

# Natural Language Processing

Lecture 1

Zhilin Yang

# Outline

- Course overview
- NLP intro
- Sub-languages
- Probabilistic view of natural language
- Word vectors

# Outline

- **Course overview**
- NLP intro
- Sub-languages
- Probabilistic view of natural language
- Word vectors

# Course Overview

- Instructor: Zhilin Yang
  - Email: [zhiliny@tsinghua.edu.cn](mailto:zhiliny@tsinghua.edu.cn)
- TA: Chonghua Liao
  - Email: [lch22@mails.tsinghua.edu.cn](mailto:lch22@mails.tsinghua.edu.cn)
- Composition
  - Assignments: 30%
    - 5 assignments in total
  - Project: 40%
  - Final: 30%
- To keep up with rapid development of the field, we choose to base this course more on mixed resources (e.g. papers) rather than using a fixed textbook
- For in-class QA, feel free to answer the questions in Mandarin
- Grades will be curved

# Course Project

- Teams of 2 or 3
  - Mostly shared scores
- Project topics are flexible, as long as it is related to NLP
  - We will also provide a default project topic
- Scored in the following aspects:
  - Literature review
  - Depth of work
  - Presentation (including oral presentation and written report)
  - Significance of results and novelty of main ideas
- Proposal 5% + milestone 5% + report 30% = whole project 40%
- Results should be reproducible
  - Code and scripts must be provided along with the report
- References should be clearly cited

# Assignment Policy

- Use of any additional materials should be acknowledged
  - Books, papers, web, collaboration
- Everything you submit should be your own work
  - We are very serious about academic integrity
  - Any plagiarism will lead to an automatic F and will be subject to penalties by school policies.
  - It is allowed to discuss, but you should ack this in the assignment. You will also have to write your own code and solutions.
  - E.g., copying any materials (including code) from others' submissions or the Web is prohibited and will be considered as plagiarism.
  - Using materials from previous years is prohibited. Asking someone who took this course before for solutions is considered as plagiarism.
- Late submissions
  - 3 late days available in total
  - After 3 late days are used, 75% discount within 24h and 50% discount within 48h
  - No score beyond 48h
  - For emergency cases, contact TA for approval of postponed deadlines

# Syllabus

Week	Date	Course	Release	Due
1	Sept 19	intro, sublanguages, word vectors, source channel		
2	Sept 26	more word vectors	hw1	
3	Oct 10	smoothing, LMs		
4	Oct 17	RNNs, LSTMs	hw2	hw1
5	Oct 24	research, project, classification	project	
6	Oct 31	MT, seq2seq, subwords	hw3	hw2
7	Nov 7	attention, transformers		
8	Nov 14	advanced transformers, encoder pretraining, decoder pretraining	hw4	hw3
9	Nov 21	seq2seq pretraining, prompting		project proposal
10	Nov 28	prompt tuning, task generalization, long-sequence transformers		hw4
11	Dec 5	structured prediction, parsing	hw5	
12	Dec 12	QA, reading comprehension, reasoning		project milestone
13	Dec 19	knowledge bases and selected topics (e.g., multilingual learning, multimodal learning)		hw5
14	Dec 16	project presentation		
15	Jan 2	<b>final exam</b>		
	Jan 16			project

# Resources

- I will be using materials (e.g., figures and text) from the following resources
  - <http://www.phontron.com/class/nn4nlp2021/schedule.html>
  - <http://demo.clab.cs.cmu.edu/NLP/#>
  - <http://web.stanford.edu/class/cs224n/index.html#schedule>
  - <http://www.cs.cmu.edu/~roni/11761/>
  - <http://www.cs.cmu.edu/~yiming/MLTM-f20-index.htm>
  - <https://www.cs.cmu.edu/~hovy/>
  - <https://sites.google.com/site/spflodd/>
  - <https://courses.grainger.illinois.edu/cs447/fa2019/index.html>
  - Hinton's deep learning course at UToronto
  - Russ's deep learning course at CMU
- Credits go to the original authors



# Goals of This Course

- Foundations of NLP approaches
  - From classics to distributed deep systems to pretraining
  - From ngrams to linear models to RNNs to transformers
- Critical problems in NLP
  - A big picture of what NLP is about
  - Understanding the challenges of key NLP problems
- The ability to solve practical NLP tasks
- (plus) an initial sense of how to perform research in NLP

# Outline

- Course overview
- **NLP intro**
- Sub-languages
- Probabilistic view of natural language
- Word vectors

# Natural Language Processing

- Question: what do you think NLP is? Can you name a couple of NLP applications?
- Natural language
  - Human language
  - Many languages exist in the world
- Processing
  - Historic meanings: segmentation, chunking, featurization, classification, etc
  - More precise modern definitions
    - Understanding: what a piece of text means
      - A form of evaluation is via question answering
    - Generation: generating text in a fluent and controlled manner
- NLP has several possible definitions (from specific to general)
  - Simple handling of text: segmentation, chunking, etc.
  - Understanding the meaning of text
  - All sorts of language technologies: understanding and generation, variants of problems. (this is what we use in this course)

# Why NLP?

- Natural language is a bridge towards cognitive intelligence
  - Most complicated ideas in philosophy, social sciences, natural sciences are expressed in natural language
  - Rely on natural language to think about abstract concepts
  - Most textbooks are written in text, with a small number of pictures
  - Dialog is sometimes regarded as the holy grail of AI
  - Language is a tool of communication for human, but I think it is also a tool of approaching AGI for machines.
- NLP solves real-world problems
  - An NLP industry is booming

# Web Search

- “We liked the name Alphabet because it means a collection of letters that represent language, one of humanity's most important innovations, and is the core of how we index with Google search!”
- Larry Page, co-founder of Google

# Google's Neural Machine Translation System

## Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean

Neural Machine Translation (NMT) is an end-to-end learning approach for automated translation, with the potential to overcome many of the weaknesses of conventional phrase-based translation systems. Unfortunately, NMT systems are known to be computationally expensive both in training and in translation inference. Also, most NMT systems have difficulty with rare words. These issues have hindered NMT's use in practical deployments and services, where both accuracy and speed are essential. In this work, we present GNMT, Google's Neural Machine Translation system, which attempts to address many of these issues. Our model consists of a deep LSTM network with 8 encoder and 8 decoder layers using attention and residual connections. To improve parallelism and therefore decrease training time, our attention mechanism connects

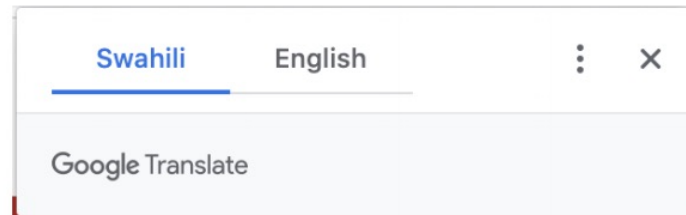
- Abstract of GNMT, translated by GNMT:
- 神经机器翻译 (NMT) 是一种用于自动翻译的端到端学习方法，具有克服传统基于短语的翻译系统的许多弱点的潜力。不幸的是，众所周知，NMT 系统在训练和翻译推理方面的计算成本都很高。此外，大多数 NMT 系统都难以处理稀有词。这些问题阻碍了 NMT 在实际部署和服务中的使用，其中准确性和速度都至关重要。在这项工作中，我们展示了谷歌的神经机器翻译系统 GNMT，它试图解决许多这些问题。我们的模型由一个深度 LSTM 网络组成，该网络具有 8 个编码器和 8 个解码器层，使用注意力和残差连接。

