Cornell Bowers C·IS
College of Computing and Information Science

Deep Learning

Week 02: Word Embeddings

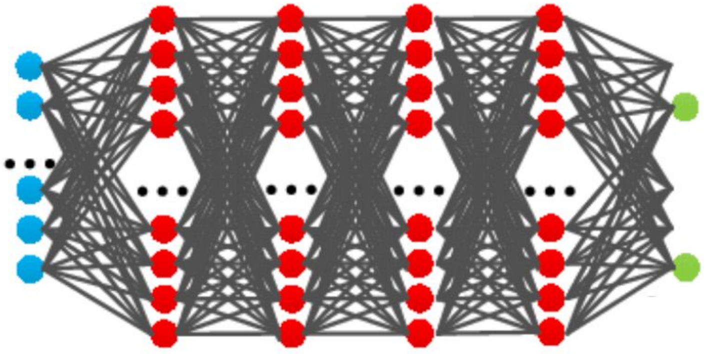
Varsha Kishore, Justin Lovelace, Vivian Chen, Anissa Dallmann, Wentao Guo,

Recap

- MLPs
- Loss functions
- Optimizers
- Regularization
- Architecture types (CNNs for images)
  - Today: Words

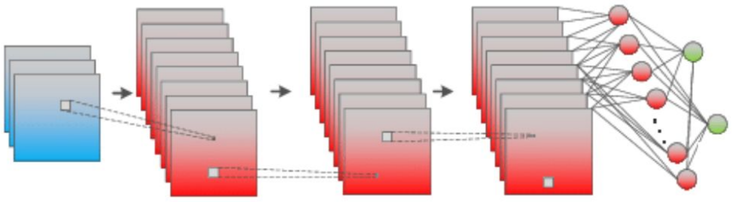
MLPs:

- Fully connected layers
- Require more parameters and computational resources
- Flexible and can handle various input types



CNNs:

- Convolutional layers with filters
- Designed specifically for structured input like images
- Inherently translation invariant due to shared weights
- Requires fewer parameters



https://www.researchgate.net/figure/The-architecture-of-MLP-and-CNN-MLP-is-consisted-of-fullyconnected-FC-layers-and-CNN_fig2_334489445

Cornell Bowers C·IS

How to handle text data?

Cornell Bowers C·IS

MI What is an easy to make pasta recipe that features garlic and red sauce?

Here is a simple pasta recipe that features garlic and red sauce:

Ingredients:

- 1 pound pasta (such as spaghetti or penne)
- 1 tablespoon olive oil
- 2 cloves garlic, minced
- 1 (28-ounce) can crushed tomatoes
- 1/4 teaspoon salt
- 1/4 teaspoon black pepper
- 1/4 teaspoon red pepper flakes (optional)
- 1/2 cup grated Parmesan cheese
- Fresh parsley or basil, chopped, for garnish (optional)

Google

san f

san f**rancisco weather**
san f**rancisco**
san f**rancisco giants**
san f**ernando valley**
san f**rancisco state university**
san f**rancisco hotels**
san f**rancisco 49ers**
san f**ernando**
san f**ernando mission**
san f**rancisco zip code**

Google Search | I'm Feeling Lucky

Spanish - detected ⇄ English

¿Cómo estás?

How are you?

Verified

Open in Google Translate • Feedback

```
// write a binary search algorithm
const binarySearch = (arr, target) => {
  let left = 0;
  let right = arr.length - 1;
  let middle = Math.floor((left + right) / 2);
  while (arr[middle] !== target && left <= right) {
    if (target < arr[middle]) {
      right = middle - 1;
    } else {
      left = middle + 1;
    }
    middle = Math.floor((left + right) / 2);
  }
  return arr[middle] === target ? middle : -1;
}
```

I don't know how to parallel park.
I'm taking my dog for a walk at the park.

Homonyms

We ate outside and swam in the lake all week.
We ate outside and in the lake all week.

Typos

Biden speaks to the media in Illinois.
The president greets the press in Chicago.

Paraphrases/ Synonyms

Although interchangeable, the body pieces on the 2 cars are not similar.
Although similar, the body pieces are not interchangeable on the 2 cars.

Word order

Cornell Bowers C·IS

Language Modeling: predict the next word

Assign probabilities to text.

Given a sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$, we want to **maximize** $P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$.

$$P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) = P(\mathbf{x}_1)P(\mathbf{x}_2|\mathbf{x}_1)P(\mathbf{x}_3|\mathbf{x}_1, \mathbf{x}_2)P(\mathbf{x}_4|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)\ldots P(\mathbf{x}_T|\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_{T-1})$$

P(I like cats because they look cute) = P(I) P(like | I) P(cats | I like) P(because | I like cats) P(they | I like cats because)

P(look | I like cats because they) P(cute | I like cats because they look)

Predict the next word given current text!

[C. E. Shannon. Prediction and entropy of printed English. Bell Sys. Tech. Jour., Vol. 30, pp. 51-64, 1951.]

n-Gram Language Model

n-Gram: chunk of *n* consecutive words

Uni-gram: “I” “like” “cats” “because” “they” “look” “cute”

Bi-gram: “I like” “like cats” “cats because” “because they” …

Tri-gram: “I like cats” “like cats because” “cats because they” …

Count the frequency of each *n*-grams and predict next word!

Assume each word only depends on previous *n - 1* words.

$$P(\mathbf{x}_t|\mathbf{x}_1,\ldots,\mathbf{x}_{t-1}) = P(\mathbf{x}_t|\mathbf{x}_{t-n+1},\ldots,\mathbf{x}_{t-1})$$
$$= \frac{\mathrm{count}(\mathbf{x}_{t-n+1},\ldots,\mathbf{x}_{t-1},\mathbf{x}_t)}{\mathrm{count}(\mathbf{x}_{t-n+1},\ldots,\mathbf{x}_{t-1})}$$

In *bi-gram* LM

P(I like cats as they look cute) = P(I) P(like | I) P(cats | like) P(because | cats) P(they | because) P(look | they) P(cute | look)

Discuss:

Do you want to have a large n or a small n in a n-gram model?

