NEURAL LANGUAGE MODELING

TODAY’S CLASS

GOALS

1. What are Natural Language Processing (NLP) Tasks?
2. How do we represent language?
3. A simple model and training paradigm
4. Word embeddings (a better way to represent language)

NATURAL LANGUAGE PROCESSING

APPLICATIONS

NLP is everywhere!

Machine translation

Speech-to-text recognition

Chatbots

Sentiment analysis

Search Engines

Many more

NATURAL LANGUAGE PROCESSING

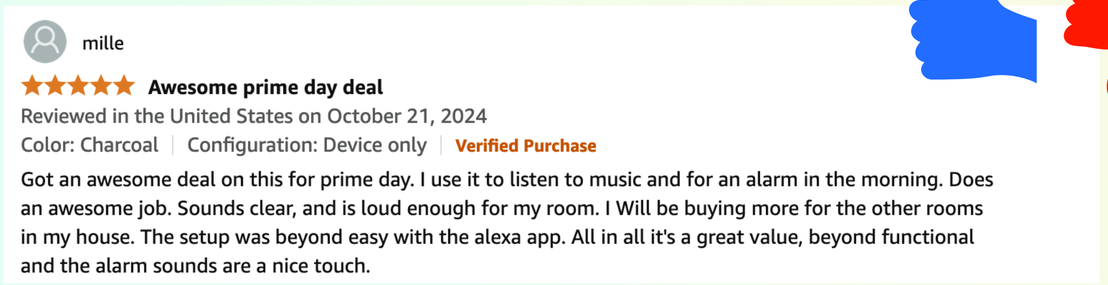
TASKS - WHAT WE TRAIN A MODEL TO DO

NATURAL LANGUAGE PROCESSING

TASKS - WHAT WE TRAIN A MODEL TO DO

Sentiment Analysis:

Is the review positive or negative



mille

★★★★★ **Awesome prime day deal**

Reviewed in the United States on October 21, 2024

Color: Charcoal | Configuration: Device only | **Verified Purchase**

Got an awesome deal on this for prime day. I use it to listen to music and for an alarm in the morning. Does an awesome job. Sounds clear, and is loud enough for my room. I Will be buying more for the other rooms in my house. The setup was beyond easy with the alexa app. All in all it's a great value, beyond functional and the alarm sounds are a nice touch.

NATURAL LANGUAGE PROCESSING

TASKS - WHAT WE TRAIN A MODEL TO DO

Sentiment Analysis:

Is the review positive or negative

Part of speech tagging:

NATURAL LANGUAGE PROCESSING

TASKS - WHAT WE TRAIN A MODEL TO DO

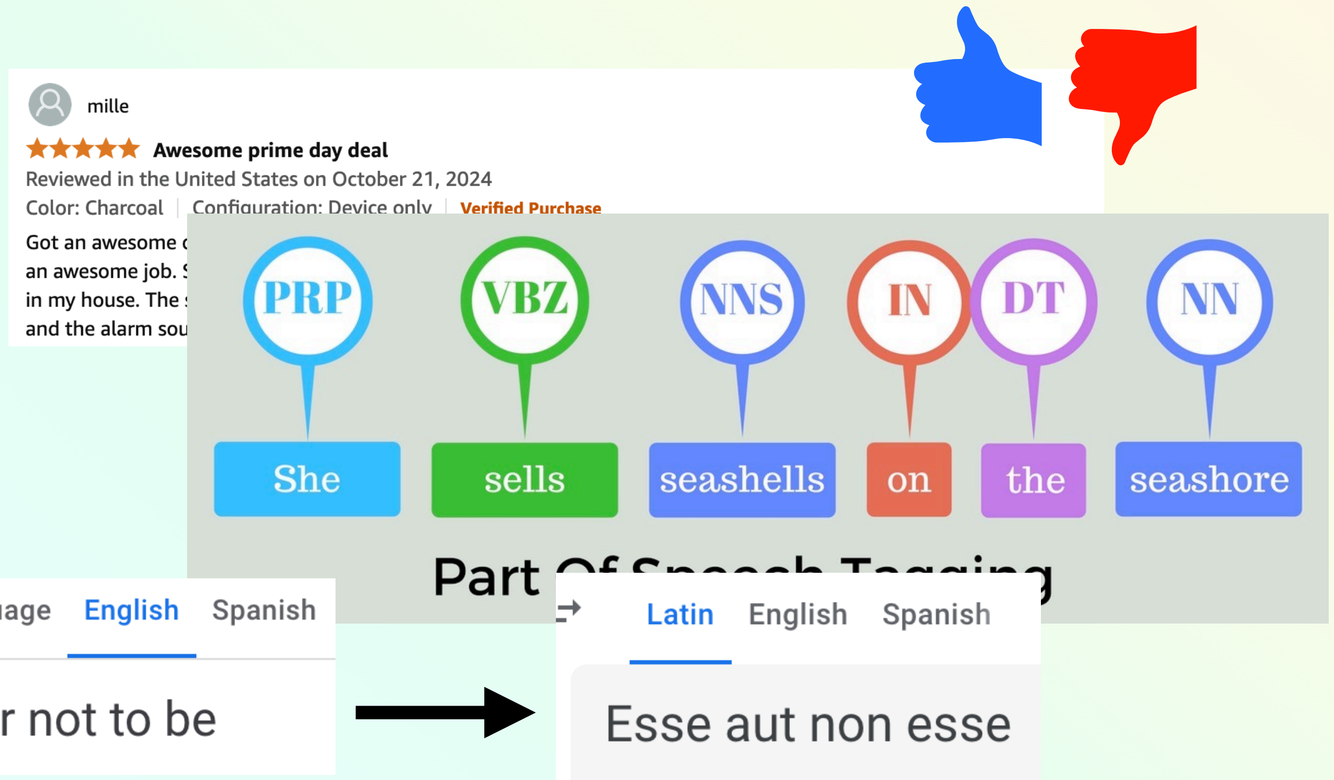
Sentiment Analysis:

