Fundamentals of Natural Language Processing

netherlands eScience center

Syllabus – Day 1

- Introduction
  - What is Natural Language Processing?
  - Why learn NLP fundamentals?
  - Defining NLP Tasks
  - A primer on linguistics
- Word Representations
  - Preprocessing Operations
  - NLP Pipeline
  - Word Embeddings
  - Explore and Train Word2Vec

Syllabus – Day 2

- Transformers
  - The Transformer Architecture
  - Introduction to BERT
  - BERT as a Language Model
  - BERT for Text Classification
  - Evaluating Classifiers
- Large Language Models (LLM)
  - What are LLMs?
  - Prompting to solve NLP Tasks
  - Using local LLMs with Ollama
  - Drawbacks and biases with LLMs

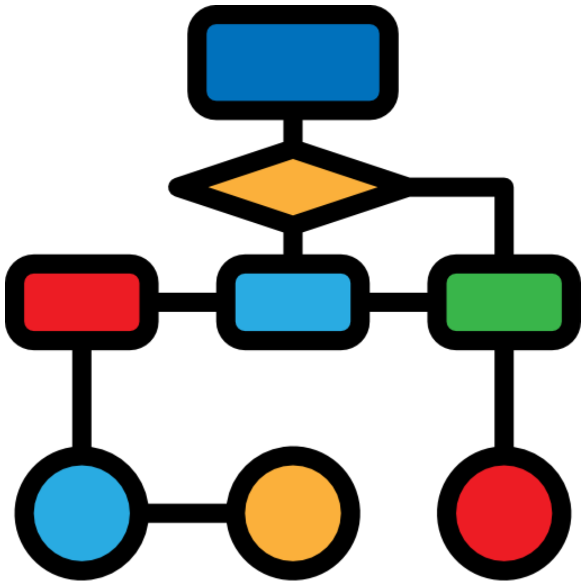
Episode 01: Introduction

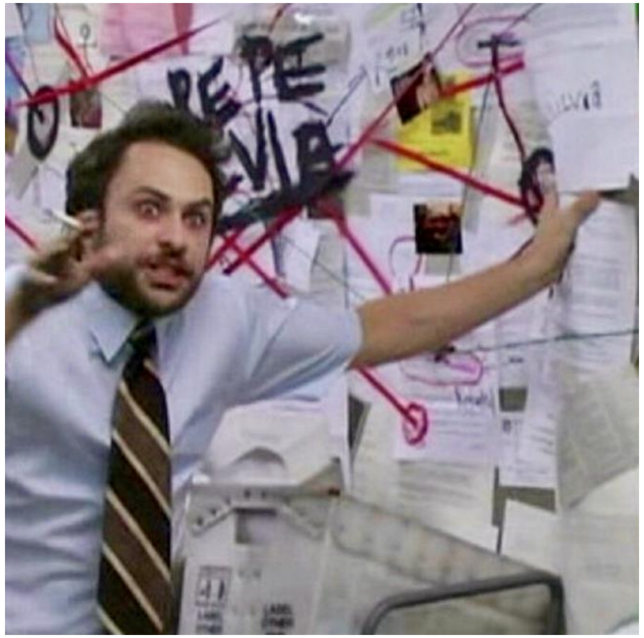
netherlands eScience center

What is Natural Language Processing (NLP)?

netherlands eScience center

Why "Natural" Language?

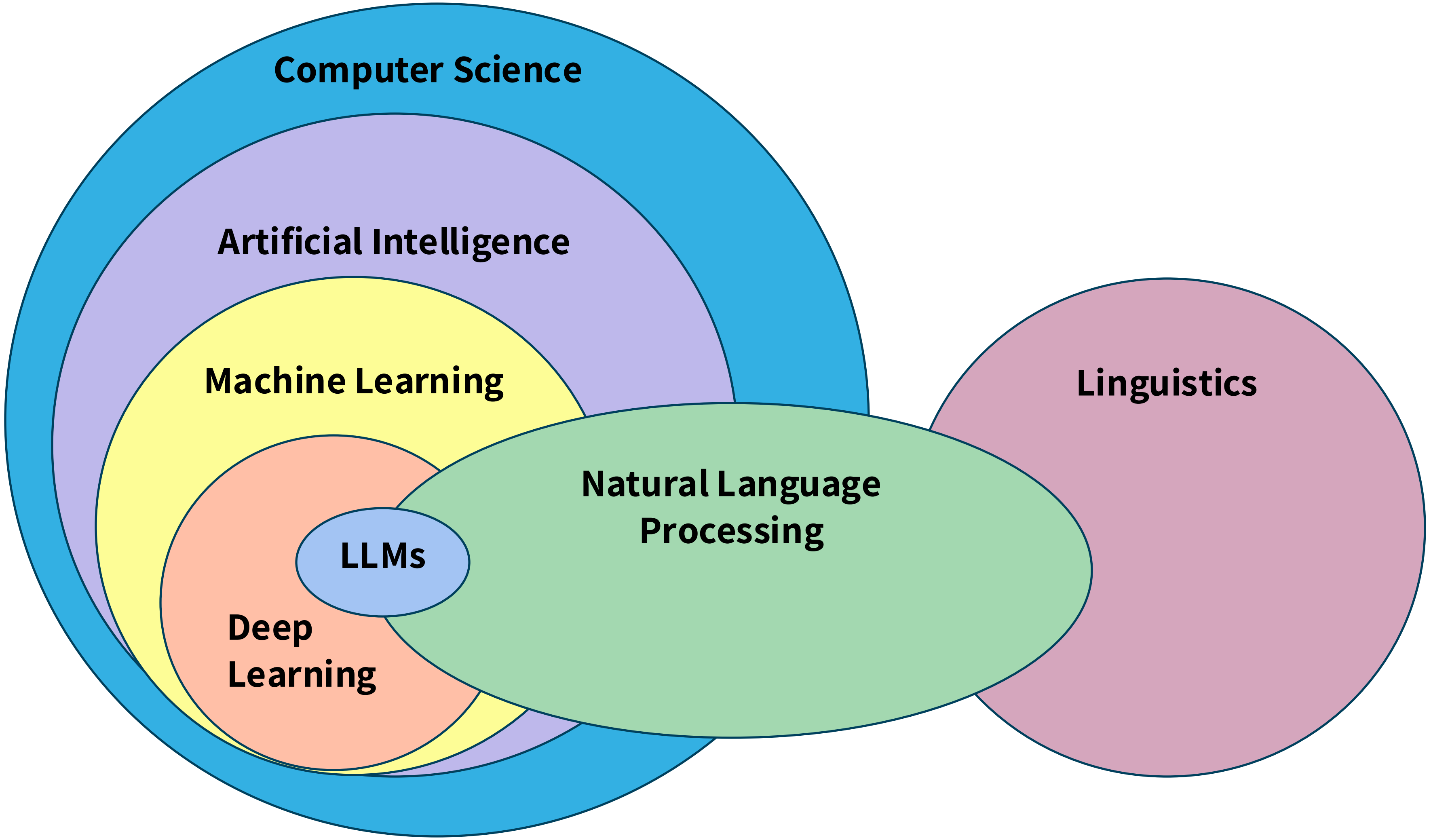




Artificial Language:
Computers excel at following **deterministic** instructions

Human Language:
Language is ambiguous and highly **context-dependent**

What is Natural Language Processing?



What is Natural Language Processing?

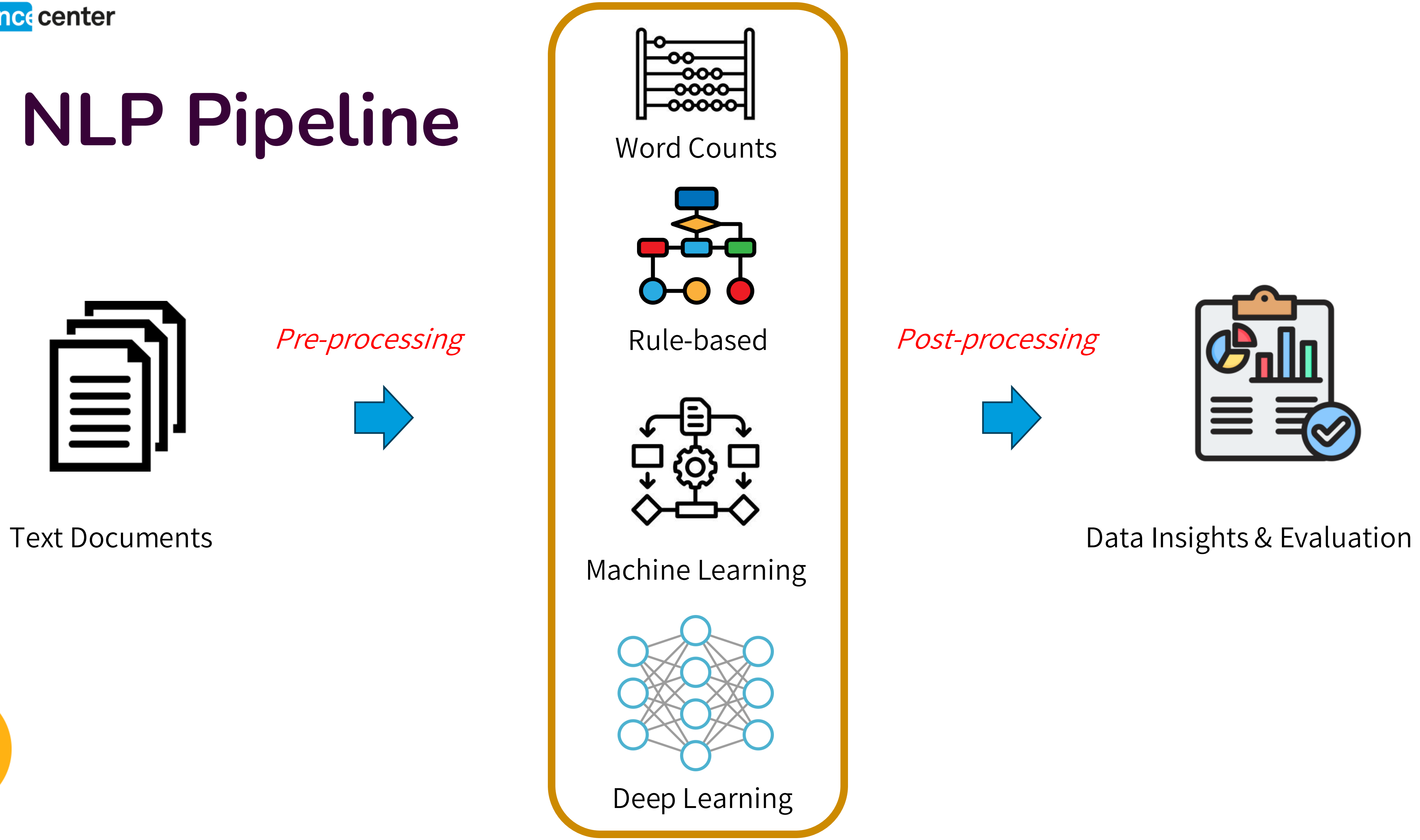
- NLP is an area of **research and application** that focuses on making human languages processable for computers.
- **More than 7000 human languages** are spoken around the world, each with its own grammar, vocabulary, and cultural context.
- **Today we focus on written English**, we leave out speech and video, as they require a different kind of input processing.
- **Most linguistic concepts remain valid across languages**, however they might require different approaches

netherlands eScience center

Challenge 1

- Name at least three tools/products that you use on a daily basis and that you think leverage NLP techniques. To do this exercise you may make use of the Web.

NLP Pipeline



Tokenizing a string…

- Text data is represented as “strings” inside computers.
- Humans break up texts into words to understand the content. In NLP, a similar process is called **tokenization.**
- This process might seem trivial at first, but even here assumptions and decisions are already required.

[CODE]

First attempt to tokenize strings in python

Does the LLM know more than me?

ChatGPT

Share

How many tokens are in "标记化并不总是那么简单"?

The sentence **"标记化并不总是那么简单"** contains **11 Chinese characters**. When tokenized using models like GPT-4 (especially with the **tiktoken** tokenizer used by OpenAI), Chinese is generally tokenized **character by character**, with **each character typically being one token**.

Tokens in Large Language Models

- LLM tokenizers do not segment text into human words.
- They use tokenizers that have been trained to statistically maximize computing resources and performance in different tasks.

- See [tiktokenizer]

Create image Please generate an illustration that displays the word "strawberry" in a colorful font. Put a red circle around every 'r'. Finally, at the bottom, write down how many r's there are in the word strawberry, as a number.

Image created

Source: [Reddit]()

The importance of NLP Fundamentals

The importance of NLP Fundamentals

- Learn to formulate the problem → **NLP Tasks**
- What are the limitations of current LLMs? → **Linguistics**
- Can I use a simpler approach to solve it? → **NLP Pipelines**
- How do I validate my results? → **Evaluation**
- If I use an LLM…
  - How do I know if I can trust the outputs? → **All of the above**
  - What are possible sources of errors → **All of the above**

Language as Data

netherlands eScience center

Language as Data

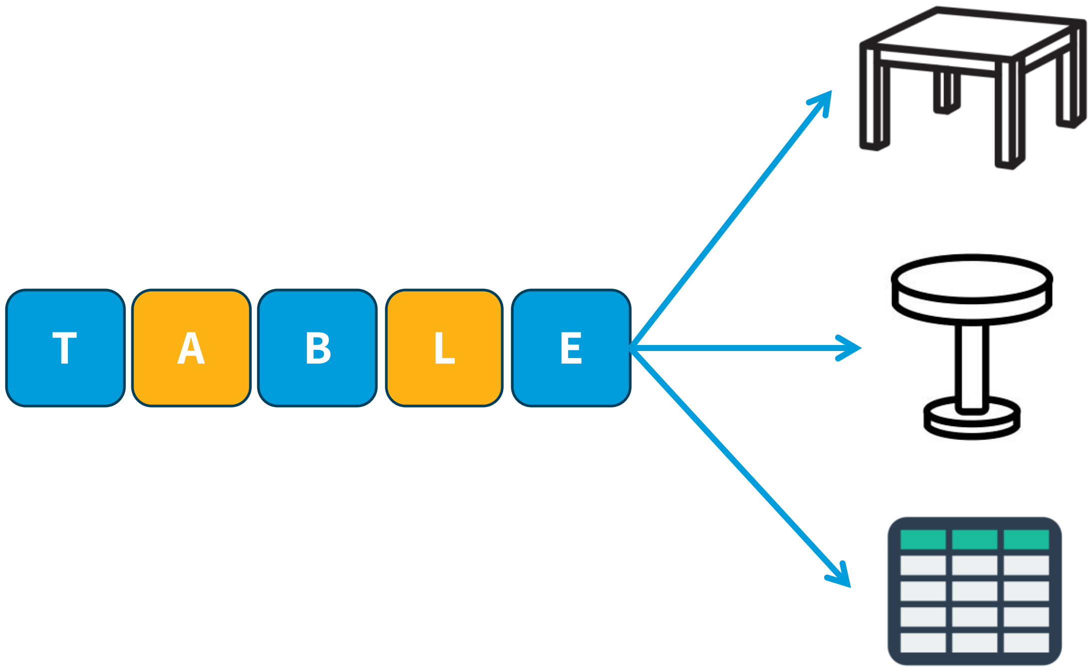
- Your first task as an NLP practitioner is to understand what *aspects* of textual data are relevant for your application
- Basic feature examples: word counts, presence of keywords or phrases, sentence length, etc…

Language as Data

- Knowing what is relevant, helps you to:
  - Identify techniques to systematically extract **meaningful features** from text.
  - Shape your text into a suitable **data format**.
  - **Choose an existing model** (ML algorithm or neural architecture) that can help solve our problem at hand.

What is a word?

- In NLP, words are the most intuitive features you can use.



What is a word?

- A **word type** is the word in the abstract.
- A **word token** refers to each occurrence of a word in a text.
- BEWARE: *token* and *word* are not always synonyms, but frequently you will find this simplification.

*Next to my dinning **table** you can see my nice coffee **table**.*

[CODE]

Using spaCy to process text files

See Spacy Docs

Pre-trained models & Fine-tuning

- A **pre-trained model** has already been optimized on relevant data for a given task, and we can use it right away with our own data. Ideally, publicly released pre-trained models have undergone rigorous testing.
- **Fine-tuning** is a common practice where we take advantage of the knowledge of a related pre-trained model and adjust it using our own data. This usually needs considerably less data compared to starting from scratch.

Challenge 2

- Use the spaCy Doc object to compute stats about one book. Example:
  - Give the list of the 20 most common verbs in the book
  - List how many Places are identified in the book (Label = GPE)
  - How many different entity categories are in the book?
  - Who are the 10 most mentioned PERSONs in the book?
  - Any other aggregate you like

NLP Tasks

NLP Task

- An NLP task is a well-defined problem that involves applying computational algorithms to achieve a specific linguistic or functional objective.
- From the Machine Learning perspective there are supervised and unsupervised tasks.
- From the Deep Learning perspective different architectures are optimal for different problems.
- Tasks can also be grouped together according to their goals.

