From CS188, FA18, UC Berkeley: http://ai.berkeley.edu.

Natural Language Processing

Introduction to Data Science
Spring 1403

Yadollah Yaghoobzadeh

Agenda

- What is NLP?
- Why NLP is important?
- NLP applications (translation, sentiment analysis, summarization)
- Word embedding (word2vec)
- RNN (sequence processing)
- Attention mechanism, Transformers
- Language modeling (task definition)
- From LMs to general-purpose chatbots

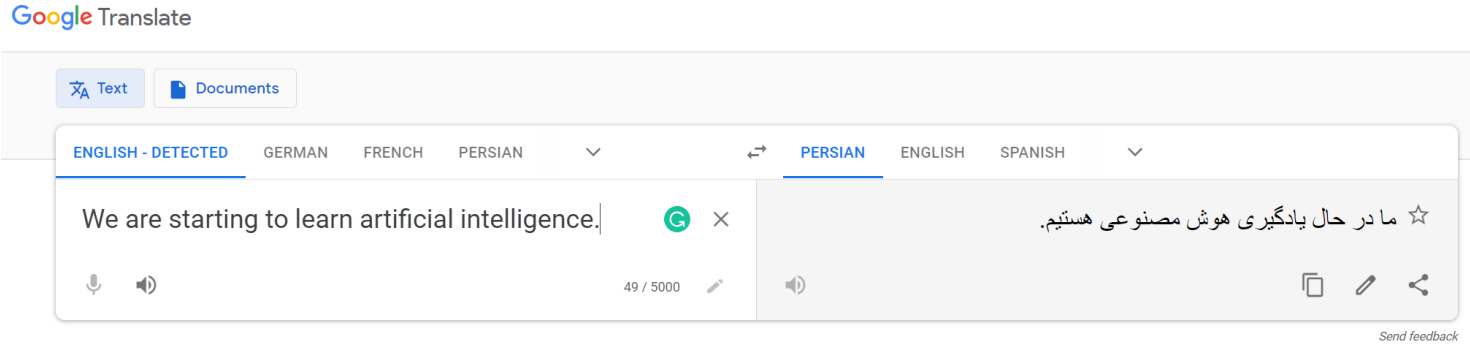
Goal

Comprehension and generation of **natural language**

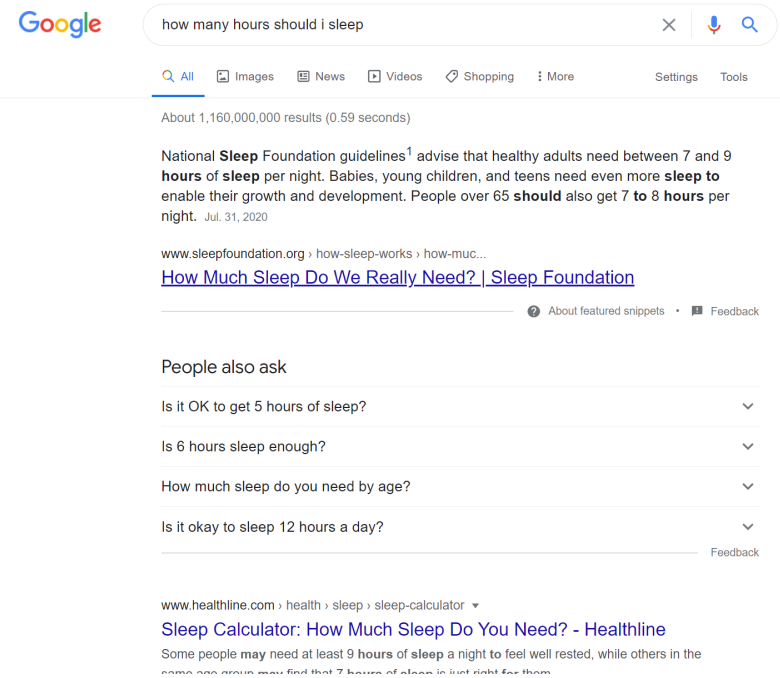
Natural language

- Languages that evolved naturally through human use
  - e.g., Spanish, English, Arabic, Hindi, etc.

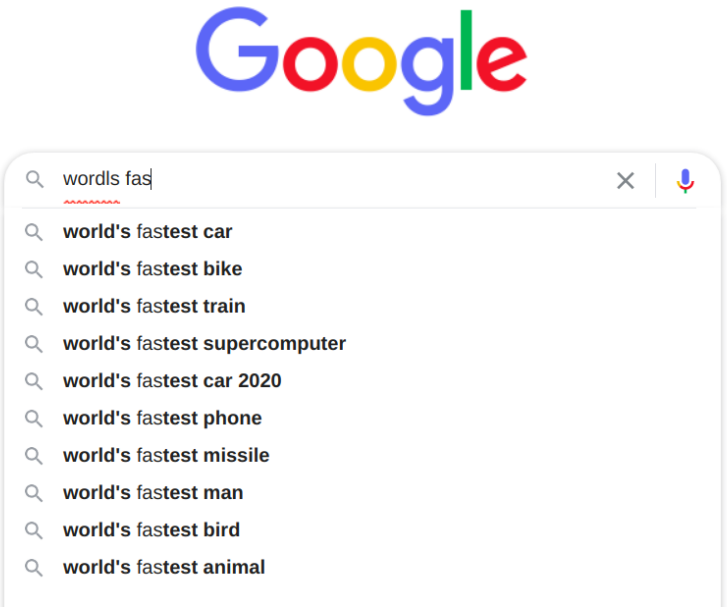
Machine translation



Search & QA



Search autocorrect and autocomplete



Social media analysis

Chatbots

Hiring and recuiretment

Voice assistants

Grammar checkers

The most common type of marketing channel is the wholesale market. Varies kinds of produce are supplied from different areas are assembled at one place and sold thro naller regional markets, etc. Fruits and vegetables s narket handling and transport methods.

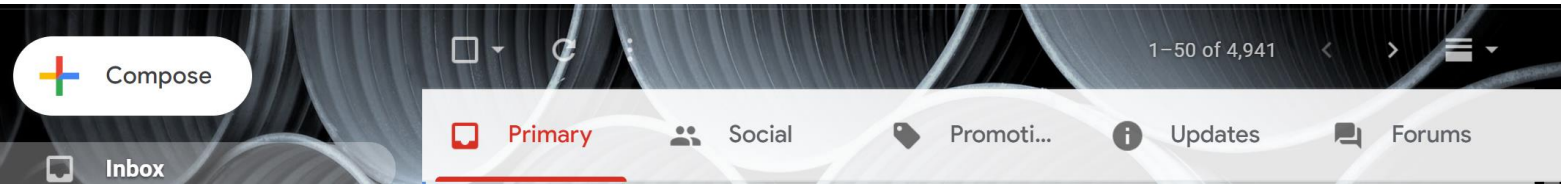
Replace the word

products

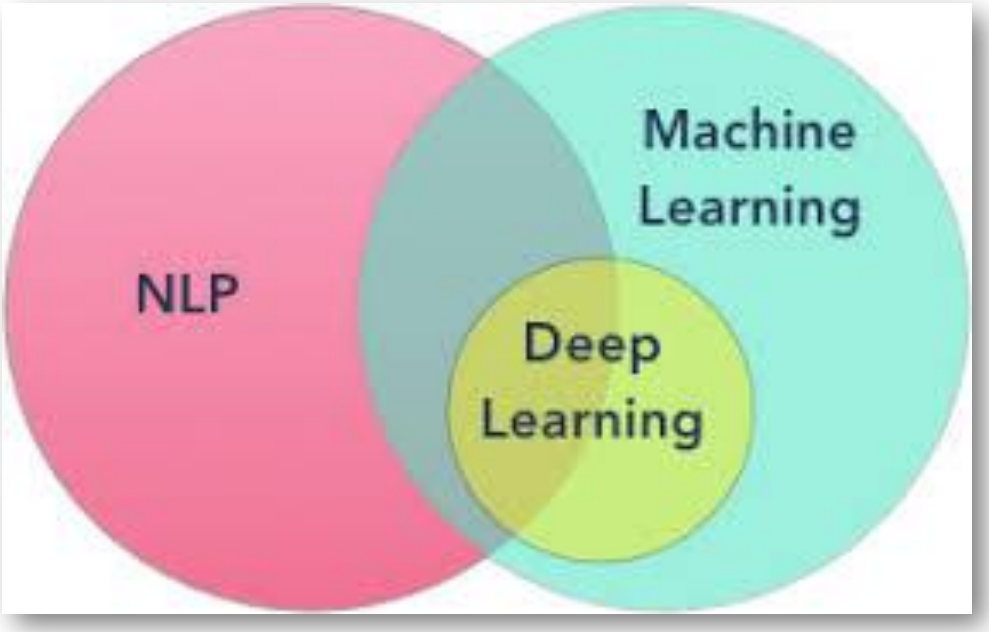
Dismiss

Suggested by Grammarly

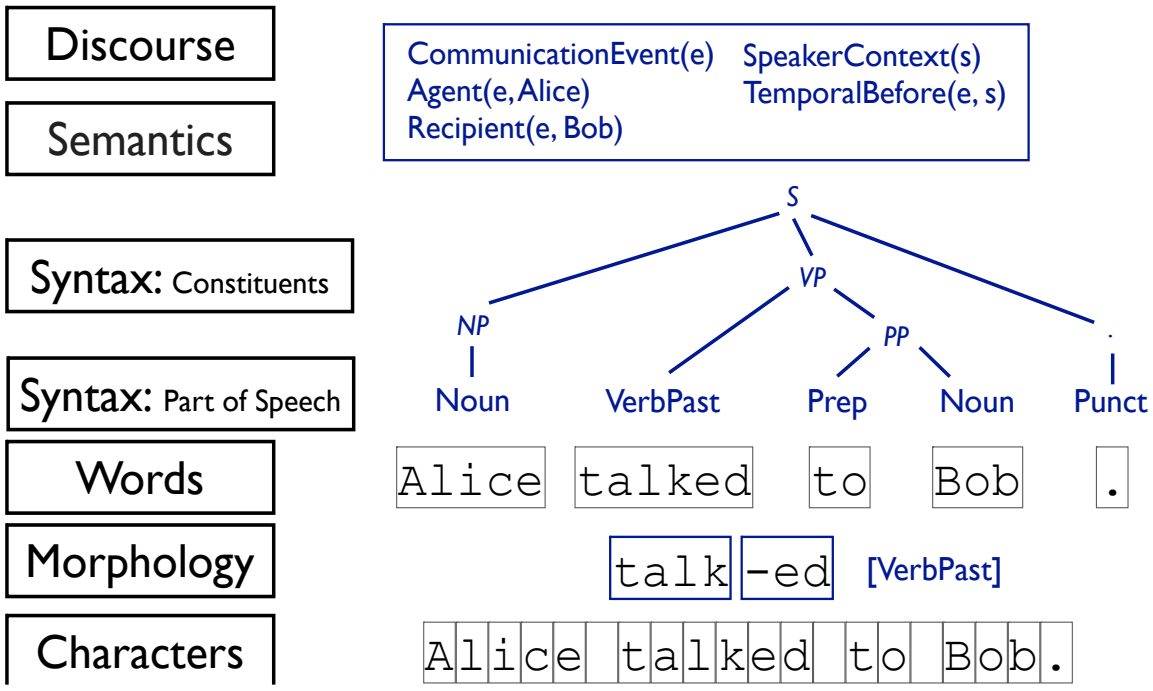
Email classification



NLP is not just machine learning



Levels of linguistic structure

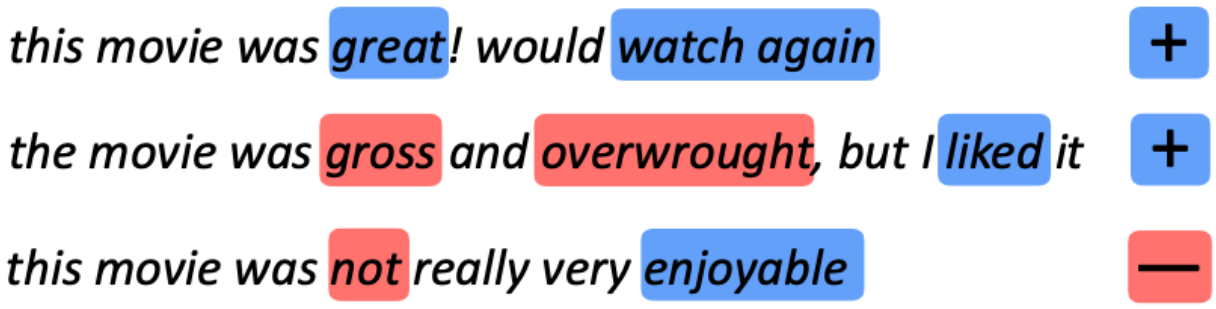


Deep Learning for Text Classification

Classification

- Output a choice from a fixed set of labels
- For sentiment:
  - Positive/negative
  - Star rating
  - …

