CS 540 Introduction to Artificial Intelligence

Natural Language Processing (before LLMs)

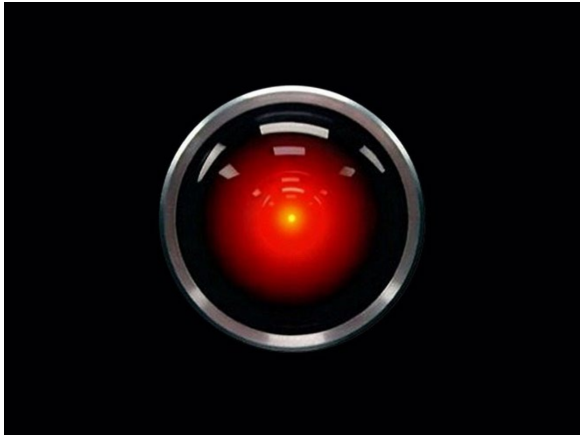
University of Wisconsin-Madison

Spring 2024

What is **NLP**?

Combining computing with human language. Want to:

- Answer questions
- Summarize or extract information
- Translate between languages
- Generate dialogue/language
- Write stories automatically



Why is it **hard**?

Many reasons:

- Ambiguity: "*Mary saw the duck with the telescope in the park*"*.* Several meanings.
- Understanding of the world
  - "Bob and Joe are fathers".
  - "Bob and Joe are brothers".

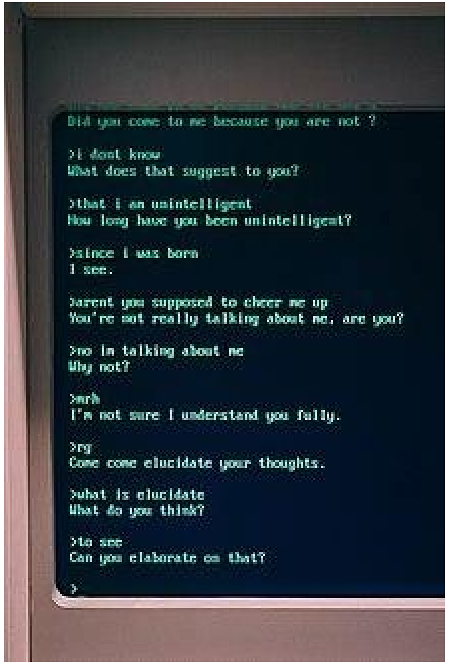
Approaches to NLP

A brief history

- Symbolic NLP: 50’s to 90’s
- Statistical/Probabilistic: 90’s to present
    - Neural nets: 2010’s to present
    - Large Language Model (LLM): GPT etc.

Lots of progress!

Lots more to work to do



```
Did you come to me because you are not ?

>i dont know
What does that suggest to you?

>that i am unintelligent
How long have you been unintelligent?

>since i was born
I see.

>arent you supposed to cheer me up
You're not really talking about me, are you?

>no im talking about me
Why not?

>mrh
I'm not sure I understand you fully.

>rg
Come come elucidate your thoughts.

>what is elucidate
What do you think?

>to see
Can you elaborate on that?

>
```

ELIZA program

Outline

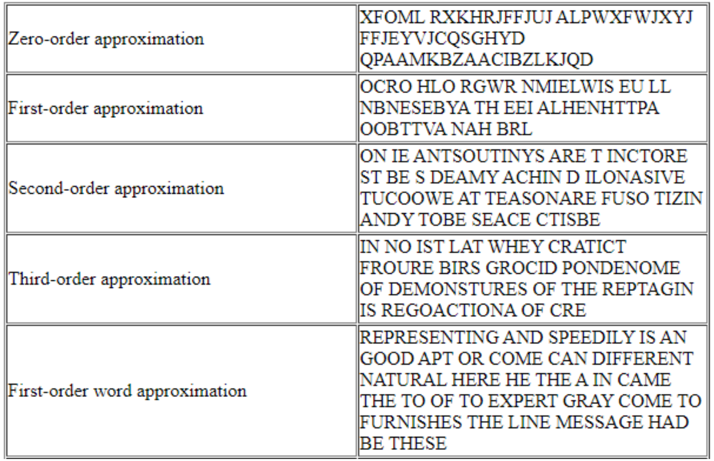
- Introduction to language models
  - n-grams, training, evaluation, generation
- Word representations
  - One-hot, word embeddings, transformer-based

