Large language models

an introduction

Benoît Crabbé

June 2024

Laboratoire de linguistique formelle

Representing the world with vectors

Classification for botanists

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
103	7.10	3.00	5.90	2.10	virginica
43	4.40	3.20	1.30	0.20	setosa
71	5.90	3.20	4.80	1.80	versicolor
136	7.70	3.00	6.10	2.30	virginica
35	4.90	3.10	1.50	0.20	setosa
64	6.10	2.90	4.70	1.40	versicolor

Iris Versicolor **Iris Setosa** **Iris Virginica**

Iris classification

- In the traditional case, data for statistical analysis comes from measures. These measures are organised as input vectors $\mathrm{x} \in \mathbb{R}^d$ that are used as input to a predictive model with some output $\mathrm{y} \in \mathbb{R}^p$. ($d = 4, p = 3$)
- For the iris, a classification model can be as simple as:

$$\mathrm{y} = \text{softmax}(\mathrm{W}\mathrm{x} + \mathrm{b})$$

where W is a matrix of parameters and b a bias vector

Representing the world with vectors

Classification for text

- Consider the sentiment classification problem for text where $\mathrm{y} \in \{pos, neg, other\}$. Your input data x cannot be straightforwardly measured anymore :
  - I dislike old cabin cruisers / negative
  - Bertram has a deep V hull and runs easily through seas / positive
  - I do not dislike cabin cruisers / positive

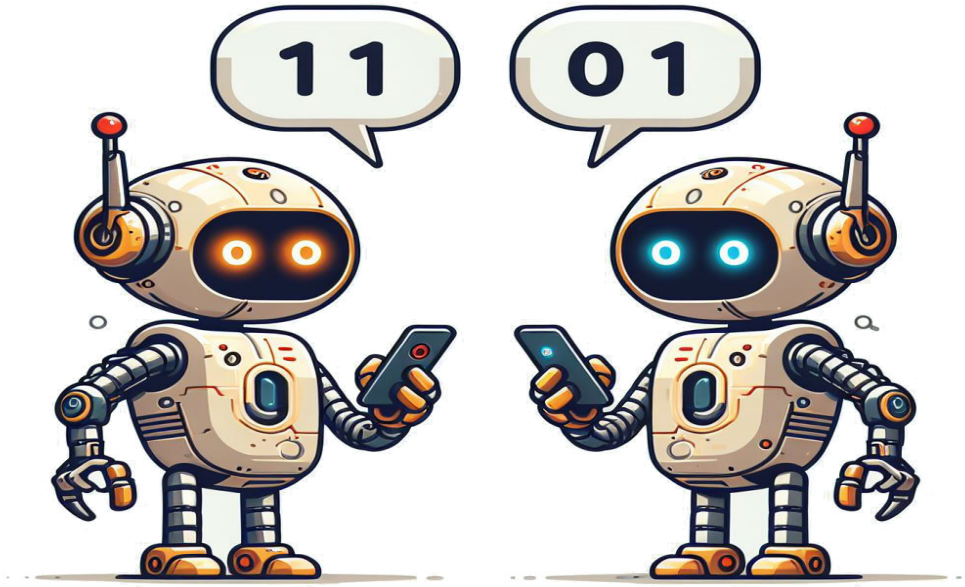
Representation function

When processing text, a key element of the modeling amounts to choose an appropriate encoding of words (symbols) into vectors. Several solutions can be considered:

- Dummy coding, one hot coding
- Word embeddings
- Neural language models

A **Large Language Model** is a (sophisticated) representation function mapping sequences of symbols to vectors

Outline



Vectors and vector space

- Vectors are arrays of (real) numbers. They are written as x and y.
- Vectors of the same size can be added together ($\mathrm{x} + \mathrm{y}$):

$$\begin{bmatrix} 2 \\ -3 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \\ -2.5 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ -2 \end{bmatrix}$$

- Vectors can be multiplied by a scalar $a \in \mathbb{R}$:

$$3 \begin{bmatrix} 12 \\ -3 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 36 \\ -9 \\ 1.5 \end{bmatrix}$$

- We picture vectors sometimes as :

Dot product

- The dot product of two vectors w and x is defined as:

$$\langle \mathrm{w}, \mathrm{x} \rangle = w_1 x_1 + \ldots + w_n x_n$$

The dot product and linear models

$$\mathrm{w} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b \end{bmatrix} \quad \mathrm{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{bmatrix}$$

Observe that: $\langle \mathrm{w}, \mathrm{x} \rangle = a_1 x_1 + a_2 x_2 + a_3 x_3 + b1$

Matrices as linear maps

- Matrices are rectangular arrays of numbers, written as $\mathrm{W}, \mathrm{U}, \mathrm{V}$.
- Matrices can be multiplied by vectors. A matrix $\mathrm{W} \in \mathbb{R}^{m \times n}$ multiplies a vector x of size *n* and yields a vector of size *m*. Example:

$$\mathrm{W} = \begin{bmatrix} 12 & 0.2 \\ -3 & 1 \\ 0.5 & 0 \end{bmatrix} \quad \mathrm{x} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\mathrm{Wx} = \begin{bmatrix} \langle \mathrm{w}_1, \mathrm{x} \rangle \\ \langle \mathrm{w}_2, \mathrm{x} \rangle \\ \langle \mathrm{w}_3, \mathrm{x} \rangle \end{bmatrix} = \begin{bmatrix} 11.8 \\ -4 \\ 0.5 \end{bmatrix}$$

Linear maps: it is important to see that matrices transform their input vector of some size *n* to an output vector of size *m* by means of linear operations

- Matrices can also be multiplied together too

Non linear vector transformations

Neural networks use non linear operators to transform vectors called activations. Here are some classical ones:

- *tanh* : $\mathbb{R}^d \mapsto [-1, 1]^d$. Each vector coordinate x_i is mapped to a value in $[-1, 1]$

$$tanh(x_i) = \frac{e^{x_i} - e^{-x_i}}{e^{x_i} + e^{-x_i}}$$

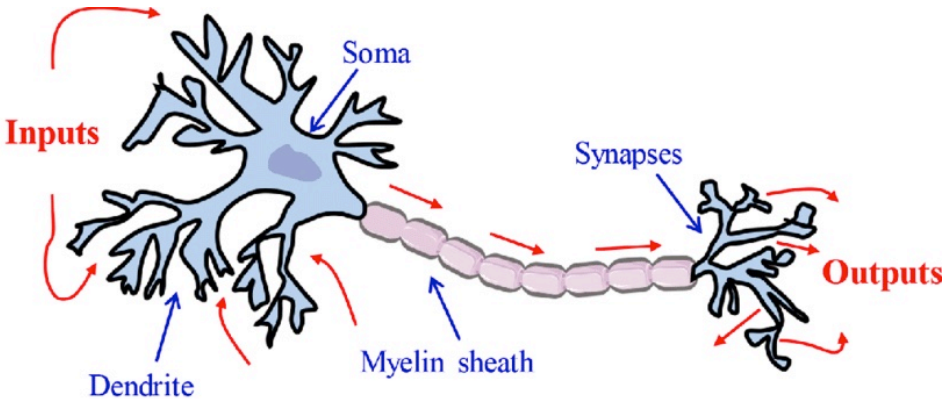
- σ : $\mathbb{R}^d \mapsto [0, 1]^d$. Each vector coordinate x_i is mapped to a value in $[0, 1]$

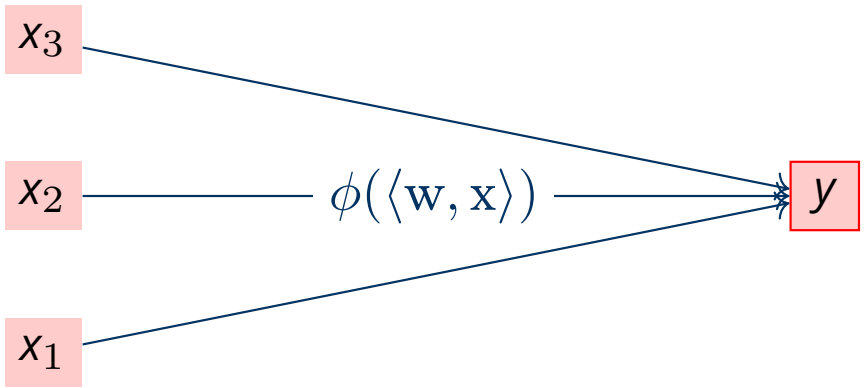
$$\sigma(x_i) = \frac{1}{1 + e^{-x_i}}$$

- *softmax* : $\mathbb{R}^d \mapsto [0, 1]^d$ with the constraint that $\sum_i x_i = 1$. That is softmax converts a vector of real numbers into a vector of probabilities.

$$softmax(x_i) = \frac{exp(x_i)}{\sum_j exp(x_j)}$$

Relation to biology





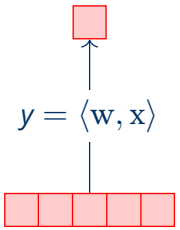
The elementary neuronal model outputs the result of the dot product of the input signals with the neuron parameters transformed by an activation

Two versions of neural networks

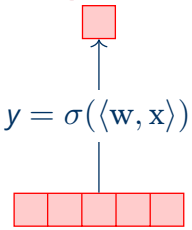
- Some neural networks are designed to be convenient to use for mathematical and computational manipulation. These are those most people use nowadays
- Some other neural networks are designed to model more explicitly biological networks, these are the **spiking networks**. In a spiking network a neuron fires only when it reaches a threshold (the potential)

Some elementary neural network architectures

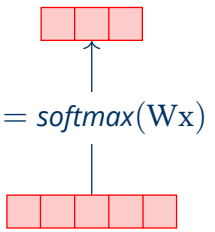
Linear regression



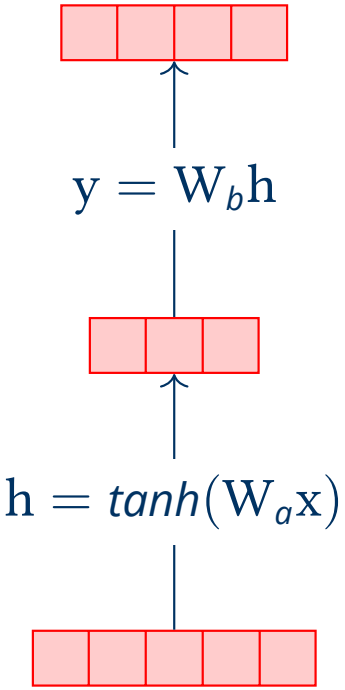
Logistic regression



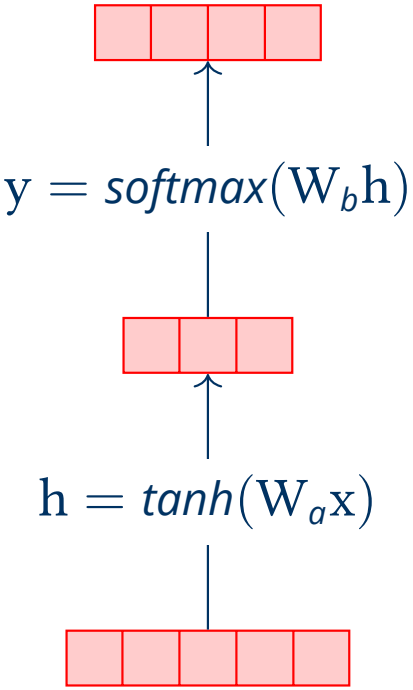
Softmax regression



Feedforward network



Multilayer softmax regression



Training a neural network

Training a neural network amounts to estimate the value of its parameters w from a data set.

