Lecture 1: Word embeddings: LSA, Word2Vec, Glove, ELMo

Dr. Reda Bouadjenek

Data-Driven Decision Making Lab (D3M)

21 May 2020

DEAKIN UNIVERSITY

Vector Embedding of Words

- Mapping a word to a vector.
  - The semantic of the word is embedded in the vector.
- Word embeddings depend on a notion of ***word similarity***.
  - Similarity is computed using cosine.
- A very useful definition is paradigmatic similarity:
  - ***Similar words*** occur in ***similar contexts -*** they are ***exchangeable.***
  - Yesterday {POTUS* / The President / Trump} called a press conference.
- Transfer learning for text.

* POTUS: President of the United States.

Word Embedding vs. Bag of Words

Traditional Method - Bag of Words Model

Two approaches:

- Either uses one hot encoding.
  - Each word in the vocabulary is represented by one bit position in a HUGE vector.
  - For example, if we have a vocabulary of 10,000 words, and "aardvark" is the ***4th word in the dictionary***, it would be represented by: [0 0 0 1 0 0 . . . . 0 0 0].
- Or uses document representation.
  - Each word in the vocabulary is represented by its presence in documents.
  - For example, if we have a corpus of 1M documents, and "Hello" is in 1th, 3th and 5th documents ***only***, it would be represented by: [1 0 1 0 1 0 . . . . 0 0 0].
- Assumes independence between words.

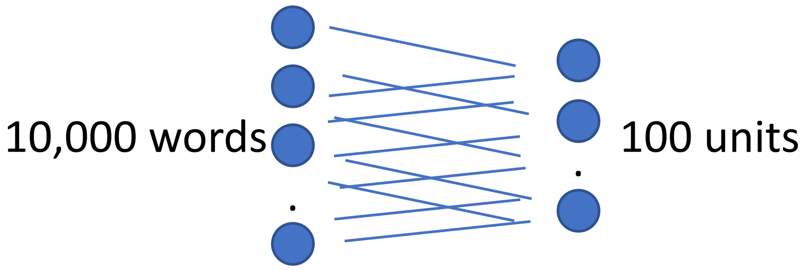
Word Embeddings

- Stores each word in as a point in space, where it is represented by a dense vector of fixed number of dimensions (generally 300) .
  - For example, "Hello" might be represented as : [0.4, -0.11, 0.55, 0.3 . . . 0.1, 0.02].
  - Dimensions are projections along different axes, more of a mathematical concept.
- Unsupervised, built just by reading huge corpus.
- Assumes dependence between words.

Word Embedding vs. Bag of Words

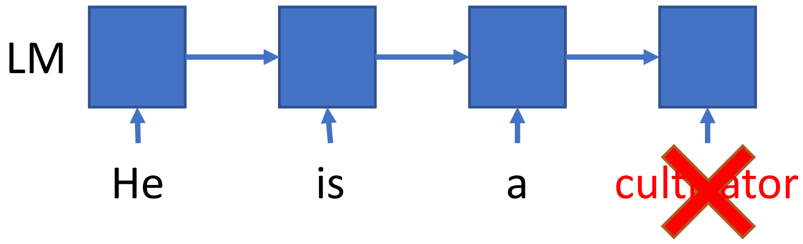
Traditional Method - Bag of Words Model

- Requires **very** large weight matrix for 1^{st} layers.



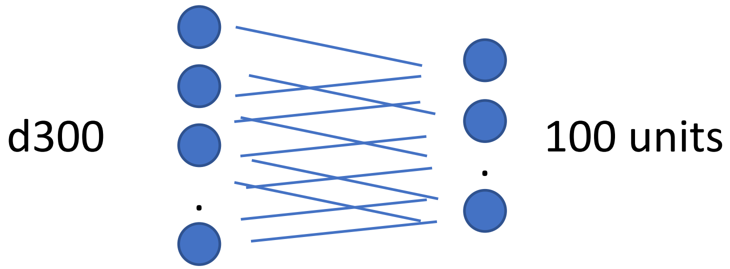
W's size is 10,000x100 = 10^6

- Models not flexible with unseen words in the training set.



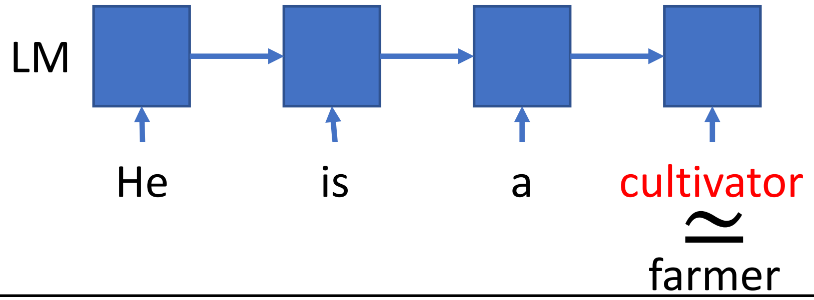
Word Embeddings

- A **compact** weight matrix for 1^{st} layers.

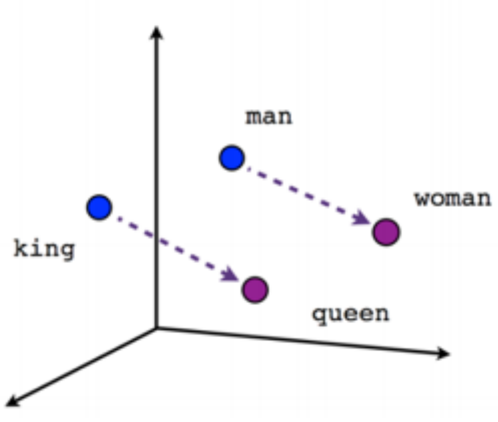


W's size is 300x100 = $3x10^4$

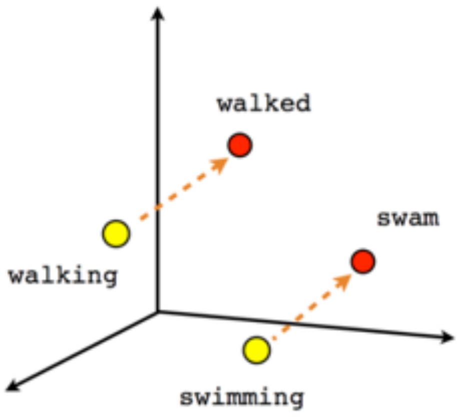
- Flexible models with unseen words in the training set.



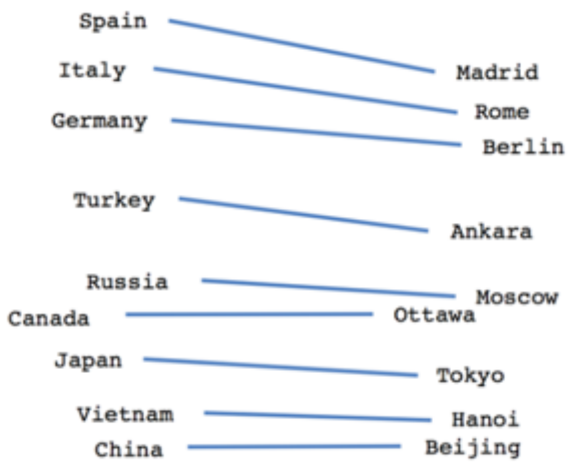
Example 1: Working with vectors



Male-Female



Verb tense



Country-Capital

- vector[Queen] ≈ vector[King] - vector[Man] + vector[Woman]
- vector[Paris] ≈ vector[France] - vector[Italy] + vector[Rome]
    - This can be interpreted as "France is to Paris as Italy is to Rome".
- May Learn unhealthy stereotypes (Covered in SIT799)
    - vector[Homemaker] ≈ vector[Women] - vector[Man] + vector[Computer Programmer]

Example 2: Working with vectors

- Finding the most similar words to $\overrightarrow{dog}$.
  - Compute the similarity from word $\overrightarrow{dog}$ to all other words.
  - This is a single matrix-vector product: $W \cdot \overrightarrow{dog}$
    - W is the word embedding matrix of **|*V*|** rows and ***d*** columns.
    - Result is a |V| sized vector of similarities.
    - Take the indices of the k-highest values.

```
dog: 0.9999999403953552
dogs: 0.8680489659309387
puppy: 0.8106428384780884
pit_bull: 0.780396044254303
pooch: 0.7627377510070801
cat: 0.7609456777572632
golden_retriever: 0.7500902414321899
German_shepherd: 0.7465174198150635
Rottweiler: 0.7437614798545837
beagle: 0.7418621778488159
```

Example 3: Working with vectors

- Similarity to a group of words
  - “Find me words most similar to cat, dog and cow”.
  - Calculate the pairwise similarities and sum them:

$$\mathrm{W} \cdot \overrightarrow{cat} + \mathrm{W} \cdot \overrightarrow{dog} + \mathrm{W} \cdot \overrightarrow{cow}$$

  - Now find the indices of the highest values as before.
  - Matrix-vector products are wasteful. Better option:

$$\mathrm{W} \cdot (\overrightarrow{cat} + \overrightarrow{dog} + \overrightarrow{cow})$$

Applications of Word Vectors

- Word Similarity
- Machine Translation
- Part-of-Speech and Named Entity Recognition
- Relation Extraction
- Sentiment Analysis
- Co-reference Resolution
- Clustering
- Semantic Analysis of Documents

Vector Embedding of Words

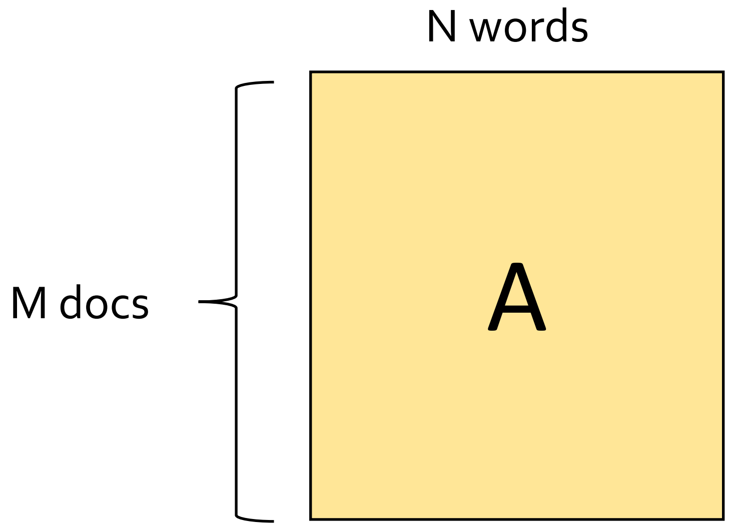
- Four main methods described in the talk :
  - Latent Semantic Analysis/Indexing (1988)
    - Term weighting-based model
    - Consider occurrences of terms at document level.
  - Word2Vec (2013)
    - Prediction-based model.
    - Consider occurrences of terms at context level.
  - GloVe (2014)
    - Count-based model.
    - Consider occurrences of terms at context level.
  - ELMo (2018)
    - Language model-based.
    - A different embedding for each word for each task.

