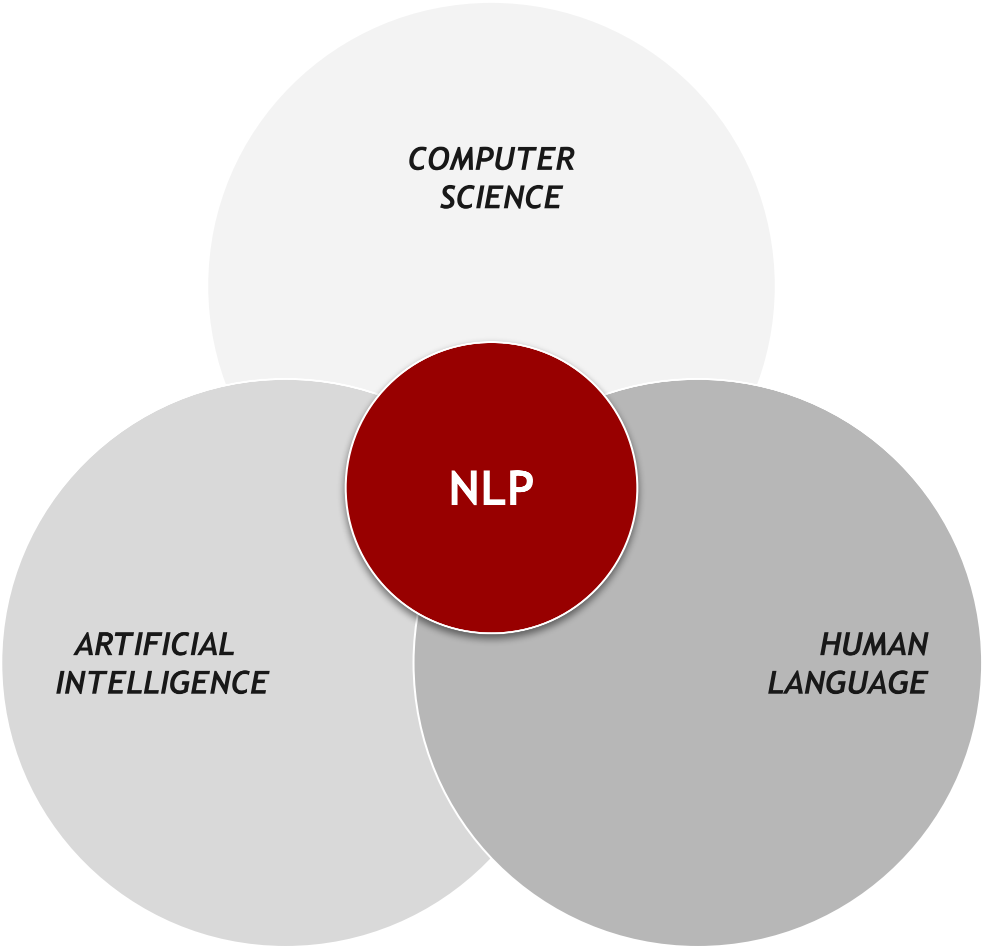
Evolution of NLP

Just enough AI Fundamentals for Generative AI

Natural Language Processing (NLP)



- *NLP is concerned with the interactions between computers and human language*
- *NLP makes interactions with computers a lot human-like*
- *It is concerned with how to program and analyze natural language data*
- *Conversational AI - Virtual personal assistants like Alexa*

Natural Language Processing (NLP)

- NLP is a field of methods to process text

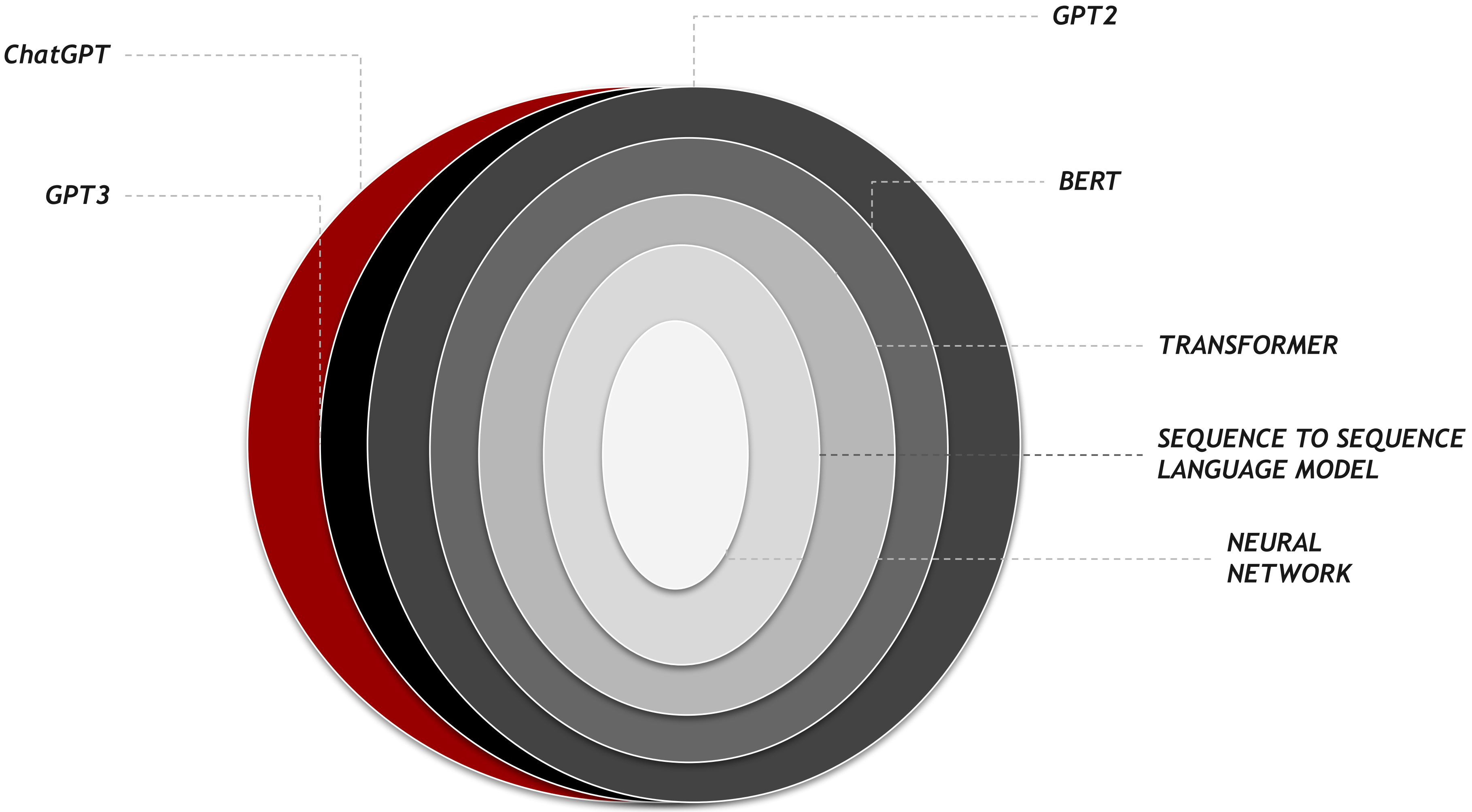
- NLP is useful: summarization, translation, classification, etc

- Language models (LMs) predict words by looking at probabilities

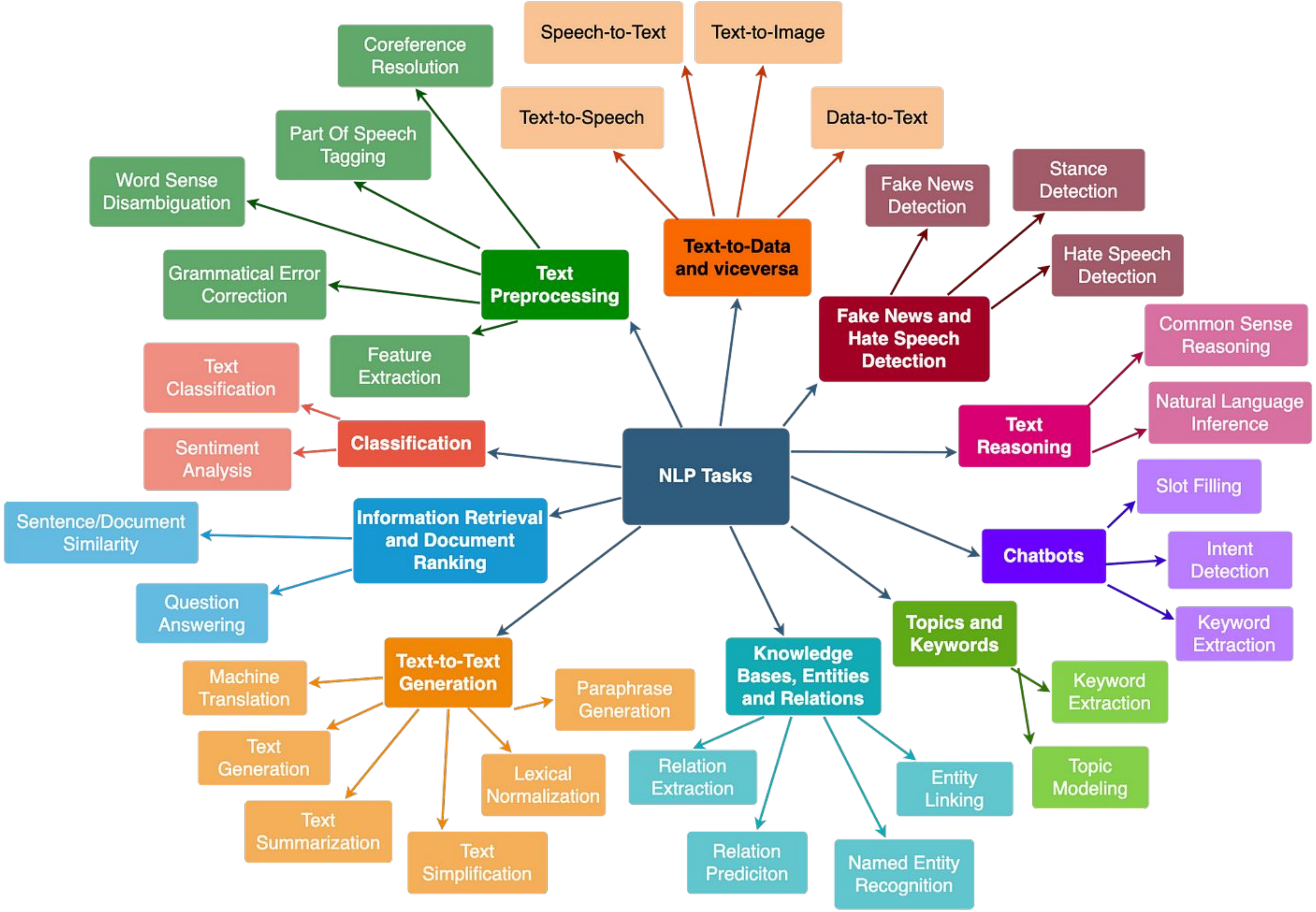
- Large LMs are just LMs with transformer architectures, but bigger

