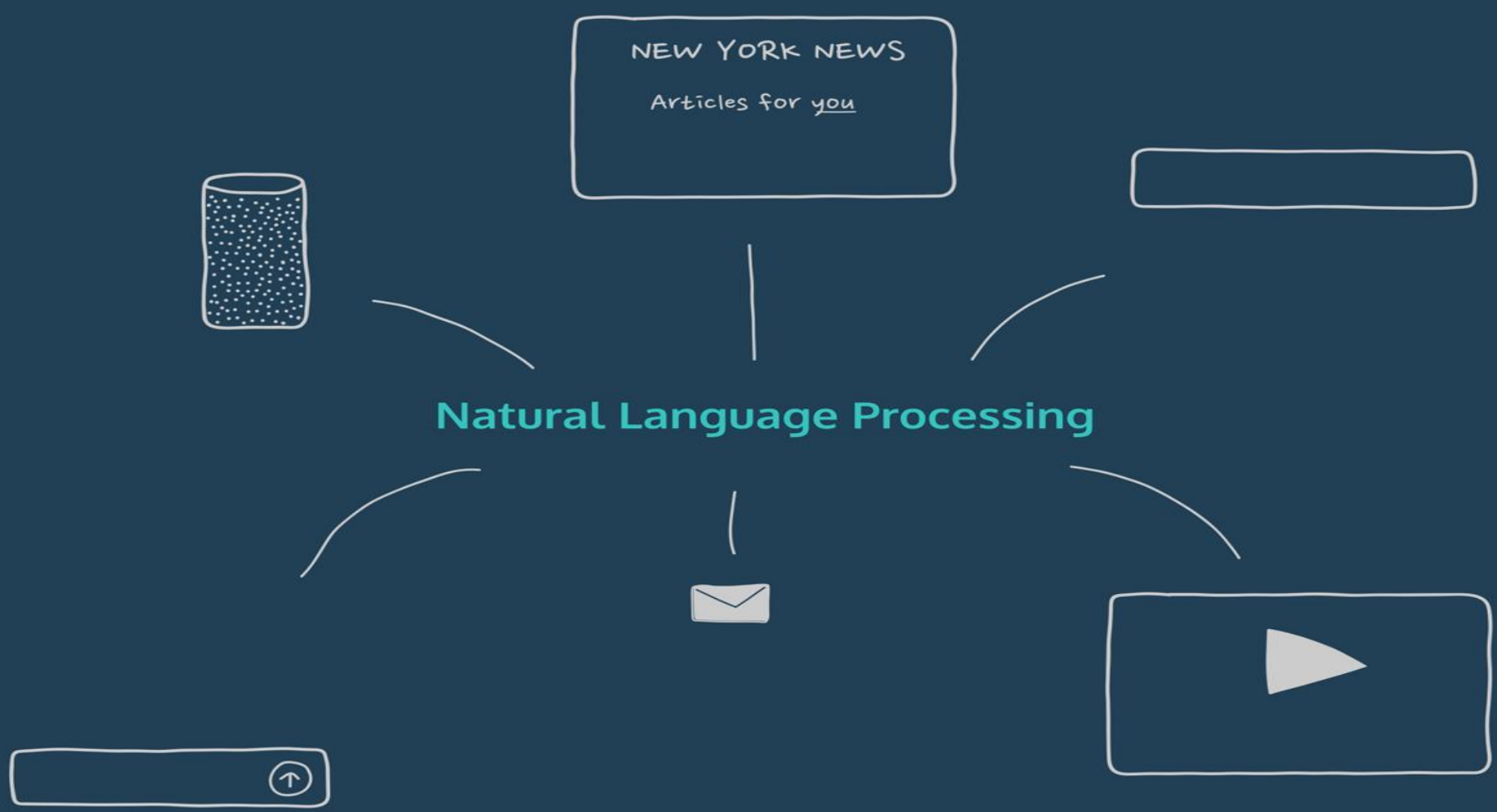
DATA, MSML, BIOI 602 Principles of Data Science

Natural Language Processing

[Slides from Fardina Fathmiul Alam]


NEW YORK NEWS
Articles for you
Natural Language Processing

What is Natural Language Processing(NLP)?

NLP is a branch of artificial intelligence that ***deals with analyzing, understanding, and generating the languages*** that humans use naturally in order to interface with computers in both written and spoken contexts using natural human languages instead of computer languages.

What is Natural Language Processing (NLP)?

A blanket term for all tasks involving human language.

If completely noise is maximally unstructured data, and tabular data is maximally structured, where do we think language falls?

What is Natural Language Processing (NLP)?

A blanket term for all tasks involving human language.

If completely noise is maximally unstructured data, and tabular data is maximally structured, where do we think language falls?

Slightly more structured than images.

- Language has some structure through grammar and syntax, while images rely on pixel arrangements for visual information.

In what way does language have structure?

There are around 170,000 words in the English language. If I pick the first word from a list of valid first words to finish a sentence, there is a much smaller subset of valid subsequent words. For the third word of the sentence, there is an even smaller subset.

- E.g. *we won't say "The the"* because that doesn't make sense.
- Reflecting the constraints imposed by grammar, syntax, and semantic coherence in constructing sentences.

In what way does language have structure?

There are around 170,000 words in the English language. If I pick the first word from a list of valid first words to finish a sentence, there is a much smaller subset of valid subsequent words. For the third word of the sentence, there is an even smaller subset.

So, when I say language has structure, I mean there are rules that govern the order in which words can appear.

Syntax vs. Semantics

- **Syntax:** The structure of what makes a sentence valid. A syntactically valid sentence is one where the subject, object and verb are all in the correct place.

"Colorless green ideas sleep furiously" is an example of a **syntactically valid** sentence.

- **Semantics:** What your sentence means. The sentence above is syntactically valid, but not semantically valid (though, if you're an English major these concepts get fuzzy real fast).

Syntax vs. Semantics

Syntax: refers to the **grammatical structure** of language

- The rules via which one forms sentences/expressions

Semantics: the study of **meaning of language.**

John is rectangular and a rainbow.

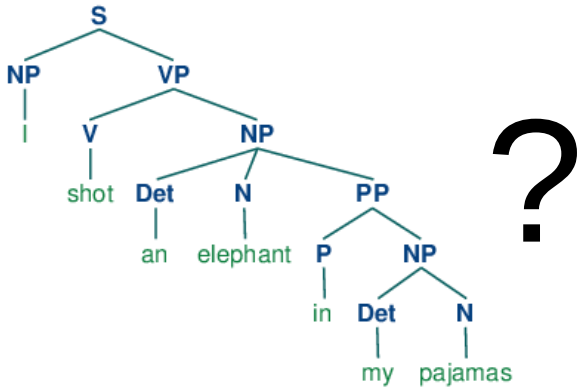
- Syntactically correct
- Semantically meaningless

UNDERSTANDING LANGUAGE IS HARD

One morning I shot an elephant in my pajamas.

How he got into my pajamas, I'll never know.

Groucho Marx

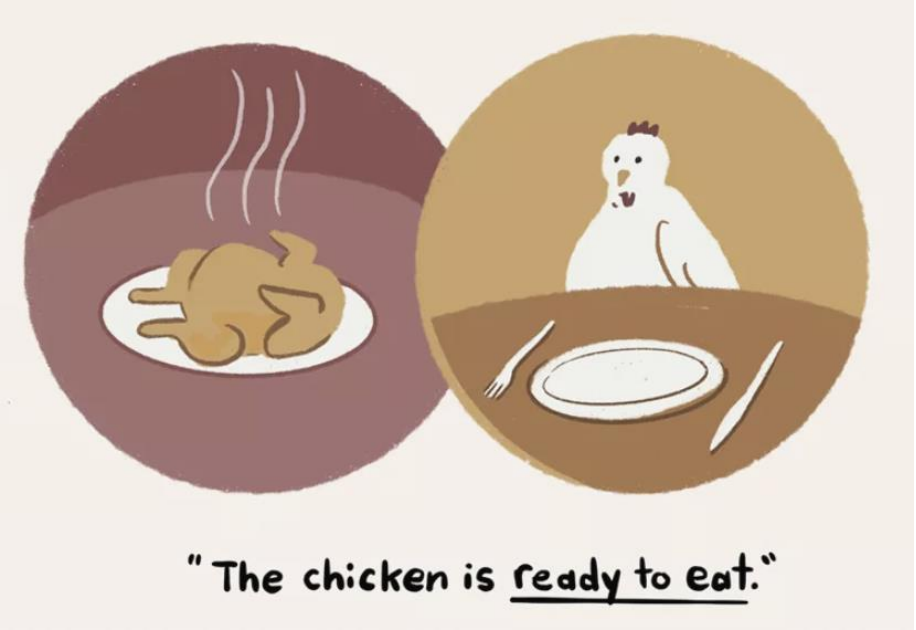


?

Potential Challenges:

Sentences in English can be ambiguous, a single sentence can have multiple different semantic meanings.

- Not everyone uses the same grammatical rules or spellings.
- Word meanings can change fairly rapidly.



NLP’s practical applications

- Summarize Documents
- Sentiment Analysis
- Topic modeling
- Auto-complete
- Question answering
- Translating

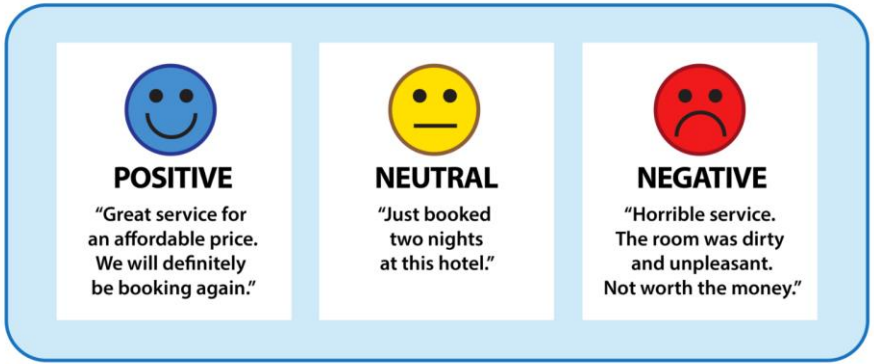
NLP's practical applications

- **Sentiment analysis:** Determining if someone feels positively or negatively towards something.
- **Topic modeling:** Identify what is this article about?
- **Question Answering:** Developing a system that answers questions in any natural language.
- **Named Entity Resolution:** Figuring out who or what a phrase is referring to.
- **Text Summarization:** What it sounds like: Condensing a text to its essential points.

Sentiment Analysis

The process of classifying the emotional intent of text.

- Generally, input → a piece of text, and output → probability that the sentiment expressed is positive, negative, or neutral.



Given text, sentiment analysis classifies its emotional quality.

SEMANTICS: SENTIMENT ANALYSIS

Deciding if reviews/opinions are positive or negative

Heavily used by ad industry today

An extremely versatile machine!, November 22, 2006

By **Dr. Nickolas E. Jorgensen "njorgens3"**

This review is from: Cuisinart DGB-600BC Grind & Brew, Brushed Chrome (Kitchen)

This coffee-maker does so much! It makes weak, watery coffee! It grinds beans if you want it to! It inexplicably floods the entire counter with half-brewed coffee when you aren't looking! Perhaps it could be used to irrigate crops... It is time-consuming to clean, but in fairness I should also point out that the stainless-steel thermal carafe is a durable item that has withstood being hurled onto the floor in rage several times. And if all these features weren't enough, it's pretty expensive too. If faced with the choice between having a car door repeatedly slamming into my genitalia and buying this coffee-maker, I'd unhesitatingly choose the Cuisinart! The coffee would be lousy, but at least I could still have children...

