Special Topics in Natural Language Processing

CS698O

Ashutosh Modi
CSE Department, IIT Kanpur

Lecture 9: Language Models Finale
Jan 23, 2020

Representations

- One of the key questions in ML/NLP is what is the best way to represent concepts (entity, abstract concept, etc.): **Representation Problem**

Representations

- One of the key questions in ML/NLP is what is the best way to represent concepts (entity, abstract concept, etc.): **Representation Problem**

- Two different views (inspiration from Cognitive Psychology) :

  - *Local Representation*: Each concept has a dedicated atomic unit. Connection between units results in interactions between concepts.

  - *Distributed Representation*: Each concept is represented by a pattern of activity distributed over several units (e.g. Neurons).

Local Representations

- *Local Representation*

$$\begin{aligned}
\text{Car:} \quad & [0,0,0,1,0,0,0,0,0,0,0] \\
\text{Bus:} \quad & [0,0,0,0,0,0,0,1,0,0,0] \\
\text{Automobile:} \quad & [0,0,0,0,0,0,0,0,0,1,0] \\
\text{Human:} \quad & [0,1,0,0,0,0,0,0,0,0,0]
\end{aligned}$$

Local Representations

- *Local Representation*

$$\begin{aligned}
\text{Car:} \quad & [0,0,0,1,0,0,0,0,0,0,0] \\
\text{Bus:} \quad & [0,0,0,0,0,0,0,1,0,0,0] \\
\text{Automobile:} \quad & [0,0,0,0,0,0,0,0,0,1,0] \\
\text{Human:} \quad & [0,1,0,0,0,0,0,0,0,0,0]
\end{aligned}$$

How many concepts can you represent with N units?

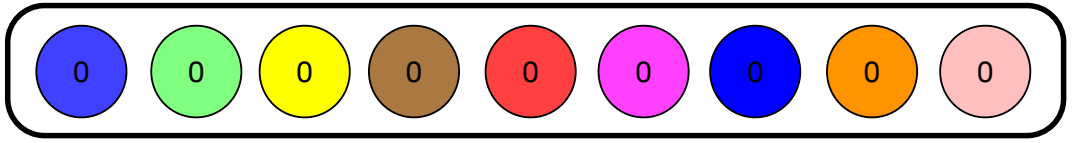
Local Representations

- *Local Representation*: Not efficient, Not easily generalizable

$$\begin{aligned}
\text{Car:} \quad & [0,0,0,1,0,0,0,0,0,0,0] \\
\text{Bus:} \quad & [0,0,0,0,0,0,0,1,0,0,0] \\
\text{Automobile:} \quad & [0,0,0,0,0,0,0,0,0,1,0] \\
\text{Human:} \quad & [0,1,0,0,0,0,0,0,0,0,0]
\end{aligned}$$

