

Word embeddings

LSA, Word2Vec, Glove, ELMo

Dr. Reda Bouadjenek

Data-Driven Decision Making Lab (D3M)



UNIVERSITY OF
TORONTO

Vector Embedding of Words

- A word is represented as a **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 $\left\{ \begin{array}{l} \text{POTUS} \\ \text{The President} \\ \text{Trump} \end{array} \right\}$ called a press conference.
 - "POTUS: President of the United States."

Vector Embedding of Words

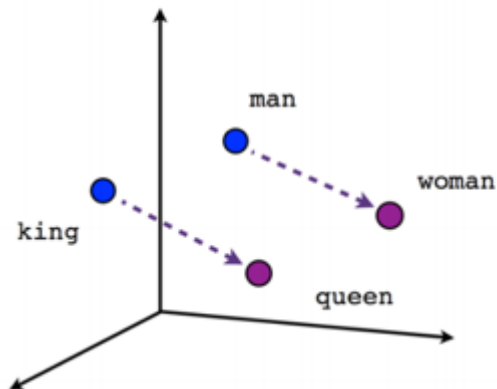
Traditional Method - Bag of Words Model

- 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 10000 words, and "Hello" is the 4th word in the dictionary, it would be represented by: 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
- Context information is not utilized.

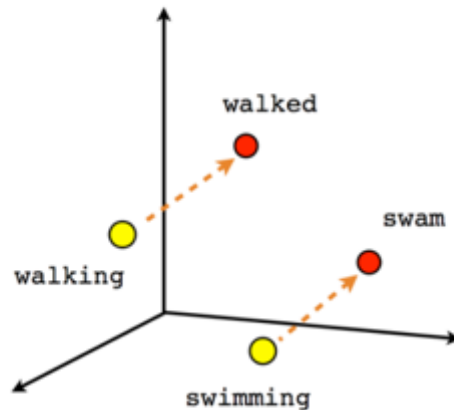
Word Embeddings

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

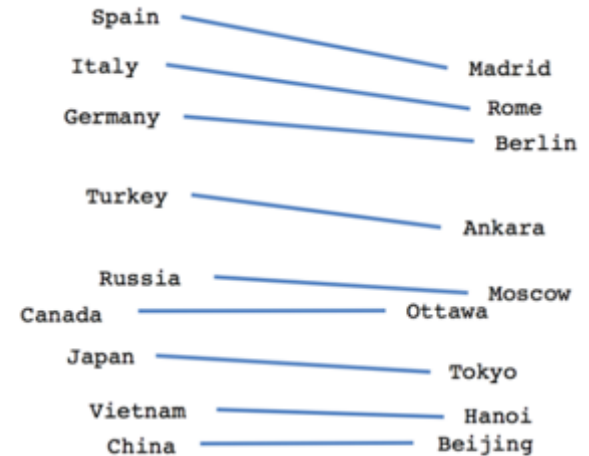
Example



Male-Female



Verb tense



Country-Capital

- $\text{vector}[\text{Queen}] \approx \text{vector}[\text{King}] - \text{vector}[\text{Man}] + \text{vector}[\text{Woman}]$
- $\text{vector}[\text{Paris}] \approx \text{vector}[\text{France}] - \text{vector}[\text{Italy}] + \text{vector}[\text{Rome}]$
 - This can be interpreted as “France is to Paris as Italy is to Rome”.

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.

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:

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

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

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

Applications of Word Vectors

- Word Similarity
- Machine Translation
- Part-of-Speech and Named Entity Recognition
- Relation Extraction
- Sentiment Analysis
- Co-reference Resolution
 - Chaining entity mentions across multiple documents - can we find and unify the multiple contexts in which mentions occurs?
- Clustering
 - Words in the same class naturally occur in similar contexts, and this feature vector can directly be used with any conventional clustering algorithms (K-Means, agglomerative, etc). Human doesn't have to waste time hand-picking useful word features to cluster on.
- Semantic Analysis of Documents
 - Build word distributions for various topics, etc.

Vector Embedding of Words

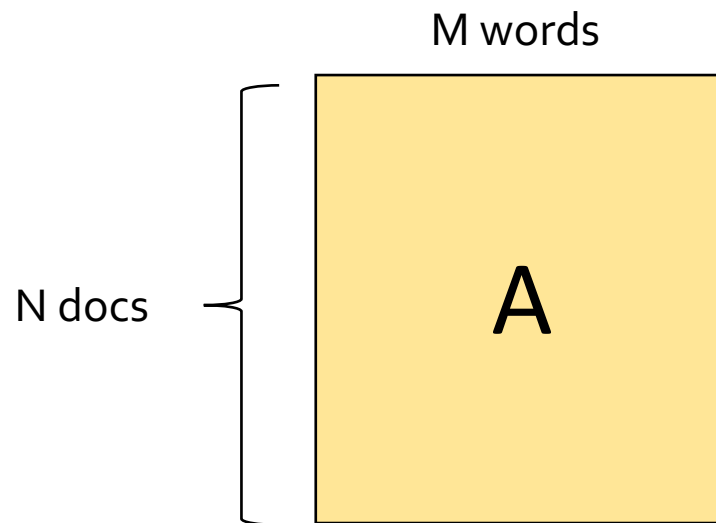
- Three main methods described in the talk :
 - Latent Semantic Analysis/Indexing (1988)
 - Term weighting-based model
 - Consider occurrences of terms at document level.
 - Word2Vec (2013)
 - Prediction-based model.
 - Consider occurrences of terms at context level.
 - GloVe (2014)
 - Count-based model.
 - 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** (1988).
 - i.e. given a matrix **A** encoding some documents: A_{ij} is the count* of word **j** in document **i**. Most entries are 0.



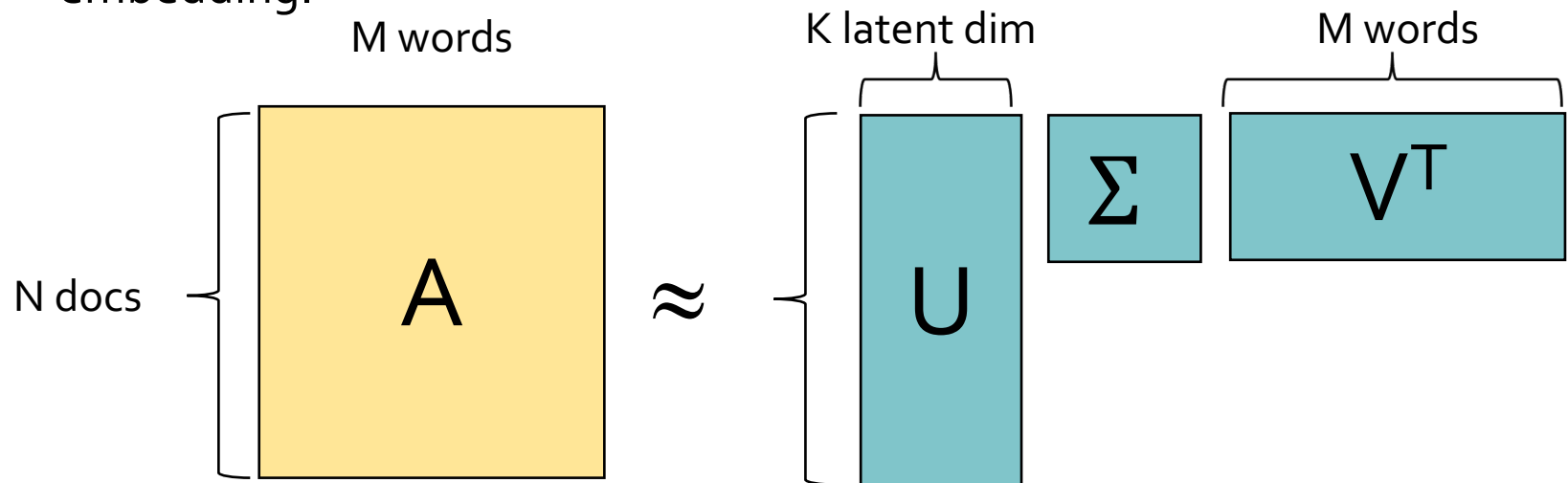
* Often tf-idf or other “squashing” functions of the count are used.

