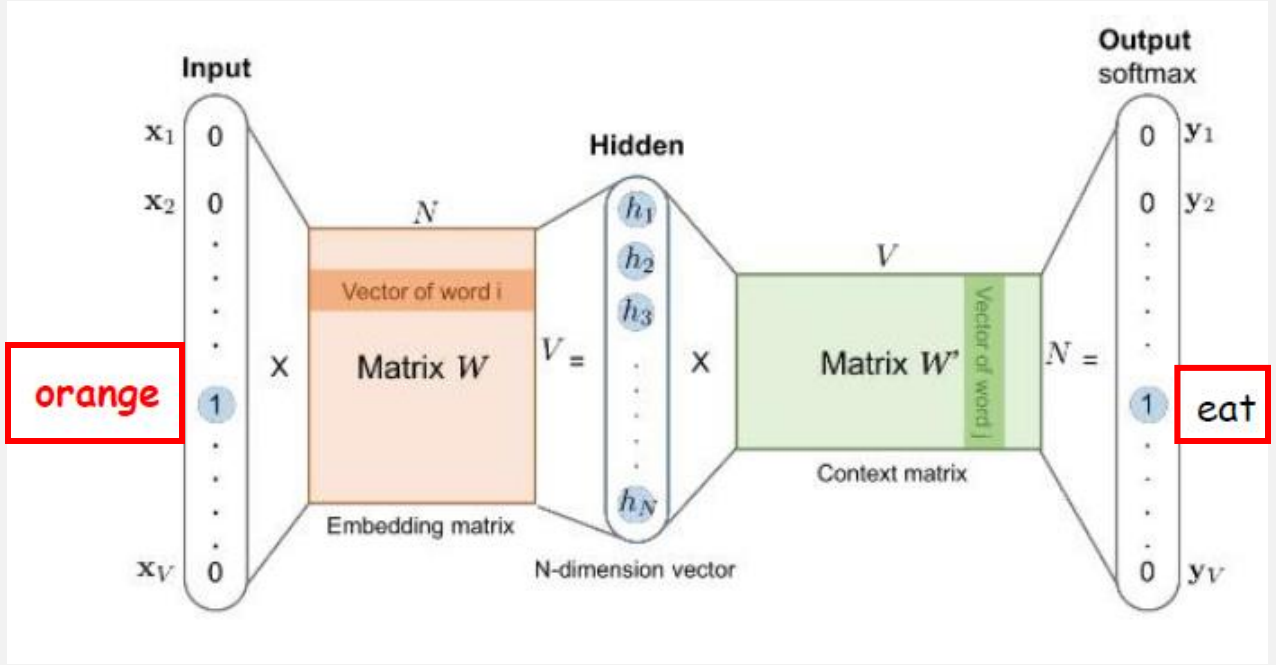
Word2Vec


Input
Corpus
Vocabulary
Builder
Lossy Counting
Vocabulary
Sentence Windows
Context
Builder
Dynamic Window Scaling
[redacted]
Pruning
CBOW
Skip-gram
Input layer
Hidden layer
Output layer
Input
Words
Output
Words
Vectors
(Final Product)
Parameter
Learner
Backpropagation
Hierarchical Softmax
[redacted]

