

Deep Learning for Natural Language Processing

Lecture 1 – Introduction to NLP and Its Applications

Zengchang Qin (PhD)

Intelligent Computing and Machine Learning Lab

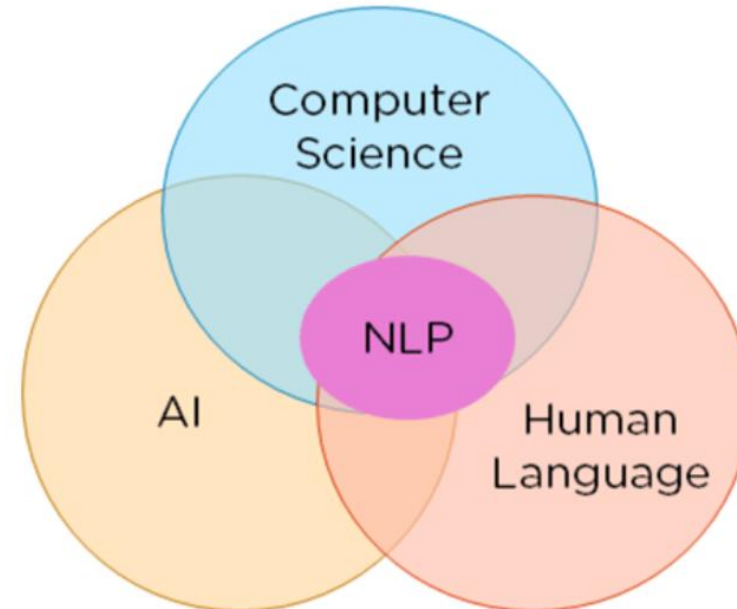
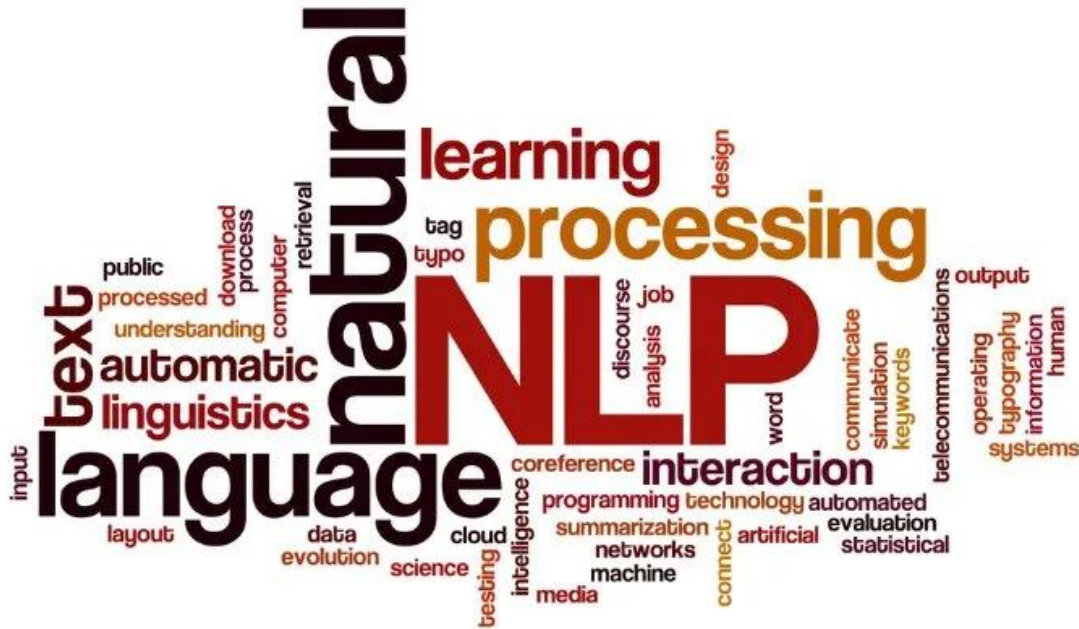
Beihang University

zengchang.qin@gmail.com

1.1 What is Natural Language Processing?

Natural Language Processing

- Natural language processing (NLP) can be defined as the ability of a machine to analyze, understand, and generate **human language**.
- Natural Language Processing bridges the gap between computers, AI, and computational linguistics.



Linguistics (Human Language)

- ❑ The scientific study of human language (structure, evolution, meaning, and usage).
- ❑ Syntax, semantics, phonetics, morphology, sociolinguistics, psycholinguistics.
- ❑ Understand how languages work, evolve, and are used by humans.

Considered the **founder** of modern linguistics, **Noam Chomsky** is one of the most cited scholars in modern history. Chomsky introduced the Chomsky hierarchy, **generative grammar** and the concept of a **universal grammar**, which underlies all human speech and is based in the innate structure of the mind/brain.



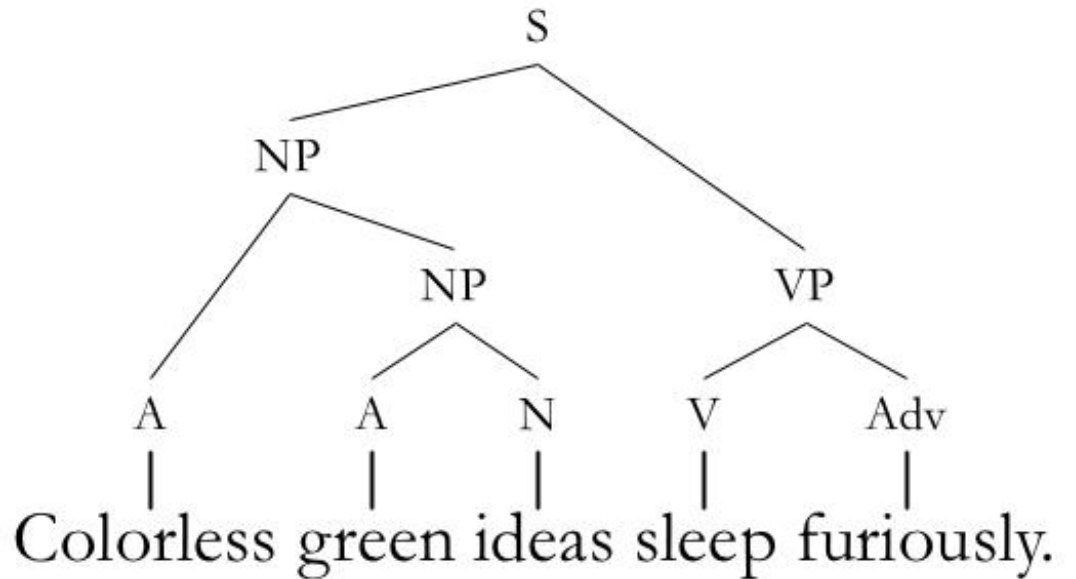
Noam Chomsky, *Syntactic Structures*, 1957.

<https://www.ling.upenn.edu/courses/ling5700/Chomsky1957.pdf>

Generative Grammar

❑ A simple example of phrase structure rules can be seen in the following system:

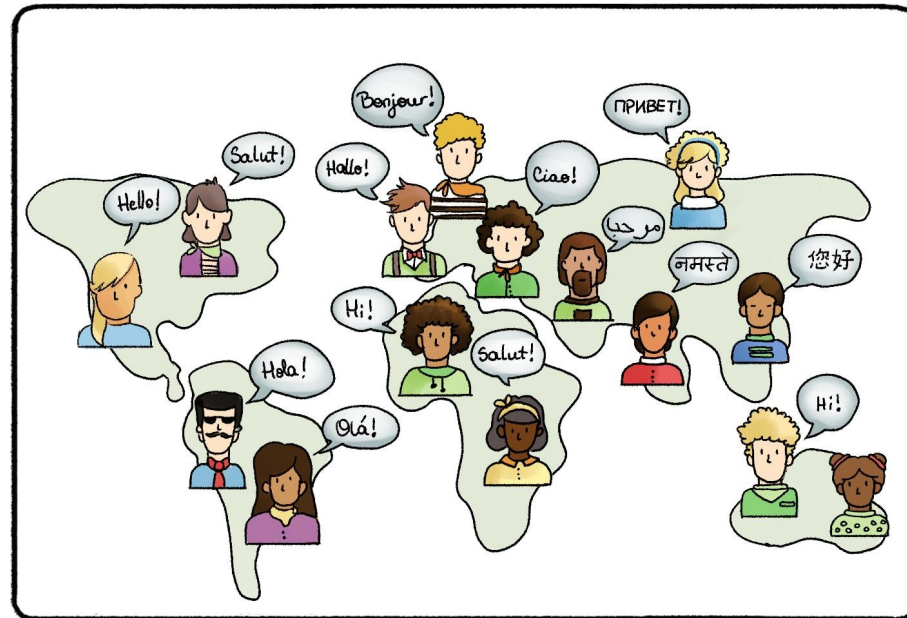
$S \rightarrow NP VP$ (A sentence is composed of a noun phrase and a verb phrase)
 $NP \rightarrow Det N$ (A noun phrase consists of a determiner and a noun)
 $VP \rightarrow V NP$ (A verb phrase consists of a verb and a noun phrase)
 $Det \rightarrow The$ (A determiner is "The")
 $N \rightarrow \text{man, ball}$ (A noun is "man" or "ball")
 $V \rightarrow \text{will hit}$ (A verb is "will hit")



❑ Using these rules, you can generate sentences like "The man will hit the ball."