- Training a neural network supposes a dataset of the form $D = \left(y_i, \mathrm{x}_i\right)_{i=1}^{n}$
- Training a neural network supposes a loss function. A loss function compares the value predicted by the model $\hat{y}_i$ with the data value y_i for each example of the data set.
- Training a neural network aims to **minimize the loss**: at each time step, the parameters are updated in such a way that the loss decreases.

Example loss

Sum of squares is the loss for linear regression. It measures the distance between the data points and the line:

$$ssq(D, \mathrm{w}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \langle \mathrm{w}, \mathrm{x}_i \rangle)^2$$

The cross entropy loss

The most common loss used in NLP is the cross entropy loss.

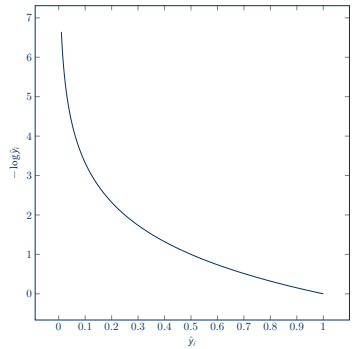
- The cross entropy loss measures the distance between a one hot vector and network probabilistic output

0	1	0	0	0

$$y_i \in \{0, 1\}$$
$$\hat{y}_i \in [0, 1]$$

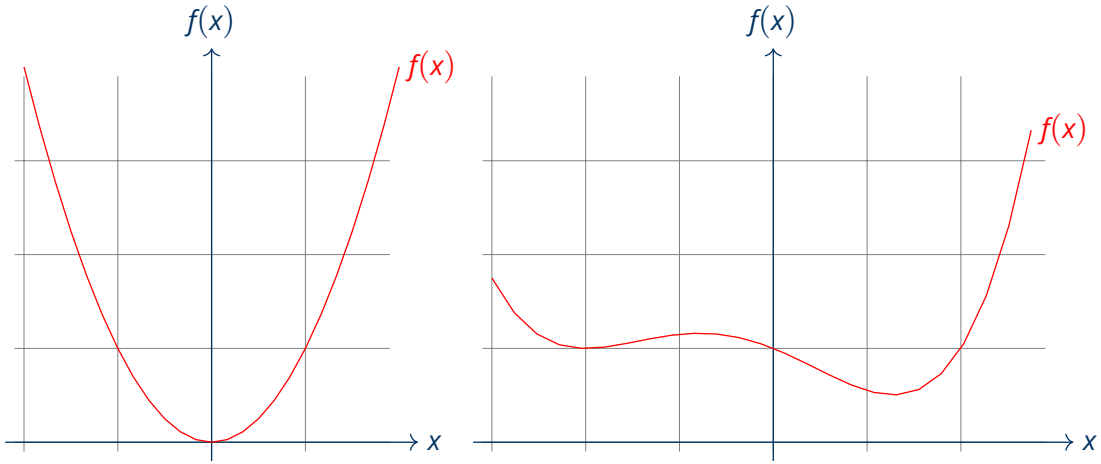
$$H(\mathrm{y}, \hat{\mathrm{y}}) = -\sum_{i=1}^{d} y_i \log \hat{y}_i$$

- The loss is essentially dependent of $-\log(\hat{y}_i)$ for the correct label:



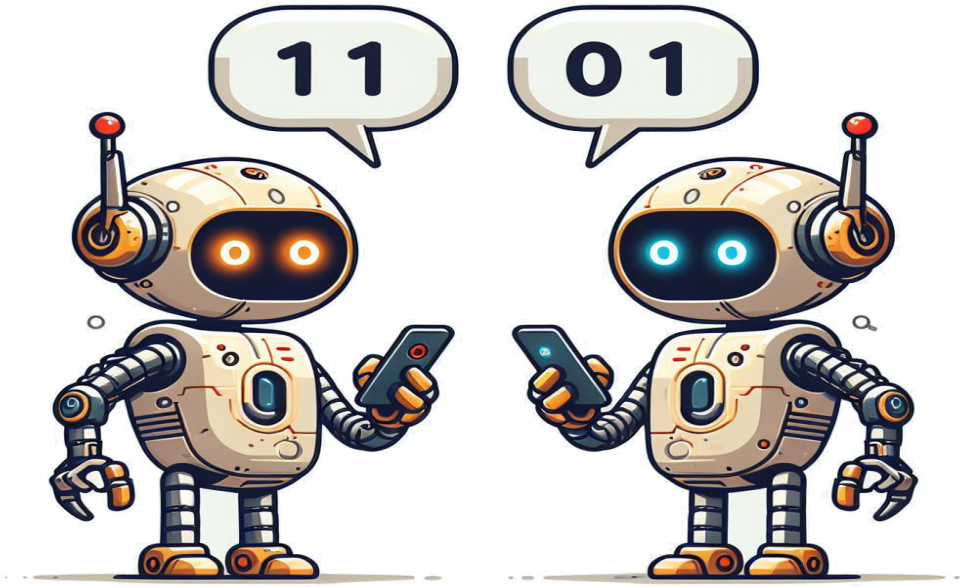
Non convexity of neural networks

- The theory of machine learning is built on the assumption that the loss is a (strictly) **convex** function: the minimum of the function is unique. In other words there is exactly one vector w for which the loss is minimal.
- In deep learning this is not true anymore. There may be several local minima. Most neural networks are deep and use several layers with non linear activations. As a result losses are generally **non convex**



- **As a result, the parameters may change from one fit to another due to randomness. This is generally nonsense to interpret the coefficients of the model.**

Outline



One hot vectors and the distributional hypothesis

- One hot vector encoding is a classical method for encoding a set C of categorical values.
  Example:

setosa	100
virginica	010
versicolor	001

Distributional hypothesis

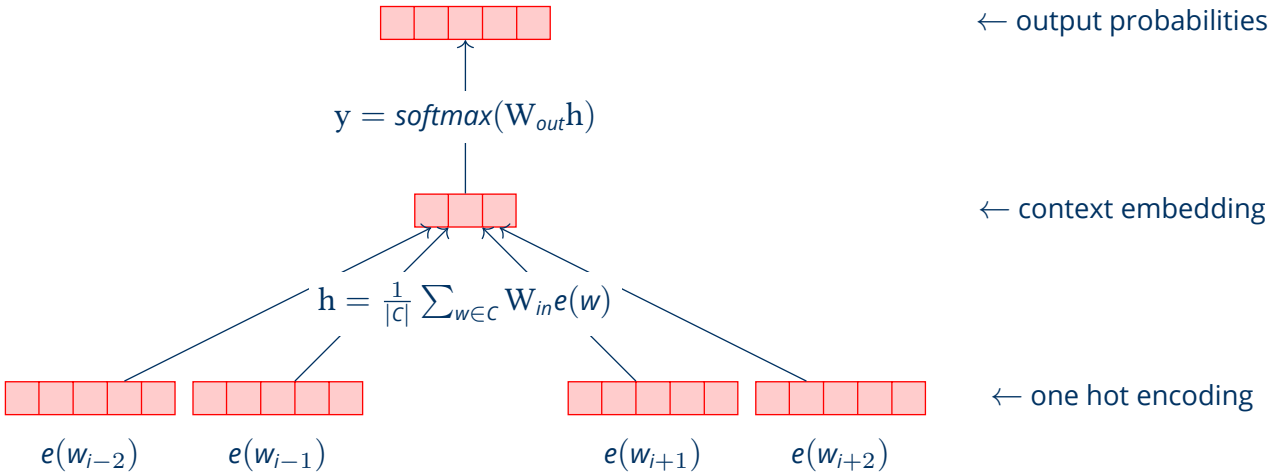
One hot vectors are sometimes fine, but for natural language it turns out that words appearing in the same textual context tend to have the same meaning.

- For instance the words ”apple” or ”pear” are more likely to be surrounded by words like ”sweet”, ”green”, ”tasty”, ”sour” rather than by ”cruel”, ”arrogant” or ”angry” (context of ”monster” or ”villain”)
- **Word embeddings** are vectors encoding words such that words occuring in the same textual contexts have similar representation vectors. Methods for computing word embeddings generate a static dictionary mapping strings to vectors

Word embeddings: an example

Word2vec (Mikolov 2013)

- There is a whole family of word embedding methods, here is one called Word2Vec (CBOW)
- The model is trained by minimizing the cross entropy of a dataset made of contexts with a missing central word.



- Once trained, the model matrix W_{in} is used as a dictionary to look up for word representations $r(w)$:

$$\mathrm{r}(w) = \mathrm{W}_{in} e(w)$$

Word embeddings and bag of words

- Word embeddings generated by word2vec are used to map word strings to word vectors by dictionary lookup.
- Word embeddings can be used to map sequence of words (sentences, paragraphs, texts) assuming that:
  - Word sense as given by the vector representation is independent of the context
  - word order is irrelevant for the problem at stake (bag of words assumption)

  Averaging is the most common method for a text $T = w_1 \ldots w_n$:

$$r(T) = \frac{1}{n} \sum_{i=1}^{n} r(w_i)$$

Limits of the bag of words assumption

- The bag of word and context independence assumptions are sometimes too strong
  - They always wanted to become Hollywood stars
  - The Milky Way contains between 200 and 400 billion stars
- These problems are tackled by language models

Language models

- The probability of a sequence of symbols $\mathrm{x} = x_1 \ldots x_n$ is given by the **chain rule** of probability:

$$P(\mathrm{x}) = P(x_1, \ldots x_n) = \prod_{i=1}^{n} P(x_i | x_1 \ldots x_{i-1})$$

  The conditional $P(x_i | x_1 \ldots x_{i-1})$ reads as "the probability of **generating** x_i given that we know $x_1 \ldots x_{i-1}$". The longer the context the easier it is to guess x_i, example:
  - the ...
  - the cat ...
  - the cat chases the ...

- A **language model** is a distribution of probability over a set $\mathcal{L}$ of variable length sequences, a formal language, such that:

$$0 \leq P(\mathrm{x}) \leq 1$$

$$\sum_{x \in \mathcal{L}} P(x) = 1$$

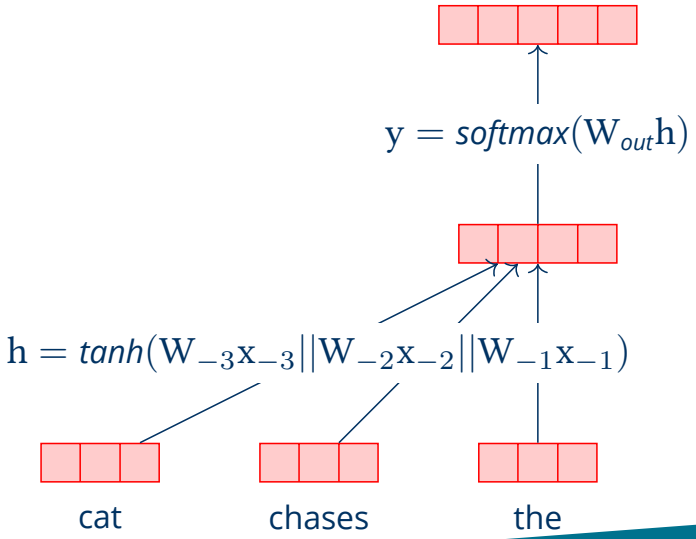
Markovian neural language model

- A **markov assumption** is a classical assumption that sets the boundary of the context and simplifies the chain rule:

$$P(\mathrm{x}) \approx \prod_{i=1}^{n} P(x_i | x_{i-k} \dots x_{i-1})$$

  The conditionals now have a limited context of order k.