Part of speech tagging:

Machine Translation:

Language Detection:

Is the review positive or negative



NATURAL LANGUAGE PROCESSING

TASKS - WHAT WE TRAIN A MODEL TO DO

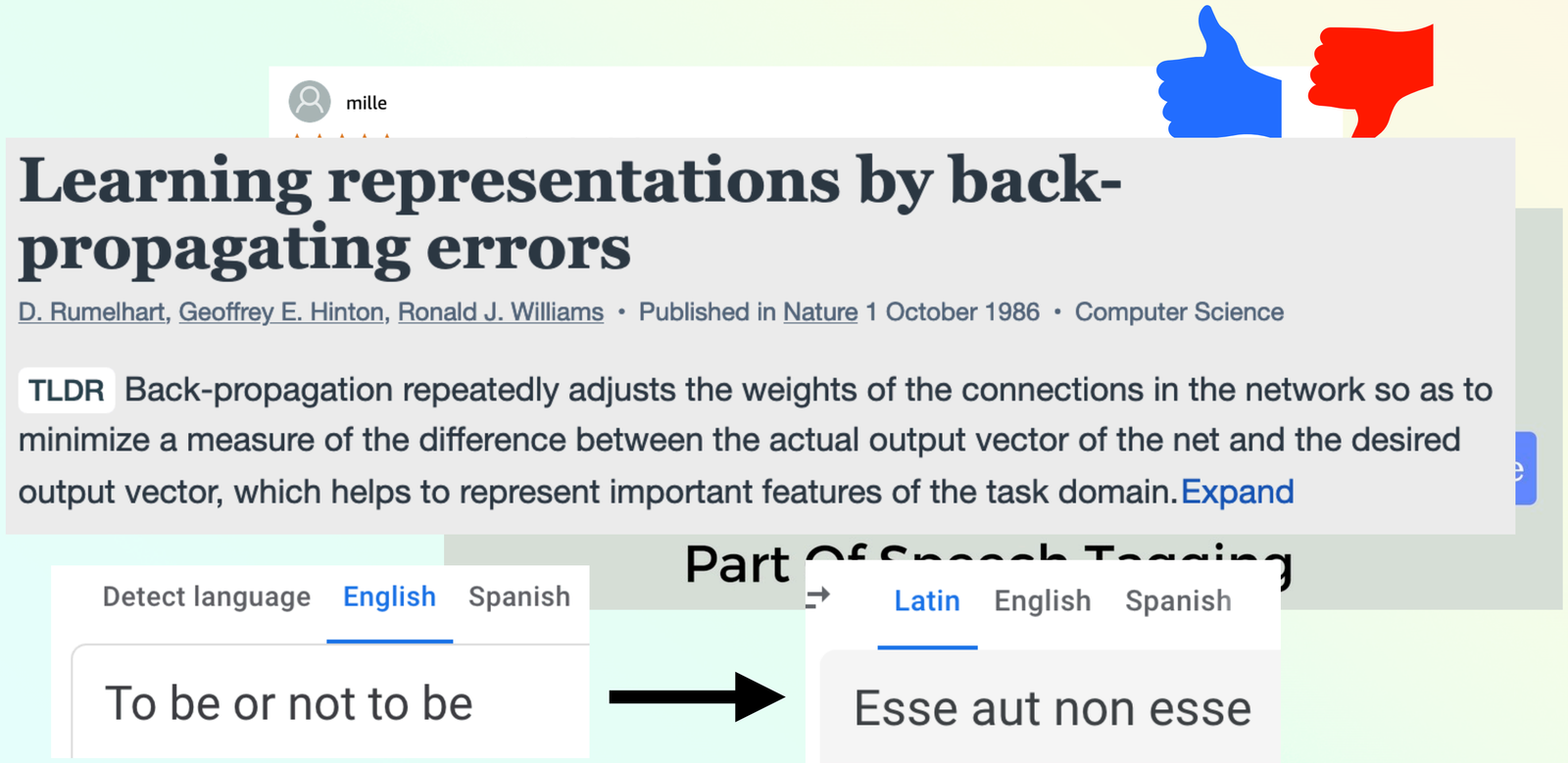
Sentiment Analysis:

Part of speech tagging:

Machine Translation:

Language Detection:

Text Summarization:



NATURAL LANGUAGE PROCESSING

TASKS - WHAT WE TRAIN A MODEL TO DO

Sentiment Analysis: Is the review positive or negative

Part of speech tagging:

Machine Translation:

Language Detection:

Text Summarization:

Language Modeling:

(LARGE) LANGUAGE MODELS

LLMS BASIC OVERVIEW

Used for many tasks based on providing a prompt to a model and having it generate a response:

The meaning of life is <model response>

Summarize this email for me …. <This email asks you to …>

What are the derivatives of the dense layer in a neural network with the format $f(h^{i-1}, W^i) = \sigma(h^{i-1}W^{i^\top})$ where \sigma is the ReLU activation function?

1. **With respect to W^i:**

$$\frac{\partial f}{\partial W^i} = D^\sigma(z^i) \cdot h^{i-1}$$

2. **With respect to h^{i-1}:**

$$\frac{\partial f}{\partial h^{i-1}} = D^\sigma(z^i) \cdot W^i$$

(LARGE) LANGUAGE MODELS

LLMS BASIC OVERVIEW

Premise: the text on the internet contains much of humanity’s knowledge; if a model can reproduce it, then it can be as smart as a human.

Method:

Model:

(LARGE) LANGUAGE MODELS

LLMS BASIC OVERVIEW

Premise: the text on the internet contains much of humanity’s knowledge; if a model can reproduce it, then it can be as smart as a human.

Method: create a very large model and train it to predict the next word(s)

Model:

(LARGE) LANGUAGE MODELS

LLMS BASIC OVERVIEW

Premise: the text on the internet contains much of humanity’s knowledge; if a model can reproduce it, then it can be as smart as a human.

Method: create a very large model and train it to predict the next word(s)

Model: large transformer architecture

- Word embedding
- Self-attention layers
- MLPs

(LARGE) LANGUAGE MODELS

LLMS BASIC OVERVIEW

Premise: the text on the internet contains much of humanity's knowledge; if a model can reproduce it, then it can be as smart as a human.

Method: create a very large model and train it to predict the next word(s)

Model: large transformer architecture

- Word embedding
- Self-attention layers
- MLPs

Next-text prediction is not enough: it needs to generate responses that are human-like.

RLHF: Reinforcement Learning with Human Feedback

Chatbots (more than an LLM) now cite sources (combine other tools)

QUIZ

REPRESENTING TEXT

CHOICES

String: “I hope her dog doesn’t bark when I knock on the door.”

UTF-(8/16/32): Each character is a integer

\x49\x20\x68\x6f\x70\x65\x20\x68\x65\x72\x20\x64\x6f\x67\x20\x64\x6f\x65\x73\x6e\xe2\x80\x99\x74\x20\x62\x61\x72\x6b\x20\x77\x68\x65\x6e\x20\x49\x20\x6b\x6e\x6f\x63\x6b\x20\x6f\x6e\x20\x74\x68\x65\x20\x64\x6f\x6f\x72\x2e

Each number is a feature — similar numbers don’t have much similarity in meaning

One-hot representation

“A” maps to [0,0,…(41st element) 1, 0, 0,…]

A string of length n is represented by a matrix of dimension $n \times d$ where d is the length of the one-hot representation

REPRESENTING TEXT

CHOICES

One-hot representation

“A” maps to [0,0,…(41st element) 1, 0, 0,…]

A string of length n is represented by a matrix of dimension $n \times d$ where d is the length of the one-hot representation. Or a vector of nd length

Con: the model has to learn to associate many characters together into words

REPRESENTING TEXT

CHOICES

String: “I hope her dog doesn’t bark when I knock on the door.”