# Machine Translation on Low-Resource Languages



## Malawi yawapoteza mawaziri 2 kutokana na maafa ya COVID-19

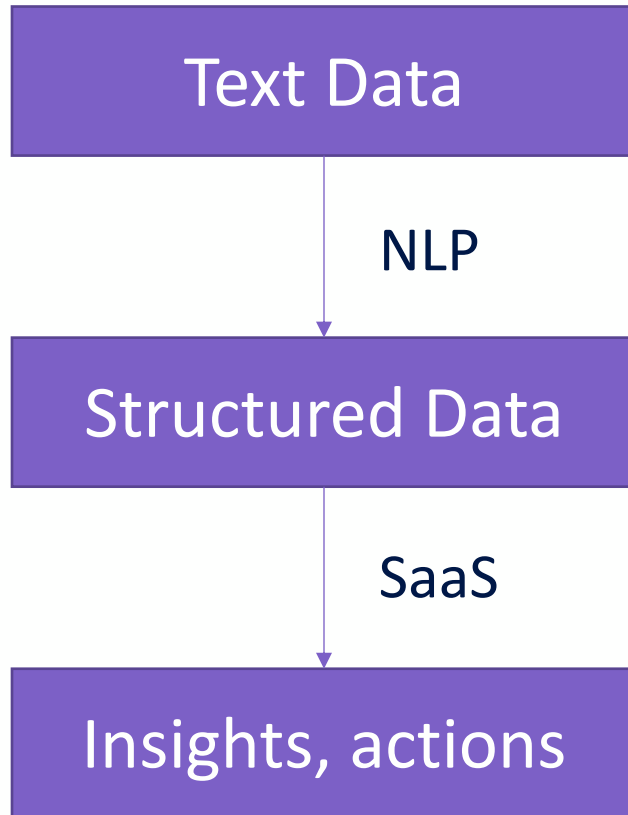
TUKO.co.ke imefahamishwa kuwa waziri wa serikali ya mitaa Lingson Belekanyama na mwenzake wa uchukuzi Sidik Mia walifariki dunia ndani ya saa mbili tofauti.



## Malawi loses 2 ministers due to COVID-19 disaster

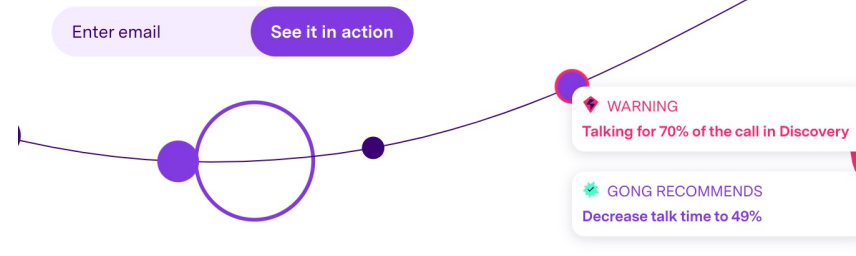
TUKO.co.ke has been informed that local government minister Lingson Belekanyama and his transport counterpart Sidik Mia died within two separate hours.

# Data Analytics



## UNLOCK REALITY. FUEL YOUR REVENUE ENGINE.

In the face of economic uncertainty, get complete visibility into all deals, team performance, and market changes. Know for sure what is happening in your deals and market, before it's too late.



Gong.io was valued at 70B dollars.

Source: <https://www.gong.io/>



# Data Analytics: Voice of Customers

What are our customers saying?

## Emerging Topics for First Notice of Loss Survey

That people are saying in Current Month compared to Last Month

Rank	Topics	Volume Increase	Sentiment Trend
1	Ease of Process	38%	⬇️
2	Follow Up	31%	⬆️
3	Processing Speed	28%	⬇️
4	Agent Helpfulness	16%	⬆️
5	Website	12%	⬇️

## Emerging Topics for Claims Call Center

That people are saying in Current Month compared to Last Month

Rank	Topics	Volume Increase
1	Clarity of Process	38%
2	Documentation Upload	25%
3	Fee Increase	18%
4	Adjuster Appointment	16%
5	Payment	14%

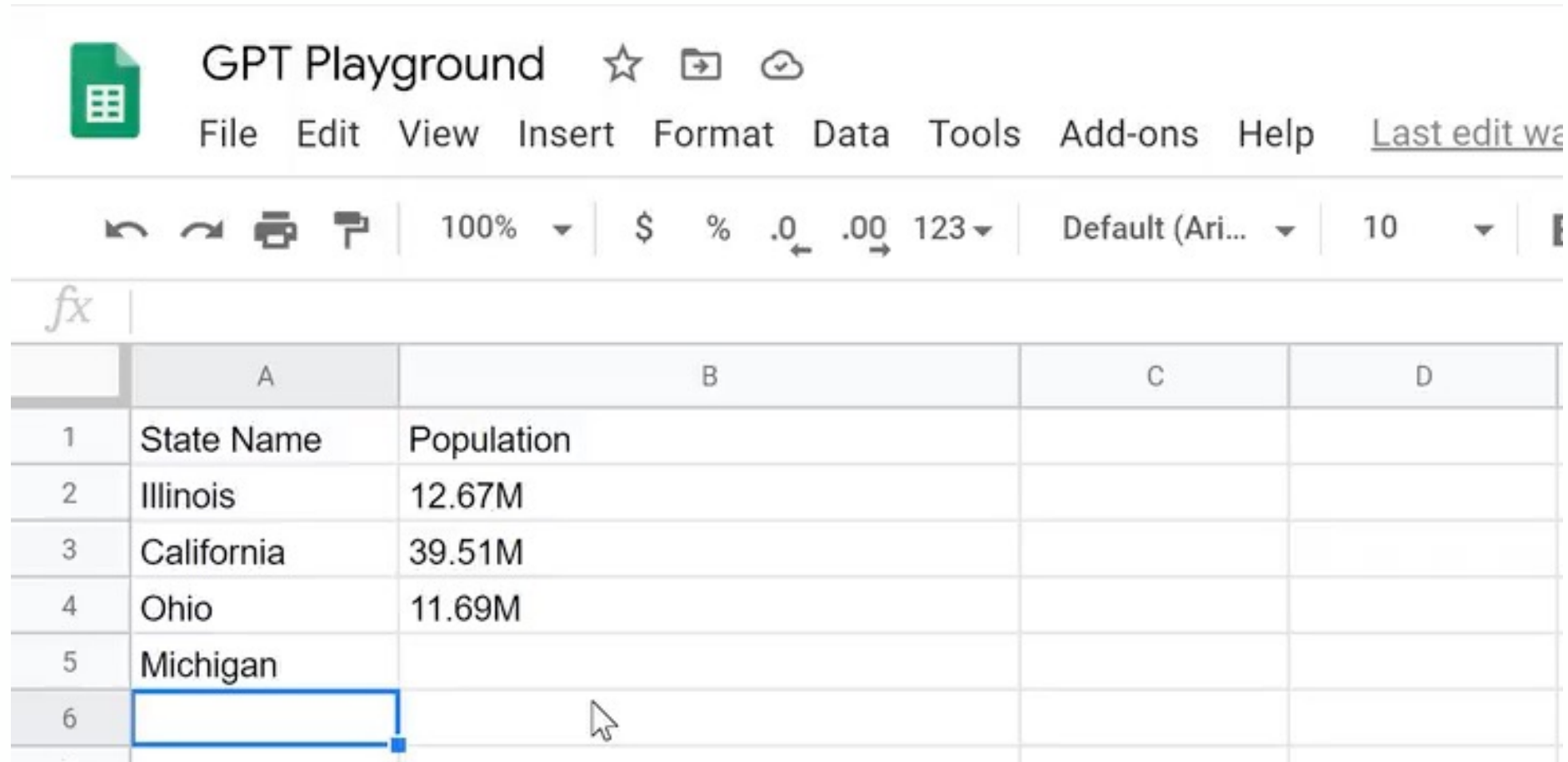
# OpenAI GPT: Natural Language Command Shell

```
[> python nlsh.py  
nlsh> █
```

```
}
```

- The developer provided GPT with a few examples of translating natural language to shell commands
- This program calls the GPT API to translate natural language into commands and execute them