Universal Grammar

- ❑ Chomsky's universal grammar is a theory that proposes humans are born with an innate ability to acquire language, guided by universal principles and parameters. This theory helps explain how children can learn complex languages rapidly and efficiently, even with limited input



Computing Languages (Programming Languages)

- ❑ Formal systems to instruct computers (e.g., Python, Java, C++).
- ❑ Syntax, semantics, compilers, algorithms.
- ❑ Create tools for writing efficient, **unambiguous** software.
- ❑ Designing syntax rules, compilers, and debugging tools.
- ❑ Focus on efficiency, scalability, and correctness.

```
1  def golden_ratio(n):
2      fib = np.ones(n)
3      golden = np.ones(n)
4
5      for i in range(2, n):
6          fib[i] = fib[i - 1] + fib[i-2]
7          golden[i] = fib[i - 1]/fib[i - 2]
8      return golden
9
10 golden_series = golden_ratio(20)
11
12 for i in range(np.size(golden_series)):
13     print(golden_series[i])
```


What is NLP About?

- ❑ Bridging human language and computers using computational techniques.
- ❑ Tasks: Translation, sentiment analysis, chatbots, speech recognition.
- ❑ Enable machines to understand, interpret, and generate human language.

Key Differences:

Aspect	Linguistics	Computing Languages	NLP
Primary Goal	Study human language	Instruct computers	Process human language with AI
Root Discipline	Humanities/Social Science	Computer Science	Interdisciplinary (CS + Linguistics)
Tools/Methods	Theoretical analysis, fieldwork	Compilers, syntax parsers	Machine learning, neural networks
Output	Theories about language	Software, algorithms	Language models, chatbots

- ❑ Develop computer science (e.g., machine learning, algorithms), uses programming languages as tools, to tackle the tasks in applications like ChatBot, Virtual assistants (Siri), machine translation (Google Translate), spam detection and etc.

Why NLP is Hard?

- ❑ Raw text data in English or other languages is an example of **unstructured data**. This kind of data does not fit into a relational database and is hard to interpret with computer programs.
- ❑ In many AI areas, Data Mining (Big Data), Computer Vision, we have structural data can be relatively easily handled.

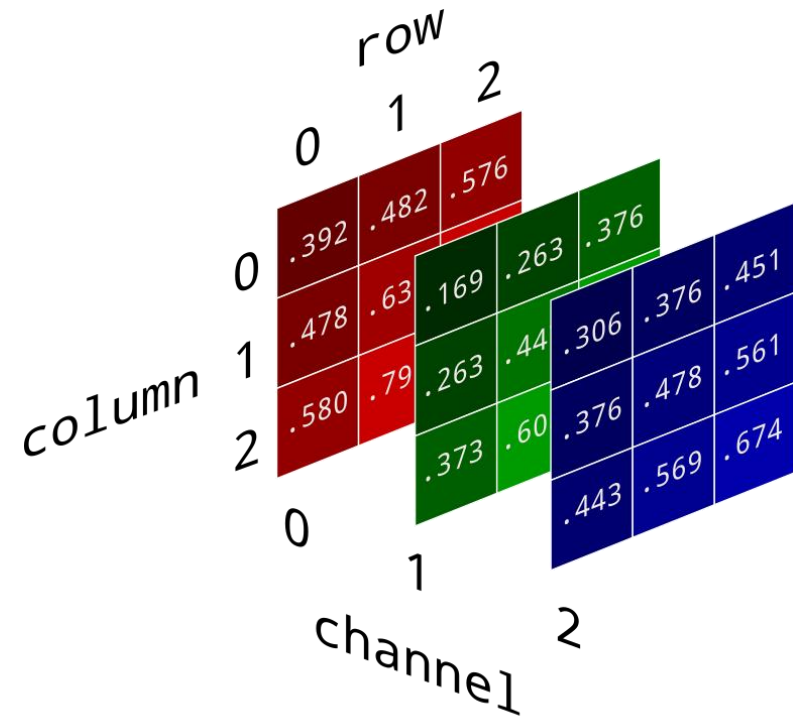
Table of baby-name data
(baby-2010.csv)

name	rank	gender	year
Jacob	1	boy	2010
Isabella	1	girl	2010
Ethan	2	boy	2010
Sophia	2	girl	2010
Michael	3	boy	2010

Field names

One row (4 fields)

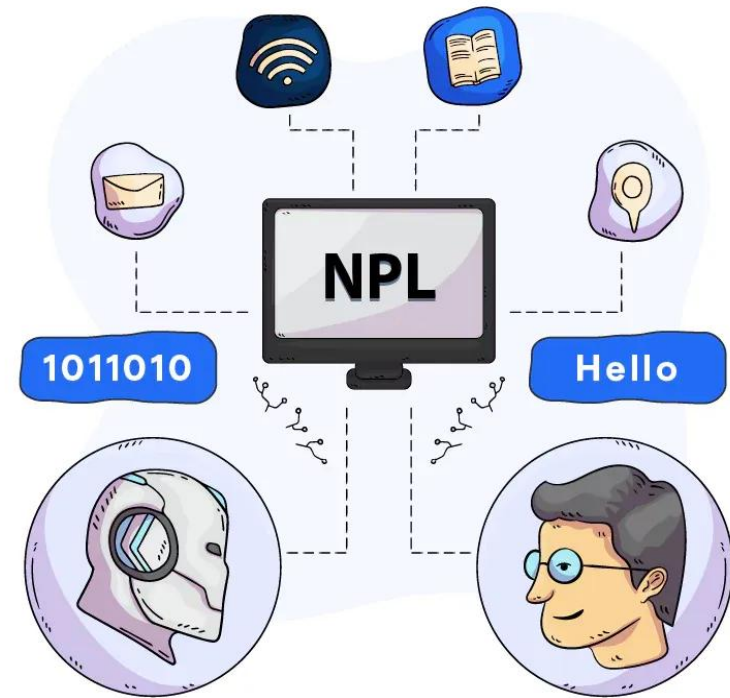
2000 rows all told



Approaches Towards NLP - Symbolic

- ❑ **Symbolic (Rule-Based) Approach:** Relies on explicit, human-crafted rules for language structure (e.g., syntax, grammar). It is transparent and interpretable (rules are human-readable).

“Every time I fire a linguist, the performance of our speech recognition system goes up.”
--- *Fred Jelinek*



Eliza Demo

- ❑ This early natural language processing program had been written in the mid-1960s at the MIT Artificial Intelligence Laboratory by Joseph Weizenbaum.
- ❑ Early chatbots (e.g., ELIZA), it has grammar checkers, expert systems for narrow domains.