Embedding: Latent Semantic Analysis

- Low rank SVD decomposition:

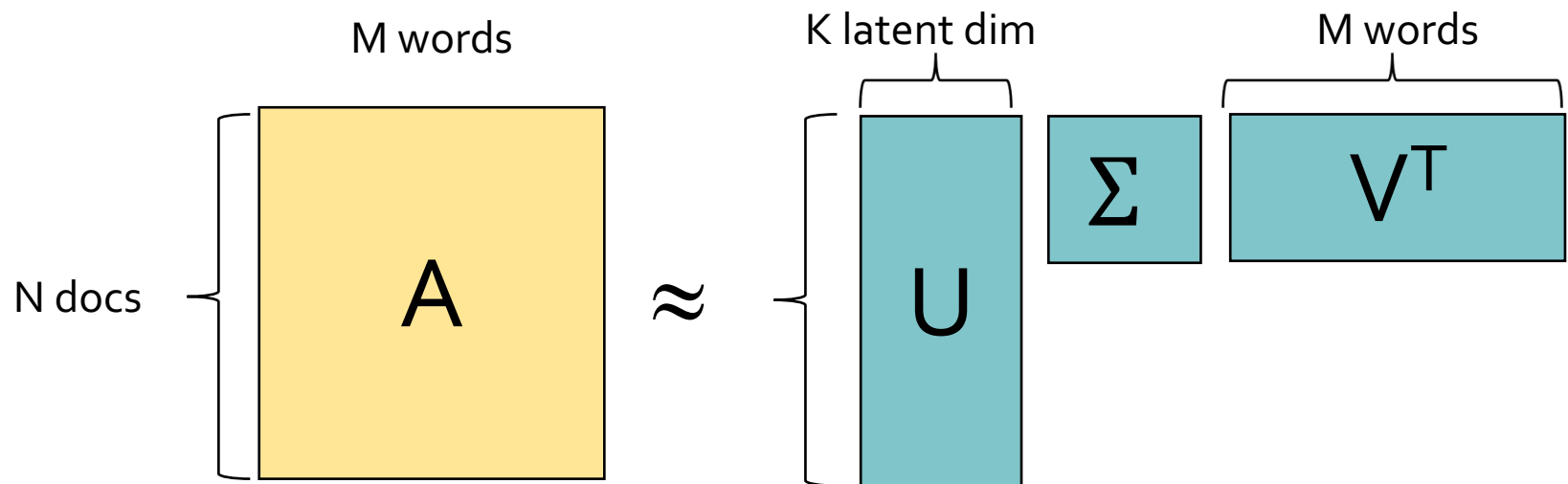
$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

- 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 \mathbf{w} (column of \mathbf{A}):
 - $\zeta = \mathbf{w}^T \times \mathbf{U}$ is the embedding (encoding) of the word \mathbf{w} in the latent space.
 - $\mathbf{w} \approx \mathbf{U} \times \zeta^T = \mathbf{U} \times (\mathbf{w}^T \times \mathbf{U})^T$ is the decoding of the word \mathbf{w} from its embedding.



Embedding: Latent Semantic Analysis

- $w \approx U \times \zeta^T = U \times (w^T \times U)^T$ 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

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: d_1 = "king brave man" , d_2 = "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]

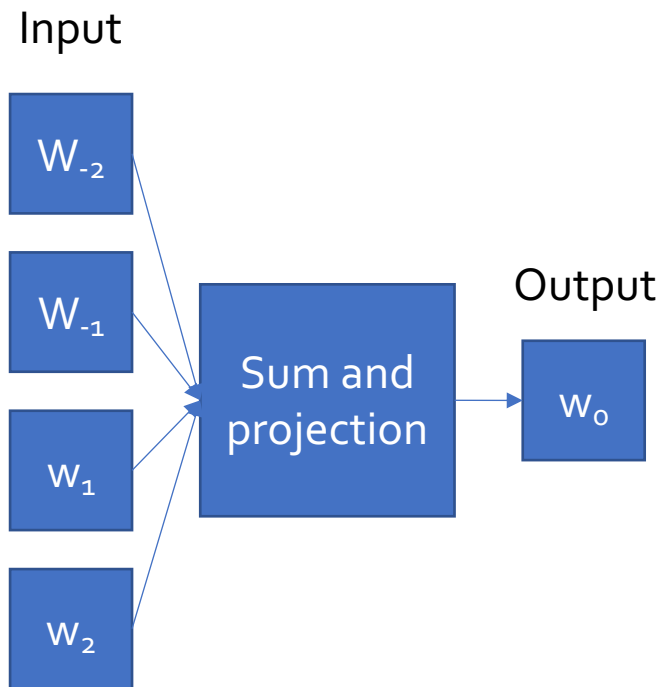
Word2Vec: Data generation (window size = 2)

- Example: d_1 = "king brave man" , d_2 = "queen beautiful women"

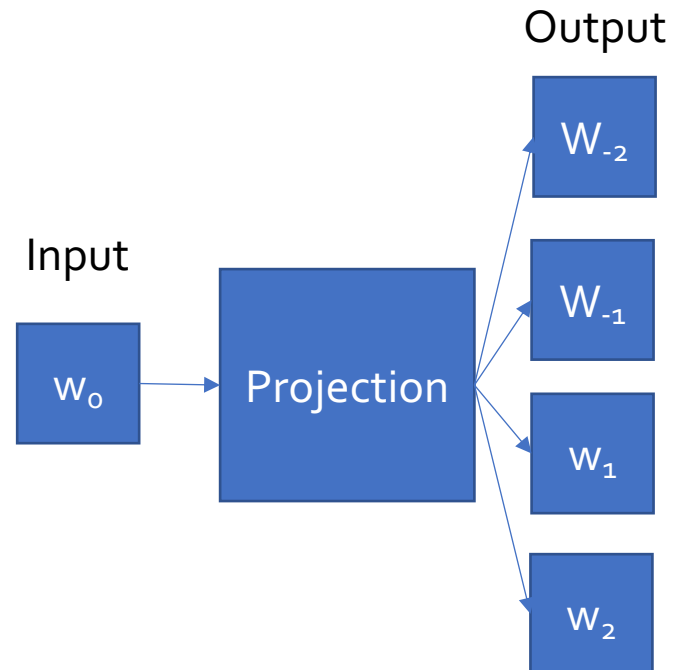
word	Word one hot encoding	neighbor	Neighbor one hot encoding
king	[1,0,0,0,0,0]	brave	[0,1,1,0,0,0]
		man	
brave	[0,1,0,0,0,0]	king	[1,0,1,0,0,0]
		man	
man	[0,0,1,0,0,0]	king	[1,1,0,0,0,0]
		brave	
queen	[0,0,0,1,0,0]	beautiful	[0,0,0,0,1,1]
		women	
beautiful	[0,0,0,0,1,0]	queen	[0,0,0,1,0,1]
		women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



