Word Embeddings and Their Use in LSA-like Models

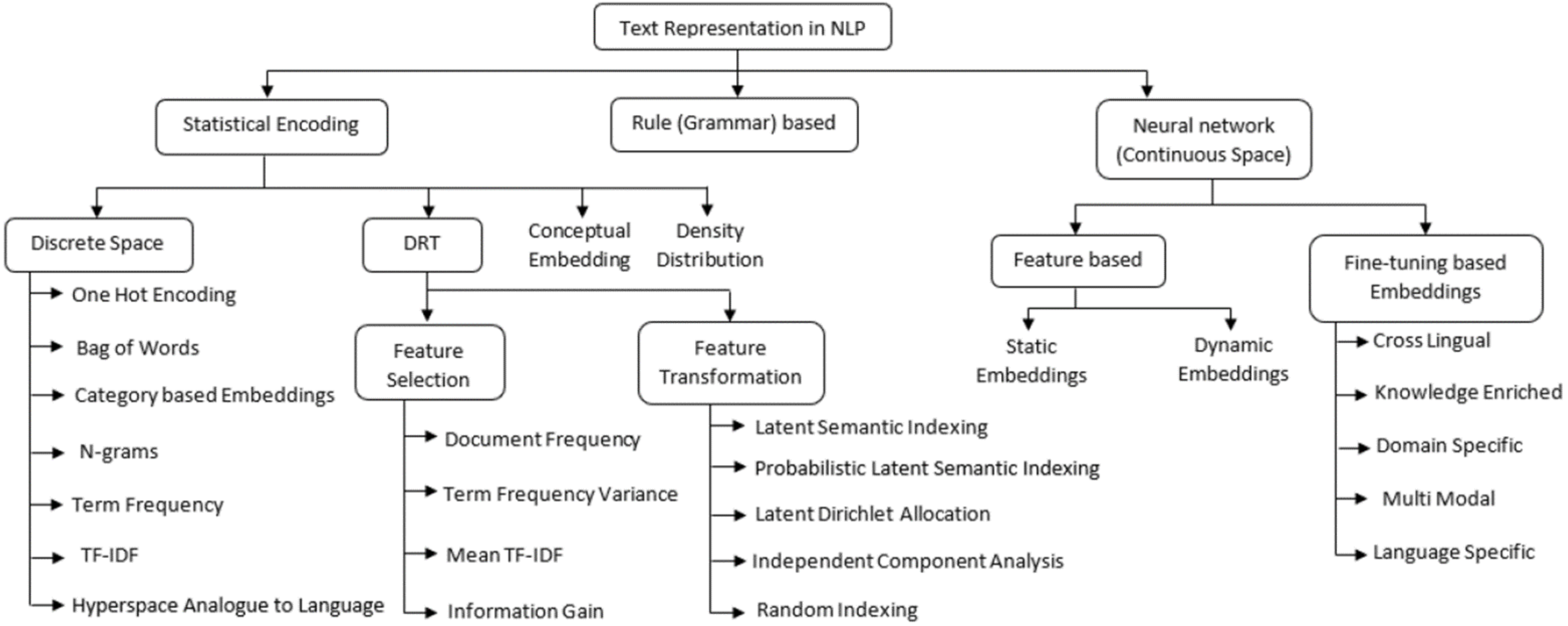
NATURAL LANGUAGE PROCESSING

KEERTHANA GOKA
KUNAL MALHAN

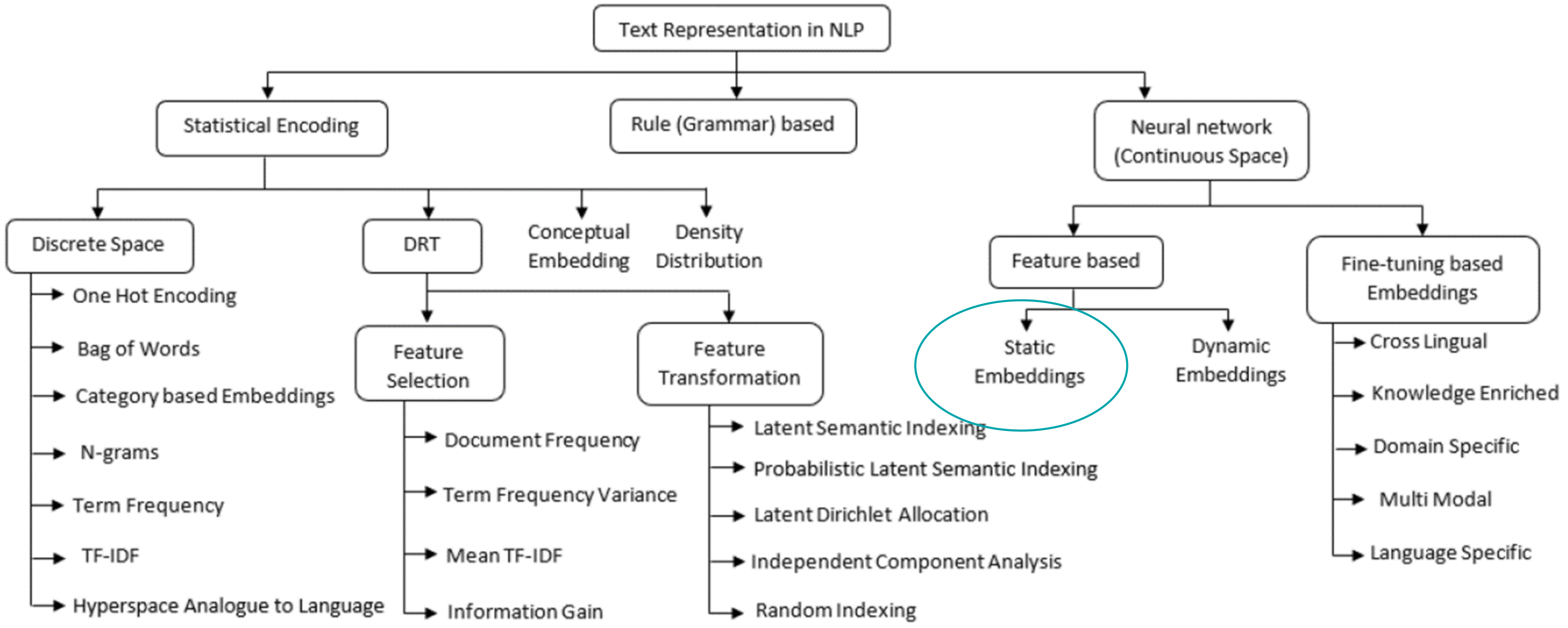
Introduction

- Why Word Embeddings?
  - For machine learning and deep learning algorithms data should be in numeric form, as these algorithms cannot understand text data.
  - Through Word Embeddings text data is converted into N-dimensional numeric data, so that computer can understand that.

Text Representation in NLP (Revisit)