What is special about this sentence by Noam Chomsky:

“Colorless green ideas sleep furiously.”

Cornell Bowers C·IS

Tokenization

How big should my vocabulary be?

Should I use words as my vocabulary or characters?

Byte-Pair Encoding

Cornell Bowers C·IS

Byte-Pair Tokenization

https://tinyurl.com/CS4782BytepairDemo

Questions:

- How does the Byte-Pair Tokenization Algorithm work?
- What changes when you switch to WordPiece?
- Why might WordPiece be preferred over Byte-Pair?
- What else would you improve on both algorithms?

Bag of Words (to represent documents)

Drawbacks:

- High dimensionality
- No semantic information

https://koushik1102.medium.com/nlp-bag-of-words-and-tf-idf-explained-fd1f49dce7c4

Document similarity?

document 1

Obama
speaks
to
the
media
in
Illinois

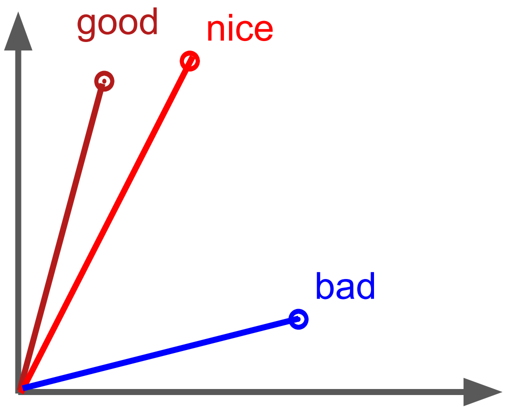
Documents have no words in common.
How can we quantify that they convey similar meanings?
(Assume B. Obama is president.)

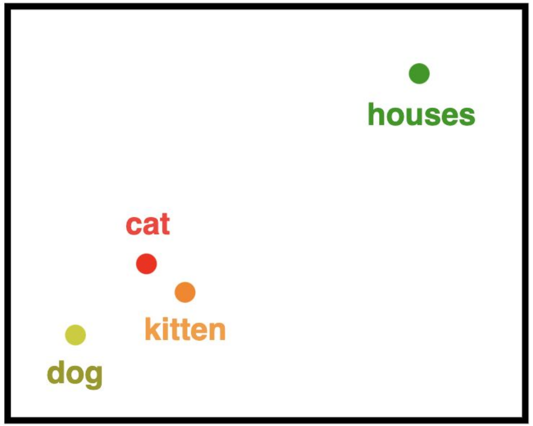
document 2

The
President
greets
the
press
in
Chicago

Semantic similarity

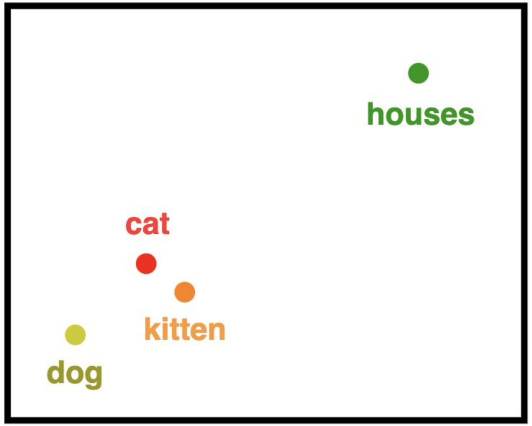
- Motivation
  - Put words into vectors so we can measure the similarity between words
  - Use cosine similarity





Why Do We Need Word Embeddings?

- Why Do We Need Word Embeddings?
  - Numerical Input
  - Shows Similarity and Distance

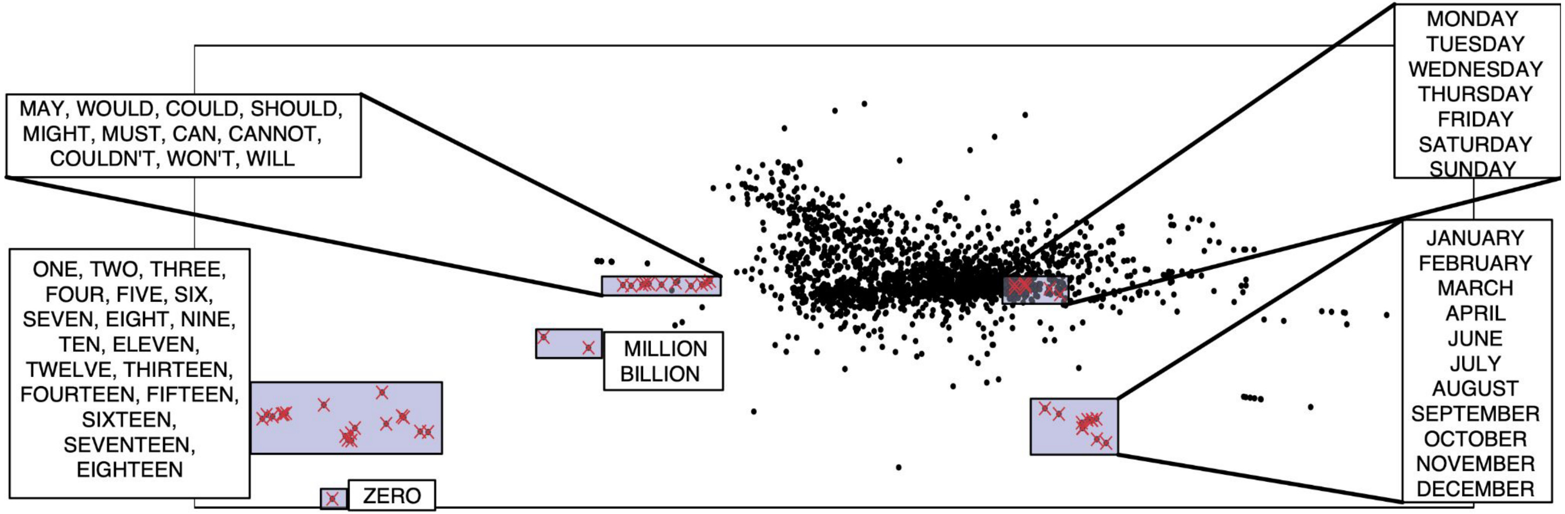


	living being	feline	human	gender	royalty	verb	plural
cat	0.6	0.9	0.1	0.4	-0.7	-0.3	-0.2
kitten	0.5	0.8	-0.1	0.2	-0.6	-0.5	-0.1
dog	0.7	-0.1	0.4	0.3	-0.4	-0.1	-0.3
houses	-0.8	-0.4	-0.5	0.1	-0.9	0.3	0.8
man	0.6	-0.2	0.8	0.9	-0.1	-0.9	-0.7
woman	0.7	0.3	0.9	-0.7	0.1	-0.5	-0.4
king	0.5	-0.4	0.7	0.8	0.9	-0.7	-0.6
queen	0.8	-0.1	0.8	-0.9	0.8	-0.5	-0.9