Some examples of binary sentiment classification



https://www.cs.utexas.edu/~gdurrett/courses/online-course/materials.html

Building a classifier

- Let's say we have 10k labeled sentences
- We want to learn a function f that
  - maps an unseen sentence to one of the labels

From strings to words

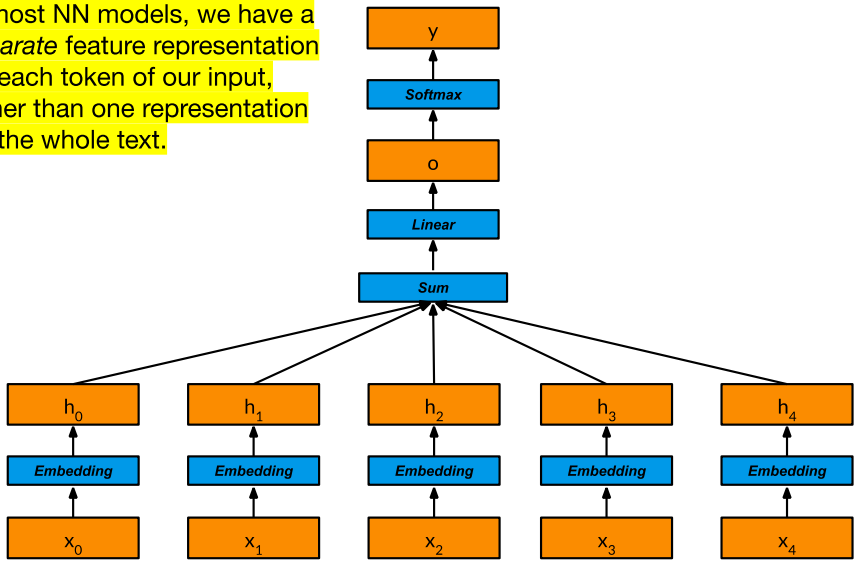
- I don't like any of Ford's trucks.

- I do n't like any of Ford 's trucks .

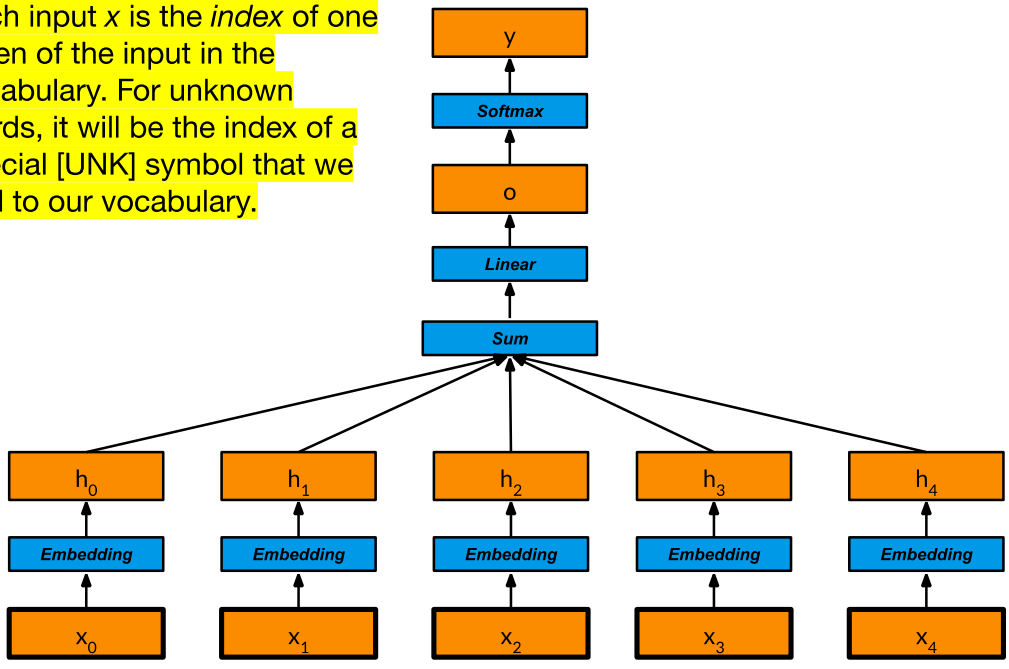
Tokenization: Turning strings into a sequence of symbols (e.g., words, subwords, characters, etc)

NN sentiment classifier

In most NN models, we have a *separate* feature representation for each token of our input, rather than one representation for the whole text.

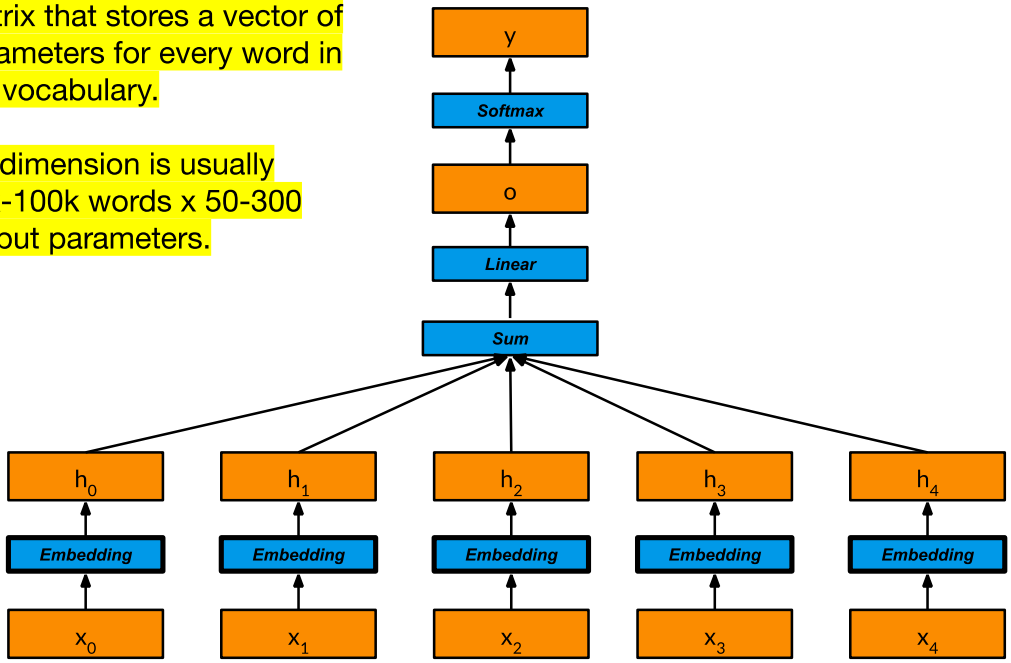


Each input *x* is the *index* of one token of the input in the vocabulary. For unknown words, it will be the index of a special [UNK] symbol that we add to our vocabulary.

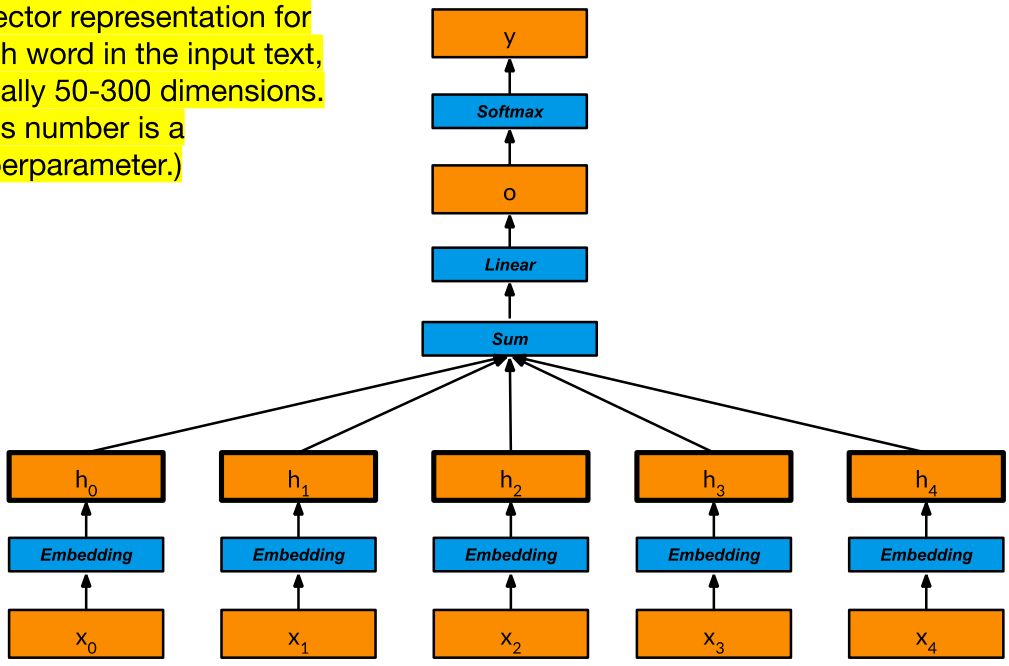


The *embedding layer* is a matrix that stores a vector of parameters for every word in the vocabulary.

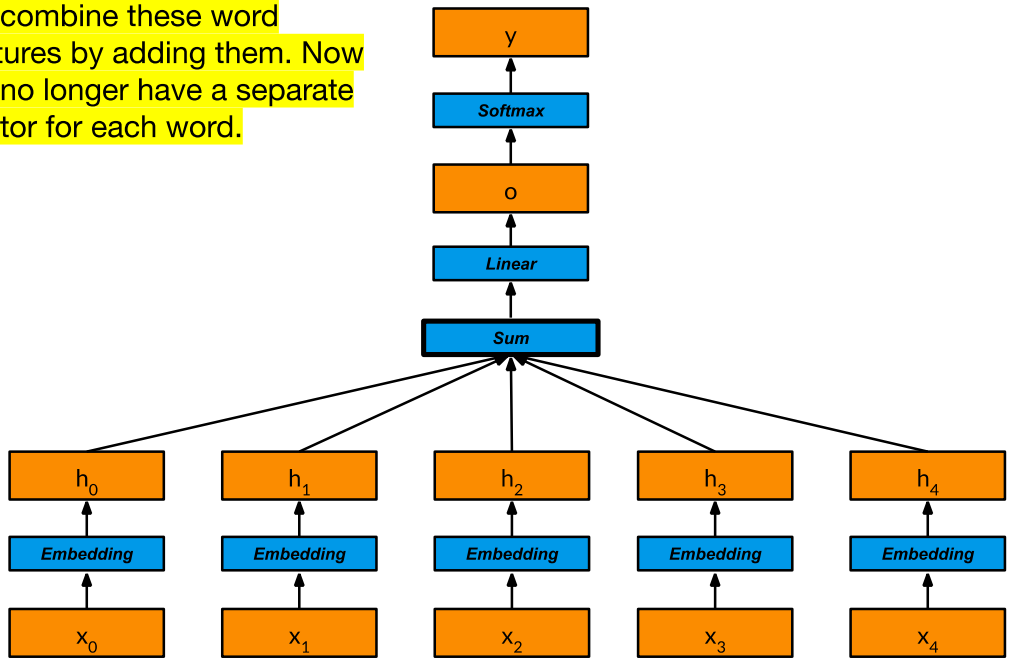
It's dimension is usually 10k-100k words x 50-300 output parameters.