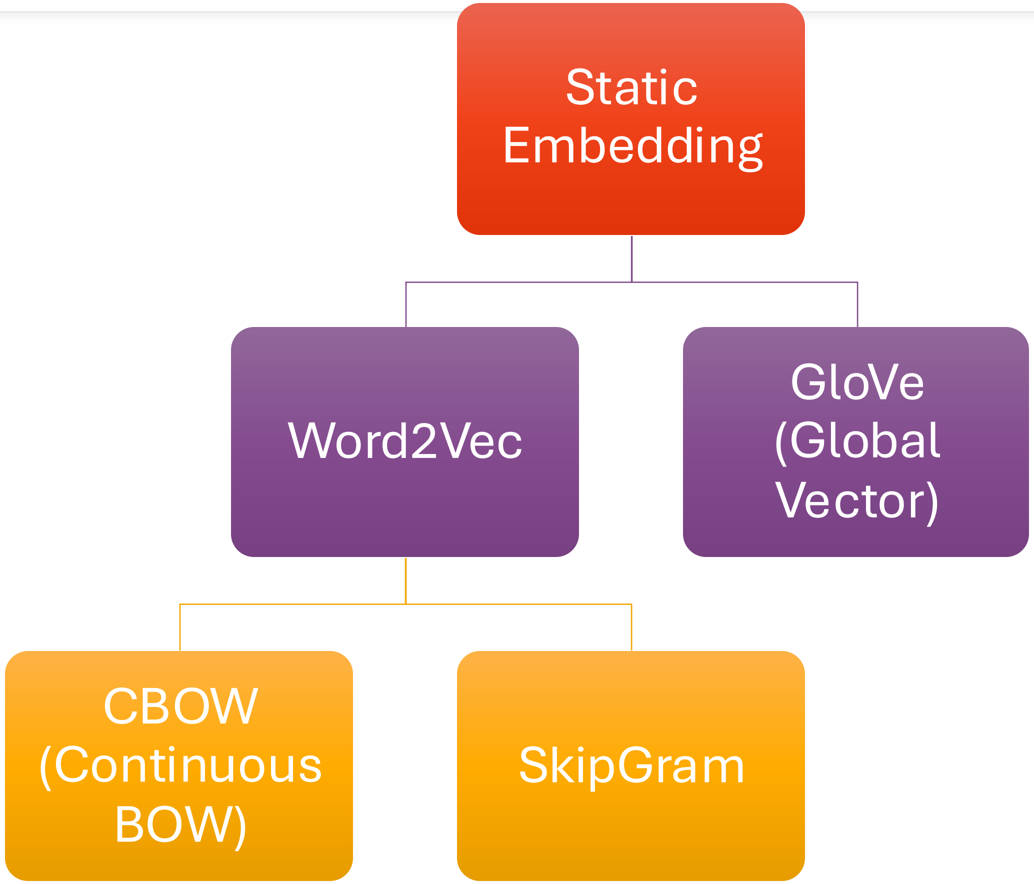


Text Representation in NLP (Revisit)



- Text Representation in NLP
  - Statistical Encoding
    - Discrete Space
      - One Hot Encoding
      - Bag of Words
      - Category based Embeddings
      - N-grams
      - Term Frequency
      - TF-IDF
      - Hyperspace Analogue to Language
    - DRT
      - Feature Selection
        - Document Frequency
        - Term Frequency Variance
        - Mean TF-IDF
        - Information Gain
      - Feature Transformation
        - Latent Semantic Indexing
        - Probabilistic Latent Semantic Indexing
        - Latent Dirichlet Allocation
        - Independent Component Analysis
        - Random Indexing
    - Conceptual Embedding
    - Density Distribution
  - Rule (Grammar) based
  - Neural network (Continuous Space)
    - Feature based
      - Static Embeddings
      - Dynamic Embeddings
    - Fine-tuning based Embeddings
      - Cross Lingual
      - Knowledge Enriched
      - Domain Specific
      - Multi Modal