SECTIONS The New York Times SUBSCRIBE NOW LOG IN

POLITICS

Trump's New Travel Ban Blocks Migrants From Six Nations, Sparing Iraq

Leer en español

By GLENN THRUSH MARCH 6, 2017 561

President Trump during a meeting in the Roosevelt Room of the White House last week. Al Drago/The New York Times

WASHINGTON — President Trump signed an executive order on Monday blocking citizens of six predominantly Muslim countries from entering the United States, the most significant hardening of immigration policy in generations, even with changes intended to blunt legal and political opposition.

The order was revised to avoid the tumult and protests that engulfed the nation's airports after Mr. Trump signed his first immigration directive on Jan. 27. That order was ultimately blocked by a federal appeals court.

The new order continued to impose a 90-day ban on travelers, but it removed Iraq, a redaction requested by Defense Secretary Jim Mattis, who feared it would hamper coordination to defeat the Islamic State, according to administration officials.

It also exempts permanent residents and current visa holders, and drops language offering preferential status to persecuted religious

What type of article is this?

- Sports
- Political
- Dark comedy
- Reality TV

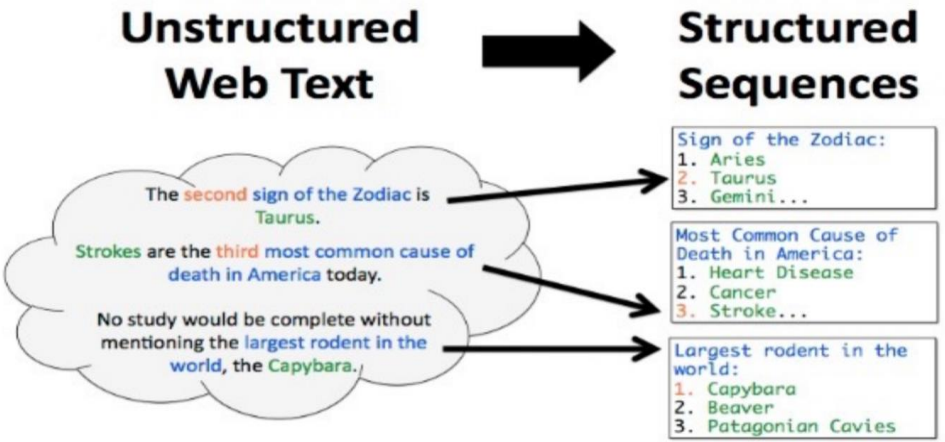
What entities are covered?

- And are they covered with positive or negative sentiment?

Semantic Information Extraction

The process of identifying and extracting meaningful information or concepts (semantic content) from text.

- This goes beyond simple keyword extraction; understands context, relationships, and intended meaning of the text.
- Enables machine understanding and reasoning.
- Essential for NLP: Question answering, info retrieval, knowledge graphs.



SEMANTICS: MACHINE TRANSLATION

Translating from one language to another

Simple substitution of words doesn't work very well

(Image: Google)

Input sentence:	*Translation (PBMT):*	*Translation (GNMT):*	*Translation (human):*
李克強此行將啟動中加總理年度對話機制，與加拿大總理杜魯多舉行兩國總理首次年度對話。	Li Keqiang premier added this line to start the annual dialogue mechanism with the Canadian Prime Minister Trudeau two prime ministers held its first annual session.	Li Keqiang will start the annual dialogue mechanism with Prime Minister Trudeau of Canada and hold the first annual dialogue between the two premiers.	Li Keqiang will initiate the annual dialogue mechanism between premiers of China and Canada during this visit, and hold the first annual dialogue with Premier Trudeau of Canada.

SEMANTICS: QUESTION ANSWERING

Answer questions posed a user with specific answers

SEMANTICS: DOCUMENT SUMMARIZATION

Quite a few tools out there today… e.g., SMMRY

autotldr commented on a post in r/SkydTech

After Supreme Court detour, Apple v. Samsung goes to a fourth jury trial (arstechnica.com)

submitted 6 hours ago by cryoskyd to r/SkydTech

autotldr • 1 point • submitted 36 minutes ago

This is the best tl;dr I could make, original reduced by 86%. (I'm a bot)

> The Apple v. Samsung lawsuit is getting a big "Reset," thanks to last year's Supreme Court ruling on design patents.
>
> The US Supreme Court said that it was wrong to give Apple damages on the entire phone because of a few design patents.
>
> Apple and Samsung made their arguments over what the test should be, but Judge Koh ended up going with the test suggested by the US solicitor general, which has four factors to determine the right "Article of manufacture." It's closer to Apple's suggestion-Samsung had suggested basically taking only the part of the product that had the patented design physically applied to it, a test that Koh said wouldn't even pass the basics of what the Supreme Court had asked for.

Extended Summary | FAQ | Feedback | *Top keywords*: **design**[#1] **patent**[#2] **Apple**[#3] **Court**[#4] **product**[#5]

SEMANTICS: NAMED ENTITY RECOGNITION

Identifying key entities in text

Named Entity Recognition (NER) is a natural language processing task that identifies and categorizes named entities within text into predefined categories such as persons, organizations, locations, dates, and more

In 1917, Einstein applied the general theory of relativity to model the large-scale structure of the universe. He was visiting the United States when Adolf Hitler came to power in 1933 and did not go back to Germany, where he had been a professor at the Berlin Academy of Sciences. He settled in the U.S., becoming an American citizen in 1940. On the eve of World War II, he endorsed a letter to President Franklin D. Roosevelt alerting him to the potential development of "extremely powerful bombs of a new type" and recommending that the U.S. begin similar research. This eventually led to what would become the Manhattan Project. Einstein supported defending the Allied forces, but largely denounced using the new discovery of nuclear fission as a weapon. Later, with the British philosopher Bertrand Russell, Einstein signed the Russell-Einstein Manifesto, which highlighted the danger of nuclear weapons. Einstein was affiliated with the Institute for Advanced Study in Princeton, New Jersey, until his death in 1955.

Tag colours:

LOCATION TIME PERSON ORGANIZATION MONEY PERCENT DATE

Evolution of Problem-Solving in NLP

Then:

- Solve problems using custom, hand-trained approaches for each task, employing a variety of clever machine learning techniques.

Now:

- Use large language models to address a wide range of problems.

This class:

- We'll cover the historical approaches because understanding them is beneficial. Then, we'll discuss how things work with modern techniques.

Ancient History

Introduction

Topic Modeling is a machine learning technique used to identify topics or themes present in a collection of text documents.

Vocab:

- **Word (or Term):** A word (A single unit of language).
- **Document:** A collection of words.
- **Corpus:** A collection of documents.
- **Topic Modeling:** Given a corpus of documents **with topic labels,** learn a mapping of documents → to assign topic labels for new documents.

TERMINOLOGY

Documents: groups of free text

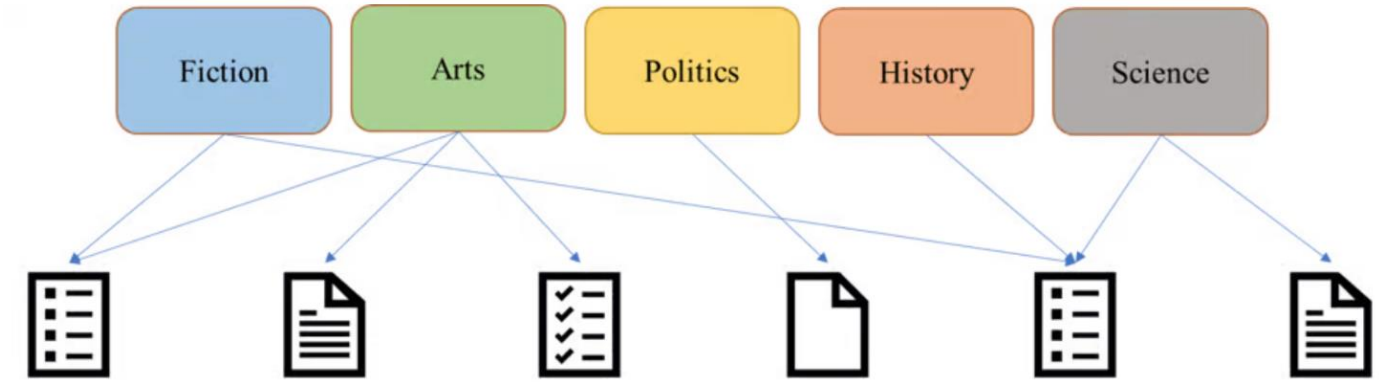
- Actual documents (NYT article, journal paper)
- Entries in a table

Corpus: a collection of documents

Terms: individual words separated by whitespace or punctuation

Topic Modeling is Just Supervised Learning!

Assign a general topic to a set of documents (corpus) that best describes and fits the contents of those documents (based on the words present).



Topic Modeling is Just Supervised Learning!

But wait...how do we get a document (in this case, let's say a book), to become a vector?

What happens if we make every word a number?

Topic Modeling is Just Supervised Learning!

But wait...how do we get a document (in this case, let's say a book), to become a vector?

What happens if we make every word a number, based on it's **position** in the **dictionary**?

"I like aardvarks"-> <1133, 3940, 1>

Topic Modeling is Just Supervised Learning!

But wait...how do we get a document (in this case, let's say a book), to become a vector?

What happens if we make every word a number, based on it's **position** in the **dictionary**?

"I like aardvarks"-> <1133, 3940, 1>

Hmm...words are more categorical than numerical or ordinal. Do we have a system for turning categorical variables into numerical?

Topic Modeling is Just Supervised Learning!

But wait...how do we get a document (in this case, let's say a book), to become a vector?

What happens if we make every word a number, based on it's **position** in the **dictionary**?

"I like aardvarks"-> <1133, 3940, 1>

Hmm...words are more categorical than numerical or ordinal (represent discrete concepts in language without inherent order or numeric value).

Do we have a system for turning categorical variables into numerical?

ONE HOT ENCODING

Turning texts into vectors

We need to convert text data into numerical vectors for machine learning tasks.

- Machine learning algorithms require numerical inputs for processing.
- Numerical vectors represent text features like word frequencies or semantic meanings, enabling algorithms to understand and analyze text.
- Many more.