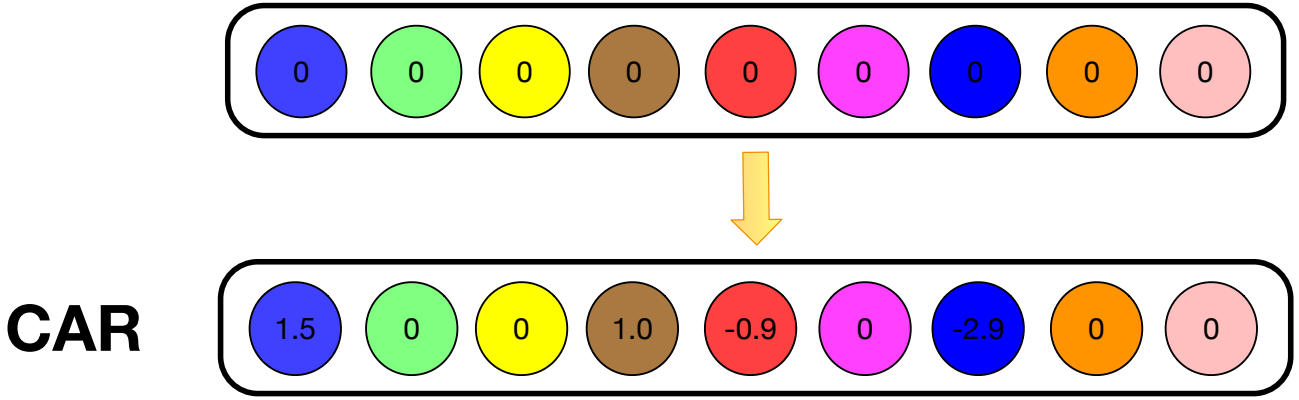
Distributed Representations

- *Distributed Representation*



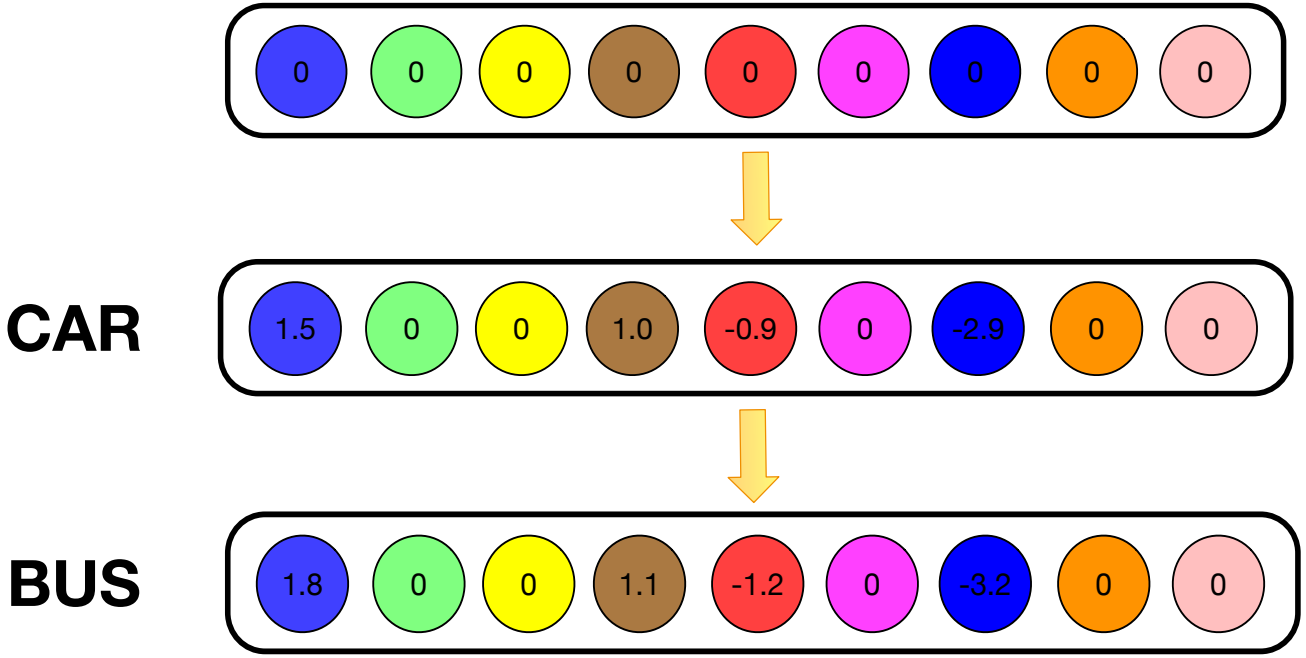
Distributed Representations

- *Distributed Representation*



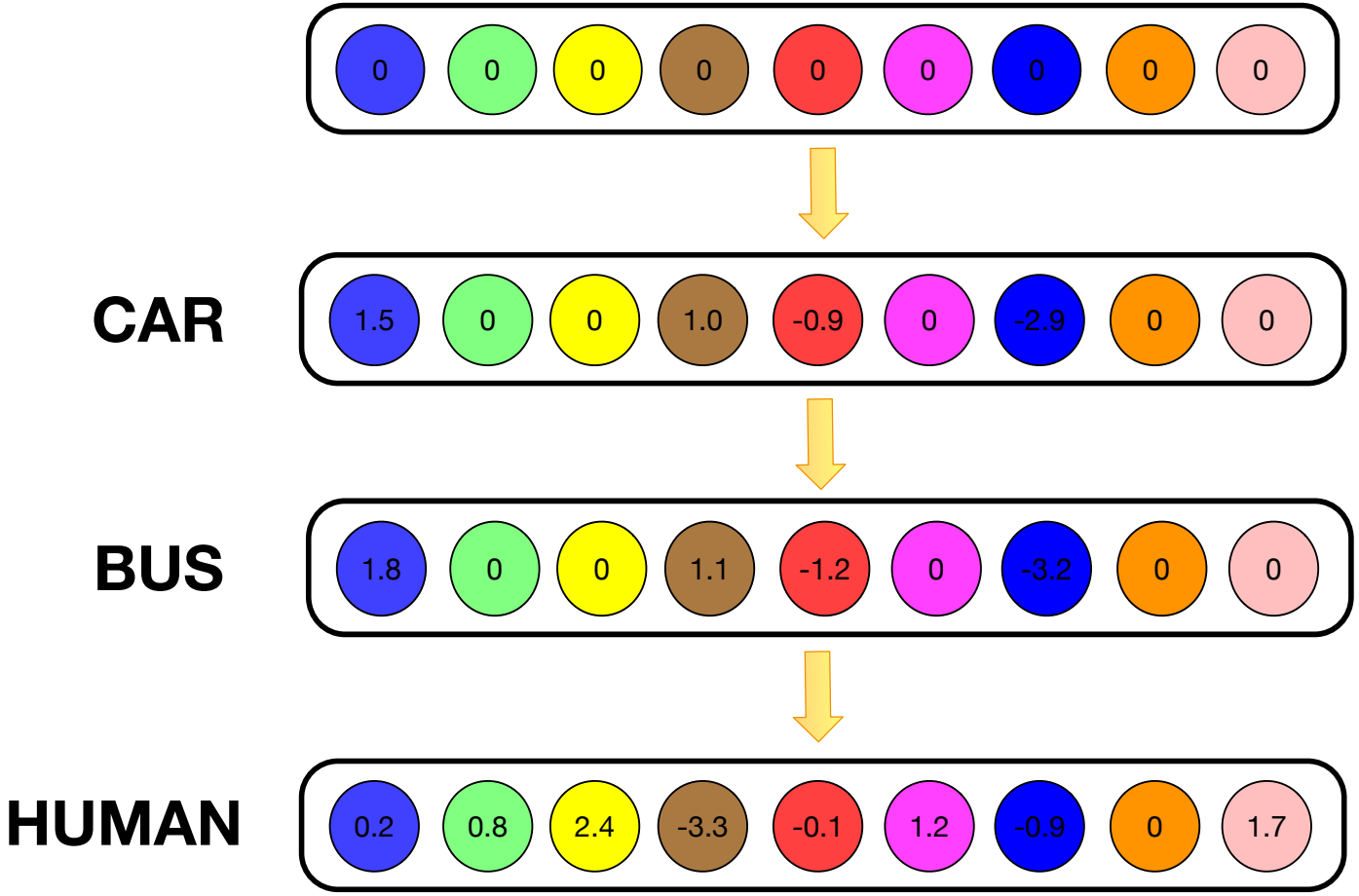
Distributed Representations

- *Distributed Representation*



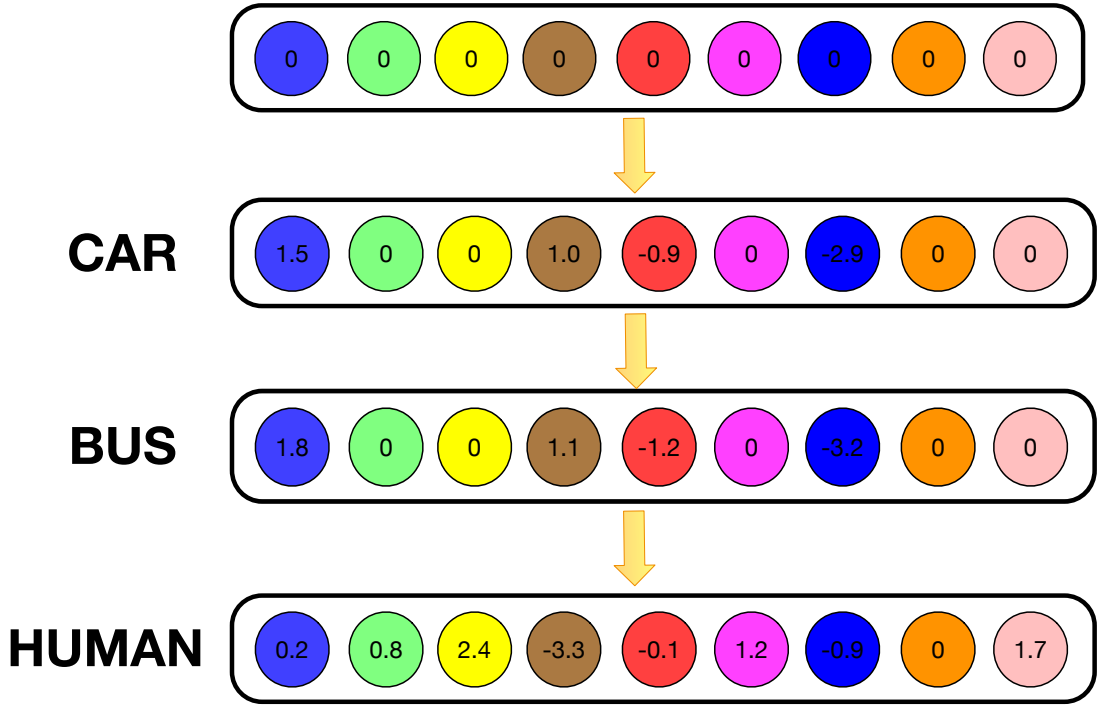
Distributed Representations

- *Distributed Representation*



Distributed Representations

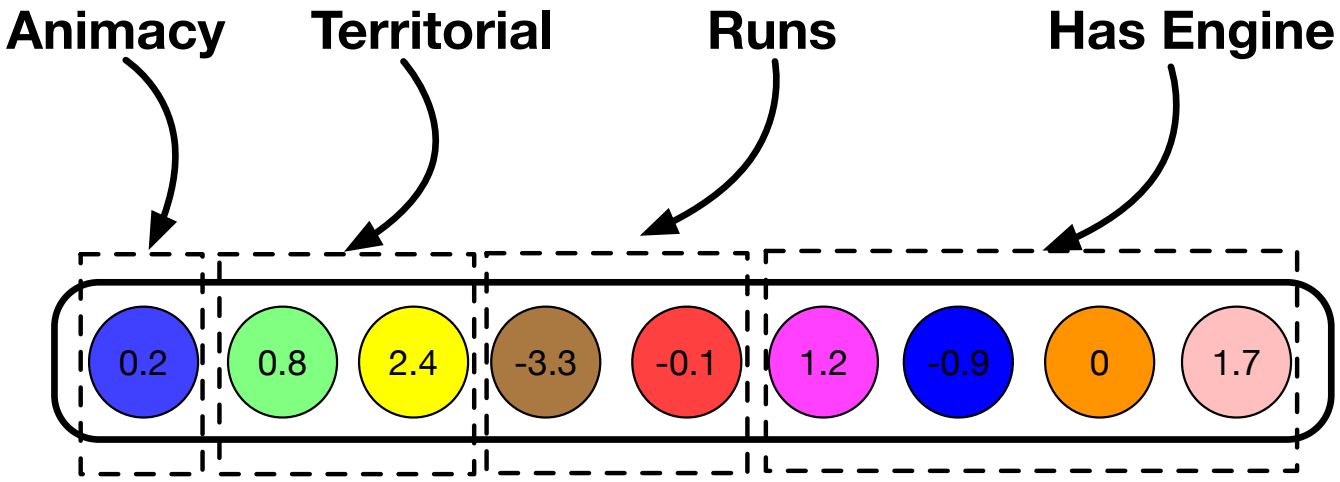
- *Distributed Representation*



How many concepts can you represent with N units?