embedding using features of words

What are word embeddings

- What are Word Embeddings?
  - vector representations of words that capture semantic relationships
  - Latent Semantic Analysis / Indexing [S. Deerwester et al 1988]



[Blitzer et al. Neurips, 2004]

What is **Underberg**?

Suppose you see these sentences:

- I love drinking **Underberg** after a meal.
- I find **Underberg** is quite strong.
- A few bottles of **Underberg** make me very drunk.

Word2Vec:

- We want vectors for words so that the context of a word can suggest the vector of this word, and vice versa
- Idea: **Similar words appear in similar contexts**

Cornell Bowers C·IS

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

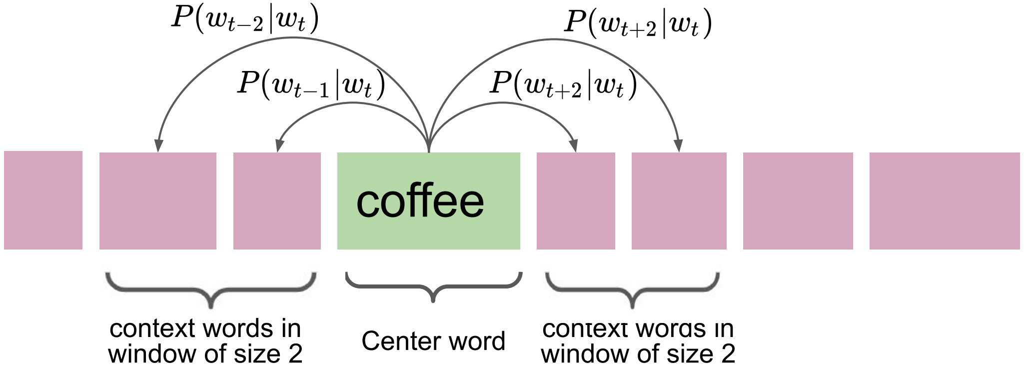
Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

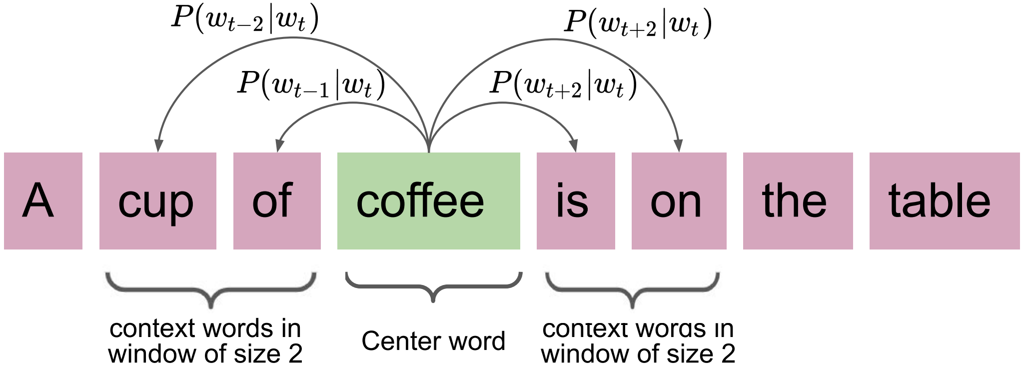
Word2Vec - Training

SkipGram - Predict context from target



Word2Vec - Training

SkipGram - Predict context from target



SkipGram - Training samples

A	cup	of	coffee	is	on	the	table

→

(coffee, cup)

(coffee, of)

(coffee, is)

(coffee, on)

Skip-gram Objective

Goal: Maximize the probability of context words given center words

$$\max_{W,W'} \sum_{t=c+1}^{T-c} \sum_{\substack{-c \le j \le c \\ j \ne 0}} \log P(x_{t+j} \mid x_t) \qquad \text{where} \qquad P(x_o \mid x_i) = \frac{\exp\left({w'_o}^\top w_i\right)}{\sum_{j=1}^{|V|} \exp\left({w'_j}^\top w_i\right)}$$

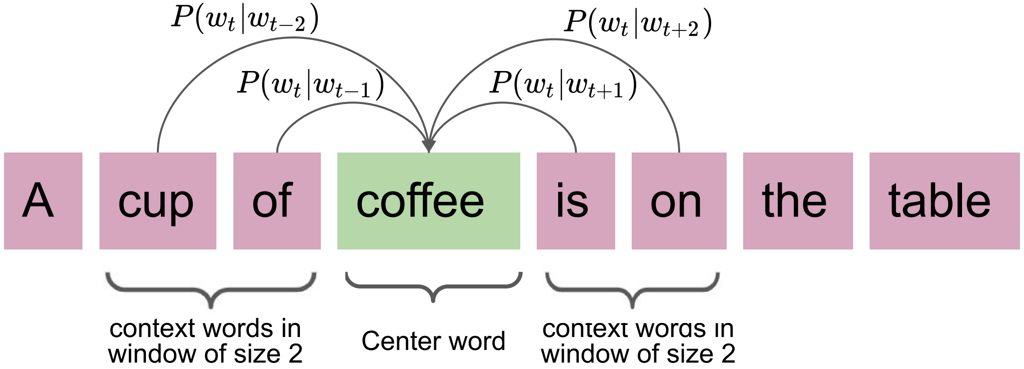
Two embedding matrices:

- $W \in \mathbb{R}^{d \times |V|}$ — center embeddings (columns w_i)
- $W' \in \mathbb{R}^{d \times |V|}$ — context embeddings (columns w'_i)

Cornell Bowers C·IS

Word2Vec - Training

Continuous Bag of Words (CBOW) - predict target from context



CBOW Objective

Goal: Maximize the probability of the center word given its context

$$\max_{W,W'} \sum_{t=c+1}^{T-c} \log P(x_t \mid x_{t-c}, \ldots, x_{t+c}) \qquad \text{where} \qquad P(x_o \mid \bar{w}) = \frac{\exp\left(w_o'^{\top} \bar{w}\right)}{\sum_{j=1}^{|V|} \exp\left(w_j'^{\top} \bar{w}\right)}$$

Context representation:

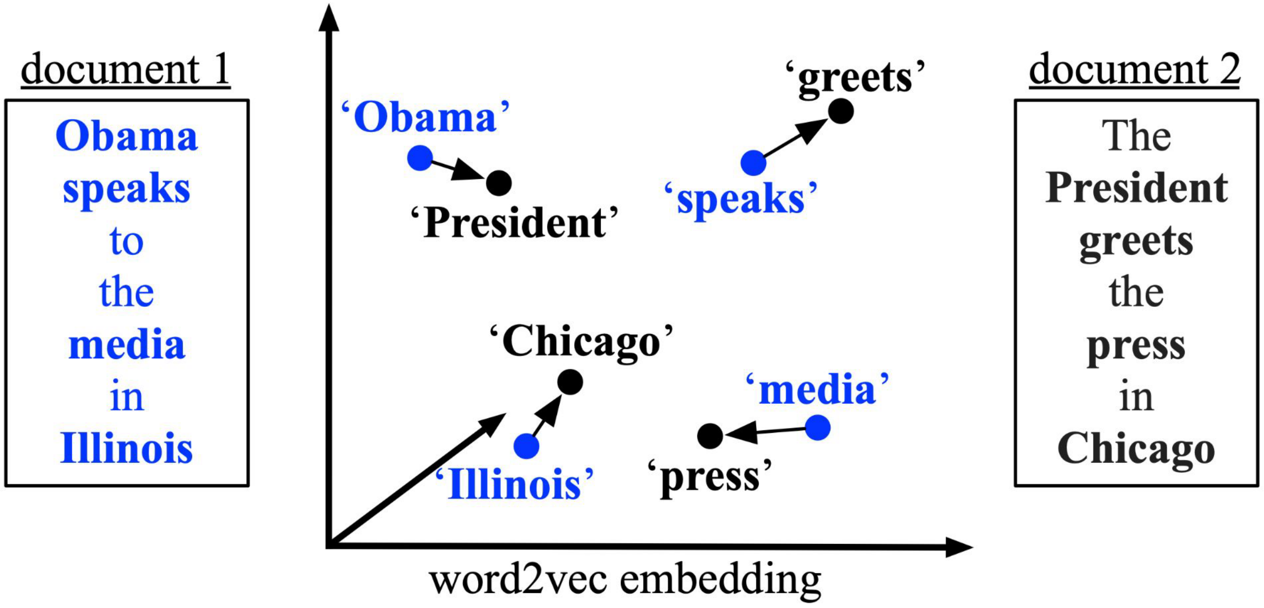
- Average the context word embeddings:

$$\bar{w} = \frac{1}{2c} \sum_{\substack{-c \le j \le c \\ j \neq 0}} w_{t+j}$$

Cornell Bowers C·IS

Word Mover's Distance [Kusner et al., 2015]

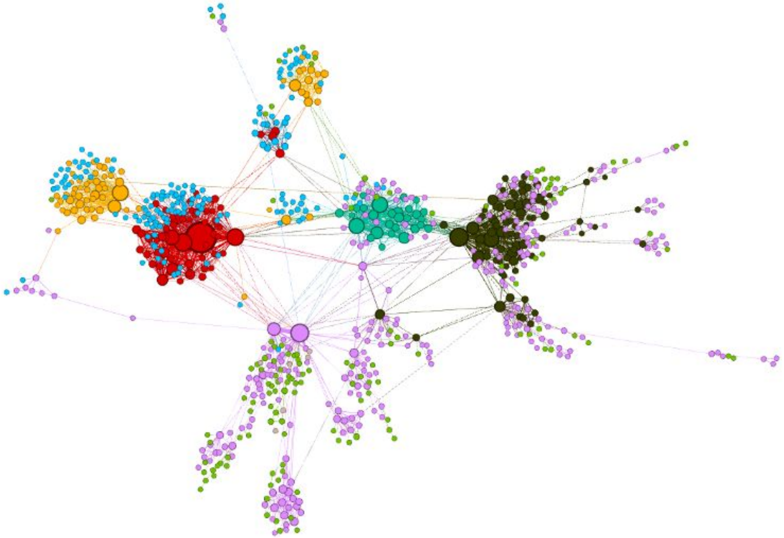
Measure similarity between documents as the minimum travel distance to match all words from one document to those of the other in word2vec space.