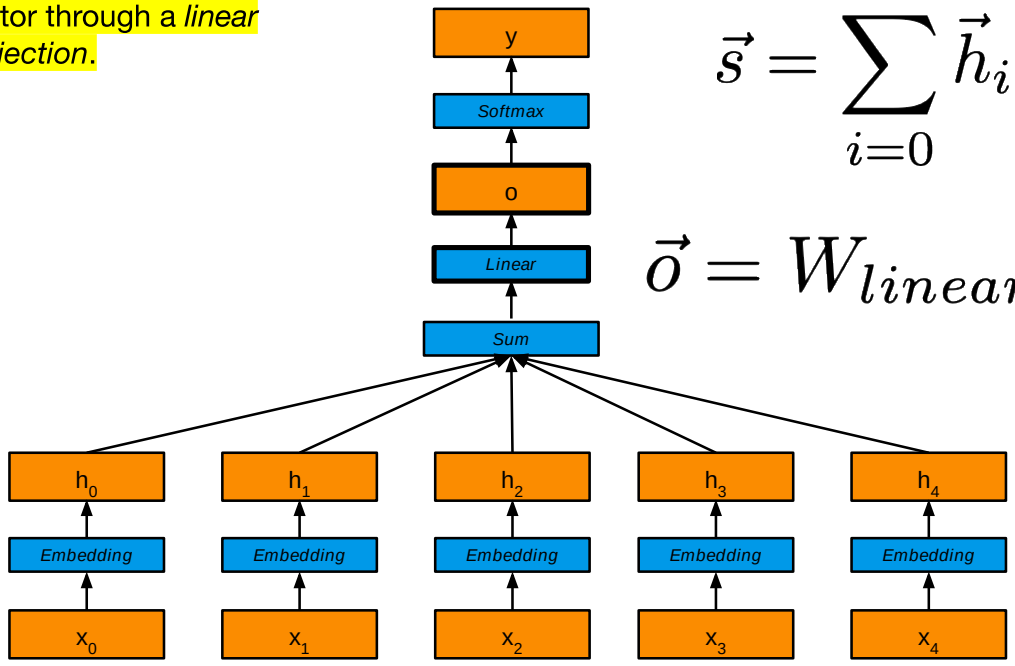
The embedding layer produces a vector representation for each word in the input text, usually 50-300 dimensions. (This number is a hyperparameter.)



In this simple neural network, we combine these word features by adding them. Now we no longer have a separate vector for each word.



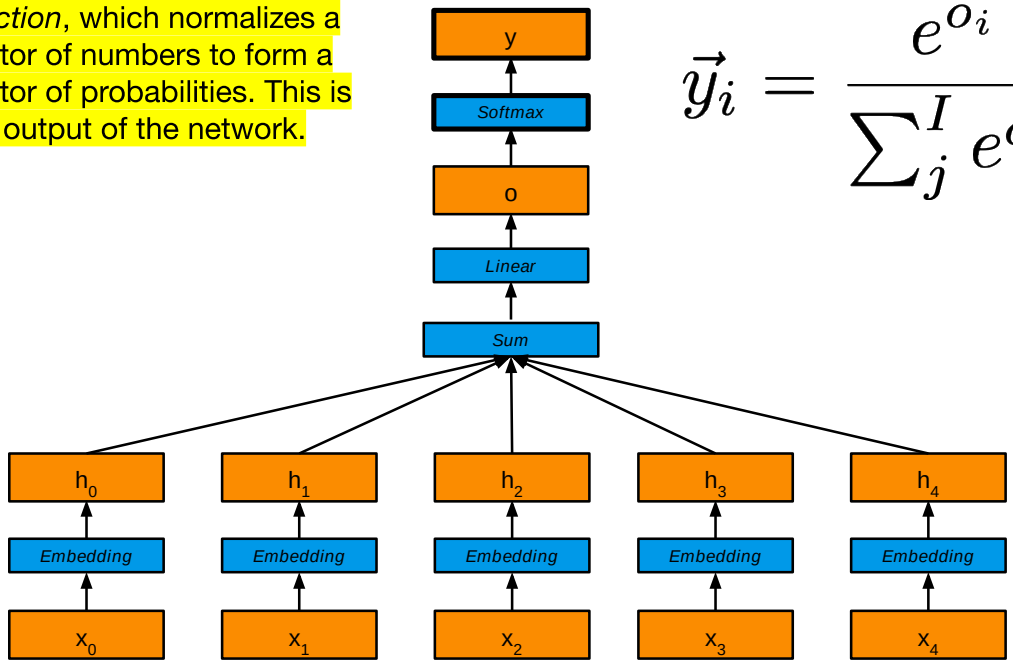
Next, we pass the resulting vector through a *linear projection*.



$$\vec{s} = \sum_{i=0}^{I} \vec{h}_i$$

$$\vec{o} = W_{linear}\vec{s}$$

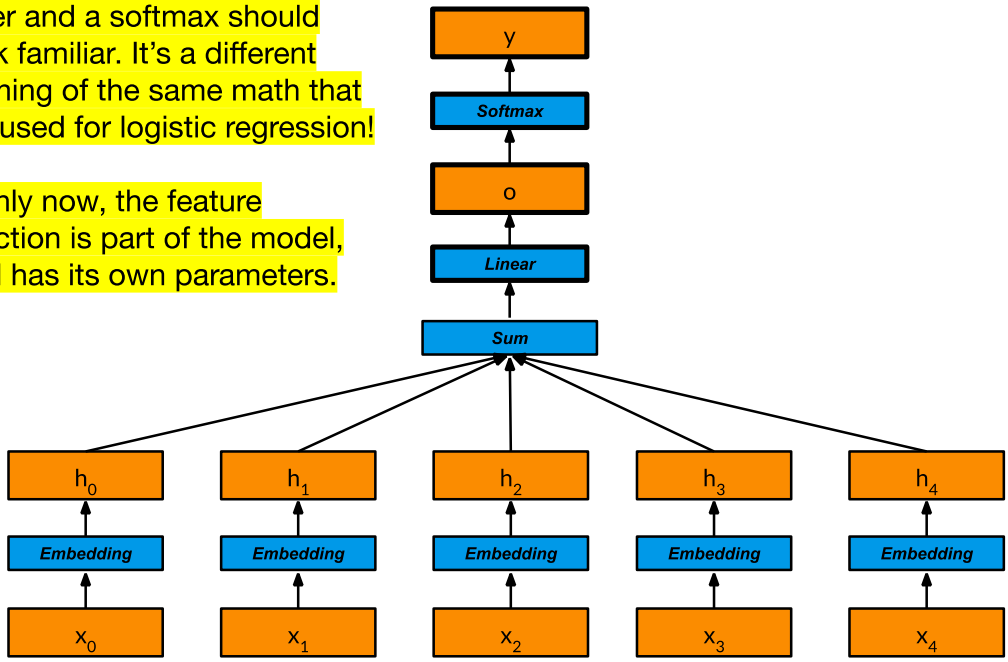
After that, we use the *softmax function*, which normalizes a vector of numbers to form a vector of probabilities. This is the output of the network.



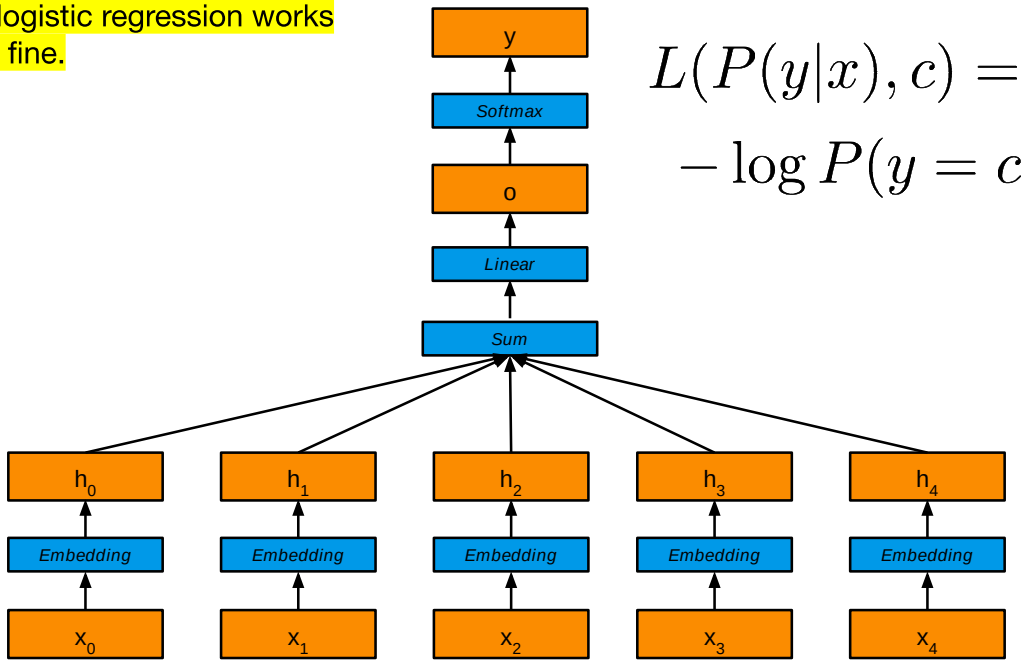
$$\vec{y}_i = \frac{e^{o_i}}{\sum_j^I e^{o_j}}$$

This combination of a linear layer and a softmax should look familiar. It's a different framing of the same math that we used for logistic regression!

...only now, the feature function is part of the model, and has its own parameters.



The *loss function* that we used for logistic regression works just fine.



$$L(P(y|x), c) = -\log P(y = c|x)$$

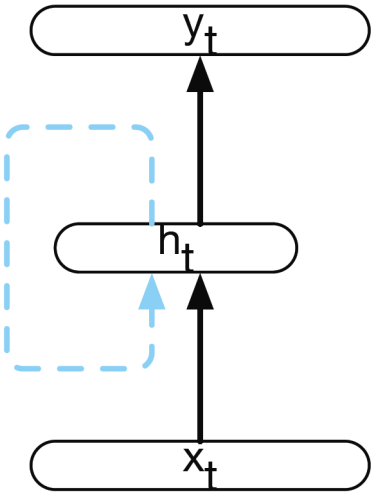
Word embedding

Word embedding

- Each word = a vector
- Similar words are "nearby in space"

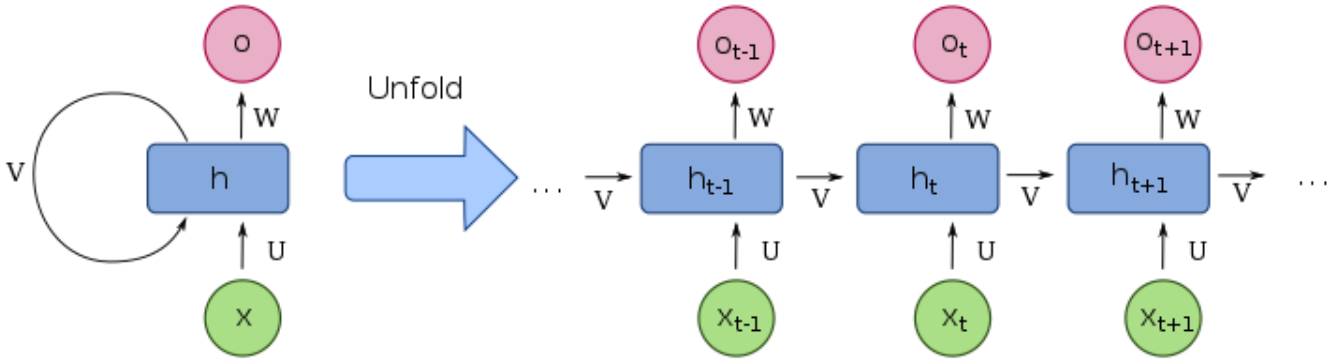
RNNs, attention, transformers

Recurrent Neural Networks (RNNs)