Split string on words:

“I hope her dog doesn’t bark when I knock on the door.”

Create a dictionary of words

Need to stem words: barking->bark, creates->create

Each word is represented with a one-hot vector to indicate which word it is.

The string is now a matrix of a size num words $\times d$, which d is the number of elements in the dictionary.

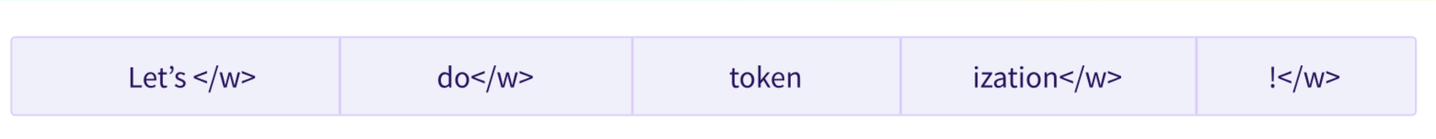
It can be a large representation, but the model starts with information about what are words

REPRESENTING TEXT

TOKENIZATION

Map string to list of tokens:

- Characters
- Words (split on spaces)
- Sub-word tokenization:
  - “Let’s do tokenization!” —>

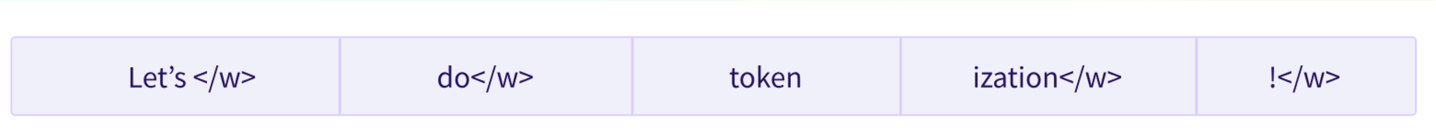


REPRESENTING TEXT

TOKENIZATION

Map string to list of tokens:

- Characters
- **Words** (split on spaces)
- Sub-word tokenization:
  - “Let’s do tokenization!” —>



Example from: https://huggingface.co/learn/nlp-course/en/chapter2/4

LANGUAGE MODELING

DATA

“I hope her dog doesn’t bark when I knock on the door.”

$X \in \mathbb{R}^{9 \times d}$

Want to predict the next word:

“I hope her dog doesn’t bark when I knock on the” —> door

There are d choices for the next word — classification task

$$-\ln \Pr\left(X_{next} = X_{9,\cdot} \,|\, X_{context} = X_{1:8,\cdot}\right)$$

LANGUAGE MODELING

MANY TRAINING POINTS FROM A SINGLE STRING

I —> hope

I hope —> her

I hope her —> dog

⋮

I hope her dog doesn't bark —> when

I hope her dog doesn't bark when —> I

I hope her dog doesn't bark when I —> knock

I hope her dog doesn't bark when I knock —> on

I hope her dog doesn't bark when I knock on —> the

I hope her dog doesn't bark when I knock on the —> door

L length string

$$l_X(\theta) = -\frac{1}{L-1}\sum_{t=1}^{L-1} \ln \Pr\left(X_{next} = X_{t+1,\cdot} \,|\, X_{context} = X_{1:t,\cdot}\right)$$

LANGUAGE MODELING

MANY TRAINING POINTS FROM A SINGLE STRING

I —> hope

I hope —> her

I hope her —> dog

⋮

I hope her dog doesn't bark —> when

I hope her dog doesn't bark when —> I

I hope her dog doesn't bark when I —> knock

I hope her dog doesn't bark when I knock —> on

I hope her dog doesn't bark when I knock on —> the

I hope her dog doesn't bark when I knock on the —> door

m strings with lengths $L_1, L_2, \ldots, L_m$

$$X \in \mathbb{R}^{m \times \max_i L_i \times d}$$

$$l_X(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \frac{1}{L_i - 1} \sum_{t=1}^{L_i - 1} \ln \Pr\left(X_{next} = X_{i,t+1,\cdot} \,|\, X_{context} = X_{i,1:t,\cdot}\right)$$

SIMPLE LANGUAGE MODEL

SIMPLE MODEL

Need to handle variable length strings:

- Fixed length input of L tokens
- If input string is shorter than L blank tokens are added
- If input string is longer than L then only the last L tokens are used
  - Fixed context window

SIMPLE LANGUAGE MODEL

FIXED CONTEXT WINDOW

I —> hope

I hope —> her

I hope her —> dog

⋮

I hope her dog doesn’t bark —> when

I hope her dog doesn’t bark when —> I

I hope her dog doesn’t bark when I —> knock

I hope her dog doesn’t bark when I knock —> on

I hope her dog doesn’t bark when I knock on —> the

I hope her dog doesn’t bark when I knock on the —> door

$$L = 10$$

SIMPLE LANGUAGE MODEL

FIXED CONTEXT WINDOW

_ _ _ _ _ _ _ _ _ _ I —> hope

_ _ _ _ _ _ _ _ _ _ I hope —> her

_ _ _ _ _ _ _ _ _ I hope her —> dog

⋮

_ _ _ _ I hope her dog doesn't bark —> when

_ _ _ I hope her dog doesn't bark when —> I

_ _ I hope her dog doesn't bark when I —> knock

_ I hope her dog doesn't bark when I knock —> on

I hope her dog doesn't bark when I knock on —> the

hope her dog doesn't bark when I knock on the —> door

SIMPLE LANGUAGE MODEL

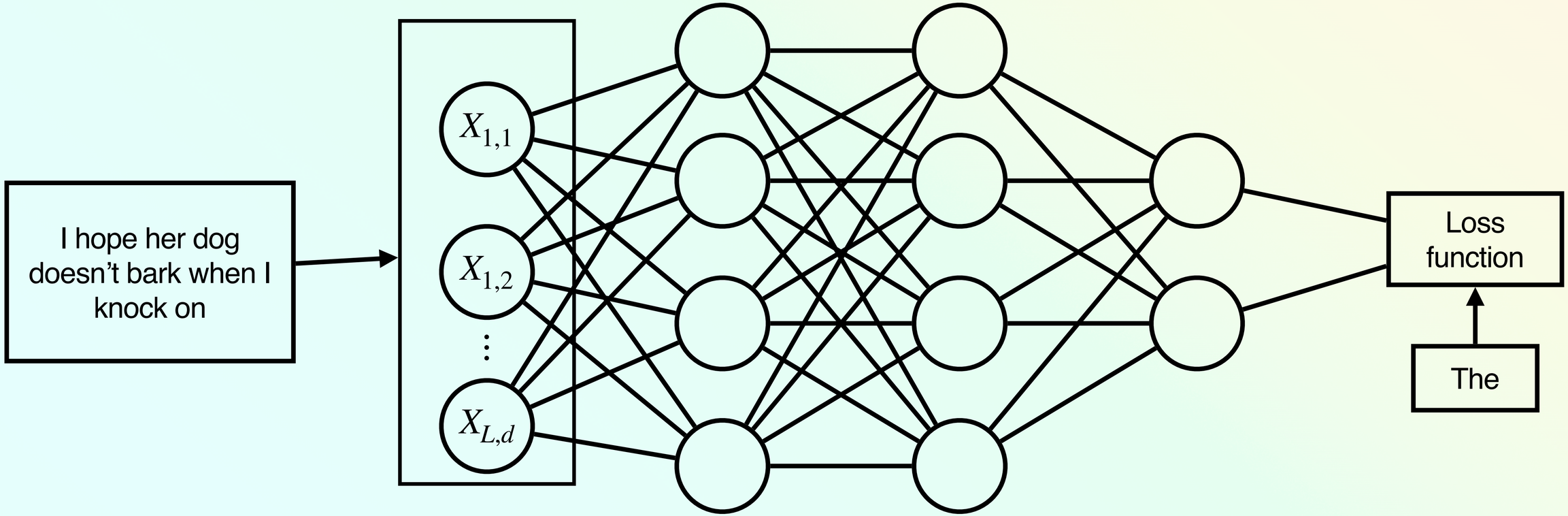
FIXED CONTEXT WINDOW

Network f: $\mathbb{R}^{L\times d} \times \Theta \to \mathbb{R}^d$

Treat X as a single Ld length vector and use MLPs

SIMPLE LANGUAGE MODEL

FIXED CONTEXT WINDOW



SIMPLE LANGUAGE MODEL

FIXED CONTEXT WINDOW

Network f: $\mathbb{R}^{L \times d} \times \Theta \to \mathbb{R}^d$

Treat X as a single Ld length vector and use MLPs

Bad idea:

- Lots of wasted compute for blank tokens
- Have to learn word relationships based on the input position
  - (Convolutions have been used in language too)

For now, think of the model as just some neural model

QUIZ

WORD REPRESENTATION

PROPERTIES

One-hot representation says all words are distinct and have an equal distance.