03 Previous Work

Word2Vec: Skip-gram Overview

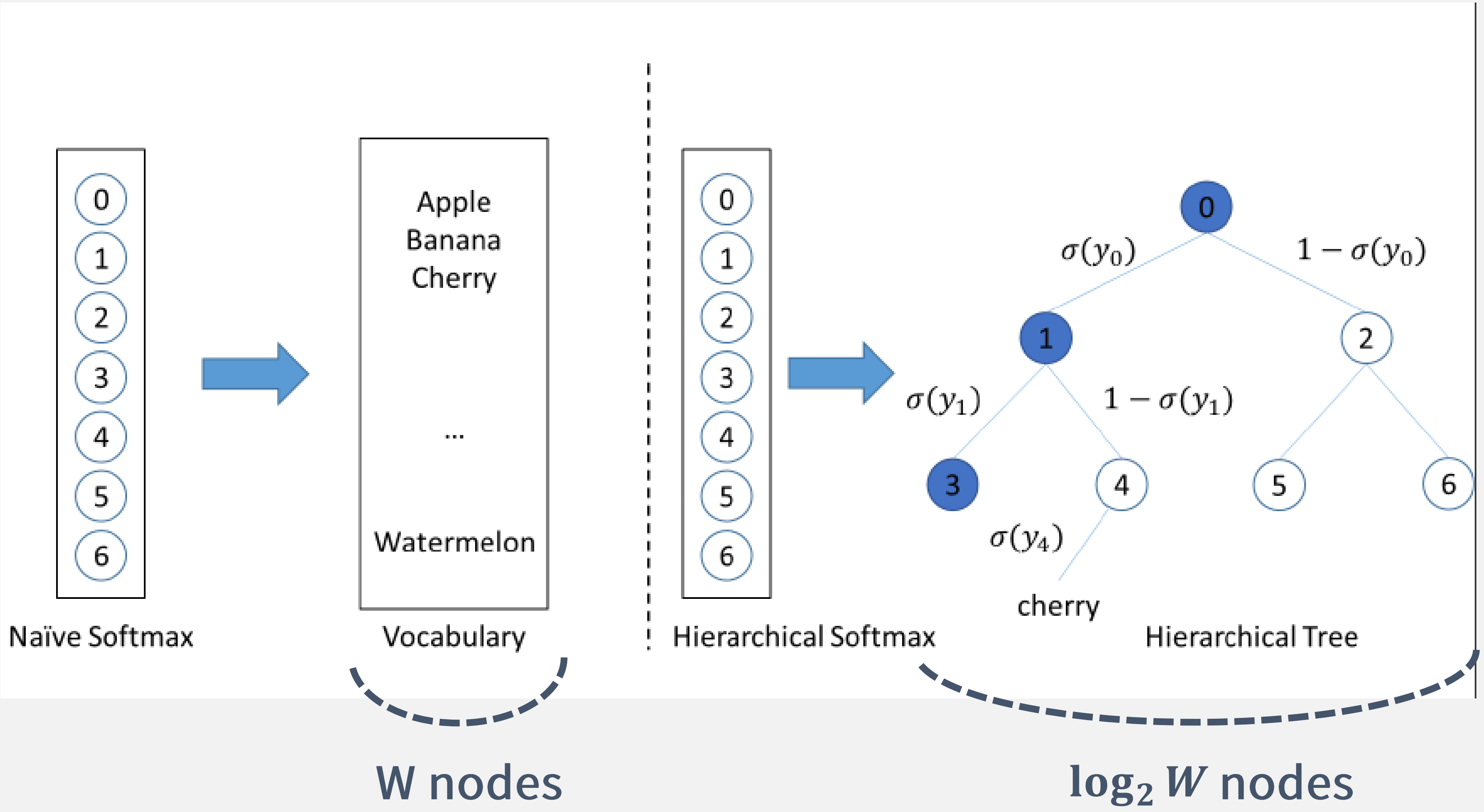
I eat an orange every day



Target Word ⇒⇒ Context Words

03 Previous Work

Word2Vec: Hierarchical Softmax



04 Contribution