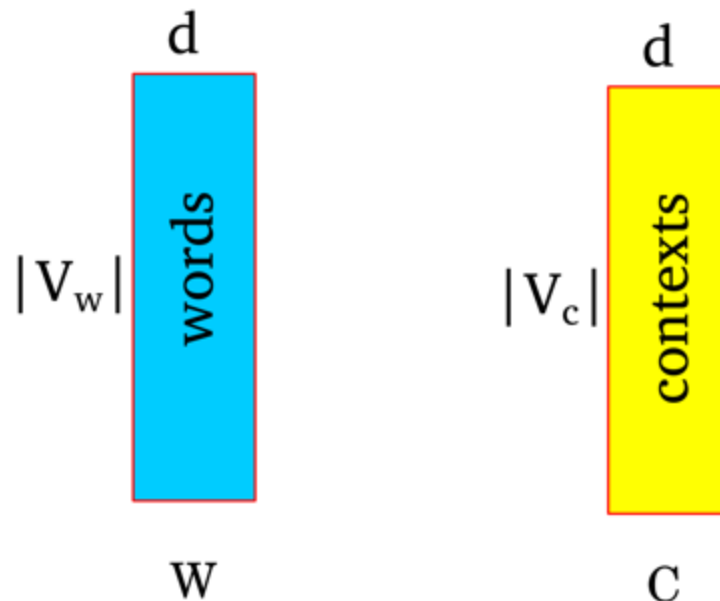
Skip-Ngram



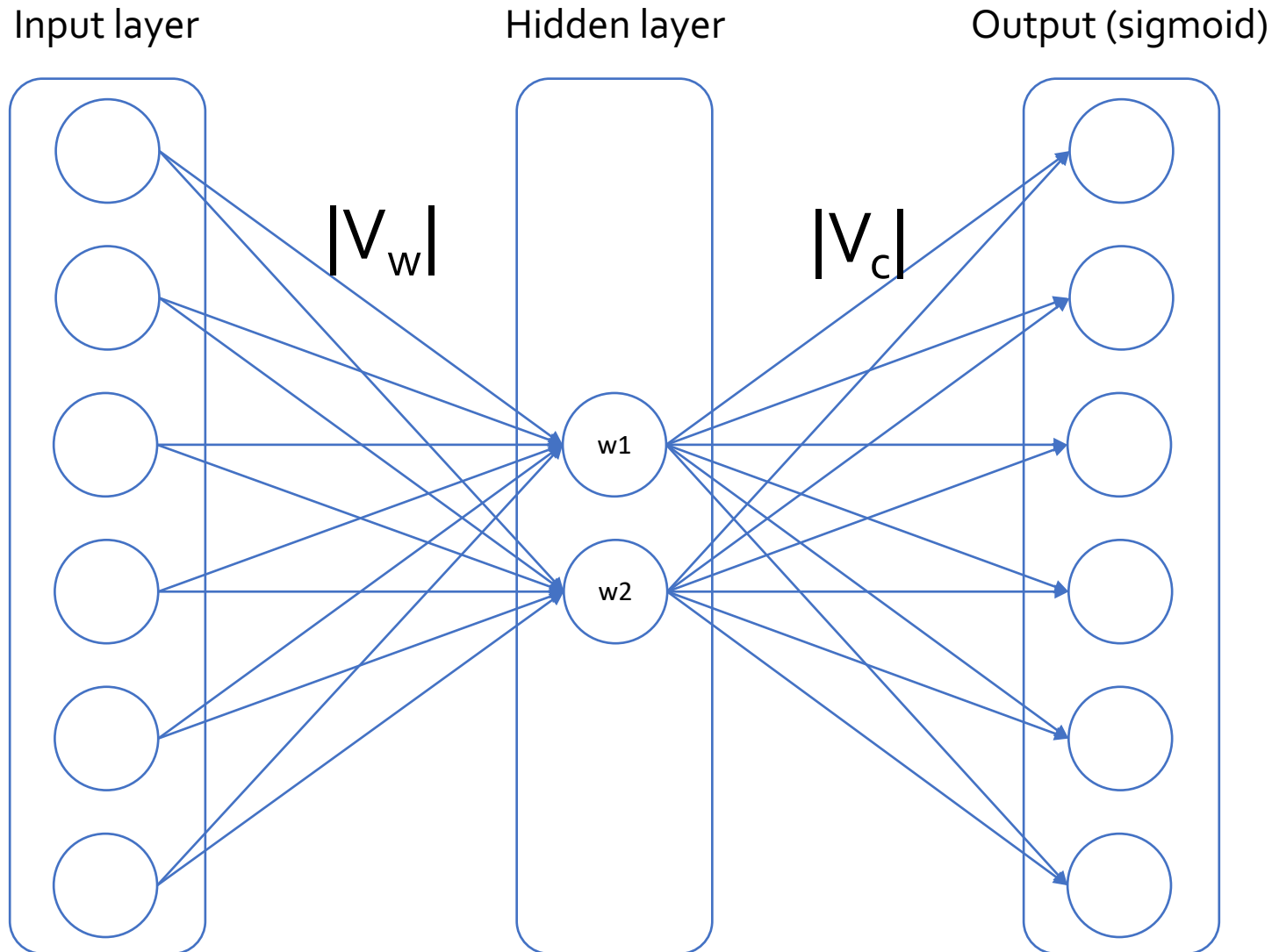
- Word2Vec is a predictive model.
- Will focus on Skip-Ngram model

How does word2Vec work?