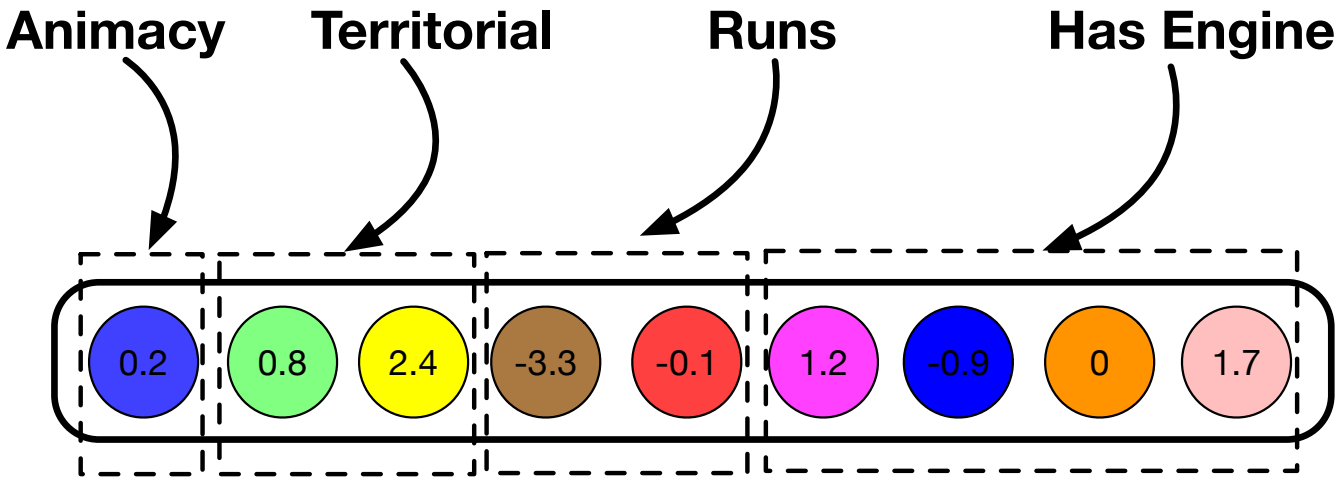
Distributed Representations

- Efficient



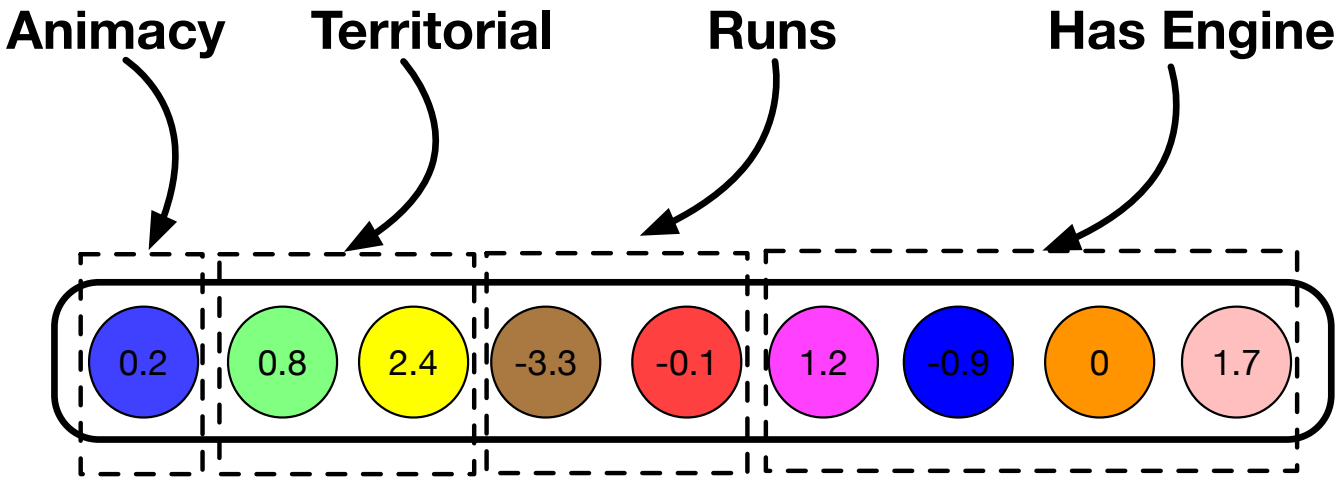
Distributed Representations

- Efficient
- Generalizable

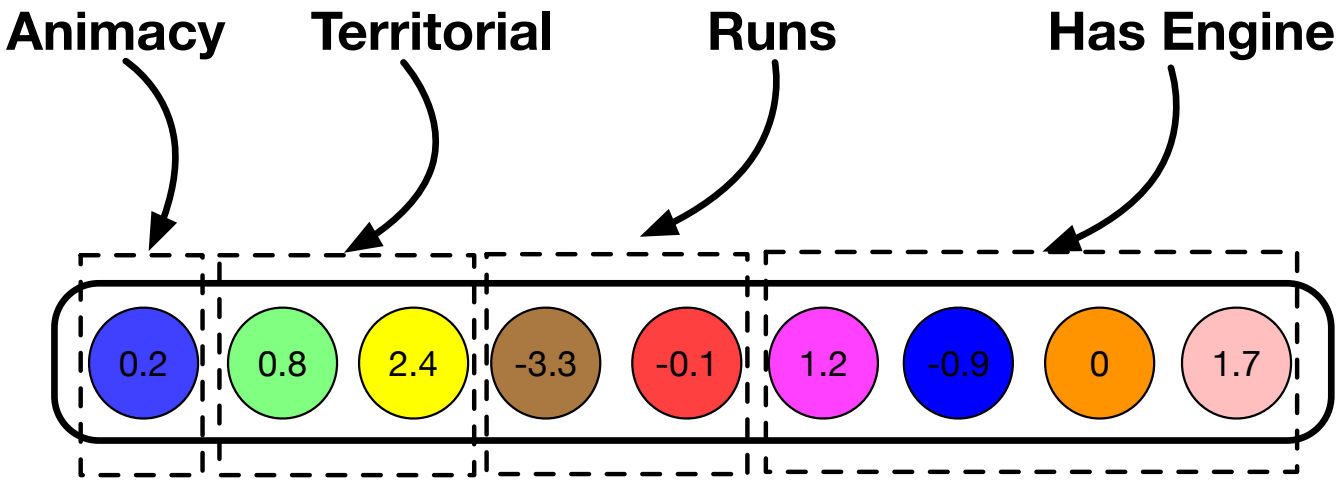


Distributed Representations

- Efficient
- Generalizable
- Concept Parallelism

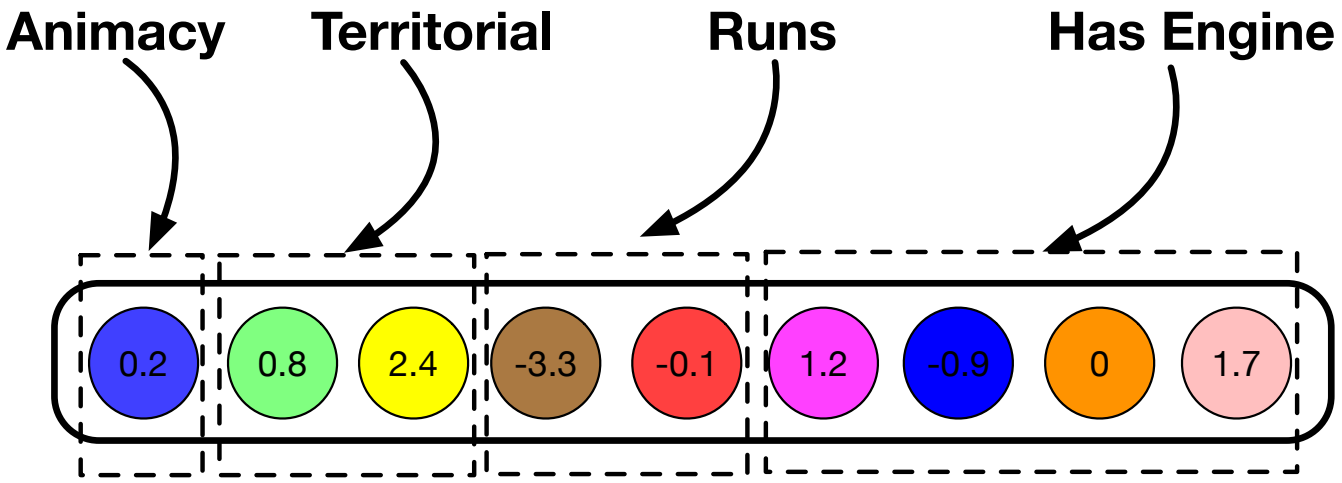


Distributed Representations



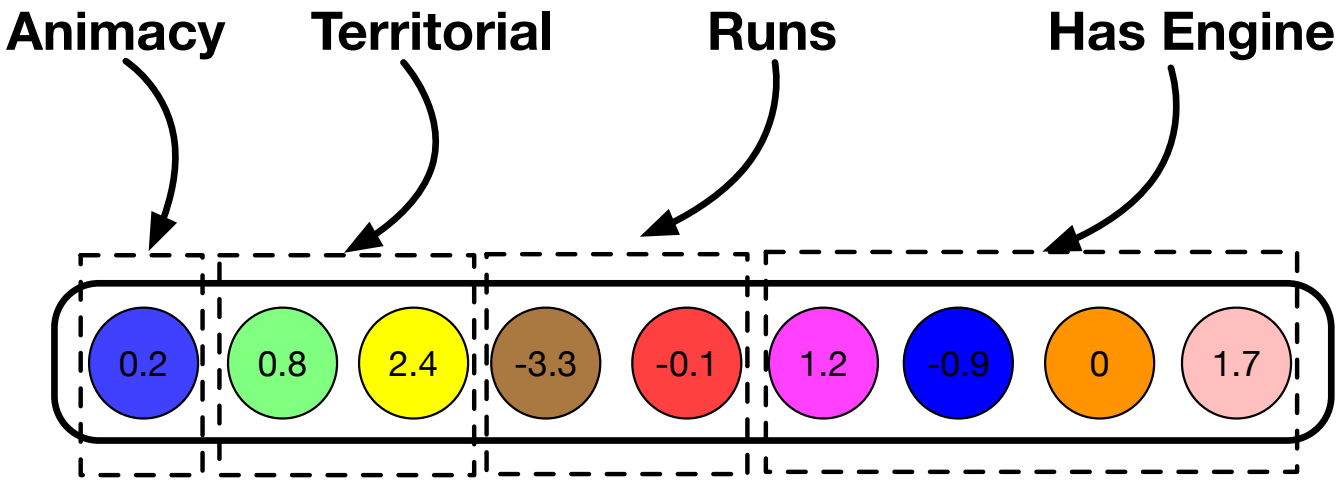
- Efficient
- Generalizable
- Concept Parallelism
- ***All for one, one for all***: Each unit contributes to a concept and each concept is distributed over all units

Distributed Representations



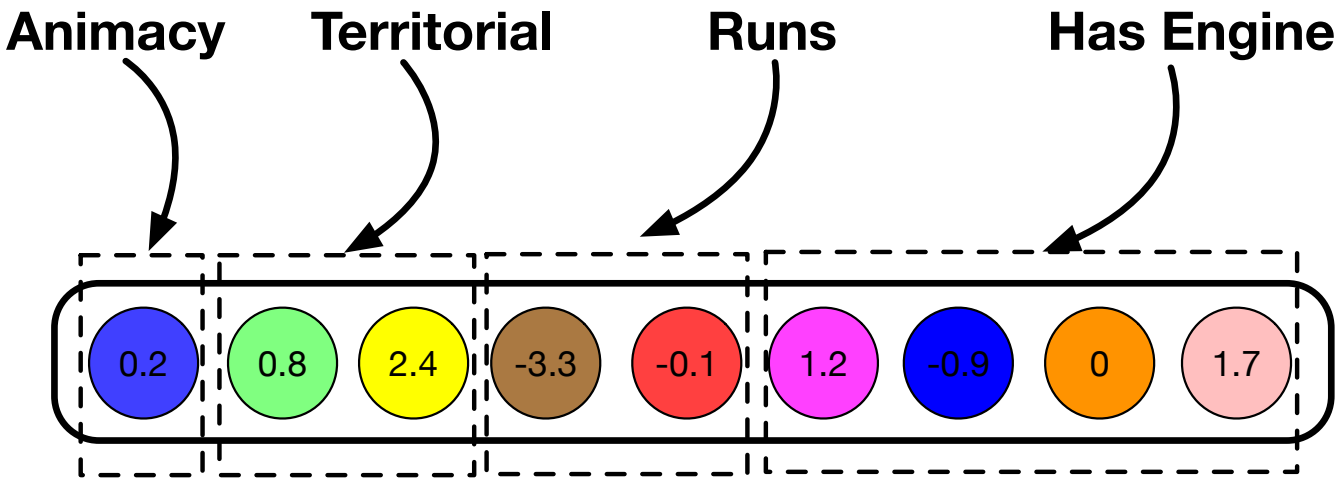
- Efficient
- Generalizable
- Concept Parallelism
- ***All for one, one for all***: Each unit contributes to a concept and each concept is distributed over all units
- Each unit (or group of units) represents a (contextual) feature which may be common across many concepts

Distributed Representations



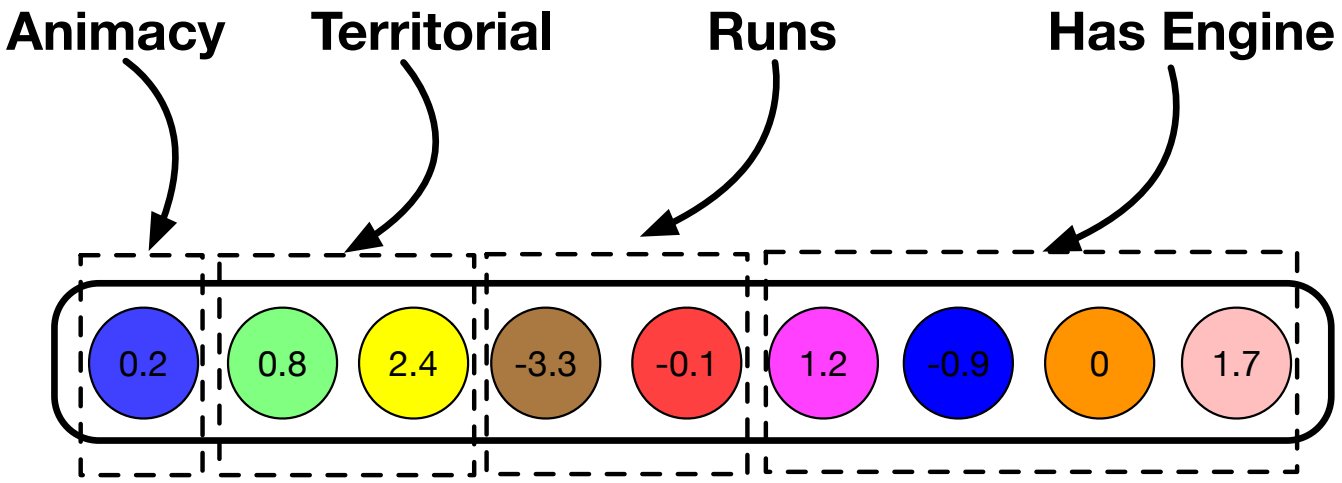
- Efficient
- Generalizable
- Concept Parallelism
- ***All for one, one for all***: Each unit contributes to a concept and each concept is distributed over all units
- Each unit (or group of units) represents a (contextual) feature which may be common across many concepts
- **Holographic**: Partial part of the representation can be used to reconstruct the whole

Distributed Representations



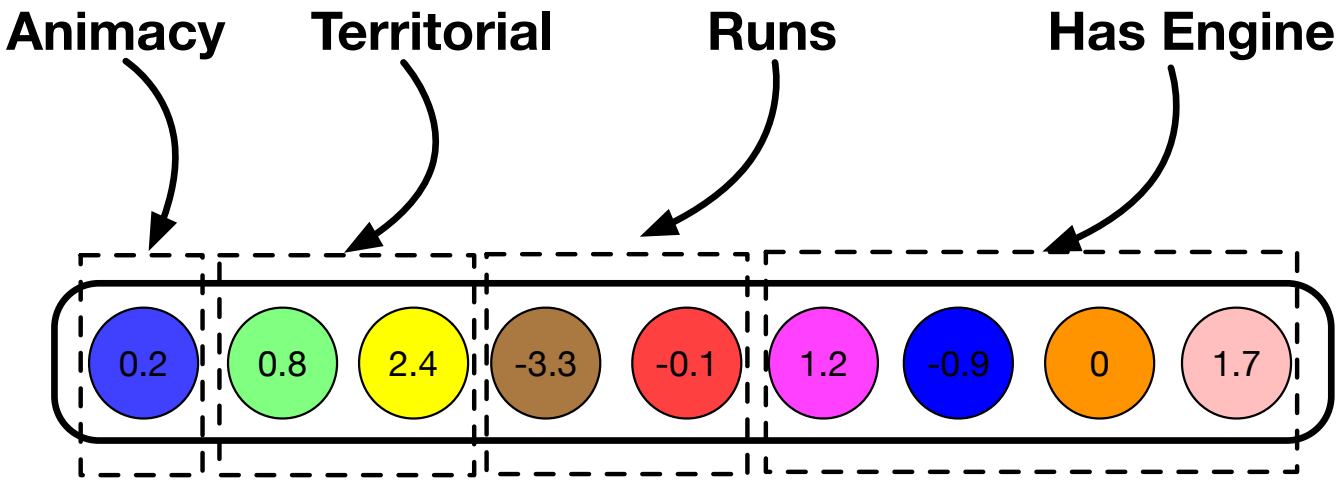
- It is easy to get representation for new concepts. This is achieved by modifying interactions between units.