Unsupervised Learning

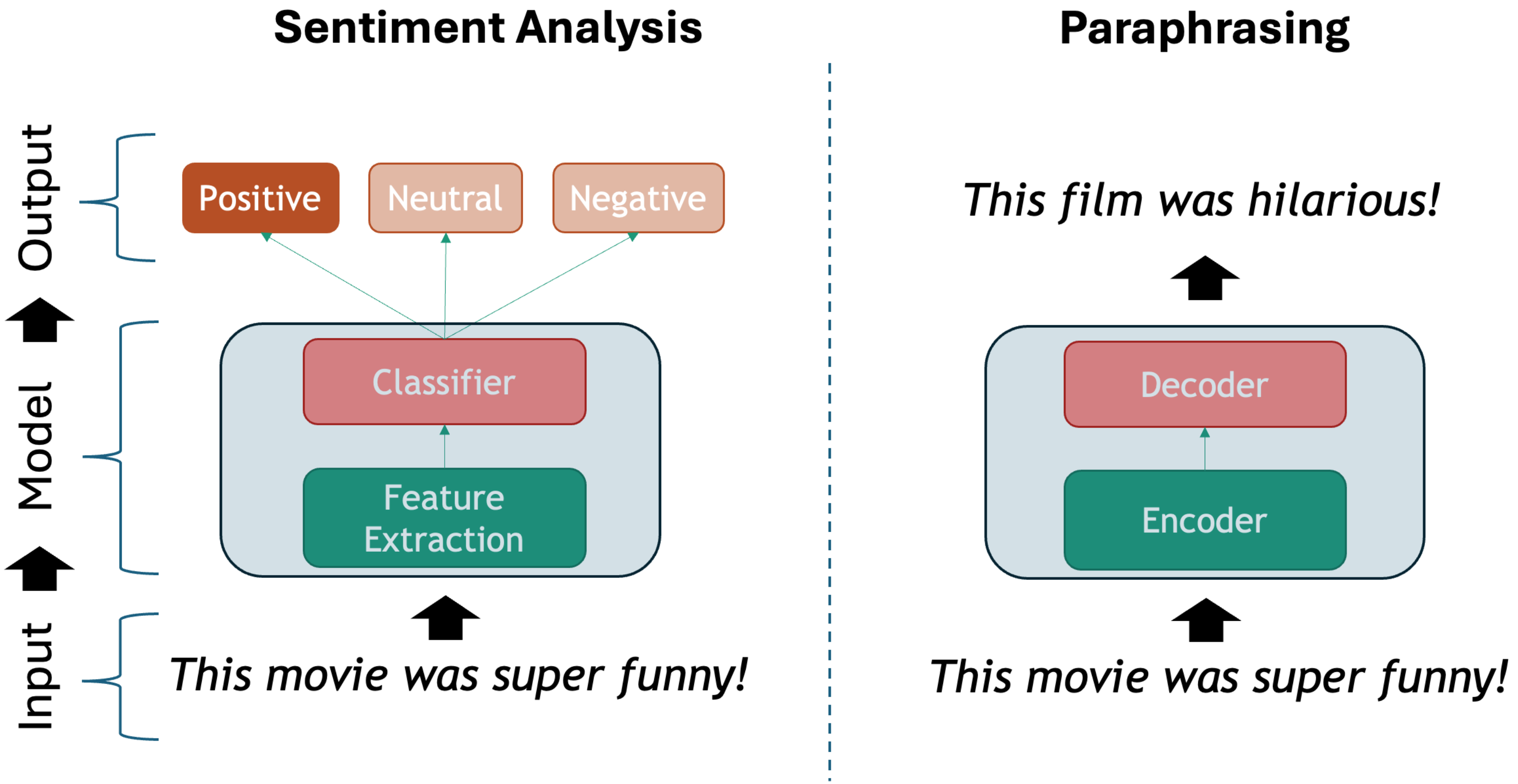
- Exploiting existing patterns from large amounts of text.

Supervised Learning

- Classify texts given a labeled set of examples
- The model learns from a manually labeled training set and *hopefully* generalizes to unseen examples

netherlands eScience center

Supervised Learning

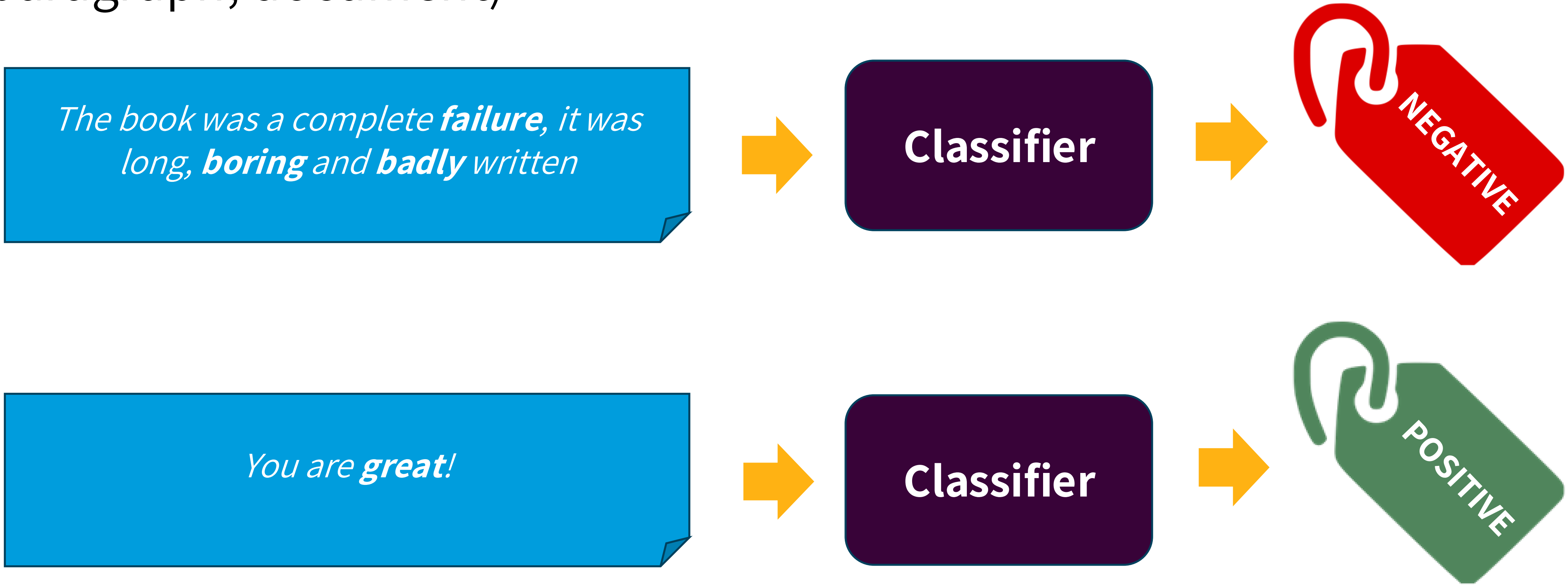


Natural Language Processing Tasks

- **Text Classification**
  - Language Identification
  - Span Filtering
  - Authorship Attribution
  - Sentiment Analysis
- **Token Classification**
  - POS Tagging
  - Word Sense Disambiguation
  - Named Entity Recognition
  - Co-reference Resolution
  - Entity Linking
- **Language Modeling**
  - Text Similarity
  - Plagiarism Detection
  - Document Clustering
  - Topic Modeling
  - Information Retrieval
- **Text Generation**
  - Machine Translation
  - Summarization
  - Paraphrasing
  - Question Answering
  - Chatbot

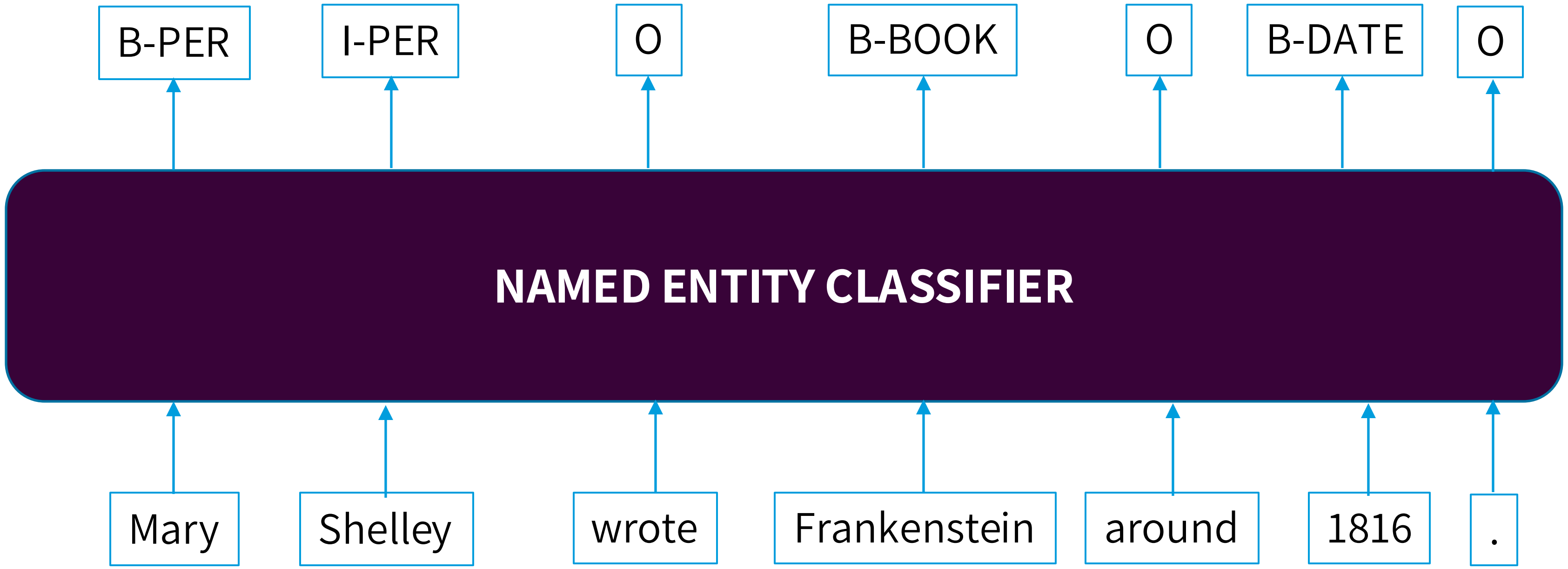
Text Classification

- A single label is assigned to a chunk of text (phrase, sentence, paragraph, document)



Token Classification

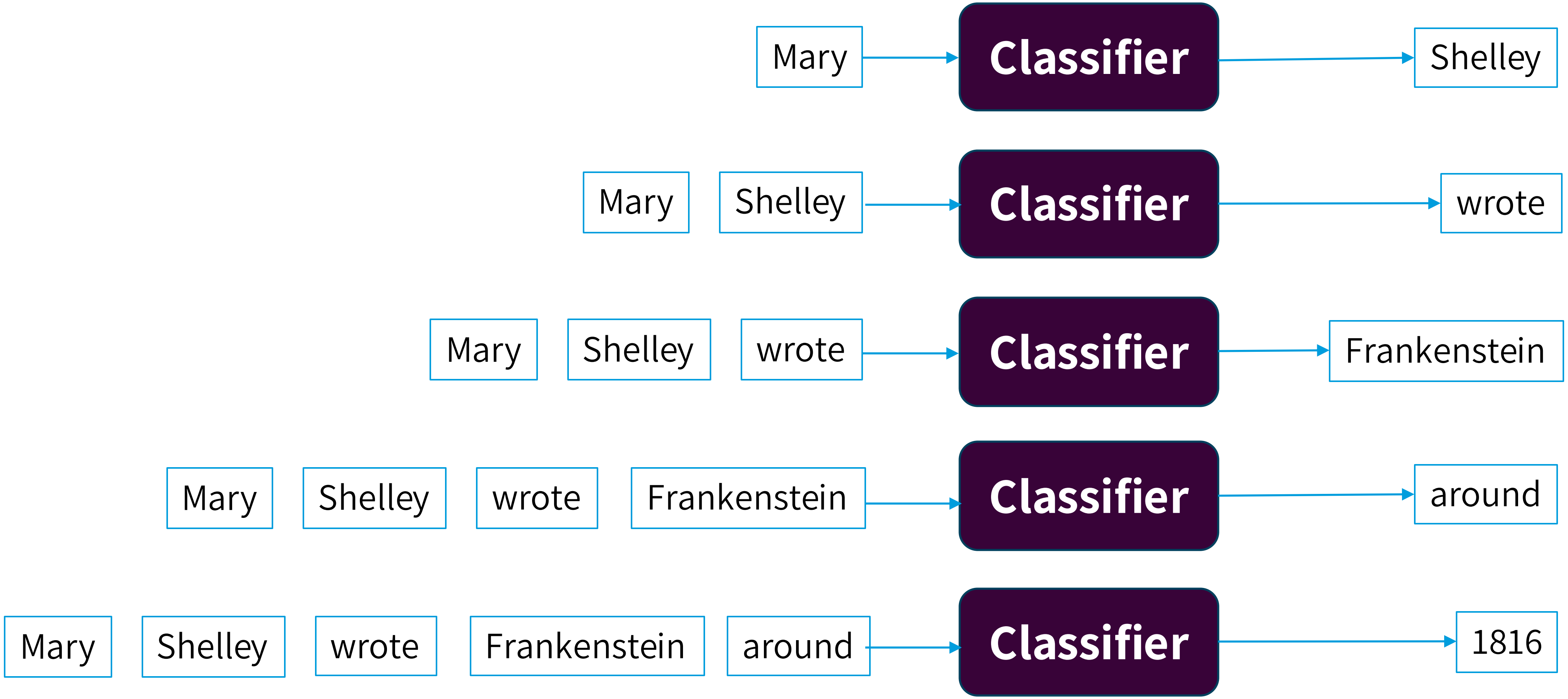
- One label per token is assigned. The whole surrounding sequence is considered to determine which label to assign.



len(Inputs) == len(Outputs)

Language Modeling

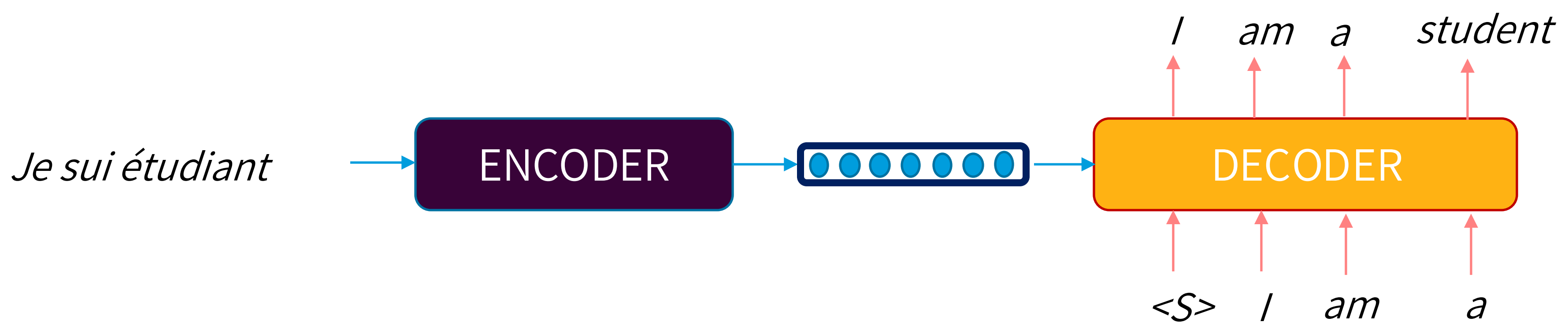
- Predicting the next word given a sequence of “history” words



Text Generation

- Learn to generate a target sequence given a source sequence and the context so far…

Parallel Corpus:
FR: *Je sui étudiant*
EN: *I am a student*



len(Inputs) != len(Outputs)

Challenge 3

- Look at the NLP Task taxonomy described above and write down a couple of examples of (Input, Output) instance pairs that you would need in order to train a supervised model for your chosen task.