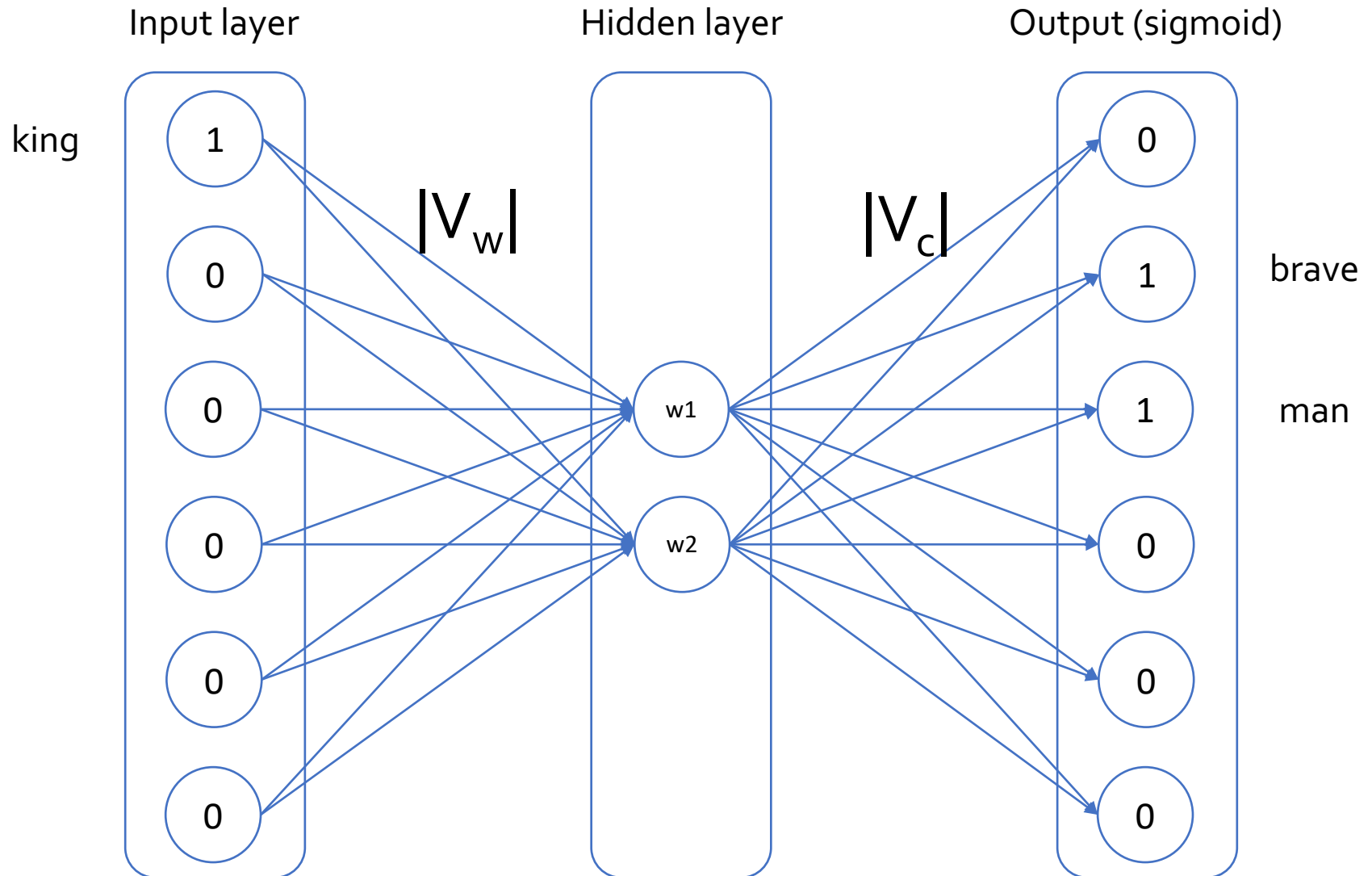
- Represent each word as a d dimensional vector.
- Represent each context as a d dimensional vector.
- Initialize all vectors to random weights.
- Arrange vectors in two matrices, W and C .



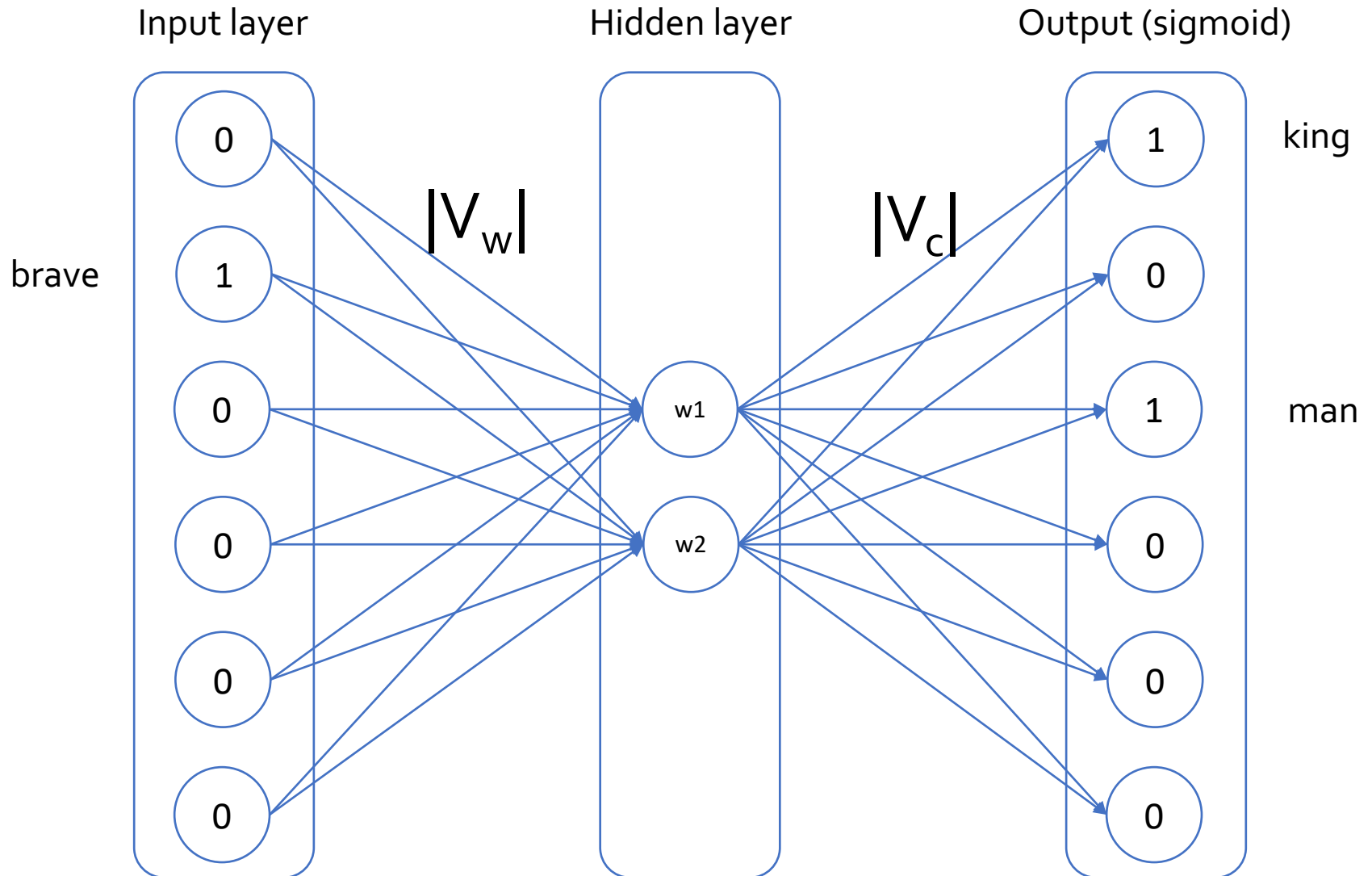
Word2Vec : Neural Network representation