Cornell Bowers C·IS

X 2 vec

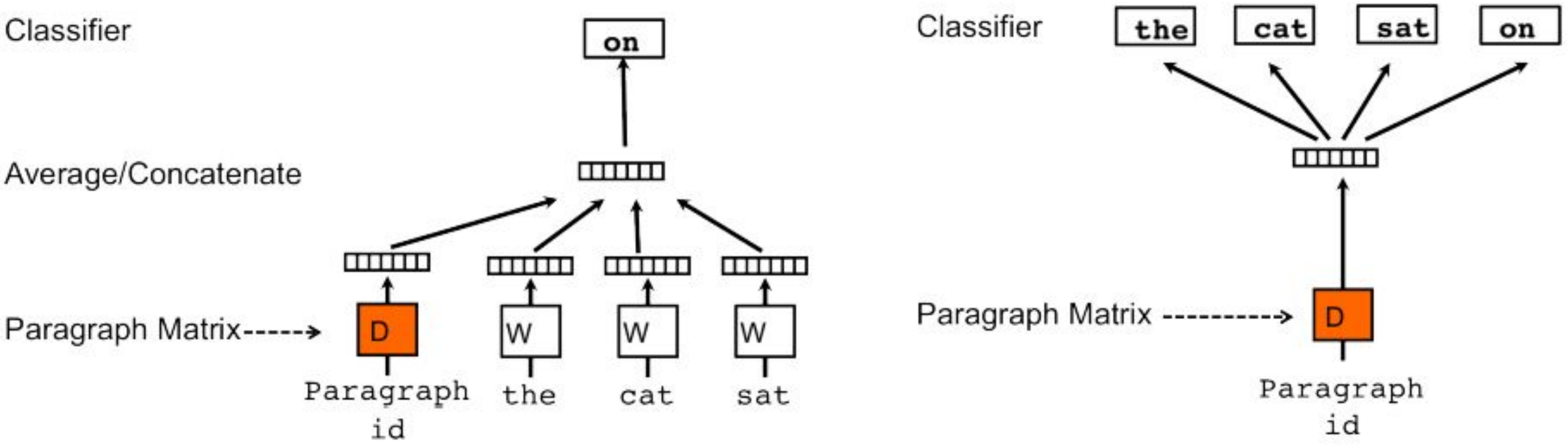
- Generate vector representations (embeddings) for various data types
- Examples:
  - Word2Vec
  - Doc2Vec
  - Node2Vec
  - Item2Vec
  - Sent2Vec



Cornell Bowers C·IS

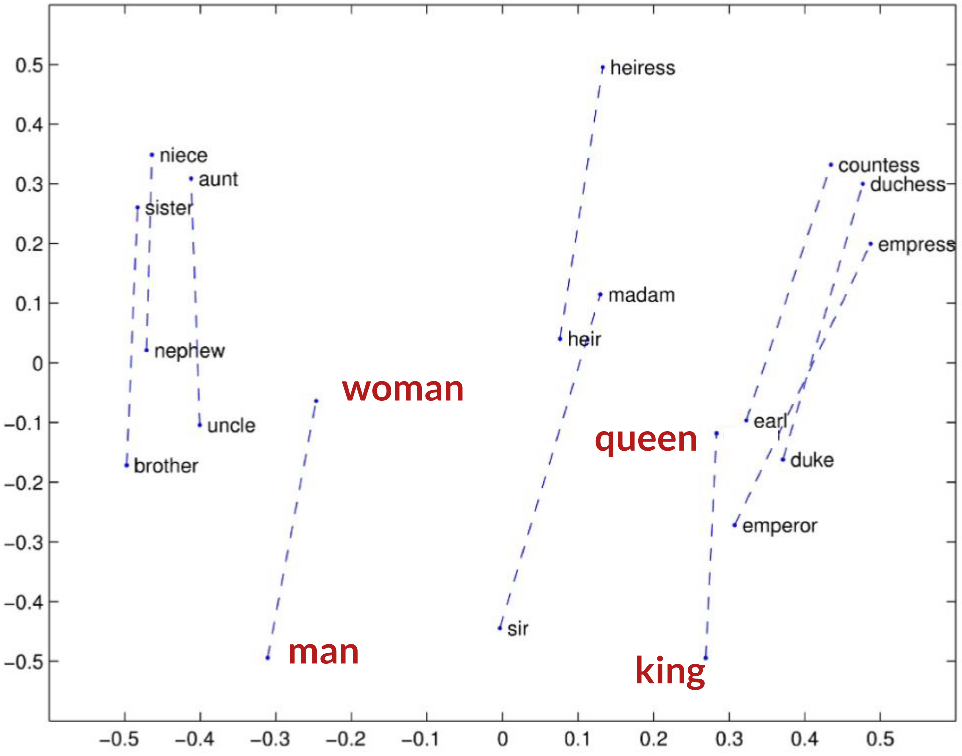
Doc2Vec

- A vector to represent a paragraph, regardless of length
  - embeddings for paragraph and words
  - Applications: Document classification, sentiment analysis, recommendation systems, and information retrieval



Cornell Bowers C·IS

In vector space…



Word embeddings capture societal biases

Extreme *she*	Extreme *he*
1. homemaker	1. maestro
2. nurse	2. skipper
3. receptionist	3. protege
4. librarian	4. philosopher
5. socialite	5. captain
6. hairdresser	6. architect
7. nanny	7. financier
8. bookkeeper	8. warrior
9. stylist	9. broadcaster
10. housekeeper	10. magician

Gender stereotype *she*-*he* analogies

sewing-carpentry	registered nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	lovely-brilliant