Talk to Eliza by typing your questions and answers in the input box.

```
* I am interested in AI
> Do you believe it is normal to be interested in AI?
* It is normal, tell me something about AI
> Can you elaborate on that?
* I want to be an AI Engineer, is it engouth?
> Suppose you got to be an AI Engineer, is it engouth?
* Yes
> Are you sure about that?
* sure!
> OK... "I want to be an AI Engineer, is it engouth". Tell me more.
```



TYPE HERE

<https://web.njit.edu/~ronkowit/eliza.html>

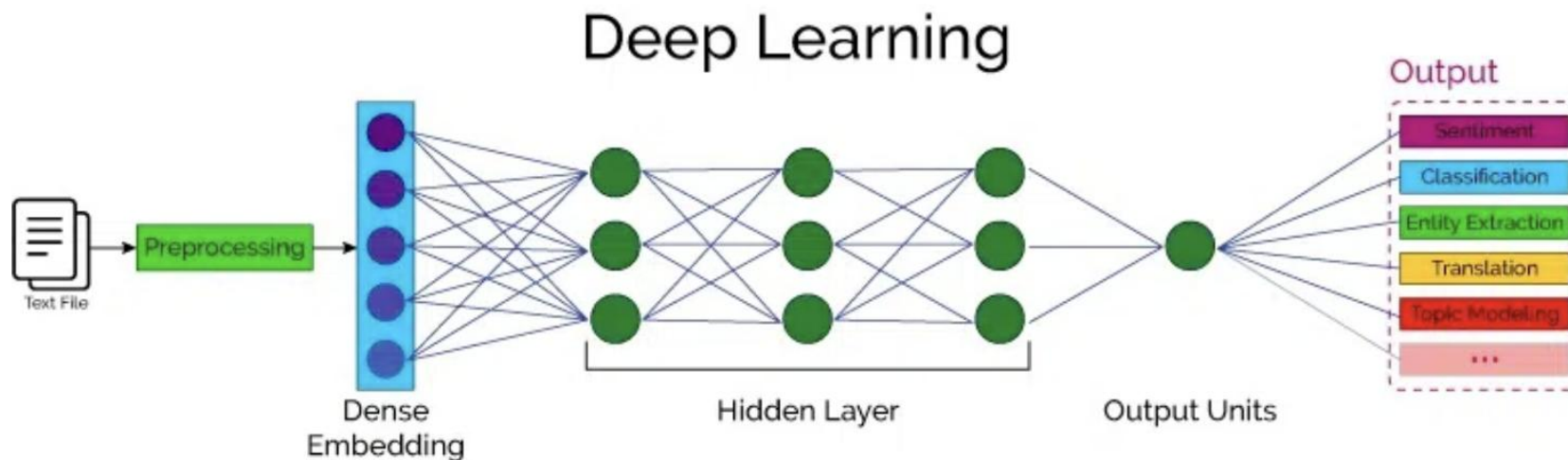
Statistical Approach of NLP

- ❑ **Statistical Approach:** Uses probabilistic models (machine learning) to learn patterns from data, without explicit rules, such as Naive Bayes, Hidden Markov Models (HMMs),
- ❑ For example, N-gram models, collocation detection. Statistical models has limited ability to handle long-range dependencies (e.g., context across paragraphs).

bigram	freq	trigram	freq
(front, desk)	2674	(front, desk, staff)	384
(great, location)	797	(non, smoking, room)	213
(friendly, staff)	775	(holiday, inn, express)	136
(hot, tub)	635	(front, desk, clerk)	122
(clean, room)	626	(flat, screen, tv)	79
(hotel, staff)	539	(smell, like, smoke)	72
(continental, breakfast)	531	(old, town, alexandria)	69
(nice, hotel)	530	(front, desk, person)	65
(free, breakfast)	522	(free, wi, fi)	62
(great, place)	514	(great, customer, service)	54

Deep Learning Approach of NLP

- ❑ Leverages neural networks to automatically learn hierarchical representations of language from raw data.
- ❑ Word Embeddings: Word2Vec, GloVe (capture semantic relationships).
- ❑ Sequence Models: RNNs, LSTMs, GRUs (handle sequential data like text).
- ❑ Transformers and Large Language Models: Self-attention mechanisms (e.g., BERT, GPT, T5) for context-aware understanding.



Comparison Table

Comparison Table

Aspect	Symbolic	Statistical	Deep Learning
Basis	Human-defined rules	Probabilistic patterns	Neural network architectures
Data Dependency	Low (rules over data)	Moderate (needs clean data)	High (massive datasets)
Flexibility	Low (rigid rules)	Moderate	High (context-aware)
Transparency	High (explicit logic)	Moderate (model-dependent)	Low (black-box)
Use Cases	Grammar checkers, expert systems	Spam detection, POS tagging	LLMs, real-time translation

Linguistic Levels

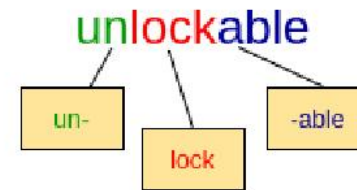
- ❑ Morphological Level: The smallest unit, focusing on the structure of words and their constituent morphemes.

Raw Text	Bag-of-words vector
it is a puppy and it is extremely cute	it 2
	they 0
	puppy 1
	and 1
	cat 0
	aardvark 0
	cute 1
	extremely 1
	...
	...


Chinese
(Mandarin)
Words normally have one or two morphemes
"young"

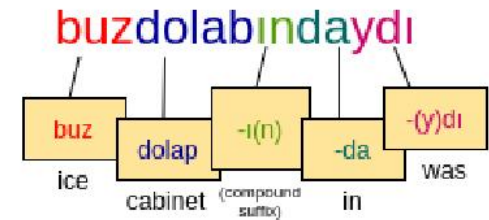



English
Words can have more morphemes than in Chinese, and it's harder to separate them out because sometimes they're not predictable and spelling may change.




Turkish
Words can have many morphemes, averaging about three per word.

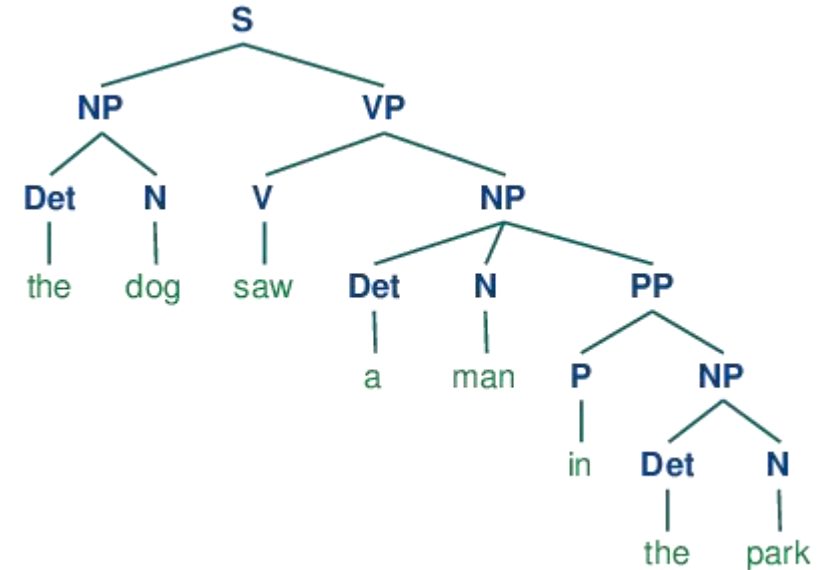
"it was in the fridge"



- ❑ Lexical Level: Concerns the vocabulary and the choice of words. For example, the model of "Bag-of-Words".

Linguistic Levels

- ❑ Syntactic Level: Deals with the structure and rules governing the arrangement of words and phrases to form sentences.



- ❑ Semantic Level: Focuses on the meaning of words, phrases, and sentences in context.
- ❑ Discourse Level: The highest level, dealing with the organization of larger linguistic units such as conversations or texts.

Tokenization

- ❑ **Word Tokenization:** This involves splitting text into individual words, treating each word as a separate token. For example:

Text: "I love NLP!"

Tokens: ["I", "love", "NLP", "!"]

- ❑ **Subword Tokenization:** This breaks words into smaller parts called subwords or morphemes. Subword tokenization is useful when dealing with out-of-vocabulary (OOV) words or rare words, as it can break them down into smaller meaningful units.

Text: "unhappiness"

Subword Tokens: ["un", "happiness"]

- ❑ **Character Tokenization:** In this approach, every single character is treated as a token, which can be useful for certain types of languages, or when dealing with noisy or unstructured data.

Text: "hello"

Tokens: ["h", "e", "l", "l", "o"]

NLP Applications

- ❑ Machine Translation: Seq2seq models, Transformer-based systems (e.g., Google Translate), low-resource language challenges.
- ❑ Sentiment Analysis & Opinion Mining: Aspect-based sentiment analysis, emotion detection.
- ❑ Text Classification: Topic labeling, spam detection, fake news identification.
- ❑ Information Extraction: Relation extraction, event detection, knowledge graph construction.
- ❑ Question Answering (QA): Open-domain QA (e.g., BERT-based systems), factoid vs. reasoning-based QA.
- ❑ Text Summarization: Extractive vs. abstractive methods, multi-document summarization.
- ❑ Speech-to-Text & Text-to-Speech: Automatic speech recognition (ASR), voice assistants (e.g., Siri, Alexa).