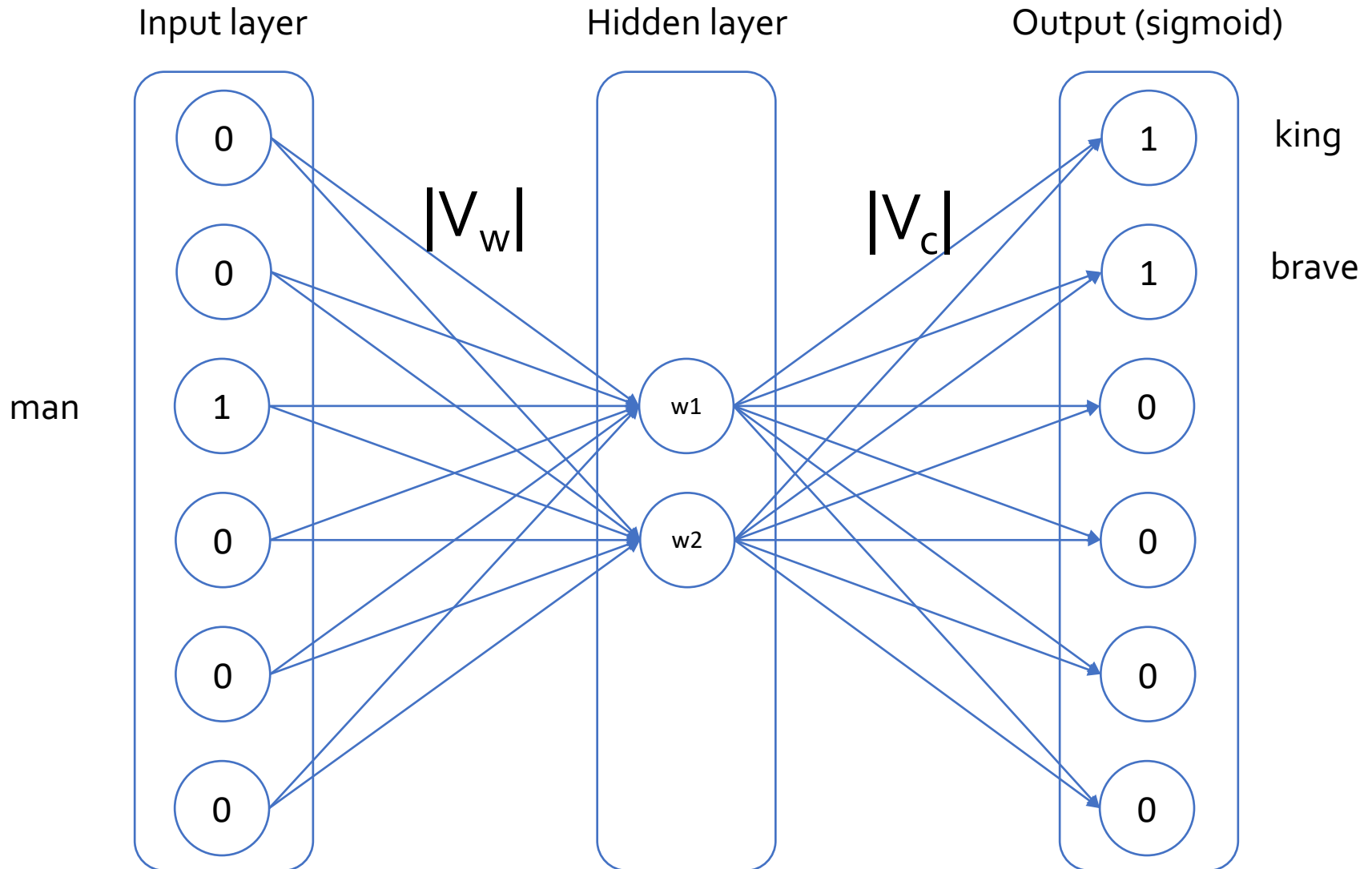
Word2Vec : Neural Network representation



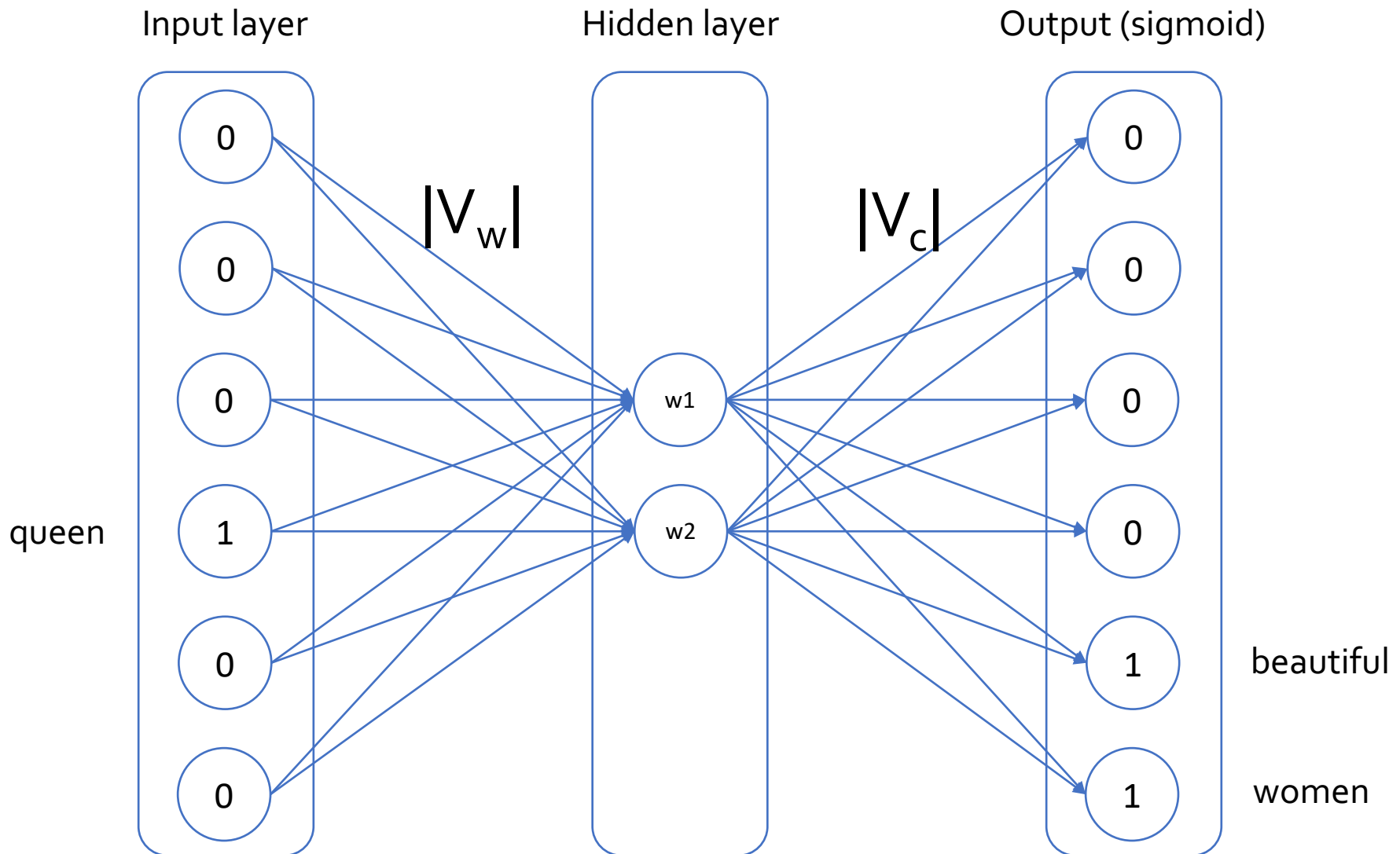
Word2Vec : Neural Network representation