Language Models

- Basic idea: use probabilistic models to **assign a probability to a sentence W**

$$P(W) = P(w_1, w_2, \ldots, w_n) \text{ or } P(w_{\text{next}} | w_1, w_2 \ldots)$$

- Goes back to Shannon
  - Information theory: letters



Zero-order approximation	XFOML RXKHRJFFJUJ ALPWXFWJXYJ FFJEYVJCQSGHYD QPAAMKBZAACIBZLKJQD
First-order approximation	OCRO HLO RGWR NMIELWIS EU LL NBNESEBYA TH EEI ALHENHTTPA OOBTTVA NAH BRL
Second-order approximation	ON IE ANTSOUTINYS ARE T INCTORE ST BE S DEAMY ACHIN D ILONASIVE TUCOOWE AT TEASONARE FUSO TIZIN ANDY TOBE SEACE CTISBE
Third-order approximation	IN NO IST LAT WHEY CRATICT FROURE BIRS GROCID PONDENOME OF DEMONSTURES OF THE REPTAGIN IS REGOACTIONA OF CRE
First-order word approximation	REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE

Training The Model

Recall the chain rule of probability:

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2|w_1)\ldots P(w_n|w_{n-1}\ldots w_1)$$

- How do we estimate these probabilities?
  - I.e., "training" in machine learning.
- From data (text corpus)
  - Can't estimate reliably for long histories.

Training: Make Assumptions

- Markov assumption with shorter history:

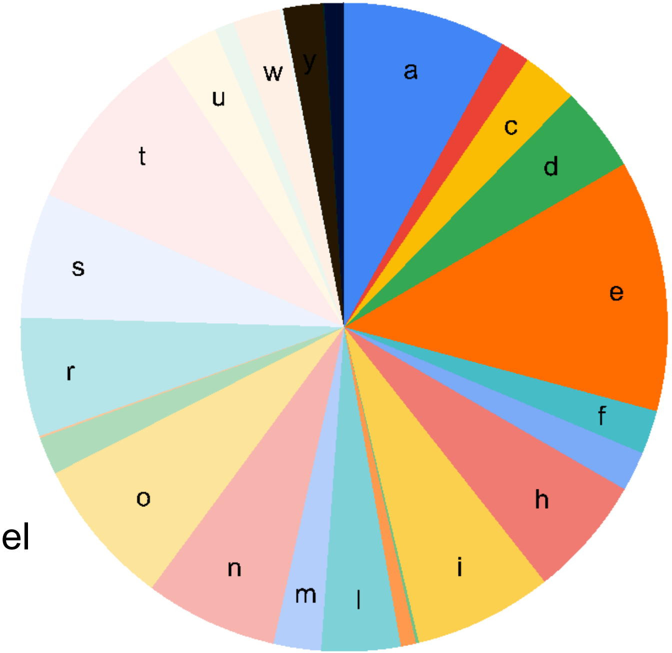
$$P(w_i|w_{i-1}w_{i-2}\ldots w_1) = P(w_i|w_{i-1}w_{i-2}\ldots w_{i-k})$$

- Present doesn’t depend on whole past
  - Just recent past, i.e., *context*.
  - What’s ***k=0?***