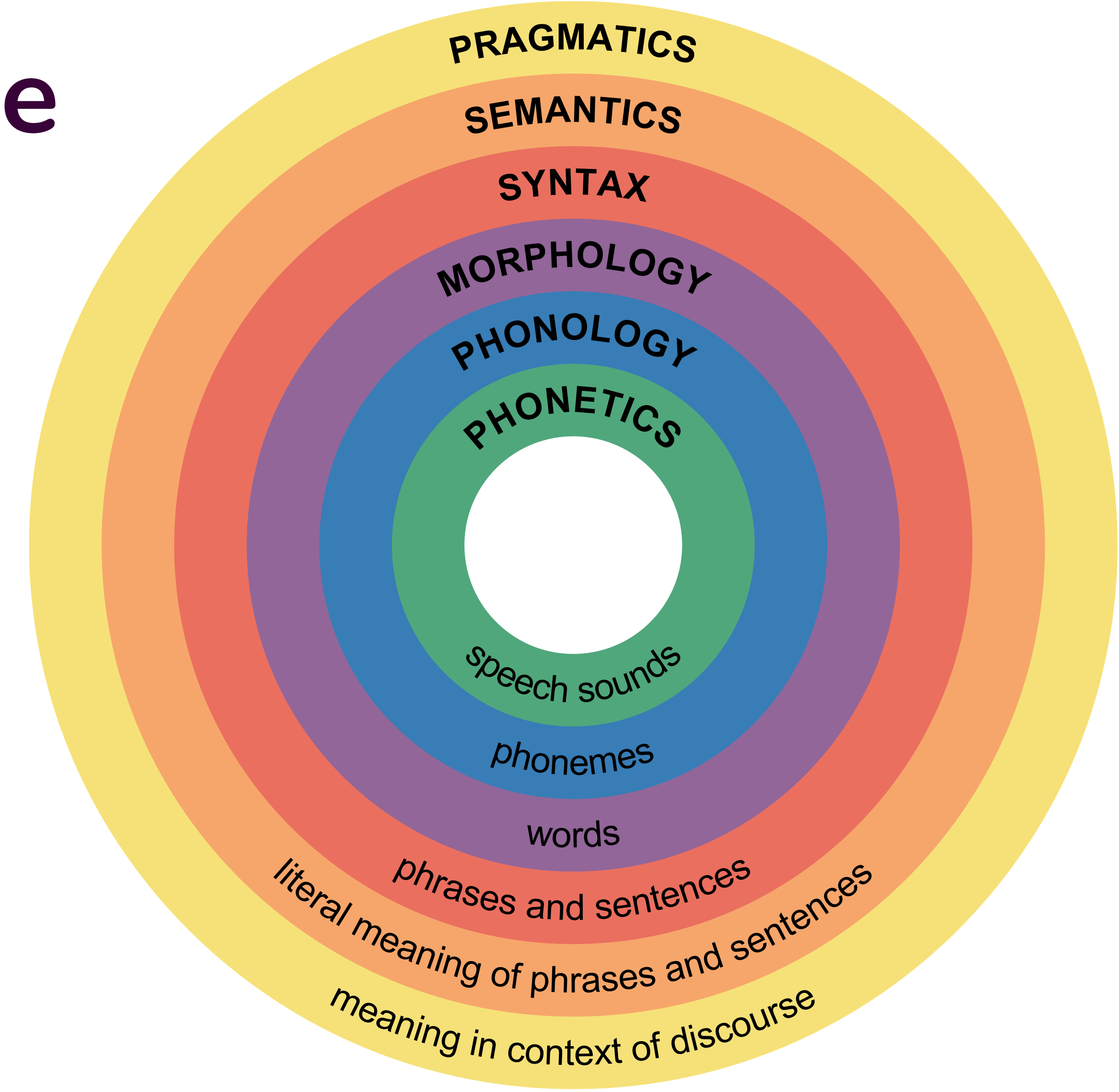
netherlands eScience center

A Primer on Linguistics

A primer on linguistics

- Language is hard to process because it is:
  - **Compositional:** structures are more than the additions of their parts
  - **Ambiguous:** the same symbols change meaning depending on the context
  - **Discrete:** words are arbitrary non-measurable symbols
  - **Sparse:** concrete phenomena occur only a handful of times

Levels of Language



Challenge 4 (In pairs)

- Discuss why the following sentences are special. To what level of language does each case belong?
  - *The door is **un-lockable** from the inside*
  - *Unfortunately, the cabinet is **unlock-able**, so we can't secure it*

  - *I saw the cat with the telescope*
  - *I saw the cat with the stripes*

  - *Please don't drive the cat to the vet!*
  - *Please don't drive the car tomorrow!*

  - *I **never** said she stole my money*
  - *I never said **she** stole my money*
  - *I never said she stole **my** money*

[CODE]

Exercises to see linguistic phenomena

Episode 02: From words to vectors

Preprocessing Operations

netherlands eScience center

Data Granularity

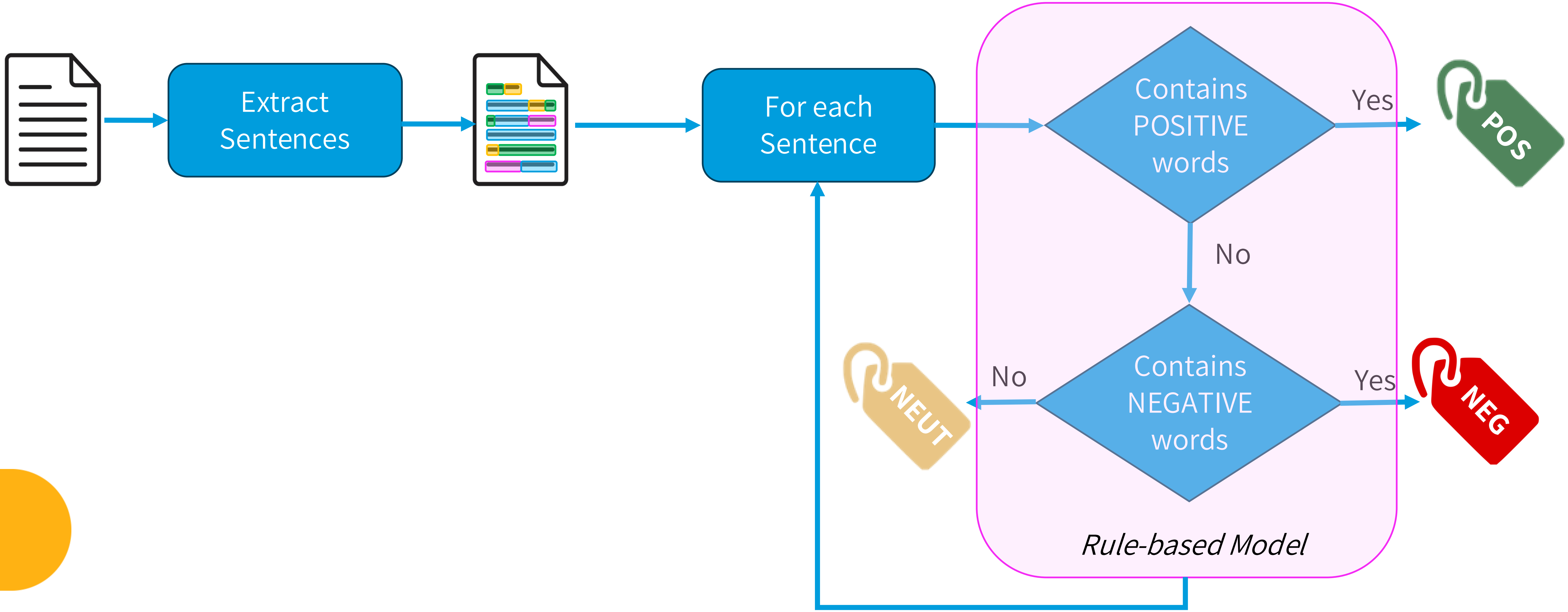
- **word level:** detecting abusive language (based on a known vocabulary).
- **sentence level:** extracting entities in each sentence, individually.
- **chunks of text:** (paragraphs or chapters), extracting key ideas (summarizing) each paragraph in a document.
- **document level:** For example, each full book should have one genre tag (Romance, History, Poetry).

Common Operations

- **Data Formatting:**
  - Extract text strings from documents formats (word, pdf, etc…)
  - Remove unwanted content
  - *Clean* your text as much as possible.
- **Tokenization:** extract words or sentences.
- **Lowercasing:** make the text more homogeneous.
- **Lemmatizing:** keep only *dictionary-entry* words.
- **Stopword Removal:** get rid of non-interesting words.

NLP Pipeline

Example: Rule-based Classifier



Challenge 1

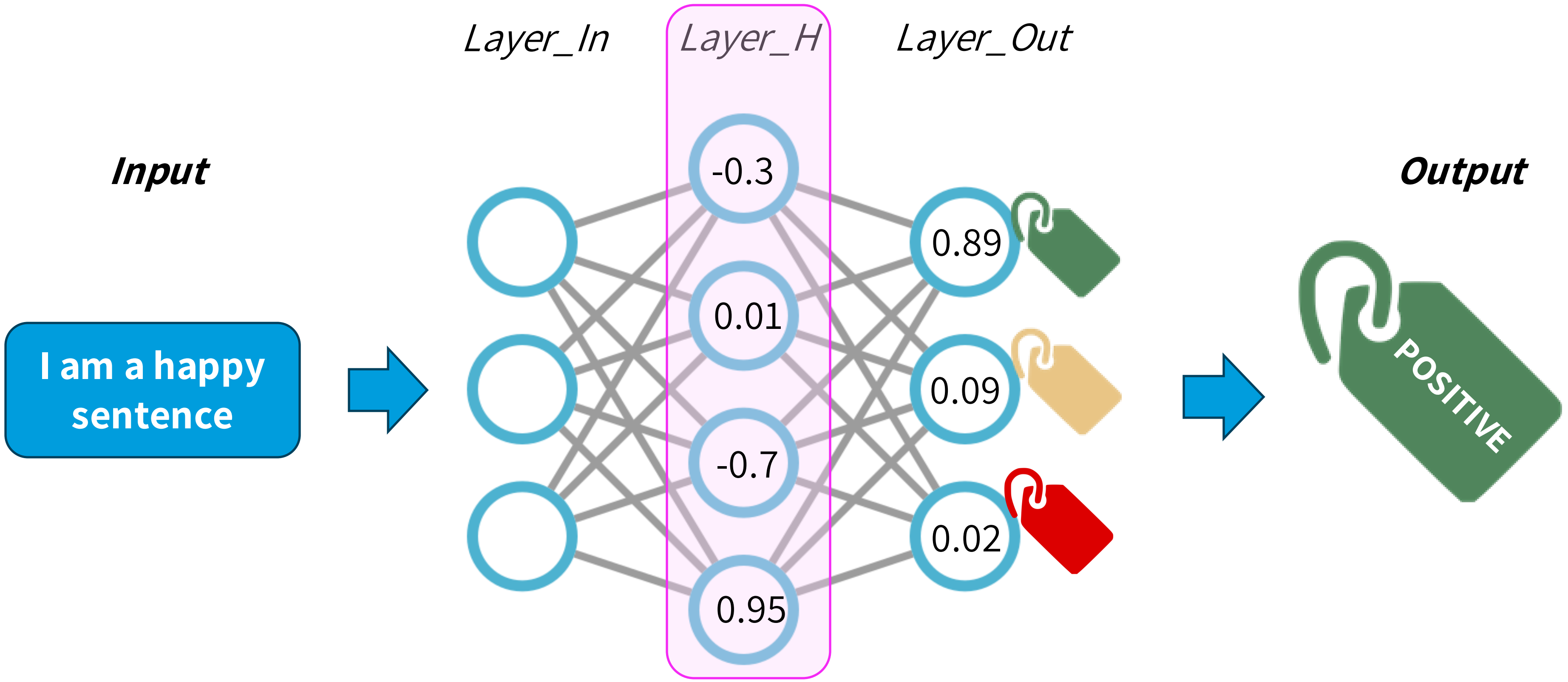
Think about the pros and cons of the proposed NLP pipeline:

- Do you think it will give accurate results?
- What do you think about the coverage of this approach? What cases will it miss?
- Think of possible drawbacks of chaining components in a pipeline.

netherlands eScience center

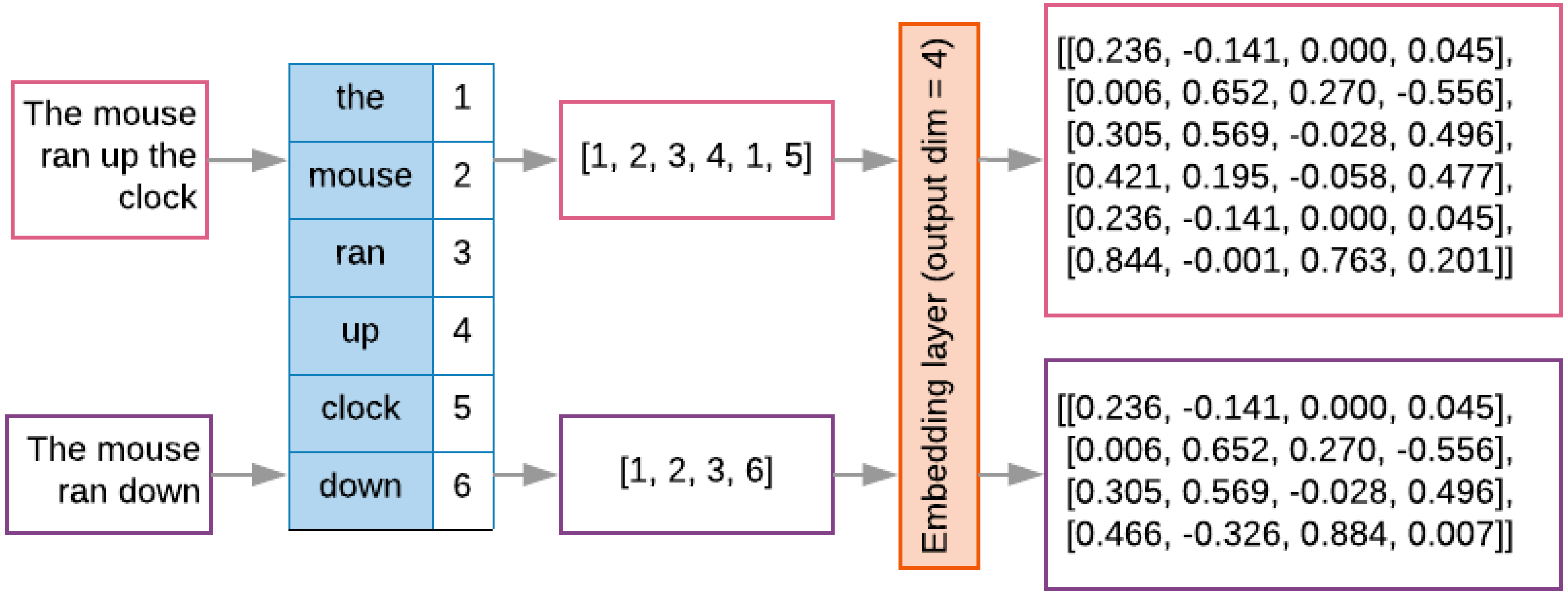
Word Embeddings

Neural Networks



After seeing thousands of examples, each layer represents different "features" that maximize the success of the task, but they are not human-readable

Word Embeddings Layer



Source: https://developers.google.com/machine-learning/guides/text-classification/images/EmbeddingLayer.png