      - Language Specific

Static Embedding



Static Embedding

Word2Vec

GloVe (Global Vector)

CBOW (Continuous BOW)

SkipGram

Word2Vec

- Feed Forward Neural Network to generate word embedding
  - Two-layer network
    - No hidden layer.
    - Simple architecture, so computational complexity is lower.
    - SoftMax activation function (cross entropy loss function)
- There are two steps involved during learning process:
  - Learning word vectors
  - N-gram training using above distributed representation of words.

Word2Vec – Typical Process

- To generate the embeddings with 1000 dimensions:
  - First, model was trained on 6 billion words from Google News dataset.
  - Then tested for accuracy and quality using:
    - Word Analogy (consisting of semantic and syntactic subtasks)
    - Word Similarity
    - Sentence completion tasks

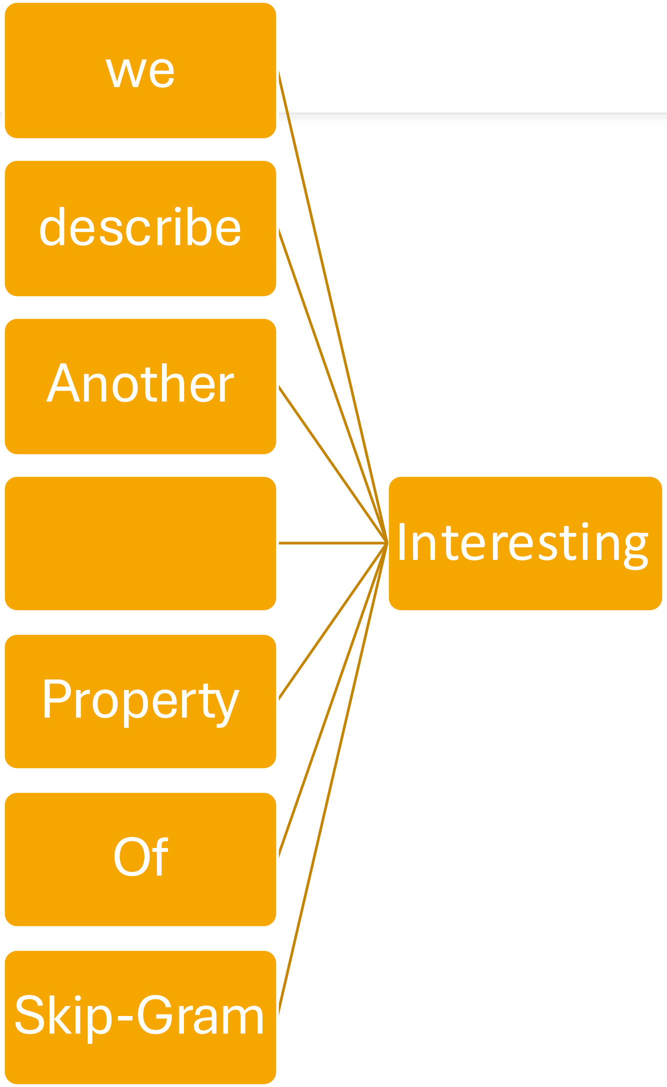
Word2Vec – Semantic Relation

- The kid said he would grow up to be Superman.
- The child said he would grow up to be Superman.