Latent Semantic Analysis

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. "Indexing by latent semantic analysis." Journal of the American society for information science 41, no. 6 (1990): 391-407.

Embedding: Latent Semantic Analysis

- Latent semantic analysis studies documents in Bag-Of-Words model (1990).
  - i.e. given a matrix ***A*** encoding some documents: A_{ij} is the count* of word ***j*** in document ***i***. Most entries are 0.



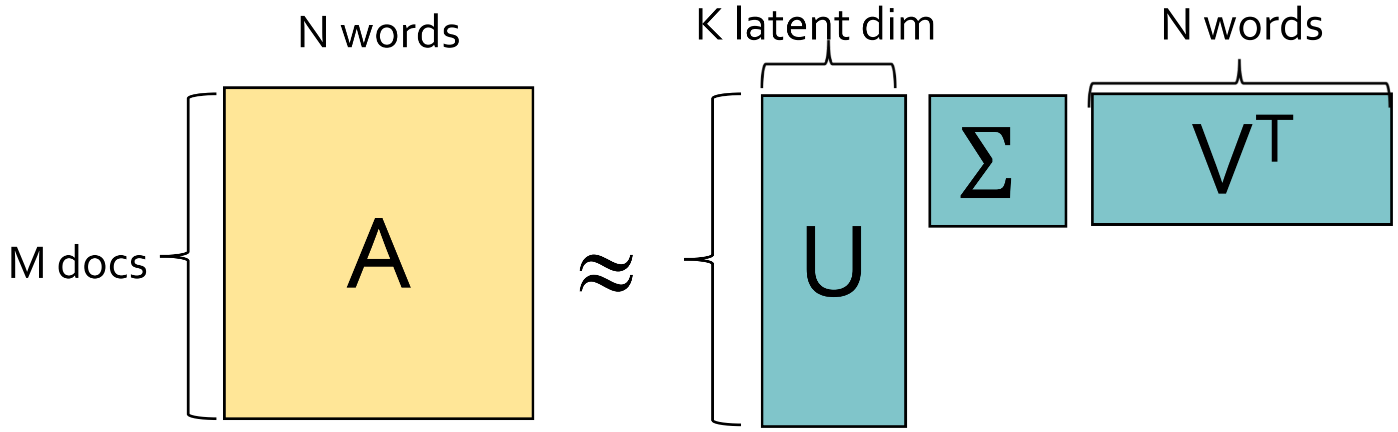
* Often tf-idf or other "squashing" functions of the count are used.

Embedding: Latent Semantic Analysis

- Low rank SVD decomposition:

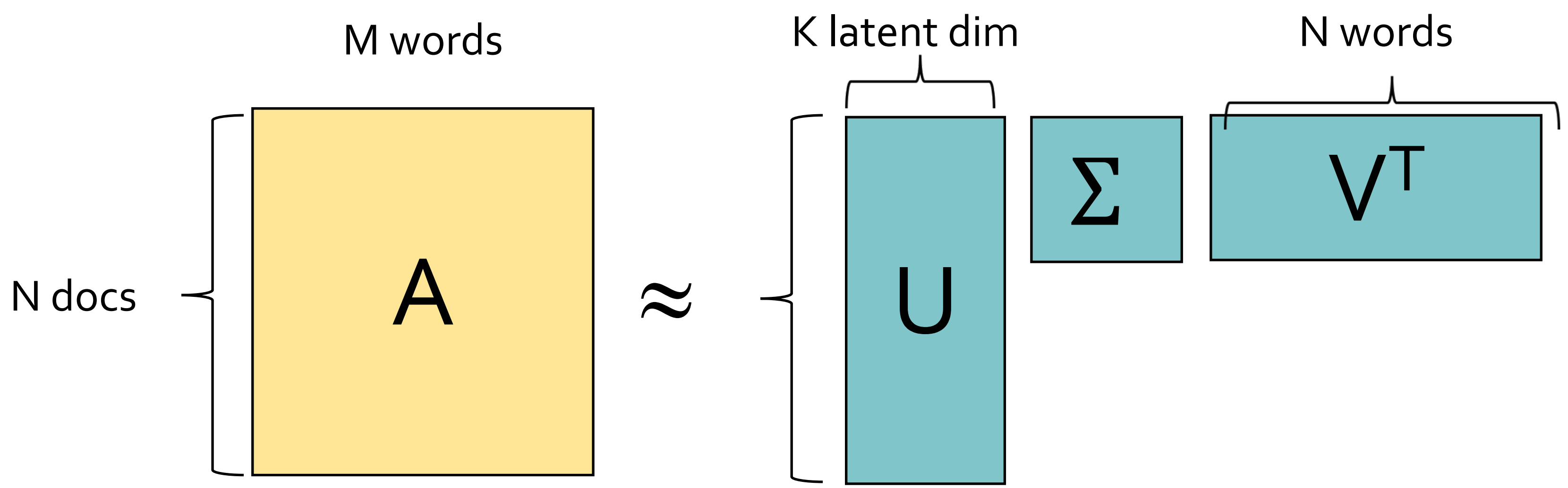
$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

  - U : document-to-concept similarities matrix (orthogonal matrix).
  - V : word-to-concept similarities matrix (orthogonal matrix).
  - Σ : strength of each concept.
- Then given a word ***w*** (column of ***A***):
  - $\varsigma = w^T \times U$ is the embedding (encoding) of the word ***w*** in the latent space.
  - $w \approx U \times \varsigma^T = U \times (w^T \times U)^T$ is the decoding of the word ***w*** from its embedding.



Embedding: Latent Semantic Analysis

- $w \approx U \times \varsigma^T = U \times (w^T \times U)^T$ is the decoding of the word ***w*** from its embedding.
  - An SVD factorization gives the **best possible reconstructions** of the a word ***w*** from its embedding.
- Note:
  - The problem with this method, is that we may end up with matrices having billions of rows and columns, which makes **SVD computationally expensive and restrictive**.



Word2Vec

Distributed representations of words and phrases and their compositionality.

T Mikolov, I Sutskever, K Chen, GS Corrado, J Dean, NIPS 2013.

word2Vec: Local contexts

- Instead of entire documents, ***Word2Vec*** uses words ***k*** positions away from each center word.
  - These words are called **context words**.

- Example for ***k=3***:
  - "It was a bright cold day in April, and the clocks were striking".
  - Center word: red (also called focus word).
  - Context words: blue (also called target words).

- Word2Vec considers all words as center words, and all their context words.

Word2Vec: Data generation (window size = 2)

- Example: d1 = "king brave man" , d2 = "queen beautiful women"

word	Word one hot encoding	neighbor	Neighbor one hot encoding
king	[1,0,0,0,0,0]	brave	[0,1,0,0,0,0]
king	[1,0,0,0,0,0]	man	[0,0,1,0,0,0]
brave	[0,1,0,0,0,0]	king	[1,0,0,0,0,0]
brave	[0,1,0,0,0,0]	man	[0,0,1,0,0,0]
man	[0,0,1,0,0,0]	king	[1,0,0,0,0,0]
man	[0,0,1,0,0,0]	brave	[0,1,0,0,0,0]
queen	[0,0,0,1,0,0]	beautiful	[0,0,0,0,1,0]
queen	[0,0,0,1,0,0]	women	[0,0,0,0,0,1]
beautiful	[0,0,0,0,1,0]	queen	[0,0,0,1,0,0]
beautiful	[0,0,0,0,1,0]	women	[0,0,0,0,0,1]
woman	[0,0,0,0,0,1]	queen	[0,0,0,1,0,0]
woman	[0,0,0,0,0,1]	beautiful	[0,0,0,0,1,0]

Word2Vec: Data generation (window size = 2)