Distributed Representations



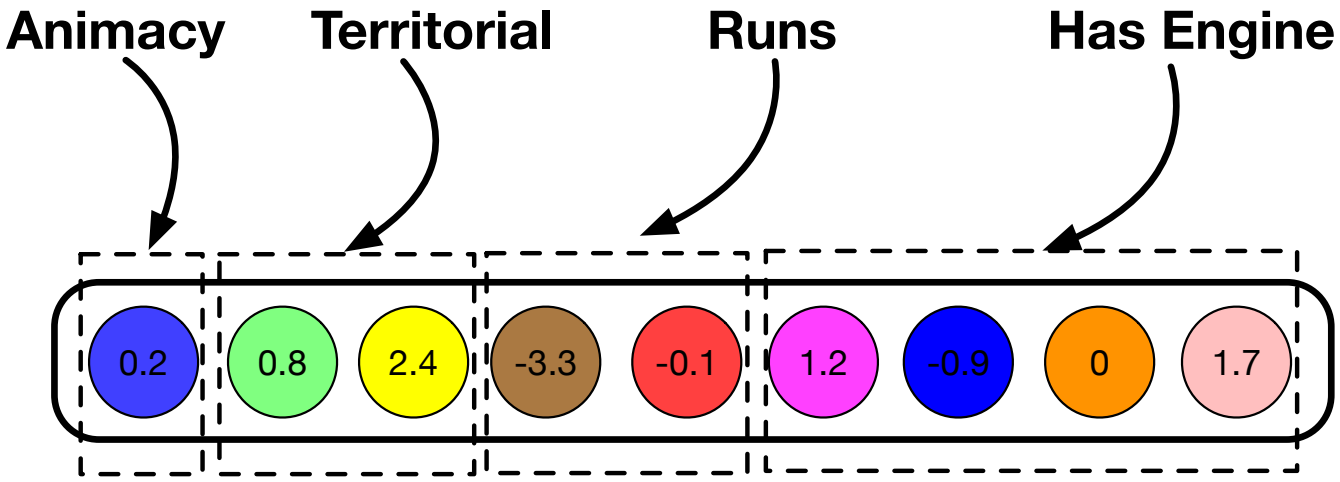
- It is easy to get representation for new concepts. This is achieved by modifying interactions between units.
- Pattern for new concept changes some (or all) units slightly.

Distributed Representations



- It is easy to get representation for new concepts. This is achieved by modifying interactions between units.
- Pattern for new concept changes some (or all) units slightly.
- There is a delicate balance between units. New concept would not randomly change the units as this will disrupt the existing knowledge.

Distributed Representations



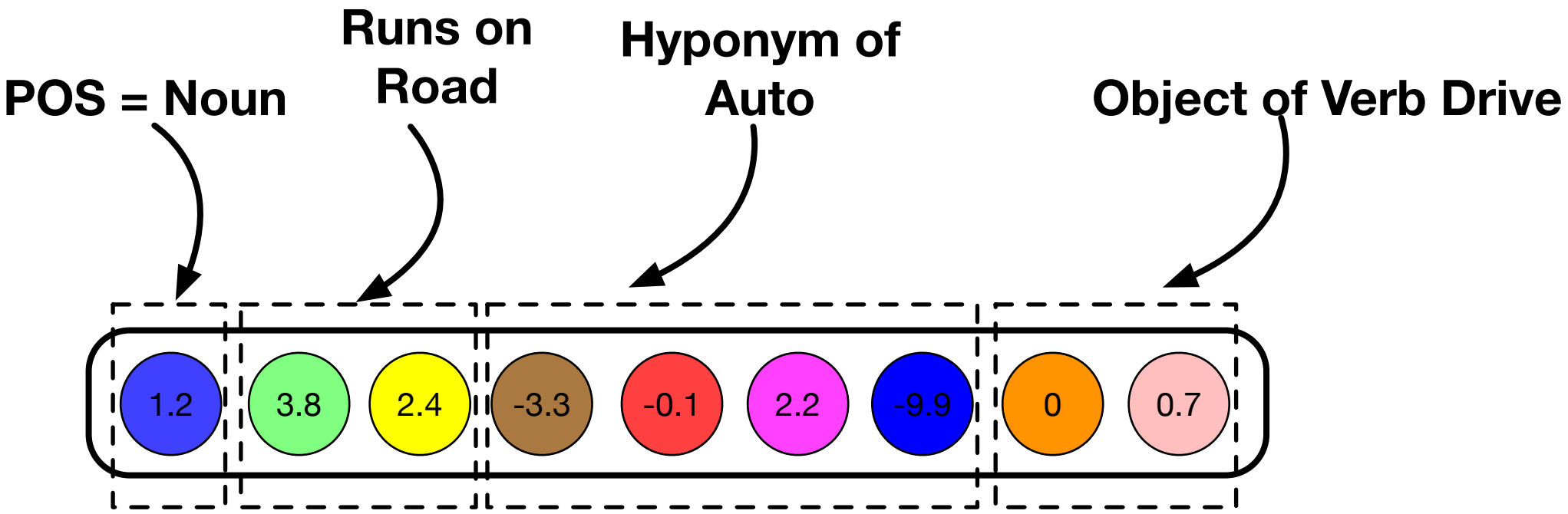
- We loose on interpretability

Distributed Representations for Words: Embeddings

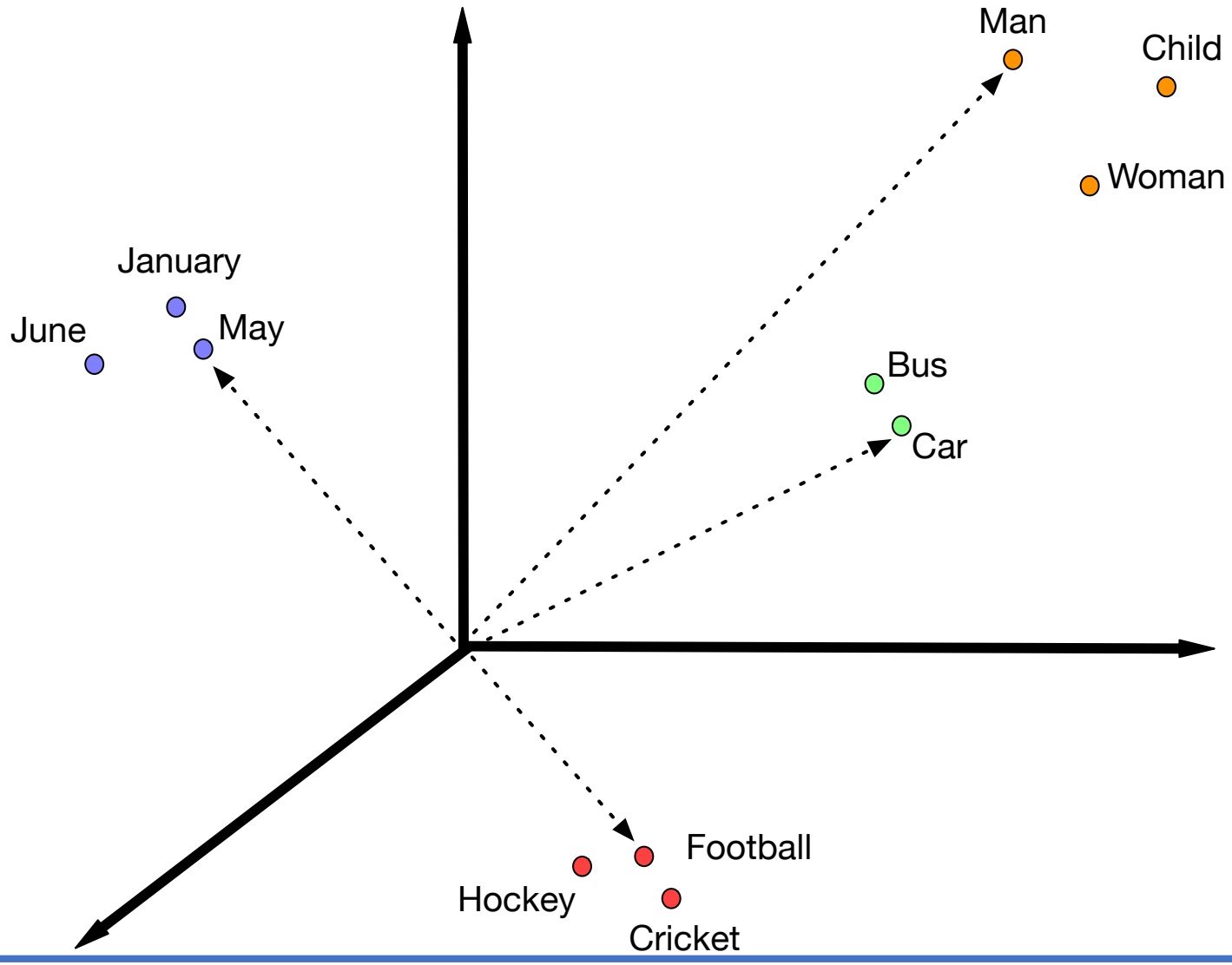
- **Captures the semantic and syntactic properties of the word**

Distributed Representations for Words: Embeddings

- **Captures the semantic and syntactic properties of the word**



Words Embeddings: Vectors of Meaning



Neural Network Language Model (NNLM)

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, w_{i-1}, \cdots, w_{i-n+1})$$

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

$$f(i, w_{i-1}, \cdots, w_{i-n+1}) = g(i, C(w_{i-1}), \cdots, C(w_{i-n+1}))$$

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

$$f(i, w_{i-1}, \cdots, w_{i-n+1}) = g(i, C(w_{i-1}), \cdots, C(w_{i-n+1}))$$

Feed-Forward Neural Network

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

$$f(i, w_{i-1}, \cdots, w_{i-n+1}) = g(i, C(w_{i-1}), \cdots, C(w_{i-n+1}))$$

Feed-Forward Neural Network

Index of (i-1)th word

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

$$f(i, w_{i-1}, \cdots, w_{i-n+1}) = g(i, C(w_{i-1}), \cdots, \underbrace{C(w_{i-n+1})}_{\text{Distributed Representation of } (i\text{-}n+1)^{th} \text{ word}})$$

Feed-Forward Neural Network

Index of $(i\text{-}1)^{th}$ word

Neural Network Language Model (NNLM)

$$p(w_1, w_2, \cdots, w_N) = p(w_1^N) \approx \prod_{i=1}^{n} p(w_i \mid w_{i-n+1}^{i-1})$$

$$p(w_i \mid w_{i-n+1}^{i-1}) \approx f(i, \underbrace{w_{i-1}, \cdots, w_{i-n+1}}_{\text{context}})$$