- So, two words kid and child are similar, and should have similar word vector.

CBOW v/s Skip-Gram

Predicts target word from the context words.



we

describe

Another

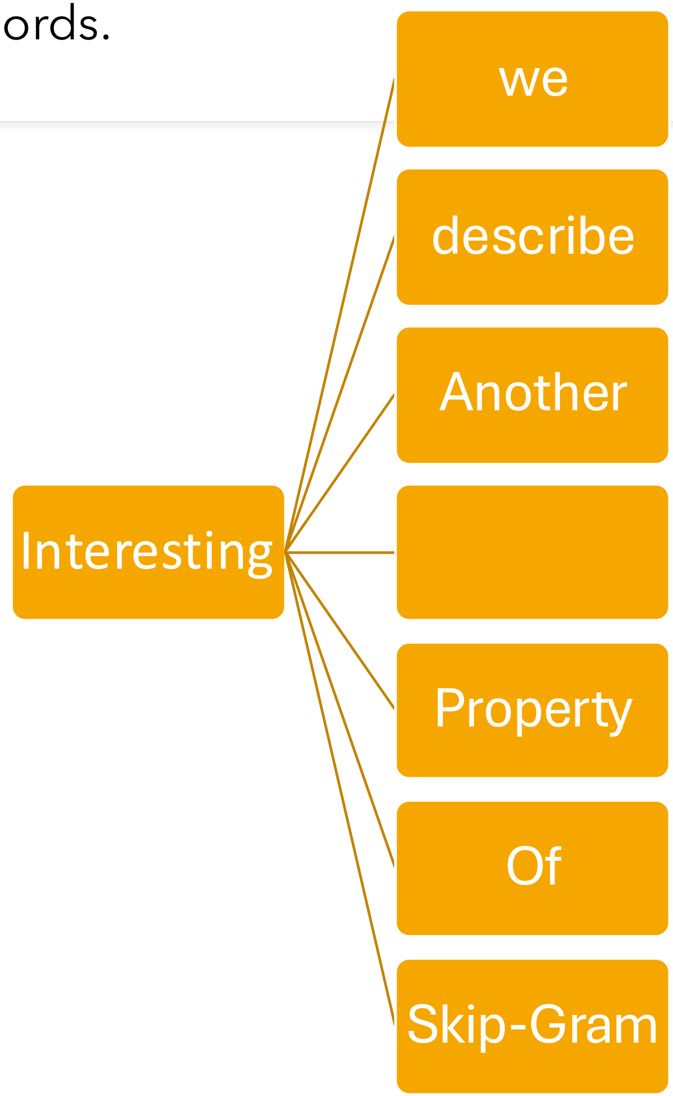
Property

Of

Skip-Gram

Interesting

Predicts the context words from target words.