Turning texts into vectors

Bag of words

N-Grams

TF-IDF

Word Embedding

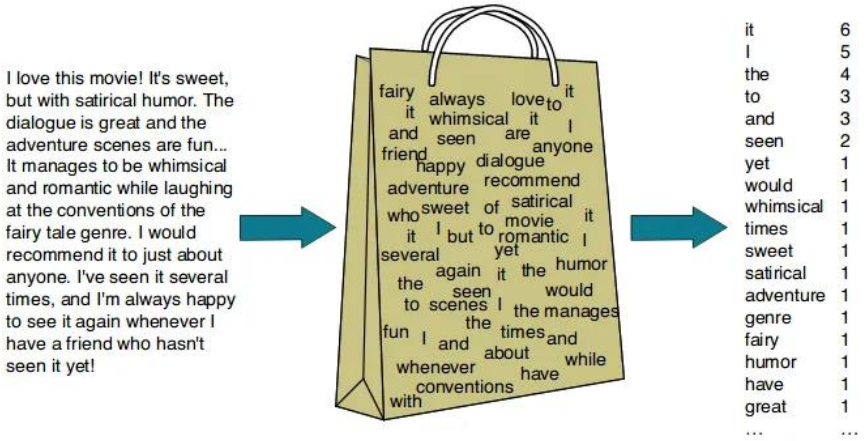
> *In language processing, the vectors x are derived from textual data, in order to reflect various linguistic properties of the text.*
>
> — Page 65, Neural Network Methods in Natural Language Processing, 2017.

Bag of Words (BoW) or Unigram Approach: Turning texts into vectors

A representation of text that describes the **occurrence of words** within a document. It involves two things:

- A vocabulary of known words.
- A measure of the presence of known words.

BoW counts the frequency of words in a document disregarding grammar and word order.



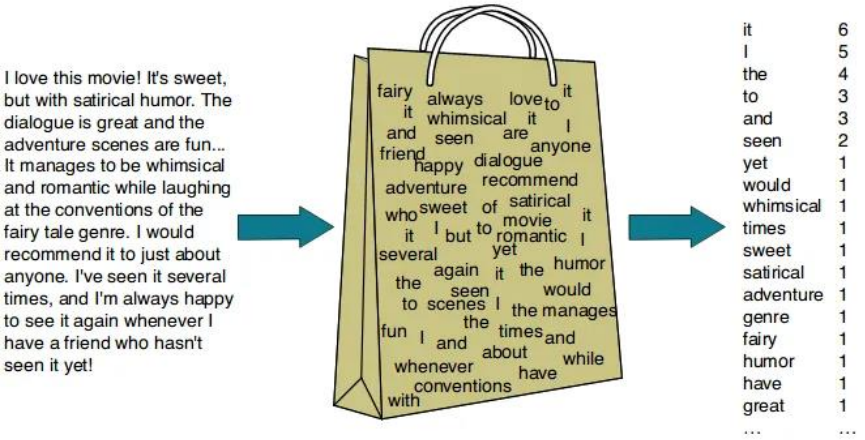
BoW represent each document as a vector of word frequencies

- Order of words does not matter, just #occurrences

Bag of Words (BoW) or Unigram Approach: Turning texts into vectors

Why called a “bag” of words?

- Any information about the **order** or structure of words in the document is **discarded**.
- Only concerned with whether known words occur in the document, not where in the document.



Note: The emphasis is on understanding document similarity through the analysis of words present in the documents, without considering their order or structure. This aligns with the principles of bag-of-words or unigram approaches in natural language processing, where documents are treated as collections of words, and their meaning is inferred from the content alone.

Turning texts into vectors

Consider, our textual data:

"I like school and I like homework" → Document 1

"I have homework to do" → Document 2

"School has homework" → Document 3

"Homework has school" → Document 4

} Entire Corpus of Documents

We can use Bag of Words (BoW) or Unigram Approach

Turning texts into vectors

"I like school and I like homework"

"I have homework to do"

"School has homework"

"Homework has school"

Design the Vocabulary: make a list of all of the words in our model vocabulary.

The unique words here (ignoring case and punctuation)

I	**Like**	**School**	**Have**	**Homework**	**To**	**Do**	**Has**	**And**

BoW

Document D1	*The child makes the dog happy* the: 2, dog: 1, makes: 1, child: 1, happy: 1
Document D2	*The dog makes the child happy* 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]

Turning texts into vectors

"I like school and I like homework"

"I have homework to do"

"School has homework"

"Homework has school"

We can use Bag of Words (BoW) or Unigram Approach

a fixed-length document representation of 9, with in each vector, **the numbers represent the frequency** of the corresponding word in the given sentence.

Create Document Vectors: score the words in each document.

I	Like	School	Have	Homework	To	Do	Has	And
2	2	1		1				1
1			1	1	1	1		
		1		1			1	
		1		1			1	

*Empty are 0

Pros and Cons: Bag of Words (BoW) or Unigram Approach:

Pros	Cons
Simplicity: Easy to implement and understand. **Computational Efficiency:** Unigram representations result in sparse vectors that are computationally efficient to process and analyze, especially for large datasets. **Language Independence:** Applicable to any language. **Capture Word Importance:** Highlights individual important words (Words with higher frequencies).	**Loss of Word Order:** Ignores word sequence. **Limited Context:** Doesn't capture word relationships. **Vocabulary Size:** Can be large, leading to high-dimensional sparse vectors. This can pose challenges in terms of storage and computation. *As such, there is pressure to decrease the size of the vocabulary when using a bag-of-words model.*

Limitation of BoW :

"I like school and I like homework"

"I have homework to do"

"School has homework"

"Homework has school"

"School has homework" and "Homework has school" have the same unigram representation, despite having different meanings (Limited Context Understanding).

I	Like	School	Have	Homework	To	Do	Has	And
2	2	1		1				1
1			1	1	1	1		
		1		1			1	
		1		1			1	

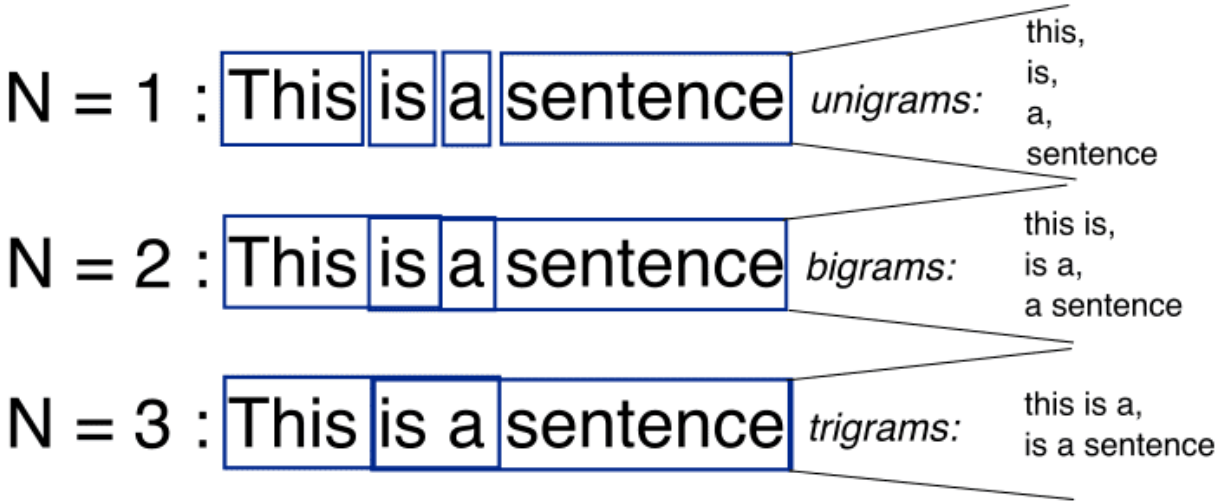
N-Grams

A more sophisticated approach is to create a vocabulary of grouped words.

If we want more meaning, we can one-hot encode every *pair* of words.

- Each word or token is called a “gram”.

- N-Grams are contiguous sequences of N items from a given text, typically words or characters.
- They capture the context and sequential information present in the text.



Bi-Grams

Creating a vocabulary of two-word pairs is, in turn, called a bigram model.

Examples:
"School has homework"

"Homework has school"

School has	Has homework	Homework has	Has school
1	1		
		1	1

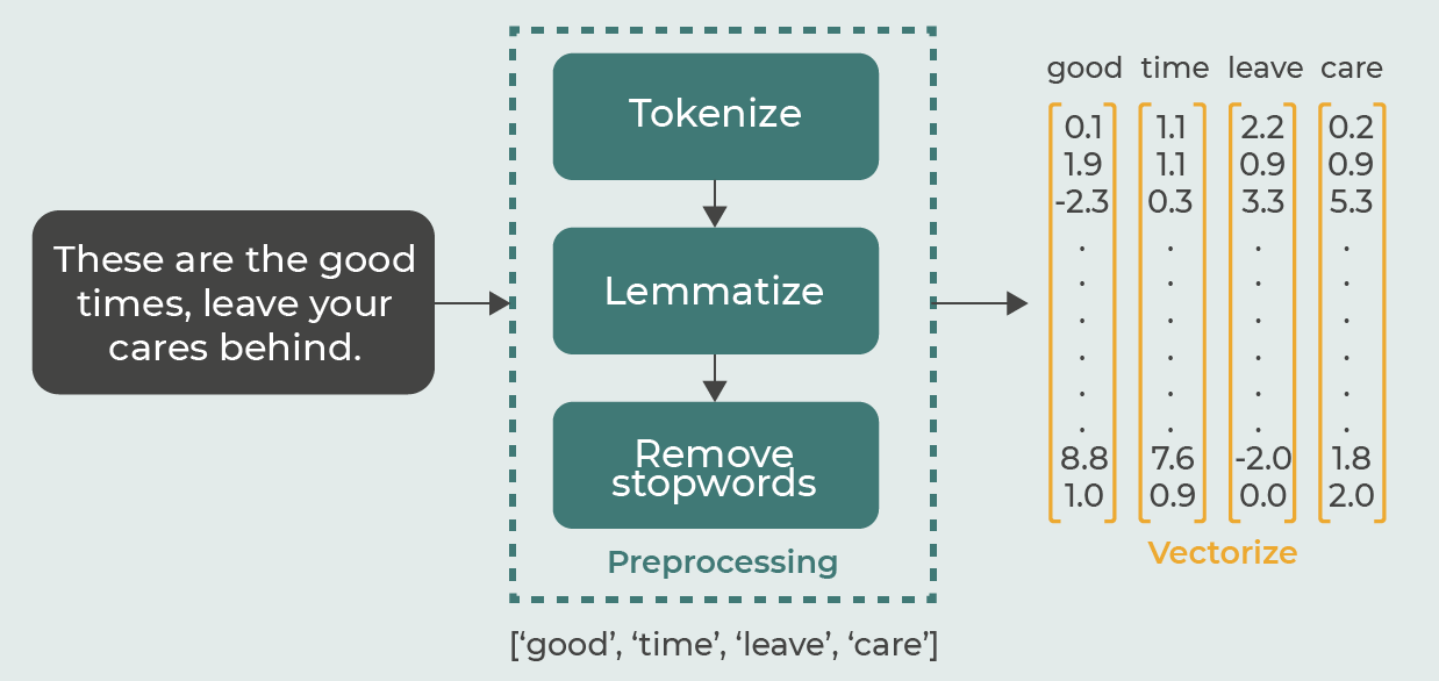
These are called 'bigrams'

Note: only the bigrams that appear in the corpus are modeled, not all possible bigrams.

Before we do Vectorization, we need to do Data Cleaning and Tokenization.

Steps of NLP

Raw Text Documents → **Data Cleaning by Tokenization →** **Vectorization (BoW/N-Grams/TF-IDF etc.)**



What about these sentences

- "In the morning I'm running"
- "In the morning I am running"
- "In the morning I'm runnin"
- "In the morning I run"
- "In the morning I am running"
- "In the monring I am running"
- "In the morning I a m running"

TOKENIZATION

First step towards text processing;

This is the process of breaking down text into smaller units on chunks, such as words, phrases, or symbols.