- Tokens are the smallest building blocks to convert text to numerical vectors, aka N-dimensional embeddings

Natural Language Processing



Common NLP Tasks

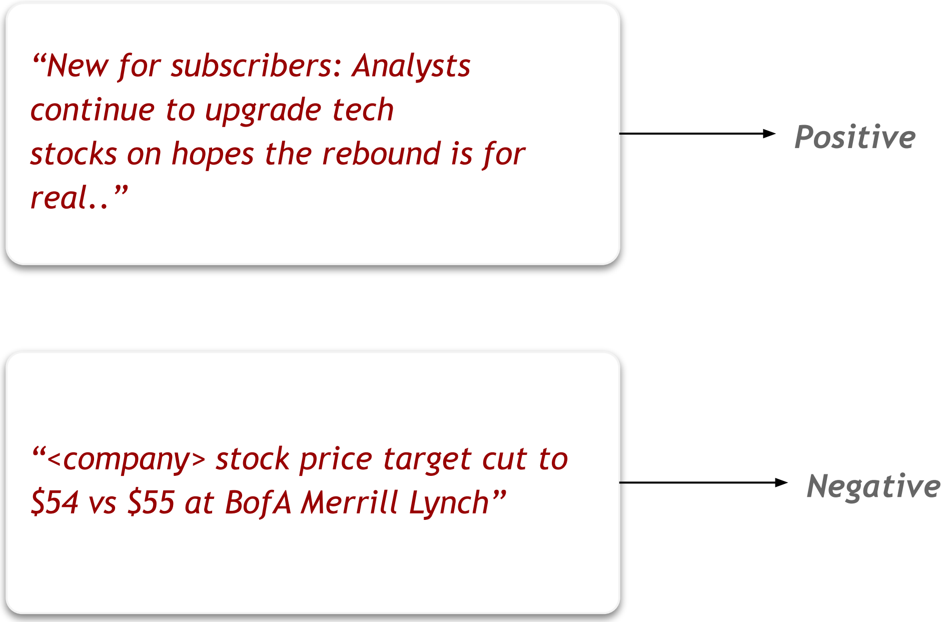


Task: Translation

Example app: Stock market analysis

I need to monitor the stock market, and I want to use Twitter commentary as an early indicator of trends

```
sentiment_classifier( tweets )
Out : [ { ‘label’ : ‘positive’, ‘score’ : 0.997 },
        { ‘label’ : ‘negative’, ’score; : 0.996 },
        ….]
```



“New for subscribers: Analysts continue to upgrade tech stocks on hopes the rebound is for real..”

Positive

“<company> stock price target cut to $54 vs $55 at BofA Merrill Lynch”

Negative

Sentiment Analysis

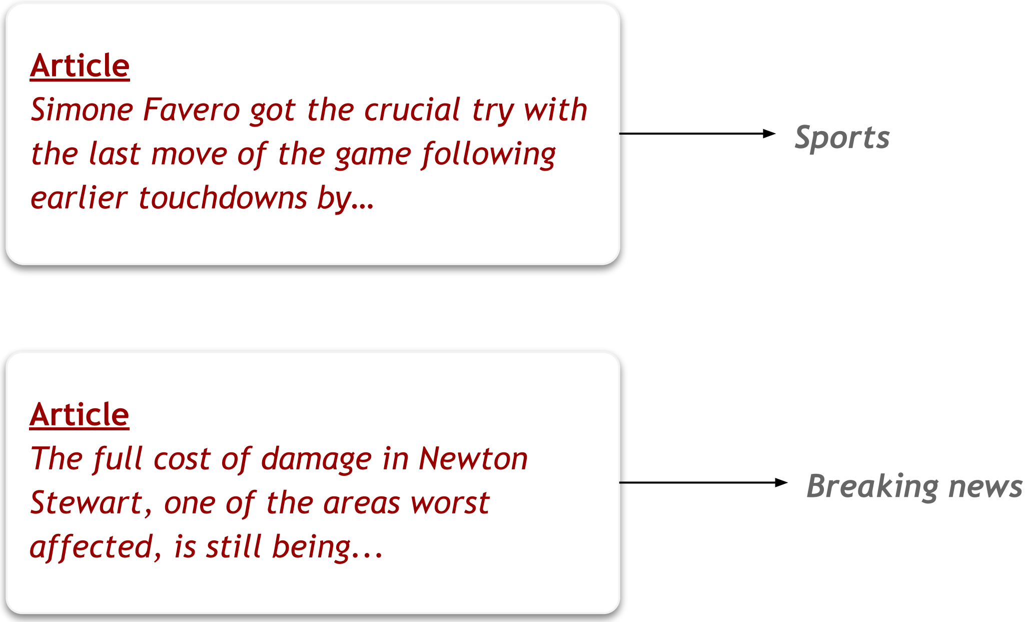
```
en_to_es_translator = pipeline(

    task=”text2text-generation”, # task of variable length

    model=”Helsinki-NLP/opus-mt-en-es”) # translates English to Spanish


en_to_es_translator(“Existing, open-source models… “)

Out : [ { ‘translation_text’ : ‘Los modelos existentes, de código abierto…’ } ]

# General models may support multiple languages and require prompts / instructions.

t5_ translator( “translate English to Romanian: Existing, open-source models. . .” )
```

Task: Zero-Shot Classification

Example app: News browser

Categorize articles with a custom set of topic labels, using an existing LLM

```
predicted_label = zero_shot_pipeline(
      sequences=article,
      candidate_labels = [ “politics”, “Breaking news”, sports” ]
)
```



Article
Simone Favero got the crucial try with the last move of the game following earlier touchdowns by…

Sports

Article
The full cost of damage in Newton Stewart, one of the areas worst affected, is still being…

Breaking news

Task: Few-Shot Classification

“Show” a model what you want

Instead of fine-tuning a model for a task, provide a few examples of that task

```
pipeline(
“””For each tweet, describe its sentiment:
[Tweet] : “I hate it when my phone battery dies.”
[Sentiment] : Negative
###
[Tweet] : “My day has been 👍 “
[Sentimenet] : Positive
###
[Tweet] : “This is the link to the article”
[Sentiment] : Neutral
###
[Tweet] : “This new music video was incredible”
[Sentiments]:””””)
```

Instruction

Example pattern for LLM to follow

Query to answer

Some useful NLP Definitions

The moon, Earth's only natural satellite, has been a subject of fascination and wonder for thousands of years

Token

Basic building block

- The
- Moon
- Earth's
- Only
- “””
- years

Sequence

Sequential list of tokens

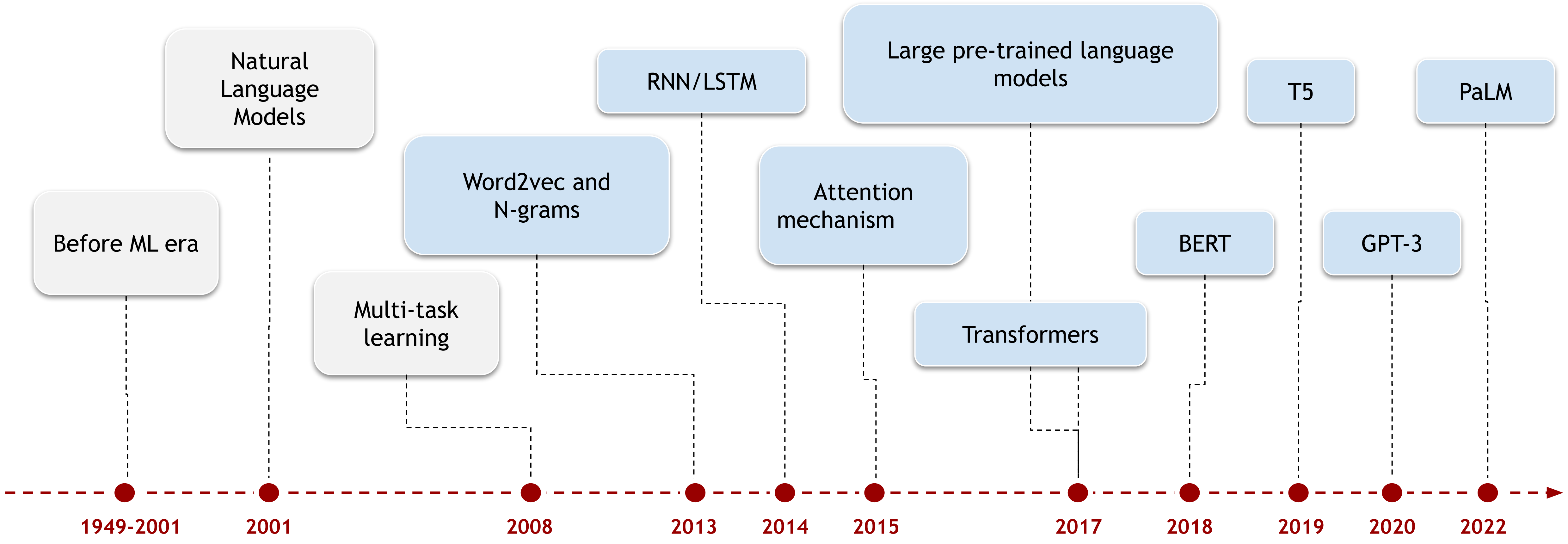
- The moon
- Earth's only natural satellite
- Has been a subject of
- “””
- Thousands of years