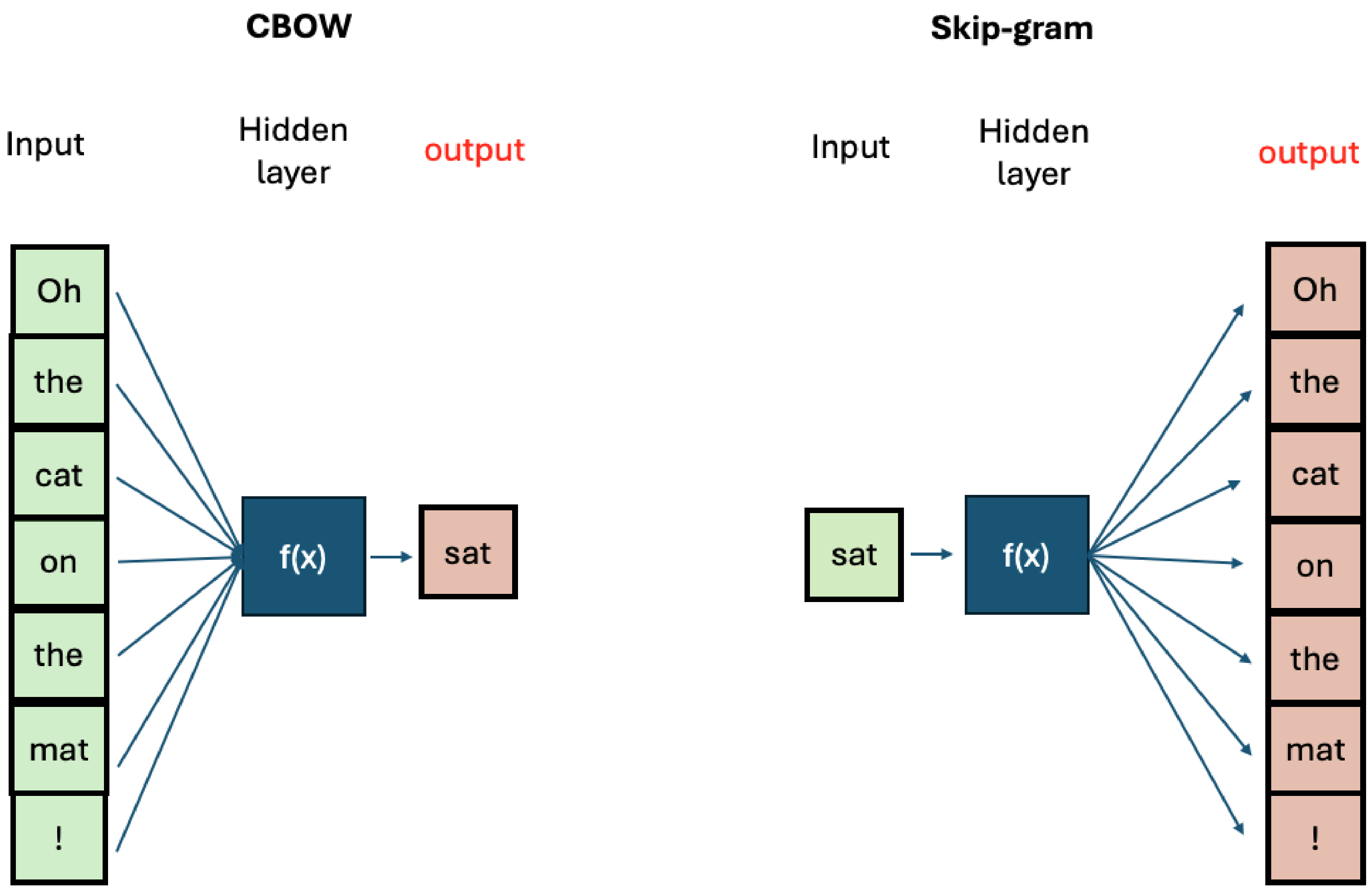
netherlands eScience center

Training Data Word2Vec

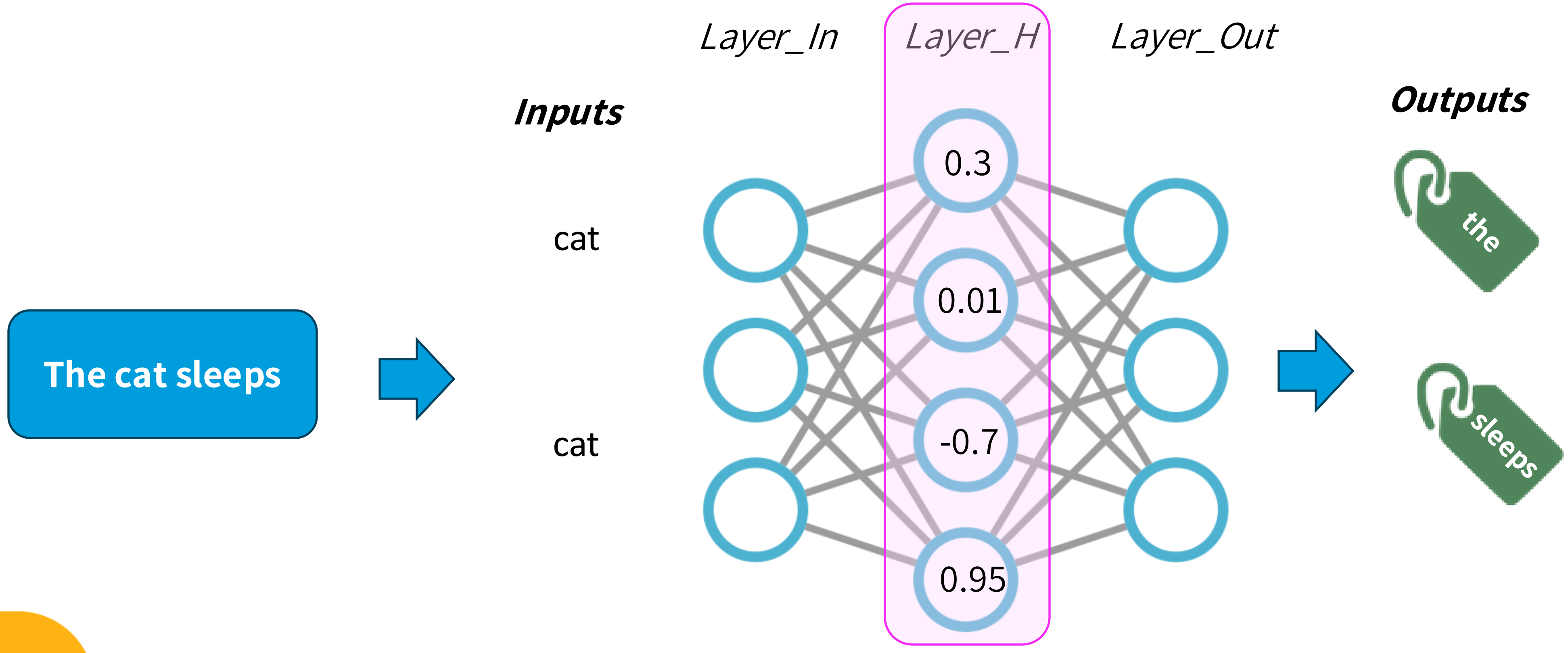
Source Text		Training Samples
The quick brown fox jumps over the lazy dog.	⟹	(the, quick) (the, brown)
The **quick** brown fox jumps over the lazy dog.	⟹	(quick, the) (quick, brown) (quick, fox)
The quick **brown** fox jumps over the lazy dog.	⟹	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown **fox** jumps over the lazy dog.	⟹	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

Source: https://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Training Algorithms Word2Vec



Word2Vec Neural Network



The NN maps a source word into a target word from the original context. After being trained, we are just interested in the hidden layer that holds the pre-trained Word Embeddings

[CODE]

Explore Word2Vec Space

Challenge 2

- Think of different word pairs and try to guess how close or distant they will be from each other.
- Use the similarity measure from the word2vec module to compute the metric and discuss if this fits your expectations. If not, can you come up with a reason why this was not the case?

[CODE]

Train your own Word2Vec

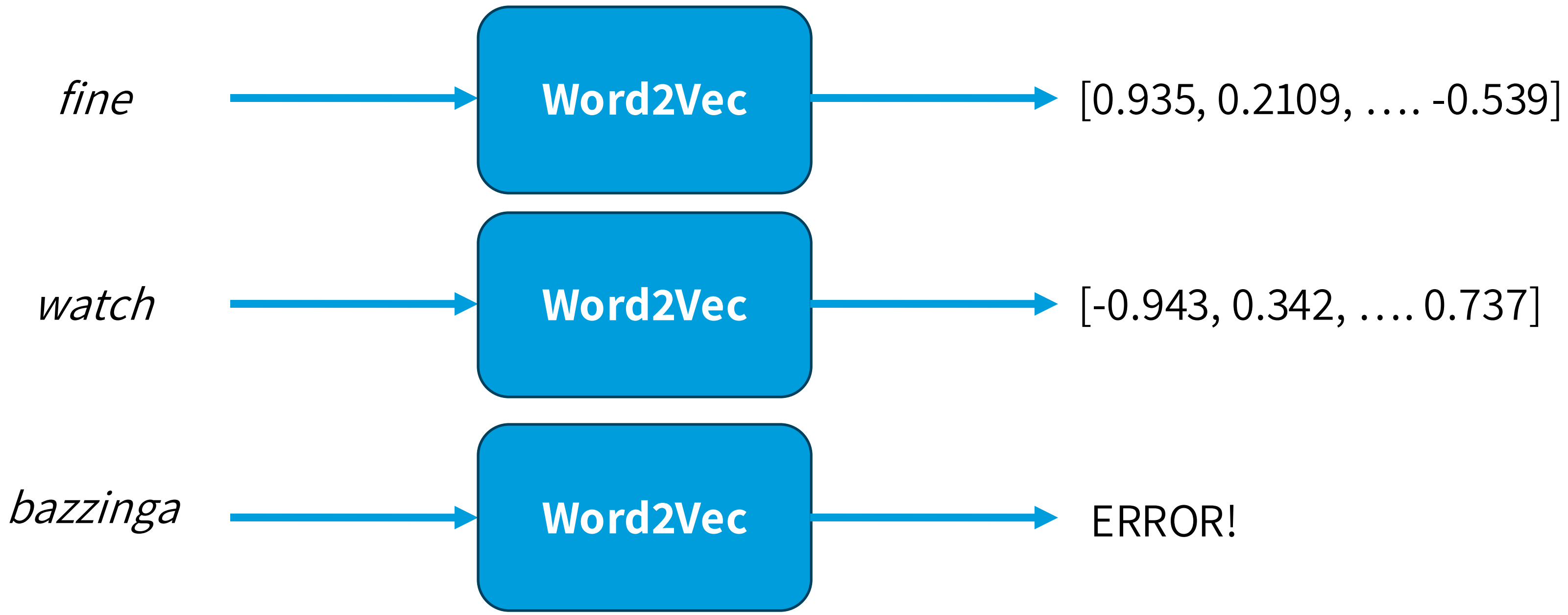
netherlands eScience center

Episode 03: Transformers & BERT

Transformers

Word2Vec Limitations

- Each word is processed in isolation
- Fixed Vocabulary Size (e.g. 10K words)
- Fixed Vector per item in vocabulary

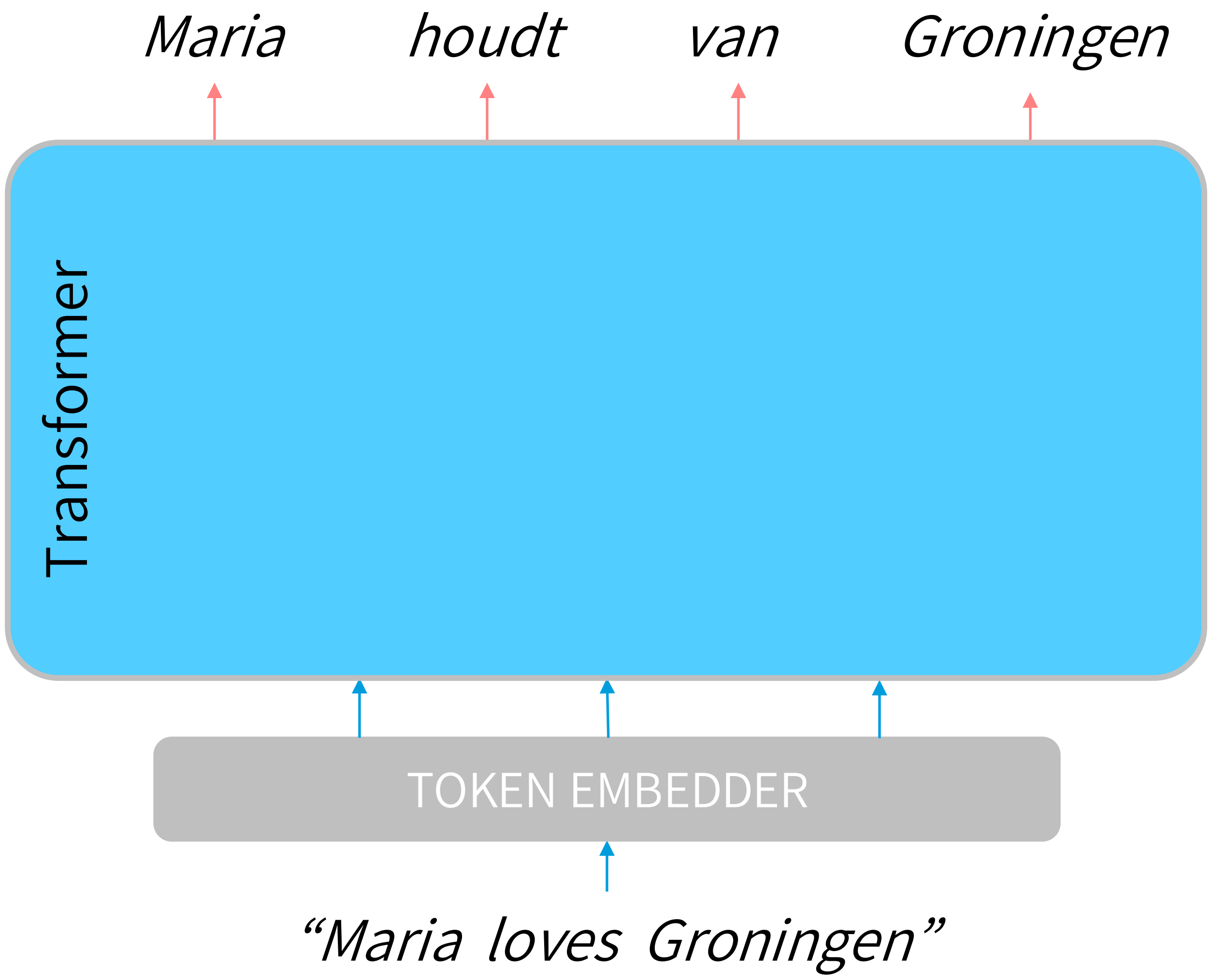


netherlands eScience center

Challenge 1

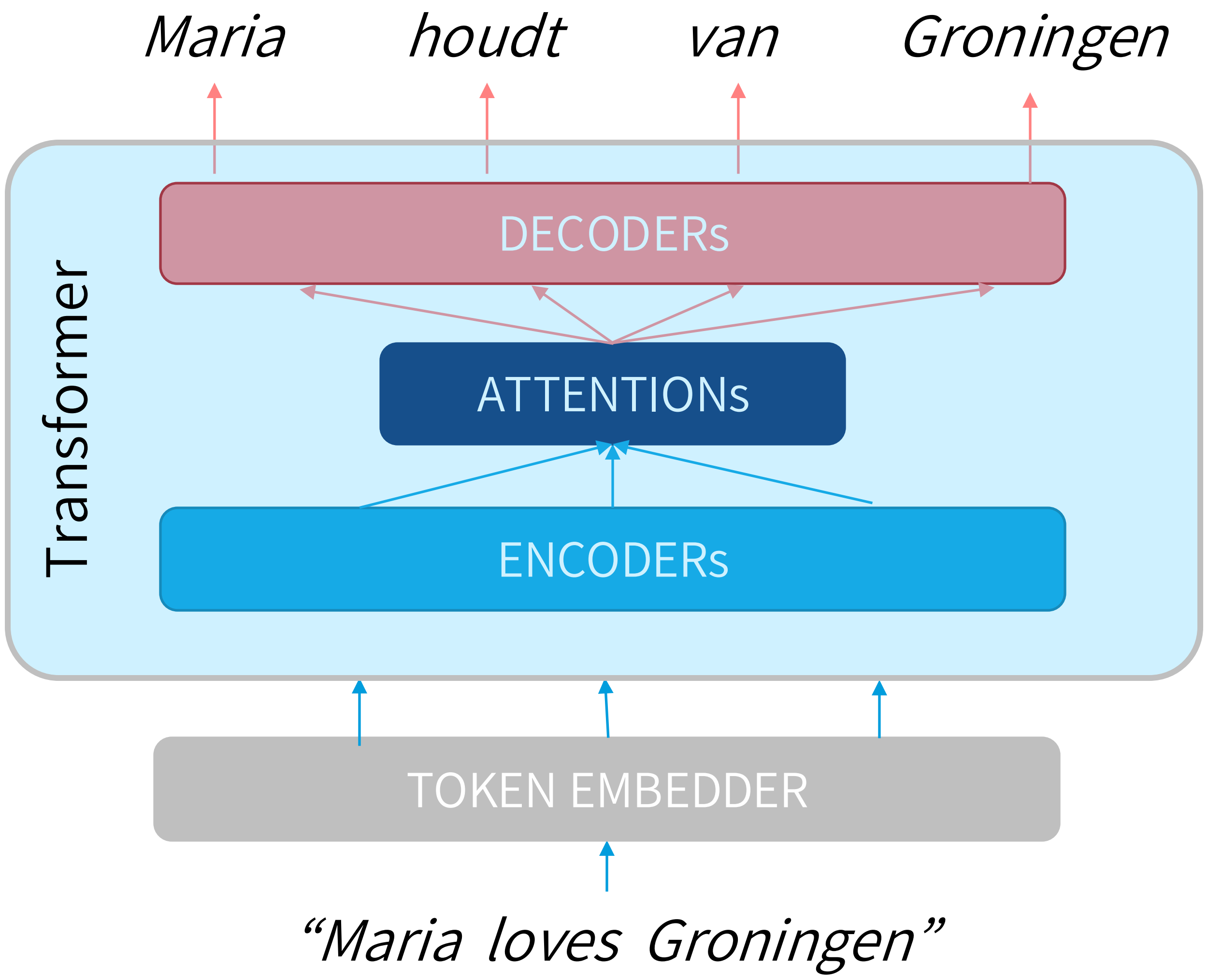
- Think of (at least 2) different words that can have more than one meaning depending on the context.
- Come up with one simple sentence per specific meaning and explain what they mean in each context.
- How do you know what of the possible meanings does the word have when you use it?