Advanced Applications

- ❑ Dialogue Systems: Task-oriented chatbots, open-domain conversational agents.
- ❑ Large Language Models (LLMs): GPT, BERT, and their applications (e.g., code generation, creative writing).
- ❑ Multimodal NLP: Integrating text with images/video (e.g., image captioning, visual QA).
- ❑ Low-Resource & Multilingual NLP: Cross-lingual transfer learning, zero-shot translation.
- ❑ AI Ethics & Fairness: Bias detection/mitigation, fairness in NLP models, toxicity detection.
- ❑ Prompt Engineering: Techniques for optimizing LLM outputs, few-shot/zero-shot learning.

1.2 Evolution of Language

Words on Brink

- ❑ 177 Old-English irregular verbs. Of these irregular verbs, 145 remained irregular in Middle English and 98 are still irregular today.
- ❑ We study how the rate of regularization depends on the frequency of word usage. A verb that is 100 times less frequent regularizes 10 times as fast.
- ❑ Our study provides a quantitative analysis of the regularization process by which ancestral forms gradually yield to an emerging linguistic rule.



New Evolution

How did cooperation evolve?



Mathematician and biologist Martin Nowak studies how the competition of natural selection can lead to cooperative behavior — and believes “nice guys finish first.”

[Learn more »](#)

[About Martin Nowak »](#)

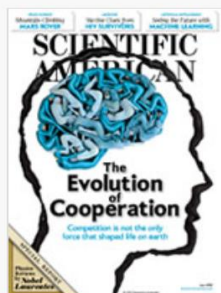
[Martin's Articles »](#)

[Program for Evolutionary Dynamics »](#)

[Harvard University Page »](#)

[Curriculum Vitae \(PDF\) »](#)

FEATURED ARTICLES



❑ How does natural selection **favor cooperation**?

❑ Mathematical models show that natural selection, on its own, opposes cooperation. Non-cooperators will do better than cooperators and wipe them out. So natural selection needs help to favor cooperation—mechanisms that make sure we get more from cooperating than being selfish. Spatial selection means that clusters of cooperators can prevail: Neighbors help each other. Group selection occurs if there is competition between groups: The members of a group help each other.

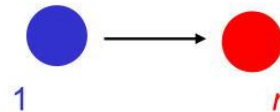
Evolution of Cooperation

□ Five mechanisms for the evolution of cooperation. **Kin selection** operates when the donor and the recipient of an altruistic act are genetic relatives. **Direct reciprocity** requires repeated encounters between the same two individuals. **Indirect reciprocity** is based on reputation; a helpful individual is more likely to receive help. **Network reciprocity** means that clusters of cooperators outcompete defectors. **Group selection** is the idea that competition is not only between individuals but also between groups..

Five Rules for the Evolution of Cooperation

Martin A. Nowak

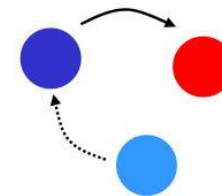
Kin selection



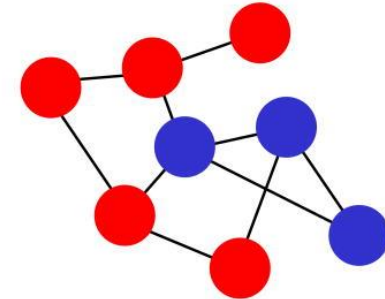
Direct reciprocity



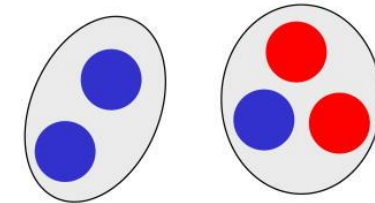
Indirect reciprocity



Network reciprocity

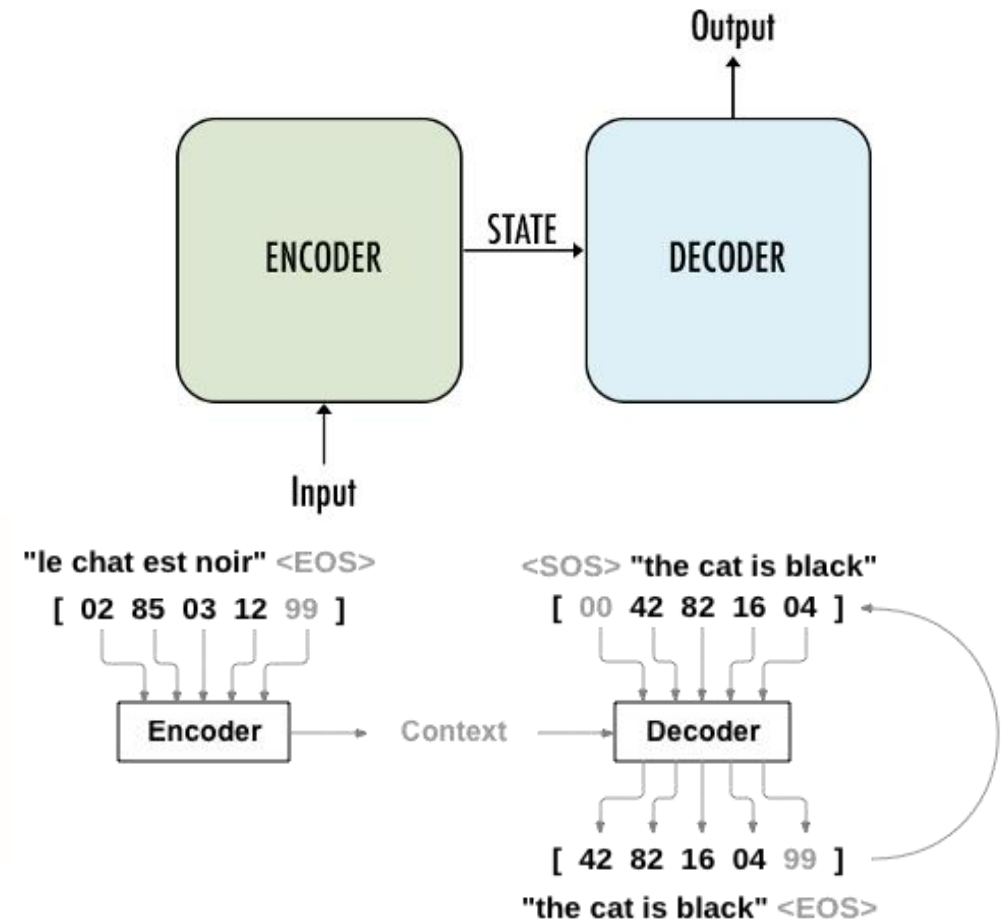
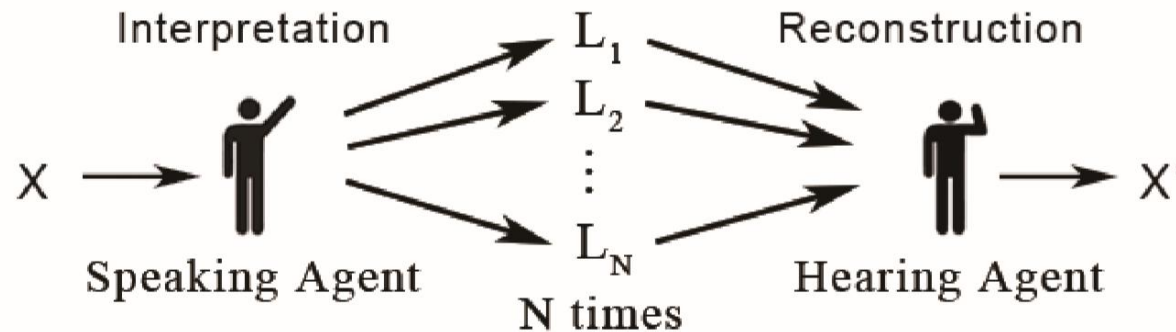


Group selection



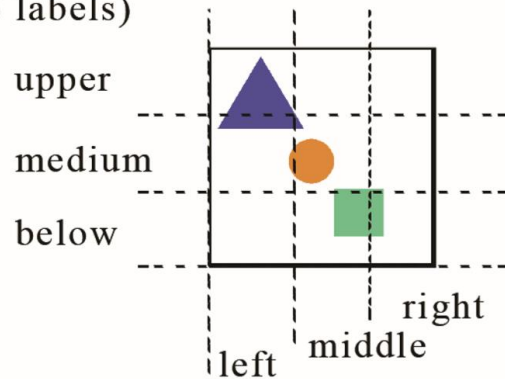
Encoder-Decoder Communications

This diagram represents a communication process between two agents—a Speaking Agent and a Hearing Agent—which resembles a model of iterative machine translation or multi-step linguistic representation and reconstruction.