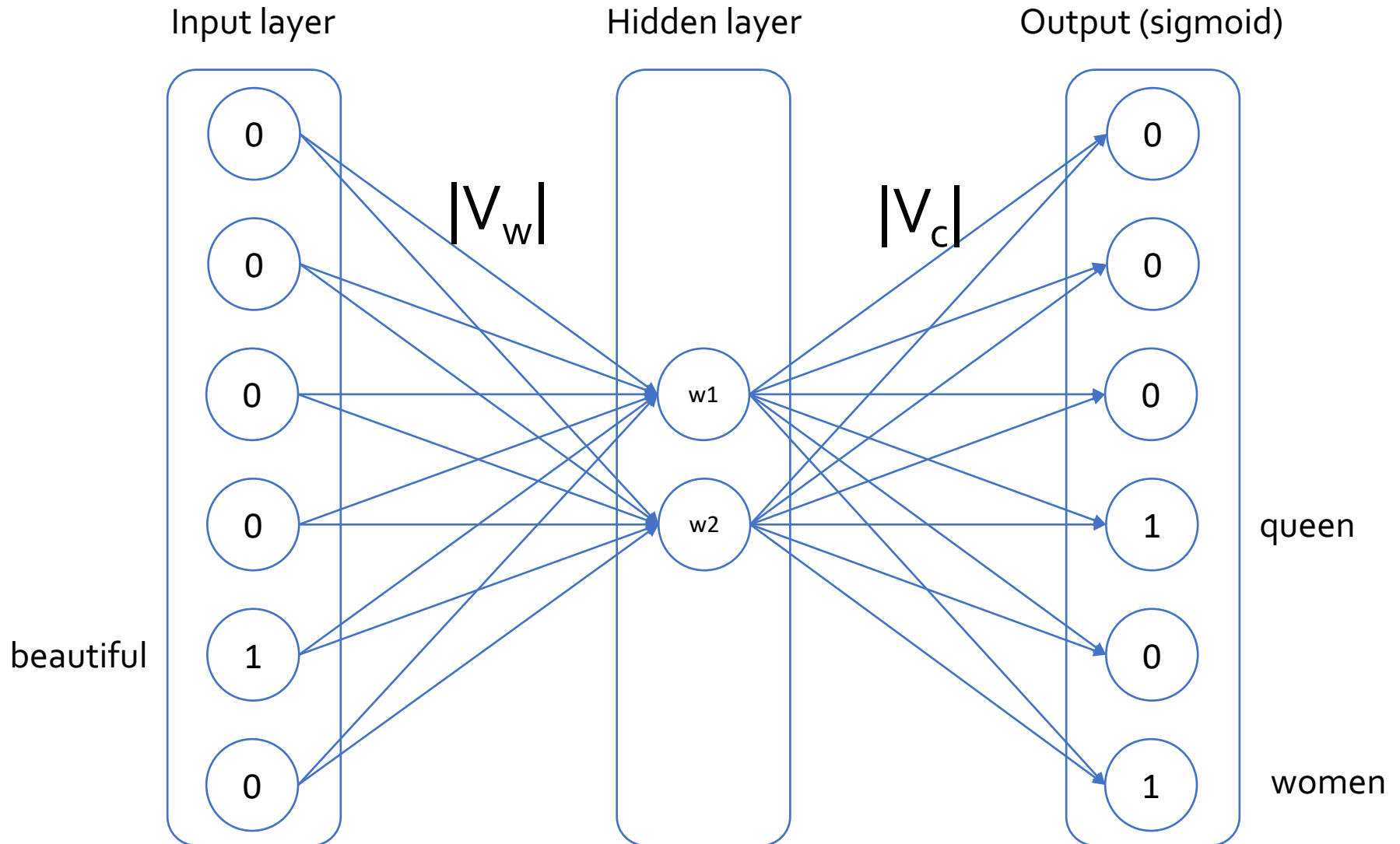
Word2Vec : Neural Network representation



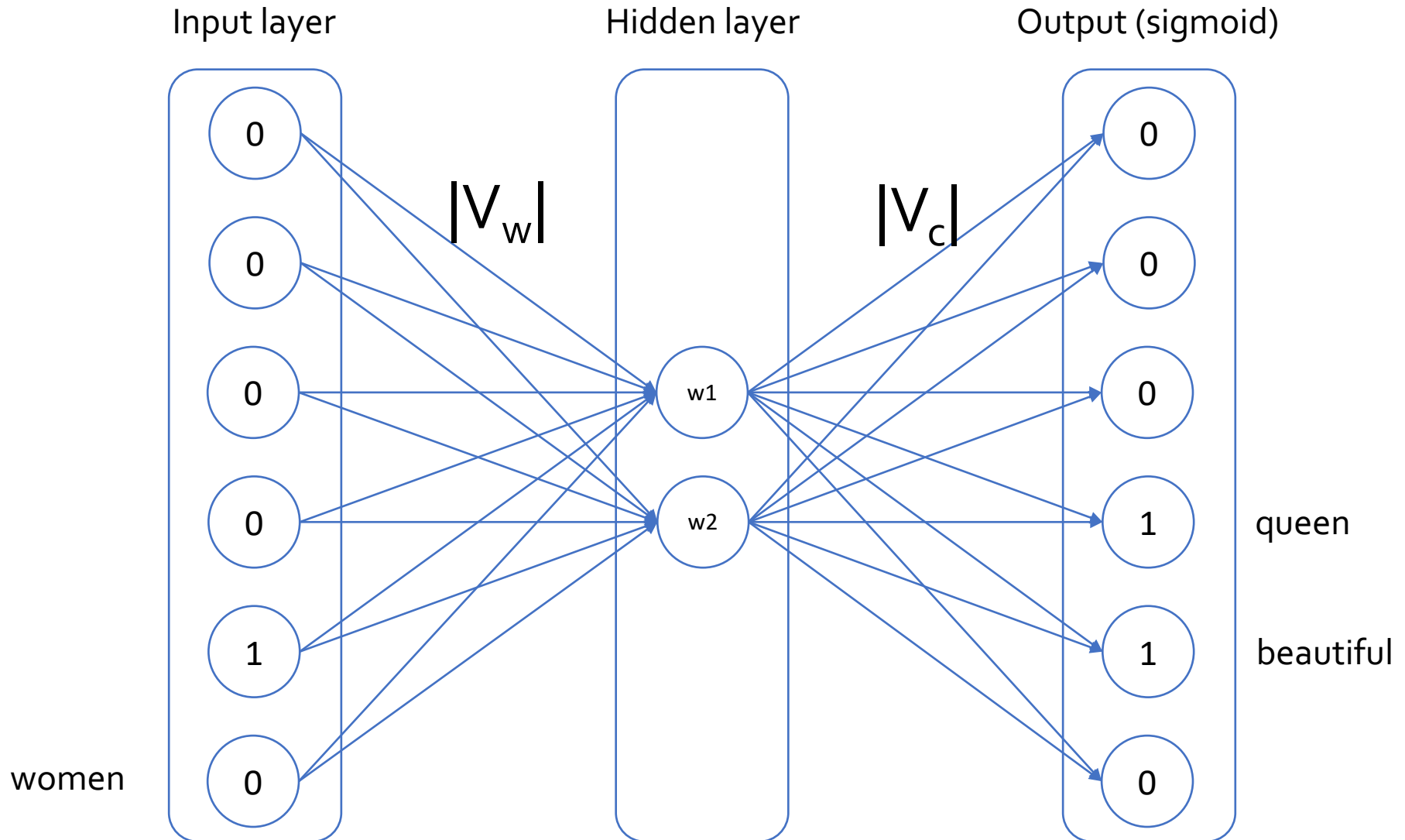
Word2Vec : Neural Network representation



Word2Vec : Neural Network representation



Word2Vec : Neural Network representation



Skip-Ngram: Training method

- The prediction problem is modeled using soft-max:

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

- Predict context words(s) **c**
- From focus word **w**
- Looks like logistic regression!
 - v_w are features and the evidence is v_c
- The objective function (in log space):

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

Skip-Ngram: Negative sampling

- The objective function (in log space):

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

- 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 $\sum_{\tilde{c} \in C} \exp(v_{\tilde{c}} \cdot v_w)$

- Mikolov et al. proposed the negative-sampling approach as a more efficient way of deriving word embeddings:

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

Skip-Ngram: Example

- While more text:

- Extract a word window:

A springer is[a cow or **heifer** close to calving]
 c_1 c_2 c_3 w c_4 c_5 c_6

- Try setting the vector values such that:

- $\sigma(w \cdot c_1) + \sigma(w \cdot c_2) + \sigma(w \cdot c_3) + \sigma(w \cdot c_4) + \sigma(w \cdot c_5) + \sigma(w \cdot c_6)$ is high!