- **Example: the sentence "In the morning I'm running" would be tokenized into ["In", "the", "morning", "I'm", "running"].**

TOKENIZATION

Tokenization is not straightforward!

- **For English, just split on non-alphanumeric characters**
  - Need to deal with cases like:
    - Contractions: e.g., "I’m" (short for "I am").
    - Possessives: e.g., "France’s" (to show something belongs to France).
    - Hyphenated Words: e.g., "Hewlett-Packard" (where two words are joined with a hyphen).
    - Decide if phrases like "San Francisco" should be one token or two.

- **Other languages introduce additional issues**
  - **French:** Decide if "L'ensemble" is one token or two.
  - **German:** Long compounds like "Lebensversicherungsgesellschaftsangestellter" are not separated.
  - **Chinese/Japanese:** No spaces between words make tokenization harder.

Data Cleaning via Tokenization

Tokenization breaks down text into meaningful units, but it's just the start. Here are additional techniques:

Stemming: Reducing words to their **stem/base or root form** (e.g. “play” from “playing”) by removing suffixes.

Lemmatization: Converting a verb to its infinitive form (lemma). E.g. The lemma of "running" is "run."

Typo Correction: Replace a word with the word in our dictionary with the nearest **edit distance.** E.g. Correcting "helo" to "hello" based on dictionary suggestions.

Stop Words: Handle common words like "the" or "and" by removing them to focus on meaningful content.

STEMMING

Reduce terms to their stems in information retrieval

Stemming is crude chopping of affixes

- language dependent
- e.g., **automate(s)**, **automatic**, **automation** all reduced to **automat**.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Lemmatization

Lemmatization

- Reduce inflections or variant forms to base form
  - am, are, is → be
  - car, cars, car's, cars' → car
- the boy's cars are different colors → the boy car be different color

However, lemma is an actual language word, whereas stem may not be an actual word.

Example: Consider the word "running."

Stemming: It might be reduced to "runn."

Lemmatization: It would be converted to "run."

Stemming often produces shorter, approximate forms like "runn," while lemmatization aims for the actual base form of the word, "run."

Next Steps

Now we can turn books into vectors. Now what? We have a set of **labeled vectors,** and we are given a new book to classify. What do we do?

Next Steps

Now we can turn books into vectors. Now what? We have a set of **labeled vectors,** and we are given a new book to classify. What do we do?

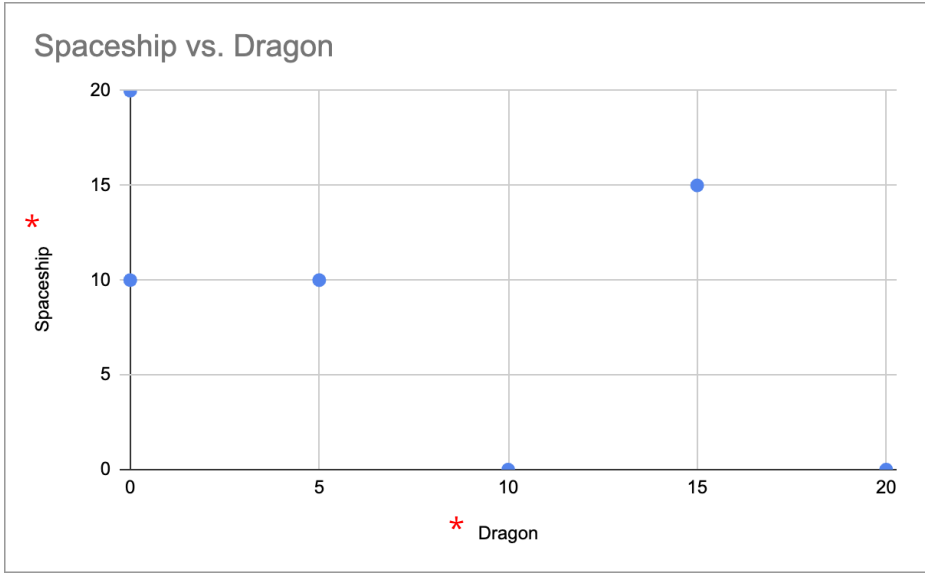
We could try a KNN approach.

A Thought Experiment: Seeing if two books are on the same topic

Imagine books made up of only two words: **Spaceship or dragon.**

The more occurences of "spaceship" it has, the more purely science fiction it is. The more occurences of "dragon" it is, the more fantasy.

Looking at this graph, **should we use Euclidean distance?**



* Frequency of words

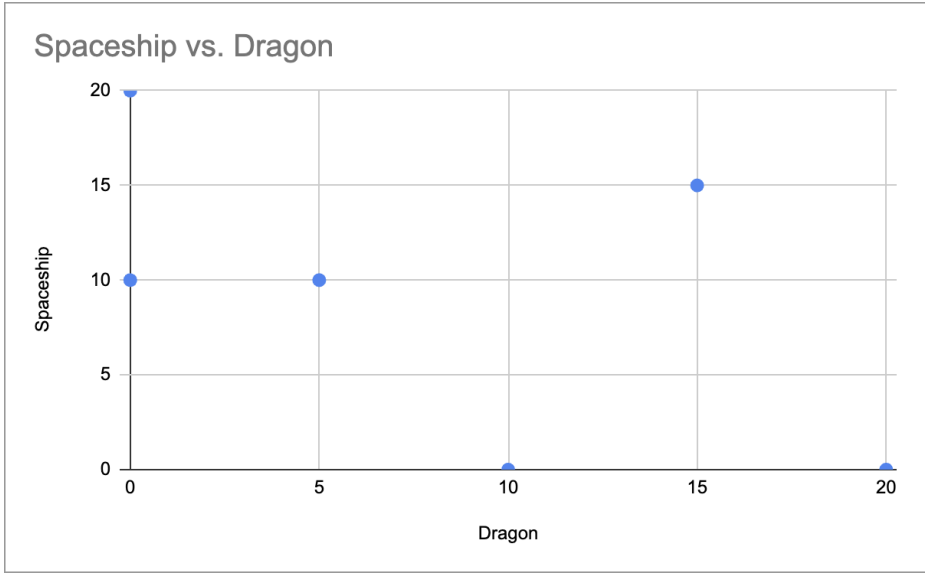
How similar these books are?

A Thought Experiment: Seeing if two books are on the same topic

Looking at this graph, **should we use Euclidean distance?**

May not sufficient, why?

it may not capture similarity accurately if magnitudes (or frequencies) differ significantly.

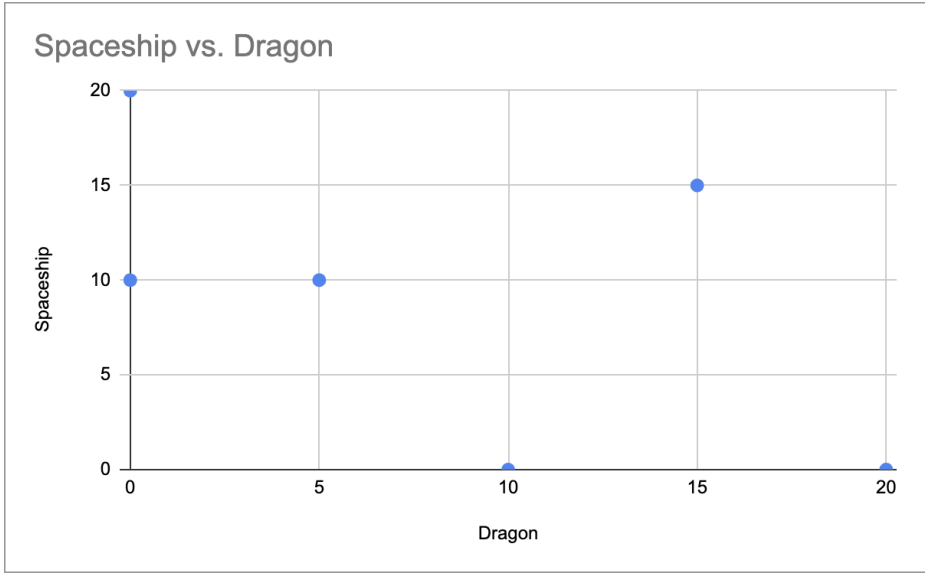


A Thought Experiment: Seeing if two books are on the same topic

Looking at this graph, **should we use Euclidean distance?**

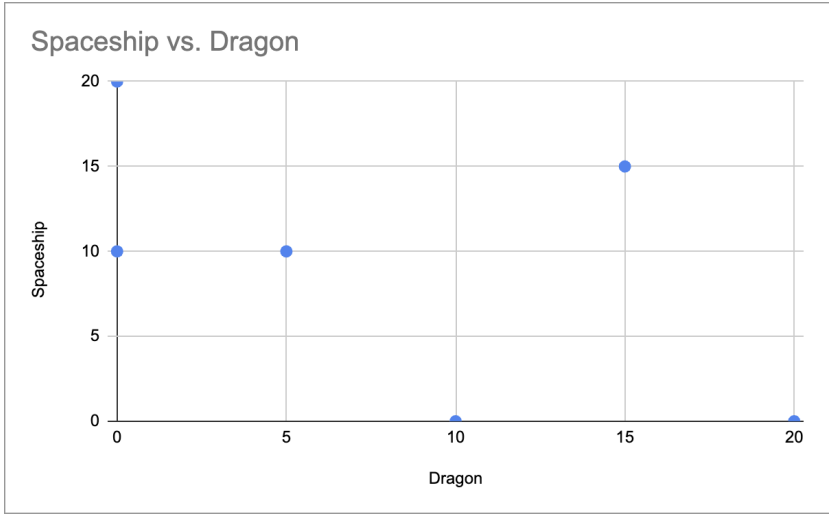
Example: Two books with similar occurrences of "spaceship" and "dragon" may appear close in Euclidean distance.

- But differences in magnitudes (total occurrences) can lead to dissimilarities.



A Thought Experiment: Seeing if two books are on the same topic

Looking at this graph, **should we use Euclidean distance?**

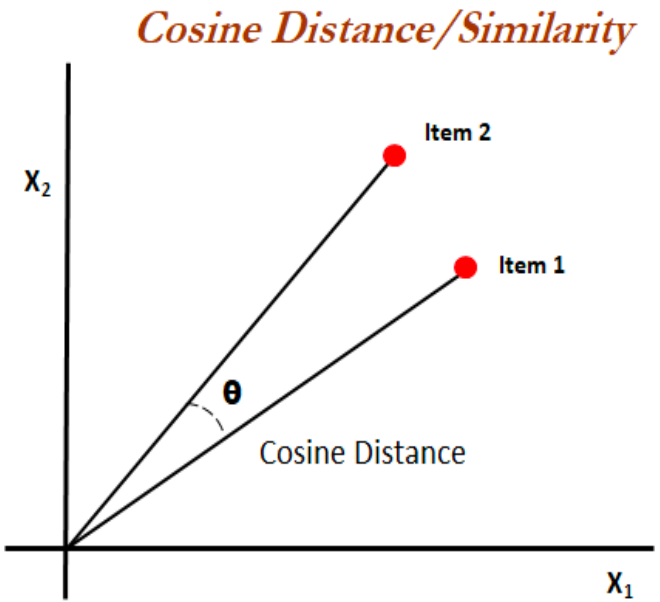


For instance, using "dragon" 10 times versus 100 times is actually similar, but Euclidean distance might not reflect this accurately.