The Transformer



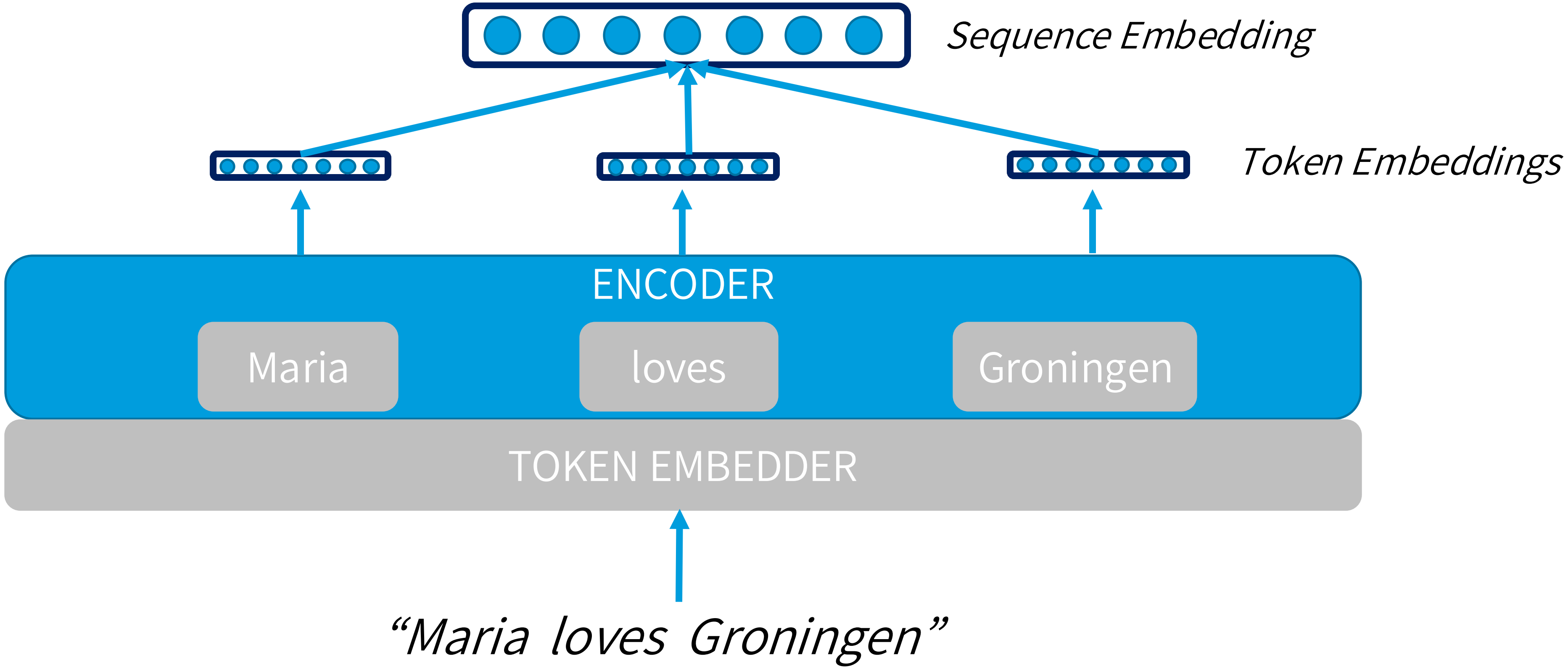
netherlands eScience center

The Transformer



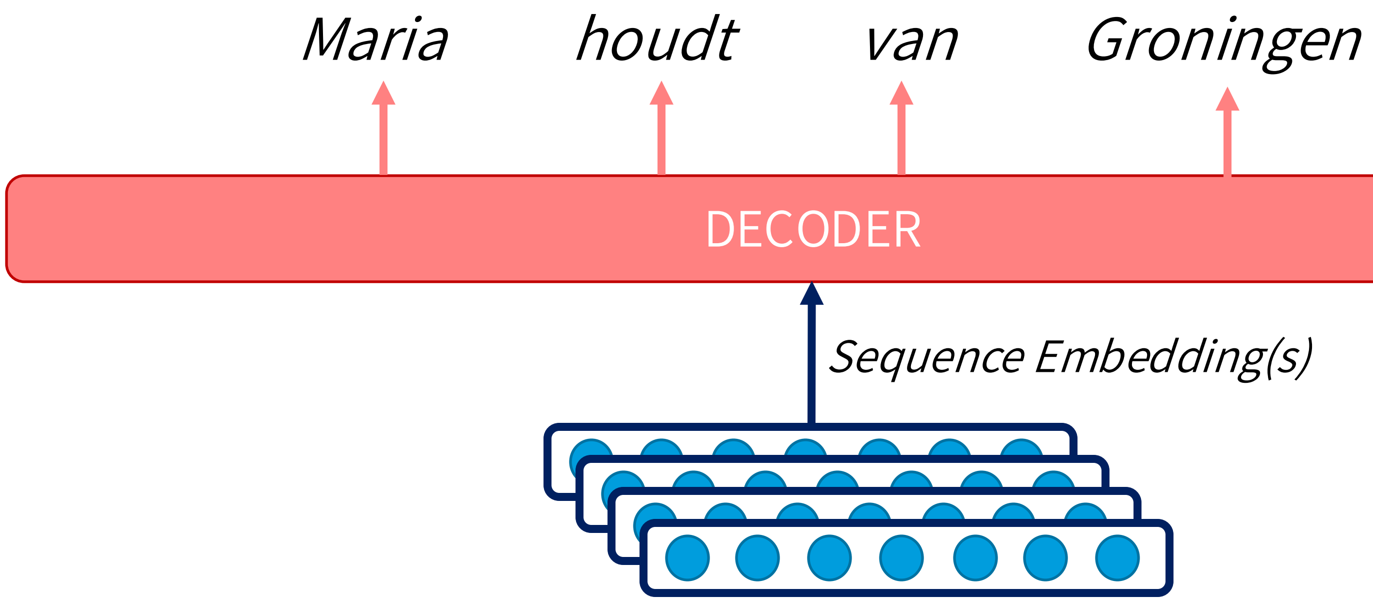
Encoder

- The encoder is trained to compress any sequence of tokens into a **fixed-size vector**.



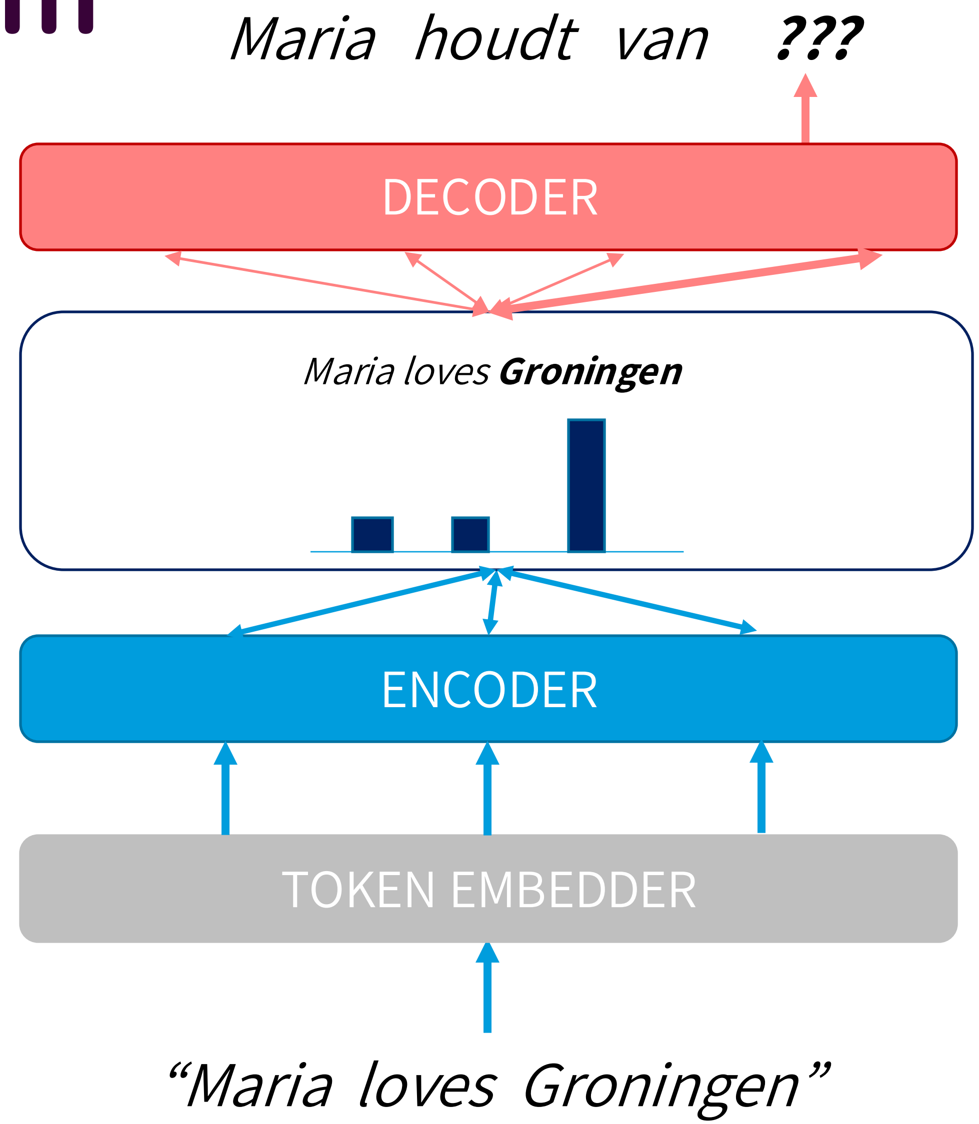
Decoder

- The decoder is the **generative model.** Emits one token at a time, conditioned on the compressed input + previous emitted tokens



Attention Mechanism

- With attention, the decoder has the opportunity to *double check into* the source each time it emits the next token
- Attention learns relevance of components across source and target sequences



Challenge 2 (In pairs)

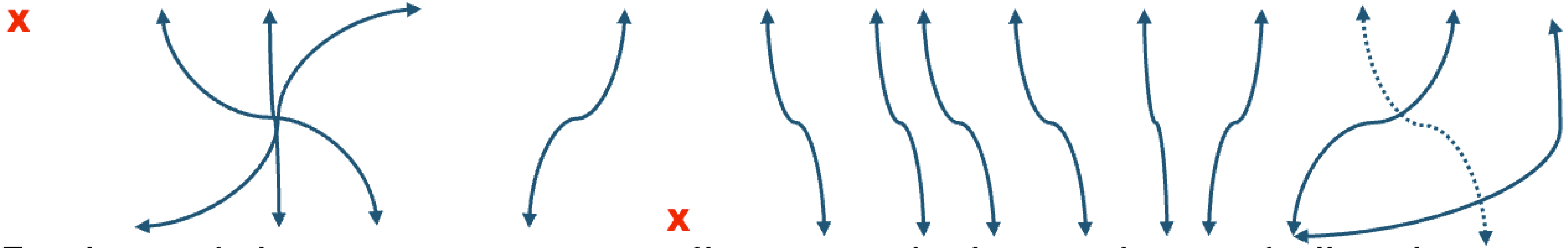
- In pairs, think of 1 short but interesting sentence in English (*langA*) and come up with its translation into a second language (*langB*):

- Draw arrows between the words or phrases from *langB* to *langA* that mean the same. Is it always possible to do this mapping one to one?

- How does this relate to attention?

Challenge 2: Solution

El monstruo de Frankenstein nunca aparece en la novela como una bestia sin razón



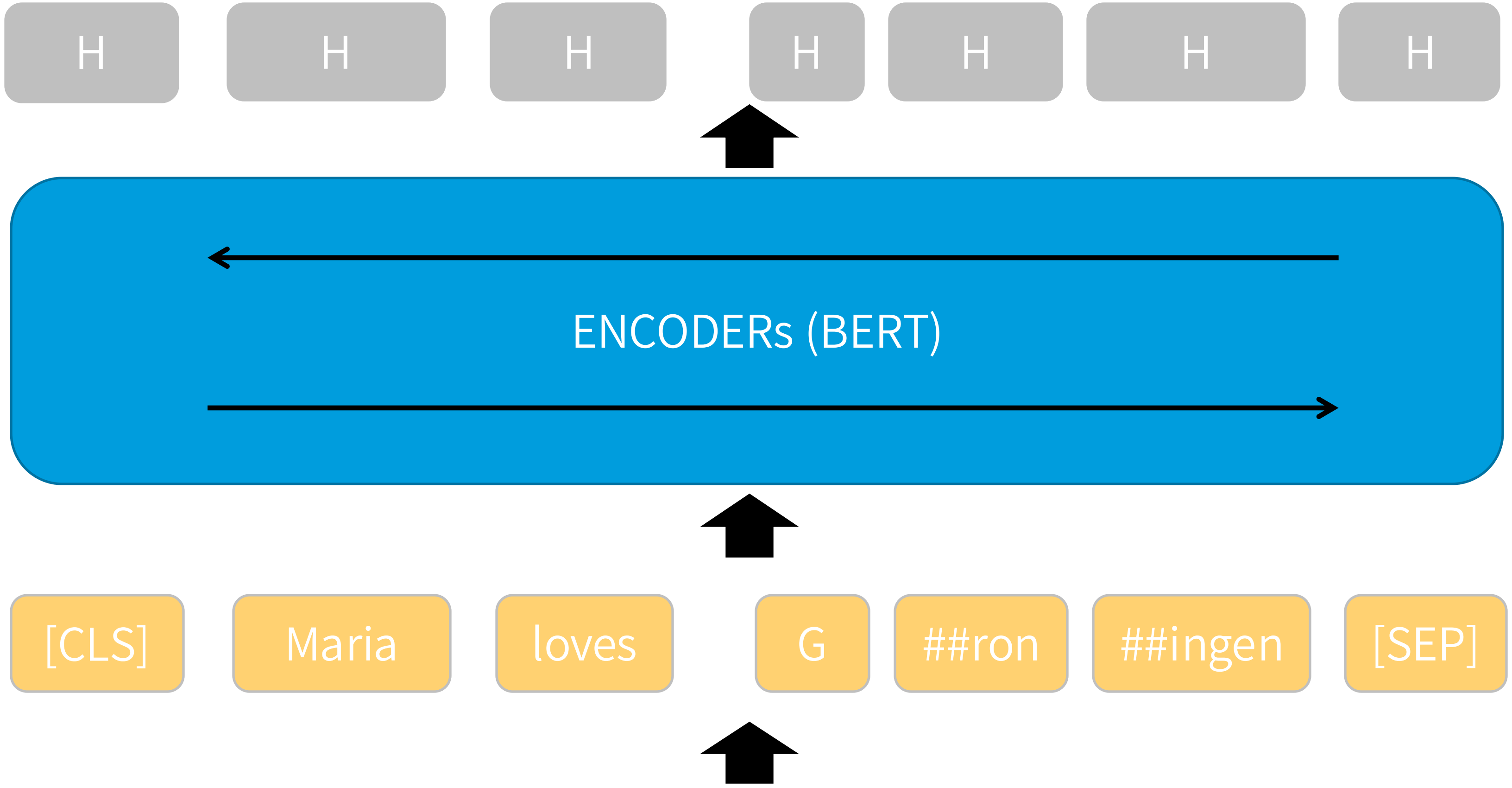
Frankenstein 's monster never actually appears in the novel as a mindless brute.

BERT

netherlands eScience center

BERT Architecture

Example:
BERT-base



Each token vector has 768 dimensions

12 Encoders are s tacked together

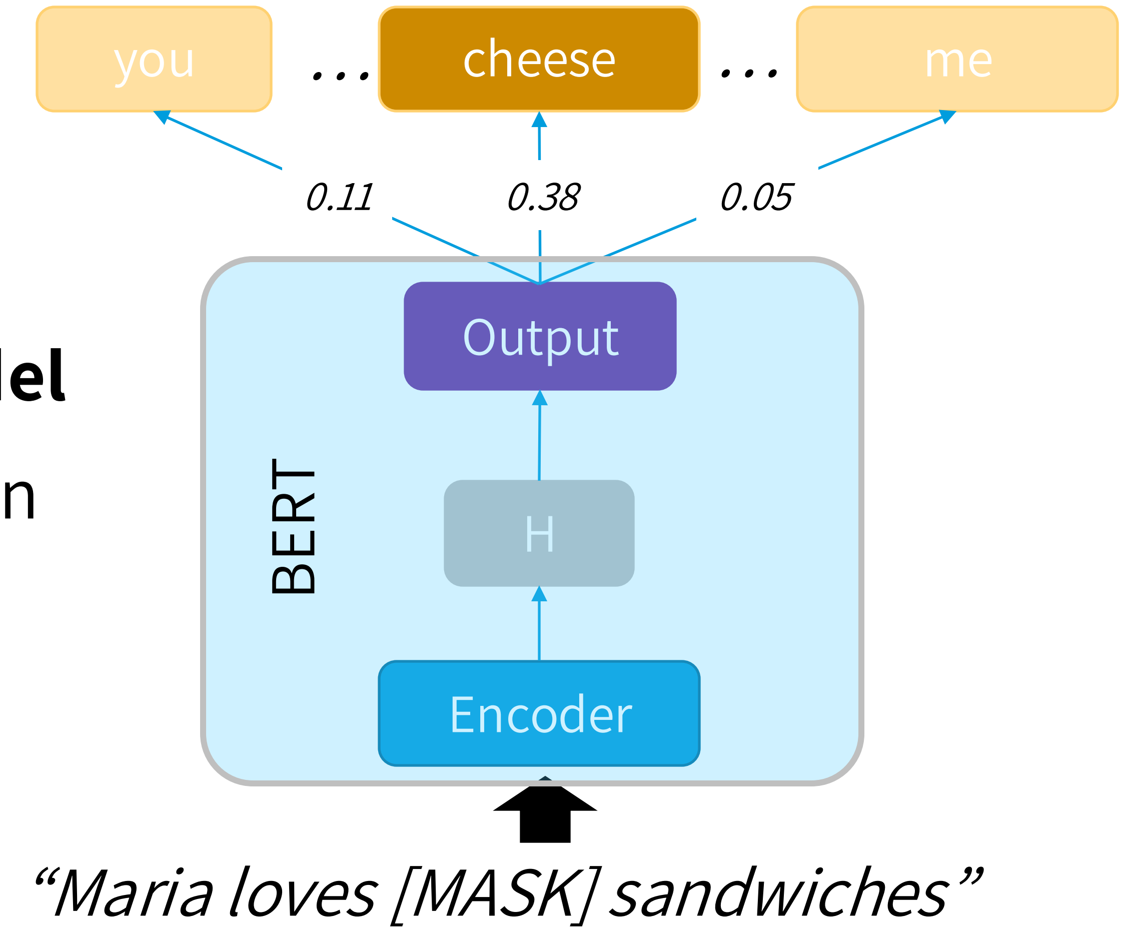
It's input size is always 512 tokens

"Maria loves Groningen"

netherlands eScience center

Training BERT

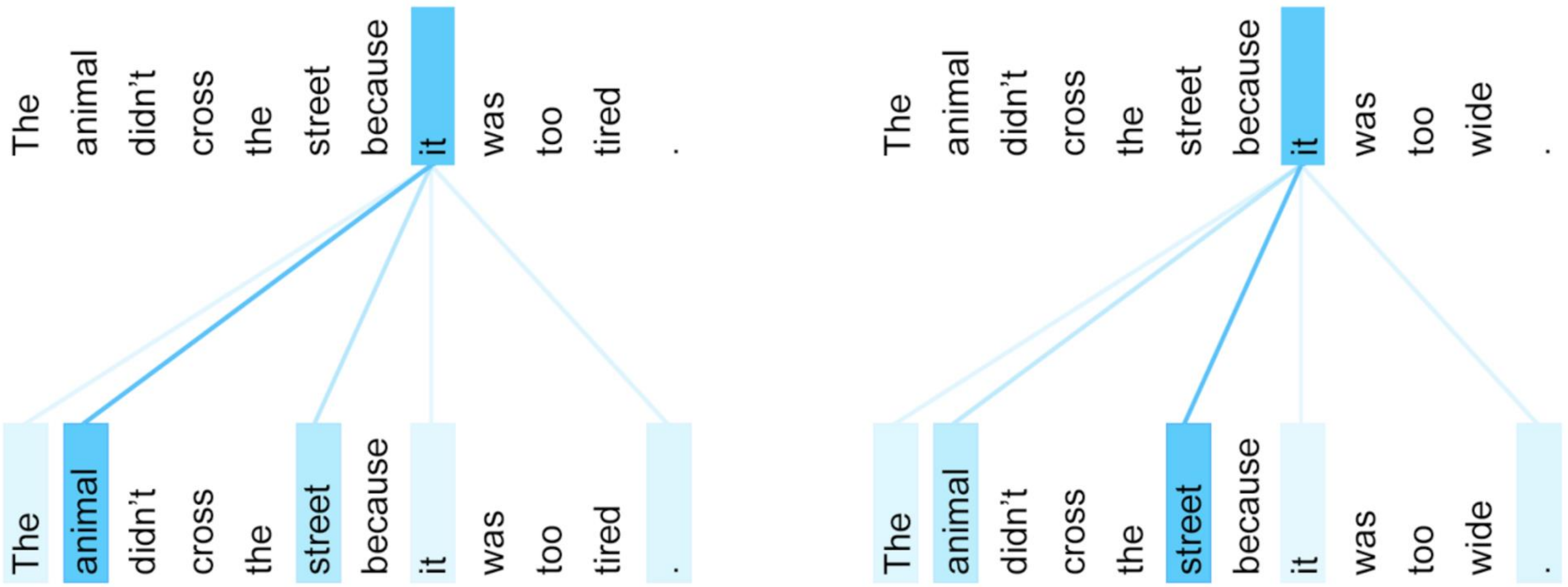
- **TASK1: Masked Language Model**
- TASK2: Next Sentence Prediction



netherlands eScience center

BERT: Self-Attention

- The BERT encoder has an internal attention component, which enhances contextual learning.