1) Proposes an extension for Word2Vec in order to boost training efficiency

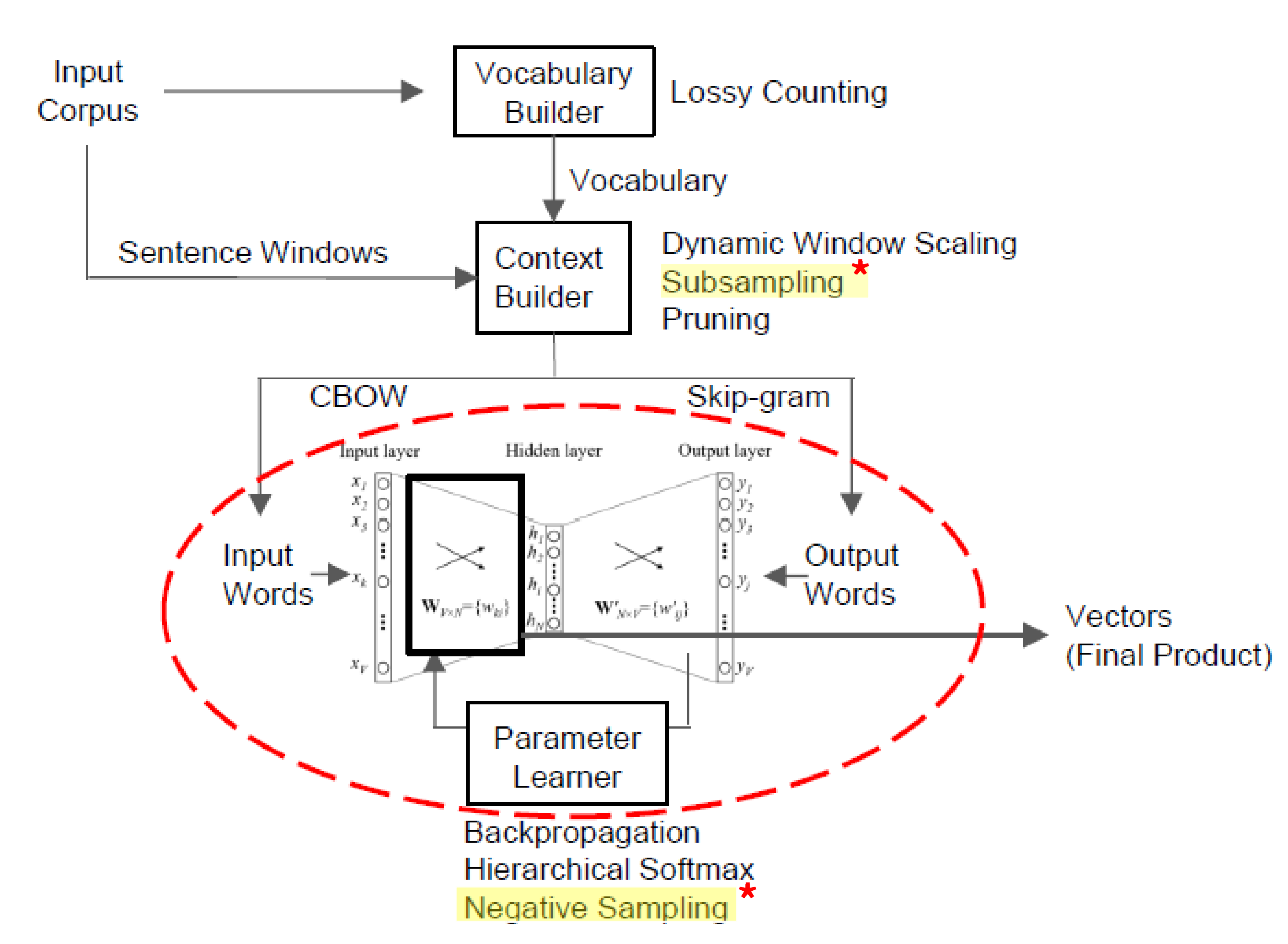
2) Proposes (1) **subsampling** & (2) **negative sampling**, the latter being a simpler alternative for Hierarchical Softmax*