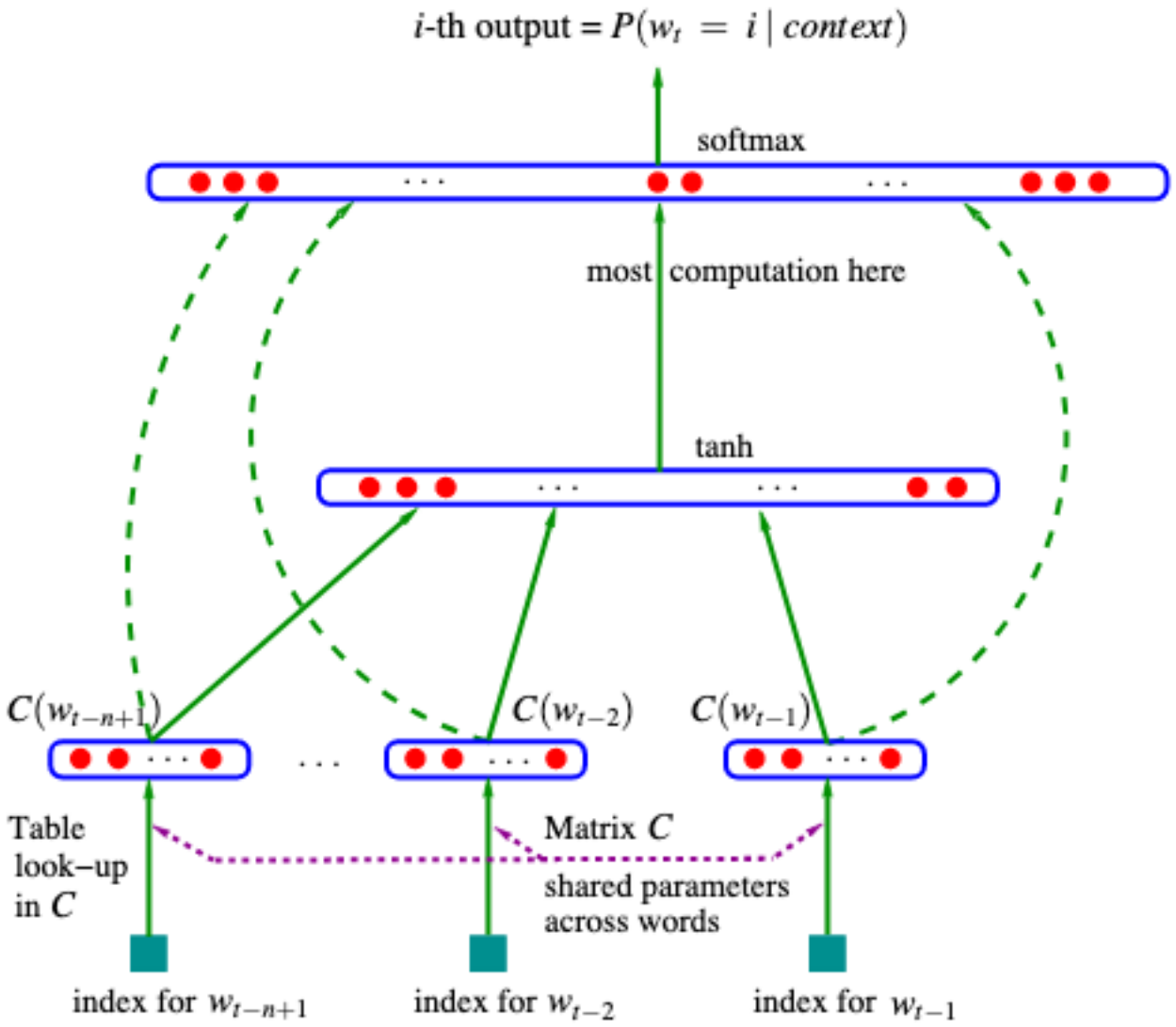
$$f(i, w_{i-1}, \cdots, w_{i-n+1}) = g(i, C(w_{i-1}), \cdots, \underbrace{C(w_{i-n+1})}_{\text{Distributed Representation of } (i\text{-}n+1)^{th} \text{ word}})$$

Feed-Forward Neural Network

Look up Matrix (Embedding Matrix)

Index of $(i-1)^{th}$ word

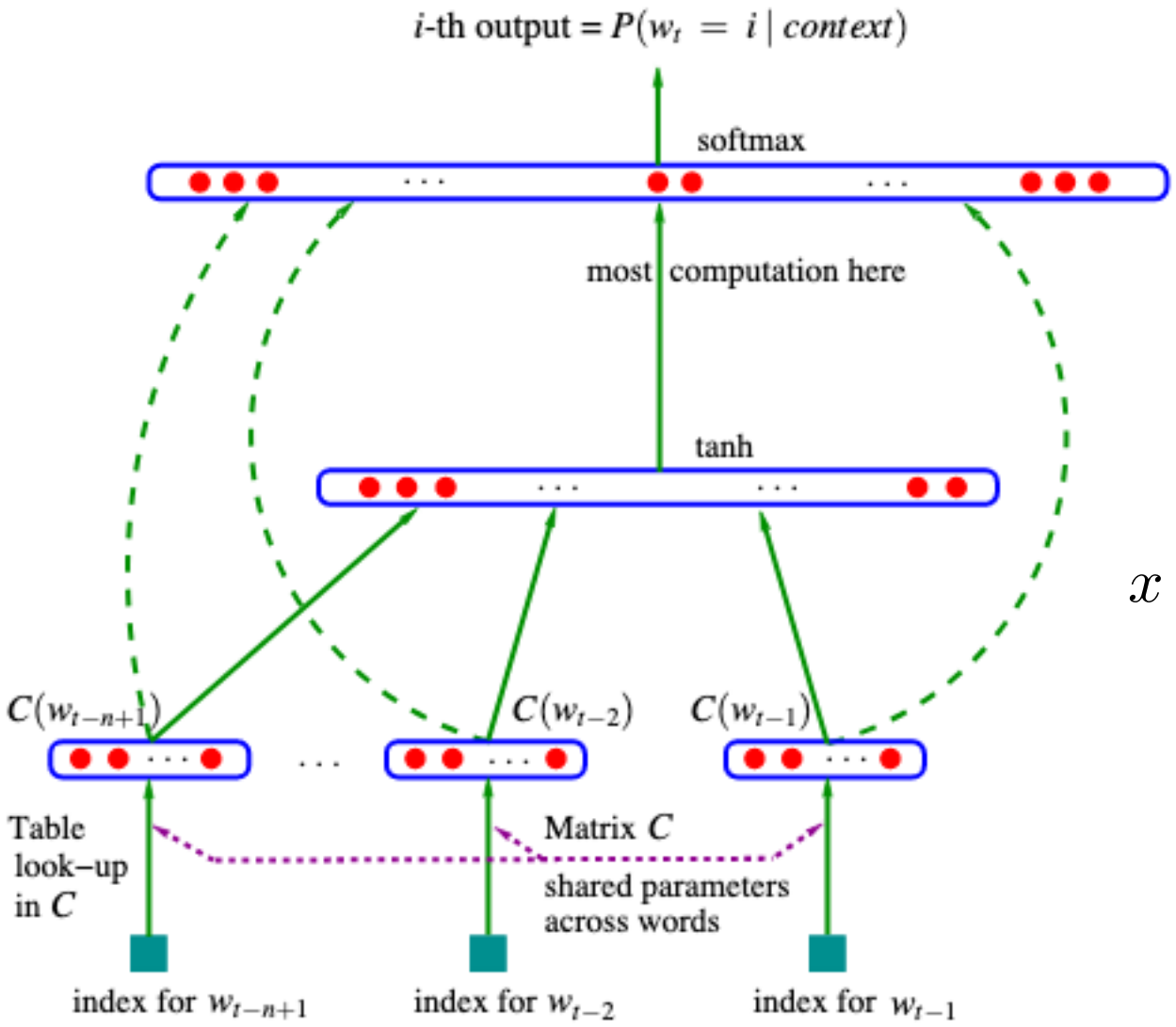
Neural Network Language Model (NNLM)



$$p(w_i \mid w_{i-n+1}^{i-1}) = \frac{exp(y_{w_i})}{\sum_{j=1}^{|V|} exp(y_{w_j})}$$

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Neural Network Language Model (NNLM)



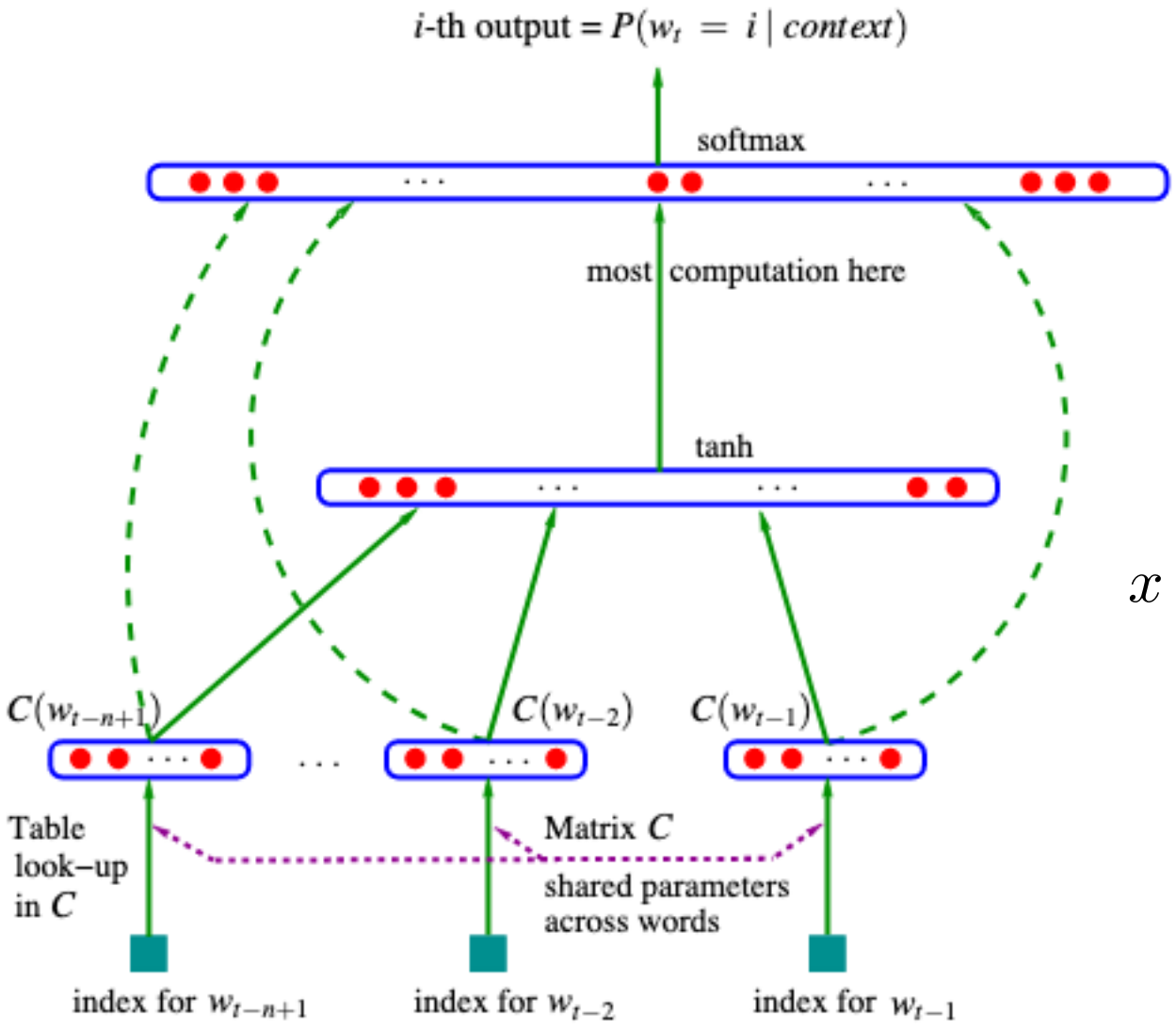
$$p(w_i \mid w_{i-n+1}^{i-1}) = \frac{exp(y_{w_i})}{\sum_{j=1}^{|V|} exp(y_{w_j})}$$

$$y = U\tanh(Hx + d) + Wx + b$$

$$x = (C(w_{i-1}), C(w_{i-2}), \cdots, C(w_{i-n+1}))$$

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Neural Network Language Model (NNLM)



$$p(w_i \mid w_{i-n+1}^{i-1}) = \frac{exp(y_{w_i})}{\sum_{j=1}^{|V|} exp(y_{w_j})}$$