- This is because Euclidean distance focuses on the spatial distance between points rather than the angle between them

We need a new measure!!

Cosine Similarity

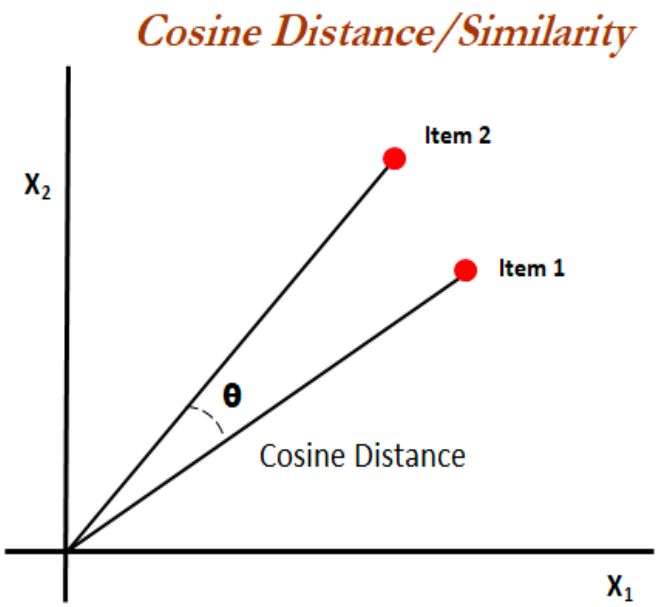


Cosine similarity is a measure of the angles between two vectors.

- Emphasizes the direction, not the magnitude (total occurrences), of the vectors.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Cosine Similarity



The resulting value ranges from -1 (completely dissimilar) to 1 (completely similar).

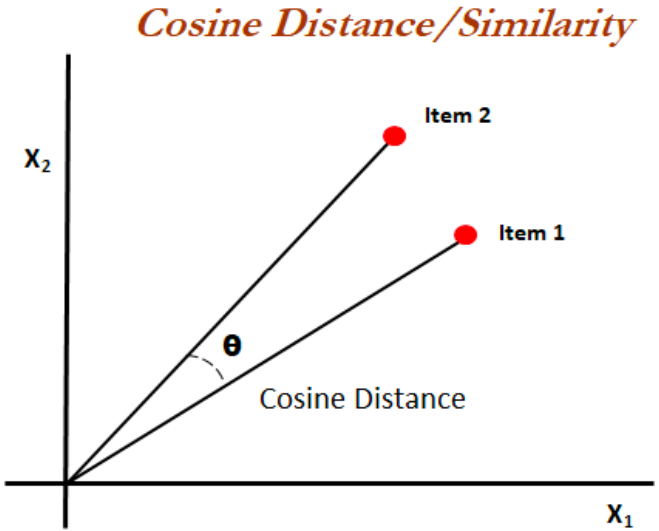
- Value of 1 : the vectors have the same directions/orientations,
- Value of -1: indicates opposite orientations.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Cosine Similarity

Books on similar topics will have **smaller angles.**

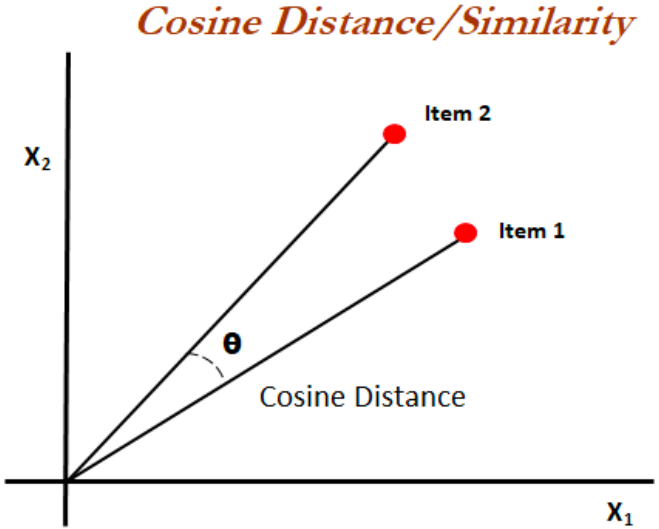
- smaller angles indicates greater similarity → similar word distributions, even if the magnitudes (total word counts) differ.



Cosine Similarity

Books on similar topics will have **smaller angles.**

- Cosine similarity will be high Suggests that even though **the magnitudes are different, the direction of the vectors (word frequencies) is similar,**indicating that the documents are likely on the same topic.



Our Process (so far)

- **Pre-process:** Depending on our domain, figure out of we want to do stemming, remove stop words, etc.
- **Create a bag of words** by one-hot encoding all the words in our document.
- When getting a new vector, **use cosine similarity** to find the nearest k vectors and classify accordingly.

What about this issue?

Let's include another common, **non-stop word**

- common words can overshadow more meaningful ones like “dragon”

Dragon	Spaceship	They
10	0	100
20	0	110
0	10	90
0	20	80
5	10	100
15	15	80

Next we will talk about “TF-IDF”

TF-IDF (Term Frequency-Inverse Document Frequency) - **a way of measuring how relevant a word is to a document** in a collection of documents.

Before that, let’s introduce some metrics: TF, DF.

Term Frequency (TF)

- Term frequency--**how many times a term/word appears in a given document.**

	Dragon	Spaceship	They
Document A	10	0	20
Document B	20	0	25
Document C	0	10	30
Document D	0	20	35
Document E	5	10	20
Document F	0	0	40

Measures the frequency(or occurrence) of a specific term (word) in a document.

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

Example: Term Frequency (TF) Calculation

Suppose we are looking for documents using the query Q and our database is composed of the documents D1, D2, and D3.

Q: The cat.
D1: The cat is on the mat.
D2: My dog and cat are the best.
D3: The locals are playing.

APPLY:
TF(word, document) = “number of occurrences of the word in the document” / “number of words in the document”

TF(“the”, D1) = 2/6 = 0.33

TF(“the”, D2) = 1/7 = 0.14

TF(“the”, D3) = 1/4 = 0.25

TF(“cat”, D1) = 1/6 = 0.17

TF(“cat”, D2) = 1/7 = 0.14

TF(“cat”, D3) = 0/4 = 0

Document Frequency (DF)

- DF is the **number of documents in which the word is present.**
  - How many documents a term occurs in

    - If the term "technology" appears in 100 out of 500 documents, the Document Frequency of "technology" is 100.

    - Ubiquitous words like "the," "and," or "is", which are common and appear in many documents, tend to have high Document Frequency.

Document Frequency (DF)

- DF is the **number of documents in which the word is present.**
  - How many documents a term occurs in

	Dragon	Spaceship	They
Document A	10	0	20
Document B	20	0	25
Document C	0	10	30
Document D	0	20	35
Document E	5	10	20
Document F	0	0	40

Equation:

DF(t)=Number of documents containing term t

DF(t)		
Dragons	Spaceship	They
3	3	6

How do we use these?

How do we use these?
We want to downweight words with high document frequency

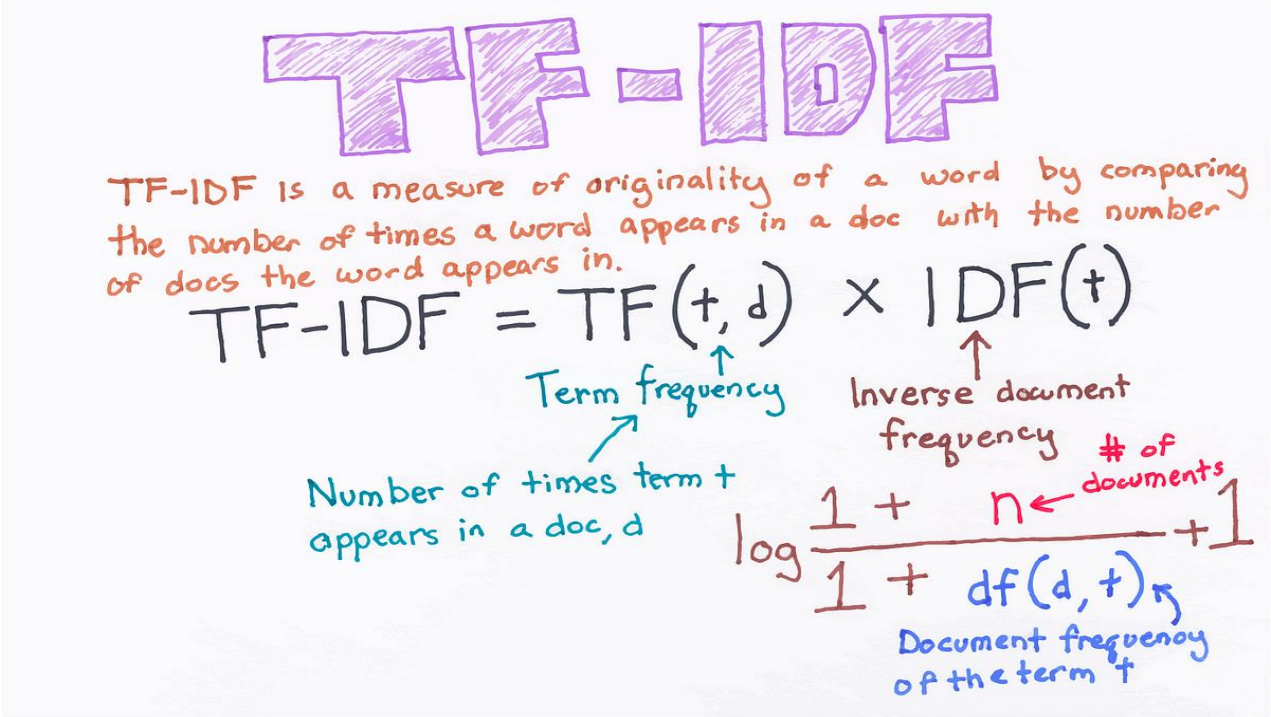
TF-IDF: Term Frequency / Inverse Document Frequency

A statistical measure that **evaluates how relevant or important a word is to a document** in a collection of documents.

This is done by multiplying two metrics:

1. **TF (Term Frequency):** how many times a word appears in a document and
2. **IDF (the inverse document frequency)** of the word across a set of documents.
    a. Rare words have high scores, common words have low scores.

TF-IDF: Term Frequency / Inverse Document Frequency



TF-IDF Origins

TF-IDF was first designed for document search and information retrieval, where a query is run and the system has to find the most relevant documents.

TF-IDF has many uses, such as in information retrieval, text analysis, keyword extraction, and as a way of obtaining numeric features from text for machine learning algorithms.

Suppose the query is the text “The bug”. The system would give each document a higher score proportionally to the frequencies of the query words found in the document, weighting more rare words like “bug” with respect to common words like “the”.

How to compute TF-IDF

Term Frequency Inverse Document Frequency

TF-IDF assigns higher weights to terms that are frequent in a document but rare across the entire document collection.