3) Suggests a simple method for **finding phrases** in text

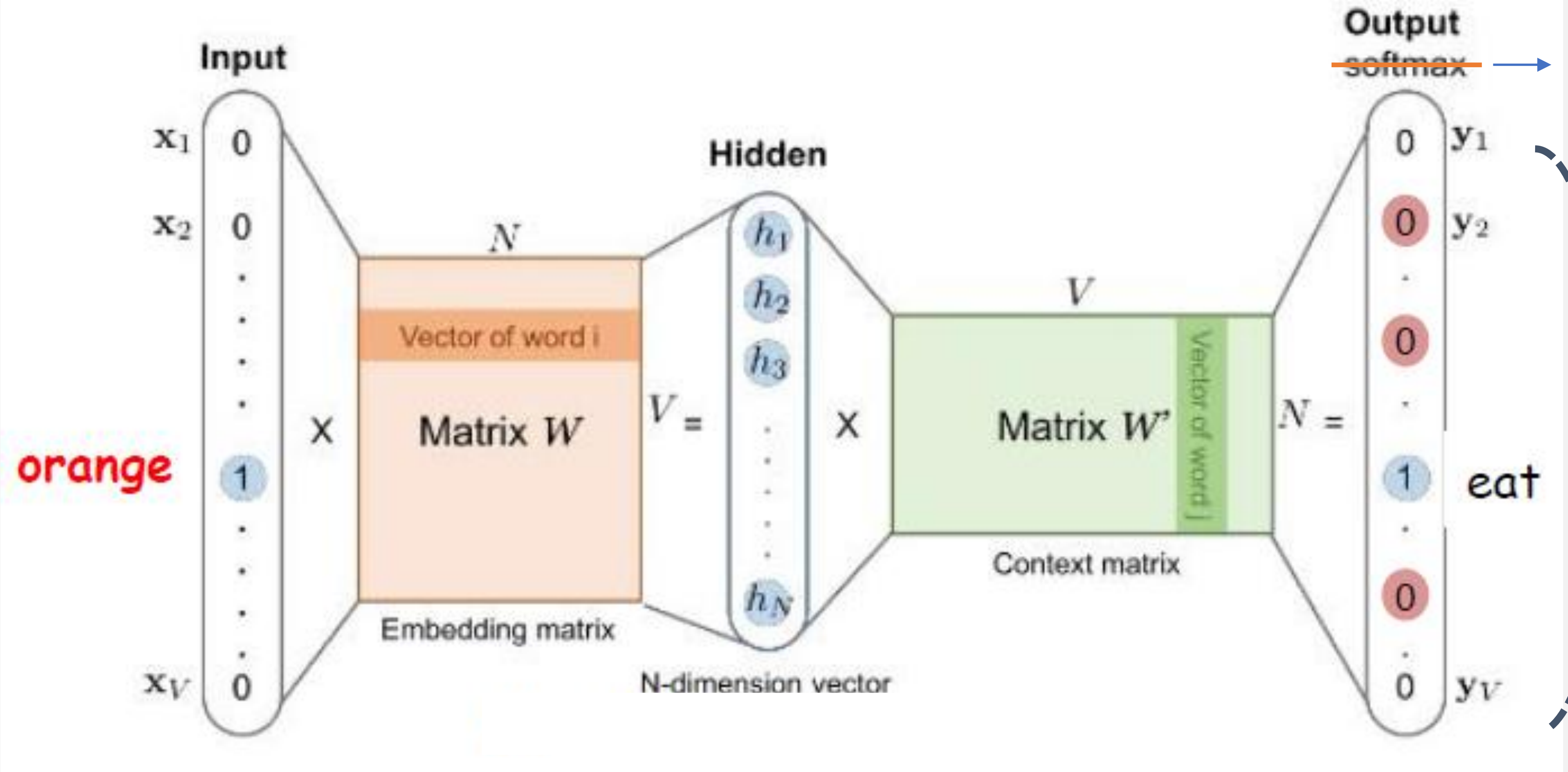
*Efficient Estimation of Word Representations in Vector Space , Mikolov et al., ‘13