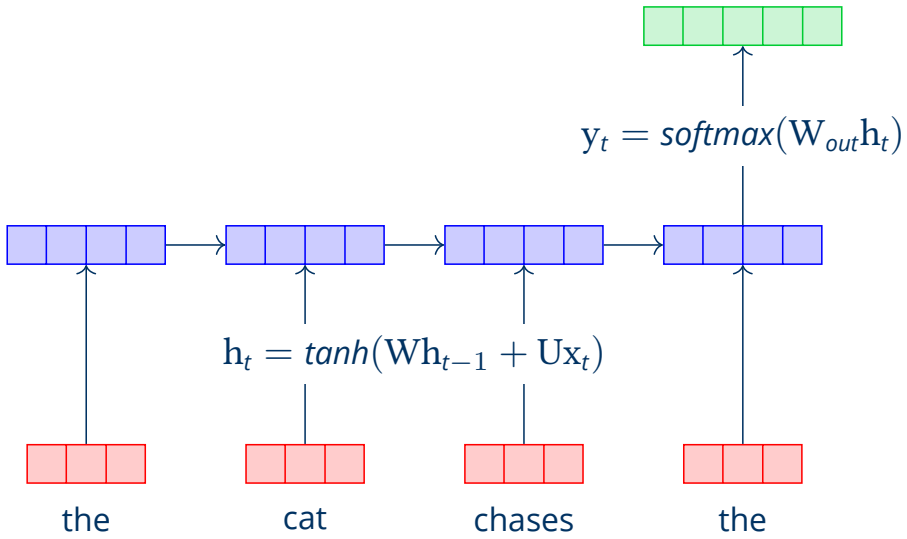
- A **markovian neural language model** (Bengio et al. 2003) uses a window of k words to predict the next word. Example for $k = 3$



Recurrent neural network Language models

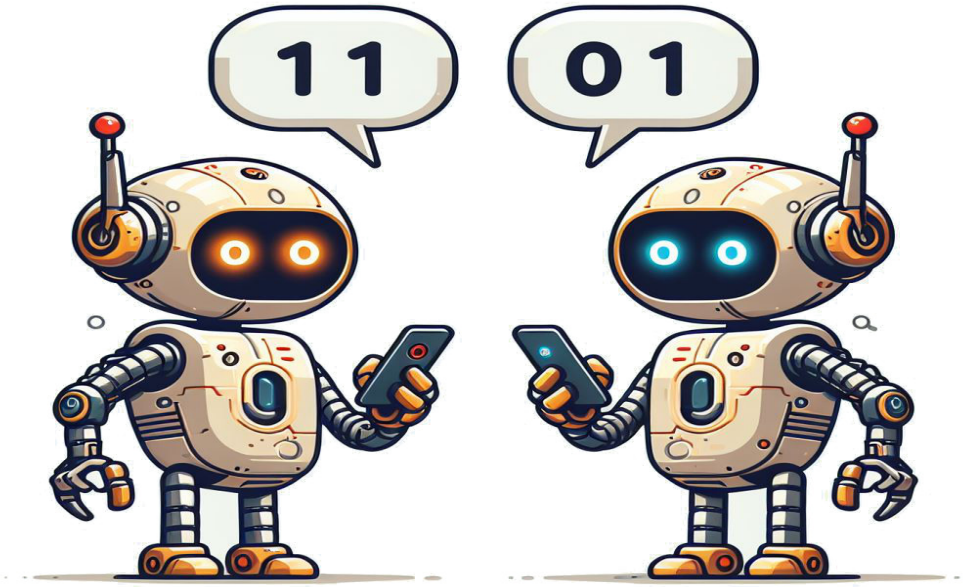
(Elman 91, Hochreiter, Schmidhuber 1997, Mikolov 2010)

- The recurrent language model uses a recurrent neural network to compute transition probabilities without markov assumptions: $P(\mathrm{x}) = \prod_{i=1}^{n} P(x_i | x_1 \ldots x_{i-1})$
- A recurrent neural network (Rnn/Lstm) is a *time dependant* model used to compute incrementally contextualized word embeddings h_t:



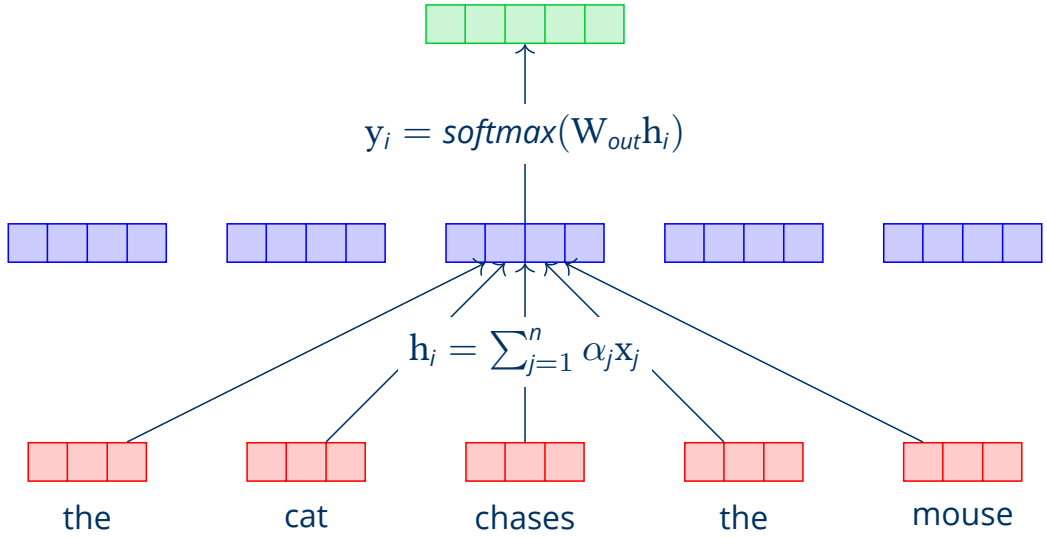
these contextualized word embeddings (in blue) are used to predict probabilities of the following word (y_t, in green) with an output softmax over the vocabulary.

Outline



Attention! recurrent networks forget the past!

- As observed in machine translation (Bahdanau 2014), RNNs (and LSTM) forget the distant past.
- **Attention models** are *time independant*, or *parallel*, models that allow to contextualize bag of words embeddings. Example of a **self attention** model:

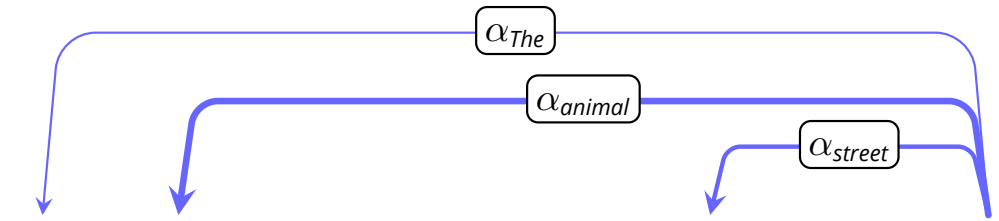


- The attention weights (α_j) are probabilities indicating how similar are the word embeddings in the sentence with the target word:

$$\alpha_j = \frac{exp(a_j)}{\sum_k exp(a_k)} \qquad a_j \propto \langle x_i, x_j \rangle \qquad (1 \leq j \leq n)$$

There is more than one way to pay attention to your neighborhood!

- Self attention says how we update the embedding of a word given its important context. Consider the α_j distribution for word *it* in the example:

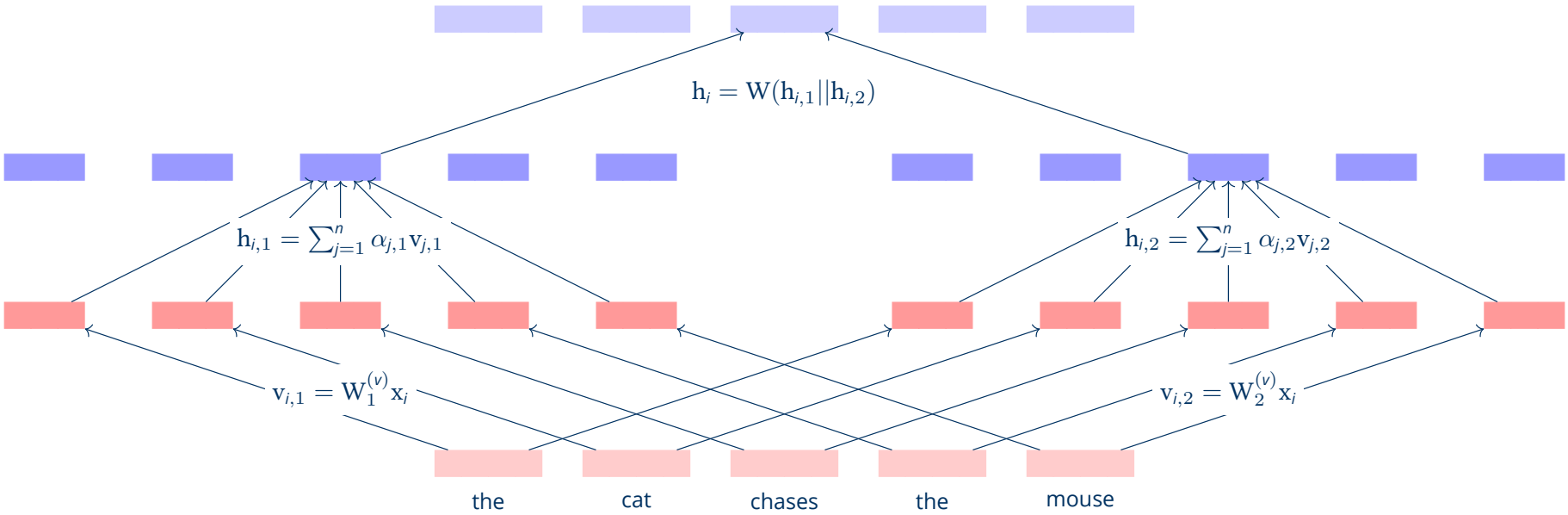


  Then self-attention updates the embedding of *it* to be the weighted sum of its potential antecedents.

- But there are other ways to pay attention to the context:
  - **(SV-agreement)** The keys to the cabinet **are** on the table.
  - **(anaphora)** The animal didn't cross the street because **it** was too tired
  - **(lex-semantics)** I walked along the pond, and noticed that one of the trees along the bank had fallen into the water after the storm.