Vocabulary

Complete list of tokens

```
{
1 : "The",
569 : "moon",
122 :
430 : "Earth",
50 : “**’s”,
…}
```

Language Model History



NLP is useful for variety of domains

Sentiment Analysis: Product Review

This book was terrible and went on about.. → *Negative*

Translation

I like This book → *Me gusta este libro*

Question answering: chatbots

What's the best sci-fi book ever? → *It really depends on your preferences. Some of the top-rated ones include*

Other use cases

Semantic similarity

- *Literature search*
- *Database querying*
- *Question-Answer matching*

Summarization

- *Clinical decision support*
- *News article sentiments*
- *Legal proceeding summary*

Text classification

- *Customer review sentiments*
- *Genre/topic classification*

Type of Sequence Tasks

Sentiment Analysis: Product Review

This book was terrible and went on about.. → *Negative*

Sequence to non sequence prediction

Translation

I like This book → *Me gusta este libro*

Sequence to sequence prediction

Question answering: chatbots

What's the best sci-fi book ever? → *It really depends on your preferences. Some of the top-rated ones include*

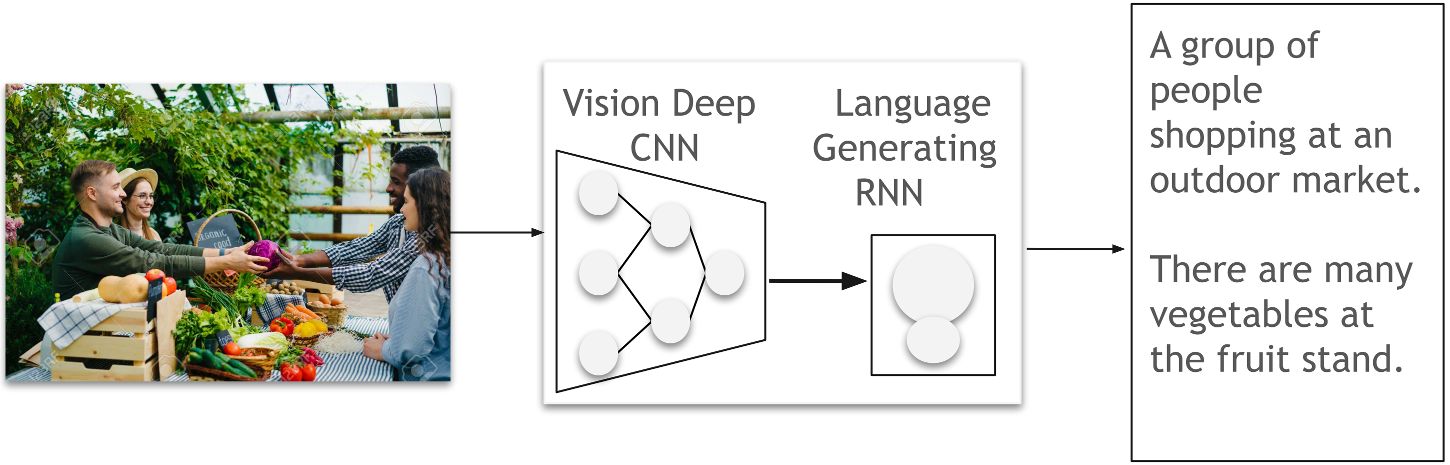
Sequence to sequence generation

NLP goes beyond Text

Speech recognition

Image caption generation

Image generation from text



A group of people shopping at an outdoor market.

There are many vegetables at the fruit stand.

Text Interpretation is challenging

"The ball hit the table and it broke"

"What's the best sci-fi book ever?"

Language is ambiguous

Context can change the meaning

There can be multiple good answers

Input data format matters

Lots of work has gone into text representation for NLP

Model size matters

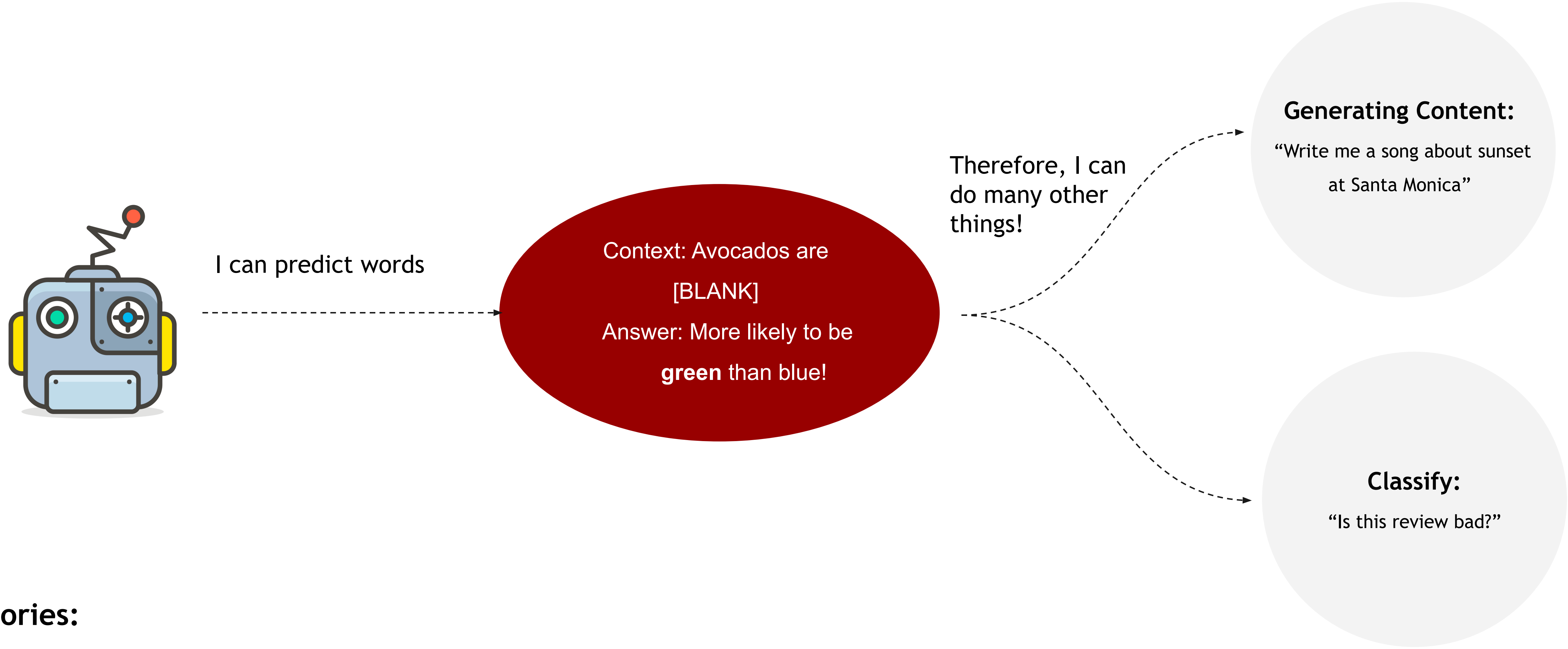
Big models help to capture the diversity and complexity of human language

Training data matters

It helps to have high-quality data and lots of it

What is a Language Model?

LMs assign probabilities to word sequences: find the most likely word



I can predict words

Context: Avocados are [BLANK]

Answer: More likely to be **green** than blue!

Therefore, I can do many other things!

Generating Content:

“Write me a song about sunset at Santa Monica”

Classify:

“Is this review bad?”

Categories:

- **Generative:** find the most likely next word
- **Classification:** find the most likely classification/answer