Interesting

we

describe

Another

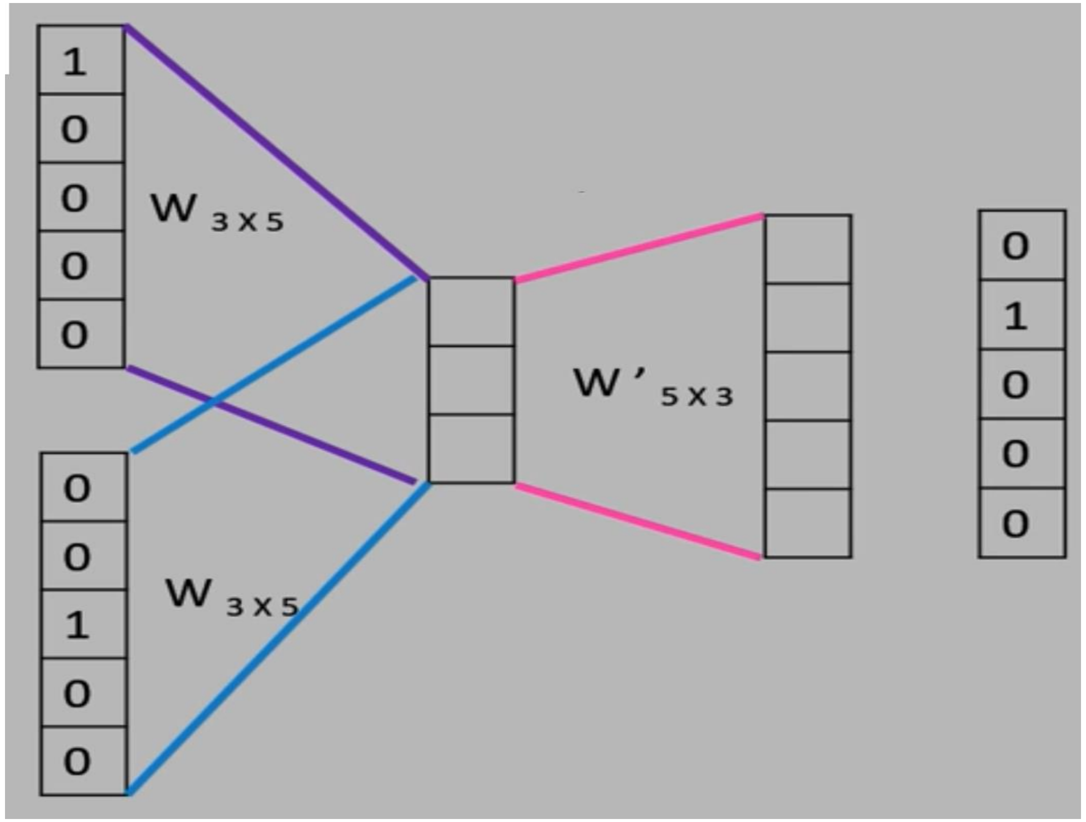
Property

Of

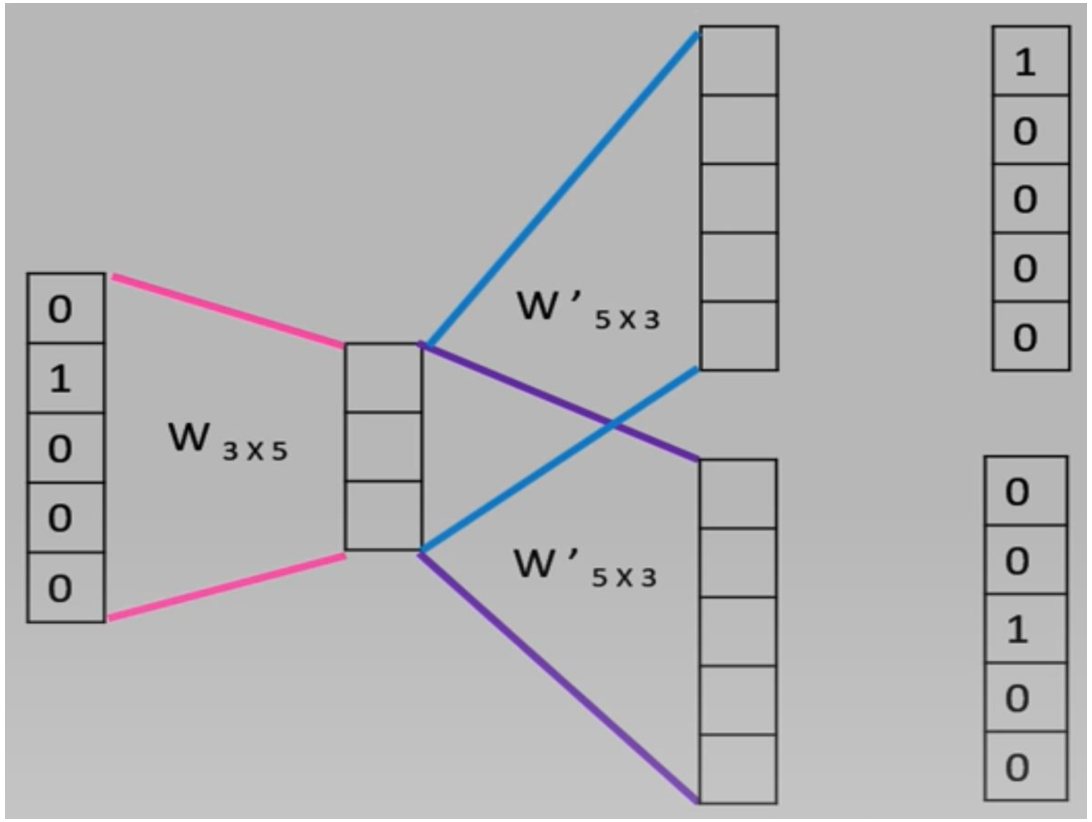
Skip-Gram

Word2Vec model training

CBOW FFNN



Skip-Gram FFNN



Challenges in Skip gram:

- Quality of the vectors
- Training speed
- Indifference to word order
- Inability to represent idiomatic phrases.