k=0: **Uni**gram Model

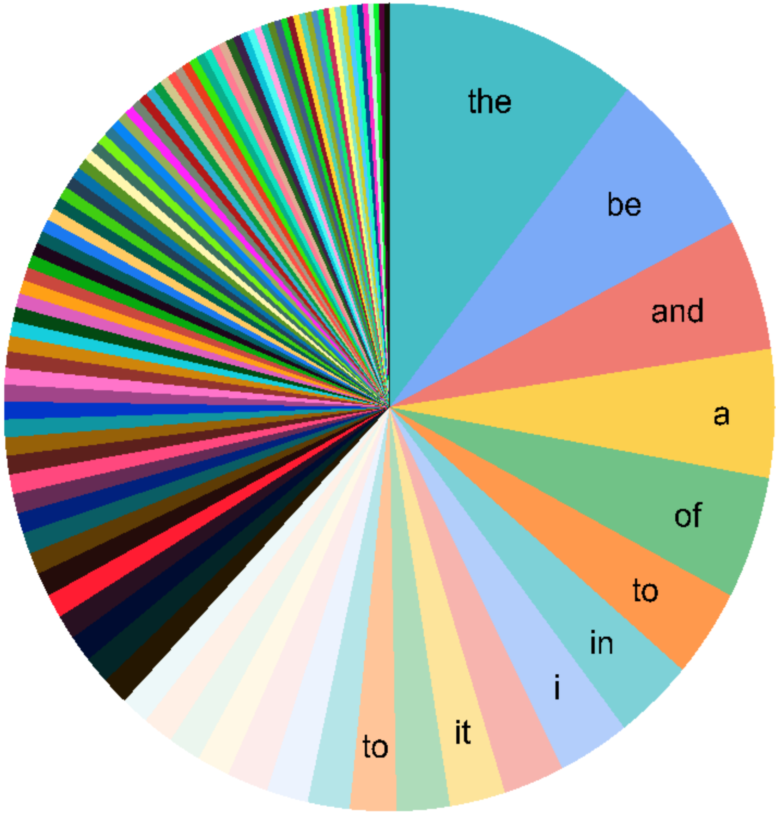
- Full independence assumption:
  - (Present doesn't depend on the past)

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2)\ldots P(w_n)$$



The English letter frequency wheel

Unigram word model



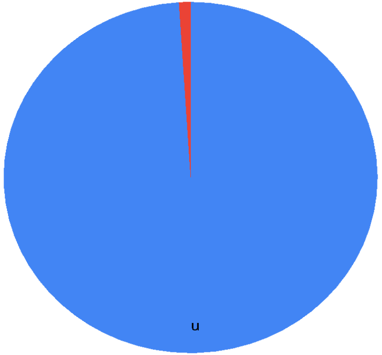
Example (from Dan Jurafsky’s notes)

```
fifth, an, of, futures, the, an, incorporated, a,
a, the, inflation, most, dollars, quarter, in, is,
mass thrift, did, eighty, said, hard, 'm, july,
bullish that, or, limited, the
```

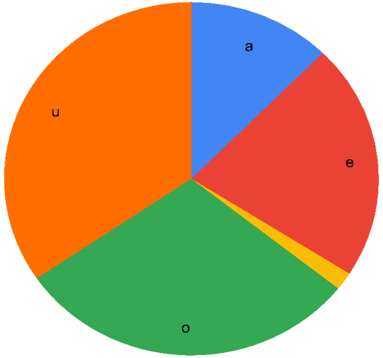
k=1: **Bi**gram Model

- Markov Assumption:
  - (Present depends on immediate past)

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_n|w_{n-1})$$



$p(.|q)$: the "after q" wheel



$p(.|j)$: the "after j" wheel

texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen outside, new, car, parking, lot, of, the, agreement, reached this, would, be, a, record, november

k=n-1: **n-**gram Model

Can do trigrams, 4-grams, and so on

- More expressive as *n* goes up
- Harder to estimate

Training: just count? I.e, for bigram:

$$P(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

Simple “generative AI” from letter bigram (Markov Chain)

Writing = sampling

- Say we start with q
- Sample from $P(\cdot \mid q)$: spin the “after q” wheel , we get u
- Sample from $P(\cdot \mid u)$: spin the “after u” wheel, say we get e
- Sample from $P(\cdot \mid e)$: spin the “after e” wheel, say we get r
- ...

Sampling Shakespeare unigram LM

- To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
- Every enter now severally so, let
- Hill he late speaks; or! a more to leg less first you enter
- Will rash been and by I the me loves gentle me not slavish page, the and hour; ill let
- Are where exeunt and sighs have rise excellency took of .. sleep knave we. near; vile like

Jurafsky & Martin, Speech and language processing, Prentice Hall, 2000.

Sampling Shakespeare bigram LM

- What means, sir. I confess she? then all sorts, he is trim, captain.
- Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
- What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?
- Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt

Jurafsky & Martin, Speech and language processing, Prentice Hall, 2000.

Sampling Shakespeare trigram LM

- Sweet prince, Falstaff shall die. Harry of Monmouth's grave.
- This shall forbid it should be branded, if renown made it empty.
- What ist that cried?
- Indeed the duke; and had a very good friend.