USE FORMULA:

IDF = Log[(Number of documents) / (Number of documents containing the word)]

Way 01:

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Variants of inverse document frequency (idf)

Variants of inverse document frequency (idf) are used to improve the effectiveness of text analysis and information retrieval.

- Improves term weighting to distinguish relevant documents.
- Lowers weight for common words, focusing on significant terms.
- Improves text classification, clustering, and retrieval.

Variants of inverse document frequency (idf)

We are going to discuss two variants:

- N = Number of documents
- N_t = Number of documents containing the word / term t

Weightning Scheme	IDF Equation
Base / Standard	$IDF = \log(N/N_t)$
Smooth IDF variant 1	$IDF = Log[(1+N) / (1+N_t)]+1$
Smooth IDF variant 2	$IDF = Log[(1+N) / (1+N_t)]$

Standard TF - IDF Calculation: TF * IDF = TF * log(N/Nt)

Document 1: read SVM algorithm article dataaspirant blog
Document 2: read randomforest algorithm article dataaspirant blog

TF-IDF Calculation Example

Words	Count		Term Frequency (TF)		Inverse Document Frequency (IDF)	TF * IDF	
	Document 1	Document 2	Document 1	Document 2		Document 1	Document 2
read	1	1	0.17	0.17	0	0	0
svm	1	0	0.17	0	0.3	**0.05**	0
algorithm	1	1	0.17	0.17	0	0	0
article	1	1	0.17	0.17	0	0	0
dataaspirant	1	1	0.17	0.17	0	0	0
blog	1	1	0.17	0.17	0	0	0
randomforest	0	1	0	0.17	0.3	0	**0.05**

These documents differ mainly in the algorithms they discuss: SVM for the first and random forest for the second. This distinction is clear in TF-IDF scores, where all other words score 0, while 'SVM' and 'randomforest' have non-zero values

WAY 02: TF-IDF = TF * Log[(1+N) / (1+N_t)]+1

Smooth IDF variant 1

Term Frequency Inverse Document Frequency

TF* IDF =TF* Log[(1+# Number of documents) / (1+Number of documents containing the word)]+1

Consider the given TF (Term Frequency) values and IDF (Inverse Document Frequency) values in the below tables:

Table: TF values

	Dragon	Spaceship	They
Document A	10	0	20
Document B	20	0	25
Document C	0	10	30
Document D	0	20	35
Document E	5	10	20
Document F	0	0	40

Table: DF & IDF values

	Dragons	Spaceship	They
Document Frequency	3	3	6
Inverse Document frequency	1.243038049	1.243038049	1

Table: TF-IDF

	Dragon	Spaceship	They
Document A	12.43038049	0	20
Document B	24.86076097	0	25
Document C	0	12.43038049	30
Document D	0	24.86076097	35
Document E	6.215190243	12.43038049	20
Document F	0	0	40

We used above formula

WAY 03: TF-IDF = TF * Log[(1+N) / (1+Nt)]

Smooth IDF variant 2

Term Frequency Inverse Document Frequency

TF* IDF =TF* Log[(1+# Number of documents) / (1+Number of documents containing the word t)]

Table: TF values

	Dragon	Spaceship	They
Document A	10	0	20
Document B	20	0	25
Document C	0	10	30
Document D	0	20	35
Document E	5	10	20
Document F	0	0	40

Table: DF & IDF values

	Dragons	Spaceship	They
Document Frequency	3	3	6
Inverse Document frequency	0.24303804868	0.24303804868	0

Table: TF-IDF

	Dragon	Spaceship	They
Document A	2.4303804868	0	0
Document B	4.86076097	0	0
Document C	0	2.4303804867	0
Document D	0	4.86076097	0
Document E	1.21519024	2.43038048	0
Document F	0	0	0

We used above formula

After we done the vectorization with TF-IDF, we will apply cosine similarity

Word embedding

Word embedding is a method to convert words into vectors, designed to capture semantic relationships beyond basic numerical representations.

- Word embedding models, like Word2Vec, GloVe, and fastText, learn to map words to vectors based on their contexts in a large corpus of text.
- In python, NLTK is a general NLP toolkit that can be used with various word embedding models

THE FUTURE

THE FUTURE (Or really just the present)

Note: The following will be highly conceptual, and it’s unlikely you’ll be tested on the details.

Feel free to relax and just listen.

Large Language Models

Large Language Models, like ChatGPT, incidentally solve most of our natural language processing problems.

Note here it answers a question, does sentiment analysis, and summarization.

Can you tell me if this review for a professor is positive or negative?

"Justin is the best human being on the planet. As a great communicator, each lecture is planned exceptionally well and each concept is explained thoroughly. He's charming and relatable, and manages to crack some jokes in each lecture. The assignments are easy; the class is no harder than it needs to be. I didn't do any of the optional practice problems from the book and pulled off a 97. The exams are straightforward, just study well and put in the time to succeed."

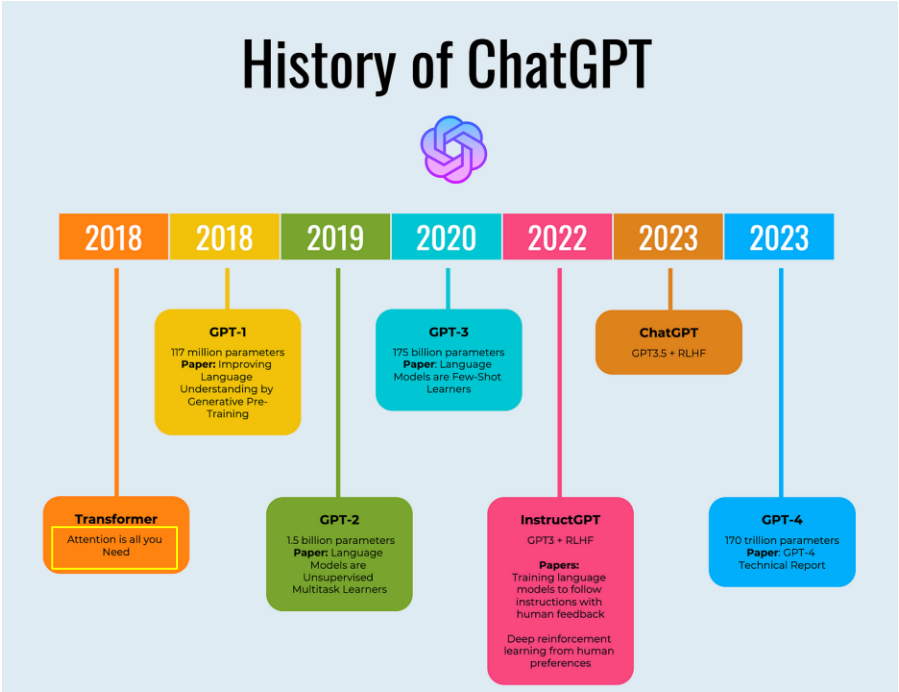
This review is overwhelmingly positive. The student praises the professor, Justin, as the best human being on the planet, commends his communication skills, teaching approach, charm, and relatability. The mention of well-planned lectures, thorough explanations, and the ease of assignments all contribute to a positive assessment. Additionally, the student's personal success in the class, scoring a 97 without doing optional practice problems, further supports the positive nature of the review.

The Evolution of LLMs

LLMs are the culmination of decades of work in Natural Language Processing.

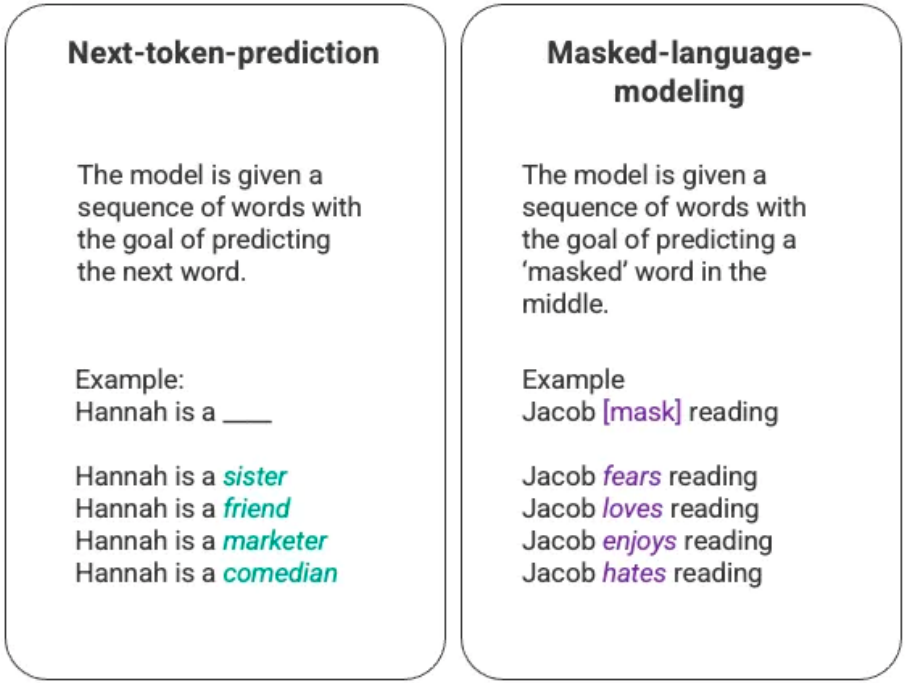
So! Time for more HISTORY.

(Note: The following pictures are just to go "whoa" at. I'm not providing enough information decipher them.)



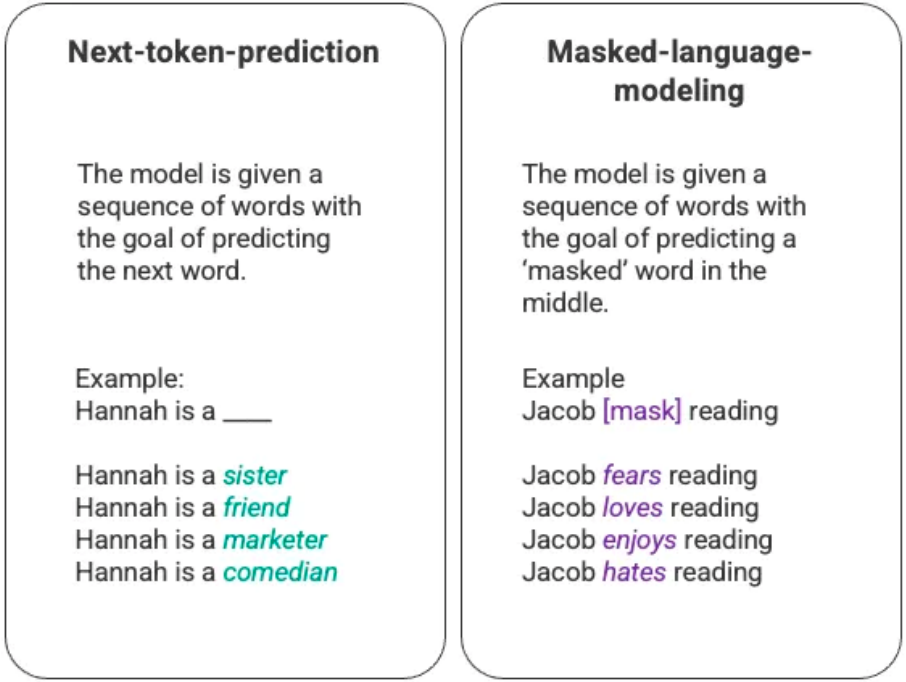
Next Token Prediction

Why do we want to do next token prediction?