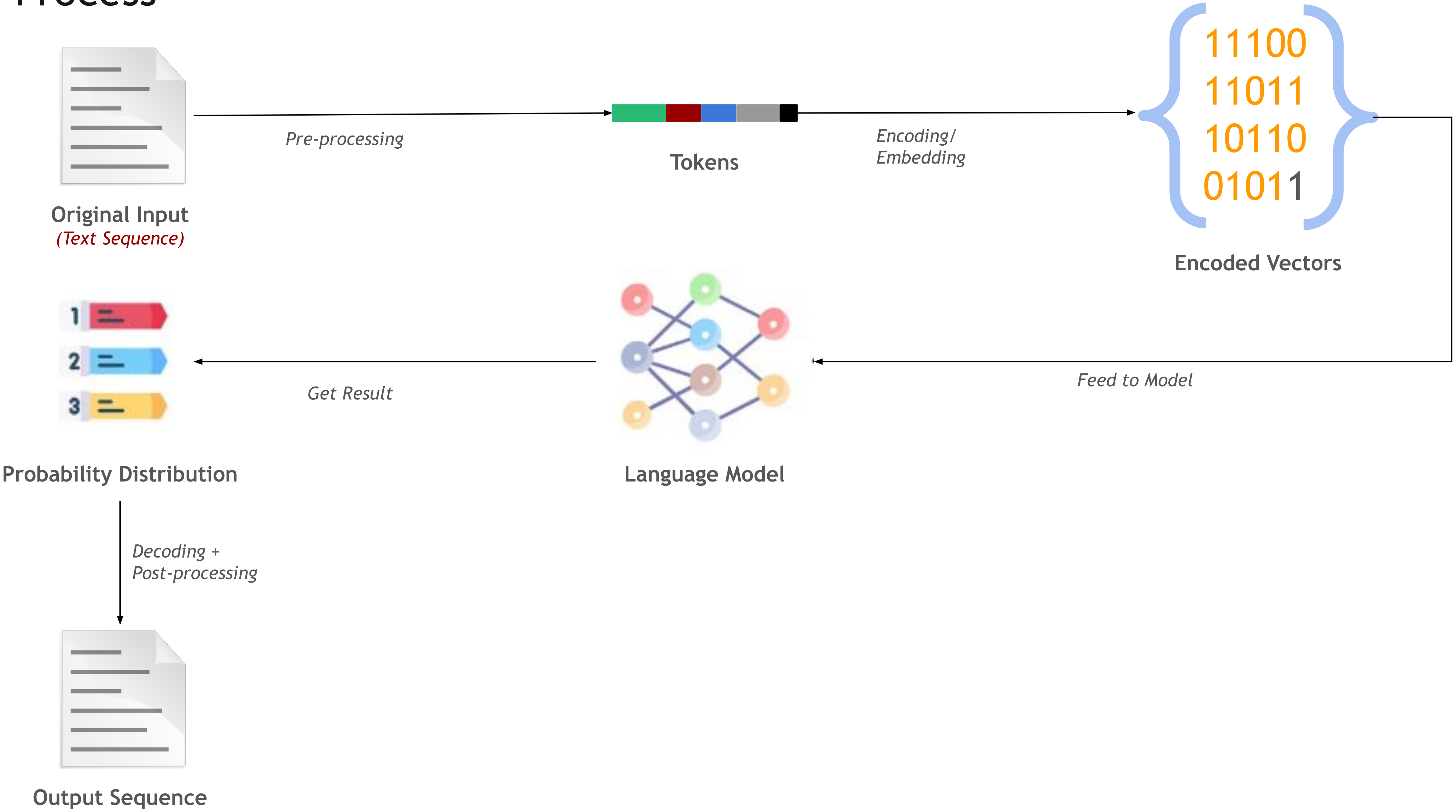
What is a “Large” Language Model?

Language Model	Description	“Large”?	Emergence
Bag-of-Words Model	Represents text as an unordered set words with no consideration for their meaning or context	No	1950-60s
N-gram Model	Considers groups of N consecutive words to capture sequence	No	1950-60s
Hidden Markov Model (HMM)	Represents language as a sequence of hidden states and observable outputs	No	1980-90s
Recurrent Neural Network	Processes sequential data by maintaining an internal state, capturing context of previous inputs	No	1990-2010s
Long Short-Term Memory Model	Extension of RNNs that captures longer-term dependencies without the problem of vanishing gradients	No	2010s
Transformers	Neural network architecture that processes sequences of variable length using a self-attention mechanism	Yes	2017-Present

Language Model & NLP

- There are many types of AI or deep learning models
- For natural language processing (NLP) tasks like conversations, speech recognition, translation, and summarization, we will turn to language models to help us
- **Language models** can learn a library of text (called corpus) and **predict words or sequences of words with probabilistic distributions, i.e., how likely a word or sequence is to occur next**
- For example, when you say ***"Tom likes to eat..."***, the probability of the next word being ***"pizza"*** would be higher than ***"table"***
- If it's predicting the next word in the sequence, it's called ***next-token-prediction***; if it's predicting a missing word in the sequence, it's called ***masked language modeling***
- Since it's a probability distribution, there can be many probable words with different probabilities
- Although you might think it's ideal to always choose best candidate with the highest probability, it may lead to repetitive sequences
- So in practice, researchers would add some randomness (**temperature**) when choosing the word from the top candidates

NLP Process



Original Input

(Text Sequence)

Pre-processing

Tokens

Encoding/
Embedding

11100
11011
10110
01011

Encoded Vectors

Feed to Model

Language Model

Get Result

Probability Distribution

Decoding +
Post-processing

Output Sequence

NLP Process

In a typical NLP process, the input text will go through the following steps:

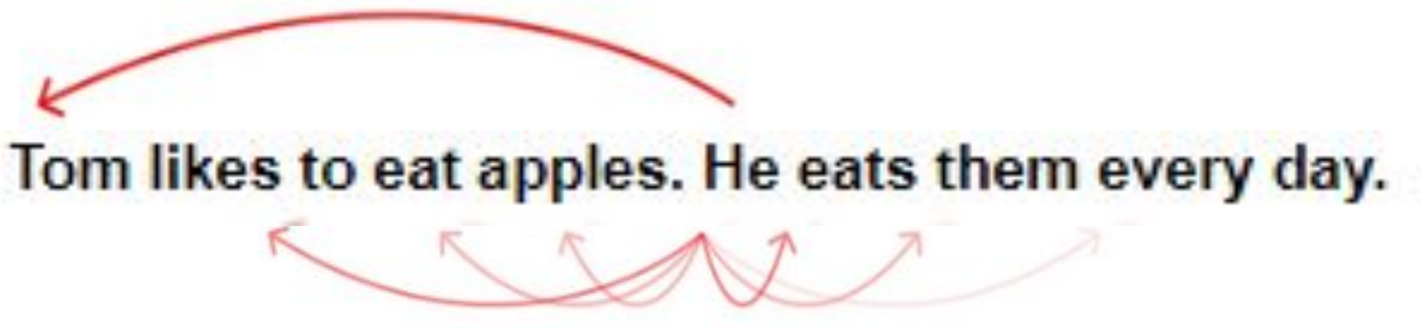
- **Preprocessing:** Cleaning the text with techniques like sentence segmentation, tokenization, stemming, removing stop words, correcting spelling, etc. For example, "Tom likes to eat pizza." would be tokenized into ["Tom", "likes", "to", "eat", "pizza", "."] and stemmed into ["Tom", "like", "to", "eat", "pizza", "."]
- **Encoding or embedding:** turn the cleaned text into a vector of numbers, so that the model can process
- **Feeding to model:** pass the encoded input to the model for processing
- **Getting result:** get a result of a probability distribution of potential words represented in vectors of numbers from the model
- **Decoding:** translate the vector back to human-readable words
- **Post-processing:** refine the output with spell checking, grammar checking, punctuation, capitalization, etc.

Transformer Architecture

- The transformer architecture is the foundation for GPT
- It is a type of neural network, which is similar to the neurons in our human brain
- The transformer can understand contexts in sequential data like text, speech, or music better with mechanisms called **attention** and **self-attention**
- **Attention** allows the model to **focus on the most relevant parts of the input and output by learning the relevance or similarity between the elements**, which are usually represented by vectors
- If it focuses on the same sequence, it's called self-attention

Transformer Architecture

- Let's take the following sentence as an example: "Tom likes to eat apples. He eats them every day."
- In this sentence, "he" refers to "Tom" and "them" refers to "apples"
- The attention mechanism uses a mathematical algorithm identifies related words by calculating a similarity score between the word vectors
- Transformers can now "make sense" of even long text sequences in a more coherent way



Transformer-based Models

- The Transformer-based model is a generative AI model that is primarily used for natural language processing tasks, such as language translation, text generation, and summarization
- The Transformer model uses a self-attention mechanism to simultaneously attend to all words in the input sequence, allowing it to capture long-range dependencies and context better than traditional NLP models
- One of the most common uses of the Transformer model for generative AI is in language translation
- With its ability to capture complex linguistic patterns and nuances, the Transformer model is a valuable tool for generating high-quality text in various contexts.

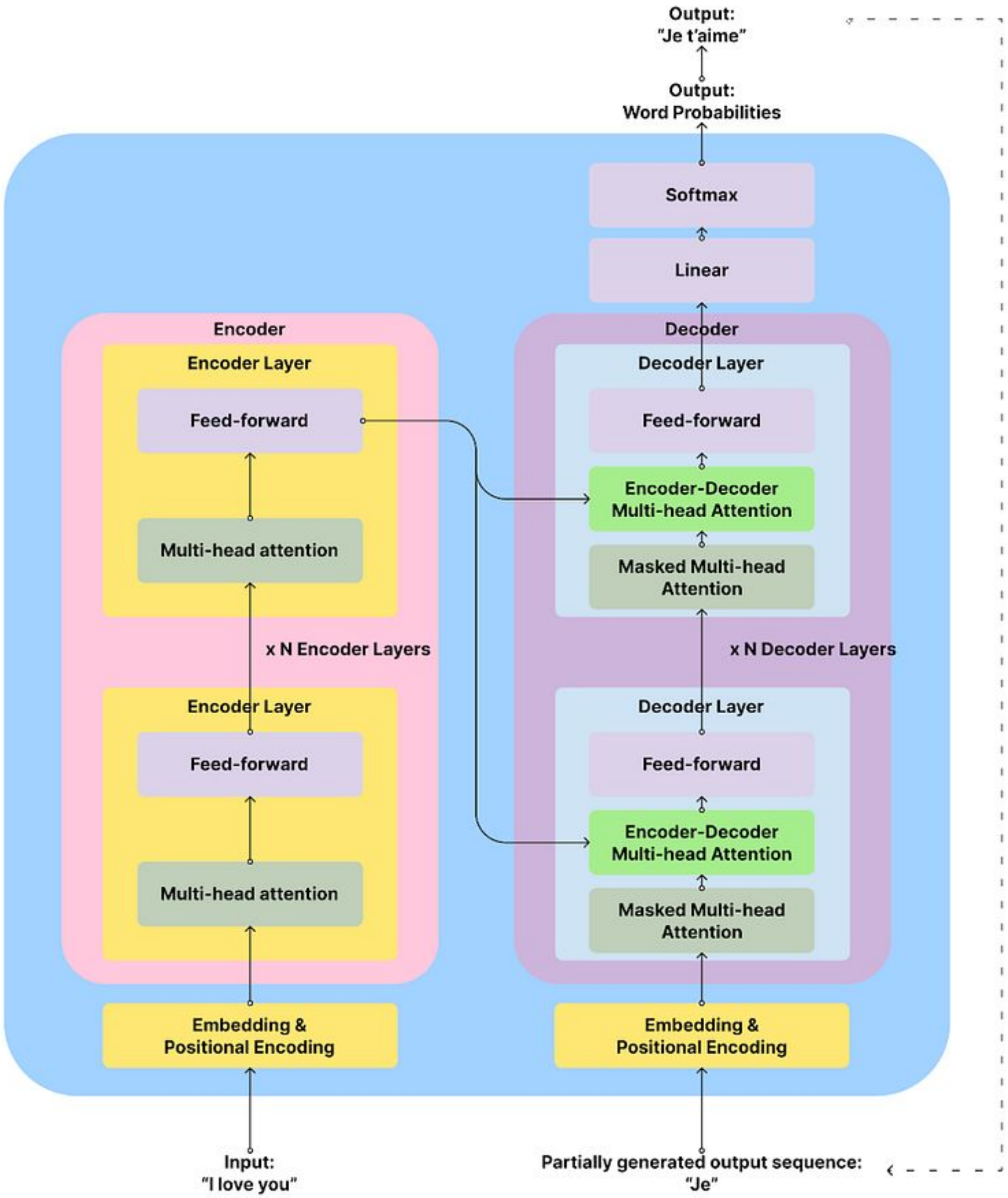
Transformer-based Models

Transformers have the following components:

- **Embedding & Positional Encoding**: turning words into vectors of numbers
- **Encoder:** extract features from the input sequence and analyze their meaning and context then output a matrix of hidden states for each input token to be passed to the decoder
- **Decoder**: generate the output sequence based on the input from the encoder and the output tokens
- **Linear & Softmax Layer**: turn the vector into a probability distribution of output words

The encoder and decoder are the main components of transformer architecture. ***The encoder is responsible for analyzing and "understanding" the input text and the decoder is responsible for generating output***

Transformer-based Models



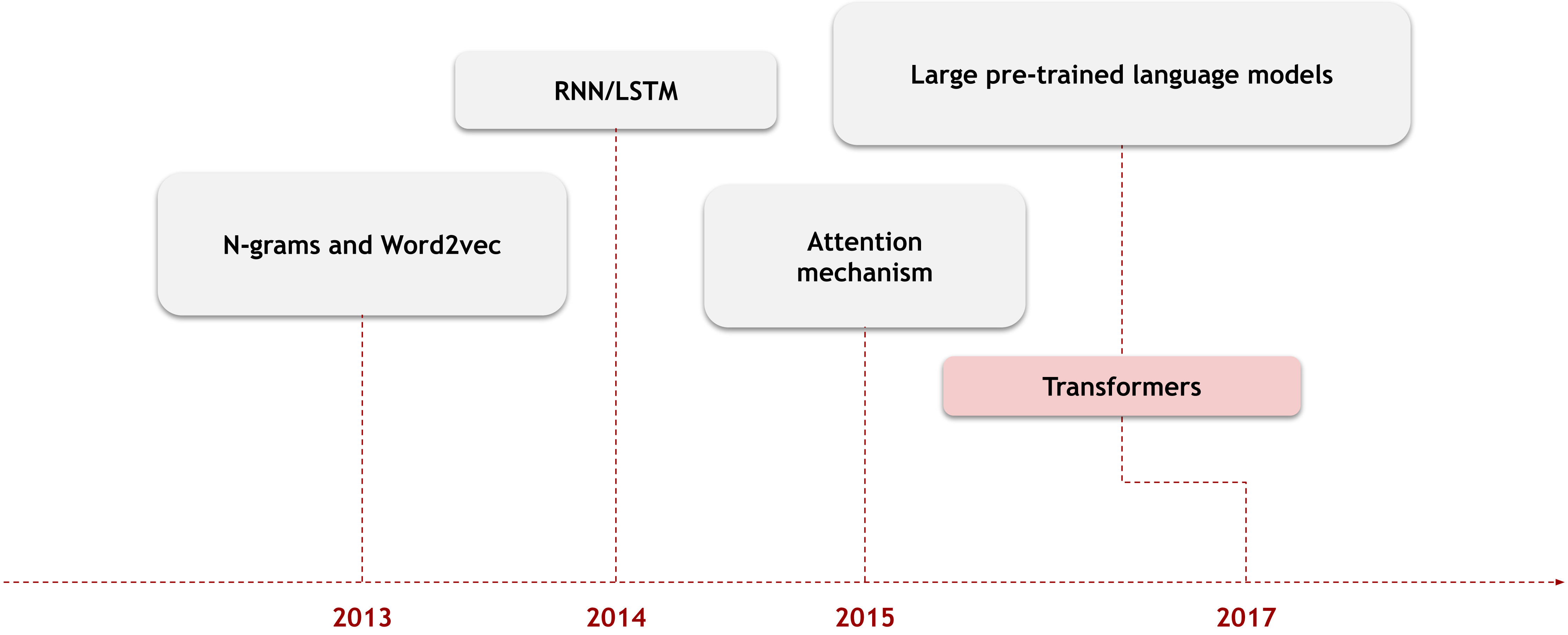
Transformer - Encoders

- The encoder is a stack of multiple identical layers (6 in the original transformer paper)
- Each layer has two sub-layers: a multi-head self-attention layer and a feed-forward layer, with some connections, called residual connection and layer normalization
- The multi-head self-attention sub-layer applies the attention mechanism to find the connection/similarity between input tokens to understand the input
- The feed-forward sub-layer does some processing to prevent overfitting, before passing the result to the next layer
- Think of encoders like you reading a book - you will pay attention to each new word you read and think about how it's related to the previous words

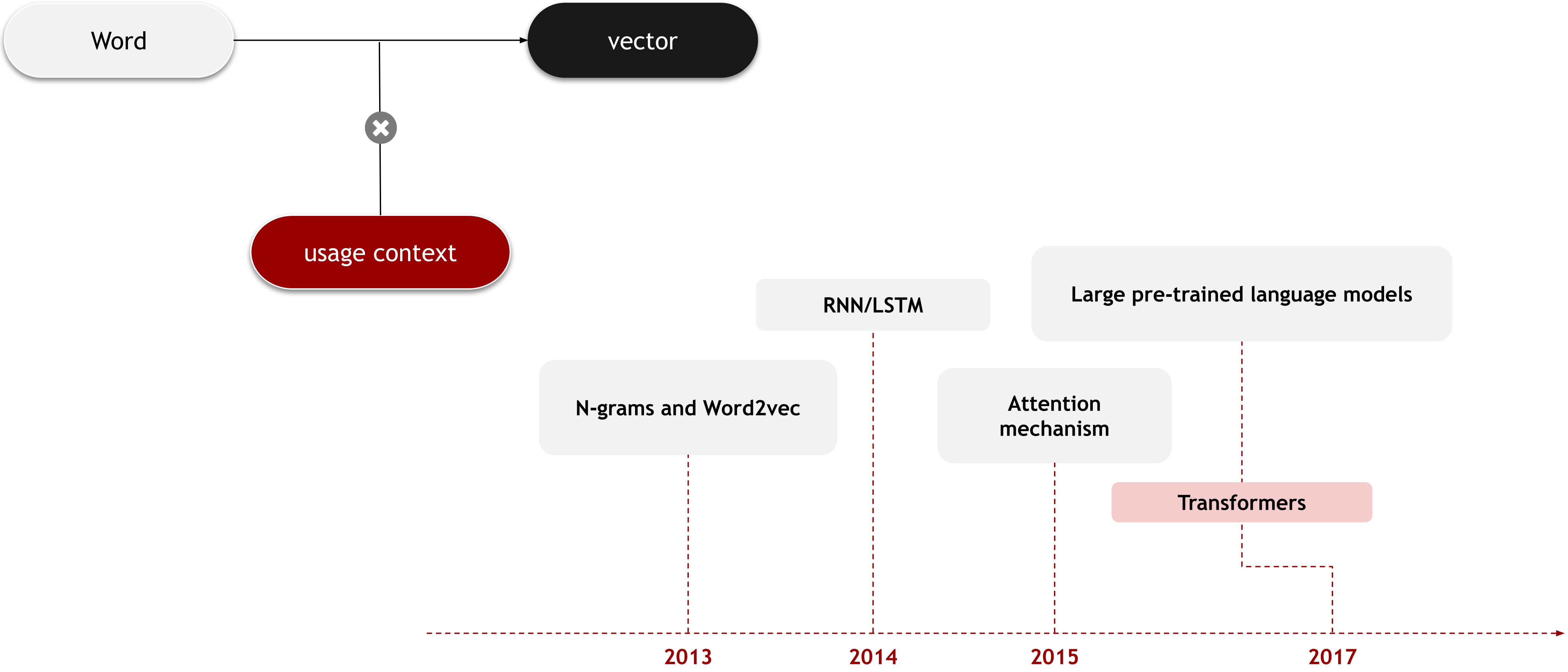
Transformer - Decoders

- The decoder, similar to an encoder, is a stack of identical layers
- But each decoder layer has an additional encoder-decoder attention layer between the self-attention and feed-forward layers, to allow the decoder to attend to the input sequence
- While translating "I love you" (input) to “Je t'aime" (output), the mode needs to know "Je" and "I" are aligned and "love" and "aime" are aligned
- The multi-head attention layers in the decoder are also different
- They're masked to not attend to anything to the right of the current token, which has not been generated yet
- You can think of decoders like free-form writing - you write based on what you've written so far and what you've read, without considering what you're going to write next