Semantic Evolution

Position (6 labels)



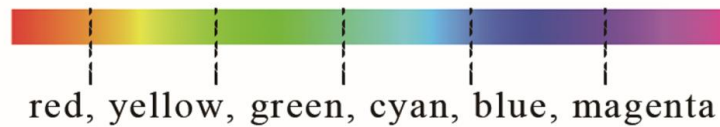
Shape (3 labels)



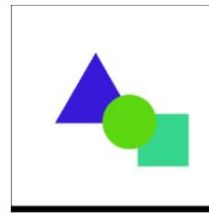
Size (3 labels)



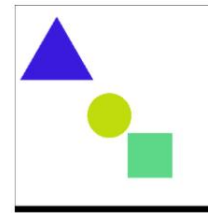
Hue Color (6 labels)



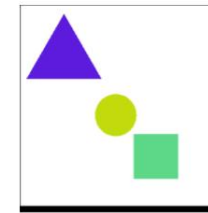
0 times



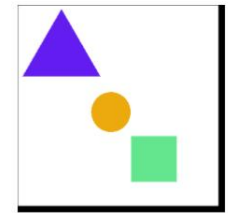
1 times



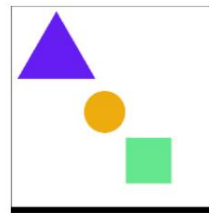
2 times



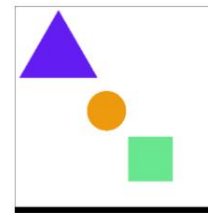
5 times



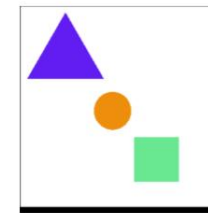
10 times



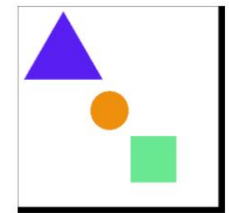
20 times



50 times



100 times



Morse Code for Communication

A ● —
B — ● ● ●
C — ● — ●
D — ● ●
E ●
F ● ● — ●
G — — ●
H ● ● ● ●
I ● ●
J ● — — —
K — ● —
L ● — ● ●
M — —
N — ●
O — — —
P ● — — ●
Q — — ● —
R ● — ●
S ● ● ●
T —

U ● ● —
V ● ● ● —
W ● — —
X — ● ● —
Y — ● — —
Z — — ● ●

1 ● — — — —
2 ● ● — — —
3 ● ● ● — —
4 ● ● ● ● —
5 ● ● ● ● ●
6 — ● ● ● ●
7 — — ● ● ●
8 — — — ● ●
9 — — — — ●
0 — — — — —

- ❑ International Morse code uses a system of dots (.) and dashes (-) to represent letters, numbers, and punctuation. The length of each code is roughly inversely proportional to the frequency of the character in the English language.
- ❑ For example, the most common letter in English, "E," is represented by a single dot (.), while less common letters like "Q" are represented by longer sequences (--.-).

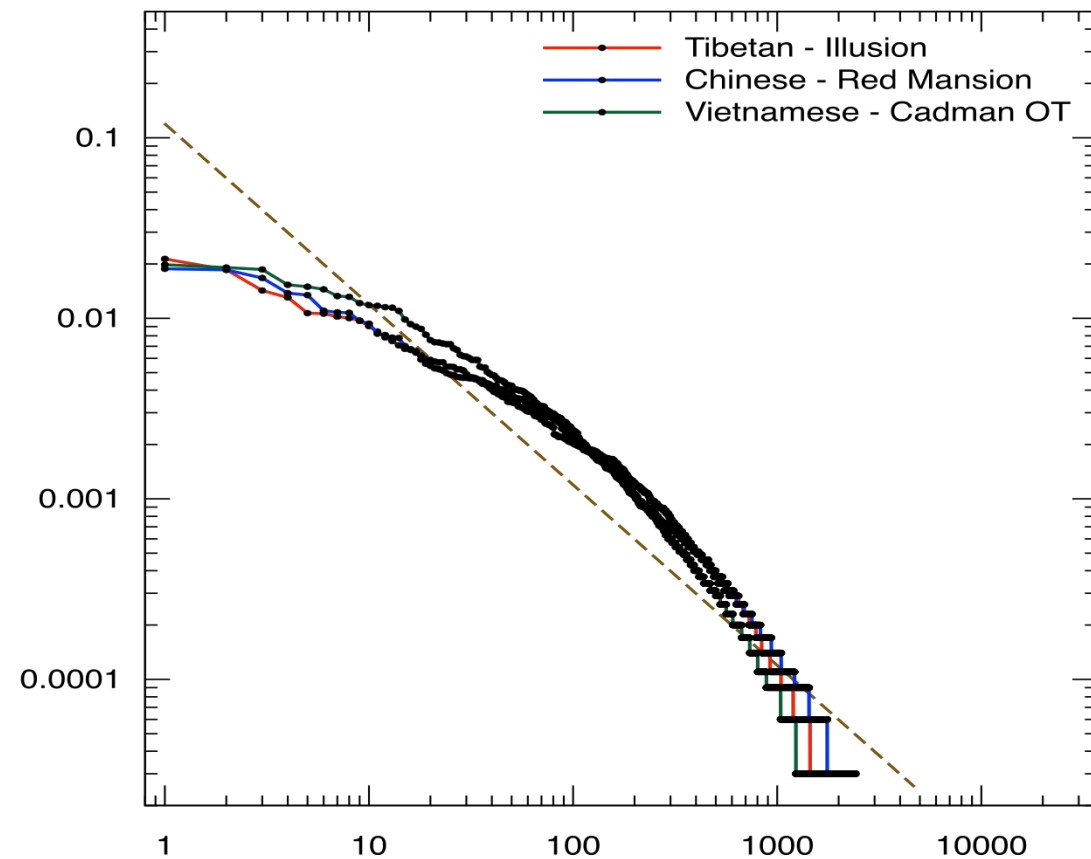
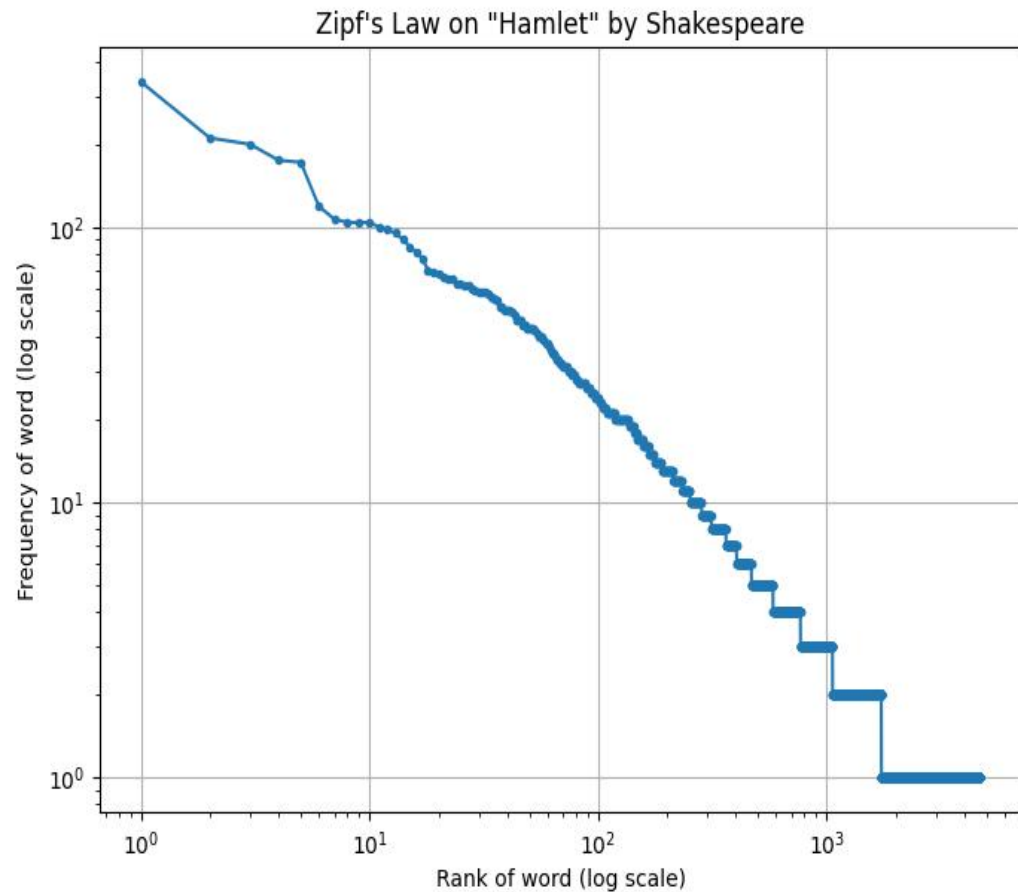
Zipf's Law