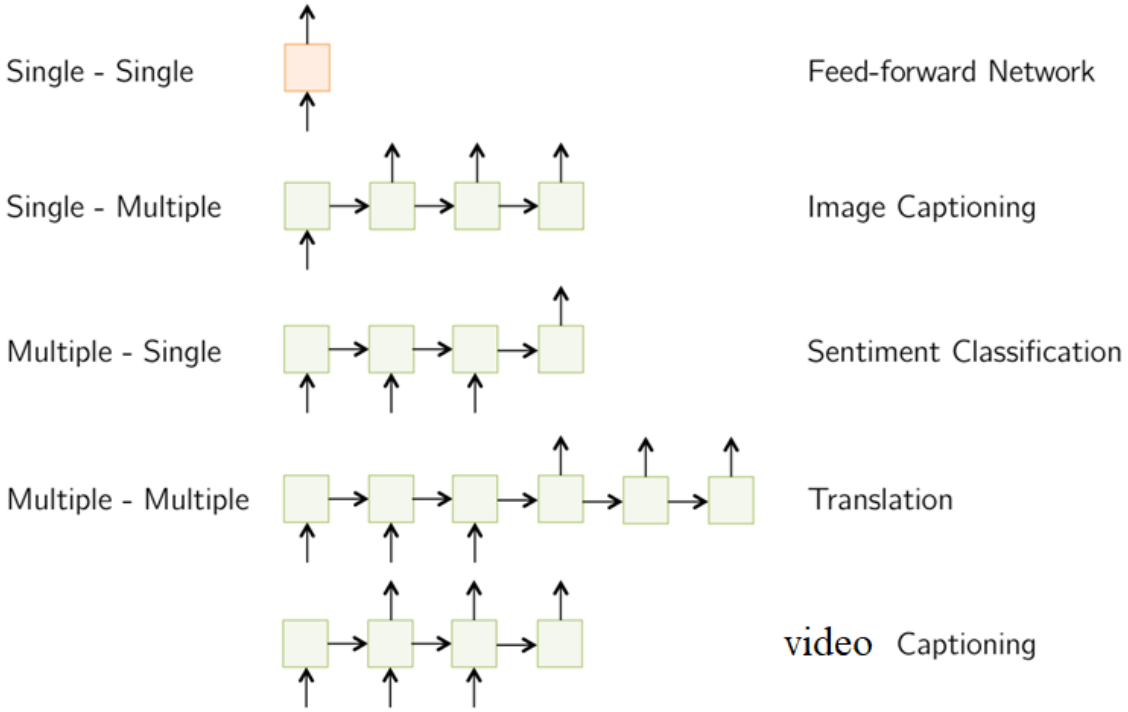


- RNNs take the previous output or hidden states as inputs!
  The composite input at time *t* has some historical
  information about the happenings at time T < *t*
- RNNs are useful as their intermediate values (state) can store information about past inputs for a time that is not
  fixed a priori

Recurrent Neural Networks (RNNs)



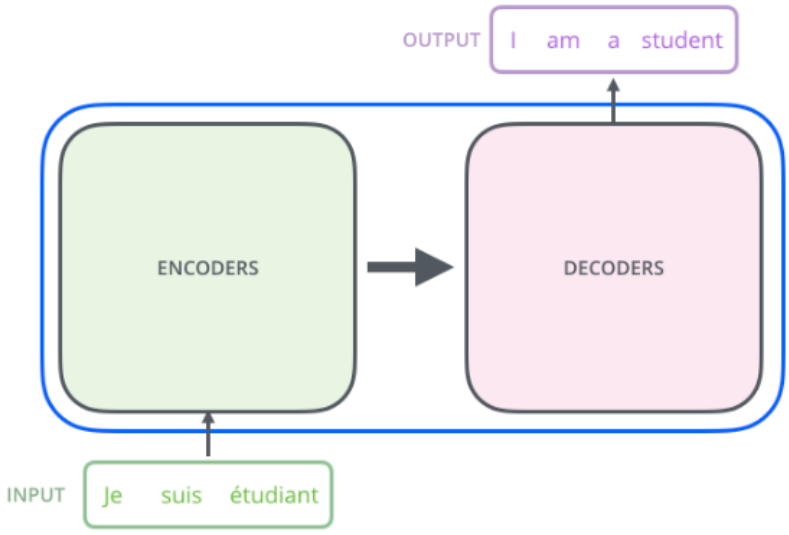
RNNs applications



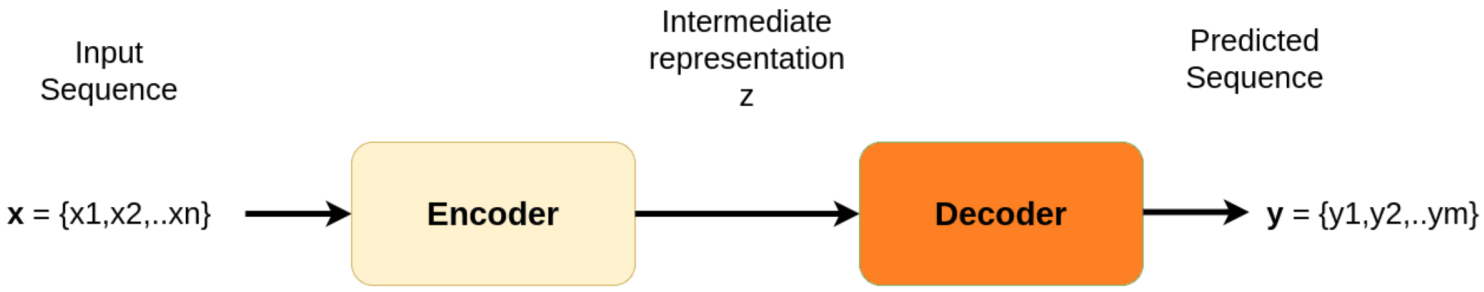
Encoder-decoder networks

- Used in a wide range of applications including machine translation, summarization, question answering, and dialogue modeling.

- RNNs were the most widely-used and successful architecture for both the encoder and decoder.

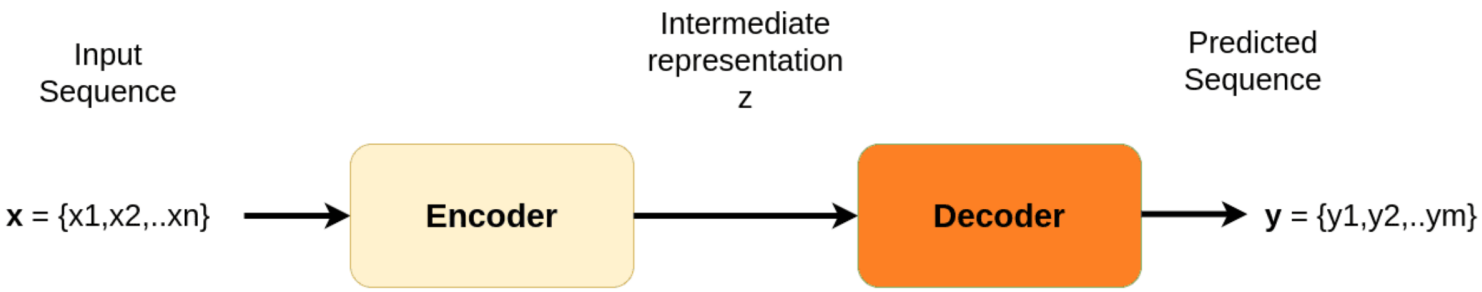


Encoder-decoder: seq2seq

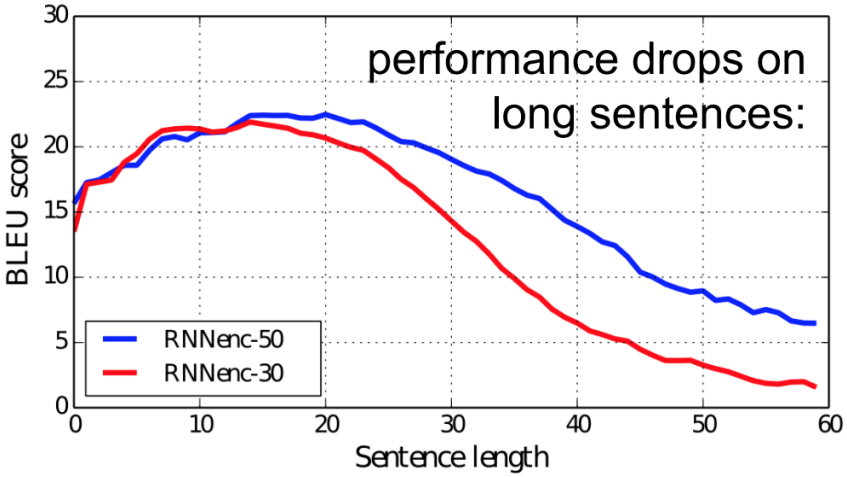


https://theaisummer.com/attention/

What if sequence length is high (say > 30)?



The vector z needs to capture all the information about the source sentence.

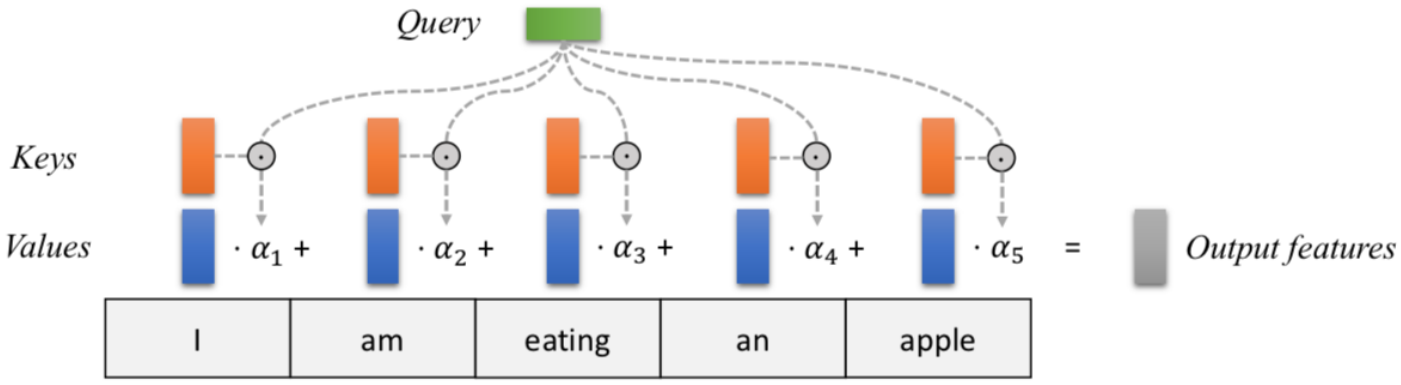


fixed size representation can be the bottleneck

The core idea of **attention** is that the context vector ***z*** should have access to **all** parts of the input sequence instead of just the last one.

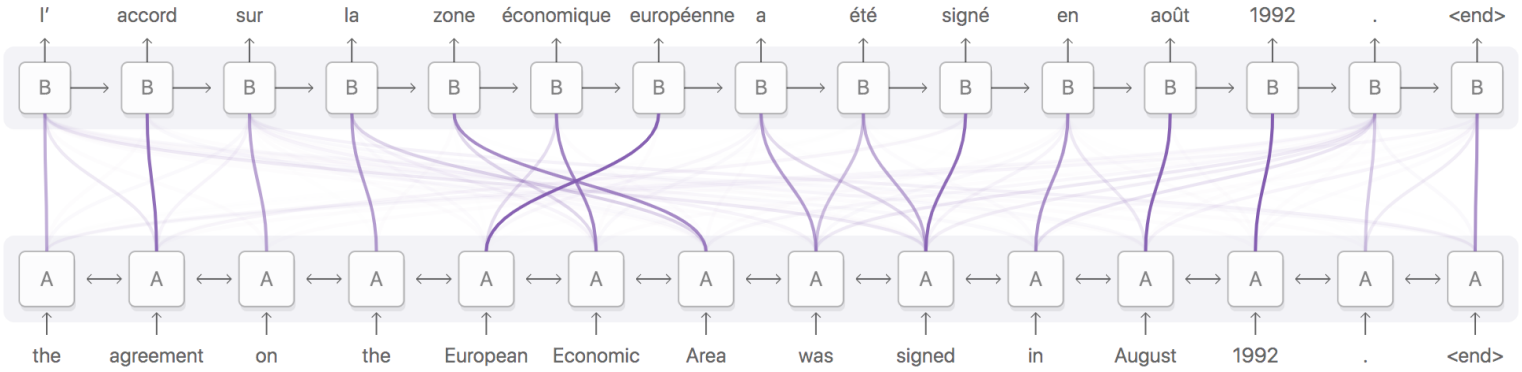
Attention

- A weighted average of (sequence) elements with the weights depending on an input query.
- The idea and the name taken from the intuition that humans attend to certain things at each time.