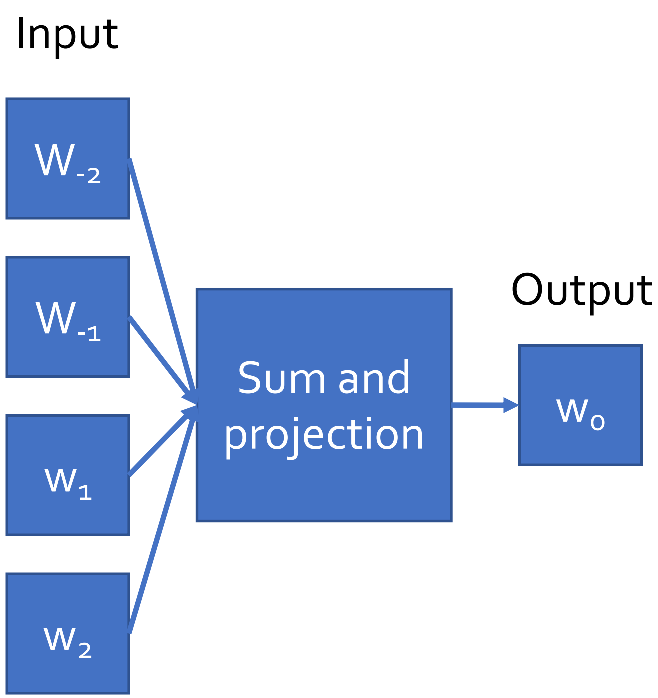
- Example: d1 = “king brave man” , d2 = “queen beautiful women”

word	Word one hot encoding	neighbor	Neighbor one hot encoding
king	[1,0,0,0,0,0]	brave	[0,1,1,0,0,0]
		man	
brave	[0,1,0,0,0,0]	king	[1,0,1,0,0,0]
		man	
man	[0,0,1,0,0,0]	king	[1,1,0,0,0,0]
		brave	
queen	[0,0,0,1,0,0]	beautiful	[0,0,0,0,1,1]
		women	
beautiful	[0,0,0,0,1,0]	queen	[0,0,0,1,0,1]
		women	
woman	[0,0,0,0,0,1]	queen	[0,0,0,1,1,0]
		beautiful	

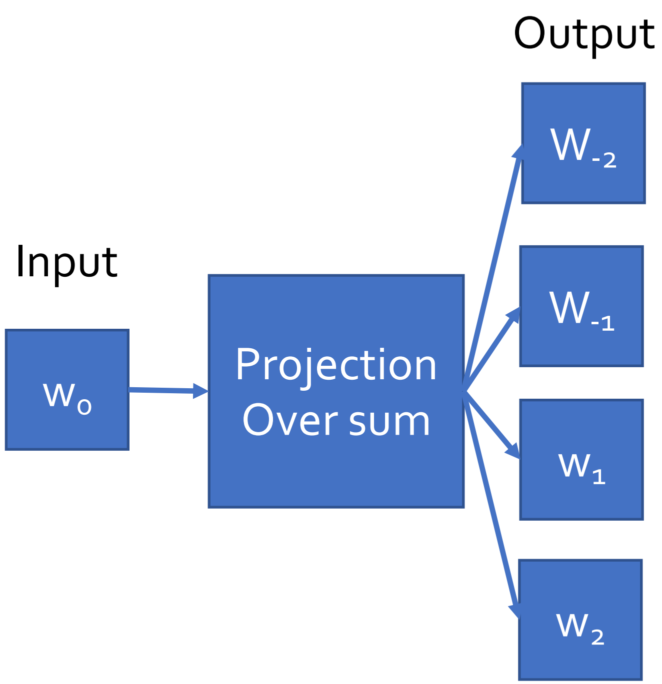
Word2Vec: main context representation models

Continuous Bag of Words

(CBOW)