Gender appropriate *she*-*he* analogies

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

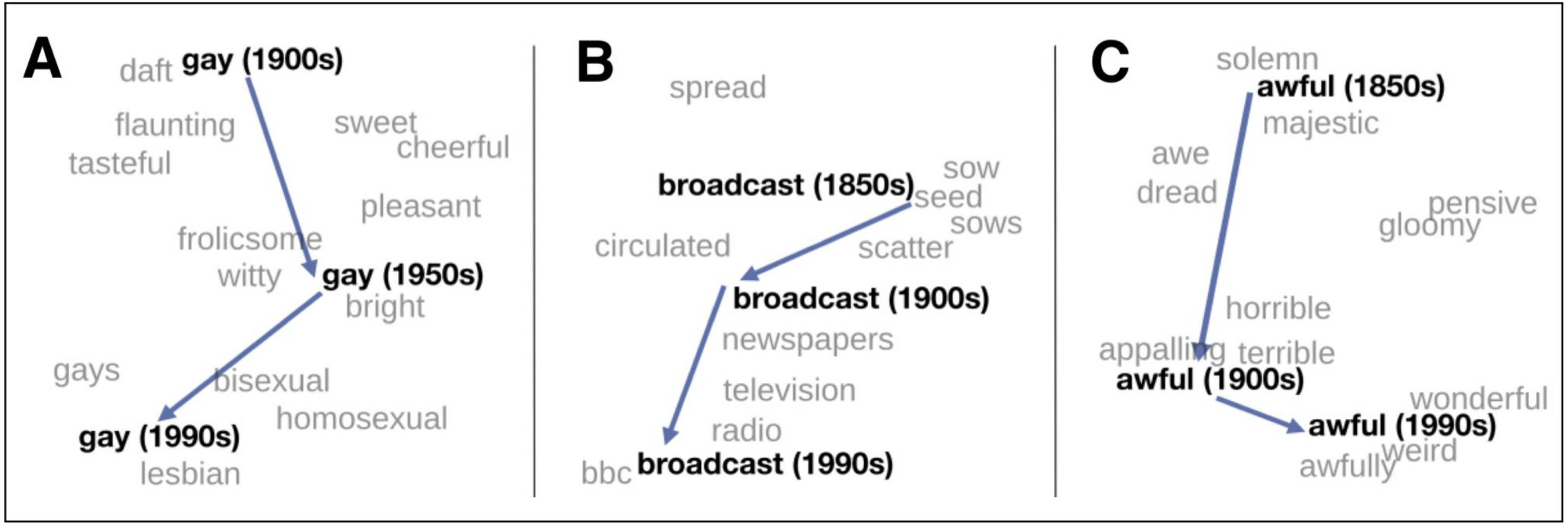
Figure 1: **Left** The most extreme occupations as projected on to the *she*−*he* gender direction on w2vNEWS. Occupations such as *businesswoman*, where gender is suggested by the orthography, were excluded. **Right** Automatically generated analogies for the pair *she*-*he* using the procedure described in text. Each automatically generated analogy is evaluated by 10 crowd-workers to whether or not it reflects gender stereotype.

[Bolukbasi et al. Neurips 2016]

Cornell Bowers C·IS

Word embeddings are time-dependent (why?)

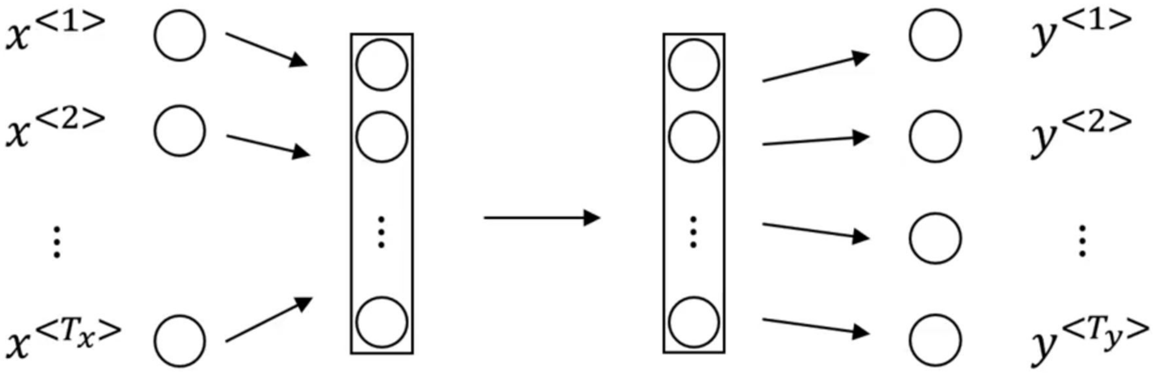
- Semantic similarity of words depends on *time.*



Problems with word2vec

- Words with multiple meanings only have one representation
  - eg. **bank** of river or **bank** of money
  - Need contextual information
- Limited Context
  - only trained on words within the context window

How to use word vectors with neural networks?