- Create a corrupt example by choosing a random word w'

[a cow or **comet** close to calving]
 c_1 c_2 c_3 w' c_4 c_5 c_6

- Try setting the vector values such that:

- $\sigma(w' \cdot c_1) + \sigma(w' \cdot c_2) + \sigma(w' \cdot c_3) + \sigma(w' \cdot c_4) + \sigma(w' \cdot c_5) + \sigma(w' \cdot c_6)$ is low!

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.
 - The counts matrix is preprocessed by normalizing the counts and log-smoothing them.

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 \tilde{W} .
- W and \tilde{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 $\frac{W + \tilde{W}}{2}$ to get word vectors.

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 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

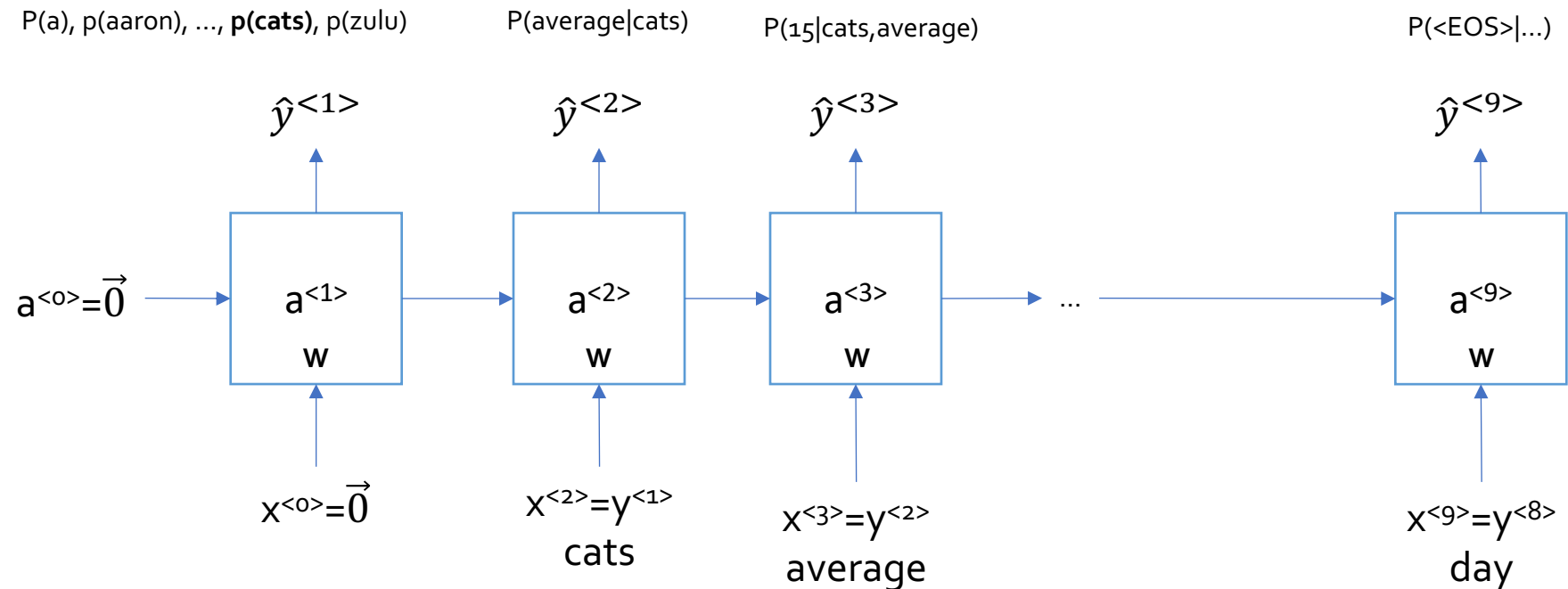
- Previous models (GloVe, Word2Vec, etc.) only have one representation per word
 - They can't capture these ambiguities.
- When you only have one representation, all levels of meaning are combined.
- Solution: have multiple levels of understanding.
 - ELMo: Embeddings from *Language Model* representations

What is language modelling?

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

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

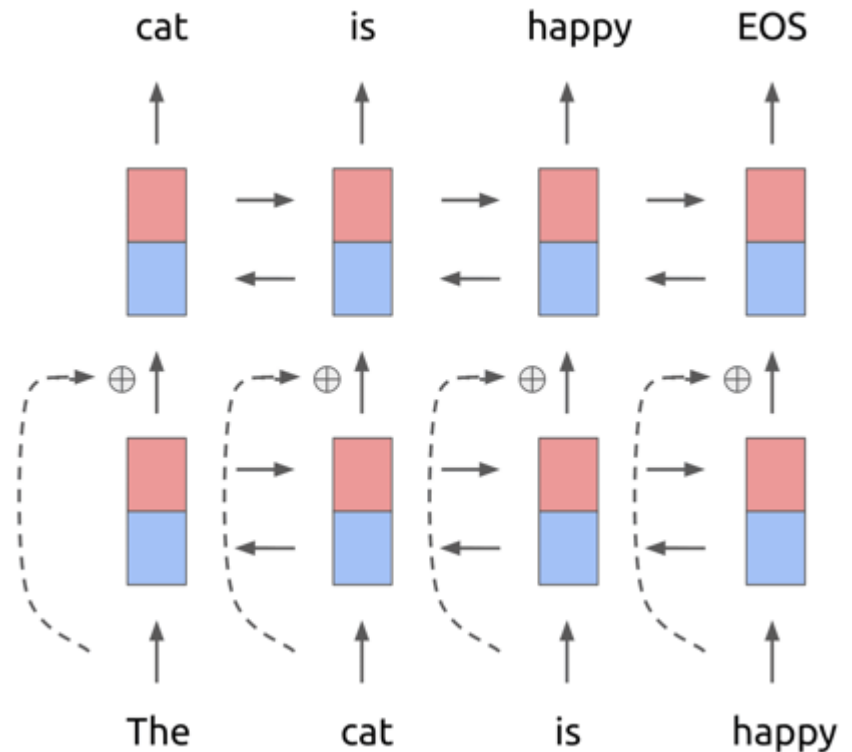
RNN Language Model



- Cats average 15 hours of sleep a day. $\langle \text{EOS} \rangle$
 - $P(\text{sentence}) = P(\text{cats})P(\text{average}|\text{cats})P(15|\text{cats}, \text{average})\dots$