- ❑ **Zipf's Law** is an empirical law that describes the distribution of frequencies of elements in many types of datasets, particularly in natural languages. The most frequent word occurs about twice as often as the second most frequent word, three times as often as the third most frequent word, and so on.

$$f_k = \frac{C}{k^s} \quad \text{or in log form:} \quad \log(f_k) \approx -s \log(k) + \log(C)$$

- ❑ f is the frequency of the word ranked
- ❑ C is a constant (related to the size of the dataset),
- ❑ k is the rank of the word
- ❑ s is the exponent that often approximates 1 in natural language datasets.

Zipf's Law in Practice



<https://github.com/zcqn/DL-nlp2025>

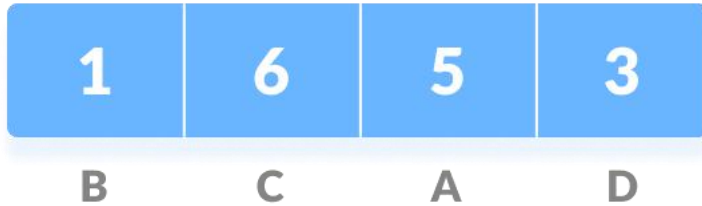
Huffman Coding

- ❑ Huffman coding is a widely used algorithm for lossless data compression. The basic idea behind Huffman coding is to use a variable-length code to represent characters, where the most frequently occurring characters are assigned shorter codes and the less frequently occurring characters are assigned longer codes.

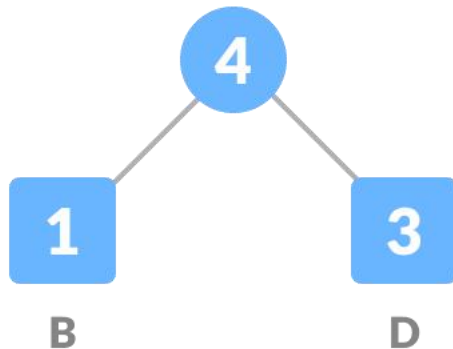
B C A A D D D C C A C A C A C

For example, if each character occupies 8 bits. There are a total of 15 characters in the above string. Thus, a total of $8 * 15 = 120$ bits are required to send this string.

Frequency Based Coding

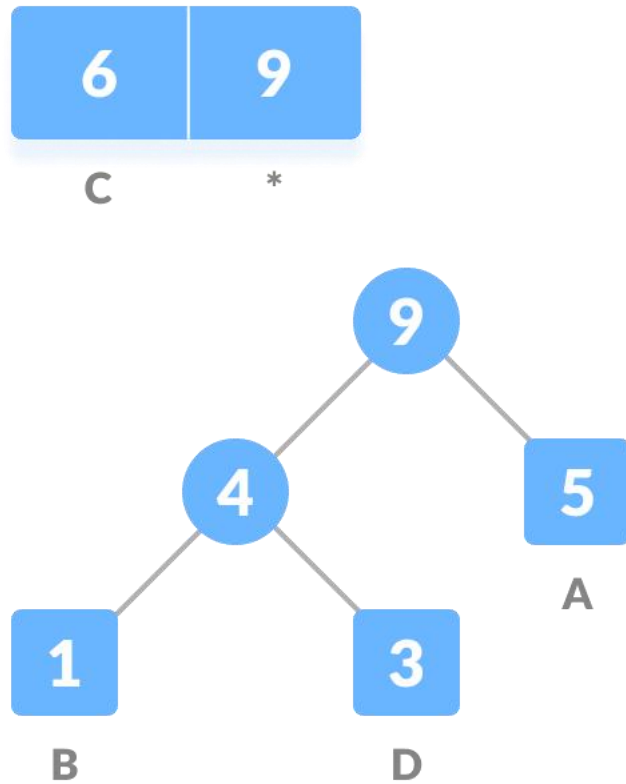


- ❑ Sort the characters in increasing order of the frequency. These are stored in a priority queue

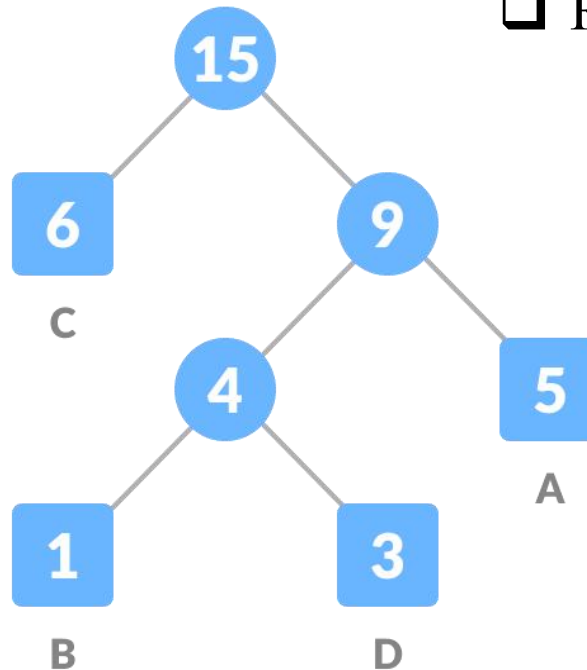


- ❑ Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above two minimum frequencies.

Huffman Tree

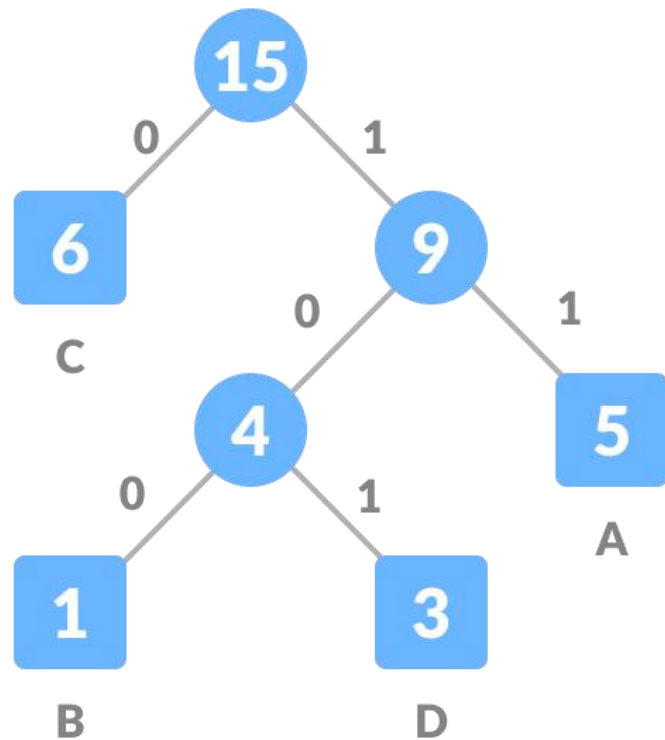


15
*



- ☐ Remove these two minimum frequencies from Q and add the sum into the list of frequencies
- ☐ Recursively add all nodes to the tree!