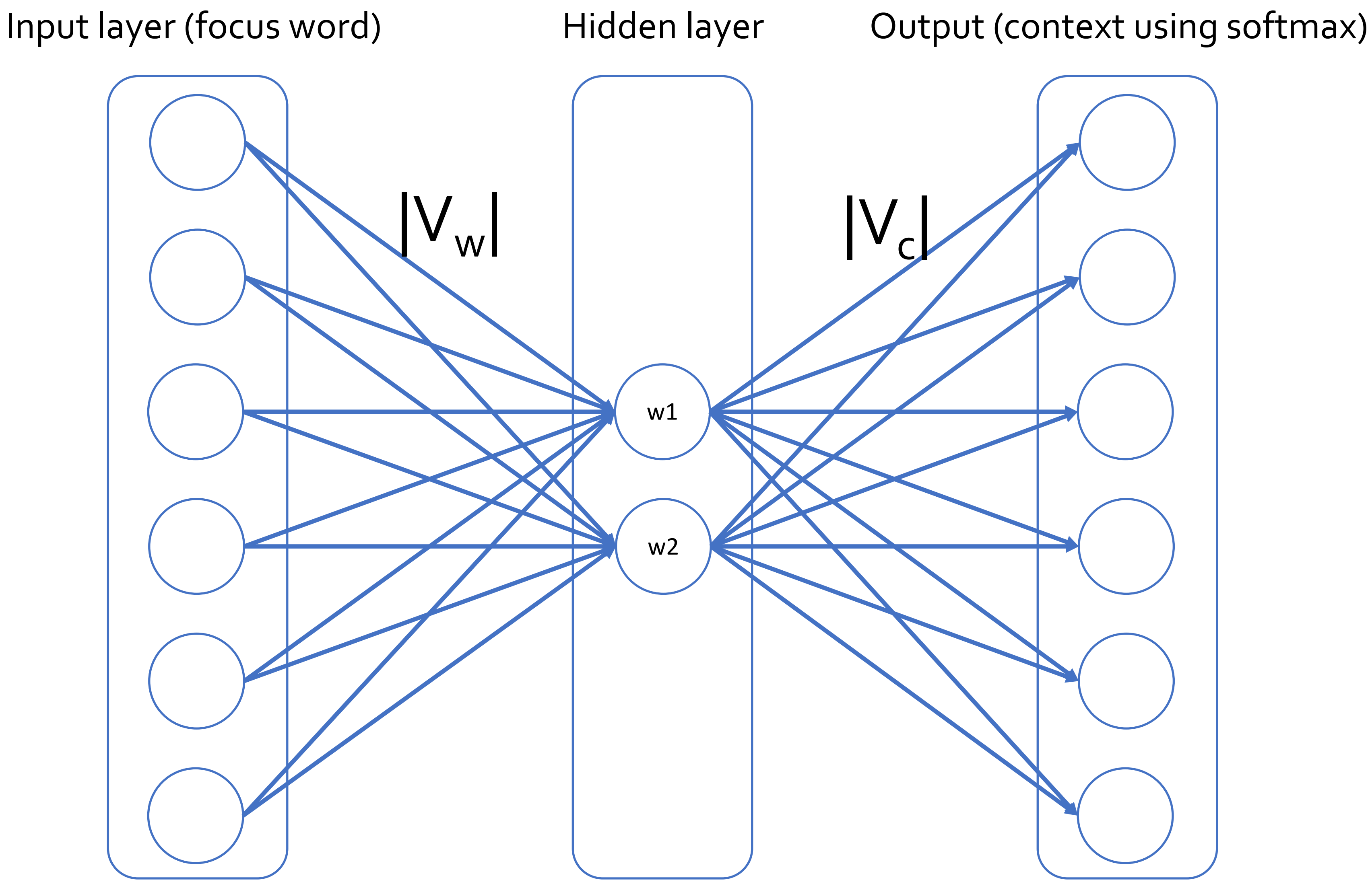


Skip-Ngram

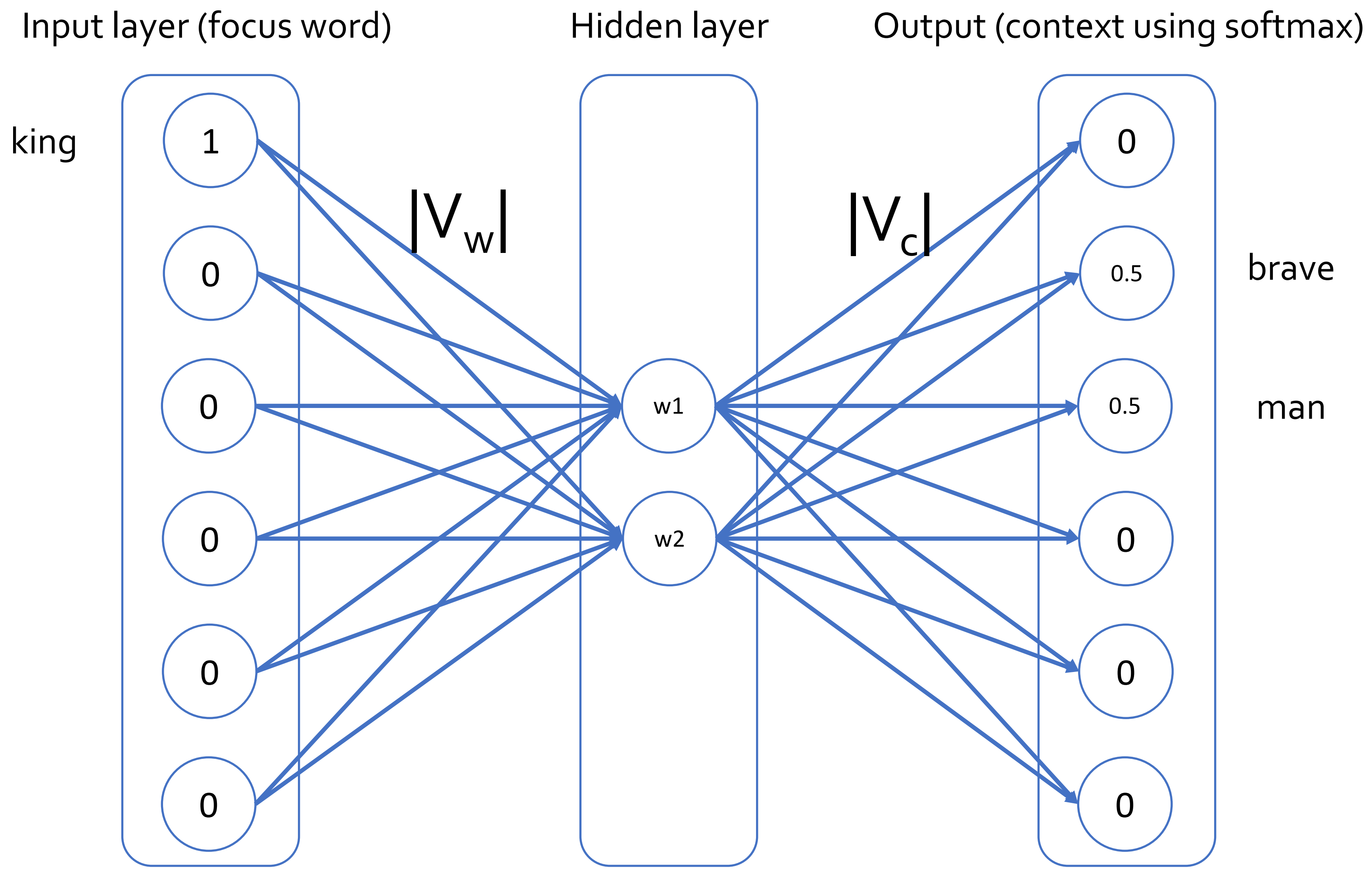


- Word2Vec is a predictive model.
- Will focus on Skip-Ngram model

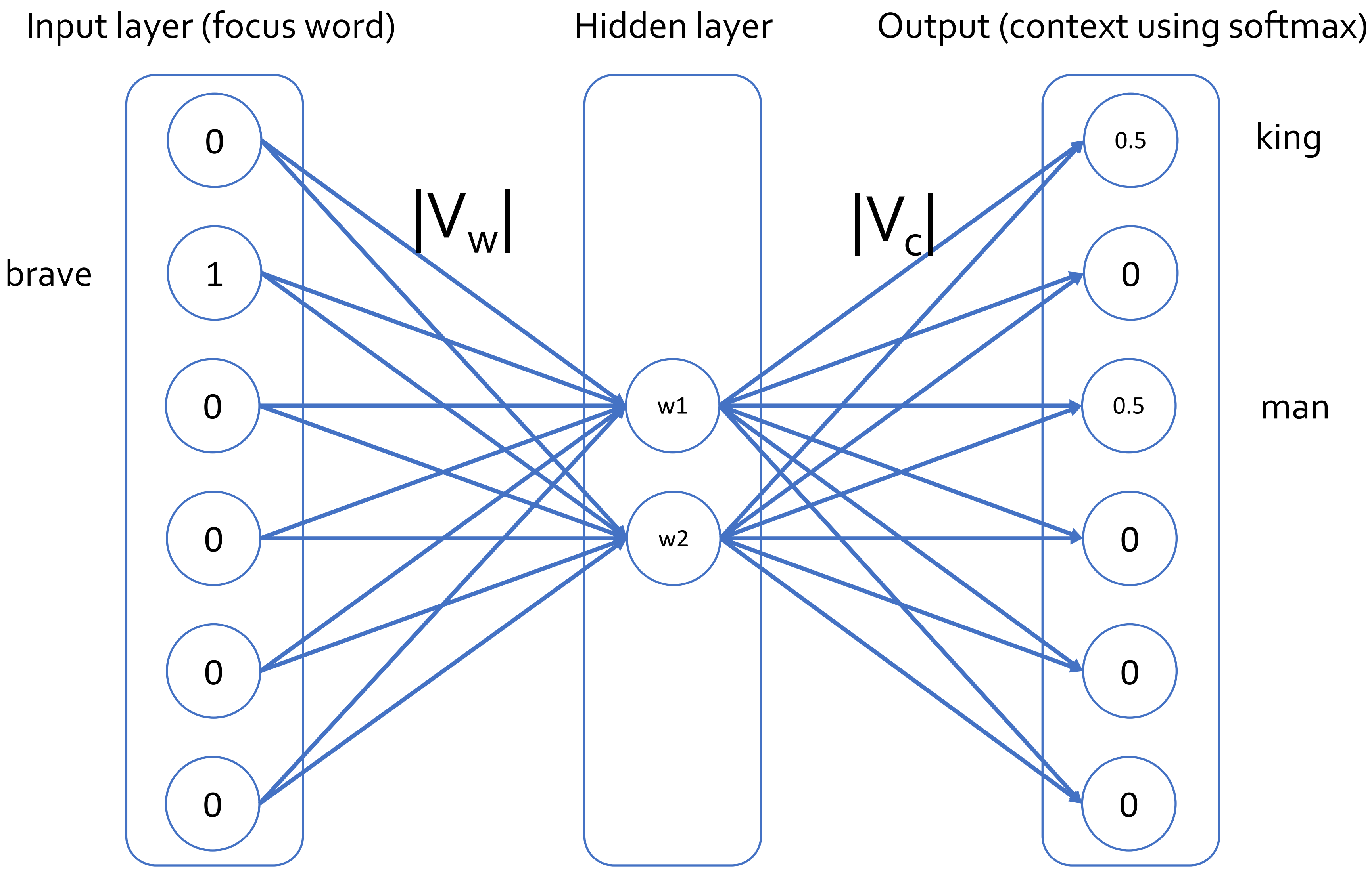
Word2Vec : Neural Network representation



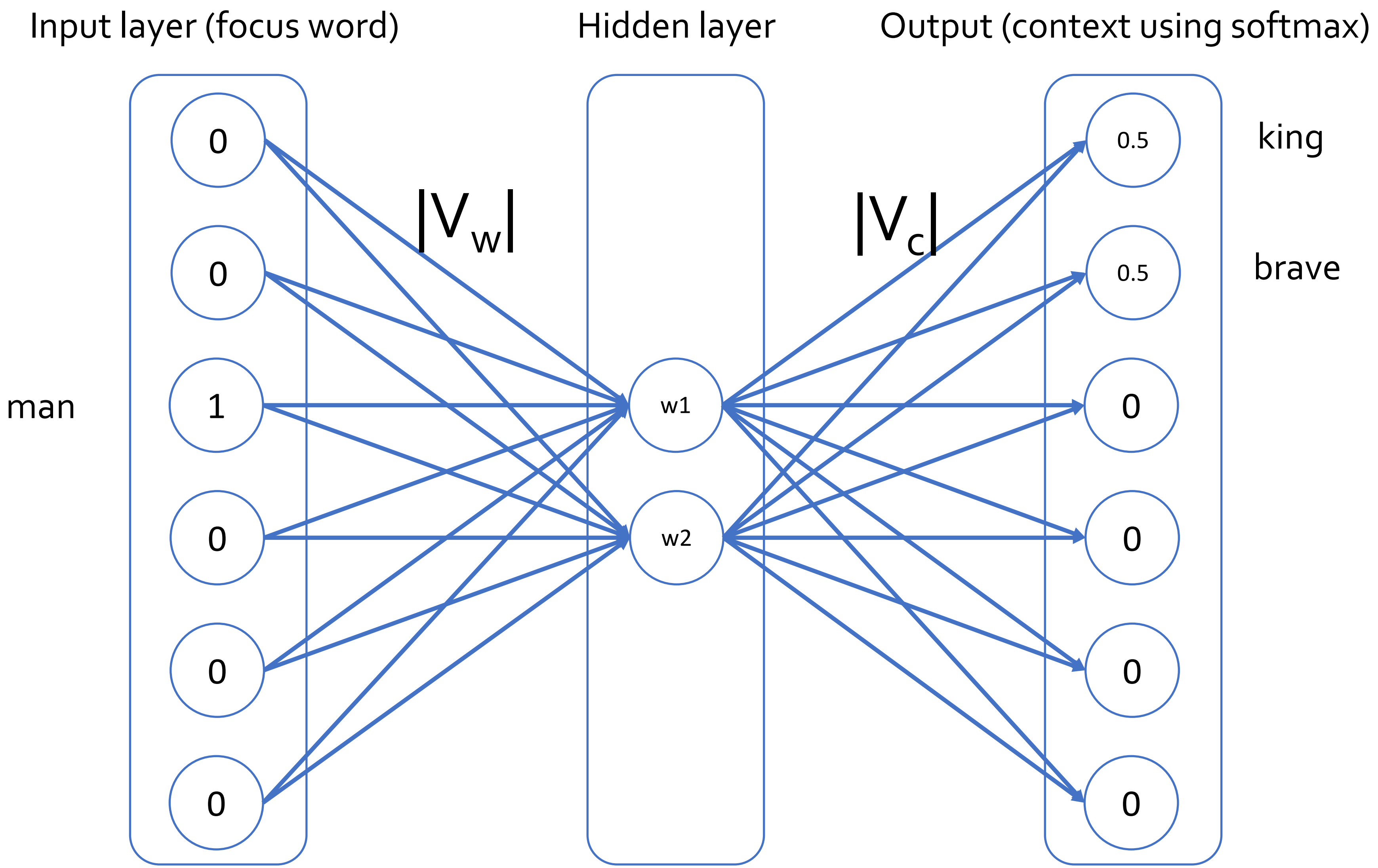
Word2Vec : Neural Network representation