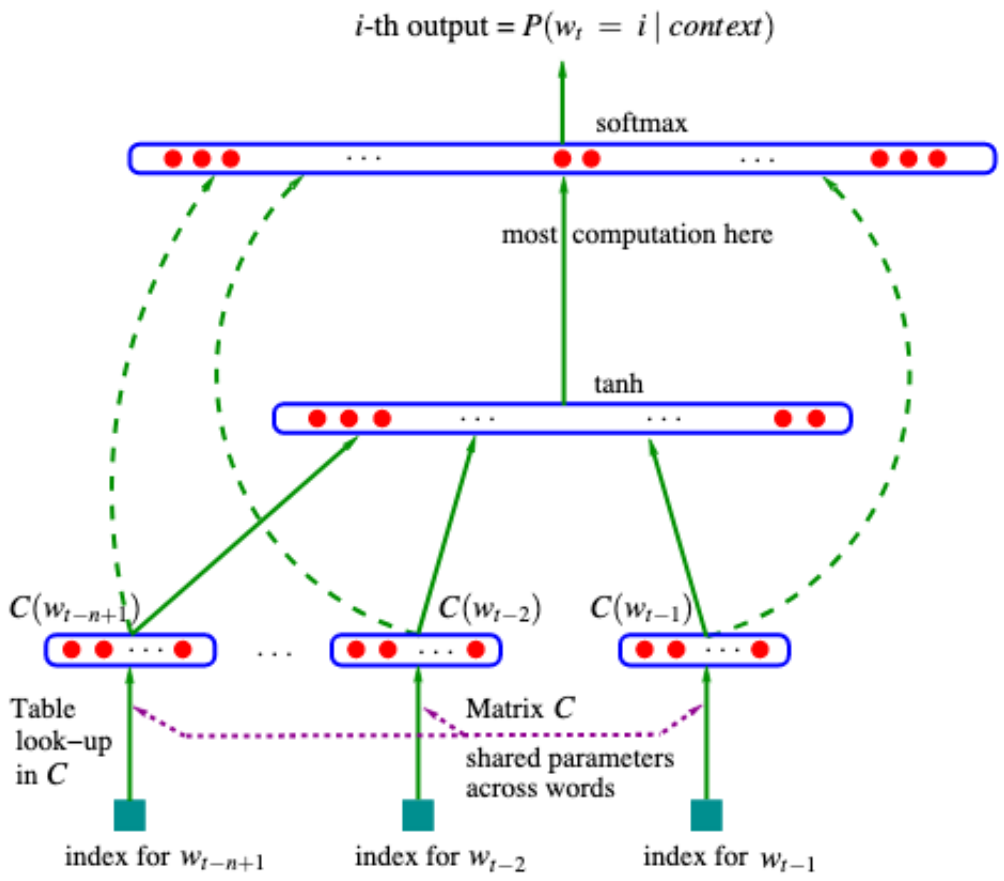
$$y = U\tanh(Hx + d) + Wx + b$$

$$x = (C(w_{i-1}), C(w_{i-2}), \cdots, C(w_{i-n+1}))$$

Parameters learned using Backpropagation and Stochastic Gradient Descent

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

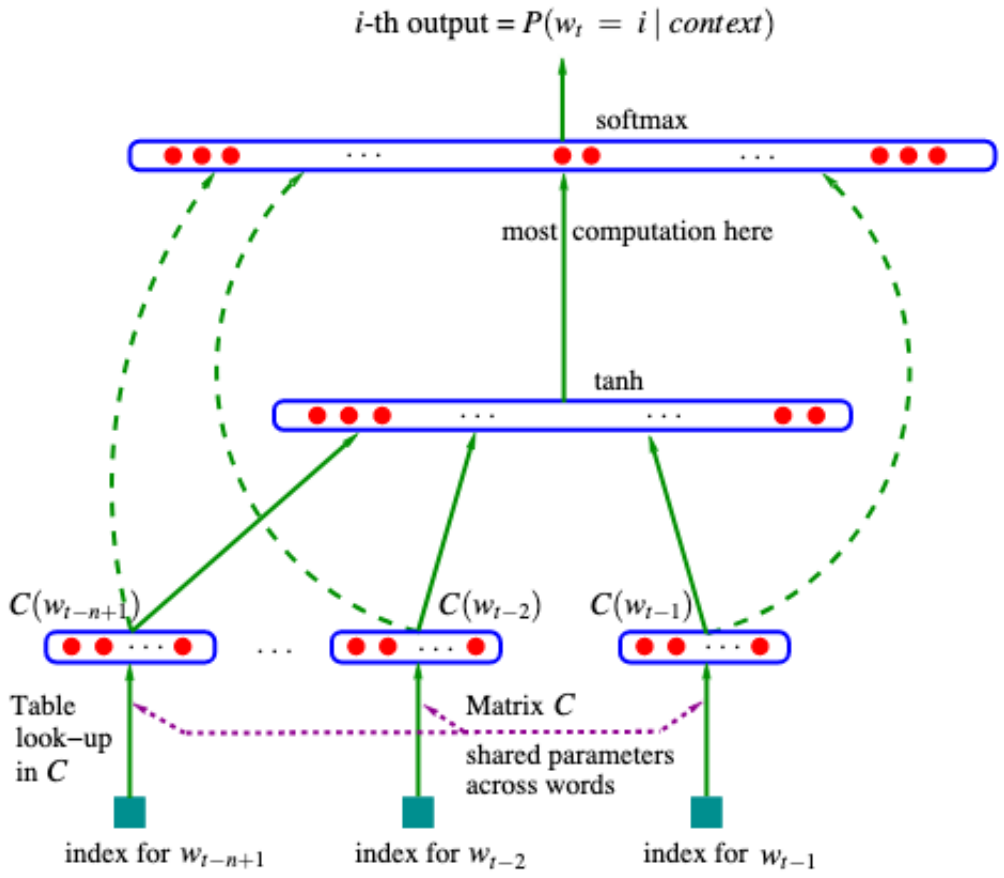
Why NNLM?



- Overcomes curse of dimensionality ($O(|V|^N)$ vs $O(|V|)$)
- Traditional n-Gram models have discrete probability distributions which are not smooth
- Traditional n-Gram models have no notion of similarity between words

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

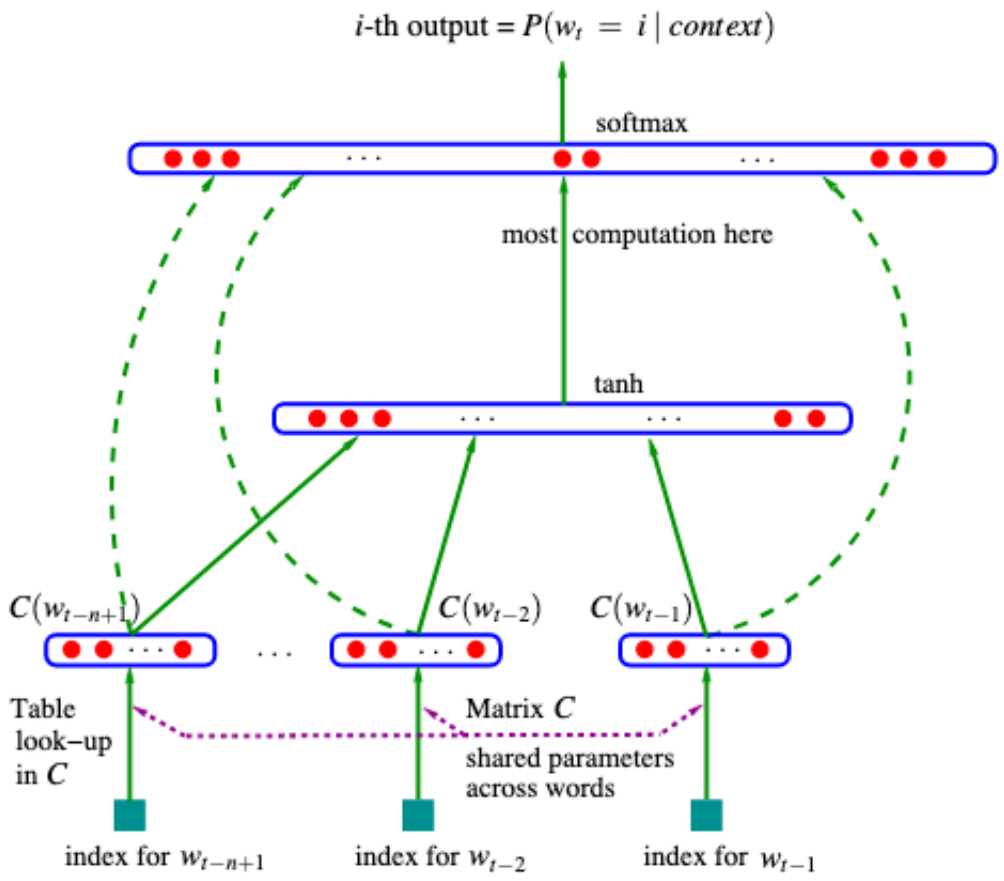
Why NNLM?



- Continuous distributed representations have smooth probability distribution
- There is notion of similarity
- NNLMs exploit the observation that words which occur in similar context have similar meaning
- Better generalization

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

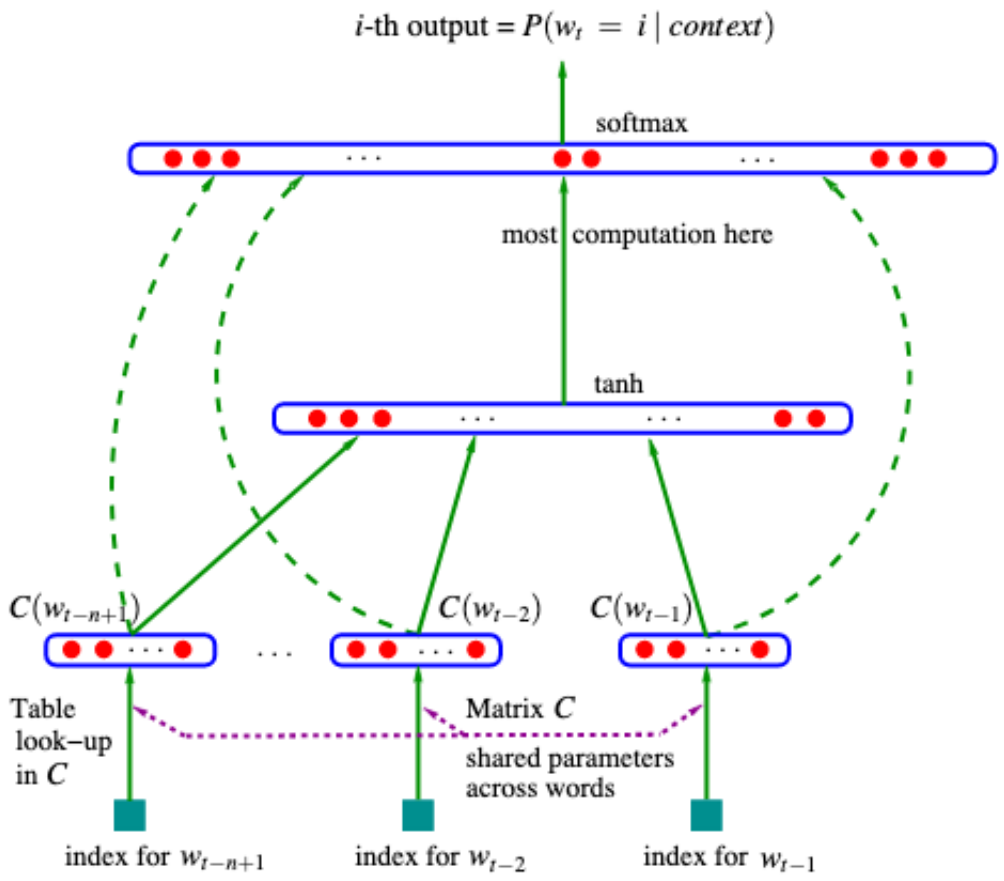
Why NNLM?