Jurafsky & Martin, Speech and language processing, Prentice Hall, 2000.

n-gram Training

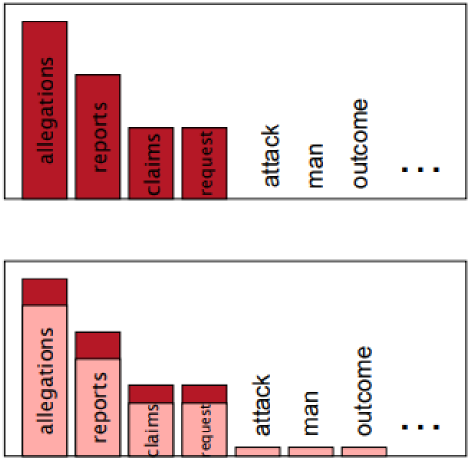
$$P(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

Issues:

- **1**. Multiply tiny numbers?
  - **Solution**: use logs; add instead of multiply
- **2.** n-grams with zero probability?
  - **Solution**: smoothing

$$P(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i) + 1}{\text{count}(w_{i-1}) + V}$$

P(*w*|denied the)



Dan Klein

Break & Quiz

Q 1.1: Which of the below are bigrams from the sentence “It is cold outside today”.

- A. It is
- B. cold today
- C. is cold
- D. A & C

Break & Quiz

Q 1.1: Which of the below are bigrams from the sentence "It is cold outside today".

- A. It is
- B. cold today
- C. is cold
- **D. A & C**

Break & Quiz

Q 1.2: Smoothing is increasingly useful for n-grams when

- A. n gets larger
- B. n gets smaller
- C. always the same
- D. n larger than 10

Break & Quiz

Q 1.2: Smoothing is increasingly useful for n-grams when

- **A. n gets larger**
- B. n gets smaller
- C. always the same
- D. n larger than 10

Evaluating Language Models

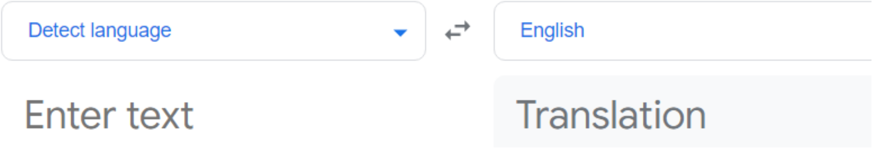
How do we know we've done a good job?

- Observation
- Train/test on separate data & measure metrics
- **Metrics**:
  - 1. Extrinsic evaluation
  - 2. Perplexity

Extrinsic Evaluation

How do we know we've done a good job?

- **Pick a task** and use the model to do the task
- For two models, M_1, M_2, compare the accuracy for each task
  - **Ex**: Q/A system: how many questions right. Translation: how many words translated correctly
- Downside: slow; may change relatively



Intrinsic Evaluation: Perplexity

Perplexity is a **measure of uncertainty**

$$\mathrm{PP}(W) = P(w_1, w_2, \ldots, w_n)^{-\frac{1}{n}}$$

Compute average PP(W) for all W from a dataset

Lower is better! Examples:

- WSJ corpus; 40 million words for training:
  - Unigram: 962, Bigram 170, Trigram 109

Further NLP Tasks

Language modeling is **not the only NLP task:**

- Part-of-speech tagging, parsing, etc.
- Question-answering, translation, summarization, classification (e.g., sentiment analysis), generation, etc.

Break & Quiz

Q 2.1: What is the perplexity for a sequence of *n* digits 0-9? All occur independently with equal probability.

$$\mathrm{PP}(W) = P(w_1, w_2, \ldots, w_n)^{-\frac{1}{n}}$$

- A. 10
- B. 1/10
- C. 10^n
- D. 0

Break & Quiz

Q 2.1: What is the perplexity for a sequence of *n* digits 0-9? All occur independently with equal probability.

$$\mathrm{PP}(W) = P(w_1, w_2, \ldots, w_n)^{-\frac{1}{n}}$$

- **A. 10**
- B. 1/10
- C. 10^n
- D. 0

$(P(w_1)*P(w_2)\ldots*P(w_{10}))^{(-1/10)} = ((1/10)*(1/10)*\ldots.(1/10))^{(-1/10)} = 10$