Multi-head attention

- Instead of using a single attention, a transformer uses *h* heads of self attention
- Computes updated embeddings for each head and remerges back the results by pooling. Example for two heads:

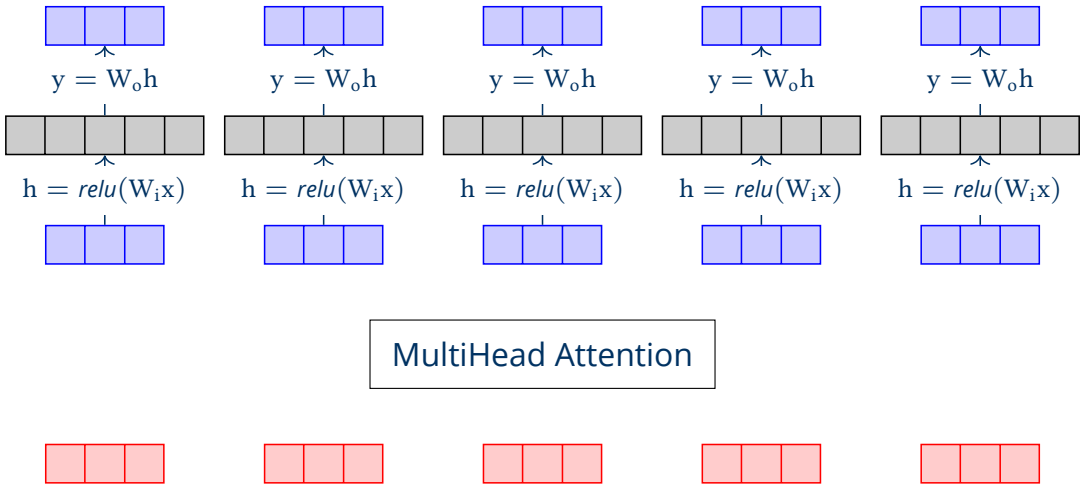


- $\alpha_{j,h} = \frac{exp(a_{j,h})}{\sum_k exp(a_{k,h})} \quad a_{j,h} \propto \langle q_{i,h}, k_{j,h} \rangle$
- $q_{i,h} = W_h^{(q)} x_i \quad k_{i,h} = W_h^{(k)} x_i$

Transformer layer

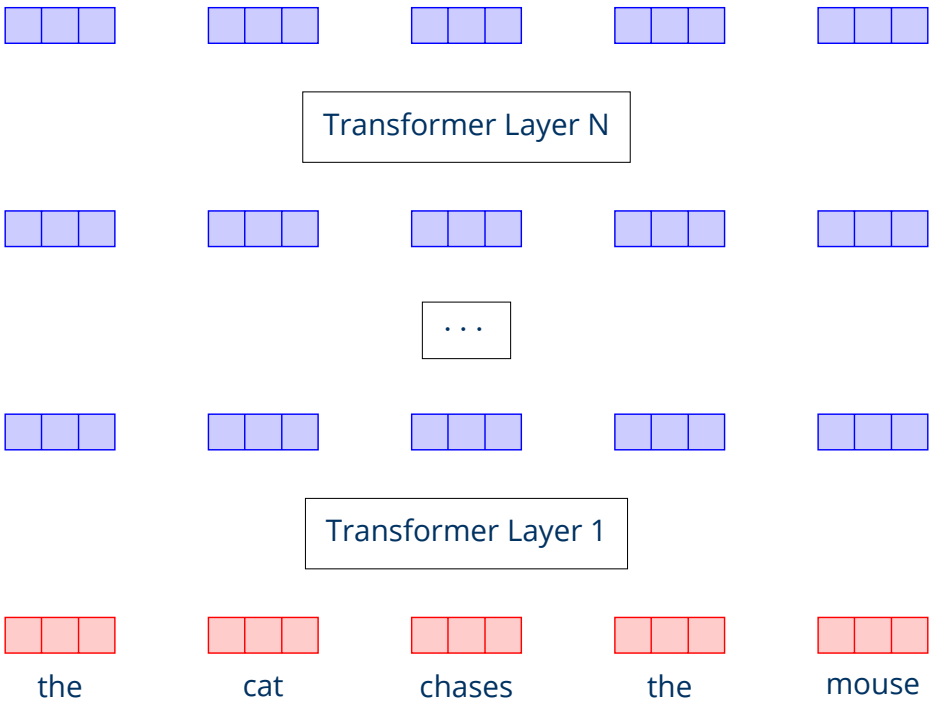
- The transformer layer is the basic unit of transformation. It is made of:
  - A multihead attention component that ”mixes” embeddings
  - A feedforward component that ”refines” embeddings

  and a formal apparatus to normalize and prevent drifting (hidden here).

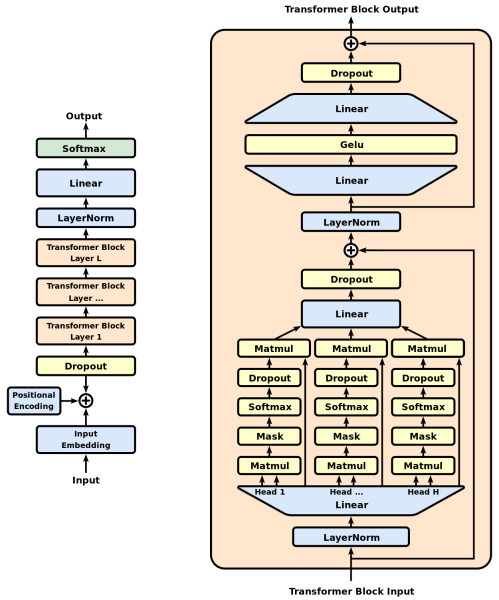


The full transformer

- Given a bag of word embeddings, a transformer **transforms** (!) them successively by applying *N* distinct layers.
- The more layers, the more the output embeddings are good at predicting discourse level properties. The embeddings of the first layers are good at predicting lexical properties



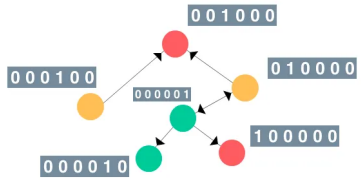
Detailed architecture



Laboratoire de linguistique formelle

Relation with Graph Neural Networks

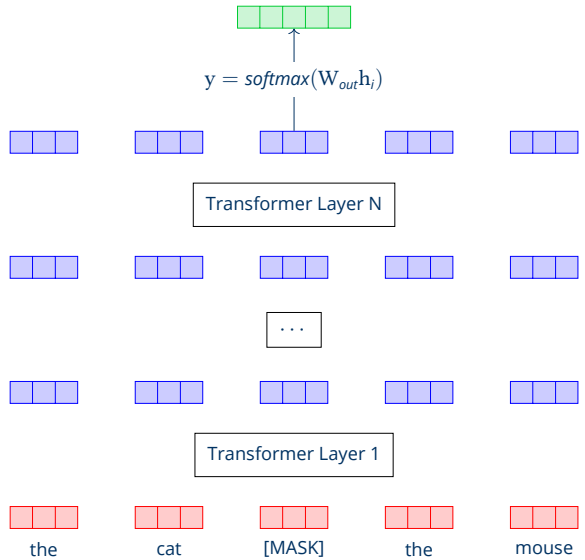
- Graph Neural Networks are networks with graph structure whose nodes are encoded by embeddings



- They are used to model large networks such as social networks, citations networks and networks in biology, physics etc.
- Graph Neural Networks are updated with an iterative message passing method usually decomposed in two steps:
  - **Aggregate**: gathers and aggregate the embeddings of the nodes neighborhood
  - **Update**: Update the node embedding for the next time step
- A transformer can be viewed as a complete graph equipped with aggregate (attention) and update steps (feed forward) that are repeated iteratively (layer iteration)
- The relation with Graph Neural Networks leads to theoretical analysis of the dynamics of transformers.

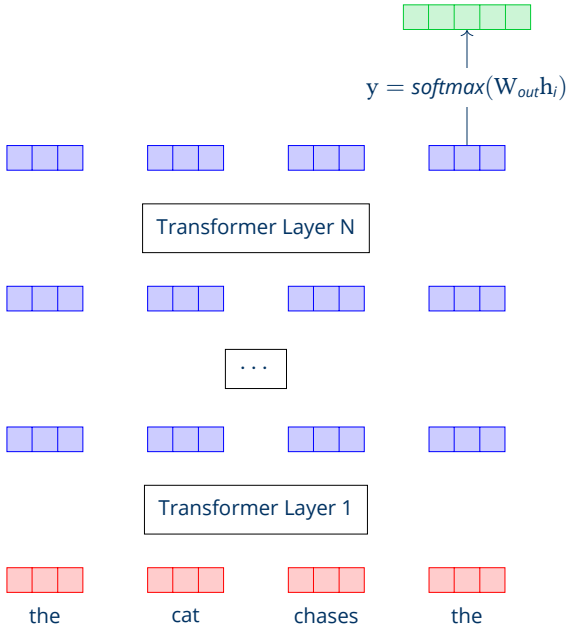
Training a transformer

- Transformers are trained using a *cloze task* scenario.
- **Masked language models** are transformers trained to predict a word randomly masked in the text. The prediction is compared to the ground truth and parameters are updated accordingly.
- BERT is the most cited representative of this family
- Example: *the cat* _____ *the mouse*

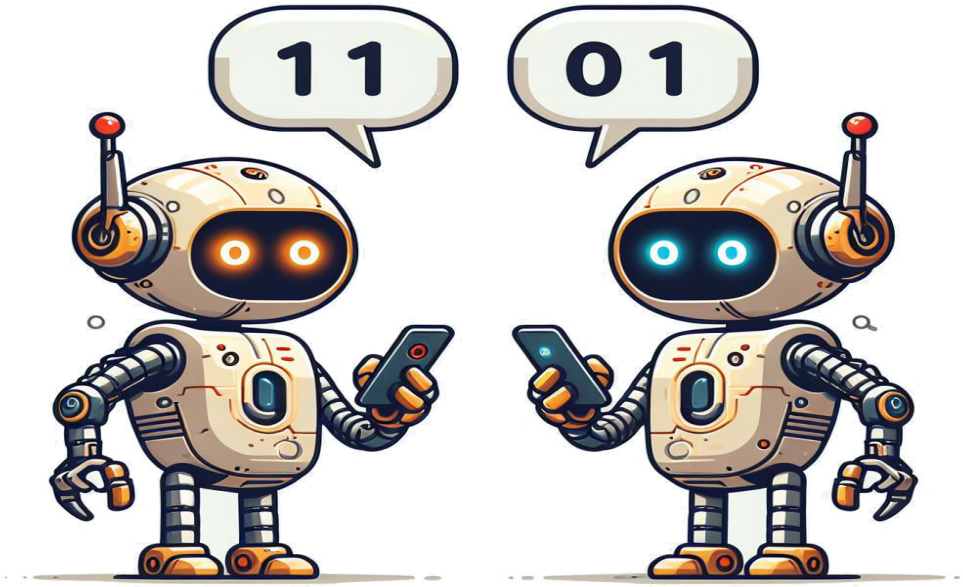


Training a generative transformer

- Transformers are trained using a *cloze task* scenario.
- **Generative language models** are trained to predict the next word given a prefix $P(x_i|x_1 \ldots x_{i-1})$. The prediction is compared to the ground truth and parameters are updated accordingly.
- Example: *the cat chases the* _____



Outline



Word tokenization

Statement of the problem

- Perhaps surprisingly, word tokenization is an old and non trivial problem in natural language processing.
- Since transformers take as input word embeddings, the vocabulary should be a finite set
- In current architectures, the storage of large embedding matrices is responsible for larger memory consumption. Reasonably small vocabularies are preferred
- It is virtually impossible to enumerate the vocabulary of a natural language. There are permanently new words, spelling errors...

The subword solution

Rather than relying naively on whitespace separated word vocabularies inferred from corpora that can be huge and for which it is impossible to guarantee completeness, language models split words into a finite set of subwords that can be kept relatively small.