- NNLMs takes into account the semantic and syntactic properties of words
- NNLMs overcome data sparsity problem and can have much longer context window
- NNLMs have competitive performance and can be combined with traditional n-gram models

A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Challenges with NNLM

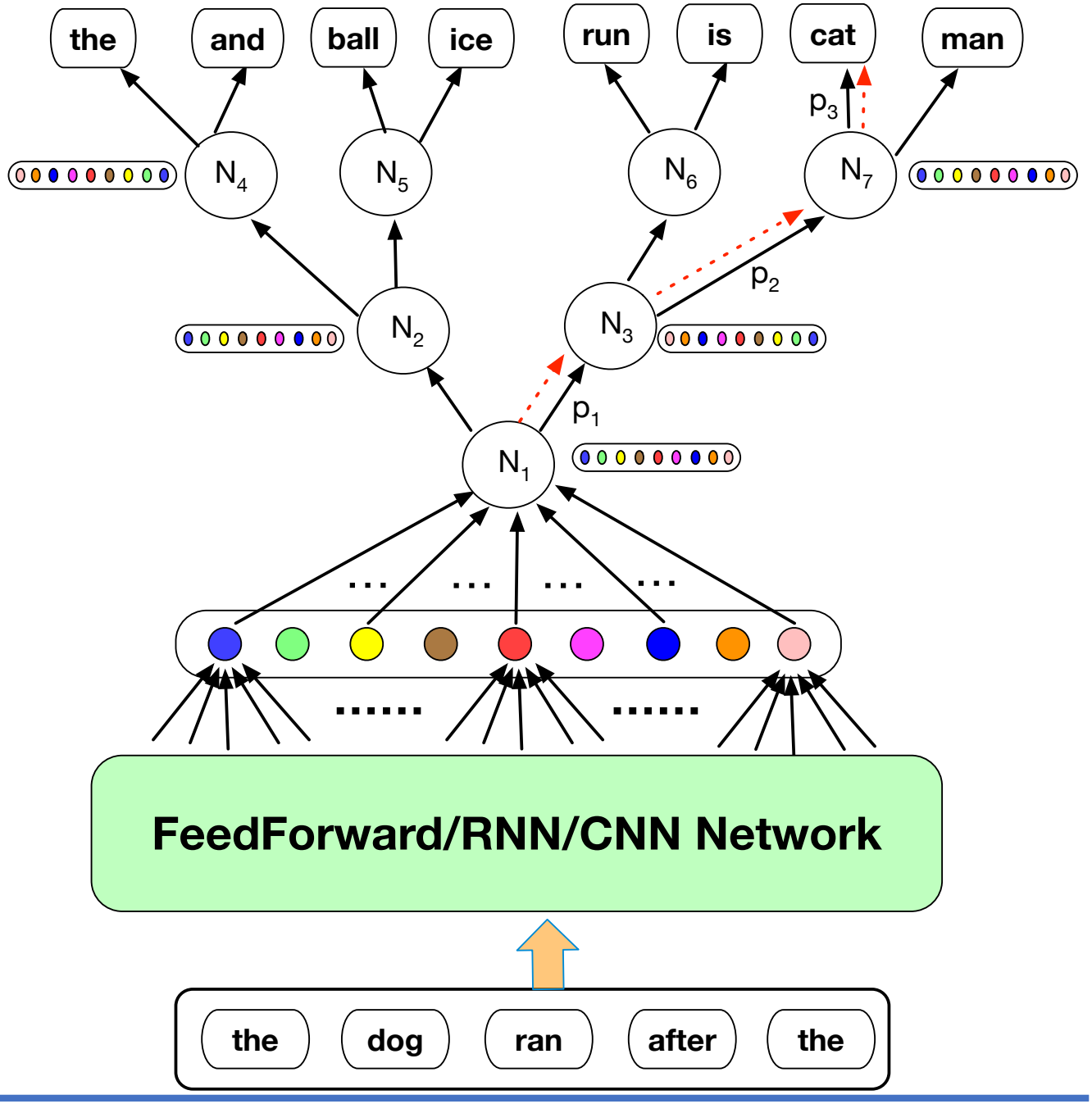


- NNLM model is linear in vocabulary size
- Main bottleneck in training NNLM is the vocabulary size
- For a corpus of 14 million tokens it took 3 weeks to train

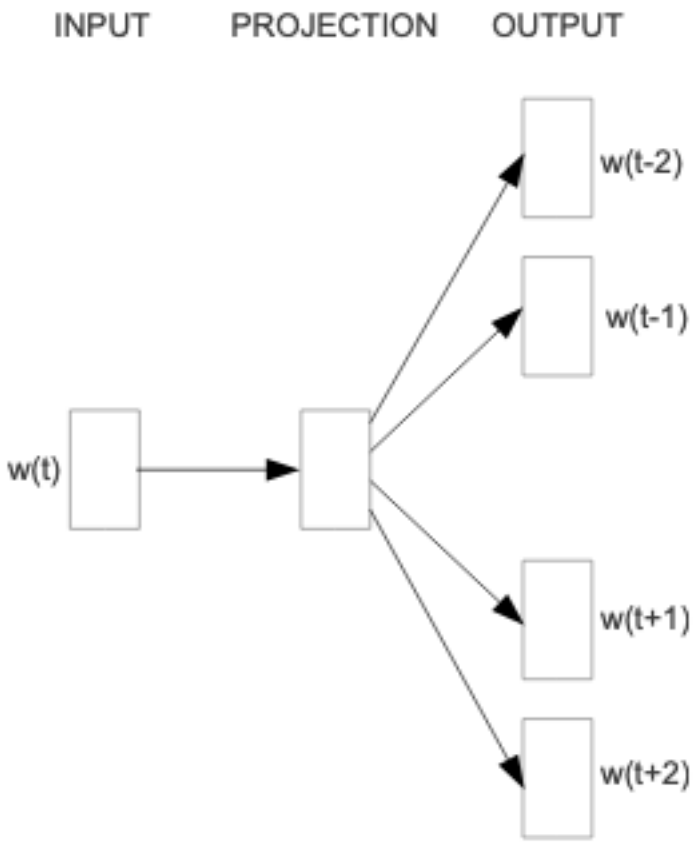
A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Hierarchical NNLM

- Create a binary tree over vocabulary representing hierarchical clustering over words
- Each word corresponds to a leaf in such tree
- Word can be predicted by following a path from root to the leaf node



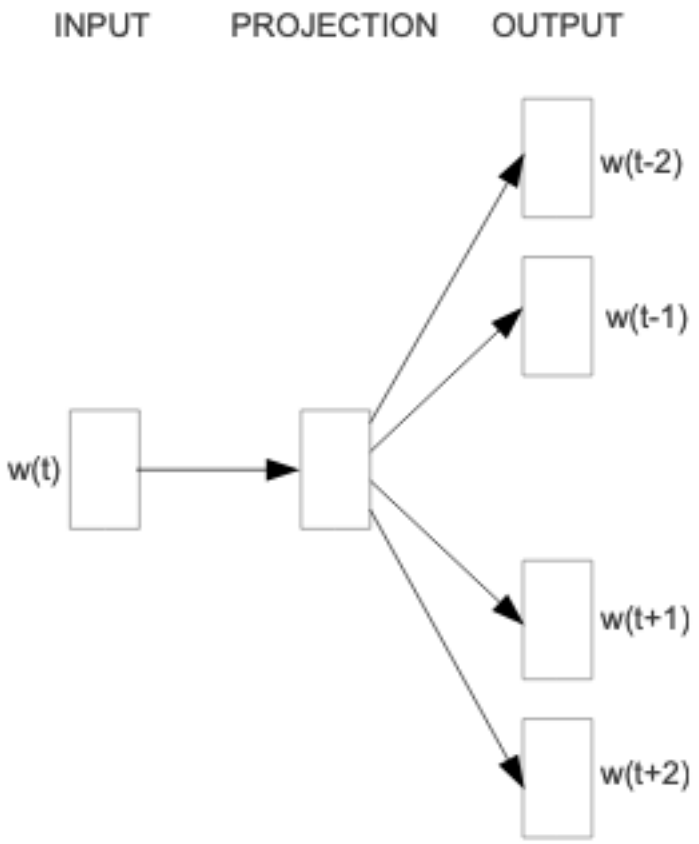
Word2Vec: Skip Gram Model



- Based on ***Distributional Hypothesis***: "a word is characterized by the company it keeps" -Firth (1957)
- The key idea is to predict the words in the context
- Learn word representation while making predictions
- Unsupervised method

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Word2Vec: Skip Gram Model

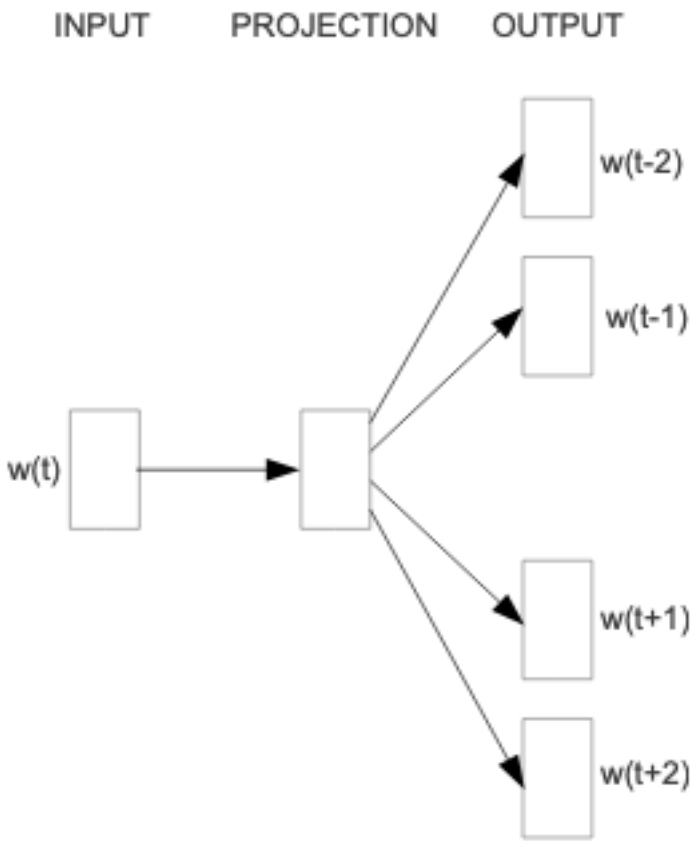


$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \sum_{-c \leq j \leq c, j \neq 0} log\ p(w_{i+j} \mid w_i)$$

$$p(w_{i+j} \mid w_i) = \frac{exp(v_{w_{i+j}}^T v_{w_i})}{\sum_{w=1}^{V} exp(v_w^T v_{w_i})}$$

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Word2Vec: Skip Gram Model



$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \sum_{-c \leq j \leq c, j \neq 0} log\ p(w_{i+j} \mid w_i)$$