Extensions of Skip-Gram Model

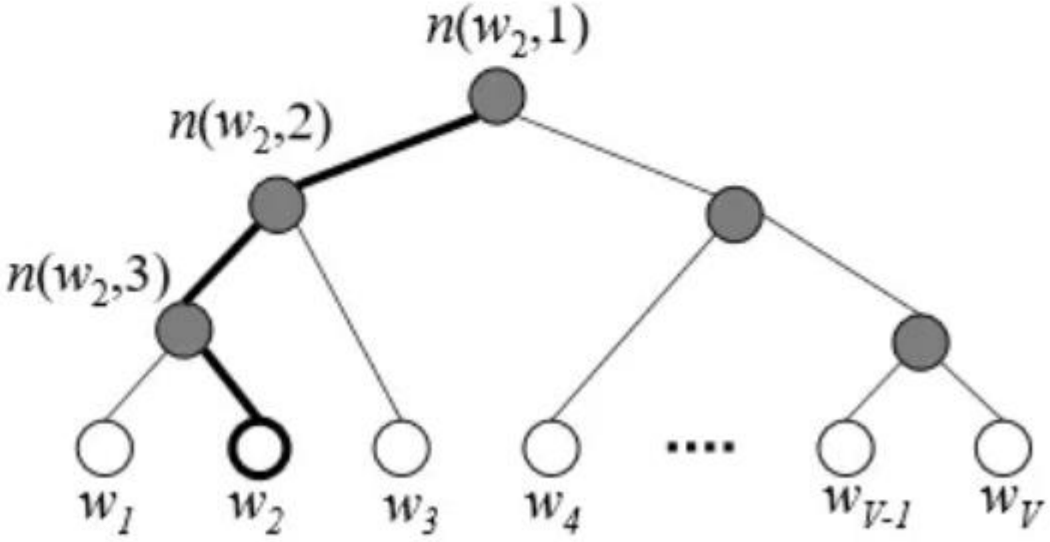
- Hierarchical SoftMax
- Negative Sampling
- Subsampling of Frequent Words
- Learning Phrases

Hierarchical Softmax

- Computationally efficient approximation of the full softmax is the hierarchical softmax.
- Uses a binary tree representation of the output layer with the W words as its leaves and, for each node, explicitly represents the relative probabilities of its child nodes.
- Instead of evaluating W output nodes in the neural network to obtain the probability distribution, it is needed to evaluate only about $\log_2(W)$ nodes.

Hierarchical Softmax

If the tree has the root node, 2 inner nodes, and leaf nodes, it is obvious that we are performing 3 steps of computations, which is a sufficient decrease in the number of operations we're doing.



Negative Sampling

I want a glass of **orange juice** to go along with my cereal.

Context word	Target word	Target Label
Orange	Juice	1
Orange	King	0
Orange	Book	0
Orange	the	0
Orange	of	0

Negative Sampling

- Distinguishes target word W_o from draws from the noise distribution $P_n(w)$ using logistic regression, with k negative samples for each data sample.
- k in the range 5–20 are useful for small training datasets, while for large datasets the k can be as small as 2–5.
- Unigram distribution $U(w)^{3/4}/Z$ outperformed significantly as the noise distribution $P_n(w)$

Subsampling of Frequent Words

- Vector representations of frequent words do not change significantly after training on several million examples.
- To counter the imbalance between the rare and frequent words, each word in the training set is discarded with probability computed by

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

where $f(w_i)$ is the frequency of word w_i and t is a chosen threshold, typically around 10−5.