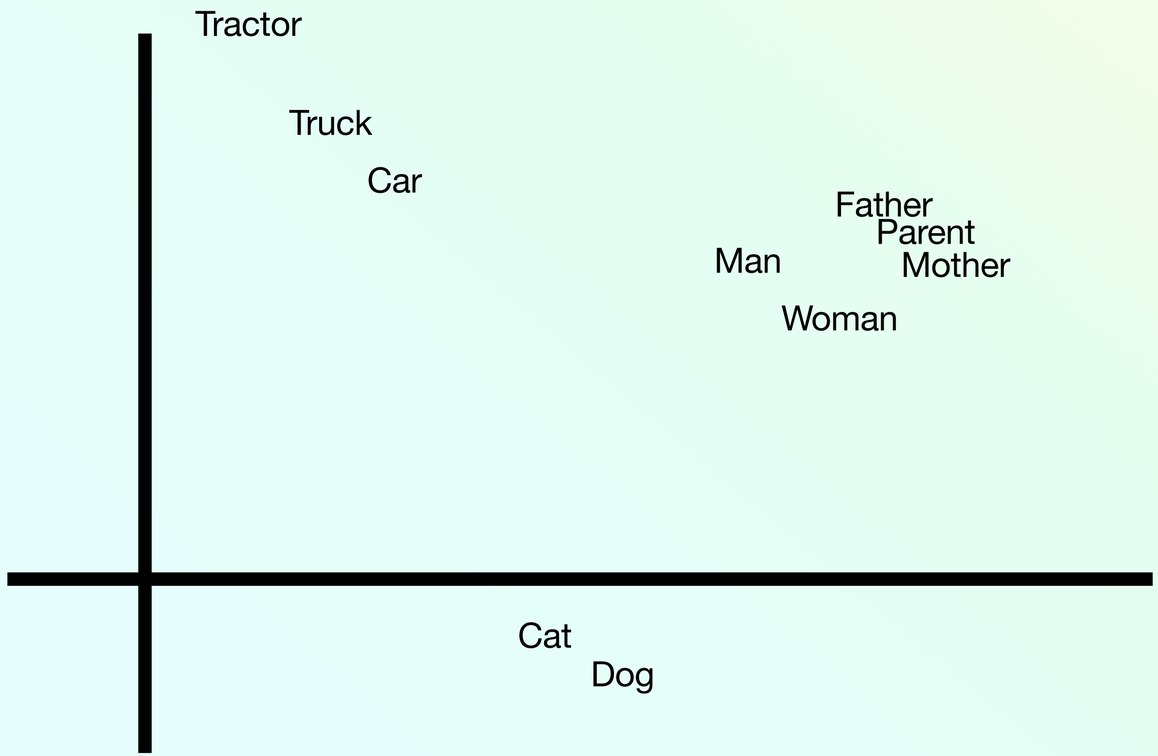
$$||father - mother|| = ||truck - cat||$$

Neural nets generalize based on interpolating over continuous values.

- One-hot representation provides a bad initialization for the network to learn relationships about words

Idea: create a word representation so that similar words are close together

WORD EMBEDDING



WORD MEANINGS

DERIVING MEANINGS FROM CONTEXT

How do you learn a new word?

What does the word Ong choy mean?

You have seen the sentences:

- Ong choy is **delicious sauteed with garlic**
- Ong choy is superb **over rice**
- Ong choy **leaves** with salty sauces

You have also seen:

- … spinach **sauteed with garlic over rice**
- Chard stems and **leaves** are **delicious**
- Collard greens and other **salty** leafy greens

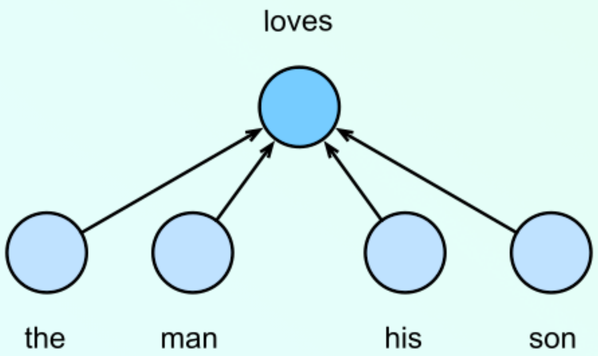
Conclusion:

Ong choy is a leafy green like spinach, chard, or collard greens.