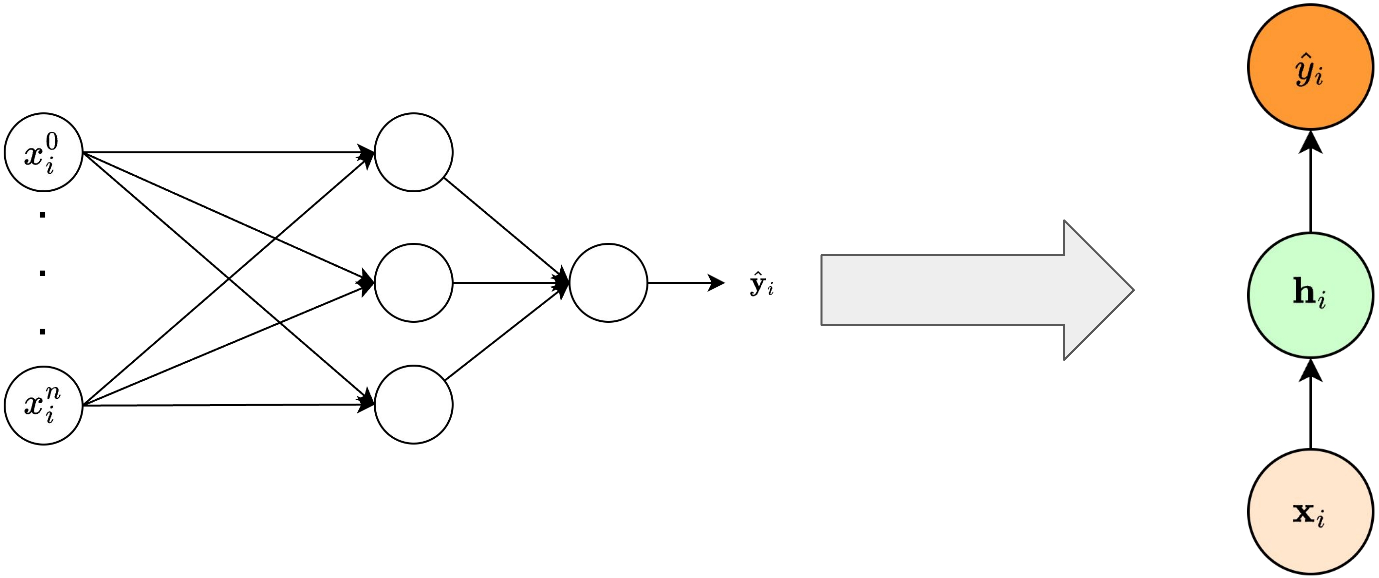


- Inputs and outputs don’t have fixed lengths
- Features are not shared

Cornell Bowers C·IS

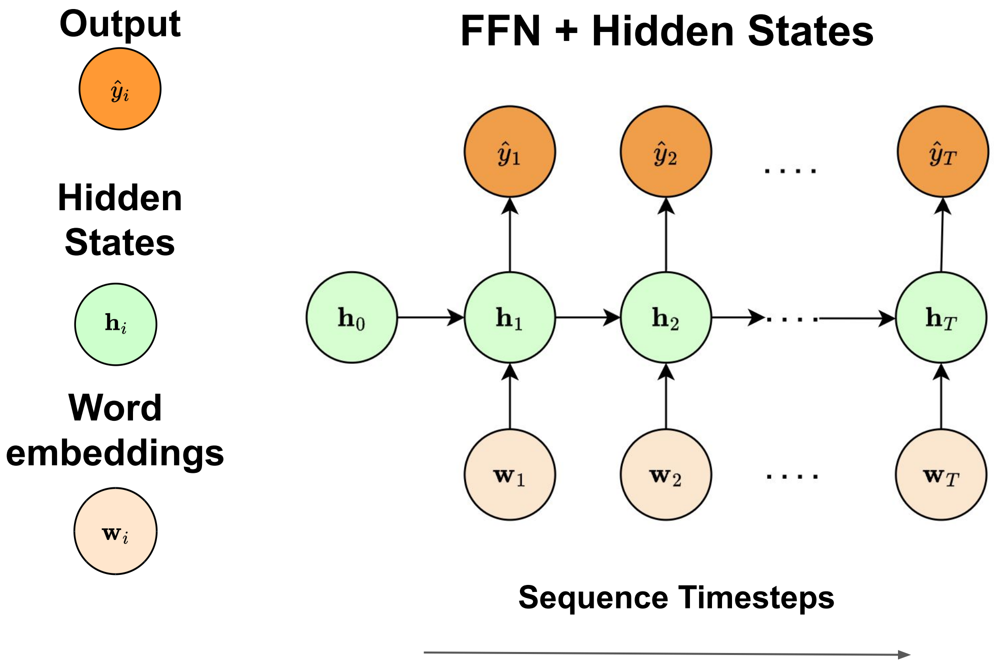
Let's simplify!

What if we have a single word and a single output?