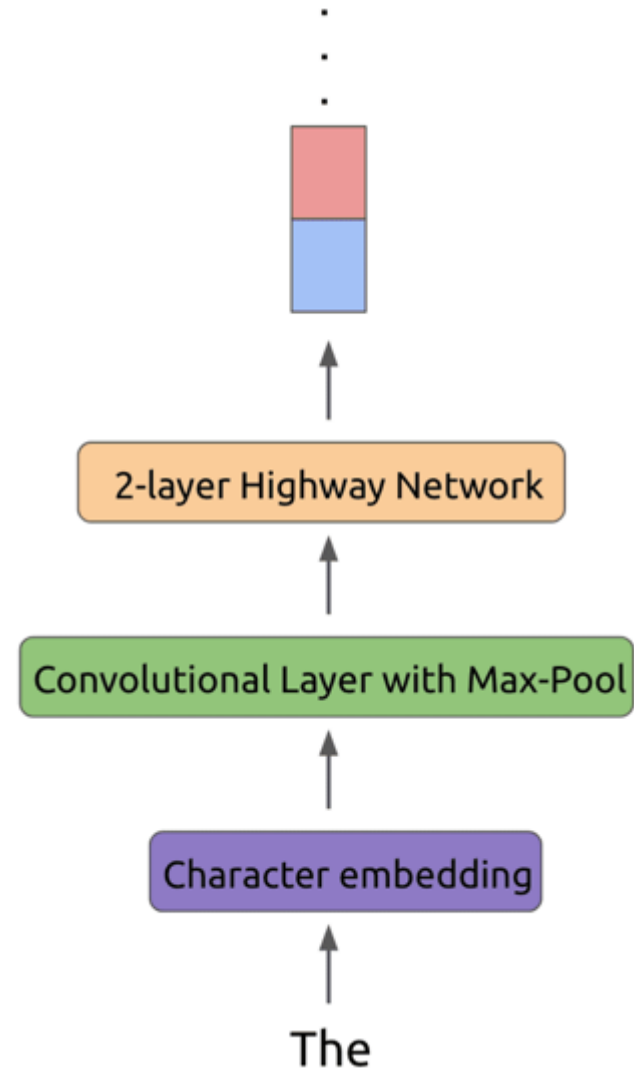
Embeddings from Language Models

- 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-hot encoding.
 - Word2Vec.
 - Glove.
 - Etc....
 - ELMo uses a more complex representation.

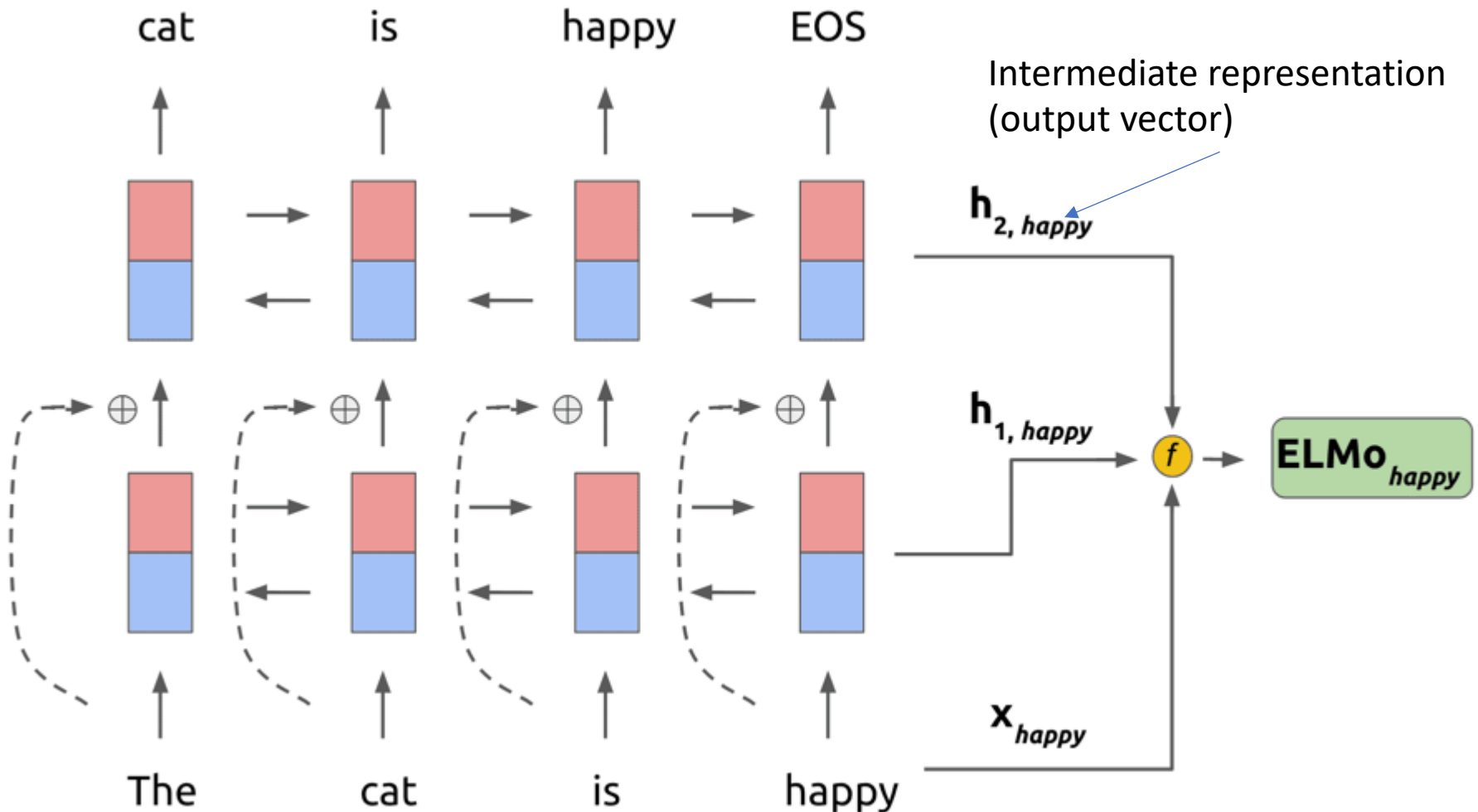


ELMo: What input?

- 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



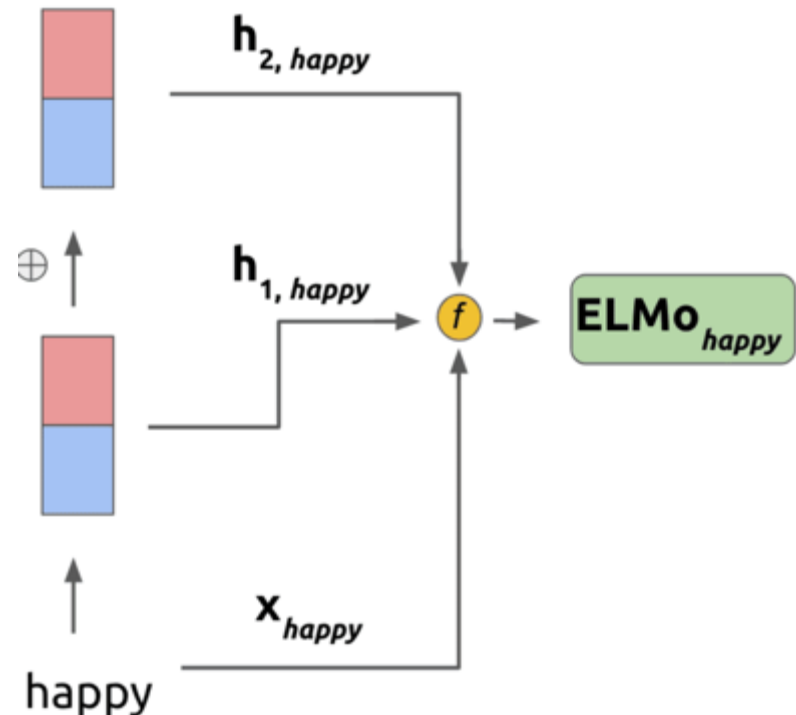
An example of combining the bidirectional hidden representations and word representation for "happy" to get an ELMo-specific representation. Note: here we omit visually showing the complex network for extracting the word representation that we described in the previous slide.

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	<u>play</u>	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular <u>play</u> 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 <u>play</u> 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.