Example: Coreference Resolution

[CODE]

BERT Tokenization: Word Pieces

[CODE]

Polysemy in BERT: Obtain contextualized vectors

[CODE]

BERT as Language Model: Fill-in the blank!

Challenge 3

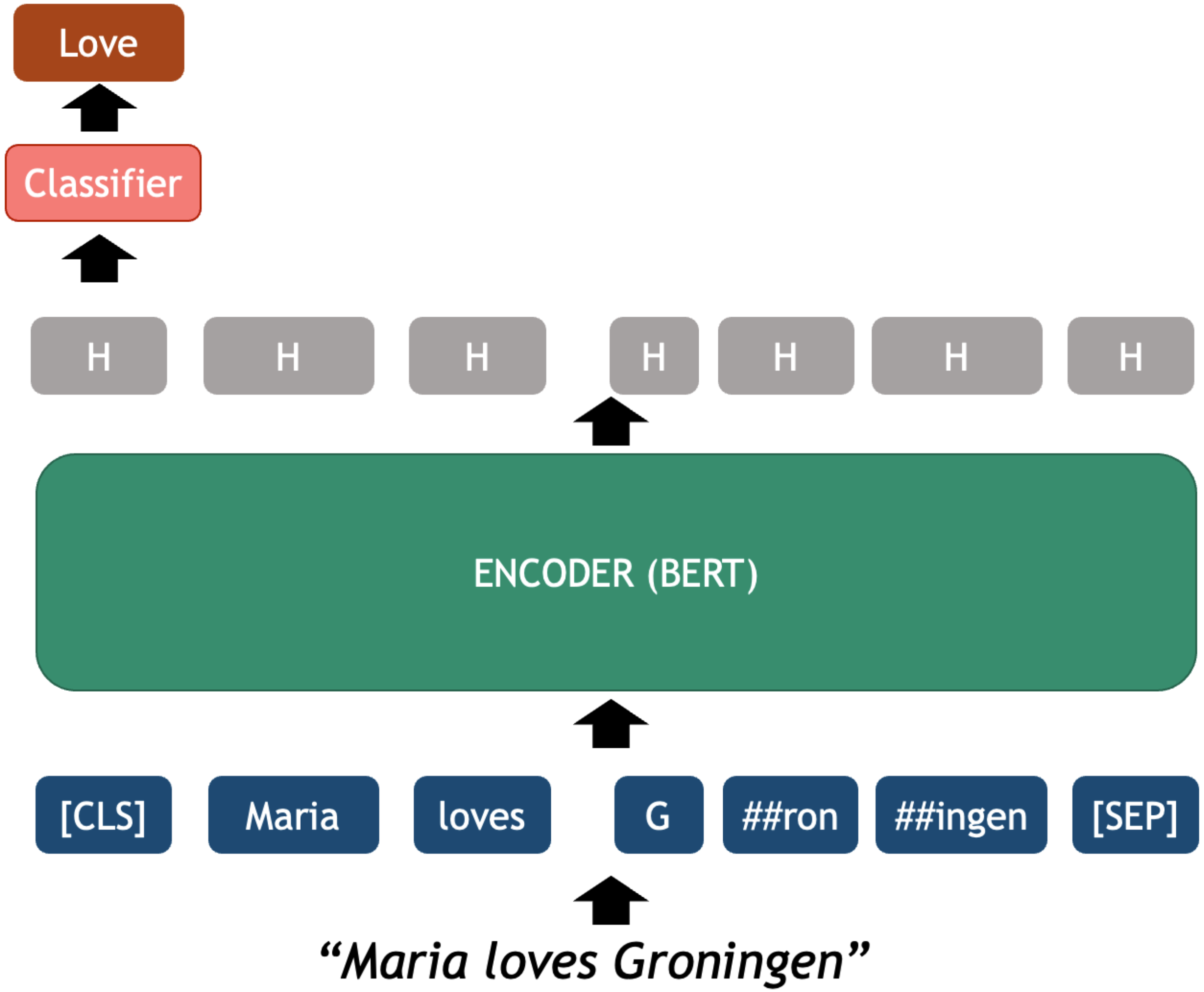
- Play with the `fill-mask` predictions
- Intuitively try to find good and bad prediction examples
- You can test different `top_k` parameters and/or use the multilingual model `bert-base-multilingual-cased`
- Can you explain the intuitions behind the predictions?

HuggingFace Pipeline

- The pipeline() is very useful to test models,
- It supports dozens of tasks, including multimodal models.
- Careful: It wraps different model architectures!
- If you need more flexibility you will have to use the tokenizer and model inference separately.

See [pipeline() docs]()

BERT for Text Classification



[CODE]

BERT-based Sentiment Classifier

netherlands eScience center

Model Evaluation

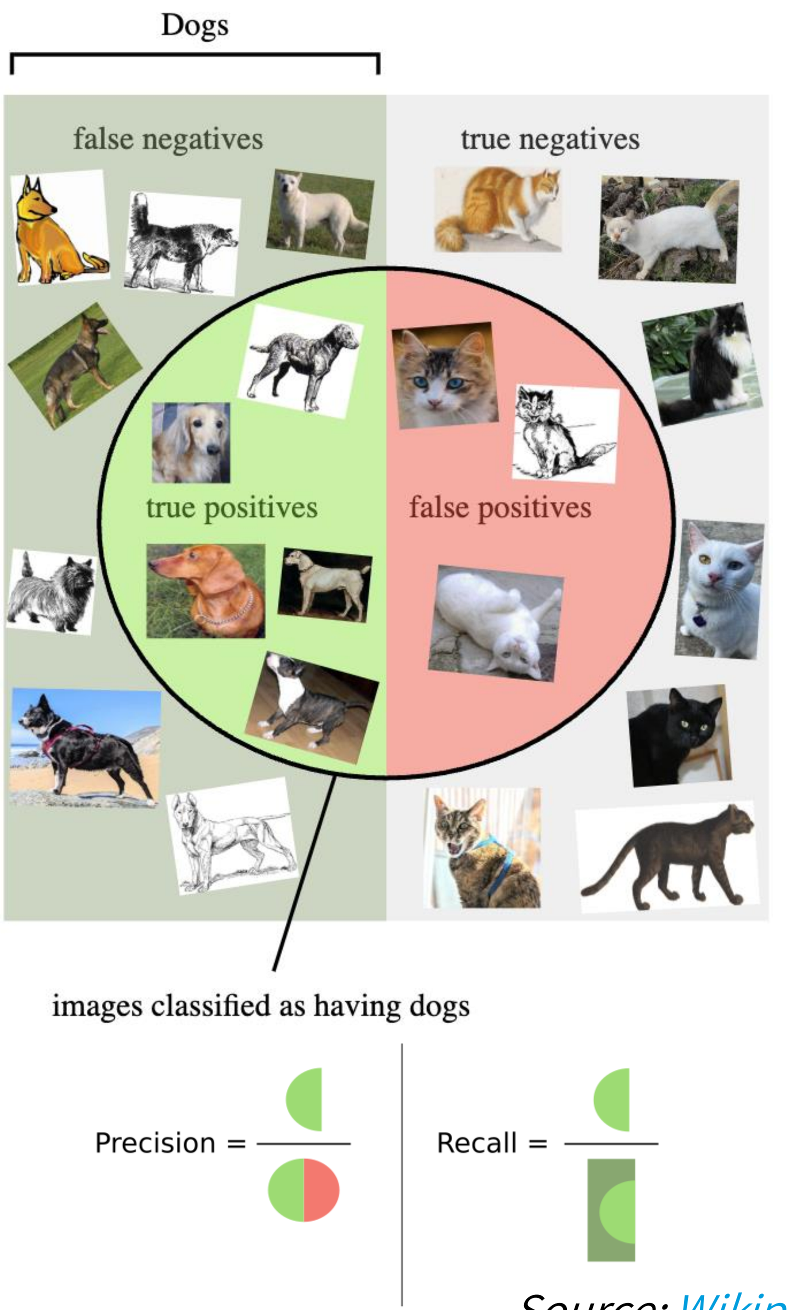
netherlands eScience center

Model Evaluation

- The purpose of **evaluation metrics** is to understand how well the model generalizes to unseen cases.
- Why Evaluate supervised models (classification)?
  - If you trained your own model, you should measure performance on a held-out test set, to **avoid overfitting.**
  - If you use a pre-trained model, it **prevents overconfidence** on external performance claims.
- The most basic metric is **accuracy**, *how many times did the model got it right?* But it is often not enough…

Metrics for Classifiers

- Each metric measures performance for each class independently
- **Precision**: Of predictions made, how many were correct?
- **Recall**: Of actual cases, how many did we catch?
- **F1-Score**: “Balance” between precision and recall

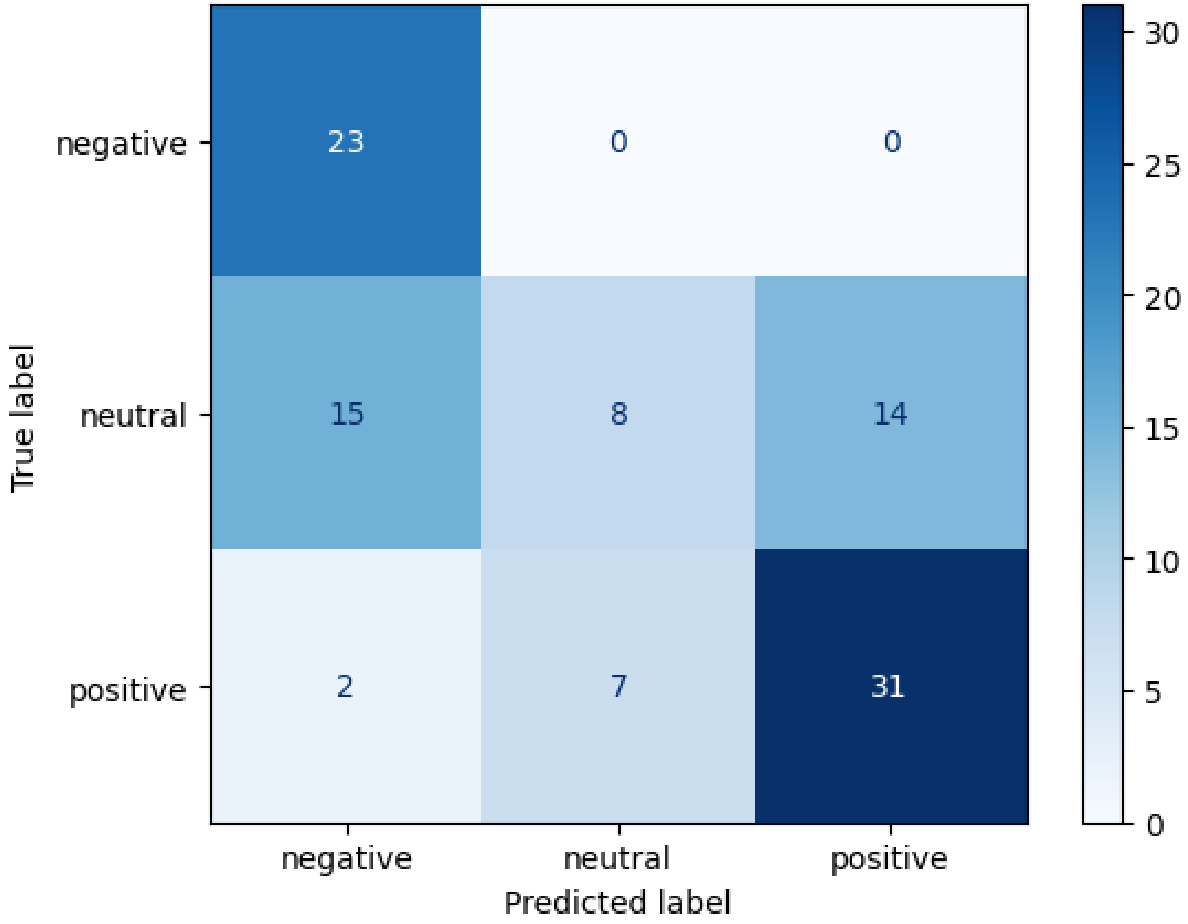


Source: Wikipedia

netherlands eScience center

Confusion Matrix

- Basic observability to model predictions
- Identify biases for specific label classes. *Which classes your model predict well vs. poorly?*
- Take steps to improve performance. *Which classes need more training data?*



Challenge 4

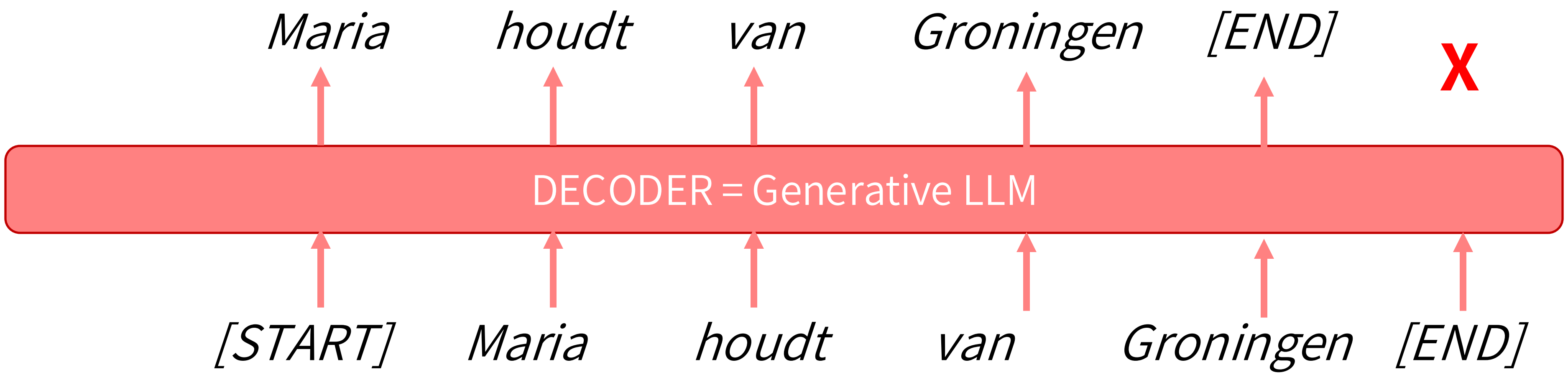
- Use the provided file to evaluate a sentiment classifier
- **File:** sentiment_film_data.tsv
- Use this **model:** "tabularisai/multilingual-sentiment-analysis"

Episode 04: Large Language Models (LLMs)

What are LLMs?

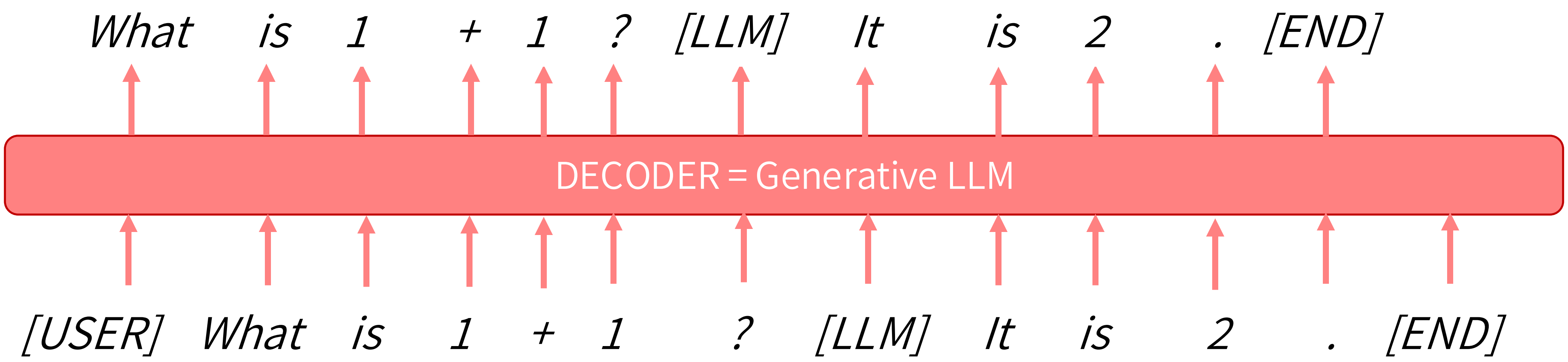
- There is no single definition for this term. Here we will define it as:
- Large language models (LLMs) are generative transformer-based language models that are trained to interact in a conversational-like manner with humans.
- The term *Large* refers to the amount of parameters and training data compared to previous Transformers.
- They do not only predict the next most likely token (LM) but have been trained to follow instructions (chat assistant).