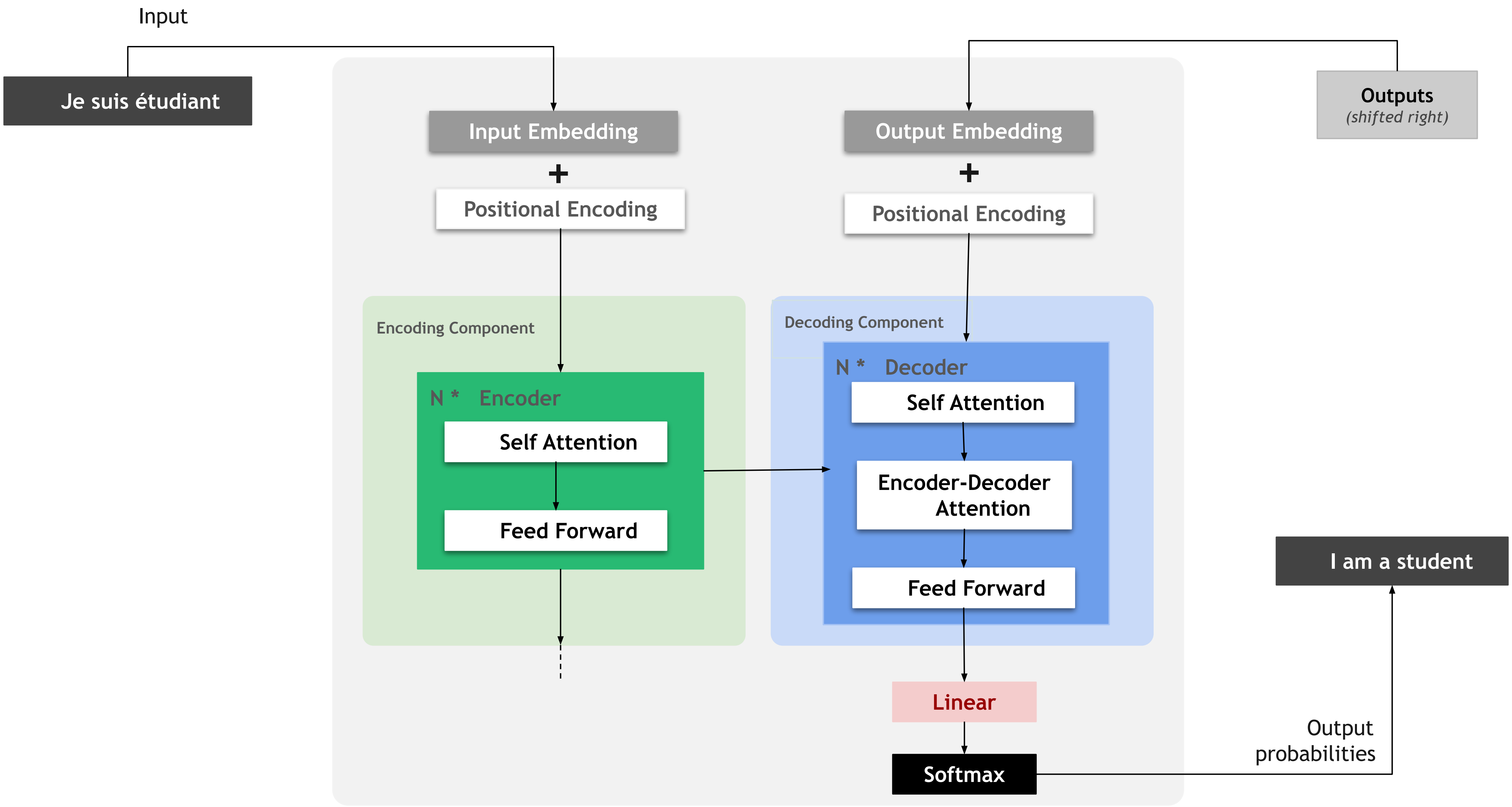
Transformer-based Models



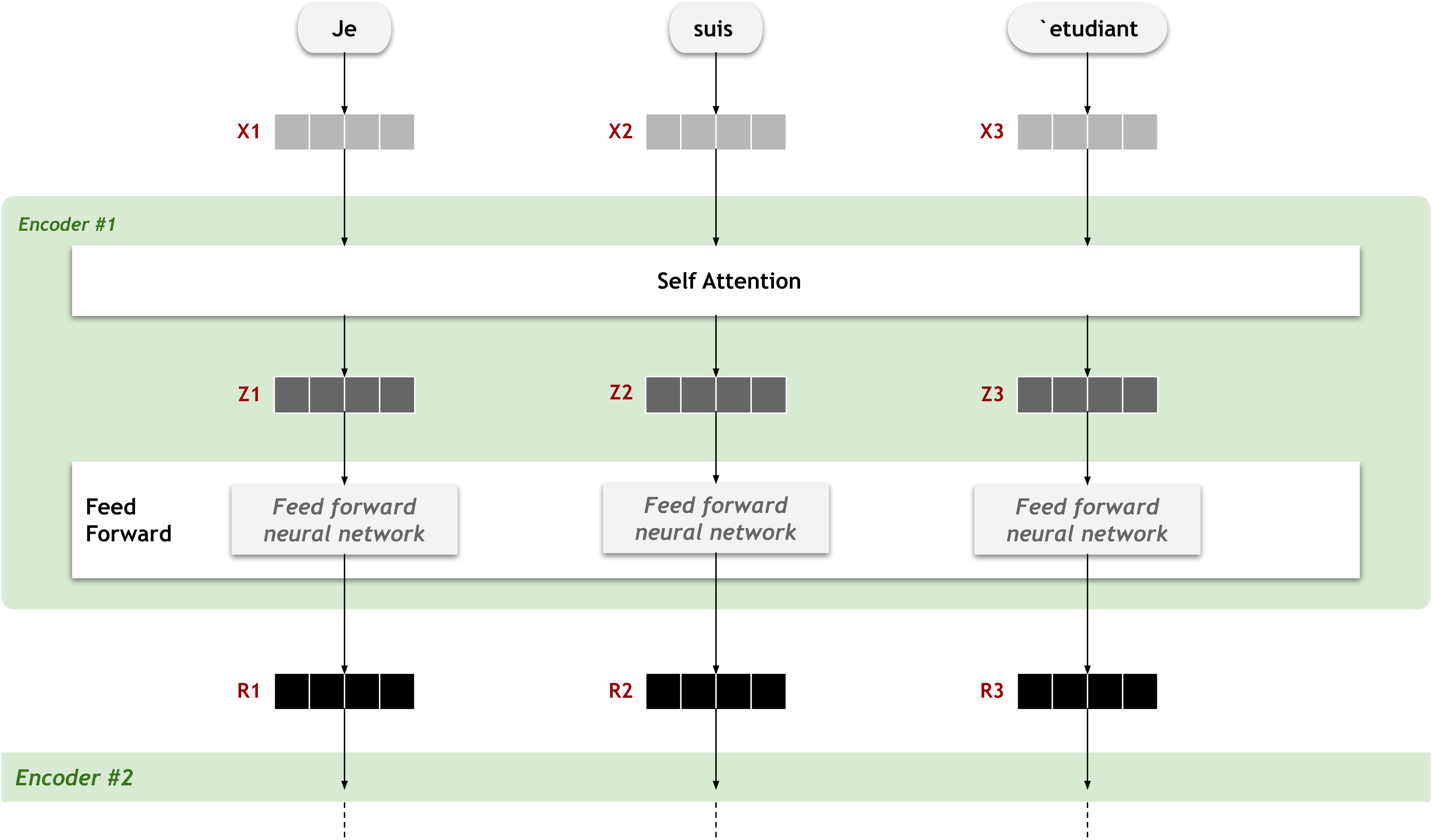
Problem of Text Presentation



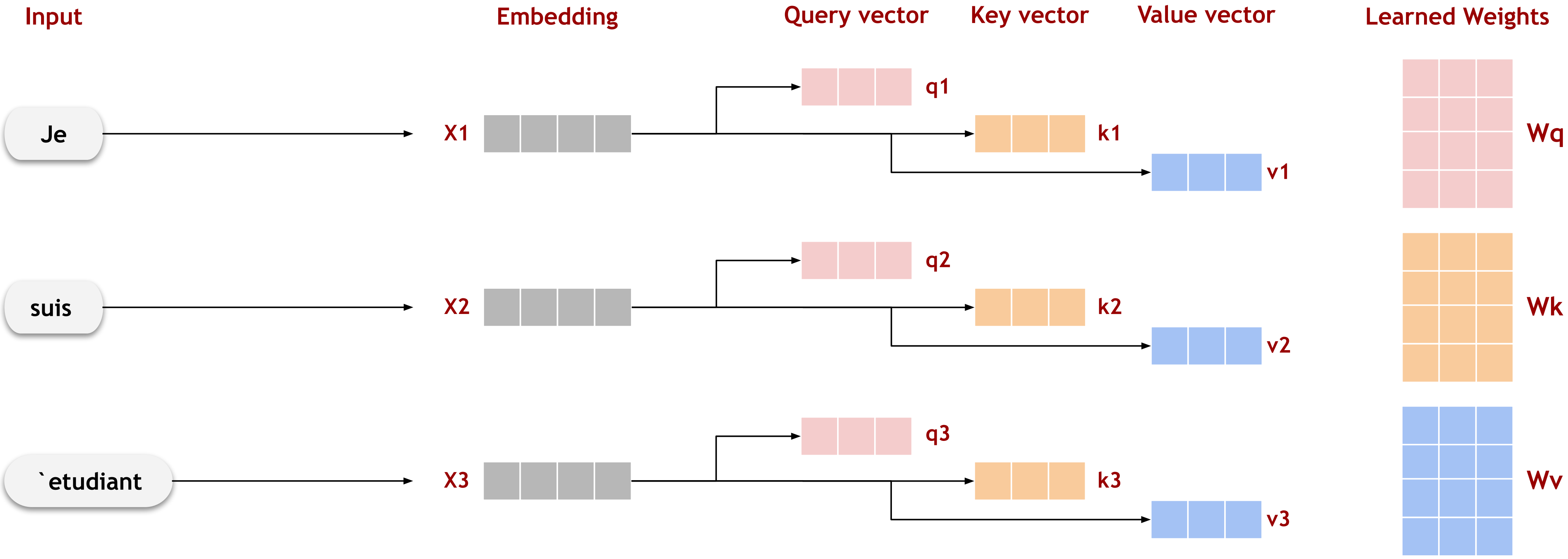
Transformer Model



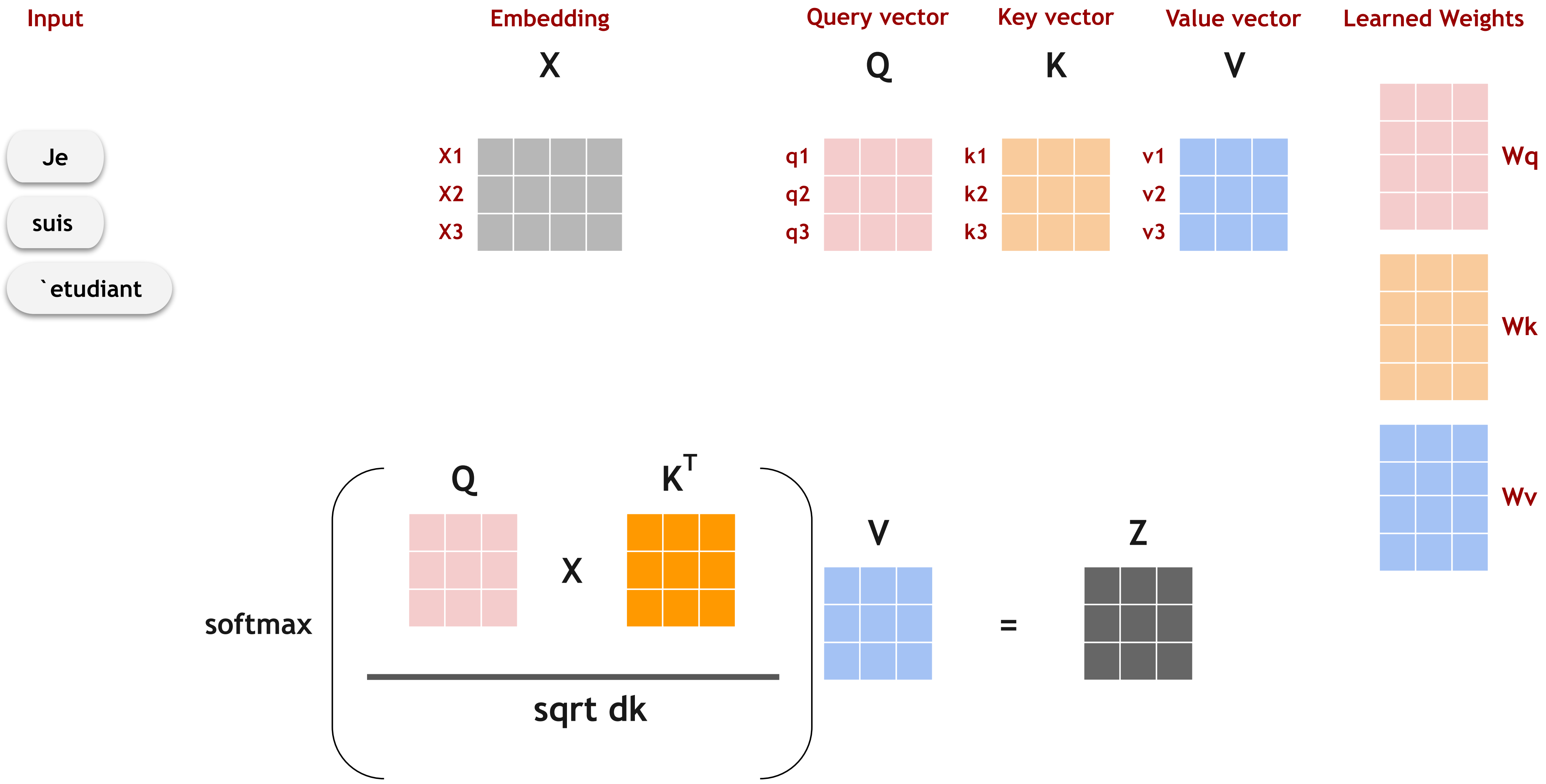
Transformer Model


Je
suis
`etudiant
X1
X2
X3
Encoder #1
Self Attention
Z1
Z2
Z3
Feed
Forward
Feed forward neural network
Feed forward neural network
Feed forward neural network
R1
R2
R3
Encoder #2

Transformer Model



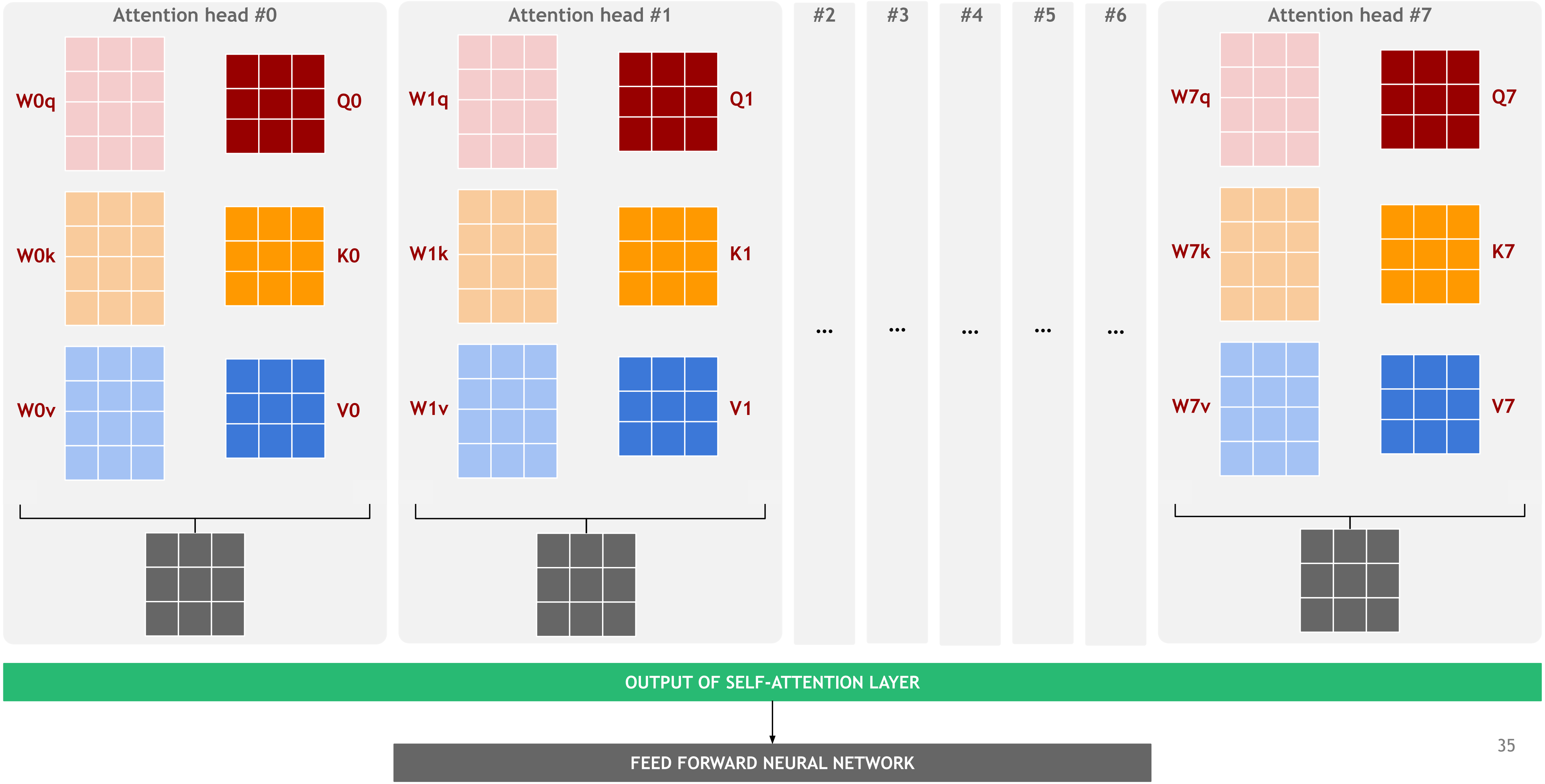
Transformer Model



Processing the final `Z` Value

- *Input natural language sentences*