04 Methodology

Extensions



Extension 1 : Negative Sampling

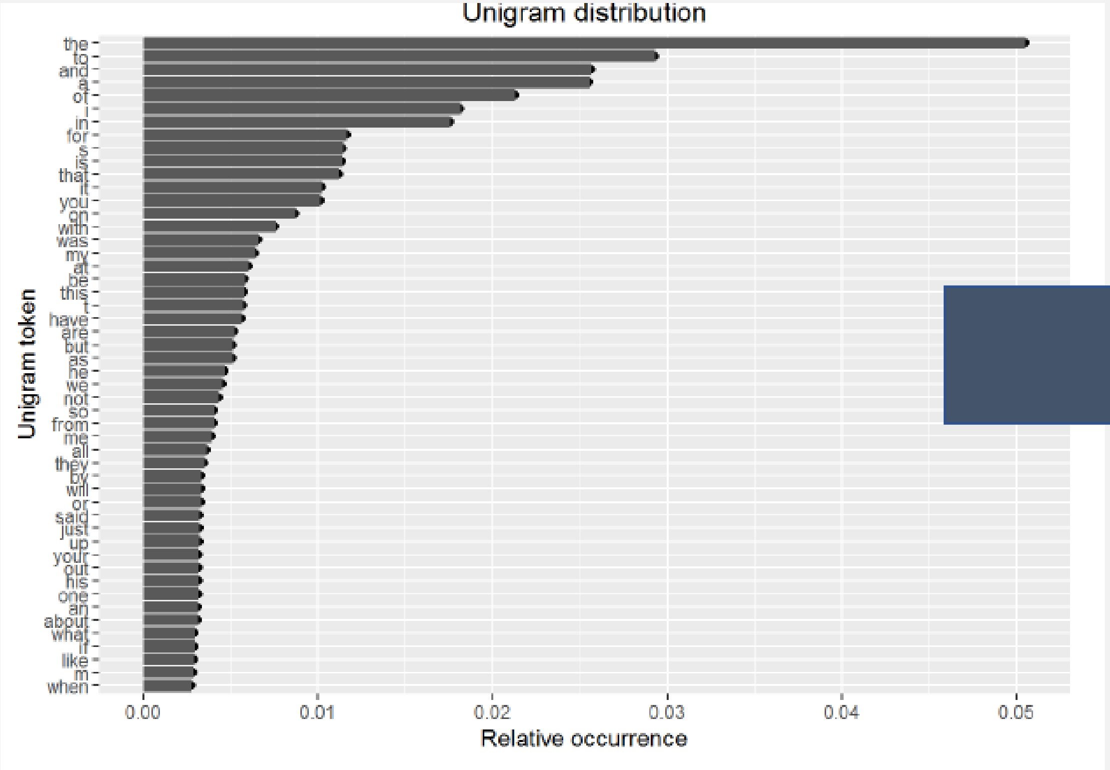


sigmoid

Only look up selected words from the context matrix

04 Methodology

Extension 1 : Negative Sampling



$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n} f(w_j)^{3/4}}$$

$f(w_i) = freq\ of\ word\ i$

Dataset ↑ : 3~5 negative samples
Dataset ↓ : 5~20 negative samples

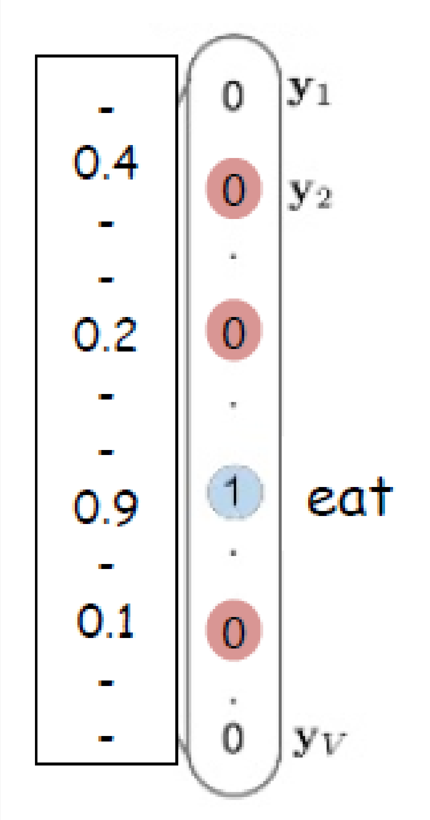
04 Methodology

Naïve vs HS vs NS

Naïve Softmax	2.2M x 300 = 660 M
Hierarchical Softmax	log(2.2M) x 300 = 6.3K
Negative Sampling	$6_{(1+ k=5)}$ x 300 = 1.8K

Additionally incorporate the data size (840 B)···

Output Dimension: 2.2M
Feature Dimension 300



04 Methodology

Extension 2: Subsampling

The orange is the fruit of the citrus species Citrus × sinensis in the family Rutaceae. It is also called sweet orange, to distinguish it from the related Citrus × aurantium, referred to as bitter orange. The sweet orange reproduces asexually varieties of sweet orange arise through mutations.

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

t = threshold (heuristically 10^-5)

1) Most frequent words ironically provide less info

OR 2) Vector representations do not change significantly after training on millions of examples

Dynamic input leads to robustness?

04 Methodology

Extension 3: Learning Phrases

Phrases cannot always be interpreted by the meaning of their separate words

Process:

1. Find words that appear frequently together (& not in other contexts) and replace e.g. “New York Times” -> single token, “this is ” -> replace x

2. Train Skip-gram model using unigram & bigram counts

Phrase Analogy Task

New York : New York Times → San Jose : San Jose Mercury News

06 Experiment

Accuracy of Skip-Gram(300d) for analogy reasoning task

Method	Time [min]	Syntactic [%]	Semantic [%]	Total accuracy [%]
NEG-5	38	63	54	59
NEG-15	97	63	58	61
HS-Huffman	41	53	40	47
NCE-5	38	60	45	53
The following results use 10^{-5} subsampling				
NEG-5	14	61	58	60
NEG-15	36	61	61	61
HS-Huffman	21	52	59	55

- Noticeable time reduction by Negative Sampling(NS), subsampling
- NS with larger samples lead to a significant performance increasement

06 Experiment

Accuracies of Skip-Gram Models on phrase analogy task

Method	Dimensionality	No subsampling [%]	10^{-5} subsampling [%]
NEG-5	300	24	27
NEG-15	300	27	42
HS-Huffman	300	19	$\boxed{47}$

- HS model with an increase of 28% by subsampling - can better performance (not causal)
- NEG model with larger k performs better (due to size of dataset…?)

- By adding 33 B words, the HS model with dimension of 1000 achieves 72%
  - Data size matters…again?

07 Conclusion

1) Proposes an extension for Word2Vec in order to boost **training efficiency**

2) Proposes **subsampling** & **negative sampling**, yielding a fraction of the original computational complexity

3) Suggests a simple method for **finding phrases** in text··· FastText?

Q&A