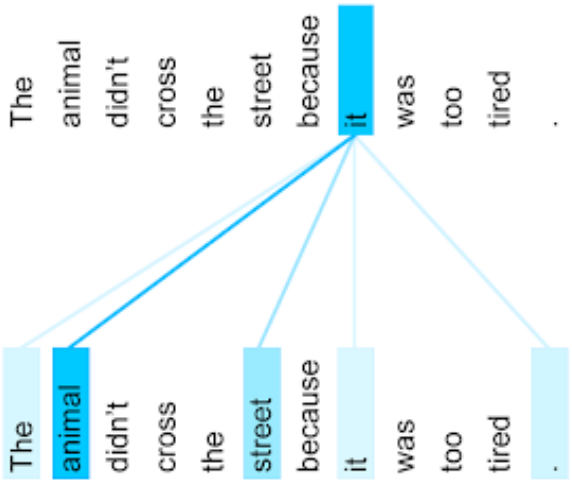
α_i are data-dependent dynamic weights

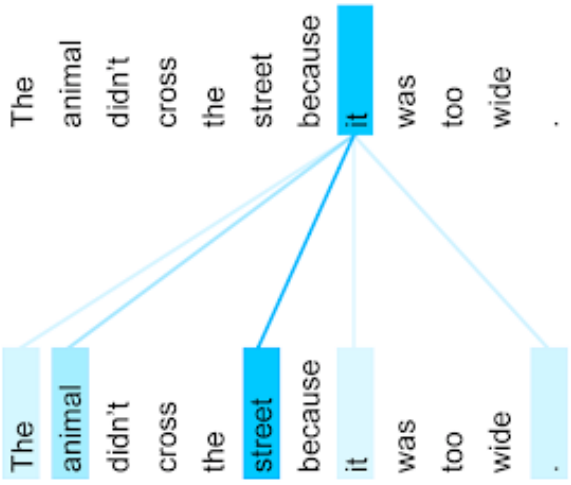
MT



Self-Attention

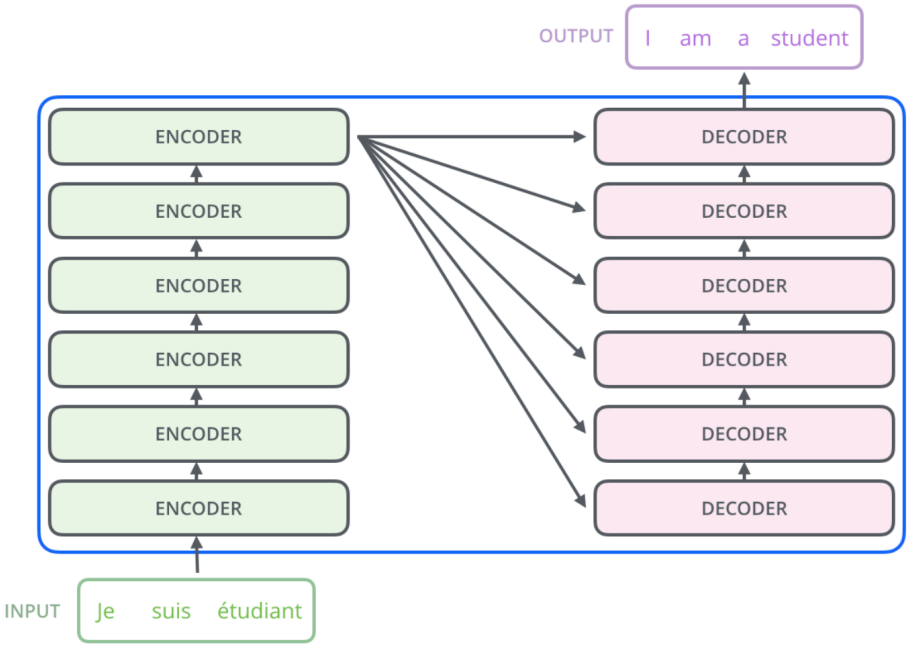
- For each word, self-attention allows the model to look at other positions in the input for a better encoding for this word.





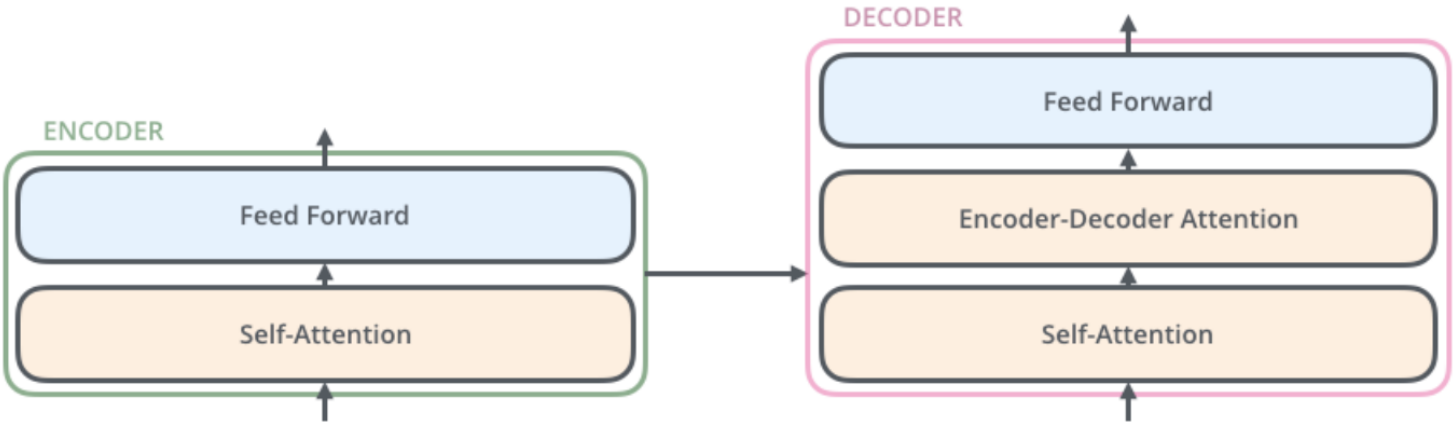
Transformer

Transformer Model



http://jalammar.github.io/illustrated-transformer/

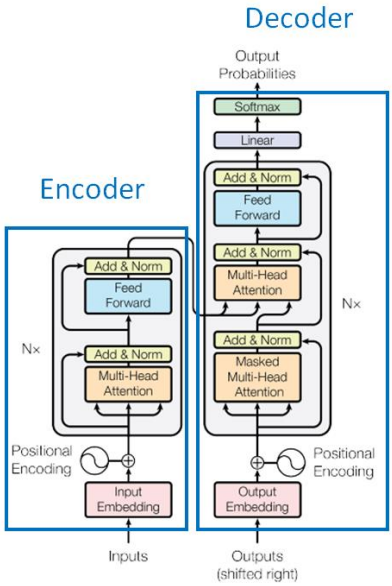
Transformer Model



http://jalammar.github.io/illustrated-transformer/

Transformer model

- Non-recurrent sequence to sequence encoder-decoder model
  - Eliminate recurrence, allows for significantly more parallelization
- Three kinds of attentions:
  - The input and output tokens (solved by traditional attention mechanism)
  - The input tokens themselves
  - The output tokens themselves
- Extend the (self-)attention mechanism to processing input and output sentences as well.



Language modeling

Slides are adopted from Advanced NLP course at UMass.

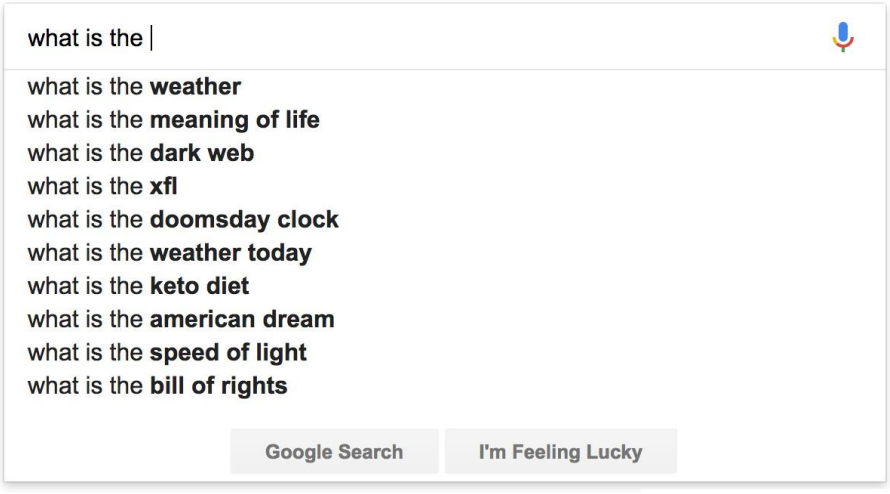
Intro

- Language models assign a probability to a piece of text.
- Why would we ever want to do this?
  - Translation:

    P(i flew to the movies) **<<<<<** P(i went to the movies)
  - Speech recognition:

    P(i saw a van) **>>>>>** P(eyes awe of an)

You use Language Models every day!



Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:
  - $P(W) = P(w_1, w_2, w_3, w_4, w_5, \dots, w_n)$
- Related task: probability of an upcoming word:
  - $P(w_5|w_1, w_2, w_3, w_4)$
- A model that computes either of these:
  - $P(W)$ or $P(wn|w_1, w_2, \dots, w_{n-1})$ is called a *language model* or LM

How to compute P(W)

- How to compute this joint probability:
  - P(its, water, is, so, transparent, that)
- Intuition: let's rely on the Chain Rule of Probability

Reminder: The Chain Rule

- Recall the definition of conditional probabilities

  $P(B|A) = P(A,B)/P(A)$ Rewriting: $P(A,B) = P(A)P(B|A)$

- More variables:

  $P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$

- The Chain Rule in General

  $P(x_1, x_2, x_3, \ldots, x_n)$
  $= P(x_1)P(x_2|x_1)P(x_3|x_1,x_2) \ldots P(xn|x_1, \ldots, xn\_1)$

Chain Rule for computing joint probability

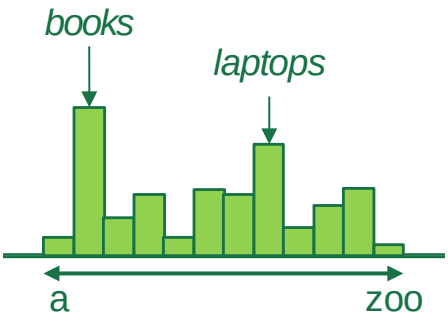
- The Chain Rule applied to compute joint probability of words in sentence

$$P(w_1 w_2 \ldots w_n) = \prod_i P(w_i \mid \overbrace{w_1 w_2 \ldots w_{i-1}}^{\text{prefix}})$$

P("its water is so transparent") =
P(its) × P(water|its) × P(is|its water)
× P(so|its water is) × P(transparent|its water is so)

Decoding from an LM

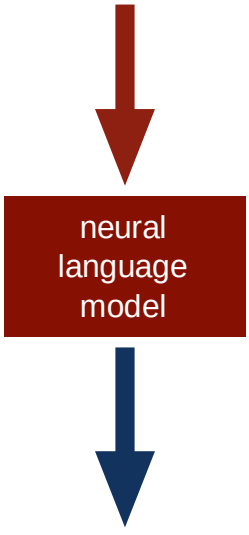
- **Prefix**: “students opened their”



Probability distribution over next word

Enter neural networks!