Byte Pair Encoding

An example

function BytePairEncoding(text,size)
    $V \leftarrow$ set of bytes found in text
    while $|V| <$ *size* **do**
        $bp \leftarrow$ find most frequent char pair in text
        if frequency(bp) = 1 **then**
            return V, T
        end if
        $s \leftarrow$ fresh character not in V
        $V \leftarrow V \cup \{s\}$
        $T \leftarrow T \cup \{s \rightarrow bp\}$
        text $\leftarrow$ replace greedily all bp with s in text
    end while
    return V, T
end function

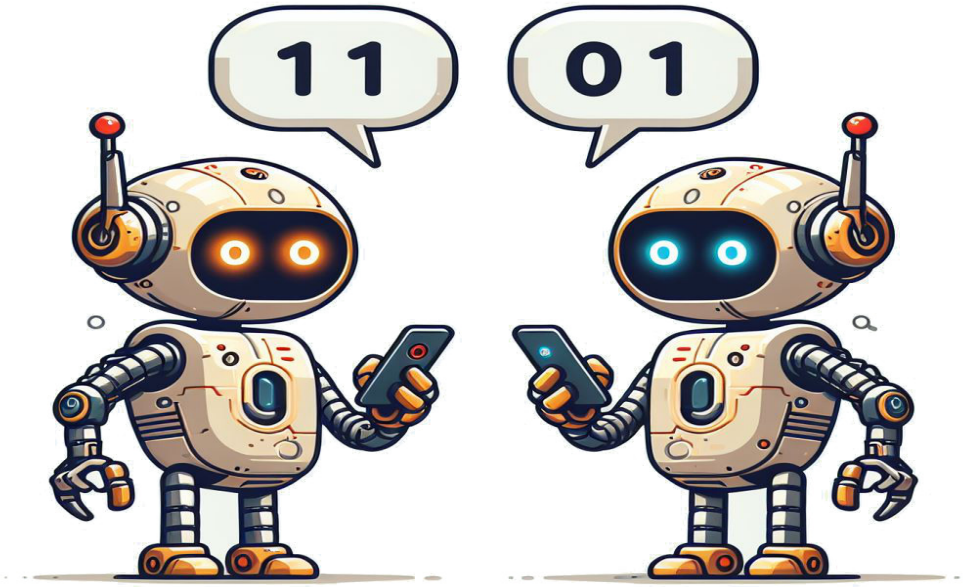
Example

1. Text: `aaabdaaabac`, $b = aa$, $s = Z$
2. Text: `ZabdZabac`, $b = ab$, $s = Y$
3. Text: `ZYdZYac`, $b = ZY$, $s = X$
4. Text: `XdXac`, done

and $T = \{aa \rightarrow Z, ab \rightarrow Y, ZY \rightarrow X\}$

- GPT-4 has a V of size ≈ 100000
- once the BPE is trained, T is used to compress words

Outline

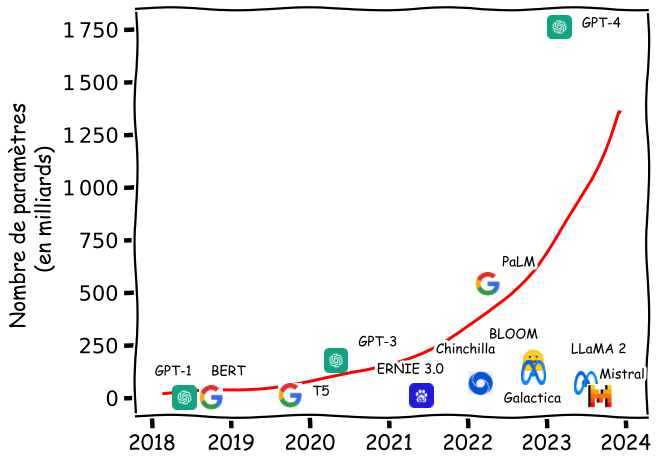


Large Language models

- Language models are **auto-supervised**. It entails that there is no need for data annotation to train them.
- There is a strong interest in training models on extra large data sets because natural language is pretty hard to sample. Vocabulary **representation** is highly biased by the topics of small sized corpora
- Transformers and in particular generative transformers are **designed to compute efficiently at scale** on Graphical Processor Units (GPU)

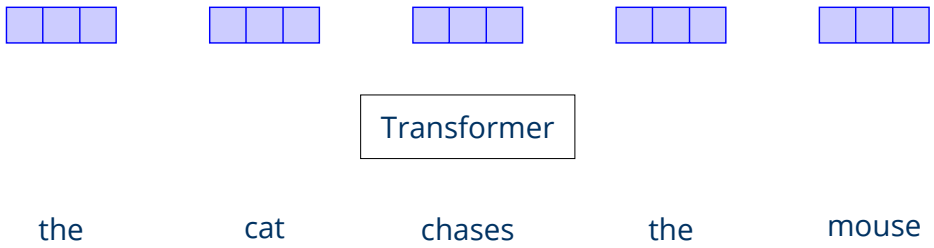
Example: GPT-3

Generative Pretrained Transformer v.3 (OpenAI) is trained over 45 TB of textual data crawled and cleaned from the internet and contains about 175 Billions parameters.



Mathematical representation of text

- Mapping strings (words, tokens) to numerical representations is an old problem for machine learning/applied statistics for natural language (experimentalists also use dummy coding/constract coding...)
- Word Embeddings (LSA, Word2vec) provide methods for mapping the vocabulary to vectors/representations
- Language models provide in context representations for tokens where one may *expect* to have disambiguated vectors, that anaphora is partly taken into account etc.
- **Representation** In sum a neural language model can be viewed as a function from Strings to Vectors that can be used as input for further statistical modeling.



Qualitative evolution of representation methods

A summary

Name	Improvement
One hot coding	baseline
Word embeddings	Distributional hypothesis
Neural Language models	Contextualized embeddings
Large Language models	Scale up (parallelism with GPU)

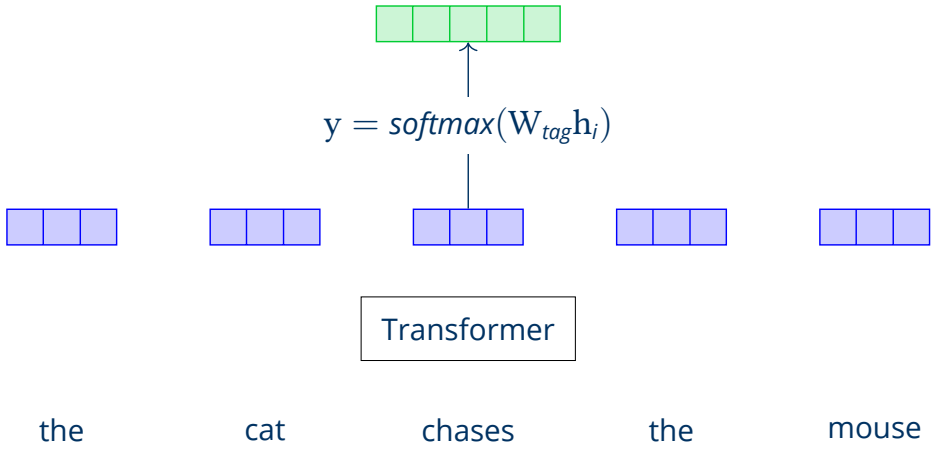
- One hot encoding and word embedding are essentially used for providing word representations, they can be used to provide basic text representations by assuming bag of words
- Neural Language models (markovian) are generally reasonable predictors with limited context and remain relatively approximate
- Neural Language models (RNN/LSTM) are better predictors because of larger context. But in practice they then to have limited memory and cannot be easily scaled up since their architecture is temporal rather than parallel

Training at scale

- Training up to a GPT-2 size language model can be achieved today on a single compute node with 8 A-100 GPU in about 4 days (excluding parameter search)
- Scaling up to the full LLM size is somewhat more involved. For instance LLama2-70B used 6000 GPU for 12 days for training from a chunk of 10TB from internet and generates a 140GB file of parameters.

Fine Tuning

- Large Language models can be refined or **fine-tuned**
- The assumption is that the large language model captures the general knowledge of language because it is trained on a very large corpus.
- Fine tuning is a specialization of the language model for a specific task where we only have a small corpus of (domain specific) annotated data.
- Example, part of speech tagging:



- Fine tuning implies training the model by updating all the parameters, including those of the transformer.

ChatGPT

- Given a prefix or a **prompt** one can generate text with a generative language model. For a zero prefix:

  $x_1 \leftarrow [START]$
  while True **do** :
  $x_{t+1} \sim P_W(x|x_1 \dots x_t)$
  end while

- For question answering or dialogue, one can give a natural language question as prompt and asks chat GPT to continue the text.

  How do you do ? *I'm doing well, thank you! How can I assist you today ?* ☺

- Since there are few dialogues in the internet corpus. The language model (GPT 3.5) is refined using **Reinforcement Learning with Human Feedback** (RLHF). The language model is refined by asking humans to annotate whether the interaction is considered as positive or negative (👍 👎)

Zero and Few shot prompting

Binz and Schulz 2023

- Few shot or zero shot prompting aims to have a language model perform a task by prompting zero or a few examples. Assume the prompt:

```
This is awesome! // Positive
This is bad! // Negative
Wow that movie was bad! // Negative
What a horrible show! //
```

  and you expect the model to output `Negative`

- When the task is too complex this does not work anymore. Consider the prompt:

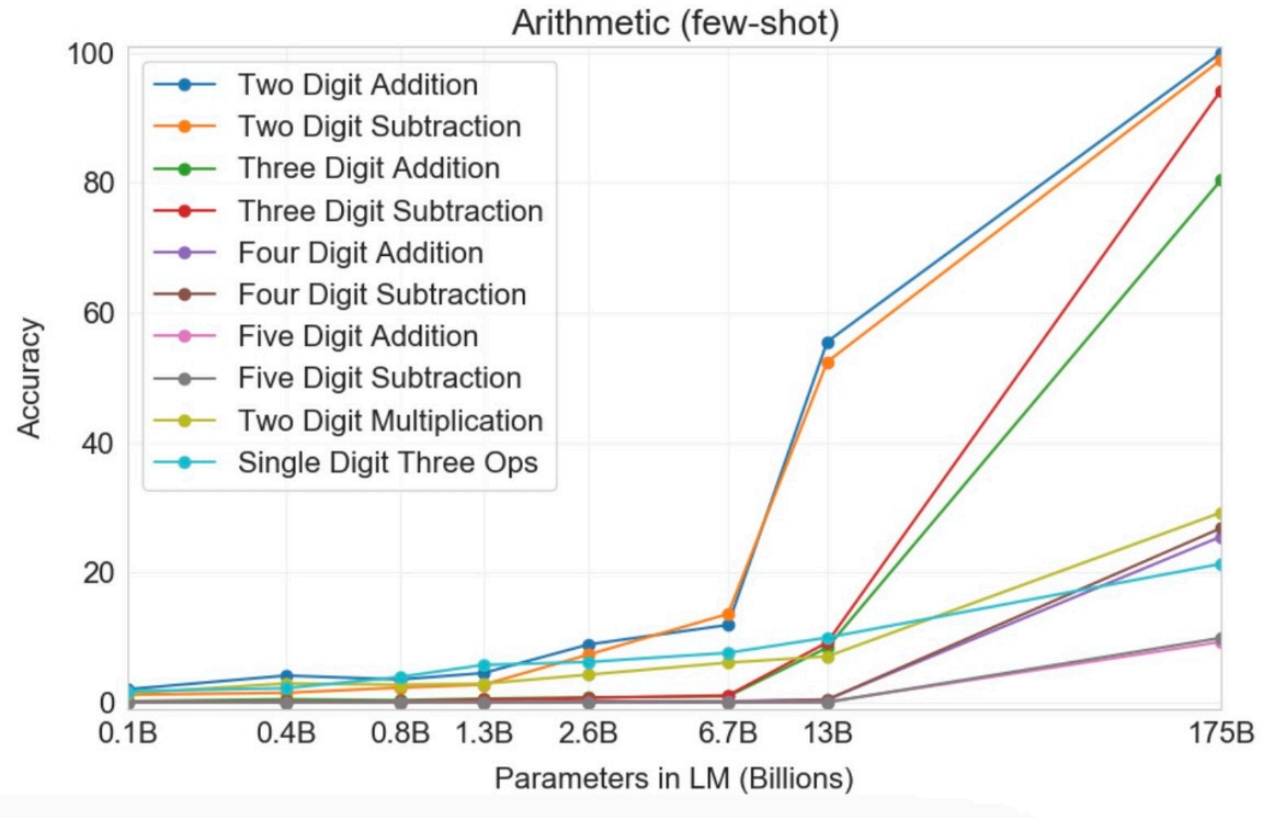
```
The odd numbers in this group add up to an even number: 4, 8, 9, 15, 12, 2, 1.
A: The answer is False.
The odd numbers in this group add up to an even number: 17,  10, 19, 4, 8, 12, 24.
A: The answer is True.
The odd numbers in this group add up to an even number: 16,  11, 14, 4, 8, 13, 24.
A: The answer is True.
The odd numbers in this group add up to an even number: 17,  9, 10, 12, 13, 4, 2.
A: The answer is False.
The odd numbers in this group add up to an even number: 15, 32, 5, 13, 82, 7, 1.
A:
```

  and you expect the model to output `The answer is False`

  (odd sum = 41) while GPT tends to fail here

Emergent abilities

- An ability is emergent if not detectable with standard or small sized models but become apparent with large models
- Example or arithmetic

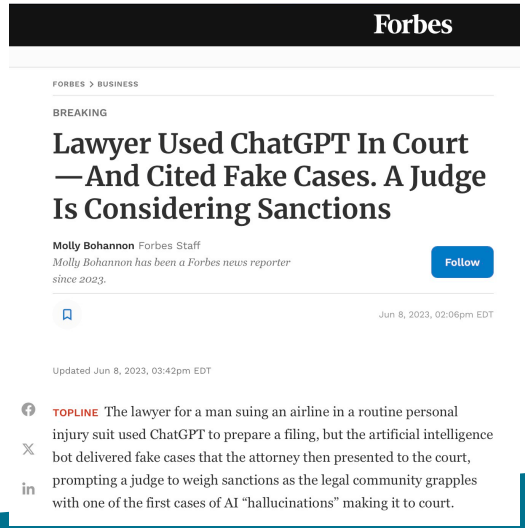


Memory, hallucinations, generalization

- A generative LLM can be seen as a database compressing lossily terabytes of data.
- Memory is not guaranteed to be consistent. Famous example (fixed since then):
  - `H: Who's Tom Cruise mother ?`
  - `M: Mary Lee Pfeiffer`
  - `H: Who's the son of Mary Lee Pfeiffer ?`
  - `M: dunno.`
- The capacity of models to generalize beyond rote memorization is badly understood.

Hallucinations

- Hallucinations are cases of text generated by LLM that are well formed but either factually incorrect, nonsensical, crafted out of the blue.
- Hallucinations have several origins: underfitting, overfitting, general data biases...
- As a result, the output of an LLM is not thrustable and it motivates explainable AI research.



Forbes

FORBES > BUSINESS

BREAKING

Lawyer Used ChatGPT In Court —And Cited Fake Cases. A Judge Is Considering Sanctions

Molly Bohannon Forbes Staff

Molly Bohannon has been a Forbes news reporter since 2023.

Follow

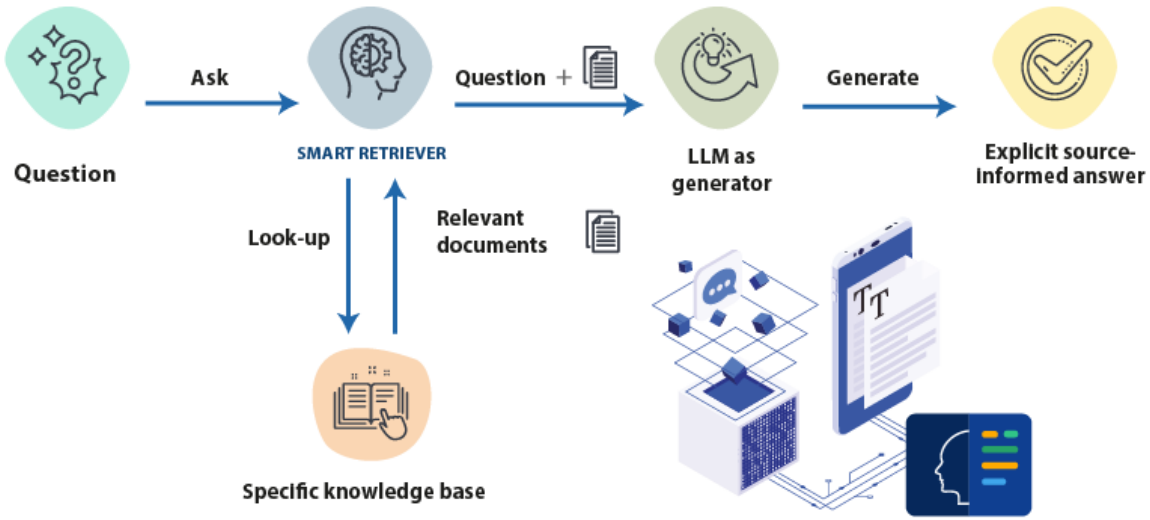
Jun 8, 2023, 02:06pm EDT

Updated Jun 8, 2023, 03:42pm EDT

TOPLINE The lawyer for a man suing an airline in a routine personal injury suit used ChatGPT to prepare a filing, but the artificial intelligence bot delivered fake cases that the attorney then presented to the court, prompting a judge to weigh sanctions as the legal community grapples with one of the first cases of AI "hallucinations" making it to court.

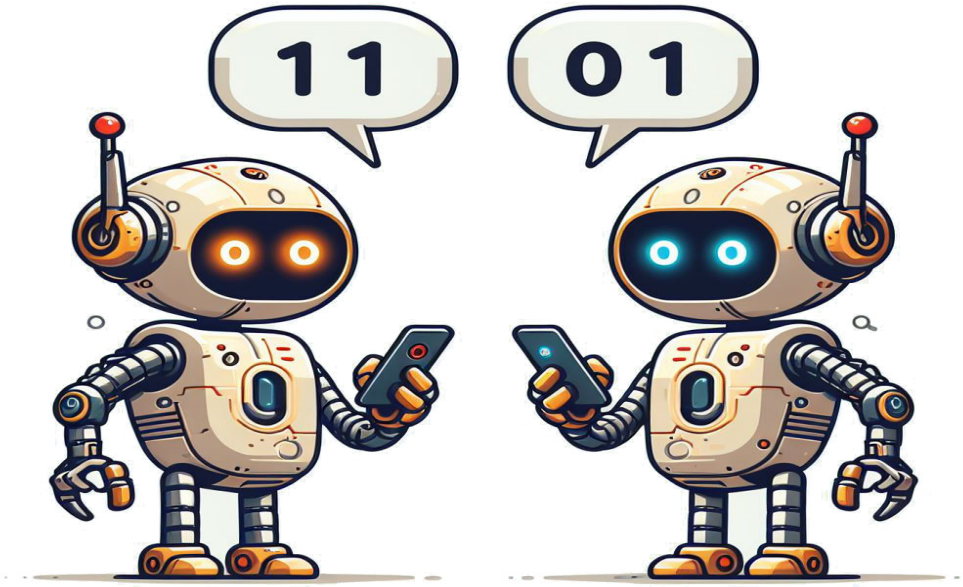
Retrieval augmented Generation (RAG)

- Retrieval augmented Generation is an example of method that aims to correct for hallucinations.
- It is used in the context where the LLM is queried for confidential, very technical or specific information
- The idea is to send the query to a search engine, retrieve the result and query the LLMs both with the query and the retrieved results



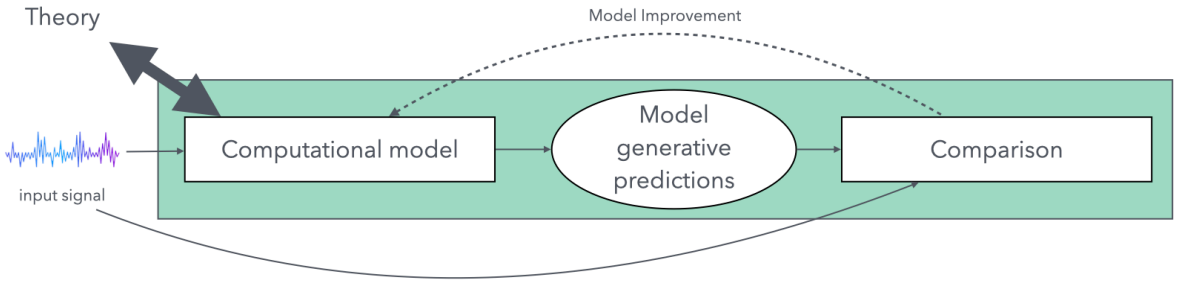
Laboratoire de linguistique formelle

Outline



Prediction for language models

Generative prediction

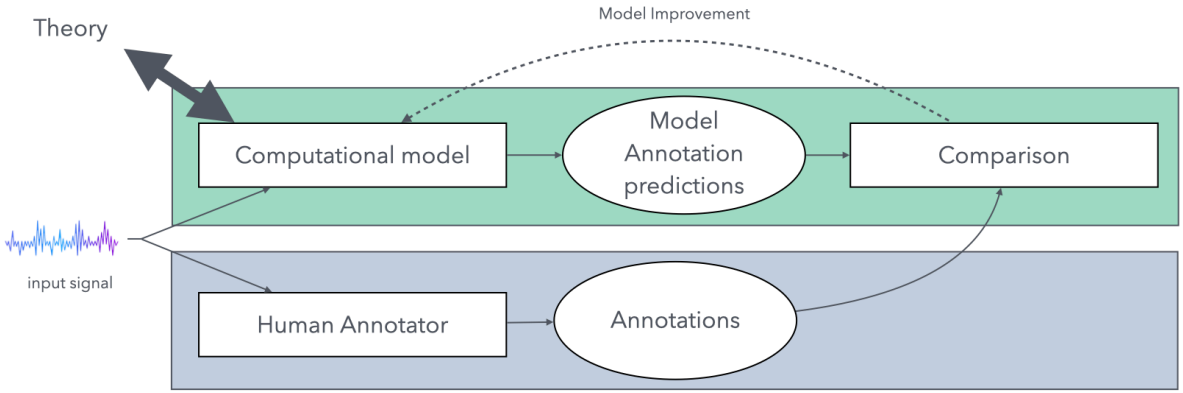


Example

- **Prompt:** The island of Porquerolles is _____
- **Model prediction:** The island of Porquerolles is a captivating island nestled in the Îles d'Hyères, located off the coast of Hyères on the French Riviera.
- **Evaluation:** similarity between the ground truth and the generated text (BLEU, ROUGE, accurracy)