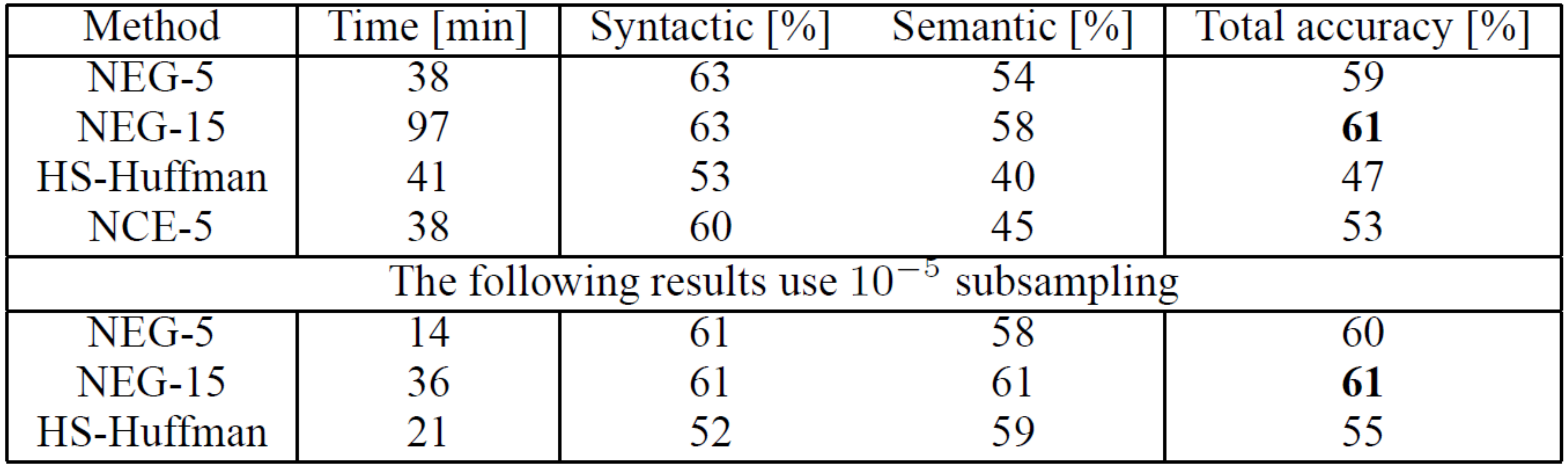
Subsampling of Frequent Words

- Aggressively subsamples words whose frequency is greater than t while preserving the ranking of the frequencies.

- Advantages:
- Accelerates learning
- Improves accuracy of the representations of less frequent words.

Empirical Results

Accuracy of various Skip-gram 300-dimensional models on the analogical reasoning task

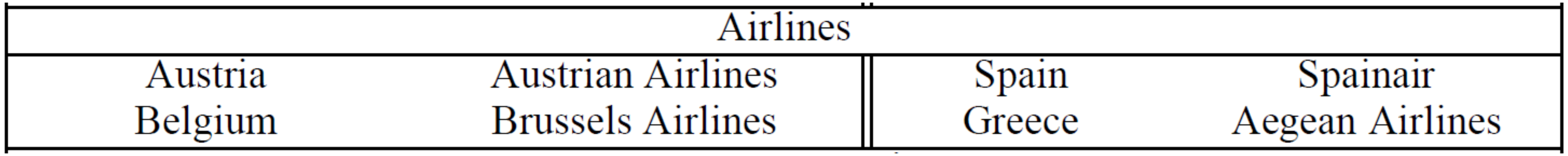


Method	Time [min]	Syntactic [%]	Semantic [%]	Total accuracy [%]
NEG-5	38	63	54	59
NEG-15	97	63	58	**61**
HS-Huffman	41	53	40	47
NCE-5	38	60	45	53
The following results use 10^{-5} subsampling				
NEG-5	14	61	58	60
NEG-15	36	61	61	**61**
HS-Huffman	21	52	59	55

- The table shows that Negative Sampling outperforms the Hierarchical Softmax on the analogical reasoning task with slightly better performance.
- The subsampling of the frequent words improves the training speed several times and makes the word representations significantly more accurate.

Learning Phrases

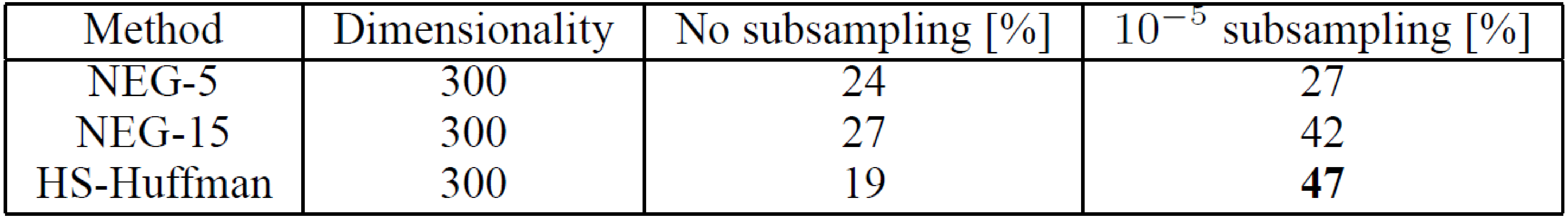
- Many phrases have a meaning that is not a simple composition of the meanings of its individual words.
- The bigrams with score above the chosen threshold are used as phrases.
- 2-4 passes over the training data with decreasing threshold value are run, allowing longer phrases that consists of several words to be formed.