Students opened their

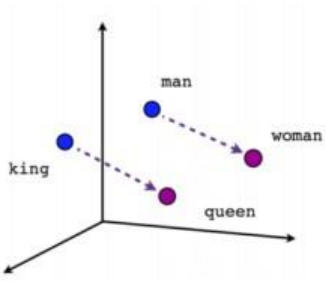


books

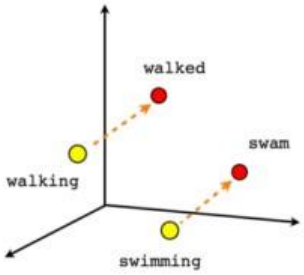
Words as basic building blocks

- Represent words with low-dimensional vectors called embeddings (Mikolov et al., NIPS 2013)

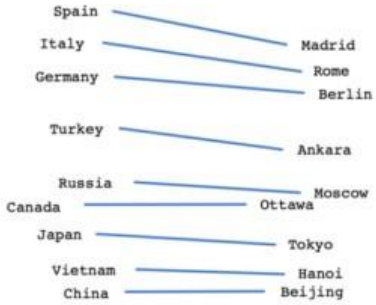
king = [0.23, 1.3, -0.3, 0.43]



Male-Female



Verb tense



Country-Capital

Composing embeddings

- Neural networks **compose** word embeddings into vectors for phrases, sentences, and documents.

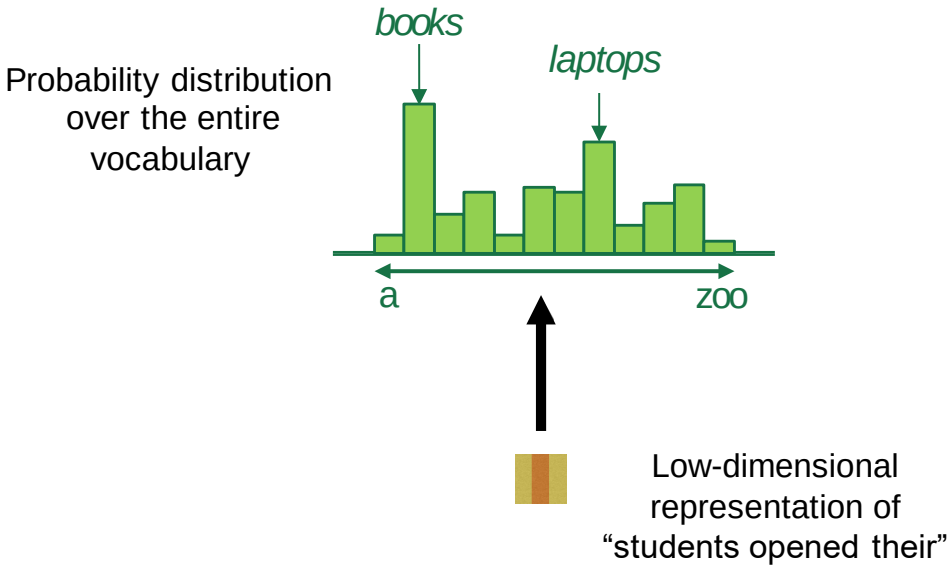
neural network (students opened their) = 

Next word prediction

- Predict the next word from composed prefix representation:

Softmax layer: convert a vector representation into a probability distribution over the entire vocabulary

$P(w_i|$ vector for "students opened their")



So to sum up…

- Given a d-dimensional vector representation x of a prefix, we do the following to predict the next word:
  - Project it to a V-dimensional vector using a matrix-vector product (a.k.a. a "linear layer", or a "feedforward layer"), where V is the size of the vocabulary.
  - Apply the softmax function to transform the resulting vector into a probability distribution.

Now that we know how to predict "books", let's focus on how to compute the prefix representation *x* in the first place!

Composition functions

- **Input:** sequence of word embeddings corresponding to the tokens of a given prefix
- **Output:** single vector
- Composition functions:
  - Element-wise functions
    - e.g., just sum up all of the word embeddings!
  - Concatenation
  - Feed-forward neural networks
  - Convolutional neural networks
  - Recurrent neural networks
  - Transformers

Large language models

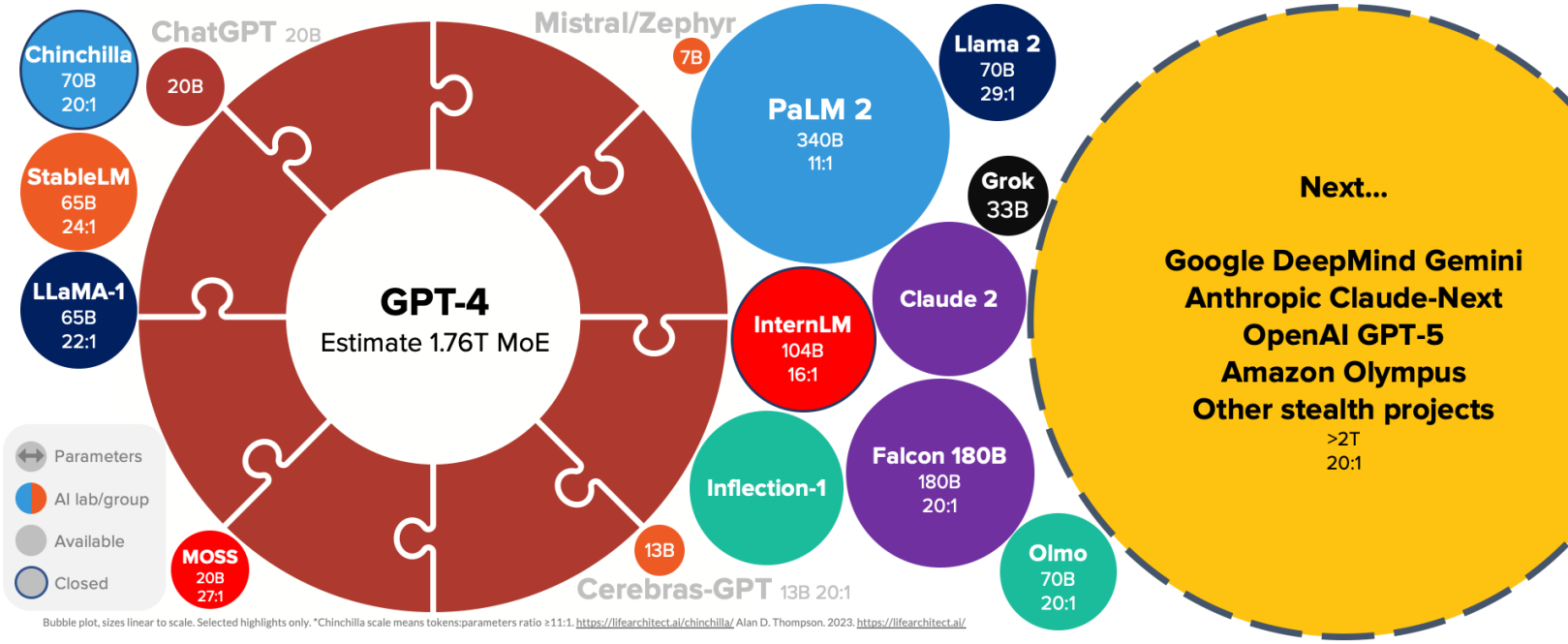
What's a large language model?

- ChatGPT, a chatbot built on top of the GPT LLM released on November 28, 2022, has popularized deep learning models trained with a text corpus.
- Language models learn a probability distribution over language:

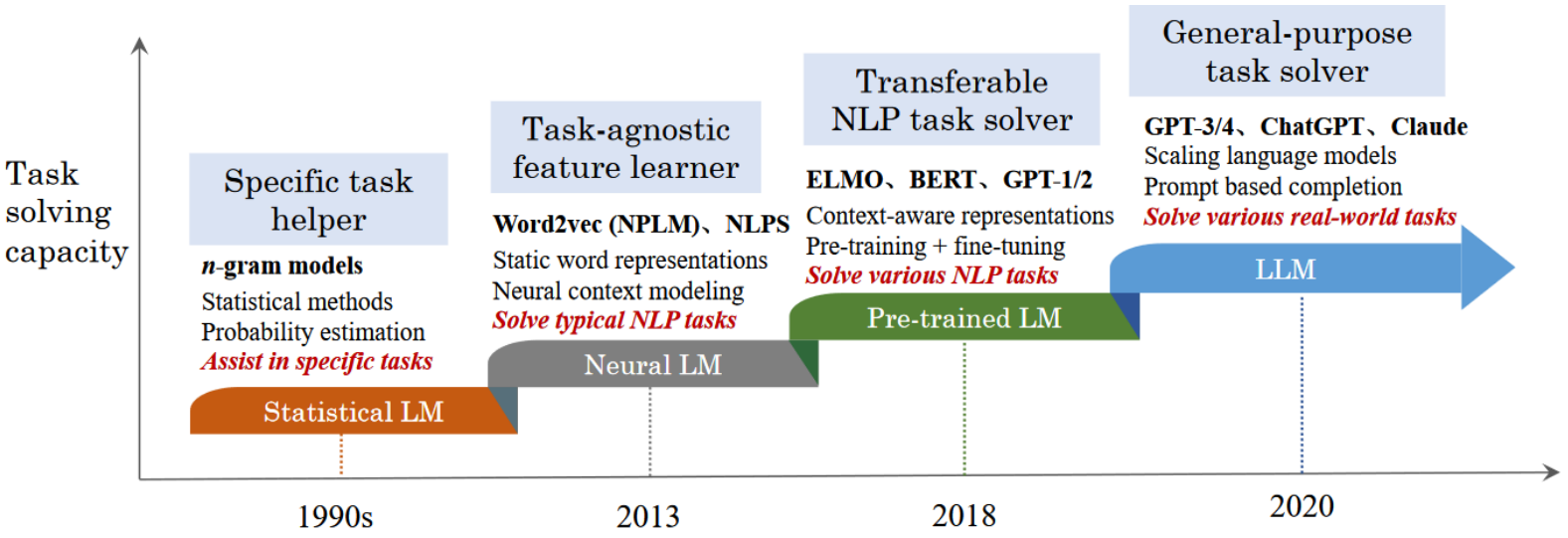
$$P(w_1, \ldots, w_m)$$

- Large in terms of training data (e.g., Common Crawl, Wikipedia, GitHub, …) and parameters (e.g., PaLM has 540 billion parameters; GPT-4 rumored to have 1 trillion).

Language Models (Up to November 2023)



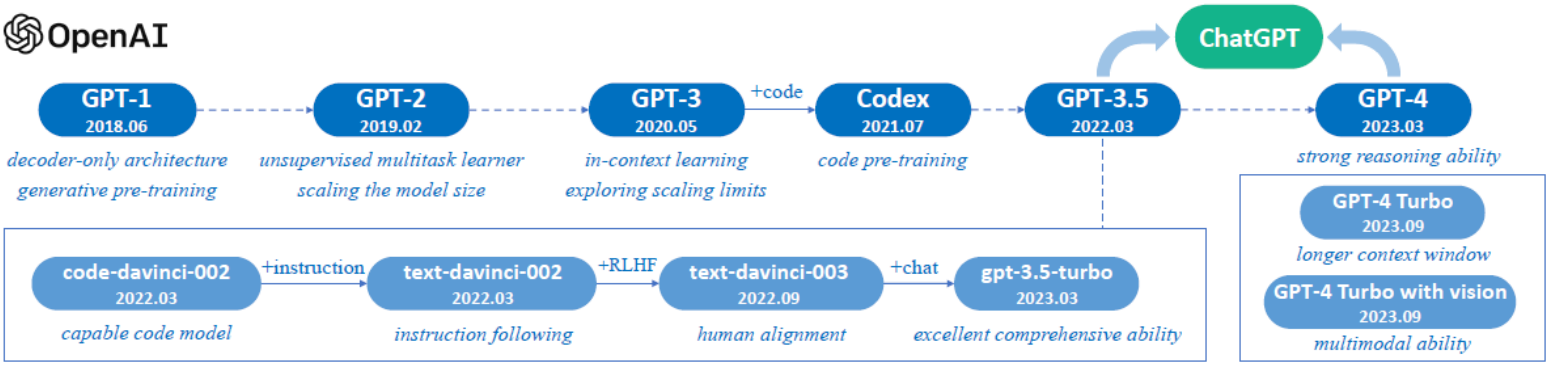
Source: https://lifearchitect.ai/models/



Three kinds of LLM

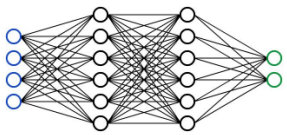
- **Generic language models:** predicting the next token. We will mostly talk about this type today.
- **Instruction tuned**
- **Dialog tuned:** ChatGPT (the base model is hard to interact with).