Representing Words

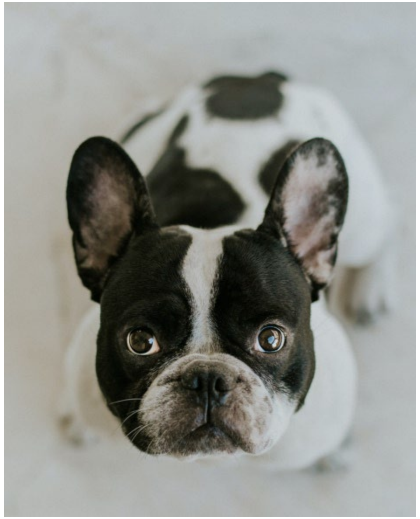
Remember value of random variables **(RVs)**

- Easier to work with than objects like ‘dog’

Traditional representation: **one-hot vectors**

$$\text{dog} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Dimension: # of words in vocabulary
- Relationships between words?



Smarter Representations

Distributional semantics: account for relationships

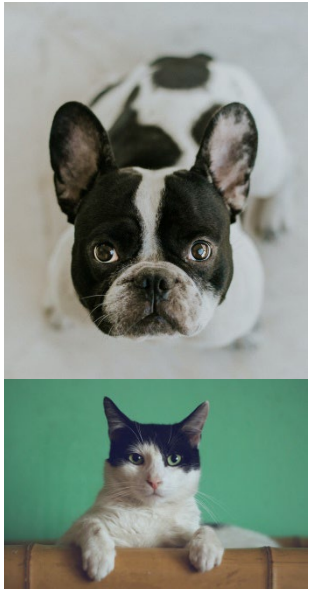
- Reps should be close/similar to other words that appear in a similar context

Dense vectors:

$$\text{dog} = \begin{bmatrix} 0.13 & 0.87 & -0.23 & 0.46 & 0.87 & -0.31 \end{bmatrix}^T$$

$$\text{cat} = \begin{bmatrix} 0.07 & 1.03 & -0.43 & -0.21 & 1.11 & -0.34 \end{bmatrix}^T$$

AKA **word embeddings**



Training Word Embeddings

Many approaches (super popular 2010-present)

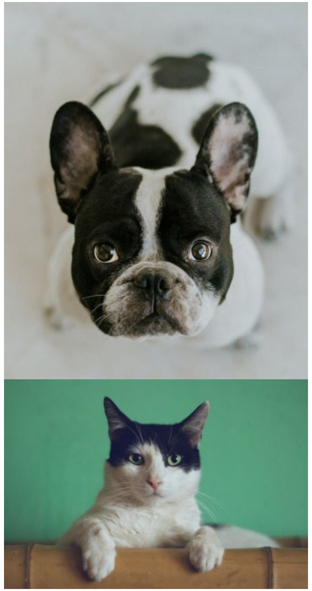
- Word2vec: a famous approach
- What’s our likelihood?

Windows of length 2a

$$L(\theta) = \prod_{t=1}^{T} \prod_{-a\leq j\leq a} P(w_{t+j}|w_t,\theta)$$

Our word vectors

All positions



Training Word Embeddings

Word2vec likelihood

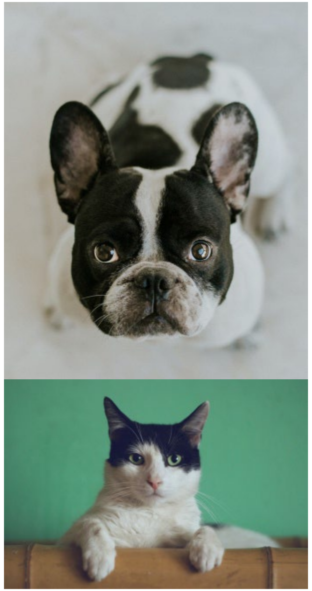
$$L(\theta) = \prod_{t=1}^{T} \prod_{-a \leq j \leq a} P(w_{t+j} | w_t, \theta)$$

- Maximize this; what’s the probability?
  - Two vectors per word. v_w, u_w for center/context