# OpenAI GPT-3: Spreadsheet Completion

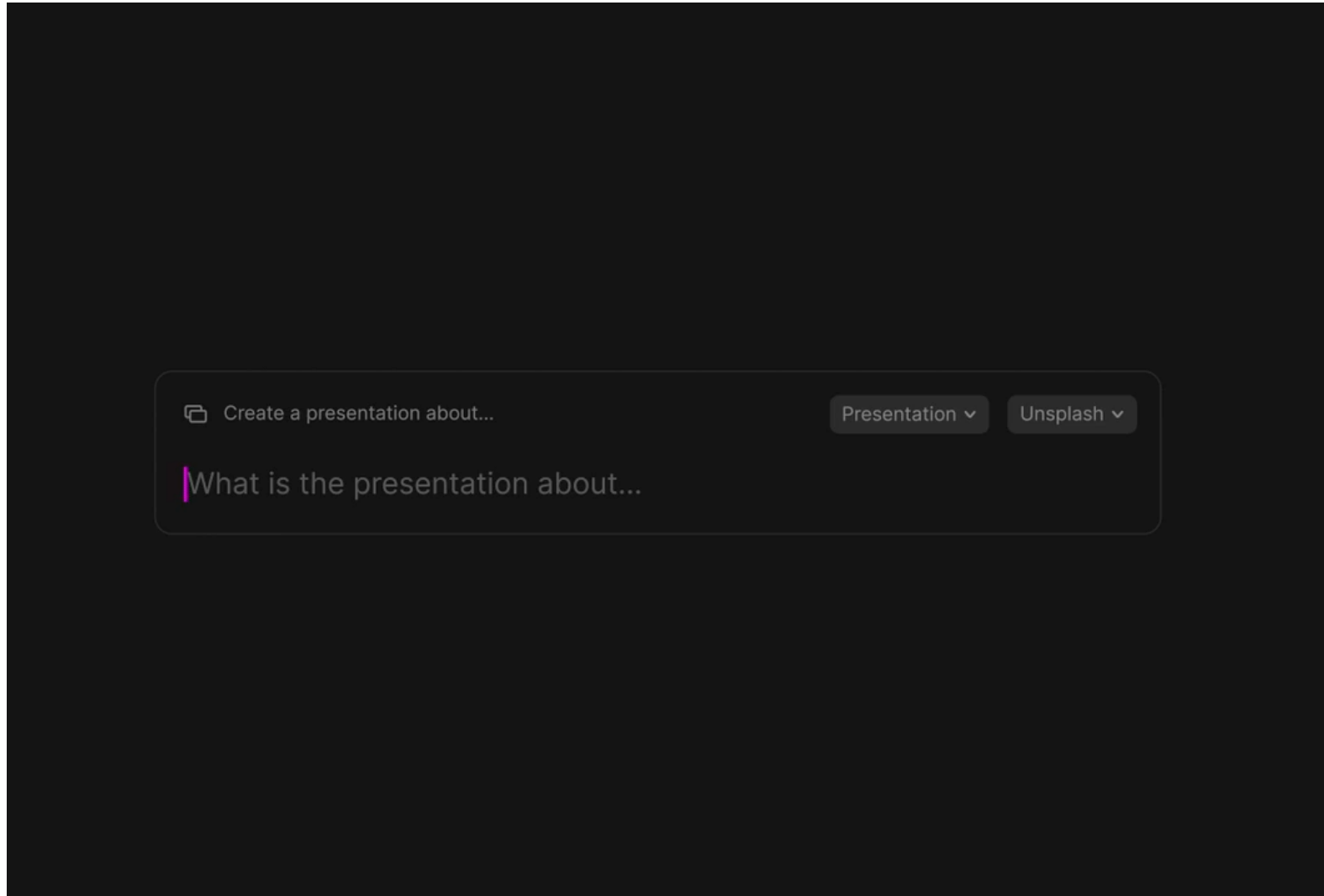


The screenshot shows the GPT Playground interface. At the top, there's a title bar with the Google Sheets icon, the text "GPT Playground", and icons for star, folder, and cloud. Below it is a menu bar with "File", "Edit", "View", "Insert", "Format", "Data", "Tools", "Add-ons", "Help", and a link "Last edit wa". A toolbar contains icons for undo, redo, print, and copy, along with settings for zoom (100%), currency (\$), percentage (%), decimal places (.0, .00), thousands separator (123), font (Default (Ari...), and font size (10). Below the toolbar is a formula bar with "fx" and an empty input field. The main area is a spreadsheet with columns A, B, C, and D, and rows 1 through 6. The data in the spreadsheet is as follows:

	A	B	C	D
1	State Name	Population		
2	Illinois	12.67M		
3	California	39.51M		
4	Ohio	11.69M		
5	Michigan			
6				

A mouse cursor is pointing at the empty cell in row 6, column A, which is highlighted with a blue border.

# Tome: Generating Slides



# Two AIs Talk About Becoming Human



Source: <https://www.youtube.com/watch?v=jz78fSnBG0s>

# Writing Assistants: an AI21 Demo

**“When machines become  
thought partners”**

Source: <https://www.ai21.com/>

**GPT-4**

# “A Wave of Billion-Dollar Language AI Startups is Coming”

- Horizontal technology
  - Cohere, Hugging Face, AI21, Primer, Inflection AI
- Search
  - You.com, ZIR AI, Algolia, Constructor.io, Hebbia, Twelve Labs
- Writing assistants
  - Grammarly, Textio, LitLingo, Writer, CopyAI
- Language translation
  - BLANC, KUDO, Lilt, NeuralSpace
- Sales intelligence
  - Gong, Aircover, Wingman
- Chatbot tools and infrastructure
- Internal employee engagement
- Conversational voice assistants
- Contact centers
- Content moderation
- Healthcare

We are probably witnessing an NLP hype. This means there will likely be a bubble burst, but also startups grown into NASDAQ-listed companies. And in the end, the entire industry realizes the maximum value of NLP technologies.

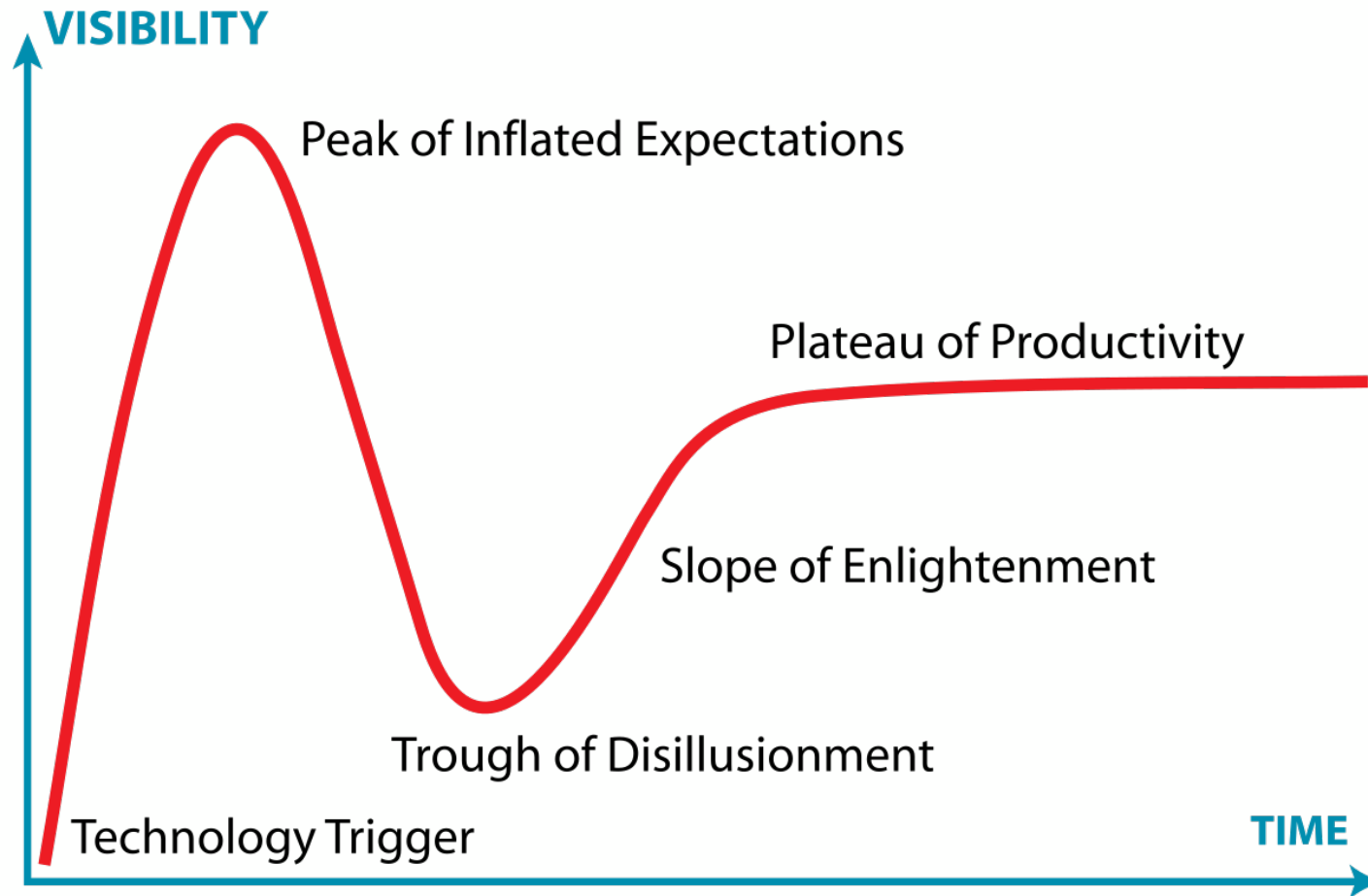
Tech giants are also heavily investing in language AI. - e.g., Meta AI and its LLaMa efforts.