CREATING A WORD EMBEDDING

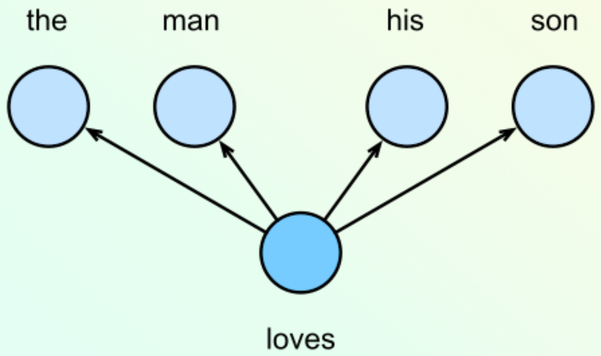
DERIVING MEANINGS FROM CONTEXT

Word2vec: map a word to vector



Continuous Bag of Words

Given set of neighboring words predict the missing word



Skip-gram

Given a word predict the surrounding context

CREATING A WORD EMBEDDING

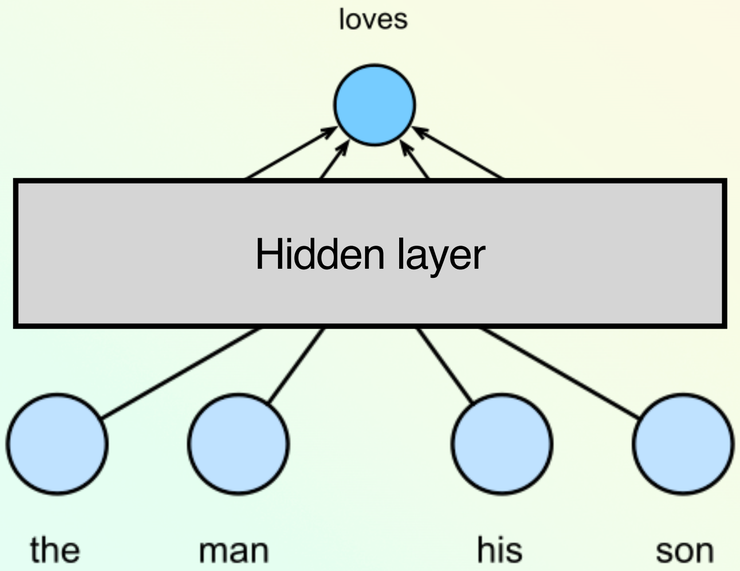
DERIVING MEANINGS FROM CONTEXT

Word2vec: map a word to vector

- Take input words and map them to a single hidden layer
- Then, map to the output layer
- Minimize NLL of missing word across a large corpose of text

Create a dictionary of [missing word]->hidden layer

Hidden layer representation is the word embedding



WORD EMBEDDING

PROPERTIES

Learned representation preserves semantic meanings

Similar words are nearby in the embedding space

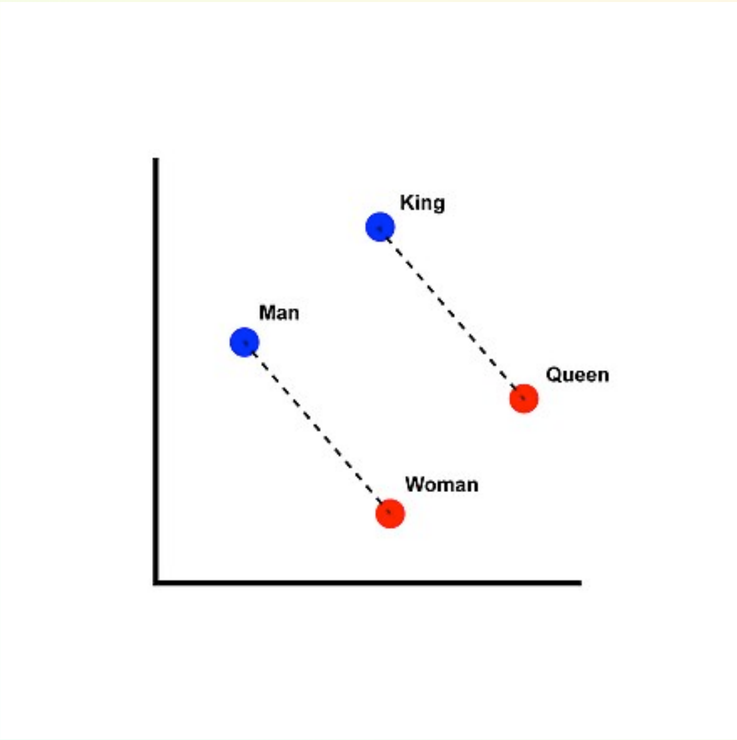
Semantic algebra:

$$King - Man + Woman \approx Queen$$

Bias:

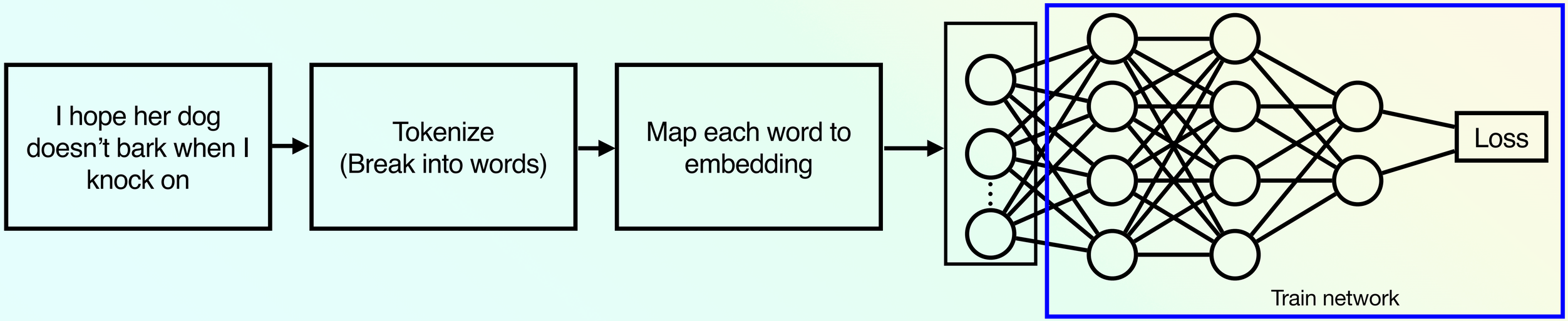
$$Doctor - man + woman \approx nurse$$

Representations are based on statistical prevalences, not actual meanings.



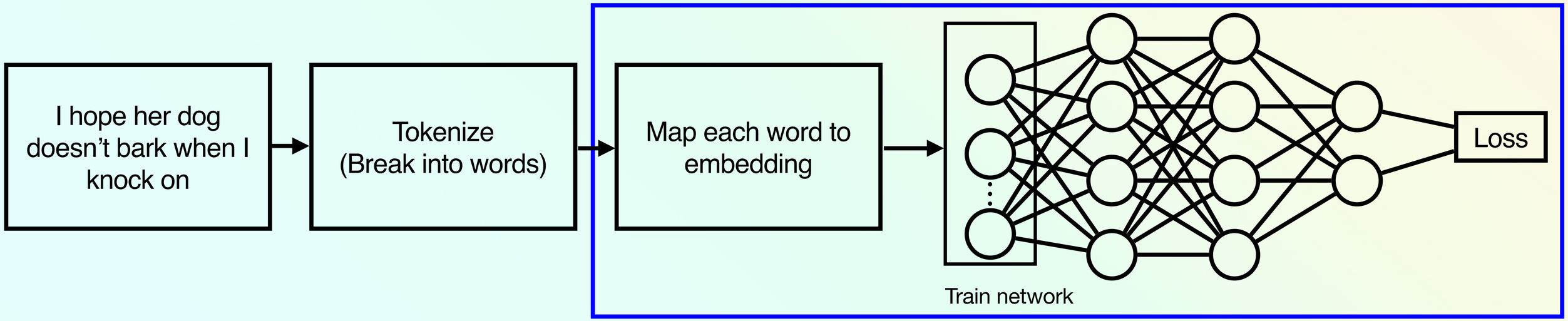
SIMPLE LANGUAGE MODEL

WITH WORD EMBEDDING



SIMPLE LANGUAGE MODEL

WITH LEARNED WORD EMBEDDING



Can learn word embeddings that are good for the specific task

LLMs start with random embeddings and learn from scratch

LANGUAGE MODELING

PROPERTIES

1. Need to tokenize a string
2. Create a token embedding so the network can easily learn relationships across tokens
3. Minimize prediction error of the next token

Word2vec style embeddings are static and represent the word in all contexts it is ever used.

Ignores context-specific meaning of word

NEXT CLASS

Context-Specific Embeddings (next step towards LLMs)