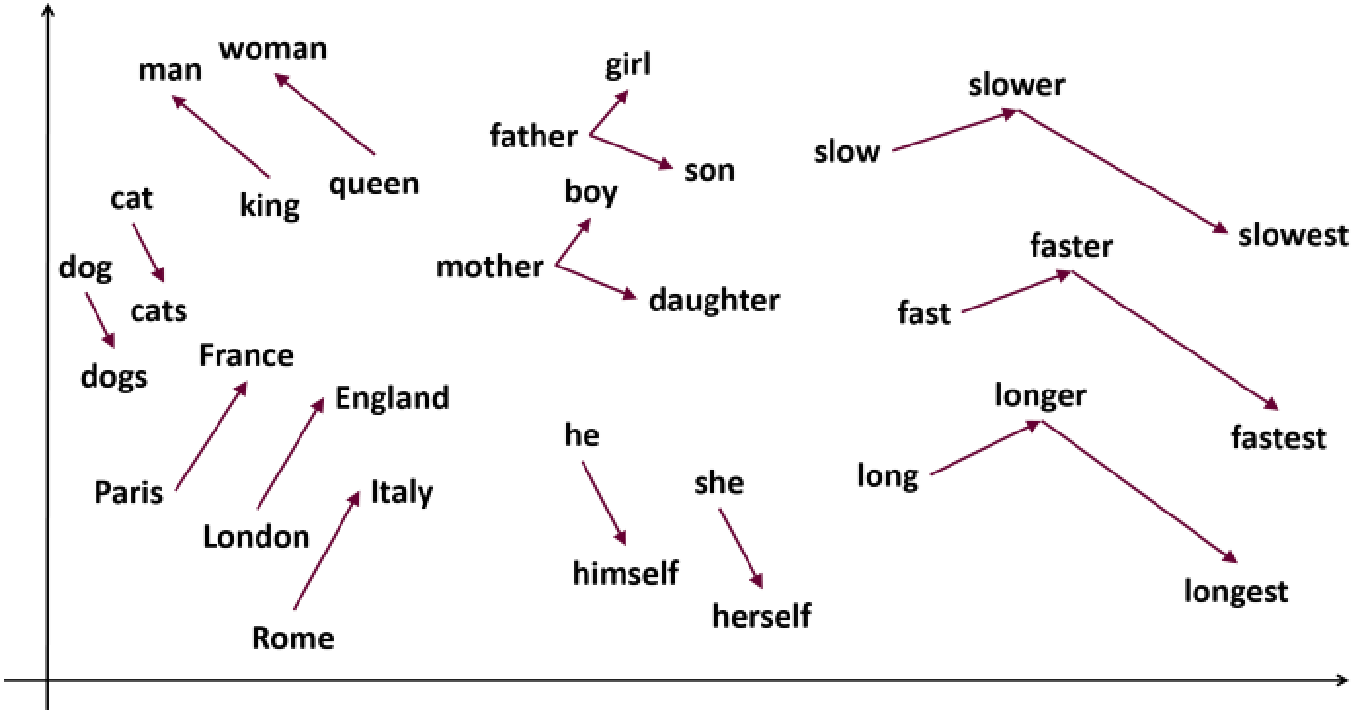
(o is context word, c is center)

Similarity

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$



Word Embeddings

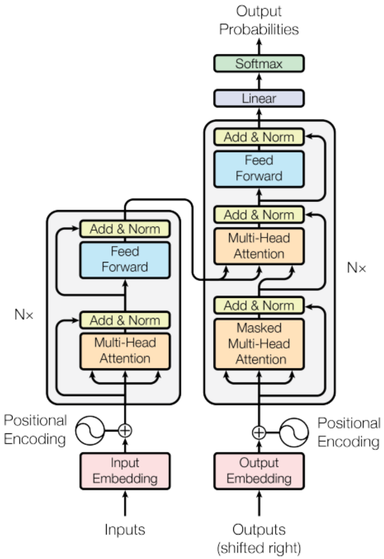


Saurabh Pal – Implementing Word2Vec in Tensorflow

Beyond “Shallow” Embeddings

- Transformers: special model architectures based on **attention**
  - Sophisticated types of neural networks
- Pretrained models
  - Based on transformers: BERT, GPT
  - Include context!

- **Fine-tune** for desired task



Vaswani et al.

Reading

- Natural Language and Statistics, Notes by Zhu. https://pages.cs.wisc.edu/~jerryzhu/cs540/handouts/NLP.pdf