Source:

<https://www.forbes.com/sites/robtoews/2022/03/27/a-wave-of-billion-dollar-language-ai-startups-is-coming/?sh=31da7bb22b14>



# Gartner Hype Cycle



Question: where is NLP?

# Outline

- Course overview
- NLP intro
- **Sub-languages**
- Probabilistic view of natural language
- Word vectors

# Words

- The word “word” is ambiguous. It might refer to “word type” or “word token”
- Word type
  - Each entry in the vocabulary
  - Consider “I like cats. Cats are cute”. The two “cats” correspond to the same word type.
- Word token
  - Each occurrence of a word type
  - Consider “I like cats. Cats are cute”. The two “cats” correspond to different word tokens.
- More examples
  - When we say, “for each word in the vocabulary”, we usually mean word type
  - When we say, “for each word in the sentence”, we usually mean word token

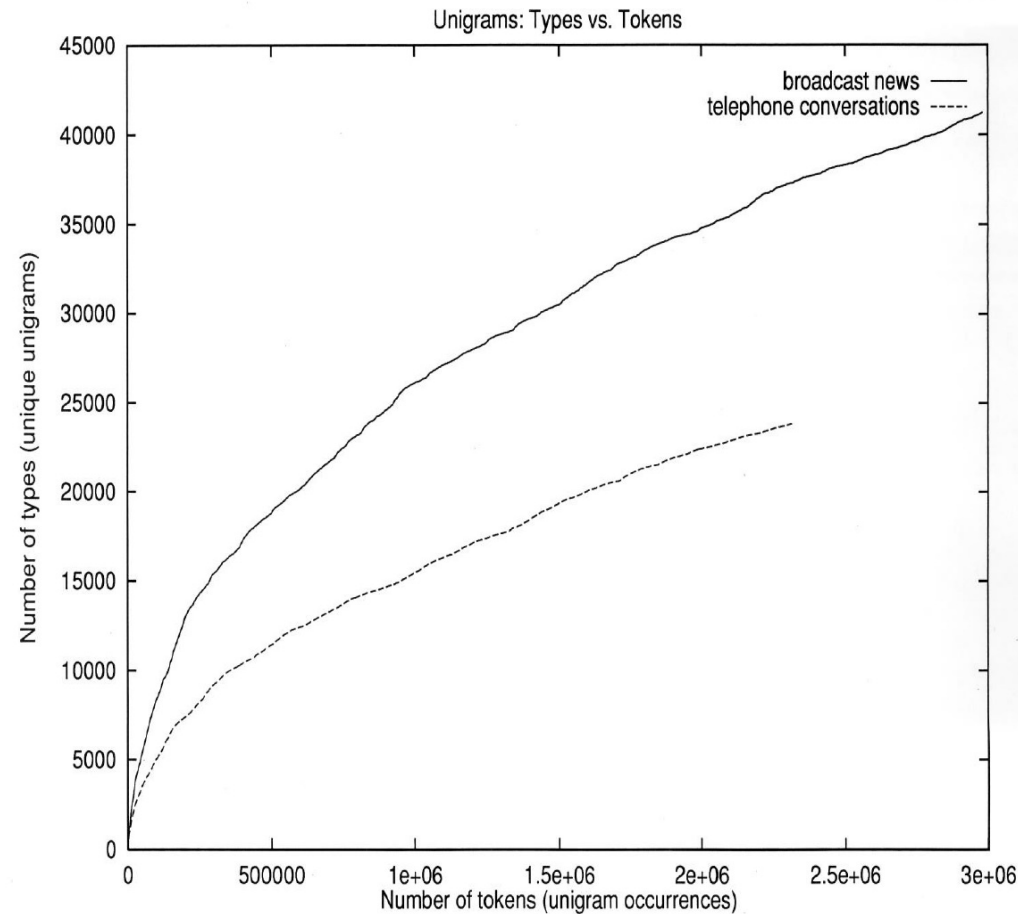
# Heterogeneity of Language

- Heterogeneity
  - Written vs spoken
  - Period, genre, register (level of formality), domain
  - topic (hierarchy), speaker, audience
- When you try to solve an NLP task, pay attention to the heterogeneity of language. It could explain many results.

# Sub-Language Examples

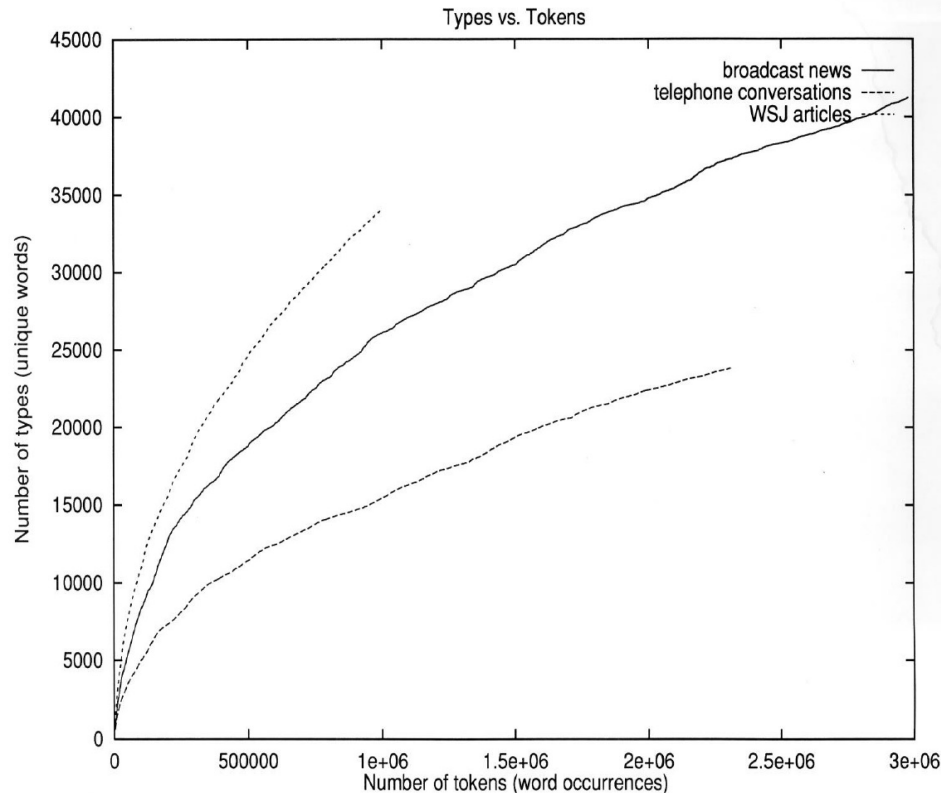
- “Wall Street Journal” Corpus (WSJ):
  - Newspaper articles, 1988-1992
  - Written English, rich vocabulary (leaning towards finance)
- “Switchboard” Corpus (SWB):
  - Transcribed spoken conversations
  - over the telephone
  - Proscribed topic (one of 70)
  - 1990’s
- “Broadcast News” Corpus (BN):
  - Transcribed TV/Radio News programs
  - Spoken, but somewhat scripted