Next Token Prediction

- Why do we want to do next token prediction?
- If I can continuously predict the next token, I can write my own sentences!
- Imagine I start with "Five open problems in natural language processing are..." and keep predicting the next token.



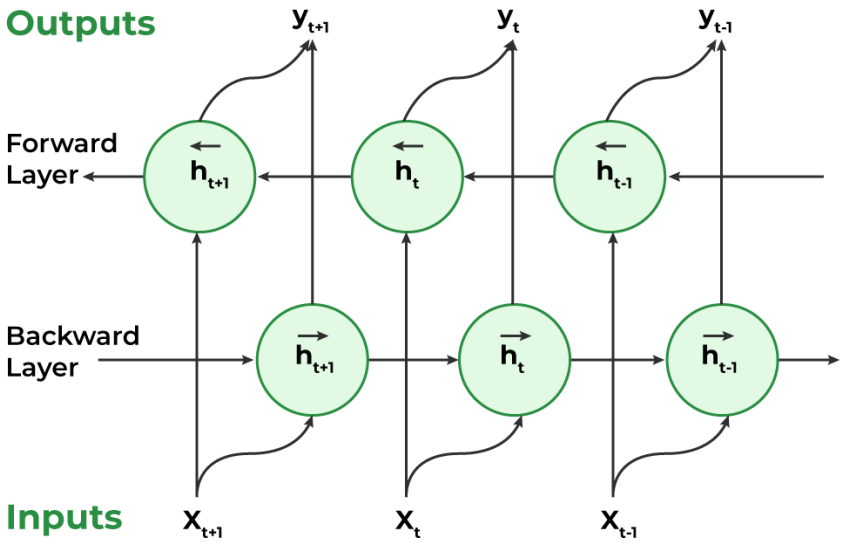
More history:

Early on, we used **feed forward** networks for images.

For natural language processing, we used **recurrent neural networks (RNN),** neural networks with cycles or loop.

- For longer sentences, especially when dealing with sequential data, like language, RNNs are introduced.

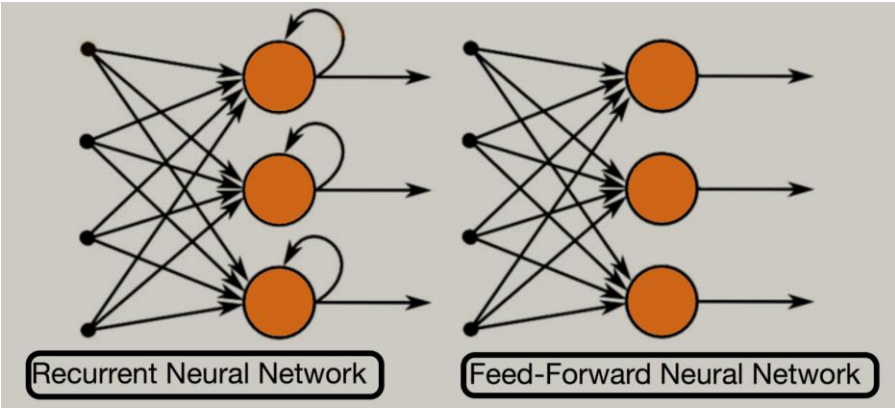
RNNs are especially helpful for longer sentences or sequences, like language, where the order of words matters.



More history:

RNN:

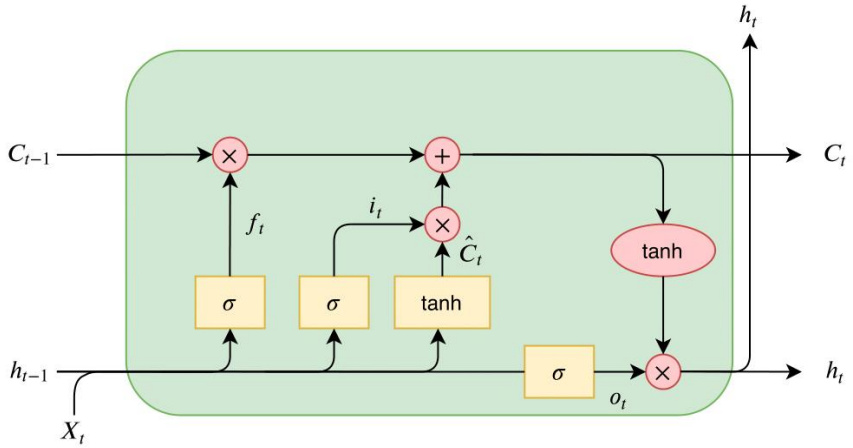
- It processes input data sequentially, and each step depends on the previous one (maintain a memory of past information at each step), making it suitable for tasks where the order of data matters.
- **Challenge:** as sequences get longer, information from earlier steps becomes harder to retain.



More history:

To the right, we see a diagram for a **long-short term memory (LSTM)** neural network.

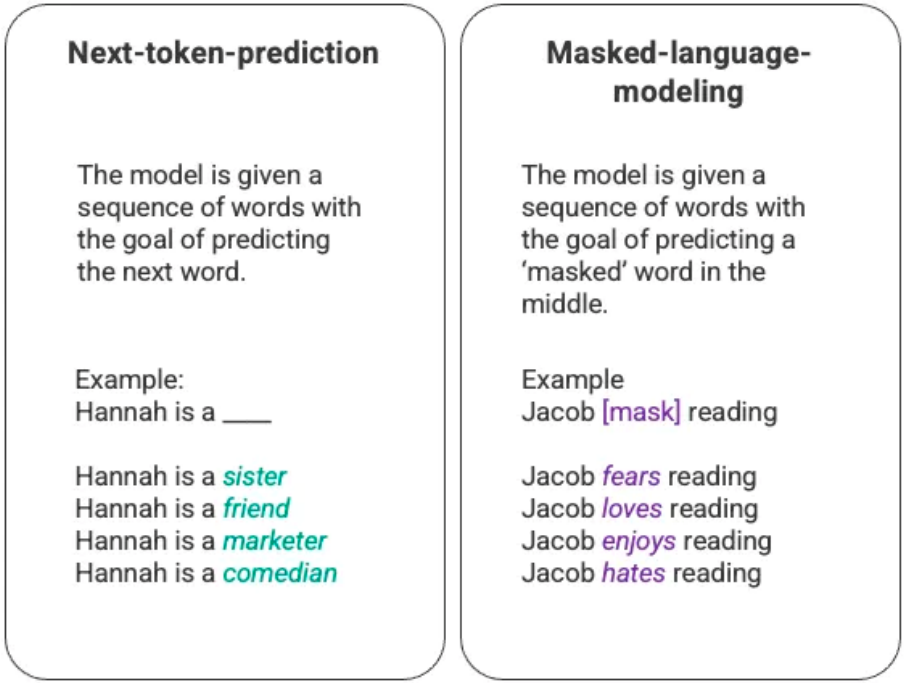
- LSTM (Long Short-Term Memory)
- A type of RNN that uses a more sophisticated memory cell structure.
  - use gating mechanisms to control the flow of information in and out of the memory cell, allowing them to selectively store or forget information.



LSTMs

Issues:

LSTM models is that they treat each word equally in terms of importance, regardless of the surrounding context.



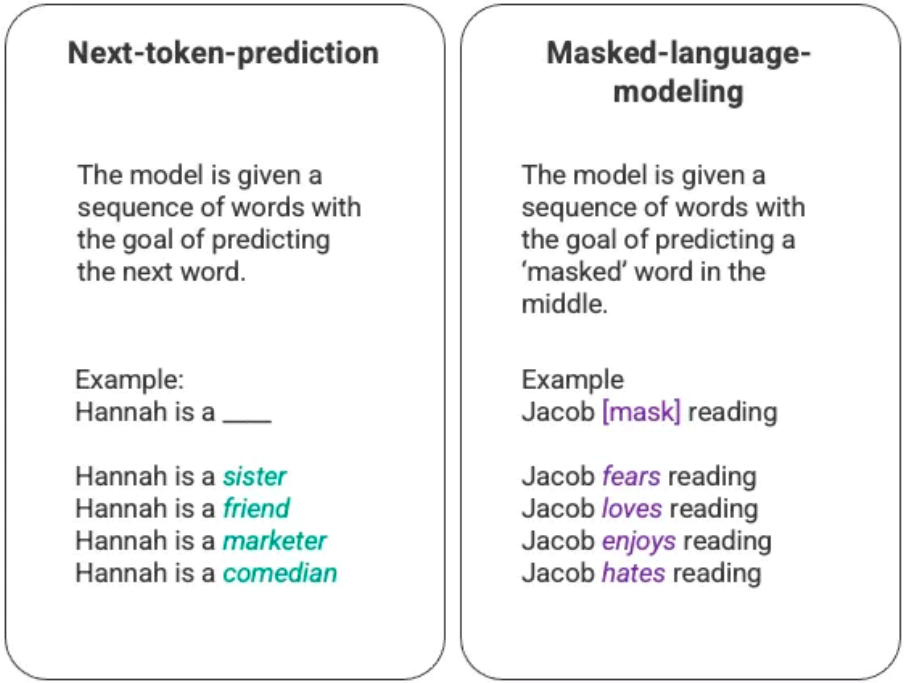
LSTMs

Issues:

1. The model is unable to value some of the surrounding words more than others.

In the given example, while ‘reading’ may most often associate with ‘hates’, in the database:

if Jacob is known as an avid reader, the model should prioritize "Jacob" over "reading," potentially leading to a more positive sentiment prediction like "love" instead of "hates".