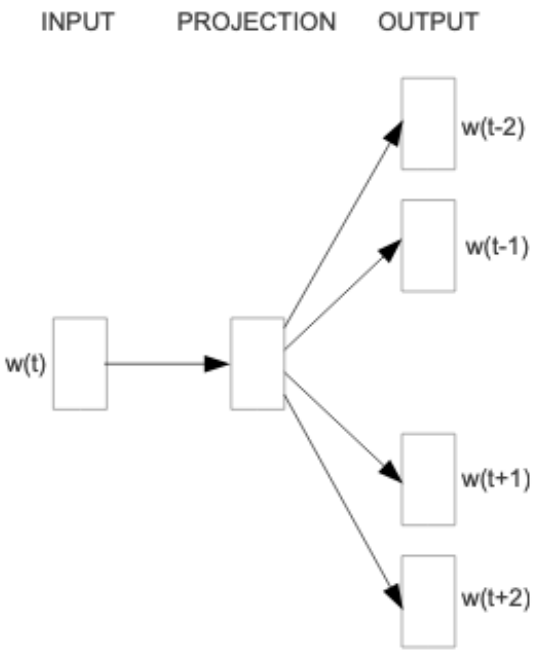
Embedding for the context word

Embedding for the input word

$$p(w_{i+j} \mid w_i) = \frac{exp(v^T_{w_{i+j}} v_{w_i})}{\sum_{w=1}^{V} exp(v^T_w v_{w_i})}$$

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Word2Vec: Scaling over a large vocabulary

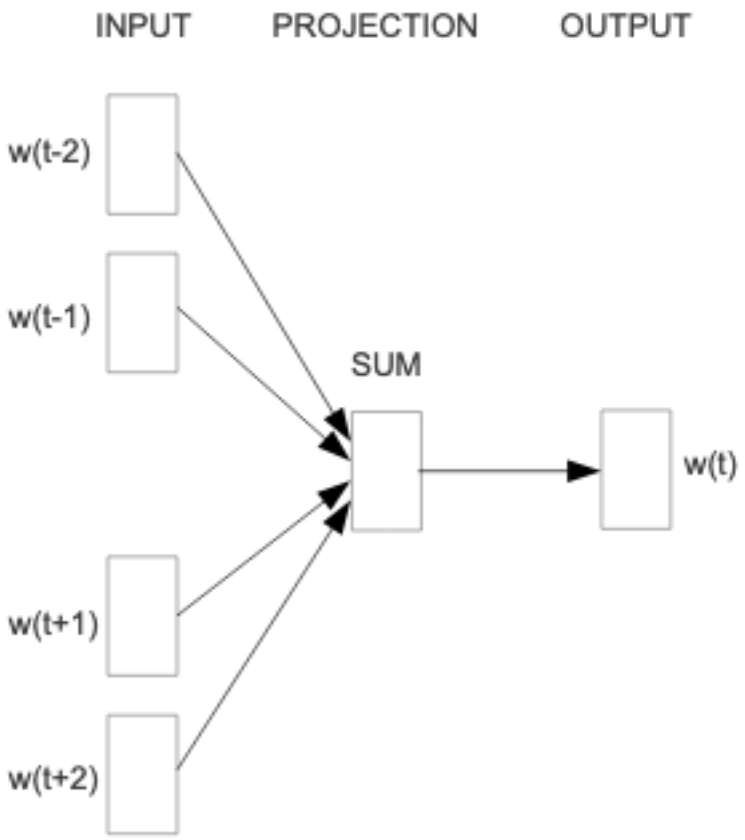


- Hierarchical Softmax
- Negative Sampling

$$log\ \sigma(v_{w_{i+j}}^T v_{w_i}) + \sum_{l=1}^{k} E_{w_l \sim P_n(w)}[log\ \sigma(-v_{w_l}^T v_{w_i})]$$

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Word2Vec: CBOW



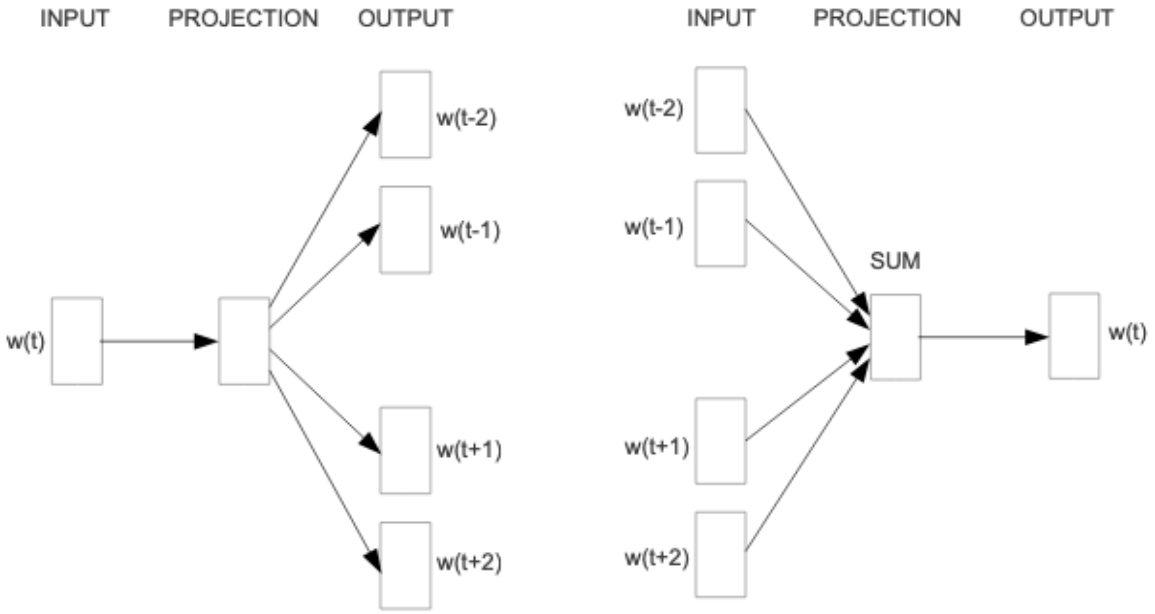
- Given the context predict the word

$$p(w_i \mid w_{i-1}, w_{i-2}, w_{i+1}, w_{i+2}) = \frac{exp(v_{w_c}^T v_{w_i})}{\sum_{w=1}^{V} exp(v_{w_c}^T v_w)}$$

$$v_{w_c} = \sum_{-c \leq j \leq c, j \neq 0} v_{w_{i+j}}$$

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Word2Vec Results

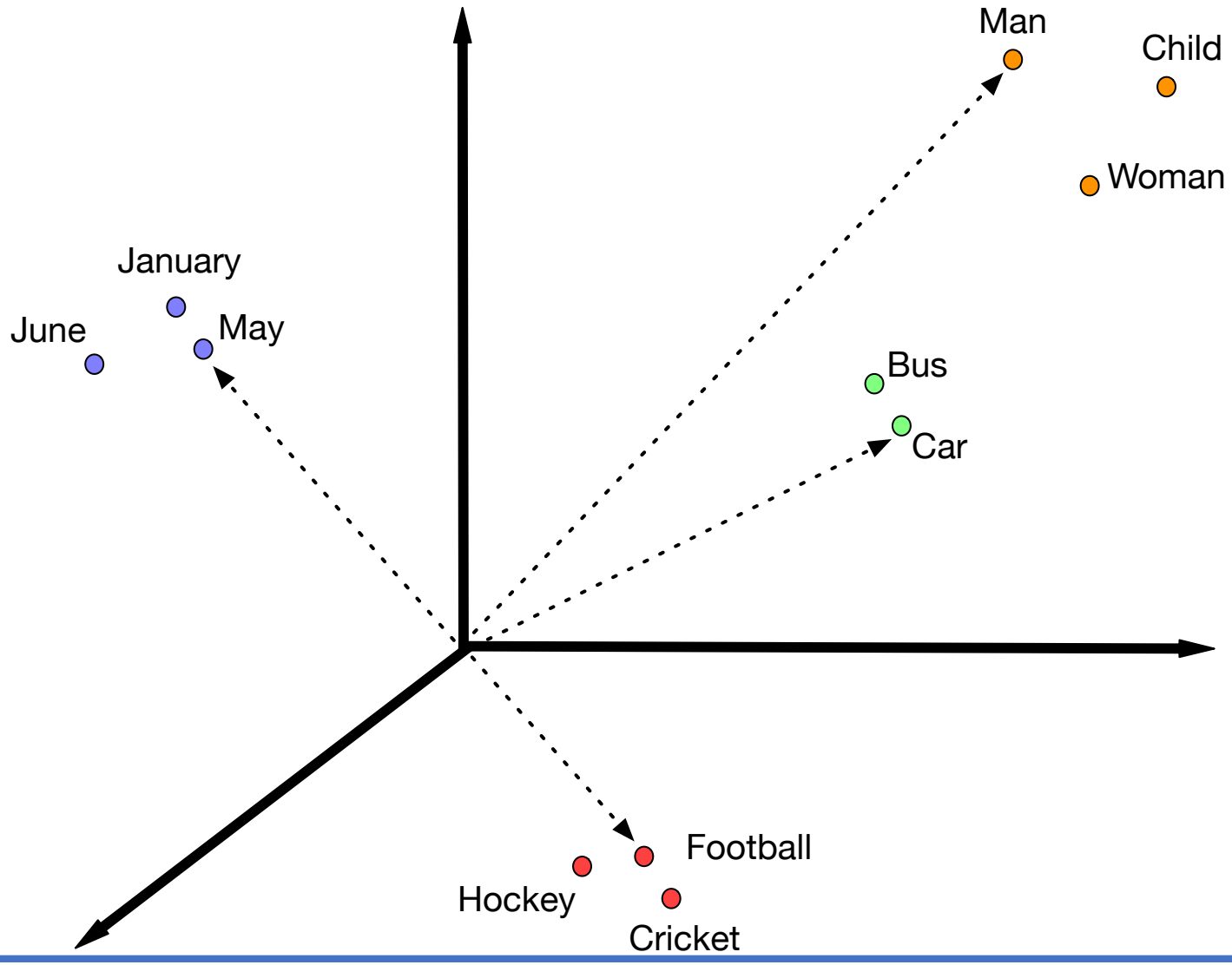


- Word2Vec learns distributed representation of a word: *Word Embedding*
- It is like a feature vector for a word which encodes the properties of the words
- Word embeddings have been found to be useful in several upstream tasks
- Word2Vec learns linear relationship between words

Vec(“Germany”) + Vec(“capital”) ~ Vec(“Berlin”)

Vec(“King”) – Vec(“Man”) ~ Vec(“Queen”) – Vec(“Woman”)

Words Embeddings: Vectors of Meaning



Summary

- Distributed Representations represent the properties of a concept
- Distributed Representations : One for all, all for one
- Neural Network Language Model (NNLM) predict the next word based on distributed representation of the context word
- NNLM generalize better and do not suffer from sparsity
- Properties/features of a word are encoded in the word embedding
- Word embedding can be learned using Skip-Gram or CBOW models

References

1. Distributed Representations: https://web.stanford.edu/~jlmcc/papers/PDP/Chapter3.pdf
2. A Neural Probabilistic Language Model: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf
3. Strategies for Training Large Vocabulary Neural Language Models : https://www.aclweb.org/anthology/P16-1186.pdf
4. Word2Vec Papers: https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf and https://arxiv.org/pdf/1301.3781.pdf
5. Notes on Backpropagation: https://www.ics.uci.edu/~pjsadows/notes.pdf

- Next class
  - Structure Prediction in NLP