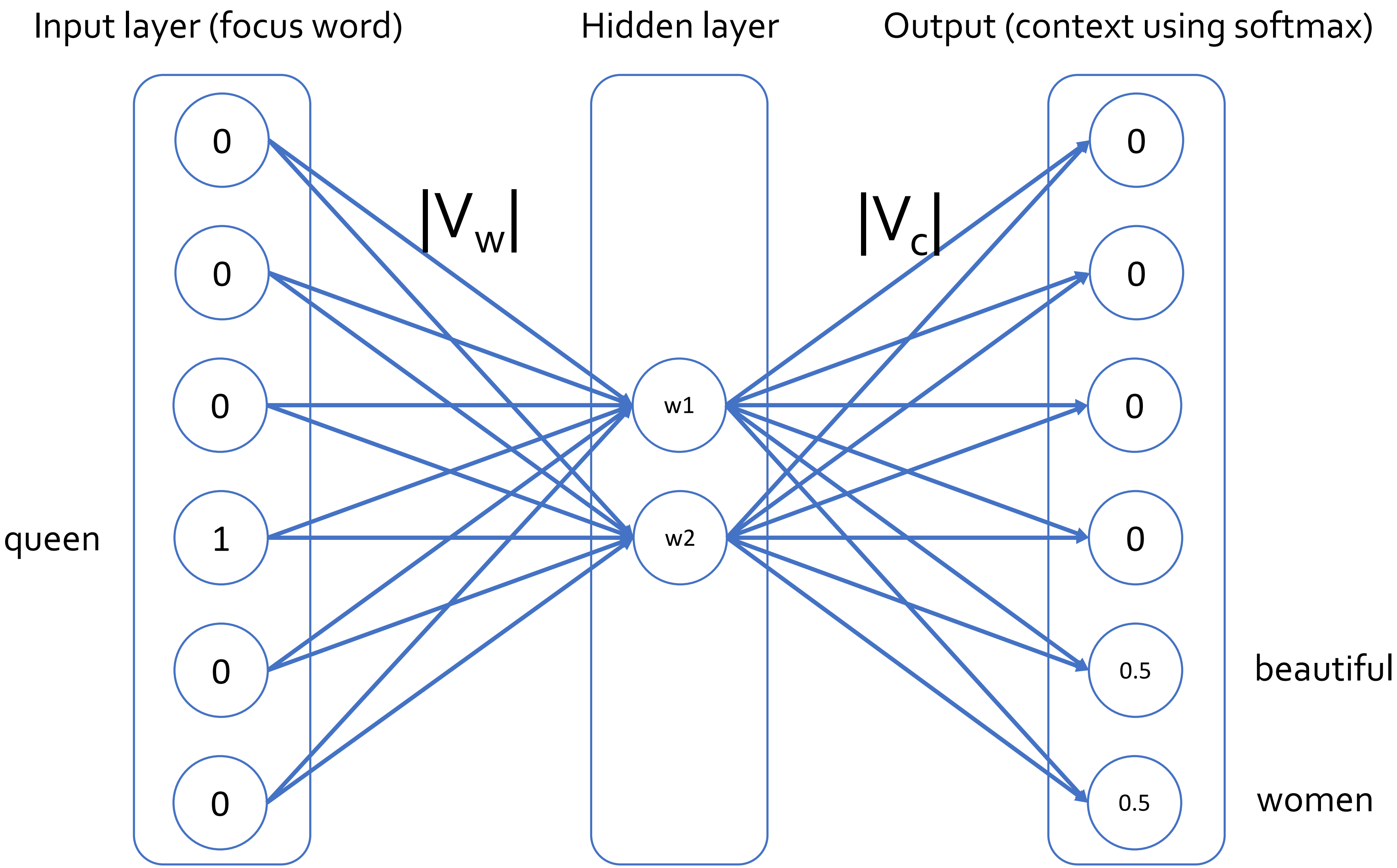
Word2Vec : Neural Network representation



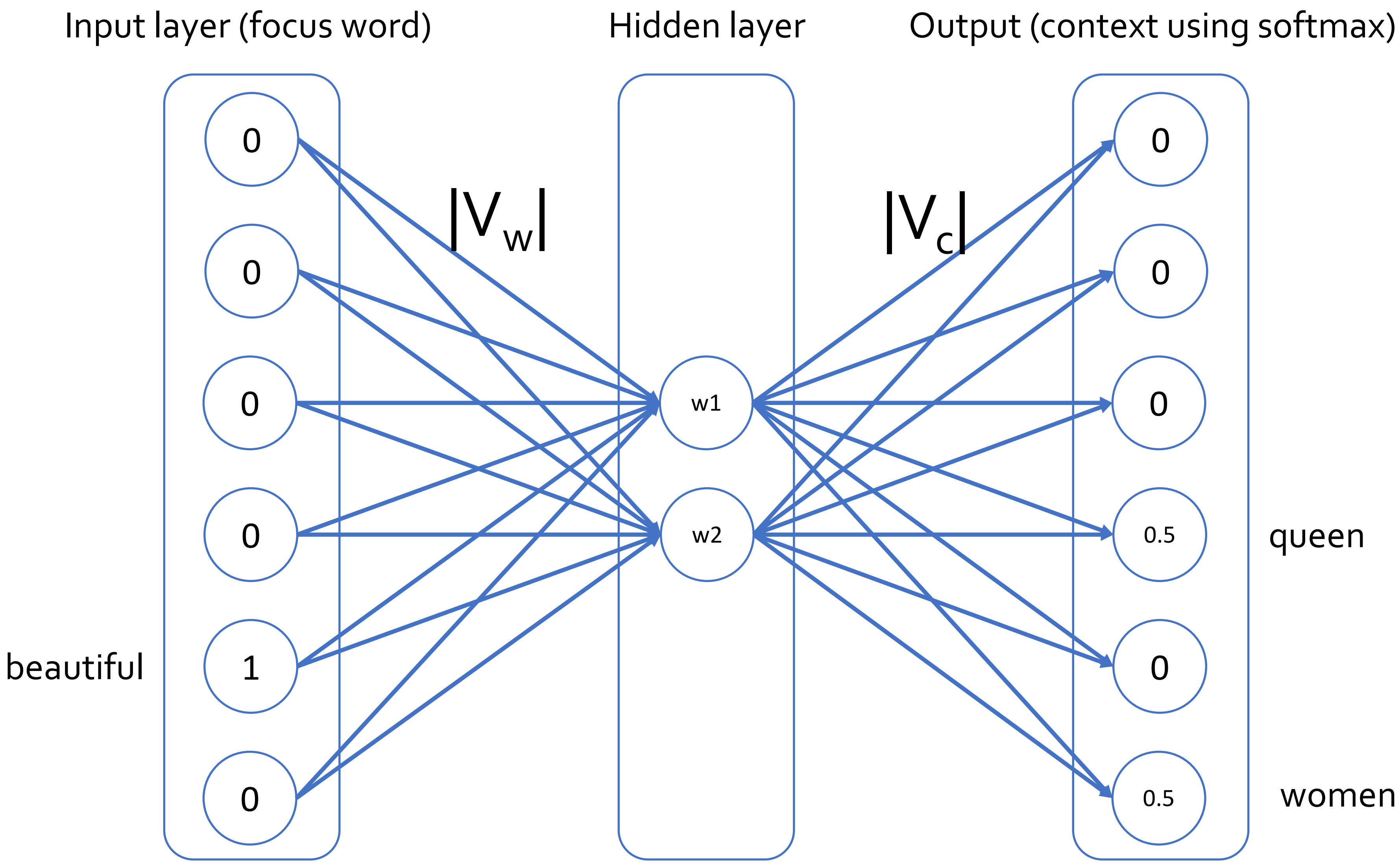
Word2Vec : Neural Network representation



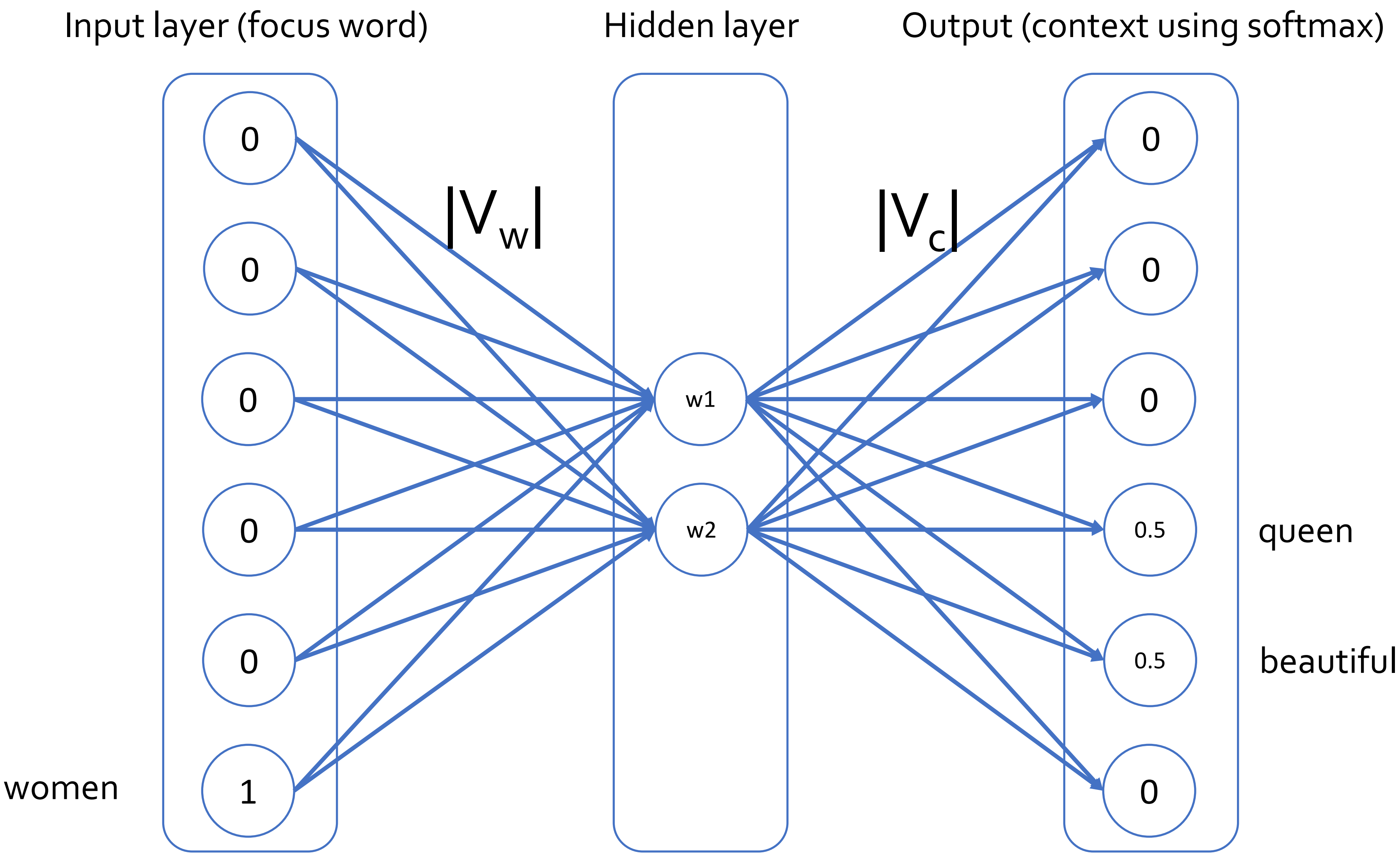
Word2Vec : Neural Network representation