- *Embed each word*
- *Perform multi-headed attention, and multiply the embedded words with the respective weight matrices*
- *Calculate the attention using the resulting QKV matrices*
- *Concatenate the matrices to produce the output matrix which is the same dimension as the final matrix*

Transformer Model



Pre-Trained Transformer Models

Encoder & Decoder	Decoder only	Encoder only
BART	*GPT-3*	*BERT*
	GPT-2	

BERT Model

BERT

B*idirectional*

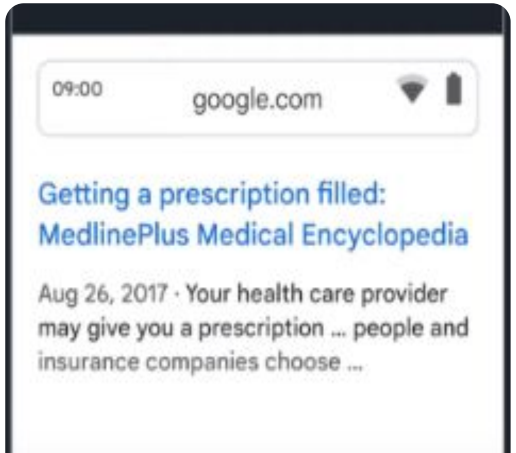
E*ncoder*

R*epresentations from*

T*ransformers*

Can you get medicine for someone from pharmacy?