# Unigram Type-Token Curve (BN vs SWB)



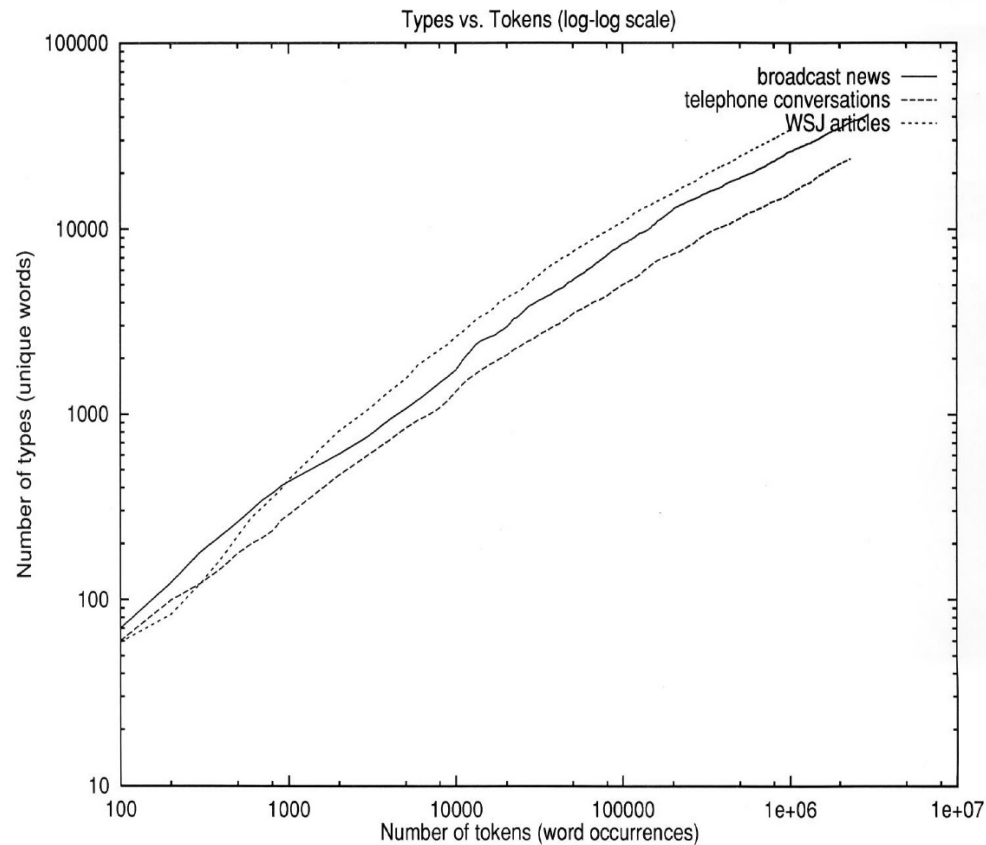
Question: what patterns do you see in the figure?

# Unigram Type-Token Curve (BN vs SWB vs WSJ)



- The type-token ratio of WSJ is much higher than that of BN and SWB
- Wall Street Journal (WSJ) is an American newspaper specialized in business and financial news.
- Broadcast news vs written news

# Unigram Type-Token Curve (BN vs SWB vs WSJ, log scale)



- Linear correlation at log scale



# Head of Word Frequency List

- Counts per 1,000 tokens
- This is also a good way to get a sense of what a dataset looks like

WSJ		BN		SWB	
THE	49	</S>	62	I	38
</S>	42	THE	49	AND	34
TO	24	TO	27	<SIL>	31
OF	24	AND	25	THE	28
A	22	A	22	YOU	26
AND	19	OF	21	UH	26
IN	19	IN	17	A	24
THAT	9	THAT	16	TO	23
FOR	9	IS	13	THAT	20
IS	8	YOU	12	IT	17
ONE	7	I	12	OF	17
ON	6	IT	10	KNOW	16
POINT	5	FOR	8	YEAH	14
AS	5	THIS	8	IN	12
SAID	5	ON	7	+NOISE+	12
WITH	5	HAVE	6	THEY	10
IT	5	ARE	6	UH-HUH	10
FIVE	5	WE	6	HAVE	10
TWO	5	THEY	6	BUT	9
DOLLARS	5	BE	6	SO	8
AT	5	WITH	6	IT'S	8
MR.	5	BUT	5	IS	8
BY	5	WAS	5	WE	8

# Zipf's Law: Frequency vs Rank (log scale)

- Brown corpus
- Zipf's law:
  - Let  $N$  be the number of word types,  $k$  be the rank of each word type, and  $s$  be a parameter
  - Zipf's law predicts the normalized frequency of the word of rank  $k$  as

$$f(k; s, N) = \frac{1/k^s}{\sum_{n=1}^N (1/n^s)}$$

- With  $s=1$  (classic Zipf's law),

$$f(k) \propto 1/k \text{ or } \log f(k) = \text{const} - \log k$$

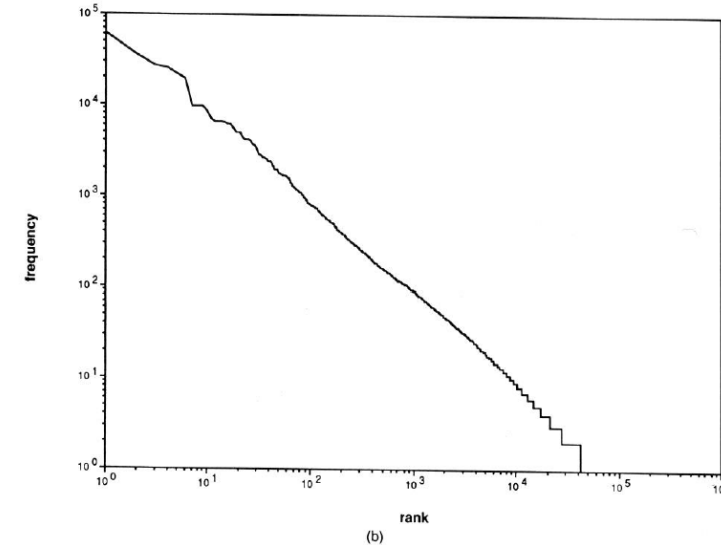


Figure 4-1(cont) (b) The same data plotted on logarithmic scales.

The second common word will occur half as often as the first, the third most common word will occur 1/3 as often as the first, the  $k$ -th common word will occur  $1/k$  as often as the first.

# Comparison with Theoretical Zipf's Law Distribution

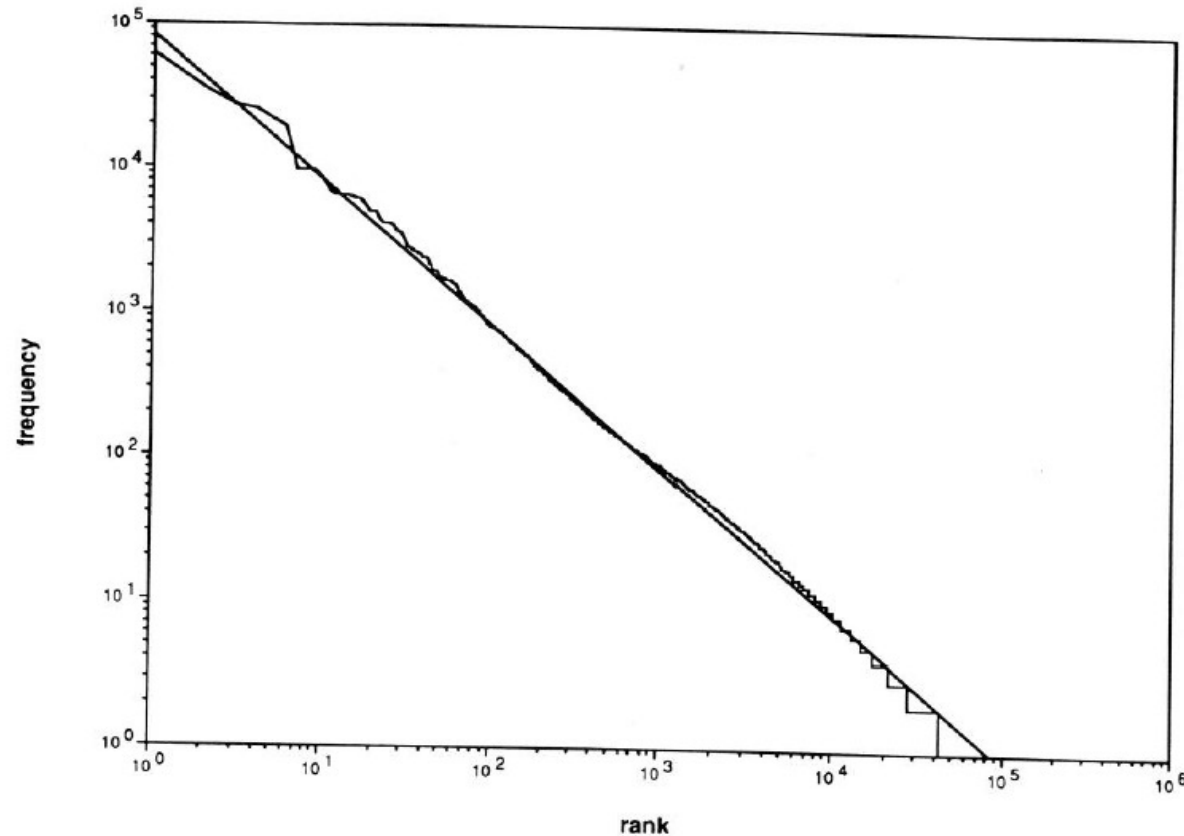


Figure 4-4 Word-frequency data from the Brown corpus with Zipf distribution (straight line).