Word2Vec : Neural Network representation



Word2Vec : Neural Network representation



Skip-Ngram: Training method

- The prediction problem is modeled using soft-max:

$$p(c|w;\theta) = \frac{\exp(v_c \cdot v_w)}{\sum_{\grave{c}\in C} \exp(v_{\grave{c}} \cdot v_w)}$$

  - Predict context word(s) ***c***
  - From focus word ***w***

- The objective function (Maximum Log Likelihood Estimate):

$$\underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log p(c|w;\theta) = \sum_{(w,c)\in D} \left[\log \exp(v_c \cdot v_w) - \log \sum_{\grave{c}\in C} \exp(v_{\grave{c}} \cdot v_w)\right]$$

  - While the objective function can be computed optimized, it is computationally expensive
    - $p(c|w;\theta)$ is very expensive to compute due to the summation $\sum_{\grave{c}\in C} \exp(v_{\grave{c}} \cdot v_w)$

Defining a new learning problem

- Example:
  - “king brave man”

Context word	Focus word	target
brave	king	1
juice	king	0
orange	king	0
mac	king	0
computer	king	0
java	king	0

k

- K = 5 to 20 for small collections.
- K= 2 to 5 for large collections.

Defining a new learning problem

- The new prediction problem is modeled using sigmoid function:

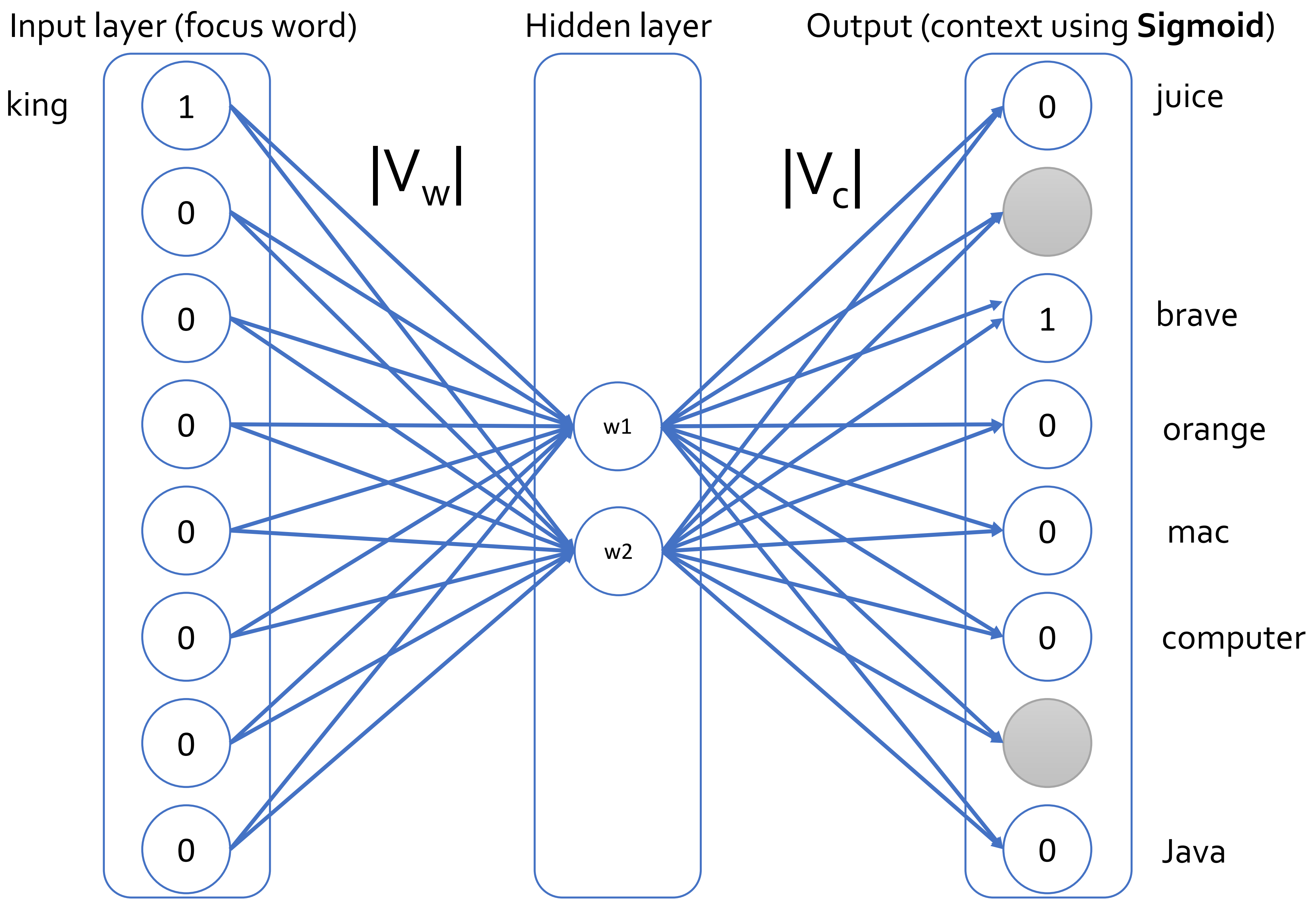
$$p(c|w;\theta) = \frac{1}{1 + e^{(-v_c \cdot v_w)}}$$

  - Predict context word ***c***
  - From focus word ***w***

- The new objective function (Maximum Log Likelihood Estimate):

$$\underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log \sigma(v_c \cdot v_w) + \sum_{(w,c)\in \acute{D}} \log \sigma(-v_c \cdot v_w)$$

Negative sampling : Neural Network representation



Skip-Ngram: How to select negative samples?

- Can sample using frequency.
  - Problem: will sample a lot of stop-words.
- Mikolov et al. proposed to sample using:

$$p(w_i) = \frac{f(w_i)^{3/4}}{\sum_j f(w_j)^{3/4}}$$

  - Not theoretically justified, but works well in practice!

Relations Learned by Word2Vec

- A relation is defined by the vector displacement in the first column. For each start word in the other column, the closest displaced word is shown.

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

- "Efficient Estimation of Word Representations in Vector Space" Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, Arxiv 2013

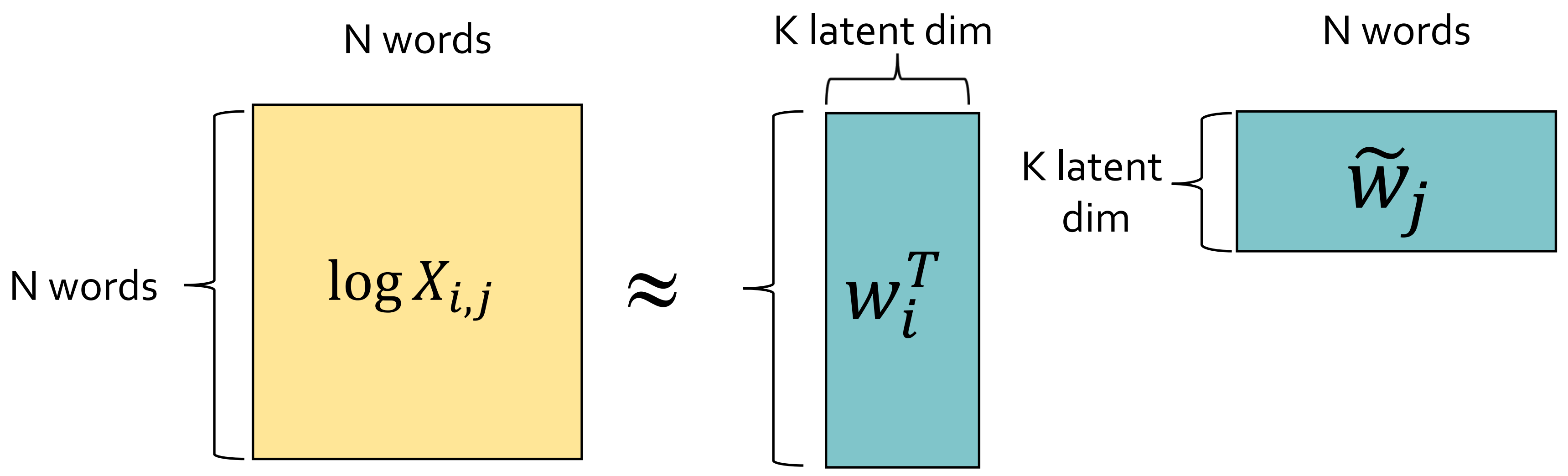
GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014.

GloVe: Global Vectors for Word Representation.

GloVe: Global Vectors for Word Representation

- While word2Vec is a predictive model — learning vectors to improve the predictive ability, **GloVe is a count-based model**.
- Count-based models learn vectors by doing dimensionality reduction on a **co-occurrence counts matrix**.
  - Factorize this matrix to yield a lower-dimensional matrix of words and features, where each row yields a vector representation for each word.