Are LLMs "Generative AI"?



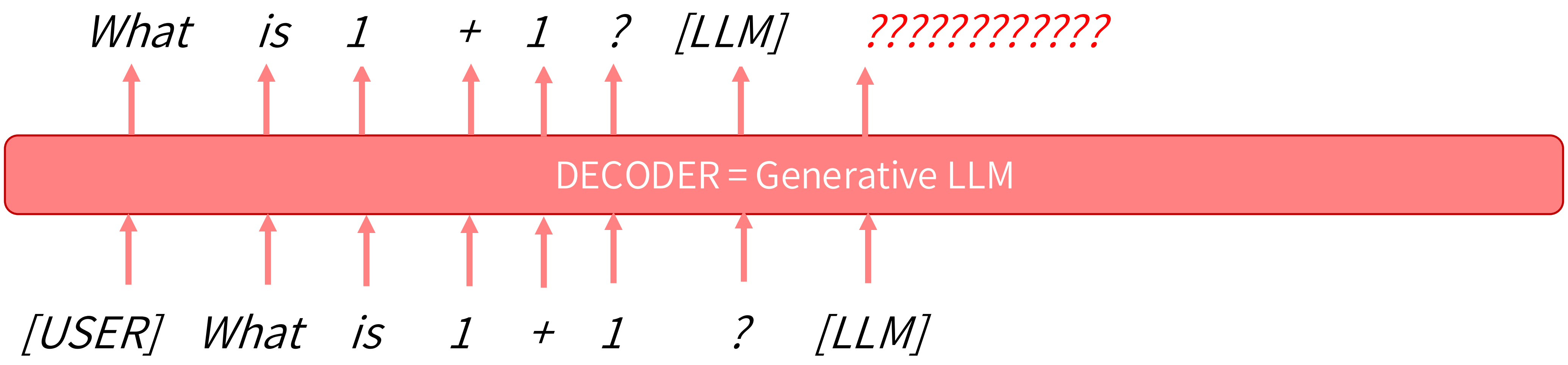
netherlands eScience center

Are LLMs "Generative AI"?

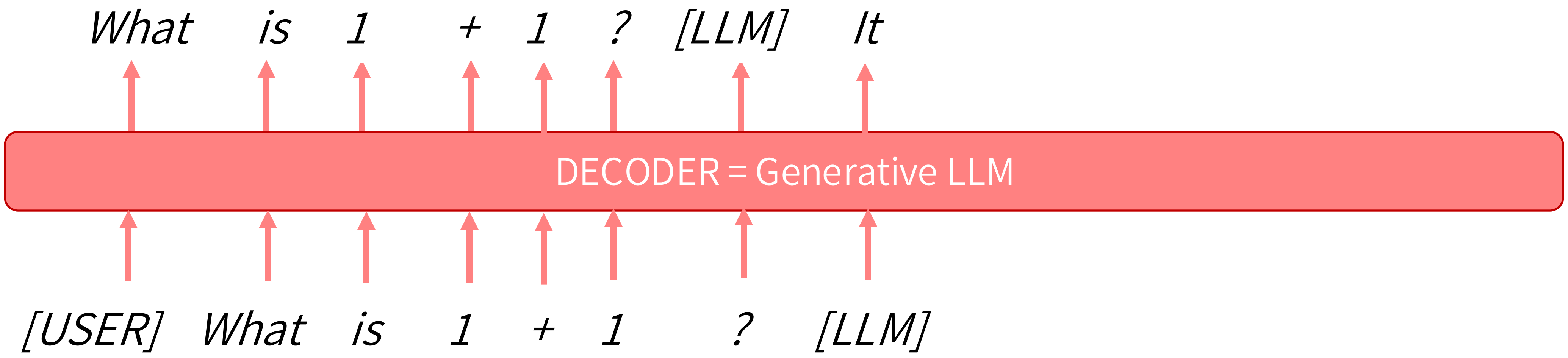


netherlands eScience center

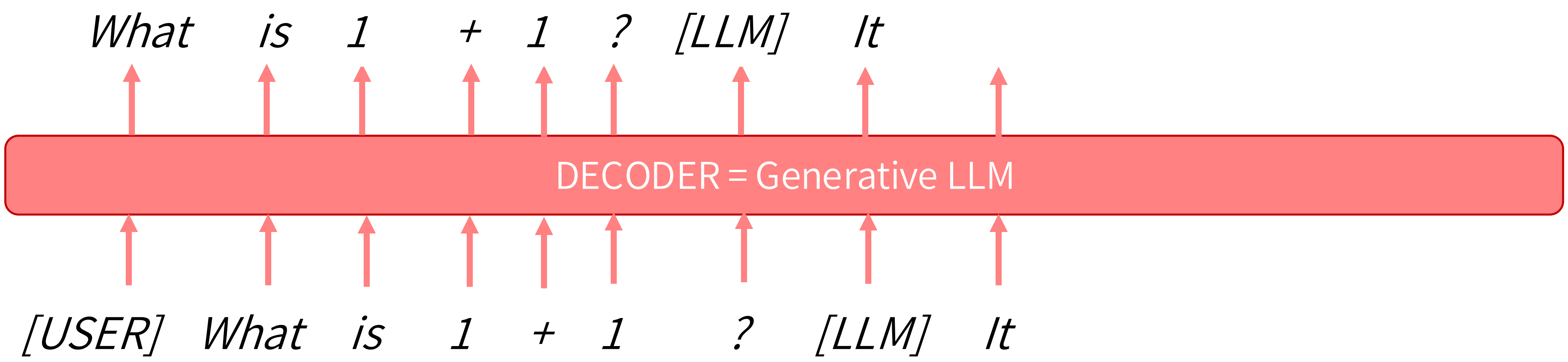
Are LLMs “Generative AI”?



Are LLMs “Generative AI”?

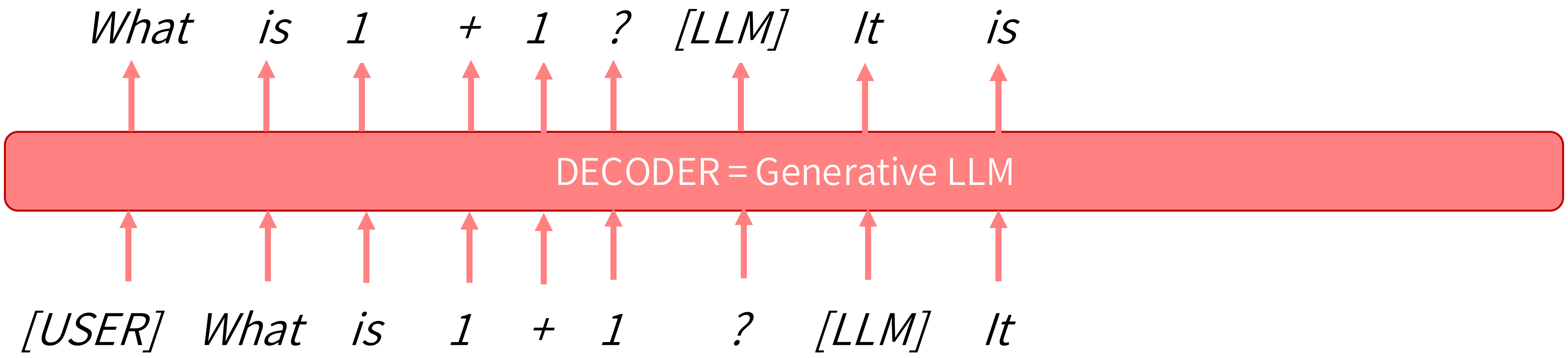


Are LLMs “Generative AI”?

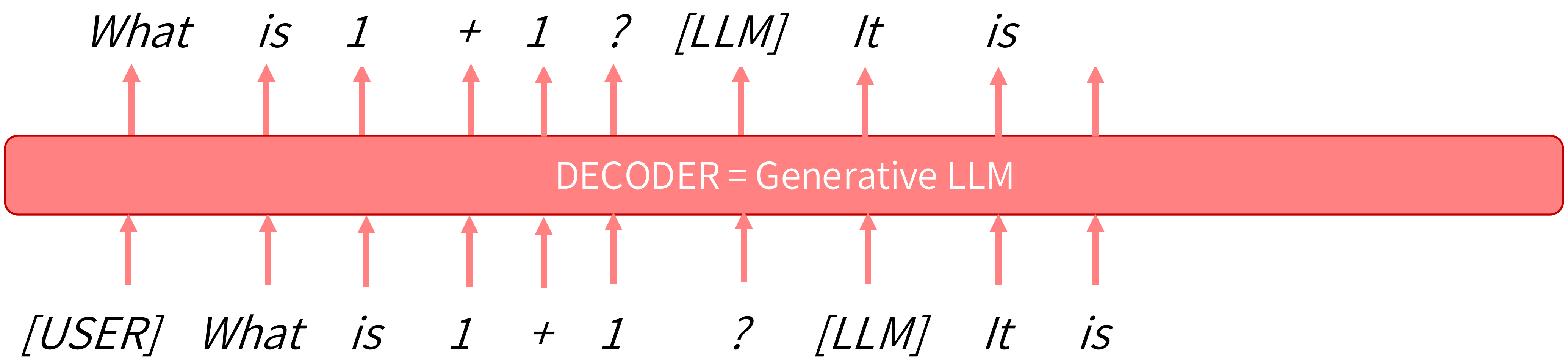


netherlands eScience center

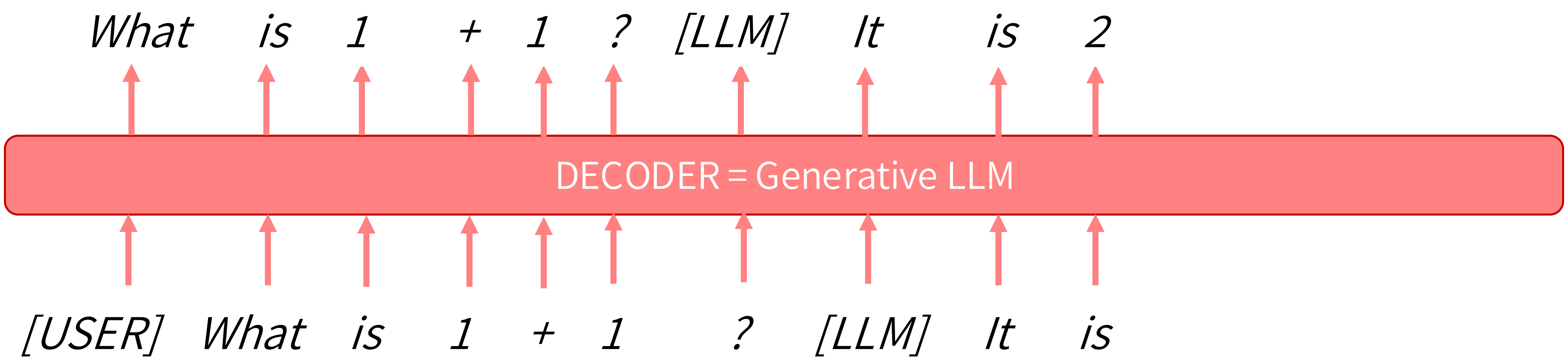
Are LLMs “Generative AI”?



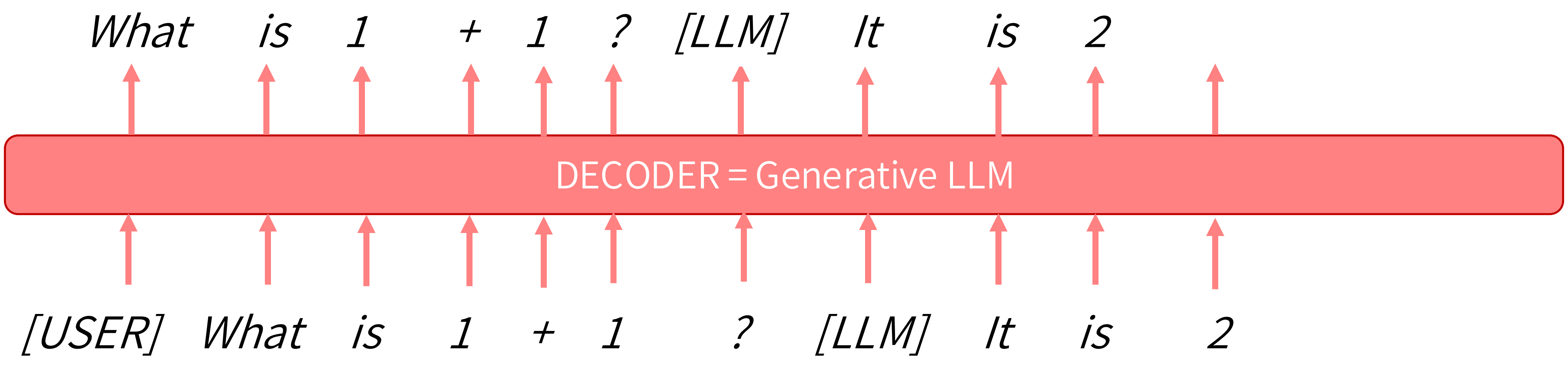
Are LLMs "Generative AI"?



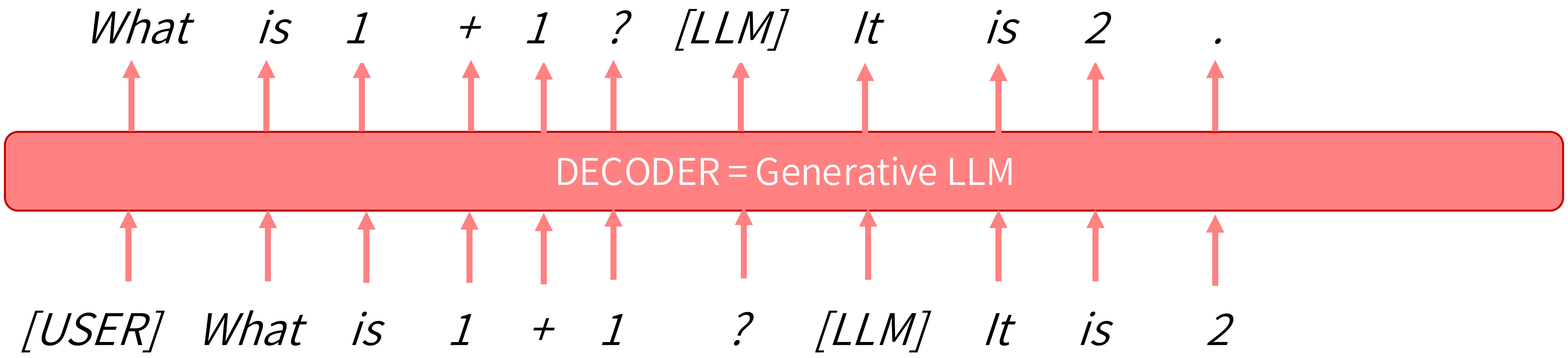
Are LLMs “Generative AI”?



Are LLMs "Generative AI"?

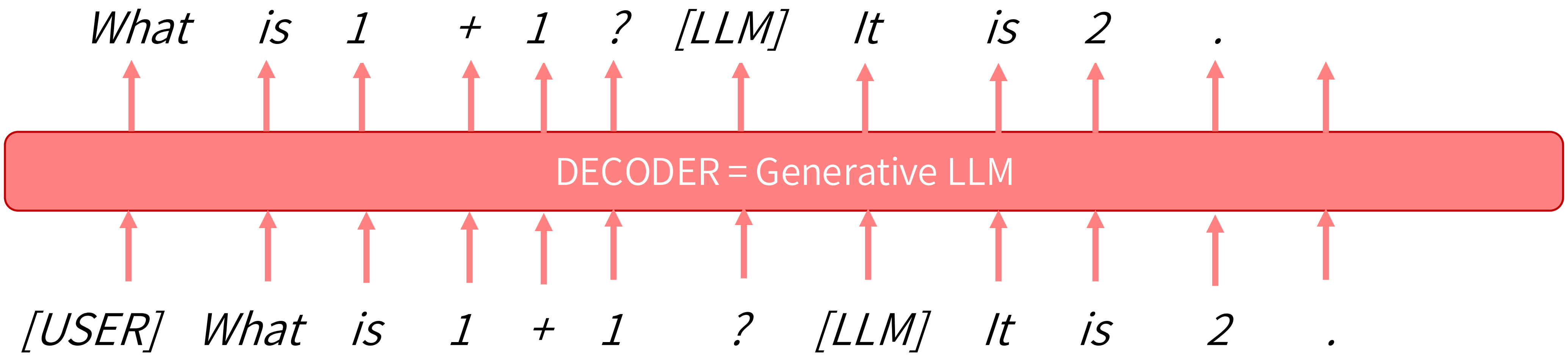


Are LLMs “Generative AI”?