- Human language follows a heavy-tailed distribution.
- Human language can be reasonably modeled by Zipf's law with  $s=1$ .

# Outline

- Course overview
- NLP intro
- Sub-languages
- Probabilistic view of natural language
- Word vectors

# A Probabilistic View of Natural Language

- (Almost) everything can be viewed as to estimate some probability
  - $P(\text{word})$
  - $P(\text{sentence})$
  - $P(\text{paragraph})$
  - $P(\text{document})$
  - Thought process, translation, sentiment, etc...
- The question is how to estimate such a targeted probability

# Surface Form

- Surface form: written form of text language

Marry saw Jane standing with a telescope by the bank. She waved to her.

Question: can you translate this sentence?

# Surface Forms and Hidden Forms

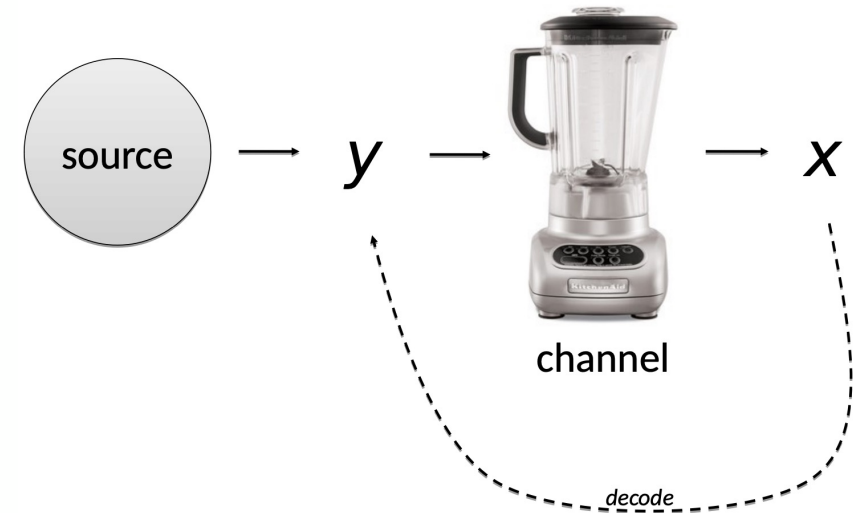
- Surface forms: written form of text language
  - Hidden forms: pronoun resolution, PP attachment, POS tagging, word sense disambiguation
  - Given an unambiguous hidden form, there might be multiple surface forms that express it.
    - Not a one-to-one relationship between hidden forms and surface forms
  - Consider joint probability  $P(\text{surface, hidden})$  or  $P(S, H)$ . Solving  $P(S, H)$  means solving most of NLP problems.
  - Example 1: computational linguistics maps  $S$  to  $H$ 
    - POS tagging, pronoun resolution, parsing
- $$H^* = \arg \max_H P(H|S) = \arg \max_H \frac{P(H, S)}{P(S)} = \arg \max_H P(H, S)$$
- Example 2: language modeling
    - Marginalize over  $H$  to obtain  $P(S)$

# Source Channel Paradigm

- Given a **source**, a **hidden variable**  $Y$  is generated. A **channel** then generates a **seen variable**  $X$  given the hidden variable  $Y$ .
- Example: surface form can be viewed as  $X$ , and hidden form can be viewed as  $Y$
- When  $X$  is a noisy version of  $Y$ , this is often called the **noisy channel model**
- Prior distribution  $P(Y)$ , likelihood  $P(X | Y)$
- Goal is to deduce the hidden variable:

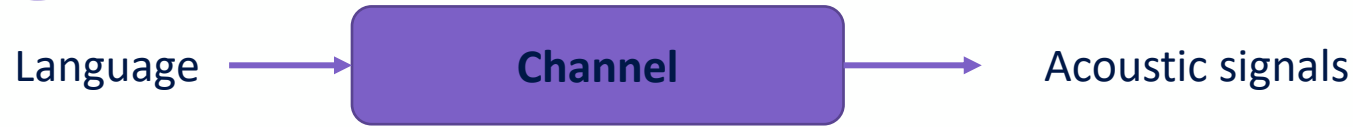
$$Y^* = \arg \max_Y P(Y|X) = \arg \max_Y \frac{P(Y)P(X|Y)}{P(X)} = \arg \max_Y P(Y)P(X|Y)$$

- Also called Bayes classifier





# Automatic Speech Recognition (ASR) Example



- Denote language as  $L$  and acoustic signals as  $A$
- ASR is to deduce

$$L^* = \arg \max_L P(L|A) = \arg \max_L \underbrace{P(L)}_{\text{Language modeling}} \underbrace{P(A|L)}_{\text{Acoustic modeling}}$$

**Speech recognition = language modeling + acoustic modeling + search**

In our previous example, language is a seen variable, but in this ASR example, language is a hidden variable. The notions of likelihood and posterior are not fixed and should be decided depending on the problem.

# Topic Detection



- Denote topic as  $T$  and language as  $L$
- Goal is to deduce (detect) topic from language:

$$T^* = \arg \max_T \underbrace{P(T|L)}_{\text{Discriminative model}} = \arg \max_T \underbrace{P(T)}_{\text{Topic distribution}} \underbrace{P(L|T)}_{\text{Topic-dependent generative model}}$$

Discriminative model

Topic  
distribution

Topic-dependent  
generative model

Question: why not just directly estimate  $P(T | L)$ ?

# Outline

- Course overview
- NLP intro
- Sub-languages
- Probabilistic view of natural language
- **Word vectors**

# Definition of Meaning in Webster

- Definition of “meaning” in Webster dictionary
  - The idea that is represented by a word, phrase, etc.
  - The idea that a person wants to express by using words, signs, etc.
  - The idea that is expressed in a work of writing, art, etc.