GloVe: Training

- The prediction problem is given by:

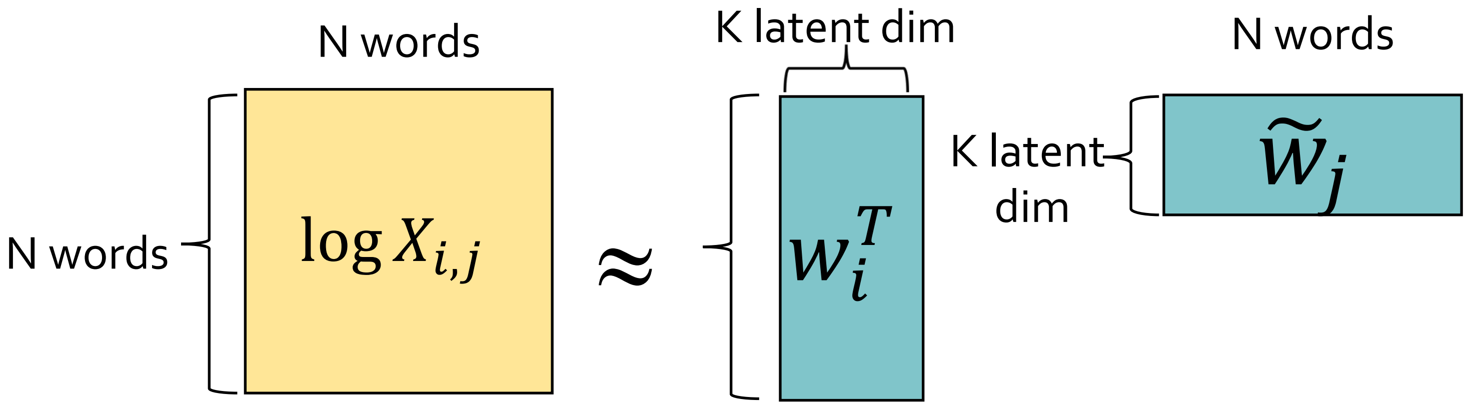
$$w_i^T \cdot \tilde{w}_j + b_i + \tilde{b}_j = \log X_{i,j}$$

  - b_w and b_c are bias terms.

- The objective function:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})\,(w_i^T \cdot \tilde{w}_j + b_i + \tilde{b}_j - \log X_{i,j})^2$$

  - $f(X_{i,j})$ is a weighting function to penalize rare co-occurrences.



GloVe: Training

- The model generates two sets of word vectors, W and $\widetilde{W}$.
- W and $\widetilde{W}$ are equivalent and differ only as a result of their random initializations.
  - The two sets of vectors should perform equivalently.
- Authors proposed to use $\frac{W+\widetilde{W}}{2}$ to get word vectors.

$$\frac{w_i^T + \widetilde{w}_j}{2}$$

ELMo: Embeddings from Language Models representations

Slides by *Alex Olson*

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer.

Deep contextualized word representations, 2018

Context is a key

- Language is complex, and ***context*** can completely change the meaning of a word in a sentence.
- Example:
  - I let the kids outside to ***play***.
  - He had never acted in a more famous ***play*** before.
  - It wasn't a ***play*** the coach would approve of.
- Need a model which captures the different nuances of the meaning of words given the surrounding text.

Different senses for different tasks

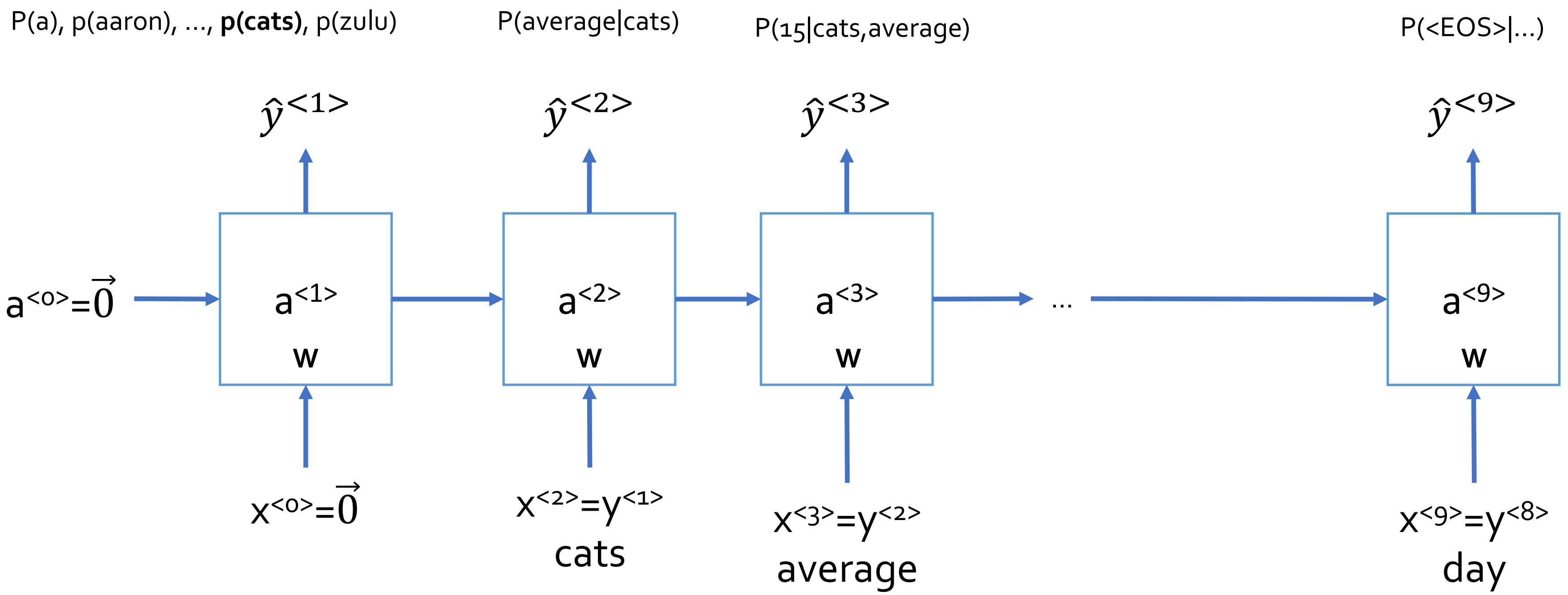
- Previous models (GloVe, Vord2Vec, etc.) only have one representation per word
  - They can't capture these ambiguities.
- When you only have one representation, all levels of meaning are combined.
- Solution: have multiple levels of understanding.
  - ELMo: Embeddings from ***Language Model*** representations.

What is language modelling?

- Today's goal: assign a probability to a sentence
  - Machine Translation:
    - P(**high** winds tonight) > P(**large** winds tonight)
  - Spell Correction
    - The office is about fifteen **minuets** from my house!
      - P(about fifteen **minutes** from) > P(about fifteen **minuets** from)
  - Speech Recognition
    - P(I saw a van) >> P(eyes awe of an)
  - + Summarization, question, answering, etc., etc.!!
  - Reminder: The Chain Rule

$$P(high\ winds\ tonight) = P(high) \times P(winds|\ high) \times P(tonigh|high, winds)$$

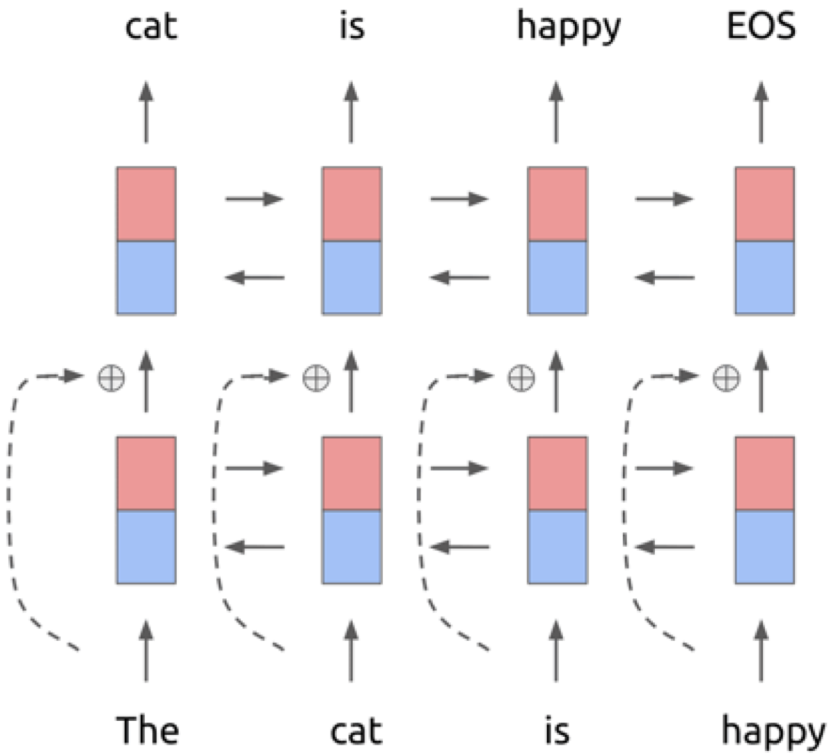
RNN Language Model



- Cats average 15 hours of sleep a day. <EOS>
  - P(sentence) = P(cats)P(average|cats)P(15|cats,average)…