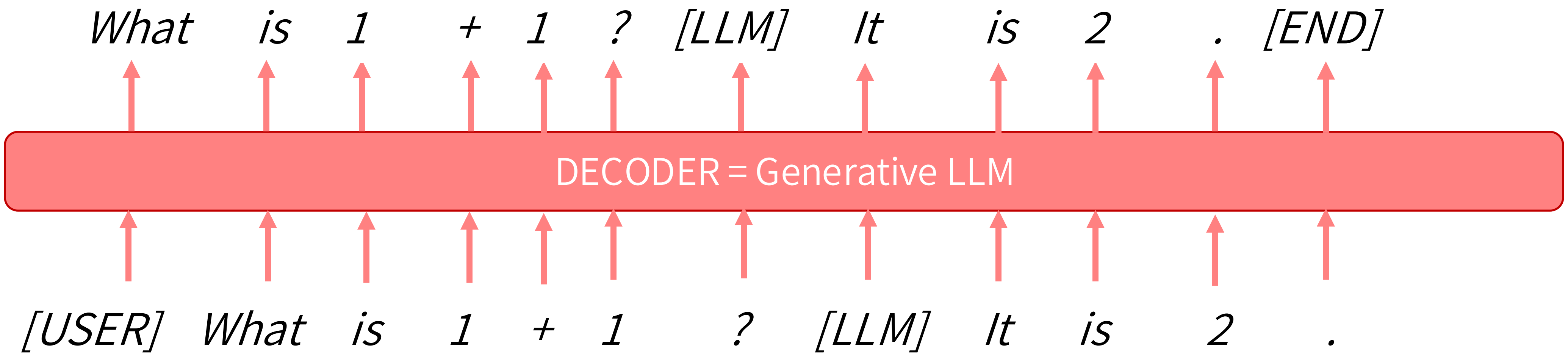
netherlands eScience center

Are LLMs "Generative AI"?



netherlands eScience center

Are LLMs “Generative AI”?



LLMs vs Transformers

- Scale
  - Training Parameters
  - Training Data Size
  - Context Window Size
- Post-training
  - Supervised Fine Tunning (SFT). NLP Tasks, Coding, "Reasoning"…
  - Reinforcement Learning with Human Feedback (RLHF). What do human users prefer?
  - Specific Safety Guardrails
- Generalization
  - Scalability promotes *unseen* correlations

“Open Source” and LLMs

- **Open code:** the actual source code that was used for pretraining is available to modify.
- **Open training data:** release or at least detailed description of the text data used for pretraining.
- **Open-weights:** release the trained model parameters while keeping training code or data proprietary.
- **Open architecture:** normally a paper describes the neural network architecture and specific configuration they used for training.
- **Commercial models:** non-transparent, accessed only through APIs and optimized for user engagement.

Provider Name	Model	Chat assistant	Open Weights	Open Training data	Open Code / Arch
Anthropic	Claude 4 Sonnet Claude 4.1 Opus	Claude (claude.ai)	❌	❌	❌
Google	Gemini 2.5 Flash Gemini 2.5 Flash-lite	Gemini (gemini.google.com)	❌	❌	❌
	Gemma 3 27B	Gemma (aistudio.google.com)	✅	❌	❌
OpenAI	GPT-5 GPT-4 o3-pro o3 o4-mini	ChatGPT (chat.openai.com)	❌	❌	❌
	gpt-oss-120B	gpt-oss playground (gpt-oss.com)	✅	❌	✅
xAI	Grok 4 Grok 3 Beta	Grok (grok.com)	❌	❌	❌
Zhipu AI	GLM-4.5	Z.ai (chat.z.ai)	✅	❌	Partial Arch
Alibaba Cloud	Qwen3 235B 2507	QWEN (chat.qwen.ai)	✅	❌	Arch
DeepSeek	DeepSeek 3.1 DeepSeek R1	DeepSeek (chat.deepseek.com)	✅	❌	Arch
HuggingFace	SmolLM-360M SmolLM-1.7B	None yet	✅	✅	✅
Meta	Llama 3.1 Llama 2 Llama-3-8B	Llama (llama3.dev)	✅	❌	Arch
NVIDIA	Llama 3.1 Nemotron 70B Llama 3.3 Nemotron Super 49B V1.5	Llama Nemotron (build.nvidia.com/nvidia/llama-3_1-nemotron-70b-instruct) Llama Nemotron Super (build.nvidia.com/nvidia/llama-3_3-nemotron-super-49b-v1_5)	✅	❌	Arch
AllenAI	Olmo Olmo2 32B	AllenAI Playground (playground.allenai.org)	✅	✅	✅
TNO, NFI, SURF	GPT-NL	GPT-NL (gpt-nl.com/signup)	Research license	❌	Planned

[CODE]

LLM Interaction: Basic Text Generation

[CODE]

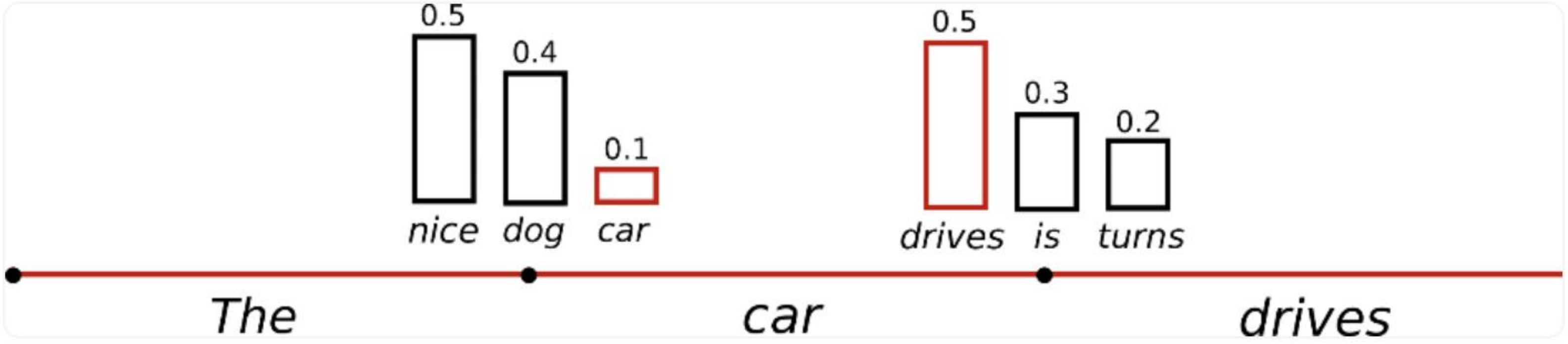
LLM Interaction: Multi-turn Conversation

Common Hyperparameters

- **Max_tokens:** hard limit in terms of tokens (not words!) for LLM output
- **Temperature:**
  - **0** *greedy decoding* (always pick the most likely token)
  - **1** sample from LLM probabilities,
  - **> 1** flattens the probability, so unlikely tokens become more likely
- **Top_p:** only allow to sample from the *top_p* most prominent tokens
- **Top_k:** hard limit on *top_k* tokens to sample (e.g. Top 10 tokens)

Sampling (Top K)

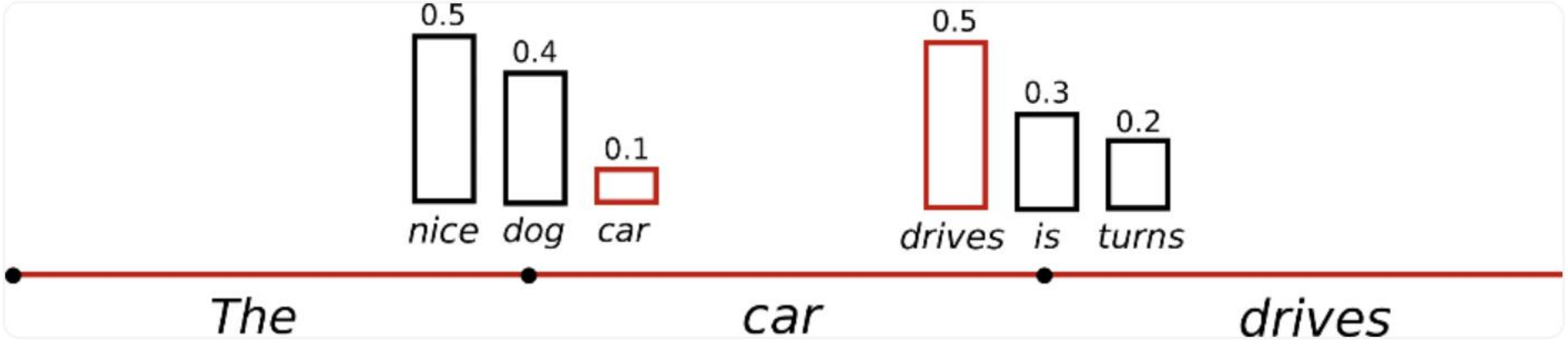
Top_k = 3



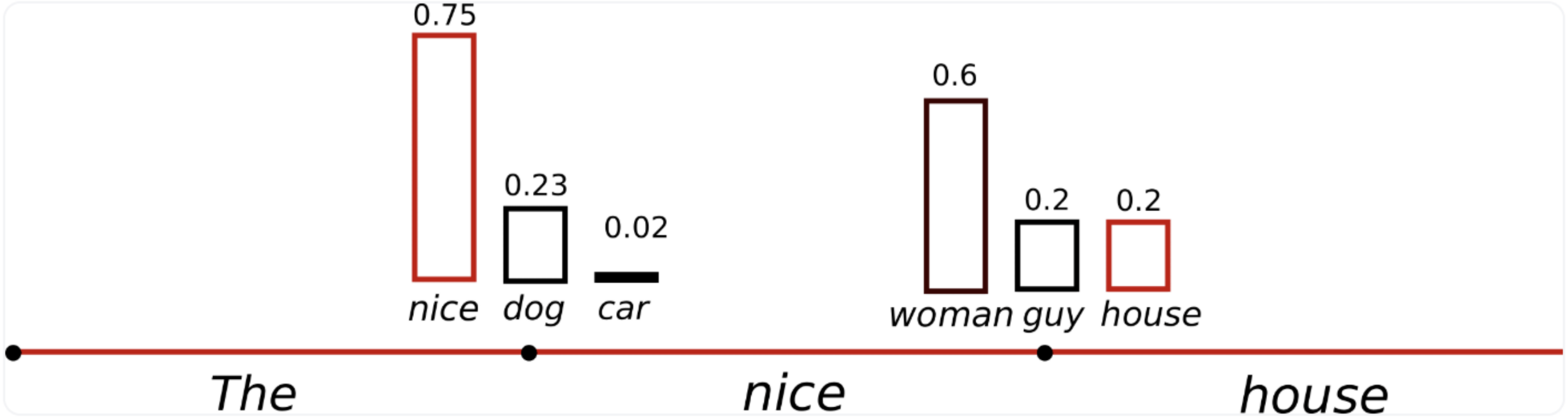
Source: *[HuggingFace]()*

Temperature

Temperature = 1



Temperature = 0.7

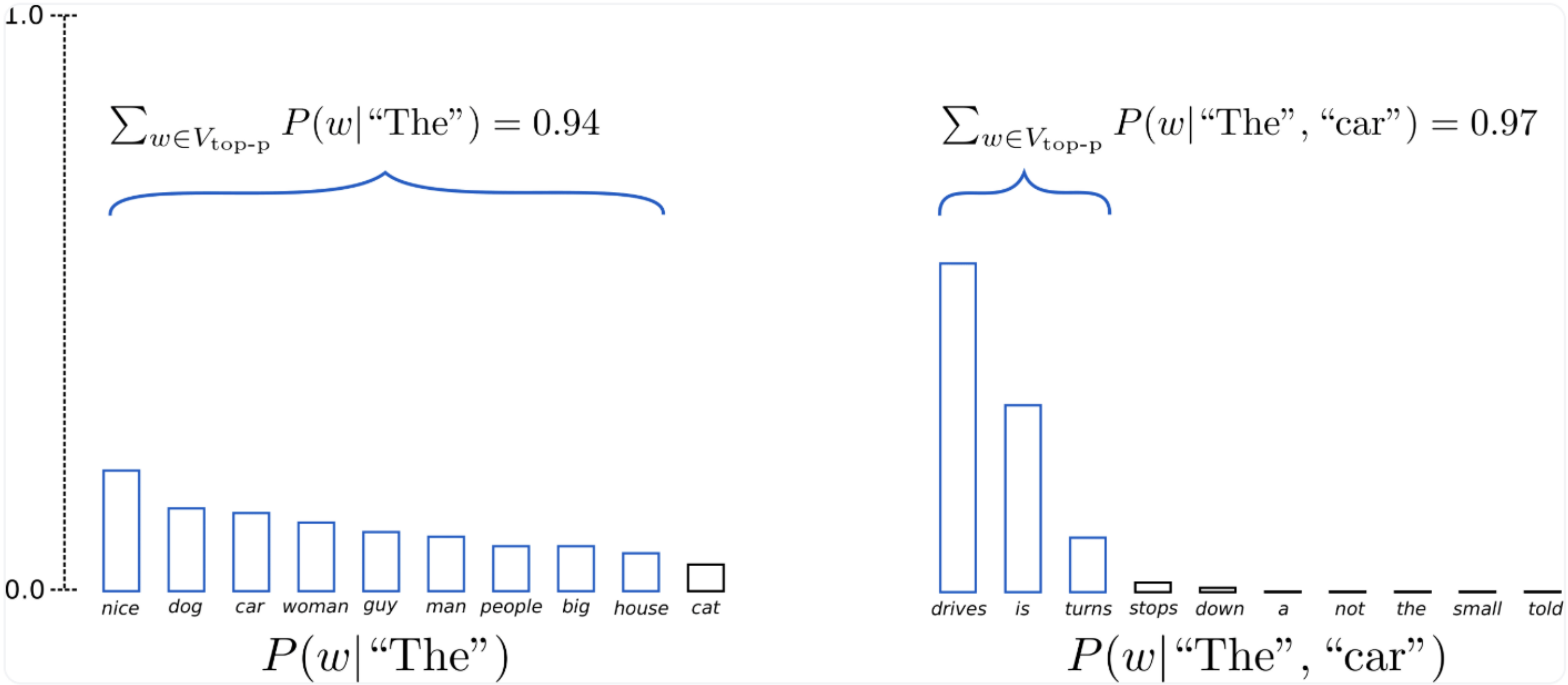


Source: *HuggingFace*

netherlands eScience center

Top_p

Top_p = 0.92



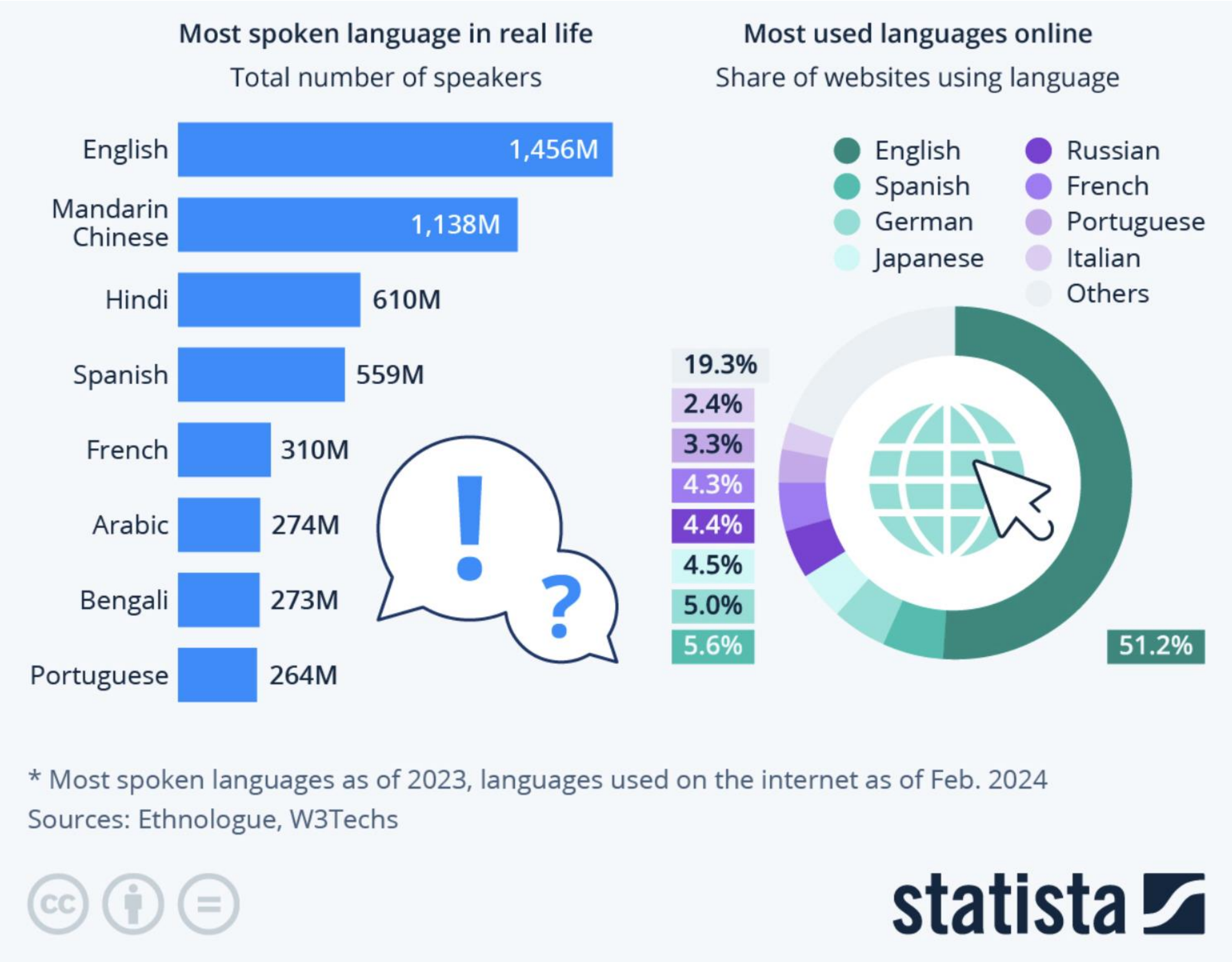
Source: *HuggingFace*

[CODE]

LLM Interaction: Experiment Hyperparameters

Biases

- Hallucinations
- Non-determinism
- Gender Bias
- Informational Bias
- Outdated Knowledge



Source: Statista