# Definition of Meaning in Wikipedia

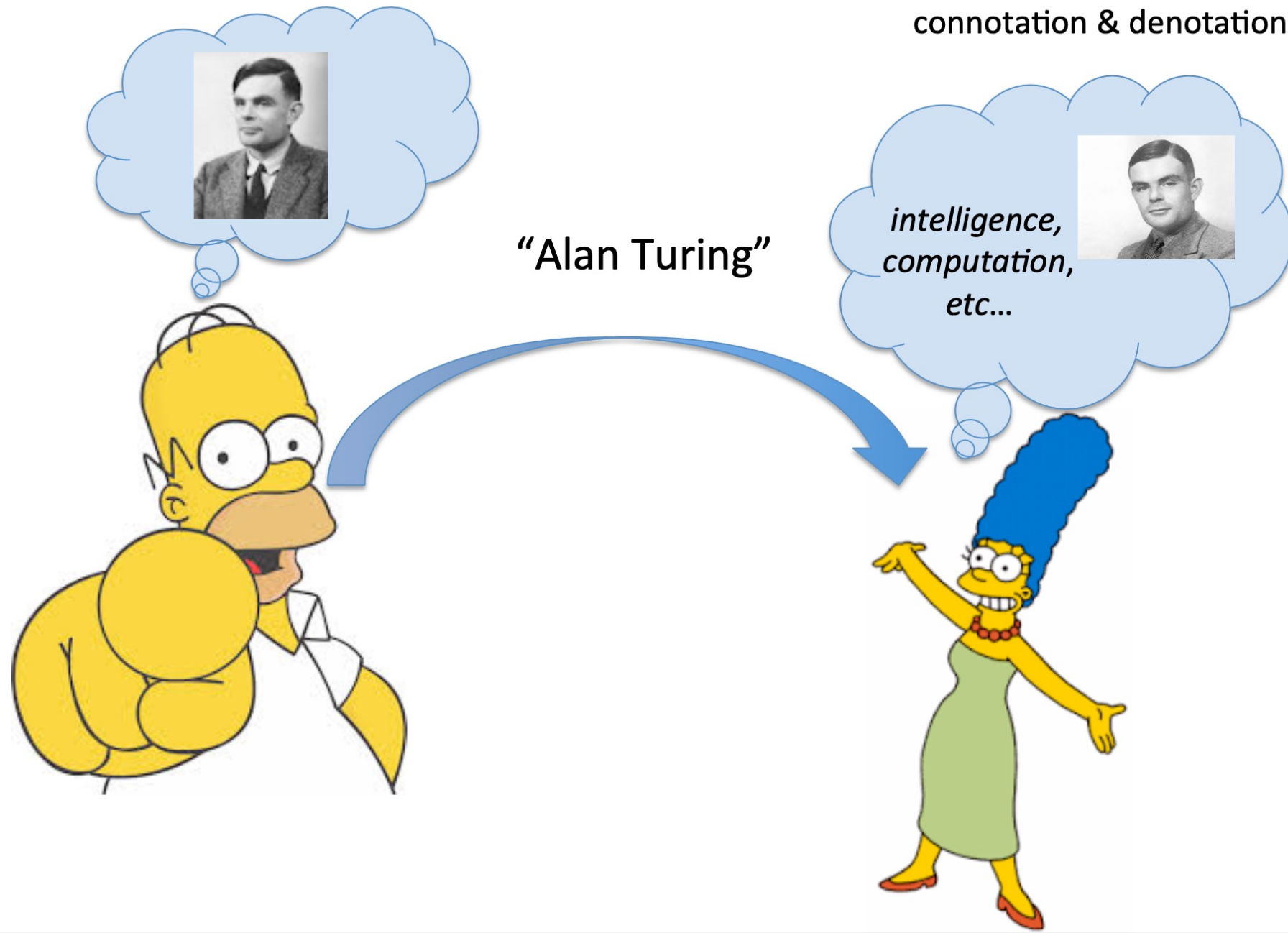
- In linguistics, **meaning** is what the source or sender expresses, communicates, or conveys in their message to the observer or receiver, and what the receiver infers from the current context.
- Semantics is the study of meaning. It focuses on the relation between **signifiers**, like words, phrases, signs, and symbols, and what they stand for, their denotation.
- A **denotation** is a translation of a sign to its meaning, more exactly, to its literal meaning. Denotation is sometimes contrasted to connotation, which translates a sign to meanings associated with it.
- A **connotation** is a commonly understood culture or emotional association that some word or phrase carries, in addition to the word's or phrase's explicit or literal meaning, which is its denotation.
- Cited from Wikipedia

# An Example

Intention/goal:  
communicate denotation

Message:  
produce signs

Interpretation:  
access & build  
connotation & denotation



# How do we know when someone “understands” the “meaning” of a sentence/word?

- Logical criterion: they can write down a formal statement that accurately describes the world's state that the sentence describes
  - Formal logic notation: Aristotle, Frege, Russell...
- Operational criterion: they can act/respond appropriately to the sentence
  - For example, answer a question
  - Most machine learning based NLP systems are evaluated in this way

# How do we have usable meanings in a computer?

- How to represent word types?
- Conventional solution: use a thesaurus containing lists of synonym sets and hypernyms (“is a” relationships)
  - E.g., WordNet

*e.g., synonym sets containing “good”:*

```
from nltk.corpus import wordnet as wn
poses = { 'n': 'noun', 'v': 'verb', 's': 'adj (s)', 'a': 'adj', 'r': 'adv' }
for synset in wn.synsets("good"):
    print("{}: {}".format(poses[synset.pos()],
                          ", ".join([l.name() for l in synset.lemmas()])))
```

```
noun: good
noun: good, goodness
noun: good, goodness
noun: commodity, trade_good, good
adj: good
adj (sat): full, good
adj: good
adj (sat): estimable, good, honorable, respectable
adj (sat): beneficial, good
adj (sat): good
adj (sat): good, just, upright
...
adverb: well, good
adverb: thoroughly, soundly, good
```

*e.g., hypernyms of “panda”:*

```
from nltk.corpus import wordnet as wn
panda = wn.synset("panda.n.01")
hyper = lambda s: s.hypernyms()
list(panda.closure(hyper))
```

```
[Synset('procyonid.n.01'),
Synset('carnivore.n.01'),
Synset('placental.n.01'),
Synset('mammal.n.01'),
Synset('vertebrate.n.01'),
Synset('chordate.n.01'),
Synset('animal.n.01'),
Synset('organism.n.01'),
Synset('living_thing.n.01'),
Synset('whole.n.02'),
Synset('object.n.01'),
Synset('physical_entity.n.01'),
Synset('entity.n.01')]
```



# Problems with Resources Like WordNet

- Great as a resource but missing nuance
  - E.g., “proficient” is listed as a synonym for “good”. This is only correct in some contexts
- Missing new meanings of words
  - Impossible to keep up-to-date
- Subjective
- Requires human labor to create and adapt
- Can’t compute accurate word similarity

# Representing Words as Discrete Symbols

- Conventional approach
- Each word type is represented as a one-hot vector
  - motel = [0 0 0 0 0 0 0 0 0 0 0 0 1 0]
  - hotel = [0 0 0 0 0 0 1 0 0 0 0 0 0 0]
  - Dimension of vector equals number of word types in vocabulary
  - Each vector has a single 1 and the rest are 0

# Problems with Words as Discrete Symbols

- Example: if user searches for “Hawaii motel”, we would like to match documents containing “Hawaii hotel”.
- However, the representations of hotel and motel are orthogonal
  - motel = [0 0 0 0 0 0 0 0 0 0 0 0 1 0]
  - hotel = [0 0 0 0 0 0 1 0 0 0 0 0 0 0]
  - It is not straightforward to compute the similarity between the two words
  - Using WordNet might not work well
    - Incompleteness
    - Not a notion of similarity
- How about encode similarity into representations?
  - Non-orthogonal

# Representing Words By Their Context

- Distributional semantics
  - A word's meaning is given by the words that frequently appear close-by
  - One of the most successful ideas of modern statistical NLP
- When a word  $w$  appears in a text, its context is the set of words that appear nearby (within a fixed-size window)
- Use many contexts of  $w$  to build a representation of  $w$

*...government debt problems turning into **banking** crises as happened in 2009...*  
*...saying that Europe needs unified **banking** regulation to replace the hodgepodge...*  
*...India has just given its **banking** system a shot in the arm...*

These **context words** will represent **banking**

# Word Vectors

- We will build a dense vector for each word, chosen so that it is similar to vectors of words that appear in similar contexts
- Word vectors are also called word embeddings

$$\textit{banking} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

# Word Meaning as a Word Vector

- Visualization



# Alternative Views of Word Vectors: Linear Transformation of One-Hot Vectors

- One-hot vector  $\mathbf{x} = [0, \dots, 1, \dots, 0] \in \mathbb{R}^V$

- Dense vector

$\mathbf{w} = \mathbf{x}\mathbf{E} \in \mathbb{R}^d$ , where  $\mathbf{E} \in \mathbb{R}^{V \times d}$  is a transformation matrix

- Learn the transformation matrix
  - This is also called the embedding matrix or lookup table
- Alternative view 1: use the one-hot vector to “look up” the dense vector in the lookup table
- Alternative view 2: apply a linear layer on top on the one-hot vectors
- Dimension is often reduced
- How to learn the embedding matrix is a key problem

# Localist Representations

- The simplest way to represent things with neural networks is to dedicate one neuron to each thing
  - Big red rectangle = (0, 0, 0, 0, 1, 0, 0, 0)
  - Small red ellipse = (0, 0, 1, 0, 0, 0, 0, 0)
  - Big blue rectangle = (1, 0, 0, 0, 0, 0, 0, 0)
- Advantages
  - Easy to understand
  - Easy to code by hand
  - Easy to learn
  - Easy to associate with other representations or responses
- Disadvantages
  - Very inefficient whenever the data has componential structure