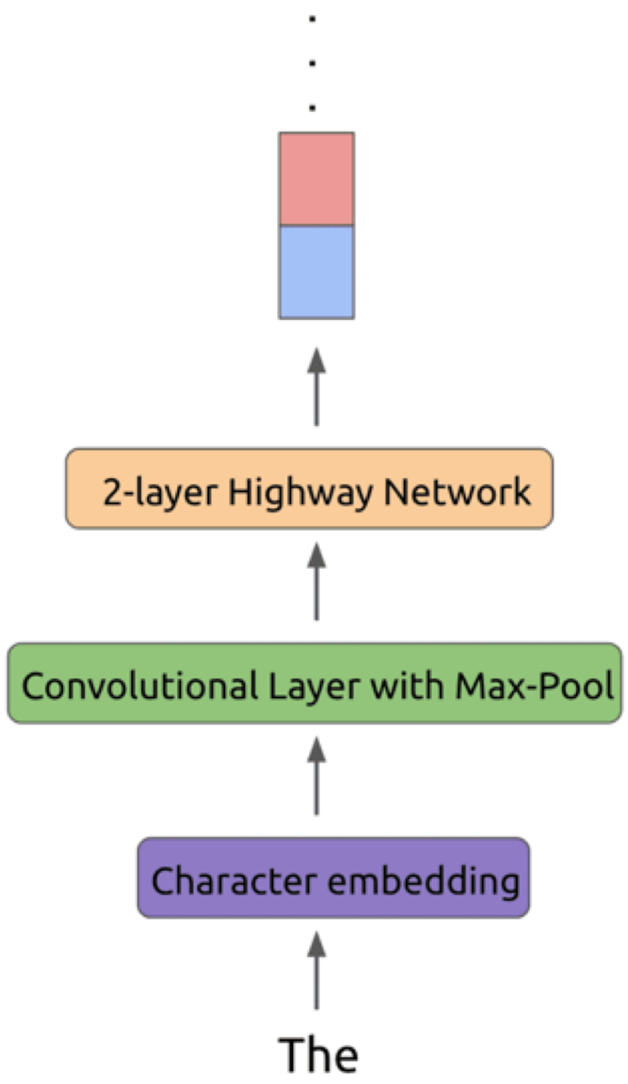
Embeddings from Language Models

- ELMo architecture trains a language model using a 2-layer bi-directional LSTM (biLMs)
- What input?
  - Traditional Neural Language Models use fixed - length word embedding.
    - One-hone encoding.
    - Word2Vec.
    - Glove.
    - Etc….
  - ELMo uses a more complex representation.

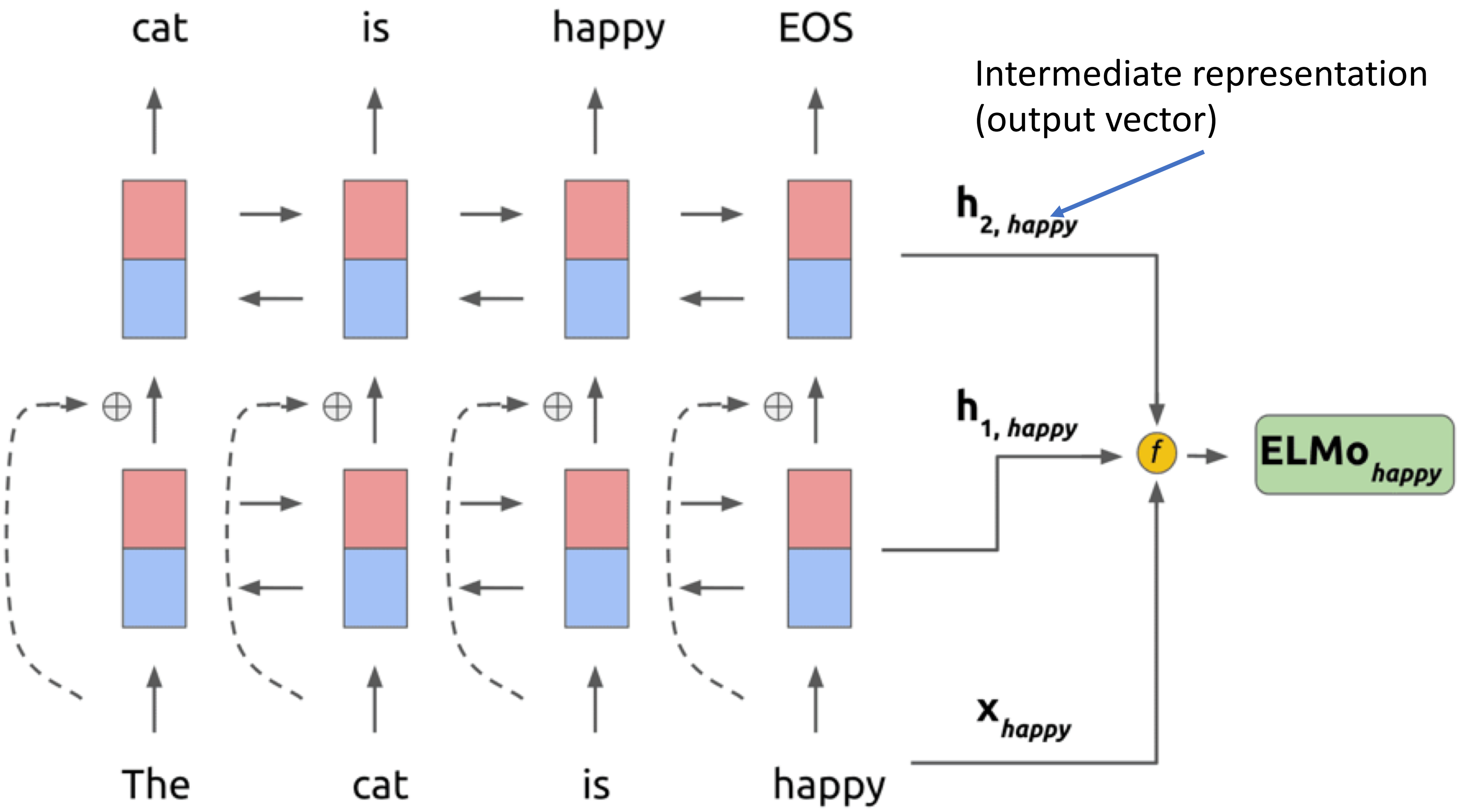


ELMo: What input?

- Transformations applied for each token before being provided to input of first LSTM layer.
- Pros of character embeddings:
  - It allows to pick up on morphological features that word-level embeddings could miss.
  - It ensures a valid representation even for out-of-vocabulary words.
  - It allows us to pick up on n-gram features that build more powerful representations.
  - The highway network layers allow for smoother information transfer through the input.



ELMo: Embeddings from Language Models



An example of combining the bidirectional hidden representations and word representation for "happy" to get an ELMo-specific representation. Note: here we omit visually showing the complex network for extracting the word representation that we described in the previous slide.

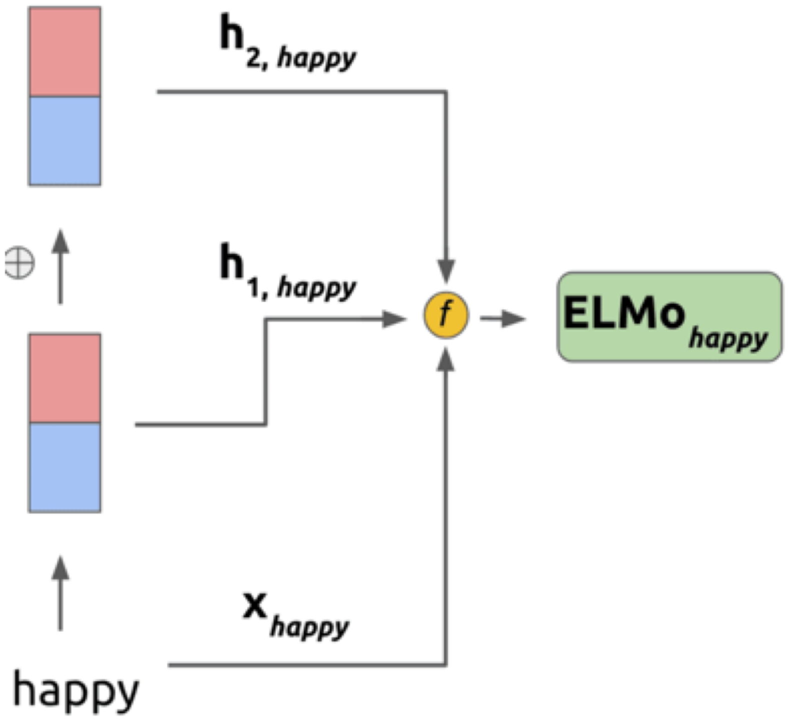
ELMo mathematical details

- The function ***f*** performs the following operation on word ***k*** of the input:

$$ELMo_k^{task} = \gamma_k \cdot (s_0^{task} \cdot x_k + s_1^{task} \cdot h_{1,k} + s_2^{task} \cdot h_{2,k})$$

  - Where s_i represents softmax-normalized weights.

- ELMo learns a separate representation for each task
  - Question answering, sentiment analysis, etc.



Difference to other methods

	Source	Nearest Neighbors
GloVe	***play***	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular ***play*** on Alusik 's grounder {. . . }	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway ***play*** for Garson {. . . }	{. . . } they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

- Nearest neighbors words to "play" using GloVe and the nearest neighbor sentences to "play" using ELMo.

Bibliography

- Mikolov, Tomas, et al. ”Efficient estimation of word representations in vector space.” arXiv preprint arXiv:1301.3781 (2013).
- Kottur, Satwik, et al. ”Visual Word2Vec (vis-w2v): Learning Visually Grounded Word Embeddings Using Abstract Scenes.” arXiv preprint arXiv:1511.07067 (2015).
- Lazaridou, Angeliki, Nghia The Pham, and Marco Baroni. ”Combining language and vision with a multimodal skip-gram model.” arXiv preprint arXiv:1501.02598 (2015).
- Rong, Xin. ”word2vec parameter learning explained.” arXiv preprint arXiv:1411.2738 (2014).
- Mikolov, Tomas, et al. ”Distributed representations of words and phrases and their compositionality.” Advances in neural information processing systems. 2013.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.
- Scott Deerwester et al. “Indexing by latent semantic analysis”. Journal of the American society for information science (1990).
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer.
- Matt Gardner and Joel Grus and Mark Neumann and Oyvind Tafjord and Pradeep Dasigi and Nelson F. Liu and Matthew Peters and Michael Schmitz and Luke S. Zettlemoyer. AllenNLP: A Deep Semantic Natural Language Processing Platform.