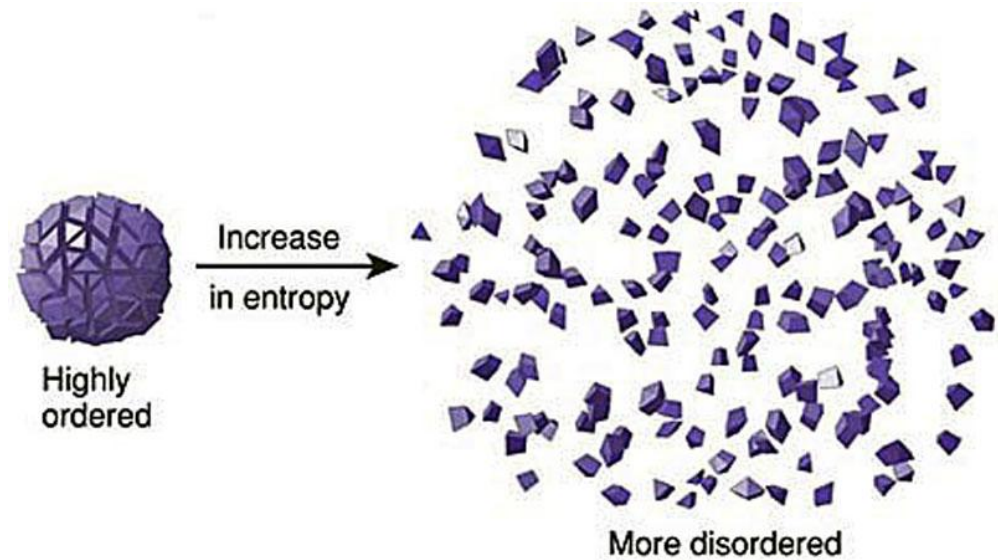
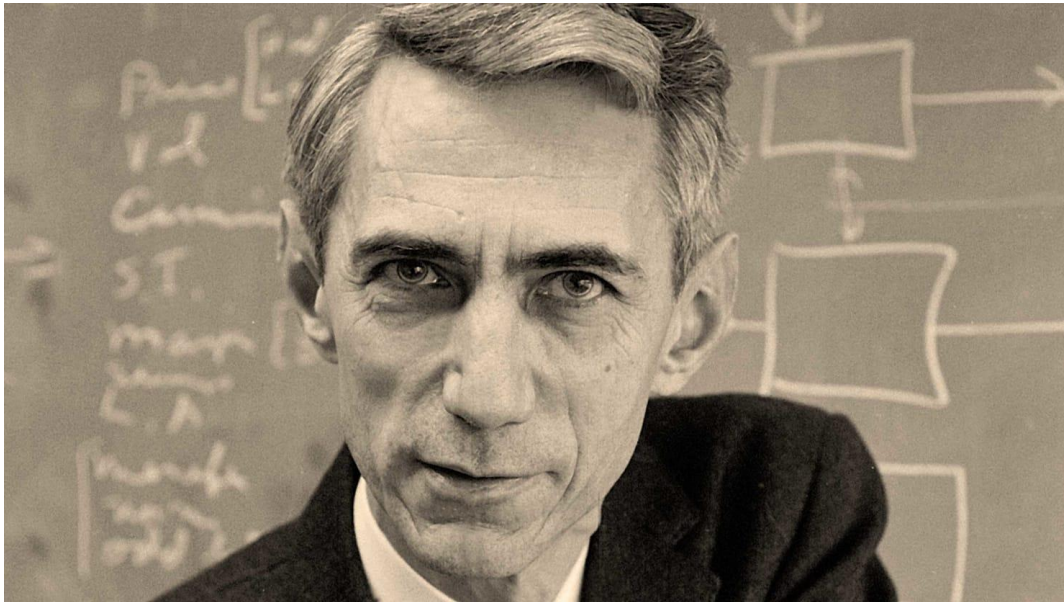
Coding of the Huffman Tree



Character	Frequency	Code	Size
<i>A</i>	5	11	$5 \times 2 = 10$
<i>B</i>	1	100	$1 \times 3 = 3$
<i>C</i>	6	0	$6 \times 1 = 6$
<i>D</i>	3	101	$3 \times 3 = 9$

Total size = 28 bits.

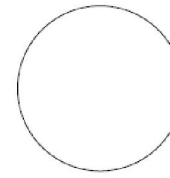
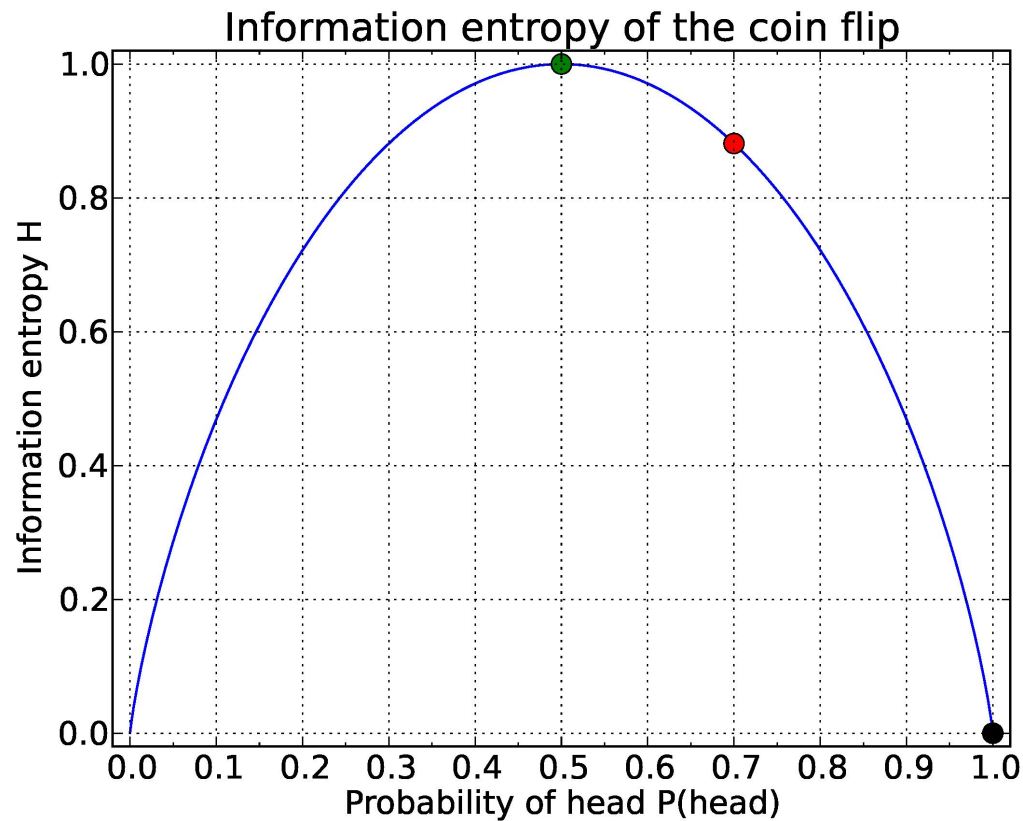
Information Theory



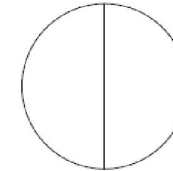
$$H(X) = - \sum_{i=1}^n P(x_i) \log_2 P(x_i)$$

- ❑ How can we come up with this equation
- ❑ What is the idea behind it?
- ❑ How probability is related to the amount of Information?

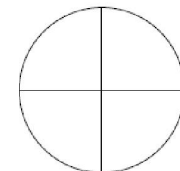
Information Entropy



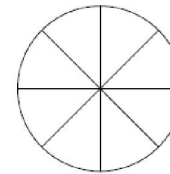
$$H(X) = 0$$



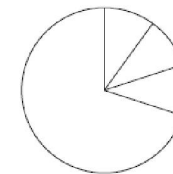
$$H(X) = 1$$



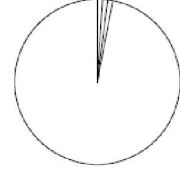
$$H(X) = 2$$



$$H(X) = 3$$



$$H(X) = 1.35678$$



$$H(X) = 0.24194$$

$$P(a) = 0.97$$

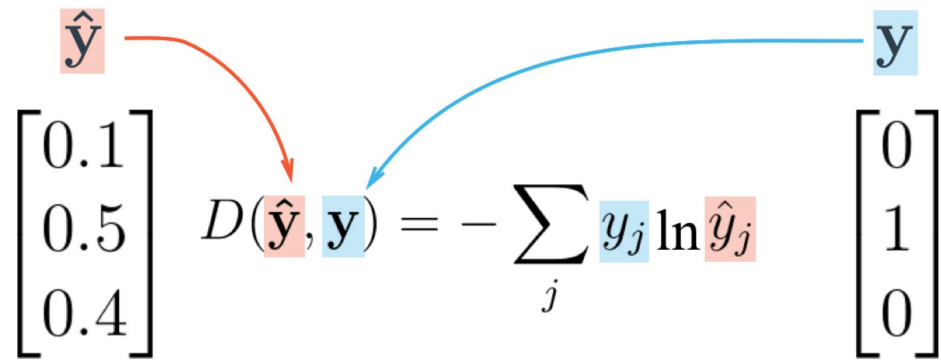
$$P(b) = 0.01$$

$$P(c) = 0.01$$

$$P(d) = 0.01$$

$$\begin{aligned} H(X) &= -0.97 \log_2 0.97 - 0.01 \log_2 0.01 \\ &\quad - 0.01 \log_2 0.01 - 0.01 \log_2 0.01 \\ &= -0.97 \log_2 0.97 - 0.03 \log_2 0.01 \\ &= -(0.97)(-0.04394) - (0.03)(-6.6439) \\ &= 0.04262 + 0.19932 \\ &= 0.24194 \end{aligned}$$

Cross Entropy


$$\hat{\mathbf{y}} = \begin{bmatrix} 0.1 \\ 0.5 \\ 0.4 \end{bmatrix} \quad D(\hat{\mathbf{y}}, \mathbf{y}) = - \sum_j y_j \ln \hat{y}_j \quad \mathbf{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

□ Cross-entropy is a concept in information theory and machine learning that measures the difference between two probability distributions. It is widely used as a **loss function in machine learning**, particularly in classification tasks.