Predicting annotations

Annotation models



Example

- **Input:** The island of Porquerolles is located on the French Riviera
- **Model Prediction:** D N P PN V VPP P D A NP
- **Evaluation:** Comparison between model predictions and human annotations. Accuracy, F-score...This scenario requires to **fine tune** the language model on annotated data

Prediction and evaluation

The Glue Benchmark (link)

9 datasets to test language models on a variety of prediction tasks. NLP oriented and initially for BERT. Here are some examples:

Grammaticality (Cola)	This building is than that one.
	agrammatical
Sentiment analysis	The movie is funny, smart, visually inventive, and most of all, alive
pos,neg,neutral	0.93 (pos)
Paraphrase	Yesterday, Taiwan reported 35 new infections, bringing the total number of cases to 418.
	The island reported another 35 probable cases yesterday, taking its total to 418.
	yes
Reference (WNLI)	Lily spoke to Donna, breaking her concentration.
	Lily spoke to Donna, breaking Lily's concentration.
	yes

Prediction and evaluation

The BLiMP Benchmark (link)

67 datasets of **minimal pairs** on a variety of linguistic properties (English). This is **artificially generated data** testing a wide range of properties in morphology, syntax and semantics. Here are some examples:

Anaphora	Many girls insulted themselves
	* Many girls insulted herself
Arg Structure	Rose wasn't disturbing Mark
	* Rose wasn't boasting Mark
S/V Agreement	These casseroles disgust Kayla
	*These casseroles disgusts Kayla
Binding	Carlos said that Lori helped him
	*Carlos said that Lori helped himself

The test amounts to compare the probability given to each element of the pair. Success if the probability is higher for the grammatical element

Prediction and evaluation

Code generation

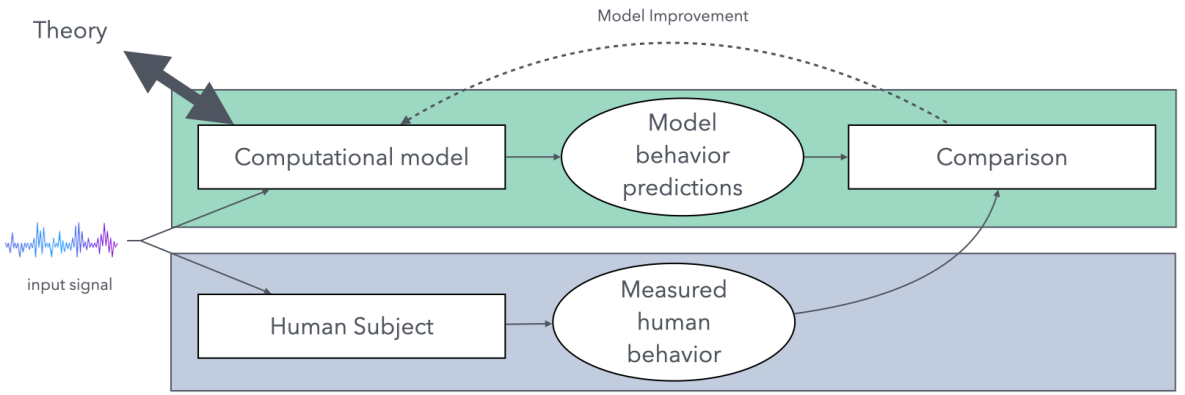
- This time the prediction is structured: predict the code given the intent.
- *Split a string in chunks and keep the separators*

```
import re

def split_and_keep_separators(s, separators):
    pattern = f"({'|'.join(map(re.escape, separators))})"
    chunks = re.split(pattern, s)
    return [chunk for chunk in chunks if chunk]

text = "Split a string; in chunks! and keep, the separators."
separators = [' ', ';', '!', ',', '.']
result = split_and_keep_separators(text, separators)
print(result)
```

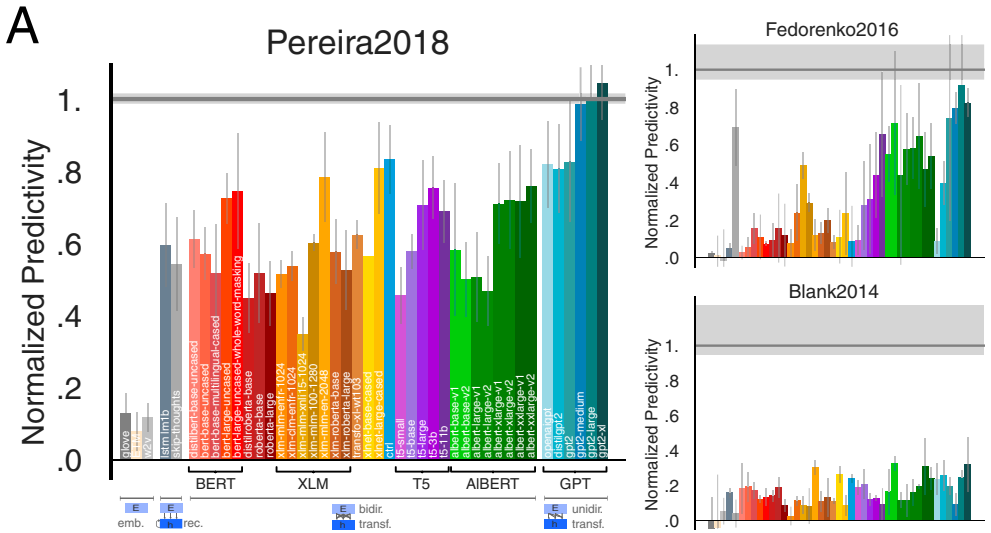
Predicting behavior or physiology

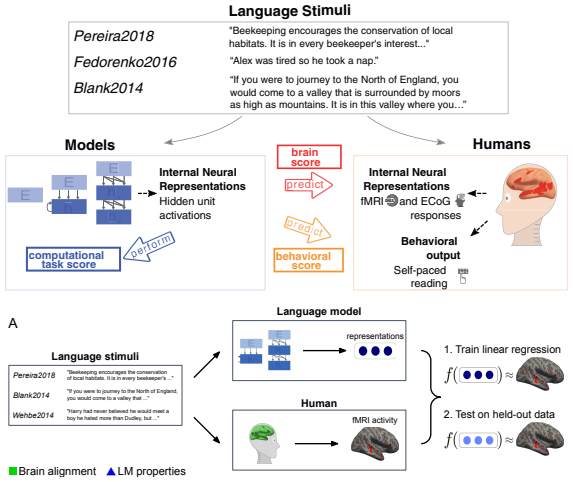


Example

- **Input:** By July 1915 all seventeen of the German Imperial Navy...
- **Model Prediction:** -1.684012 -1.203219 -2.527704 -0.137419 ...
- **Evaluation:** Comparison between model predictions and human measures (Eye Reading, physiological). correlation, R^2, likelihood function...

Example (Schrimpf et al. 2021)





Datasets

(Pereira 2018) fMRI, unconnected short passages (reading)

(Fedorenko 2016) ElectroCorticoGraphy, unconnected sentences (reading)

(Blank 2014) fMRI, Natural Stories (listening)

Brain score

Schrimpf et al. (2021) use a Brain Score metric. They train a model that predicts the signal given the language model representations and compute the pearson correlation with the signal. The **brain score** is a normalization of the correlation score with a noise ceiling.

Many other papers are out, e.g. (Caucheteux and King 2022). Methods and metrics vary from paper to paper and comparisons are difficult (Karamolegkou et al. 2023).

Some datasets

Natural stories (SPR)

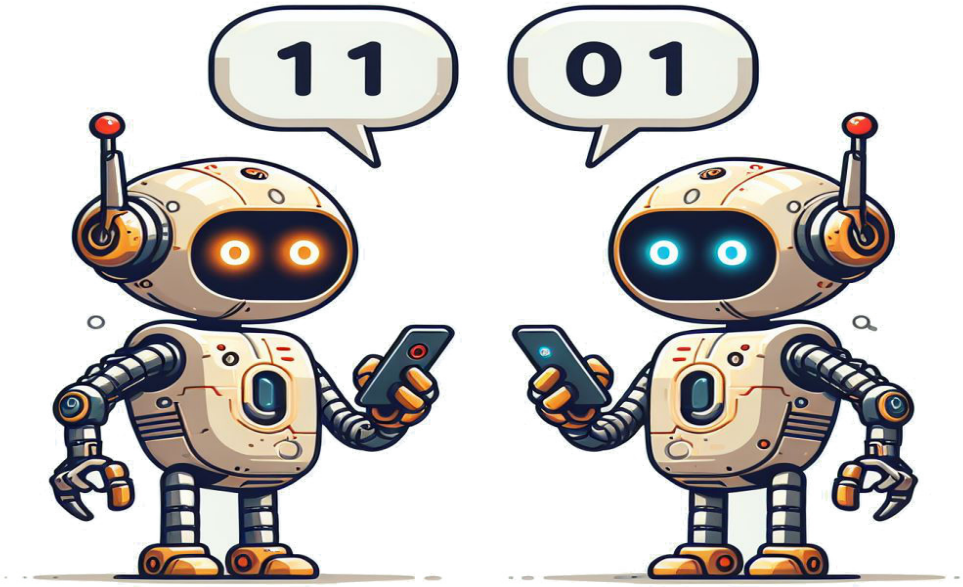
id	form	RT_mean	RT_std	sent_id
1	If	369.011905	160.579935	1
2	you	368.183908	168.027166	1
3	were	344.318182	224.916666	1
4	to	354.639535	310.065644	1
5	journey	349.674157	198.212855	1
6	to	376.370787	334.078937	1
7	the	327.310345	181.580869	1
8	North	365.494382	336.905271	1
9	of	344.931034	207.073001	1
10	England,	400.269663	295.972080	1
11	you	372.590909	212.704838	1
12	would	350.379310	159.260737	1
13	come	319.080460	117.275983	1
14	to	355.931034	318.756297	1
15	a	343.858824	213.122095	1
16	valley	330.732558	180.977699	1

Common datasets

Dataset	Method	Authors
Harry Potter	fMRI	Wehbe 2014
Harry Potter	MEG	Wehbe 2014
Pereira Dataset	fMRI	Pereira 2018
Natural Stories	SPR	Futrell et al. 2017
Natural Stories Audio	fMRI	Zhang et al 2020

Natural stories is a self paced reading corpus made of 10 stories of about 1000 words each. The stories are read by 181 mechanical Turks (most participants read 5 stories). The original corpus was transformed to increase the amount of syntactically complex constructions.

Outline



Explanation: the black box problem

- **The question**. Models do predictions. The question is to find out what component(s) of the model is(are) responsible for the prediction: feature significance, feature importance, rules used, sequence of inferences...
- **Black box problem** Deep learning models cannot be analyzed variable wise. They are not immediately transparent as the physical model because it manipulates millions or billions of interdependent parameters
- **Explainable AI (XAI)** There are however methods developed in Artificial Intelligence for explaining and interpreting models.