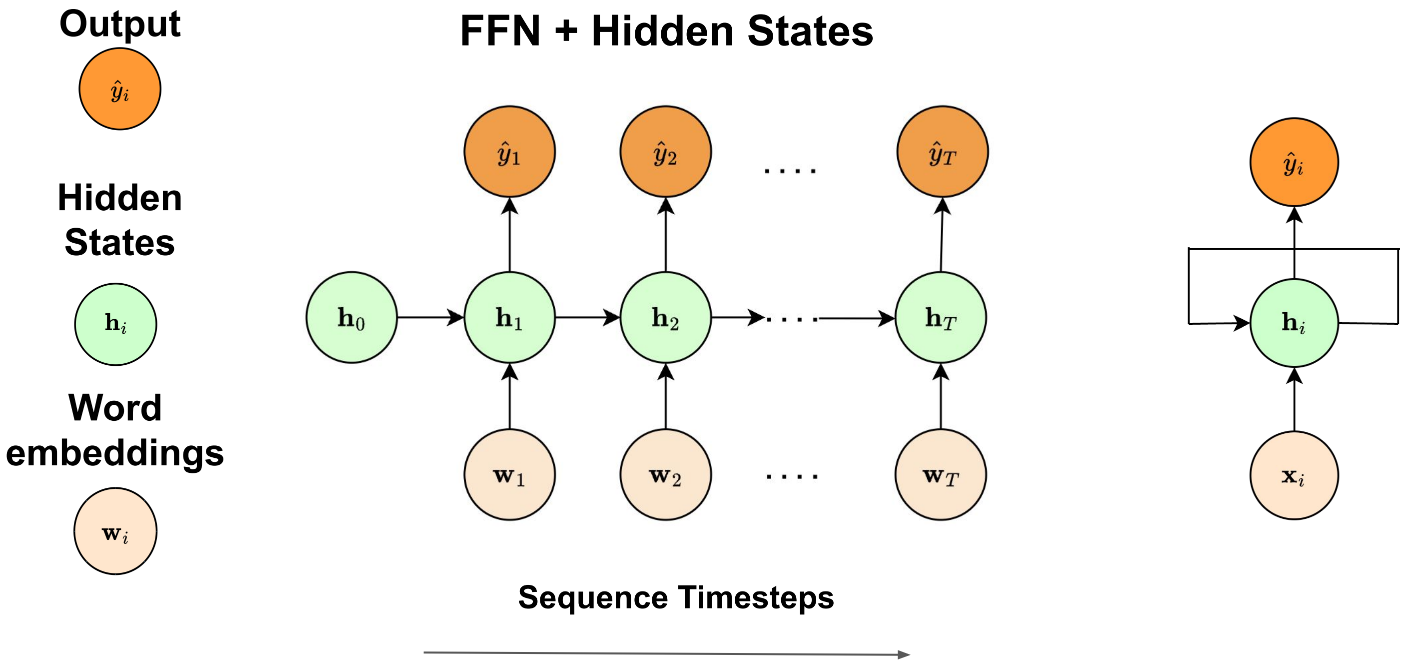


Cornell Bowers C·IS

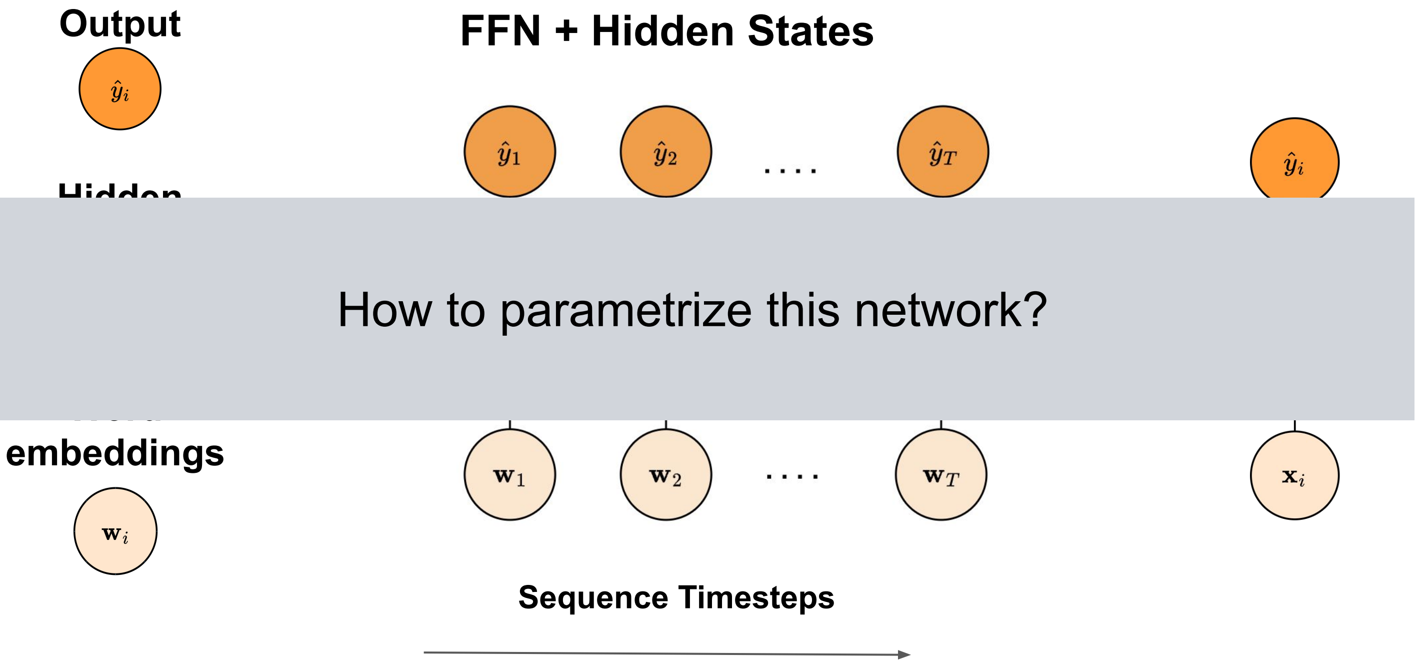
Recurrent neural network (RNN)



Recurrent neural network (RNN)



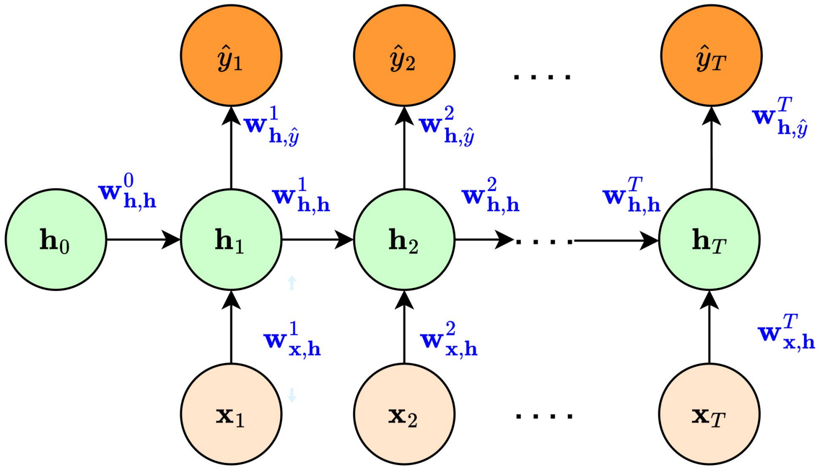
Recurrent neural network (RNN)



How to parametrize this network?

Cornell Bowers C·IS

Parameterize RNN

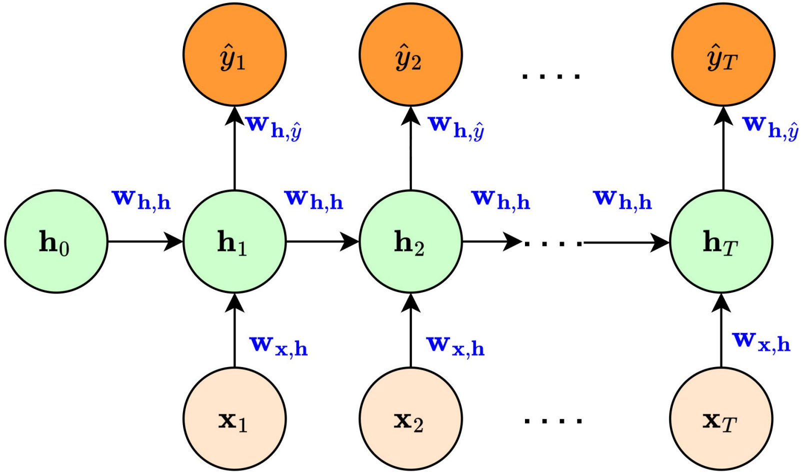


- Too many parameters if we have a long sequence!
- Longer sequence parameters will not receive many updates
- What if sequence lengths vary?

Cornell Bowers C·IS

RNN w/ parameter-sharing

Simple fix: use **the same parameters** across different timesteps.



$$\mathbf{h}_t = \phi\left(\mathbf{W}_{xh}x_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + b_h\right)$$

$$y_t = \psi\left(\mathbf{W}_{hy}\mathbf{h}_t + b_y\right)$$

transition functions (e.g. ReLU, Sigmoid, tanh)

Recap

- **N-gram models**
- **Bag-of-words representations**
- **Word2Vec**
  - CBOW: use context to predict target word
  - SkipGram: use target word to predict context
- **RNN**
  - Has an internal state (memory)
  - Can handle arbitrary sequences of inputs
  - Trained with back propagation through time

Cornell Bowers C·IS

Image credits:

https://web.stanford.edu/~jurafsky/slp3/6.pdf

https://lilianweng.github.io/posts/2017-10-15-word-embedding/