□ Think of cross-entropy as a way to measure how surprised a model is when it predicts something. If the model's prediction **is close to reality, the cross-entropy is low**. If the **prediction is far off, the cross-entropy is high**.

□ $H([1, 0, 0], [0.7, 0.2, 0.1]) = -(1 \cdot \log 0.7 + 0 \cdot \log 0.2 + 0 \cdot \log 0.1) = -\log 0.7 \approx 0.357$

□ $H([1, 0, 0], [0.9, 0.1, 0]) = -(1 \cdot \log 0.9 + 0 \cdot \log 0.1 + 0 \cdot \log 0) = -\log 0.9 \approx 0.105$

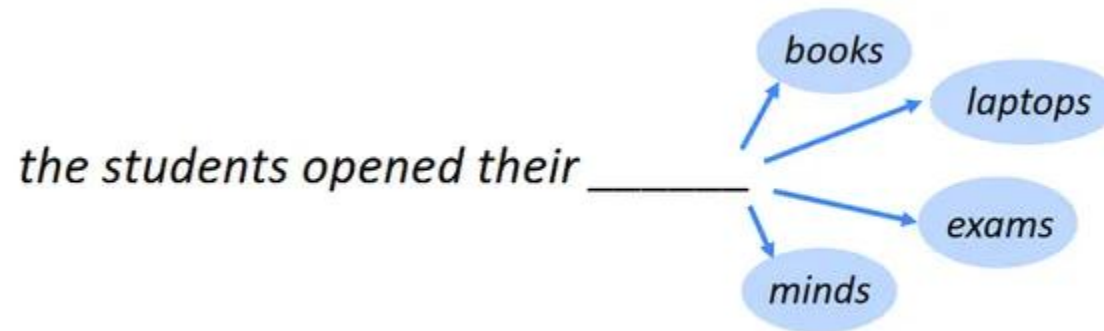
1.3 Statistical Language Model

Next Word Prediction

A language model is a probabilistic model that captures the rules and patterns of a language. It answers questions like:

- ❑ What is the probability of a given sentence or sequence of words?
- ❑ Given a sequence of words, what is the most likely next word?

For example, consider the sentence:



Types of Language Models

❑ Statistical Language Models

These models use statistical methods to estimate the probability of word sequences.

Example: N-gram models (e.g., bigram, trigram).

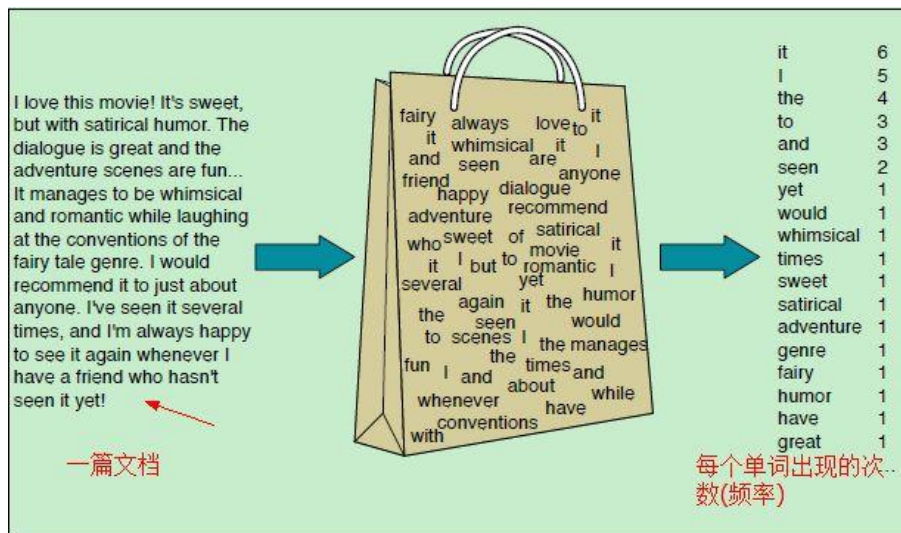
❑ Neural Language Models

These models use neural networks (e.g., RNNs, LSTMs, Transformers) to learn complex patterns in language.

Example: GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers).

Bag-of-Words

The Bag-of-Words (BoW) model is to represent a text (like a sentence or a document) as an unordered set of words, disregarding grammar and word order but keeping track of the frequency of each word.



Document D1	<i>The child makes the dog happy</i> the: 2, dog: 1, makes: 1, child: 1, happy: 1
Document D2	<i>The dog makes the child happy</i> the: 2, child: 1, makes: 1, dog: 1, happy: 1



	child	dog	happy	makes	the	BoW Vector representations
D1	1	1	1	1	2	[1,1,1,1,2]
D2	1	1	1	1	2	[1,1,1,1,2]

Simplest LM

$$p(x_1 \dots x_n) = \frac{c(x_1 \dots x_n)}{N}$$

This is, however, a very poor model: in particular it will assign probability 0 to any sentence not seen in the training corpus. Thus it fails to generalize to sentences that have not been seen in the training data.

A **Bi-gram model** is an extension of the Bag-of-Words (BoW) model that considers pairs of consecutive words.

I love AI in Bi-gram: [('I', 'love'), ('love', 'AI')]

N-Gram Model

Let's say we have a small **corpus** of 5 sentences:

"I love AI."
"AI is amazing."
"I enjoy learning."
"Love is learning."
"Robot, the future."

"I" = 2, "love" = 2, "AI" = 2, "is" = 2,
"amazing" = 1, "learning" = 2, "the" = 1,
"future" = 1, "enjoy" = 1, "robot" = 1

The corpus has 15 words with above word frequency.

$$P(w) = \frac{\text{Frequency of } w}{\text{Total number of words in the corpus}}$$
 For **N=1**, it is just the bag-of-words model and the probability of **"I love AI" = P(I)P(love)P(AI)**

$$P(\text{"I love AI"}) = P(I) \times P(\text{love}) \times P(\text{AI}) = \frac{2}{15} \times \frac{2}{15} \times \frac{2}{15} = \frac{8}{3375}$$

N-Gram Model (N=2)

For the bigram model, the probability of a sentence is the product of the probabilities of each word given the previous word:

$$P(w_1, w_2, \dots, w_n) = \prod_{i=2}^n P(w_i | w_{i-1}) \quad P(w_i | w_{i-1}) = \frac{\text{Frequency of bigram } (w_{i-1}, w_i)}{\text{Frequency of word } w_{i-1}}$$

$$P(\text{"I love AI"}) = P(\text{love} | \text{I}) \times P(\text{AI} | \text{love}) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$

Because: $P(\text{love} | \text{I}) = \frac{\text{Frequency of ("I", "love")}}{\text{Frequency of "I"}} = \frac{1}{2}$

$$P(\text{AI} | \text{love}) = \frac{\text{Frequency of ("love", "AI")}}{\text{Frequency of "love"}} = \frac{1}{2}$$

N-Gram Model (N=3)

For the trigram model, the probability of the sentence is the product of the probabilities of each word given the previous two words:

$$P(w_1, w_2, w_3) = P(w_1) \times \prod_{i=2}^3 P(w_i | w_{i-1}, w_{i-2})$$

But notice here that "I love AI" is form a trigram (there are no sequences of 3 consecutive words in the sentence) given in the corpus, therefore:

$$P(AI | love, I) = 1$$

You are welcomed to calculate the probabilities of any given sentences constructed from this courpus.

Thanks!