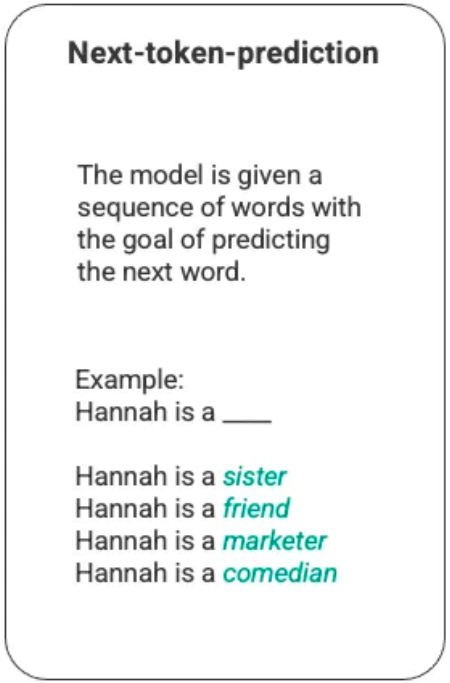


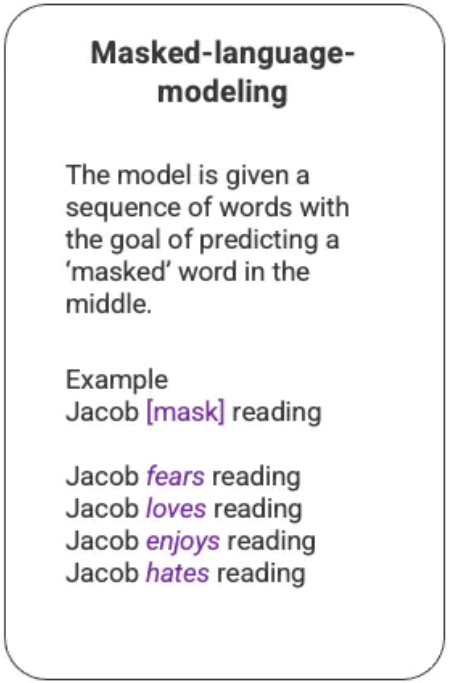
LSTMs

Issues:

2. The input data is processed individually and sequentially rather than as a whole corpus.

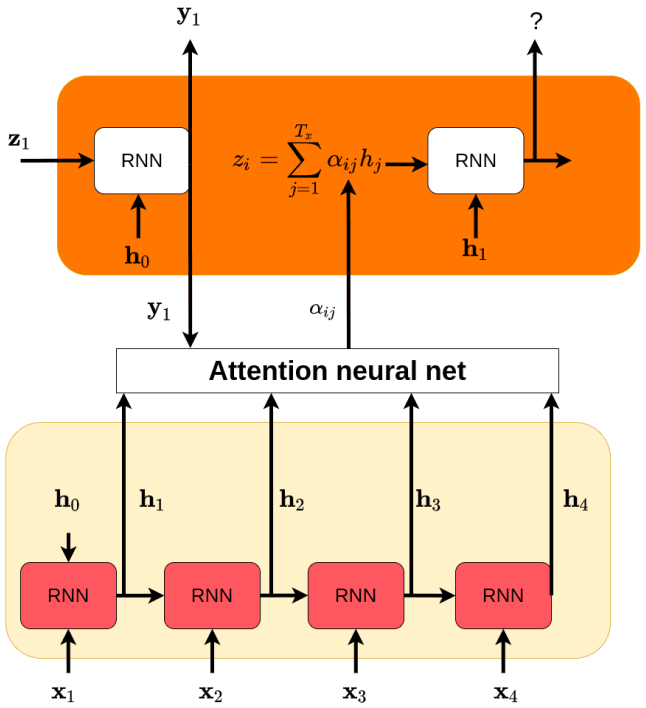
Means, data is handled one step at a time, not all together. Each input is looked at individually, considering only a small number of preceding words as context. This limits how much the model can understand the relationships between words and their meanings





Something Called Attention

Attention is a system where a secondary neural network tells the original neural network where to focus its attention.

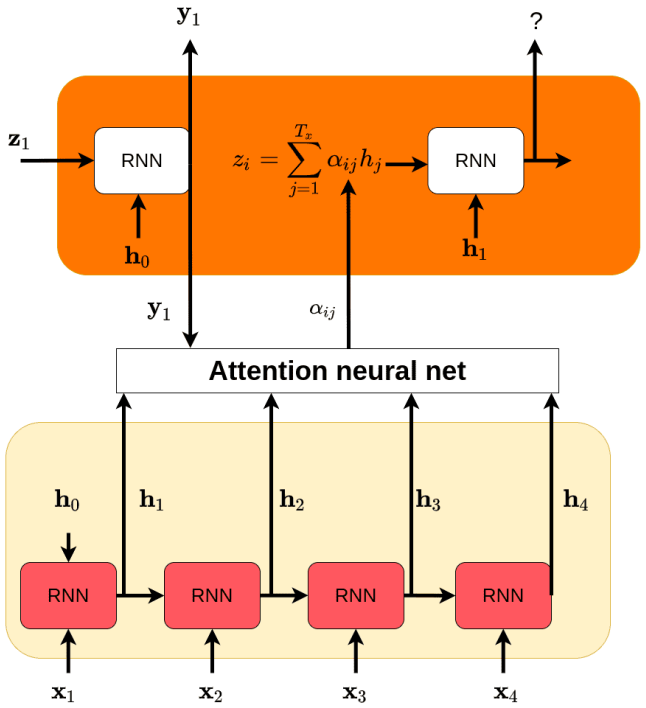


Something Called Attention

Sentence: "The bright red car raced down the winding road."

<u>Without attention:</u> each word is treated equally during processing.

<u>With Attention:</u>
Task: predict the color of the car —> attention might emphasize 'red' and 'bright' during prediction.



Transformers

Attention was used to make **transformers**, a complicated recurrent neural network that forms the basis of large language models.

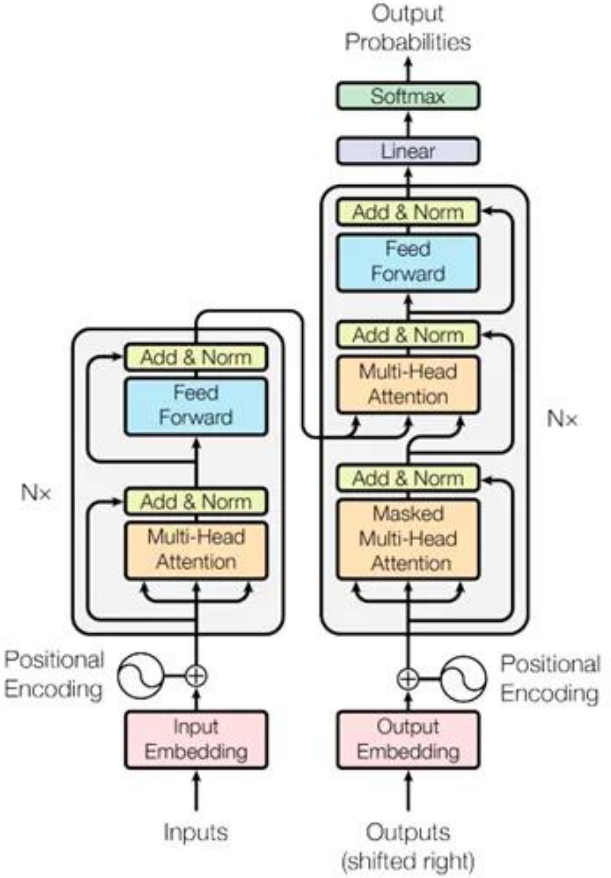


Figure 1: The Transformer - model architecture.

Transformers

Key Innovation: Self-attention mechanism for capturing long-range dependencies in sequences.

Components:

Encoder-Decoder Structure: Commonly used for sequence-to-sequence tasks.

Attention Heads: Multiple parallel self-attention mechanisms.

Positional Encoding: Accounts for the sequential order of input data.

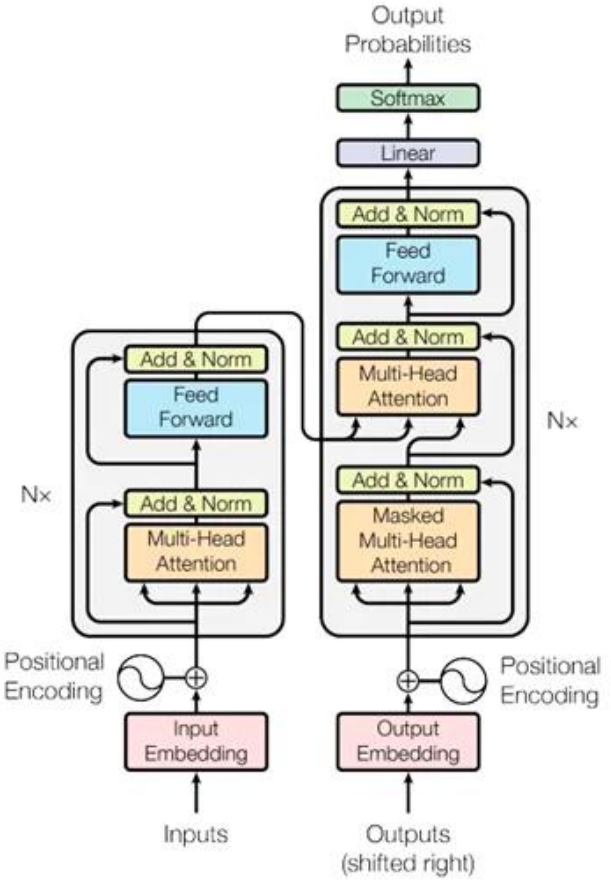


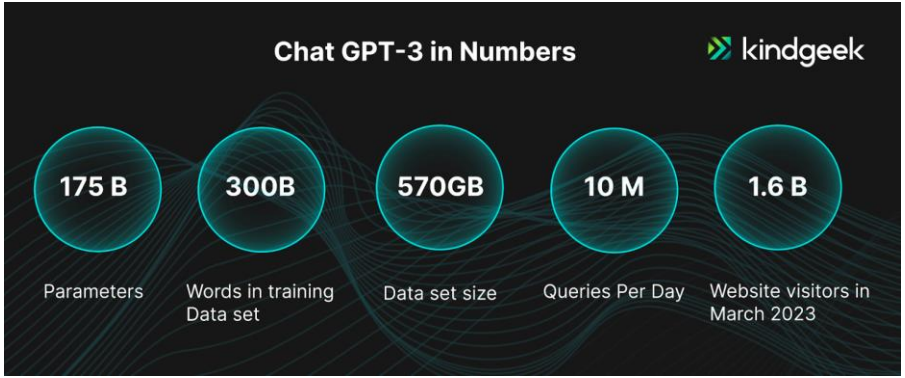
Figure 1: The Transformer - model architecture.

Chat GPT

Chat GPT uses Transformers to predict the most likely next token in any given sentence.

It's been trained over a dataset of 300 billion words.

Is that enough?



Nope! That's stage 1 of training

At this point ChatGPT:

- Often makes things up
- Is *hella* racist
- Is not always very helpful

There are *three more training steps after this.*

Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.

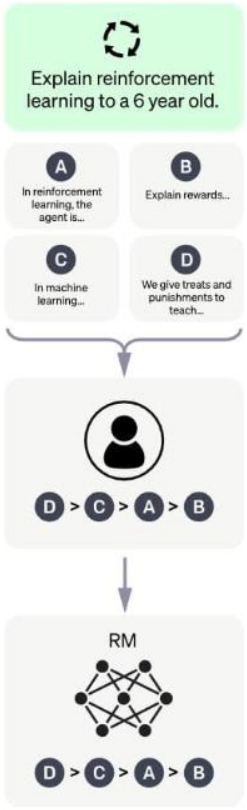
Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

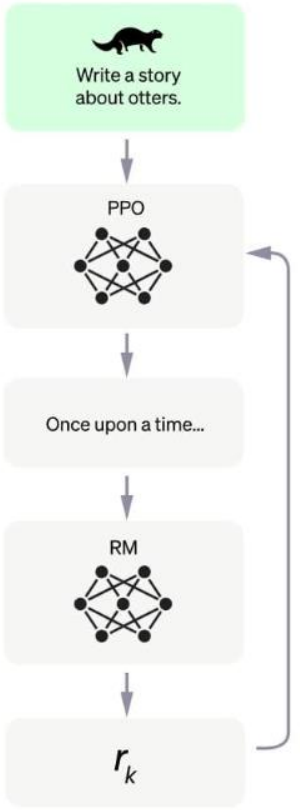
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



The Result

The result of all this is an encoding of the contents of the internet.

ChatGPT:

- Isn't always right; it's optimized to sound correct
- Will sometimes make stuff up

But it is an insanely powerful tool, and LLMs and transformers will continue to be the cutting edge of NLP.