# Distributed Representations

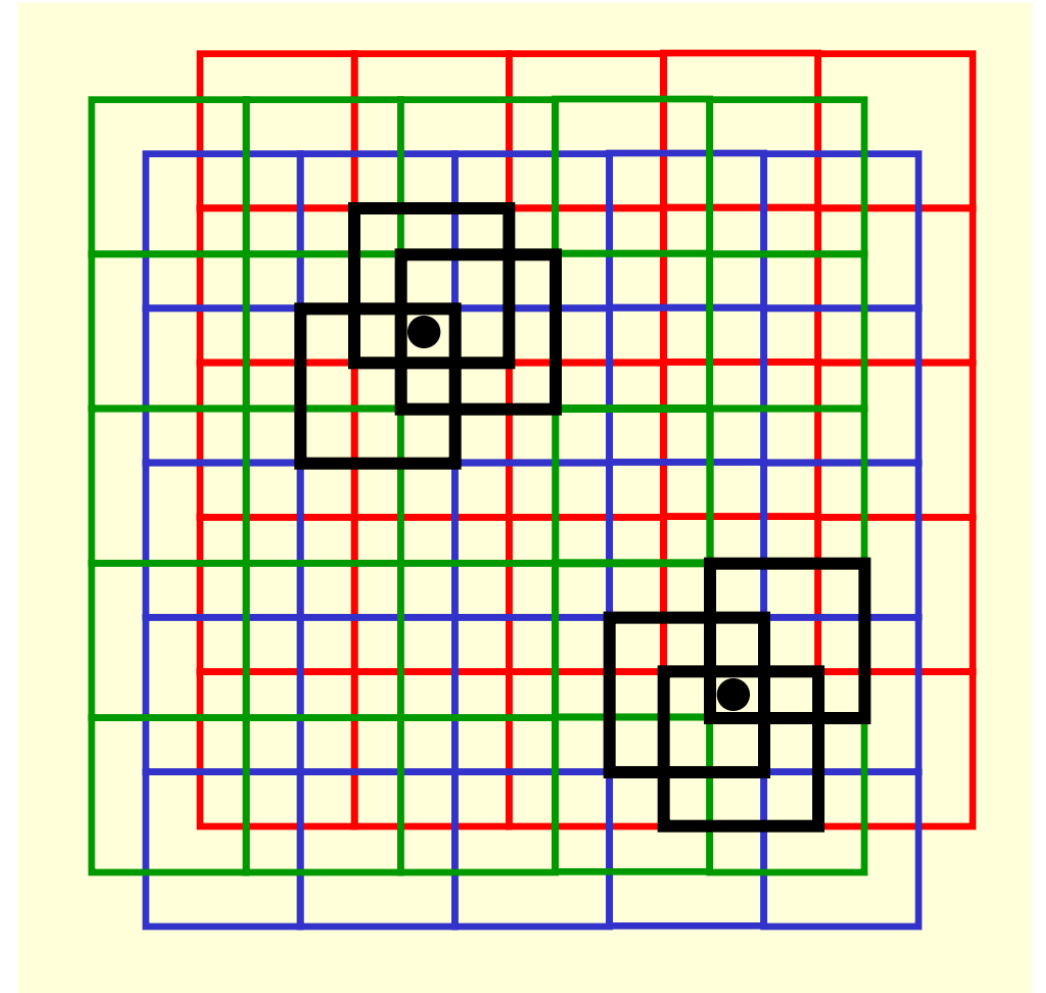
- Componential structures
  - Consider there are many objects. Each object has many properties like shape, color, size. How to represent each object?
- Consider objects with three properties
  - Big/small, red/blue/, rectangle/ecclipse
  - 8 possible combinations in total
- Distributed representations
  - (big/small, red/blue, rectangle/ecclipse): each dimension represents a property
  - Big red rectangle = (1, 1, 1)
  - Small red ecclipse = (0, 1, 0)
  - Big blue rectangle = (1, 0, 1)

# Distributed Representations

- “Distributed representation” means a many-to-many relationship between two types of representations (such as concepts and neurons)
  - Each concept is represented by many neurons
  - Each neuron participates in the representation of many concepts
- Distributed representation is arguably one of the two foundational ideas of deep learning (the other is nonlinearity).

# Coarse Coding: A Case Study

- Large cells and fine cells
- Use three overlapping arrays of large cells to get an array of fine cells
  - If a point falls in a fine cell, code it activating 3 coarse cells
  - Each cell is represented by a combination of a red, a green, and a blue array
- This is more efficient than using a neuron for each fine cell
  - It needs 3 arrays to represent a cell
  - Each array participates in representing 3x3 cells
  - Overall, using arrays is 3x more efficient



# How to create distributed representations?

- There is more than one way
- Method 1: use logical association
  - As in the above “big red rectangle example”
  - Usually binary representations
- Method 2: feature learning
  - Learn it in a neural network
  - As in word embeddings
  - Usually continuous representations
- Tradeoff about dimensionality
  - Higher dimension: less efficient, more sparse
  - Lower dimension: less capacity
- Everything can be vectorized: word2vec, doc2vec, tweet2vec, node2vec, etc

# Distributed vs Distributional

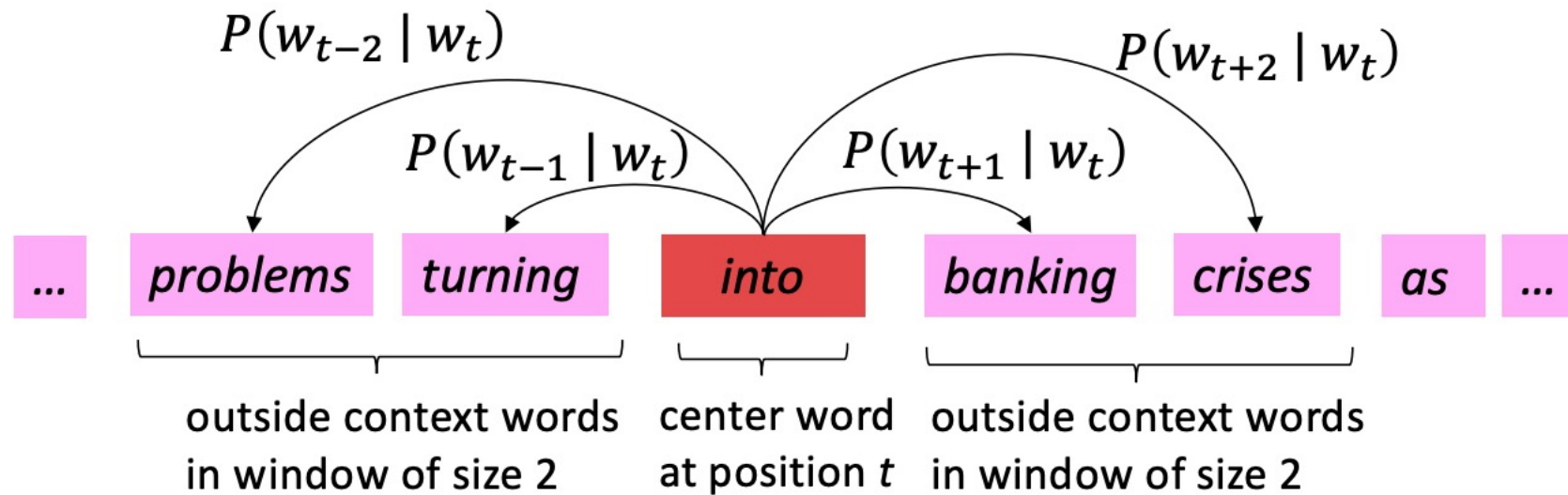
- Distributional representations
  - Words are similar if they appear in similar contexts; distribution of words indicative of usage
  - In contrast: non-distributional representations created from lexicon resources such as WordNet, etc.
- Distributed representations
  - Something is represented by a compact vector of values, each representing activations
  - In contrast: localist representations, one-hot vectors
- Word vectors are usually **both** distributional and distributed

# Word2vec: Overview

- Word2vec is a framework for learning word vectors
- Idea:
  - We have a large corpus of text
  - Every word in a fixed vocabulary is represented by a vector
  - Go through each position  $t$  in the text, which has a center word  $c$  and context words  $o$
  - Use the similarity of the word vectors for  $c$  and  $o$  to calculate the probability of  $o$  given  $c$  (or vice versa)
  - Keep adjusting the word vectors to maximize this probability

# Word2vec Overview

- Example windows and process for computing  $P(w_{t+j} | w_t)$



# Word2vec Overview

- Example windows and process for computing  $P(w_{t+j} | w_t)$