Airlines			
Austria	Austrian Airlines	Spain	Spainair
Belgium	Brussels Airlines	Greece	Aegean Airlines

Example of the analogical reasoning task for phrases (the full test set has 3218 examples).
The goal is to compute the fourth phrase using the first three. Best model achieved an accuracy of 72%.

Phrase Skip-Gram Results



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

Accuracies of the Skip-gram models on the phrase analogy dataset. The models were trained on approximately one billion words from the news dataset.

Inferences:

- Subsampling of the frequent words results in both faster training and significantly better representations of uncommon words.
- Negative sampling algorithm, an extremely simple training method, learns accurate representations especially for frequent words.
- Learning representations of phrases is a powerful yet simple way to represent longer pieces of text, while having minimal computational complexity.

What is GloVe?

- GloVe (Global Vectors for Word Representation) is a word embedding model that learns word vectors from co-occurrence statistics of words in a corpus.
- Combines advantages of both **global matrix factorization** (like LSA) and **local context-based learning** (like Word2Vec).

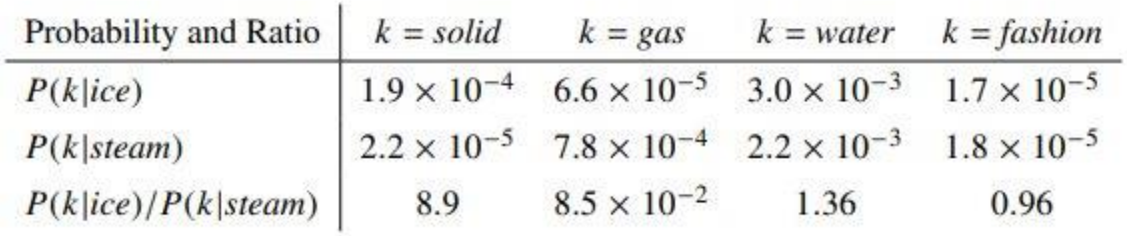
Challenge:

- How can we effectively generate meaningful word representations from co-occurrence statistics?
- GloVe addresses the question of how meaning is embedded in these statistics by using co-occurrence probabilities.

Co-occurrence Matrix

Definition:

- Let **X** be the co-occurrence matrix wl X_{ij} is the number of times word *j* occurs in the context of word *i*.
- **Example**: Table 1 from the paper shows how the co-occurrence probabilities between words such as “ice” and “steam” differ for context words like "solid" and "gas."



Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Co-occurrence Ratios

Key Insight:

- GloVe captures the meaning through the **ratios** of co-occurrence probabilities.
- Words related to a particular concept (e.g., ice vs. steam) show distinct ratios when compared with related context words (e.g., "solid", "gas").
- The ratio helps distinguish between relevant and irrelevant contexts more effectively than raw probabilities.

The GloVe Model

Modeling Word Relationships:

•The model’s core idea is based on the equation:

$$w_i^T \tilde{w}_k + b_i + b_k = \log(X_{ik})$$

where w_i and $\tilde{w}_k$ are word vectors, and X_{ik} is the co-occurrence count.

•The goal is to optimize the vectors so that their dot products approximate the log of the co-occurrence counts.

Weighted Least Squares Regression

Objective Function:

•GloVe employs a **weighted least squares** regression approach, addressing problems from rare and frequent word pairs.

•The function:

$$J = \sum_{i,j} f(X_{ij}) \left(w_i^T \tilde{w}_j + b_i + b_j - \log(X_{ij})\right)^2$$

- Where $f(X_{ij})$ is a weighting function designed to prevent rare co-occurrences from dominating the learning process.

Weighting Function

Importance of Weighting:

•Rare co-occurrences should not be overweighted, and common co-occurrences should not be ignored.

•The weighting function $f(X_{ij})$ is :

$$f(x) = \begin{cases} (x/x_{\max})^{\alpha} & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

•Empirically, α=3/4 worked best, which echoes the fractional scaling used in Word2Vec models.

Performance and Use

Applications:

- GloVe has been applied to various NLP tasks, such as word analogy, named entity recognition, and machine translation.
- It outperforms traditional methods in capturing semantic relationships between words.

Conclusion

Why GloVe?

- GloVe's focus on **global corpus statistics** and its use of **co-occurrence ratios** make it a powerful alternative to models like Word2Vec.
- Efficient, scalable, and effective in representing complex word relationships in vector space.

Code Example

Thank you

References:

Distributed Representations of Words and Phrases and their Compositionality

"GloVe: Global Vectors for Word Representation" by Pennington et al.