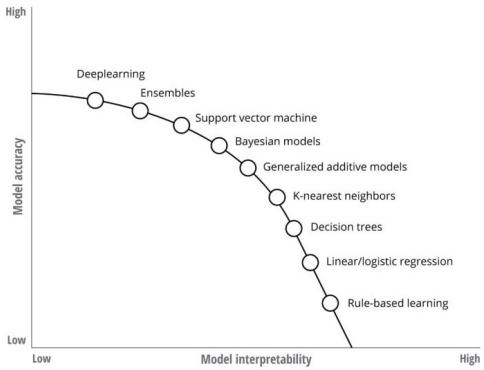
Deep learning is not an exception

This is not a property specific to deep learning. Large multivariate statistical models or large symbolic rule systems are already hard to interpret unless properly controlled.

Explanation: the black box problem

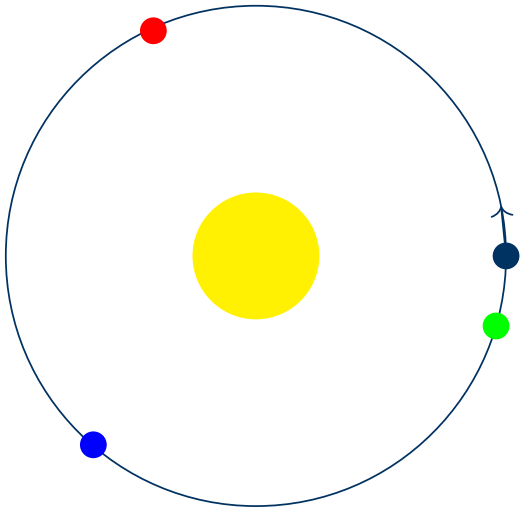
- **The question**. Models do predictions. The question is to find out what component(s) of the model is(are) responsible for the prediction: feature significance, feature importance, rules used, sequence of inferences...
- **Black box problem** Deep learning models cannot be analyzed variable wise. They are not immediately transparent as the physical model because it manipulates millions or billions of interdependent parameters
- **Explainable AI (XAI)** There are however methods developed in Artificial Intelligence for explaining and interpreting models.

Deep learning is not an exception



Explanation: transparent models

Ideal situation (circular orbits)



α	θ_0	t
2	0	0
2	0	1
2	0	2
2	0	3

$$\begin{cases} x_t &= r\cos(\alpha t + \theta_0) \\ y_t &= r\sin(\alpha t + \theta_0) \end{cases}$$

Equation vs NN

- With the equation we predict the position of the satellite with 4 parameters: the radius r of the orbit, the angular speed of the satellite α, the time t and the initial angle θ_0.
- With a generative neural network, we would take successive snapshots of the orbit, and build a model with thousands of parameters predicting which image follows the current generated images.

Explanation: transparent models

Close to ideal situation (applied statistics methods for language)

- **Regression with a limited number of boolean predictors** Example: prediction of adjective noun in French (postverbal complements, dative shift...boolean word order) given few non correlated binary predictors:
  - x_1 = {A longer than noun, otherwise}, x_2 = {Noun is color , otherwise}...
  - y = {A before noun, otherwise}

$$P(Y = 1|\theta, \mathrm{x}) = \sigma(\theta_1 x_1 + \ldots + \theta_n x_n + \theta_b)$$

For language models

- Sometimes we do not have access to the parameters
- Interpreting the parameters is meaningless

Explanation : model comparison (ablation)

- Comparison of two **nested models**, a full model M_F and an ablated model M_A is common in applied statistics. It is supported by a log-likelihood ratio test. We have that:

$$\log L_F(\mathrm{x}, \theta) \geq \log L_A(\mathrm{x}, \theta')$$
$$2\left[\log L_F(\mathrm{x}, \theta) - \log L_A(\mathrm{x}, \theta')\right] \approx \chi^2$$
$$2 \log \frac{L_F(\mathrm{x}, \theta)}{L_A(\mathrm{x}, \theta')} \approx \chi^2$$

yielding the so called **log likelihood ratio test**.

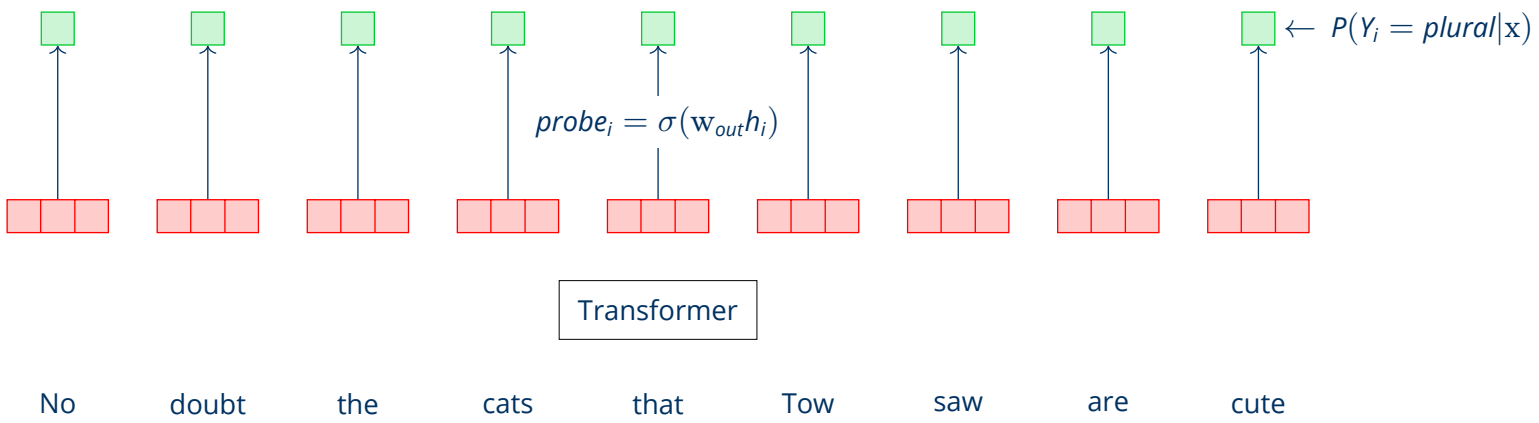
Supposes a reasonably small model

- This test can be used for simple regression models to test whether some group of predictors are improving the fit
- On the other hand, model comparison cannot be performed easily with Large Language Models, training time is prohibitive

Explanation for black box models: probing

- Probing is a post-hoc method, applied on an already trained model.
- It allows to observe the model representations
- It consists in training a probe on the model representations. The probe will predict some interpretable property.

Example: how does a transformer encode subject verb agreement ?



Rationale: if the number information is encoded in the embeddings we can learn a classifier to predict the number.

Explanation : counterfactual method

What if scenarios

The counterfactual method aims to identify causal relations in what if scenarios: what happens to the model prediction if one changes/removes a factual information ?

- Example (past participle agreement in French in object relative):
  how does a Language model predicts agreement ?

Les	mouettes	~~que~~	Pierre	a	vu | vues
The	seagulls	that	Peter	has	seen

- Post-hoc method: prevent the transformer to attend at *que* when predicting the past participle
- Applies to a full dataset of controlled examples
- Note that sometimes it is better to modify the model rather than the data. For instance removing a word in a sentence is likely to create agrammaticality. Instead removing attention when predicting is thought to be better.

Local Interpretable Model-Agnostic Explanations

LIME (Ribeiro et al. 2016)

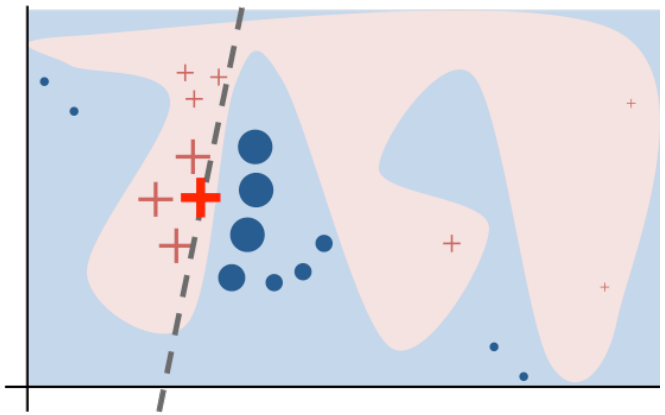
Approximate a single decision by a linear model

The LIME method aims to explain a local prediction by sampling around *D* similar examples. It compares the prediction of the model with those of an interpretable linear model trained on *D*

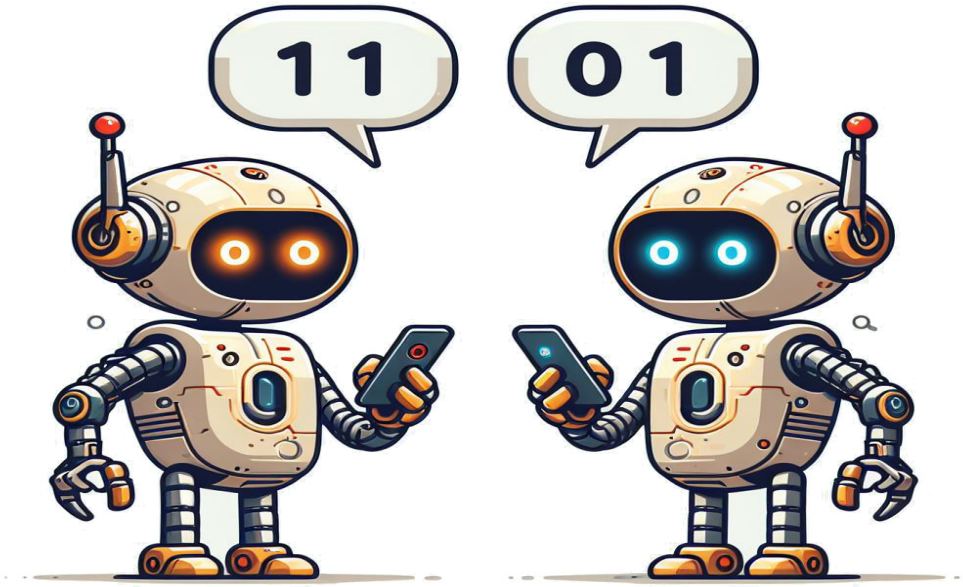
Les	mouettes	~~que~~	Pierre	a	vu | vues
The	seagulls	that	Peter	has	seen
1	1	0	1	1	

Les	~~mouettes~~	que	Pierre	a	vu | vues
The	seagulls	that	Peter	has	seen
1	0	1	1	1	

Les	mouettes	que	~~Pierre~~	a	vu | vues
The	seagulls	that	Peter	has	seen
1	1	1	0	1	



Outline



The landscape of neural language models

- Neural language models are **self supervised** and **GPU friendly**: they can be trained at unprecedented scale.
- The training set is somewhat ”natural”, at least sampled with some opportunism from the web with unknown biases.
- Large Language models have **surprisingly good predictive properties** for a wide range of tasks including predicting behavioral and neural signal.
- **Explanation** is a research topic in itself:
  - Annotations and annotated data sets may be used for providing explanations
  - The size of the training set and the time to train a model makes model comparison hard at LLM scale. Model comparison remains possible in laboratories (although expensive) at GPT scale.
- **Evaluation:** the Natural Language processing community has set up evaluation **benchmarks** (e.g. Glue and SuperGlue) that allow to evaluate and compare language models by probing on standard **downstream tasks**: question answering, sentiment analysis, paraphrase detection, textual entailment, acceptability judgments...