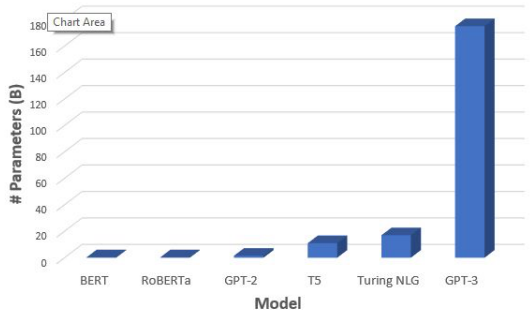
History of OpenAI GPT models



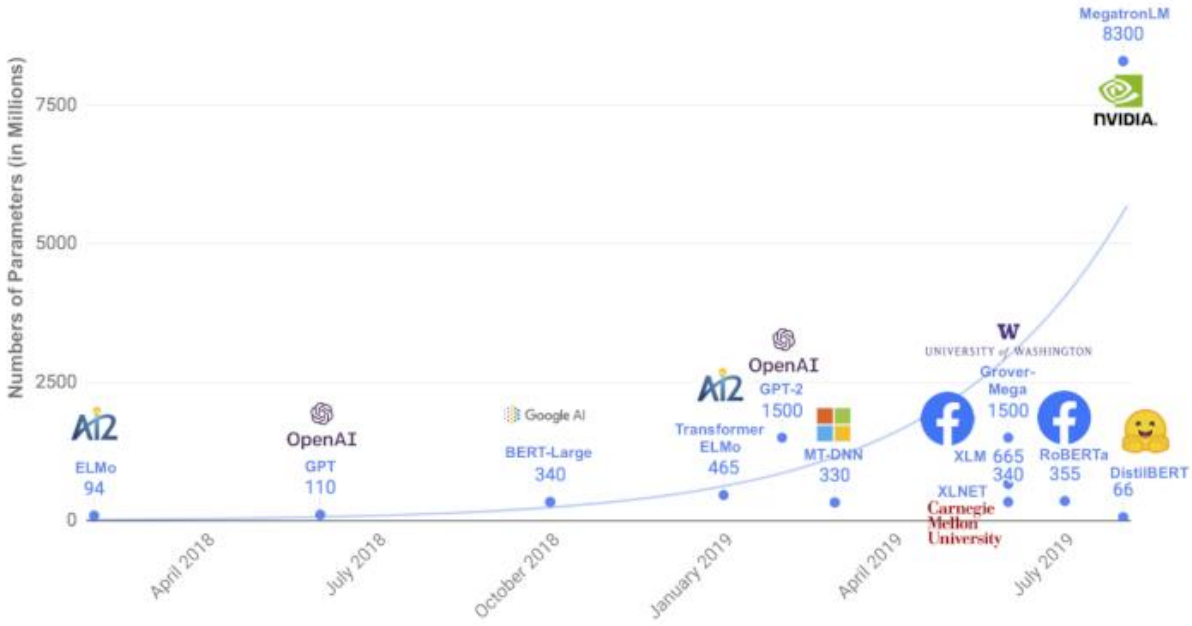
self-supervised learning + scale

In 1885, Stanford University was _____

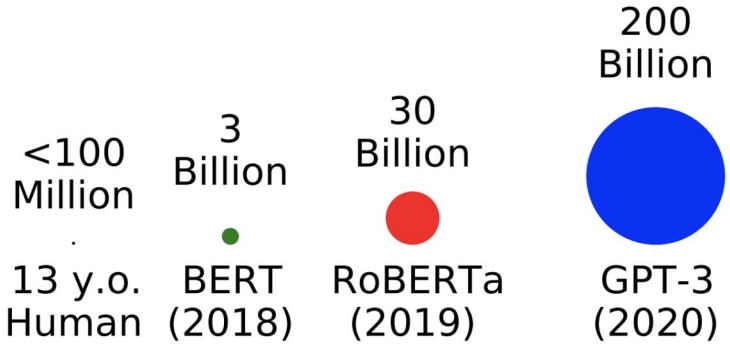




Scale in #parameters



Scale in #words

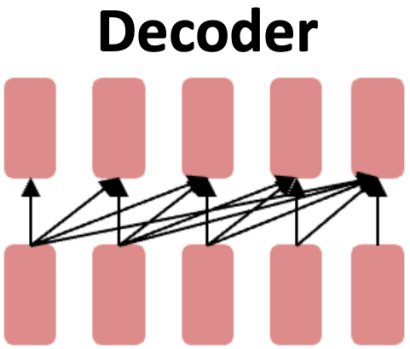


#tokens seen during training

GPTs: Generative Pre-trained Transformers

Large language models

GPT-2 (1.5B parameters; Radford et al., 2019)



Same architecture as GPT, just bigger (117M -> 1.5B)

But trained on much more data: 4GB -> 40GB

OpenAI

Emergent zero-shot learning

OpenAI

The ability to do many tasks with no examples, and no gradient updates

Language Models are Unsupervised Multitask Learners

Alec Radford [*1] Jeffrey Wu [*1] Rewon Child [1] David Luan [1] Dario Amodei [**1] Ilya Sutskever [**1]

Emergent abilities of GPT-3 (2020)

OpenAI

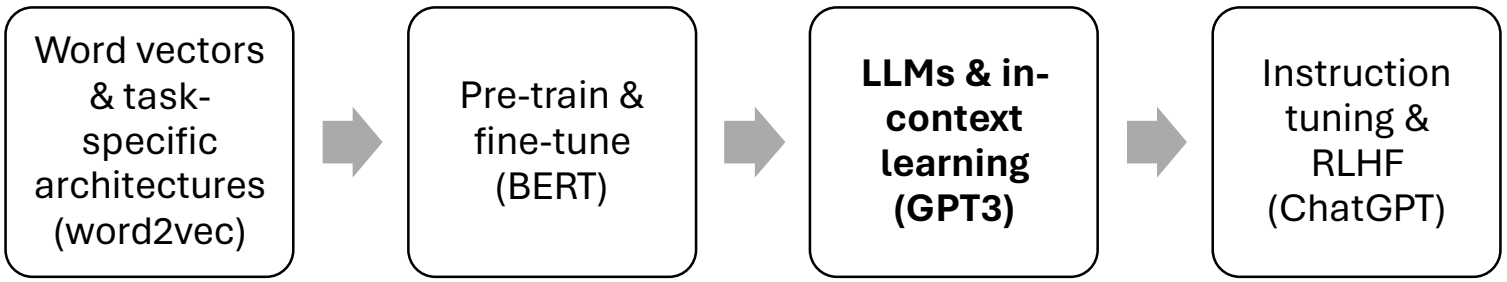
- GPT-3 (175B parameters; Brown et al., 2020)
  - Another increase in size (1.5B -> 175B)
  - and data (40GB -> over 600GB)

Language Models are Few-Shot Learners

Tom B. Brown* Benjamin Mann* Nick Ryder* Melanie Subbiah*

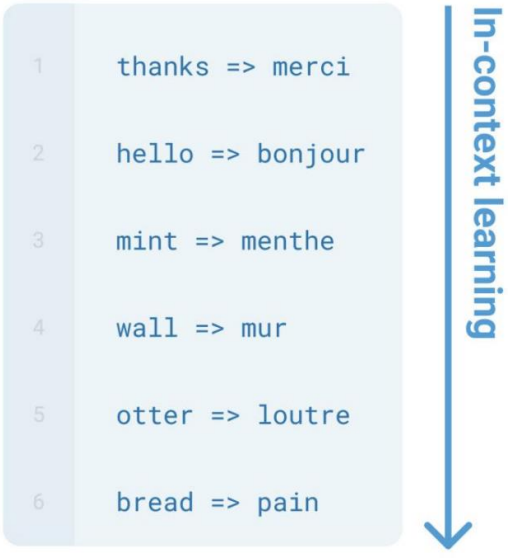
In-context Learning

Recent history



Emergent in-context learning

OpenAI



Few-shot GPT3 can beat strong task-specific fine-tuned models.

Language modeling ≠ assisting users

PROMPT	*Explain the moon landing to a 6 year old in a few sentences.*
COMPLETION	GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

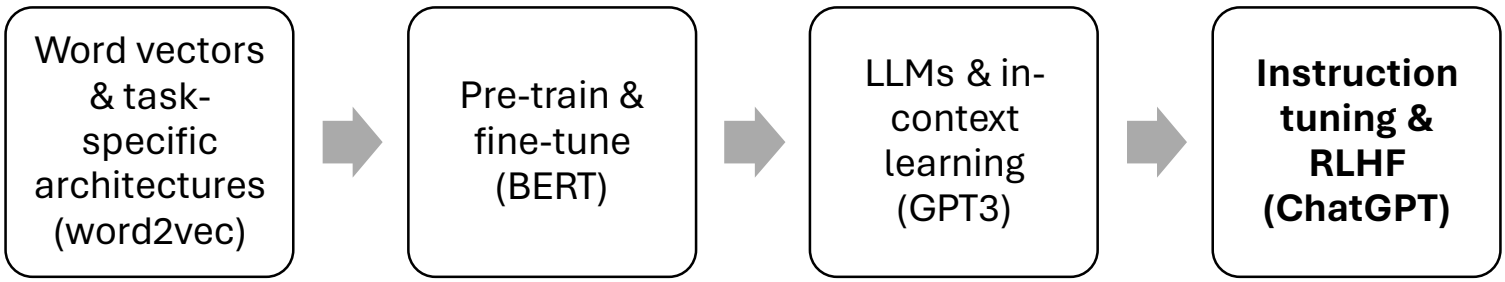
Language modeling ≠ assisting users

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION **Human**
A giant rocket ship blasted off from Earth carrying astronauts to the moon. The astronauts landed their spaceship on the moon and walked around exploring the lunar surface. Then they returned safely back to Earth, bringing home moon rocks to show everyone.

Instruction Tuning

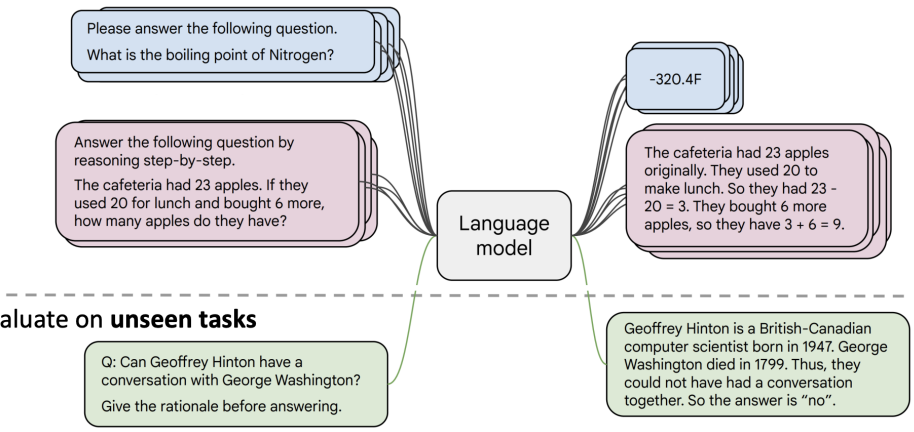
Recent history



Instruction tuning

Collect examples of (instruction, output) pairs across many tasks and finetune an LLM

- **Collect examples** of (instruction, output) pairs across many tasks and finetune an LM



- Evaluate on **unseen tasks**

[FLAN-T5; Chung et al., 2022]

Limitations of instruction-tuning

- No right answer for tasks like open-ended creative generation
- Mismatch between the LM objective and the objective of "satisfy human preferences"!

Can we explicitly attempt to **satisfy human preferences**?

Reinforcement learning to the rescue

For each LM sample s, imagine we had a way to obtain a human reward of that summary: $R(s) \in \mathbb{R}$, higher is better.

SAN FRANCISCO, California (CNN) -- A magnitude 4.2 earthquake shook the San Francisco ... overturn unstable objects.

An earthquake hit San Francisco. There was minor property damage, but no injuries.

s_1

$R(s_1) = 8.0$

The Bay Area has good weather but is prone to earthquakes and wildfires.

s_2

$R(s_2) = 1.2$

We want to maximize the expected reward of samples from our LM

Optimizing for human preferences

$$\mathbb{E}_{\hat{s} \sim p_\theta(s)}[R(\hat{s})]$$

How do we model human preferences?

OpenAI

Model their preferences as a separate (NLP) problem!

An earthquake hit San Francisco. There was minor property damage, but no injuries.

s_1

$R(s_1) = 8.0$

The Bay Area has good weather but is prone to earthquakes and wildfires.

s_2

$R(s_2) = 1.2$

Train an LM $RM_\phi(s)$ to predict human preferences from an annotated dataset, then optimize for RM_ϕ instead.

How do we model human preferences?

A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage.

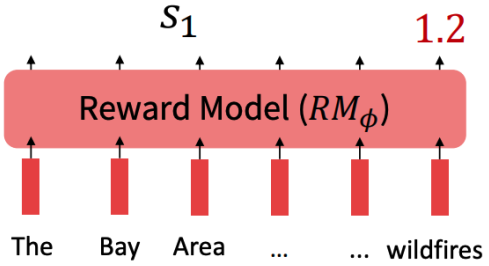
s_3

$R(s_3) =$ 4.1? 6.6? 3.2?

How do we model human preferences?

Asking pairwise comparisons

An earthquake hit San Francisco. There was minor property damage, but no injuries. > A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage. > The Bay Area has good weather but is prone to earthquakes and wildfires.



s_3 s_2

Bradley-Terry [1952] paired comparison model

$$J_{RM}(\phi) = -\mathbb{E}_{(s^w, s^l)\sim D}\left[\log \sigma(RM_\phi(s^w) - RM_\phi(s^l))\right]$$

"winning" sample — "losing" sample — s^w should score higher than s^l

RLHF: Putting it all together [Christiano et al., 2017; Stiennon et al., 2020]

From GPT3 to InstructGPT

OpenAI

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION

GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

InstructGPT

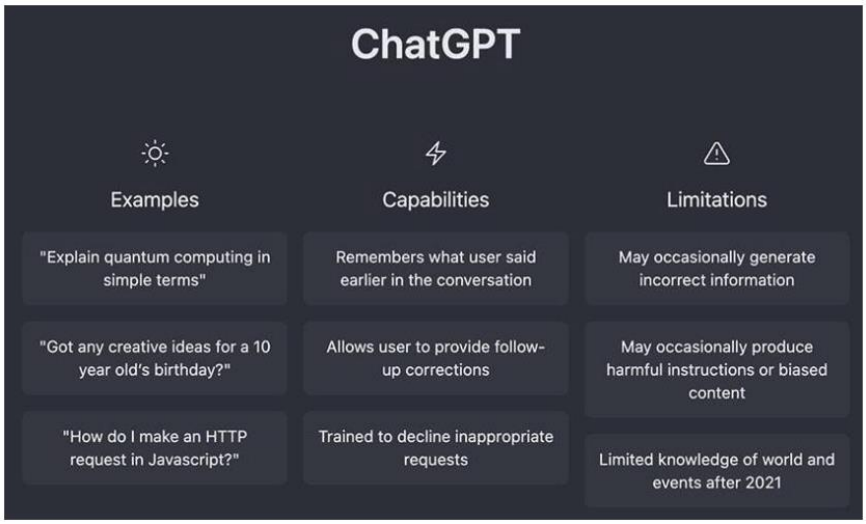
```
People went to the moon, and they took pictures of what they saw, and
sent them back to the earth so we could all see them.
```


ChatGPT: Instruction finetuning + RLHF for **dialog** agents

From:

Tehran is located in _____

To:



Practical Applications

Use in Education

Industry Applications

Accessibility and Language Support

Limitations of RL + Reward Modeling

Human preferences are unreliable!

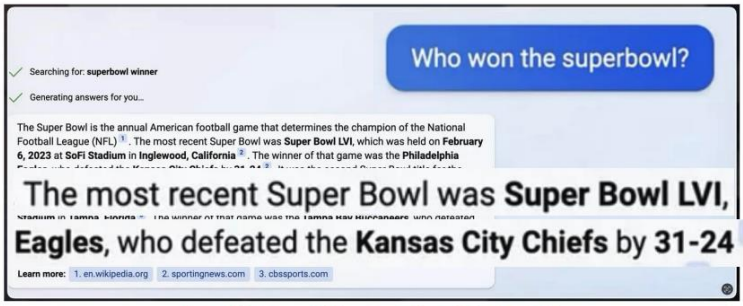
"**Reward hacking**" is a common problem in RL

Google shares drop $100 billion after its new AI chatbot makes a mistake

February 9, 2023 · 10:15 AM ET

https://www.npr.org/2023/02/09/1155650909/google-chatbot--error-bard-shares

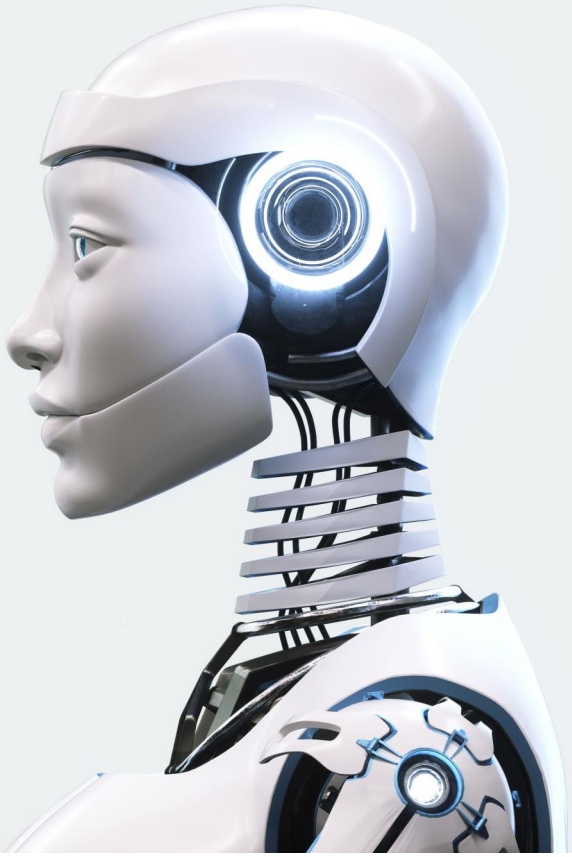
Bing AI hallucinates the Super Bowl

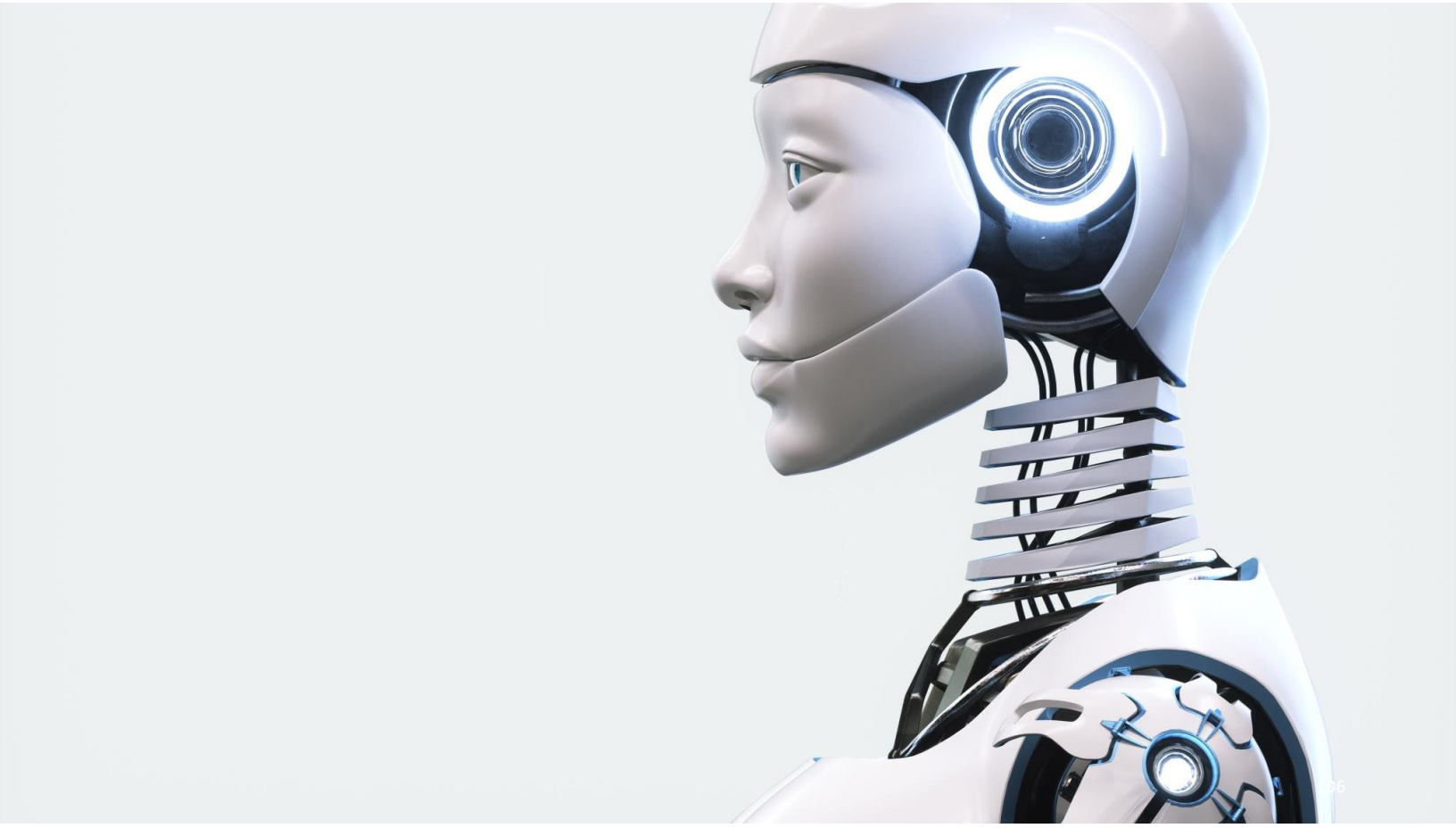


https://news.ycombinator.com/item?id=34776508
https://apnews.com/article/kansas-city-chiefs-philadelphia-eagles-technology-science-82bc20f207e3e4cf81abc6a5d9e6b23a

Chatbots are rewarded to produce responses that seem **authoritative** and **helpful**, regardless of truth

Models of human preferences are even more unreliable!





ChatGPT hallucinates

Prompt	Where do people smoke more on average than in Russia?
ChatGPT Generation	There are several countries where people smoke more on average than in Russia. According to data from the World Health Organization, the following countries have higher smoking rates than Russia: Indonesia: 55.2% of adults smoke China: 52.9% of adults smoke Serbia: 47.5% of adults smoke Serbia and Montenegro: 45.6% of adults smoke Montenegro: 45.6% of adults smoke Greece: 44.5% of adults smoke Bosnia and Herzegovina: 41.8% of adults smoke Kazakhstan: 41.3% of adults smoke Kosovo: 41.3% of adults smoke Moldova: 40.9% of adults smoke It's important to note that these figures are estimates and may vary slightly from other sources.
Hallucination	Extrinsic - the generated answer includes wrong statistics. (Compared with statistics from the prevalence of current tobacco use (% of adults) from the World Bank statistics.[16])

Numbers marked in red are wrong, ChatGPT imagines them.

[Bang et al., 2023:https://arxiv.org/pdf/2302.04023.pdf]

ChatGPT has a timestamp

AS Who won the most recent FIFA World Cup?

The most recent FIFA World Cup was held in 2018, and the winner was the French national team. They defeated Croatia 4-2 in the final to win their second World Cup title. The tournament was held in Russia from June 14 to July 15, 2018, and featured 32 teams from around the world.

The answer should have been Argentina, but it is not always trained on the most recent data.

Recent history