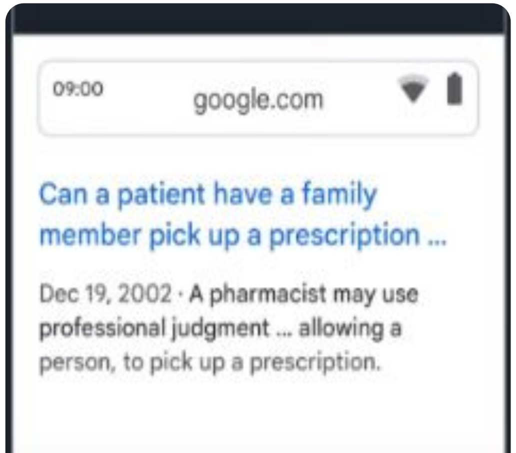
Before



Getting a prescription filled: MedlinePlus Medical Encyclopedia

Aug 26, 2017 · Your health care provider may give you a prescription ... people and insurance companies choose ...

After



Can a patient have a family member pick up a prescription ...

Dec 19, 2002 · A pharmacist may use professional judgment ... allowing a person, to pick up a prescription.

BERT Overview

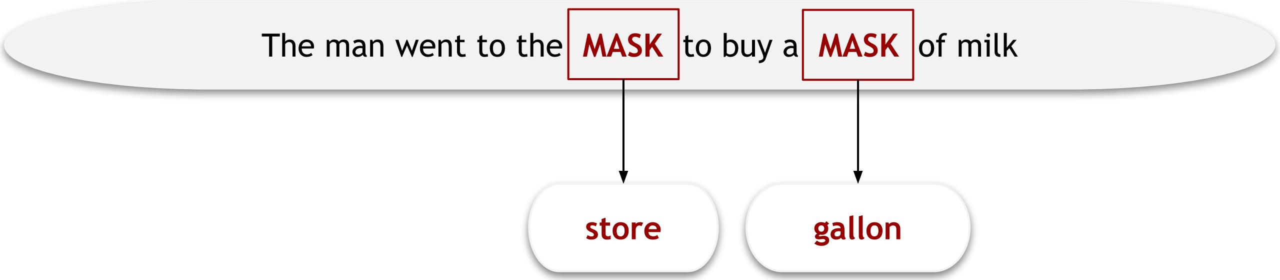
- *Trained in two variations - Base and Large*
- *Able to handle long input context*
- *Trained on entire Wikipedia and BookCorpus*
- *Trained for one million steps*
- *Targeted at multi-task objective*
- *Trained on TPU*
- *Works at both sentence-level and token-level tasks*
- *Can Be fine-tuned for many different tasks*

<table>
<tr><th colspan="4">BERT Versions</th></tr>
<tr><th></th><th>BERT Base</th><th>BERT Large</th><th>Transformer</th></tr>
<tr><th>Layers</th><td>12</td><td>24</td><td>6</td></tr>
<tr><th>Feed Forward Network</th><td>768</td><td>1024</td><td>512</td></tr>
<tr><th>Attention Heads</th><td>12</td><td>16</td><td>8</td></tr>
</table>

Masked Language Model (MLM)

Mask out k% of the input words, and then predict the masked words

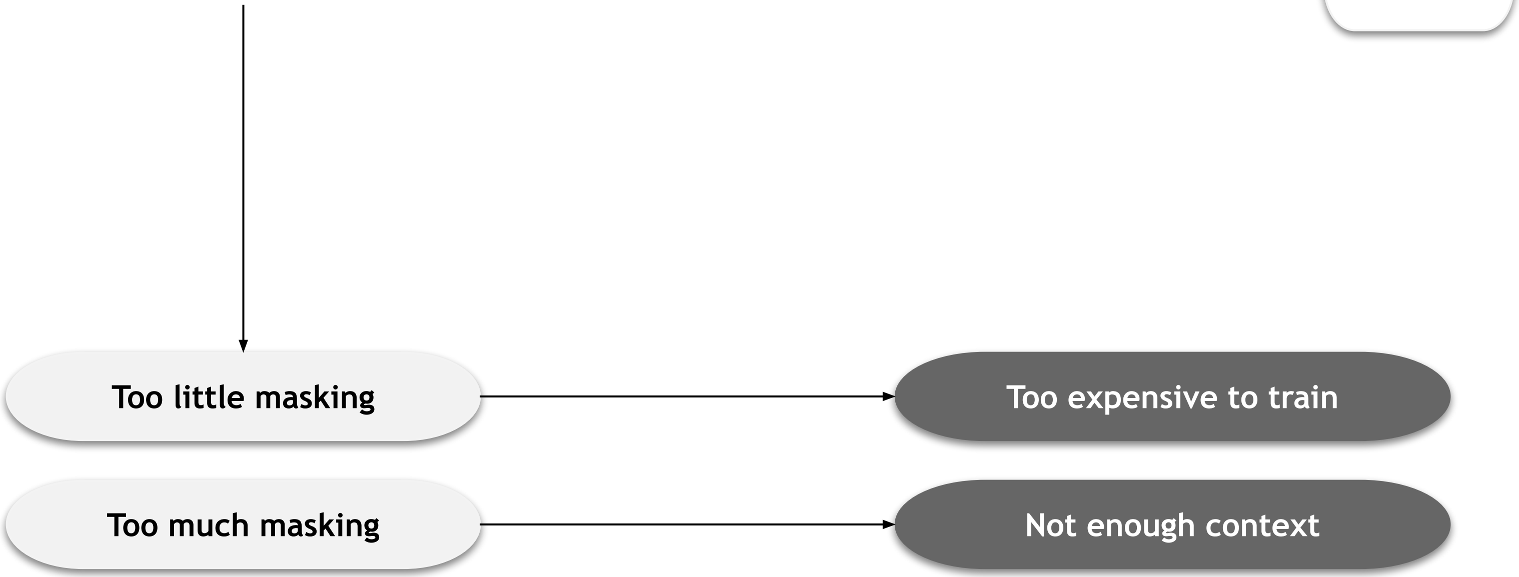
- Recommendation use k = 15%



The man went to the MASK to buy a MASK of milk

store

gallon



Too little masking

Too expensive to train

Too much masking

Not enough context

Next Sentence Prediction (NSP)

Learn the relationships between sentences and predict the next sentence given the first one.

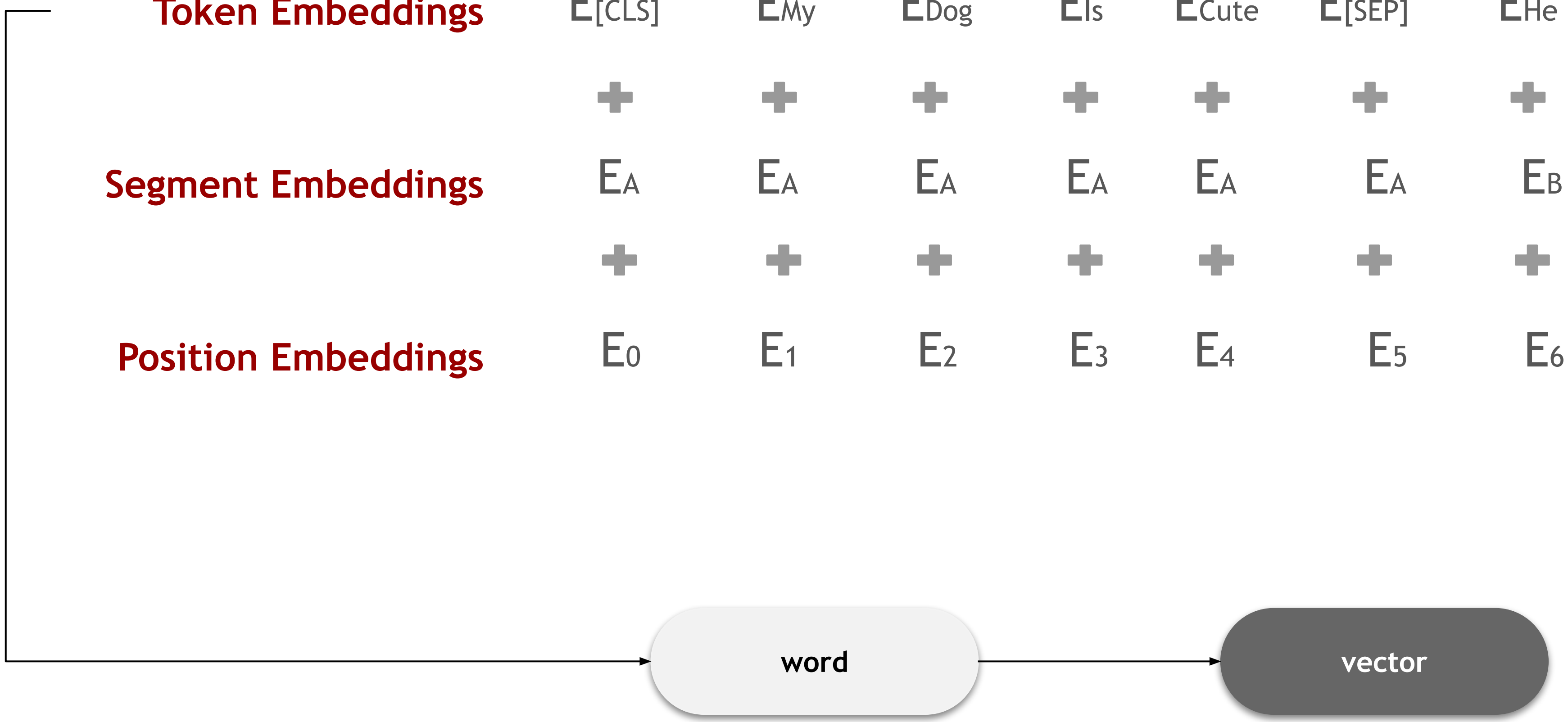
Binary classification task

Sentence A	The man went to the store
Sentence B	He bought a gallon of milk
Label	IsNextSentence

Sentence A	The man went to the store
Sentence B	Penguins are flightless birds
Label	NotNextSentence

BERT Input Embedding

Input	[CLS]	My	Dog	Is	Cute	[SEP]	He	Likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{My}	E_{Dog}	E_{Is}	E_{Cute}	$E_{[SEP]}$	E_{He}	E_{Likes}	E_{play}	$E_{\#\#ing}$	$[E_{SEP}]$
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}



BERT Input Embedding

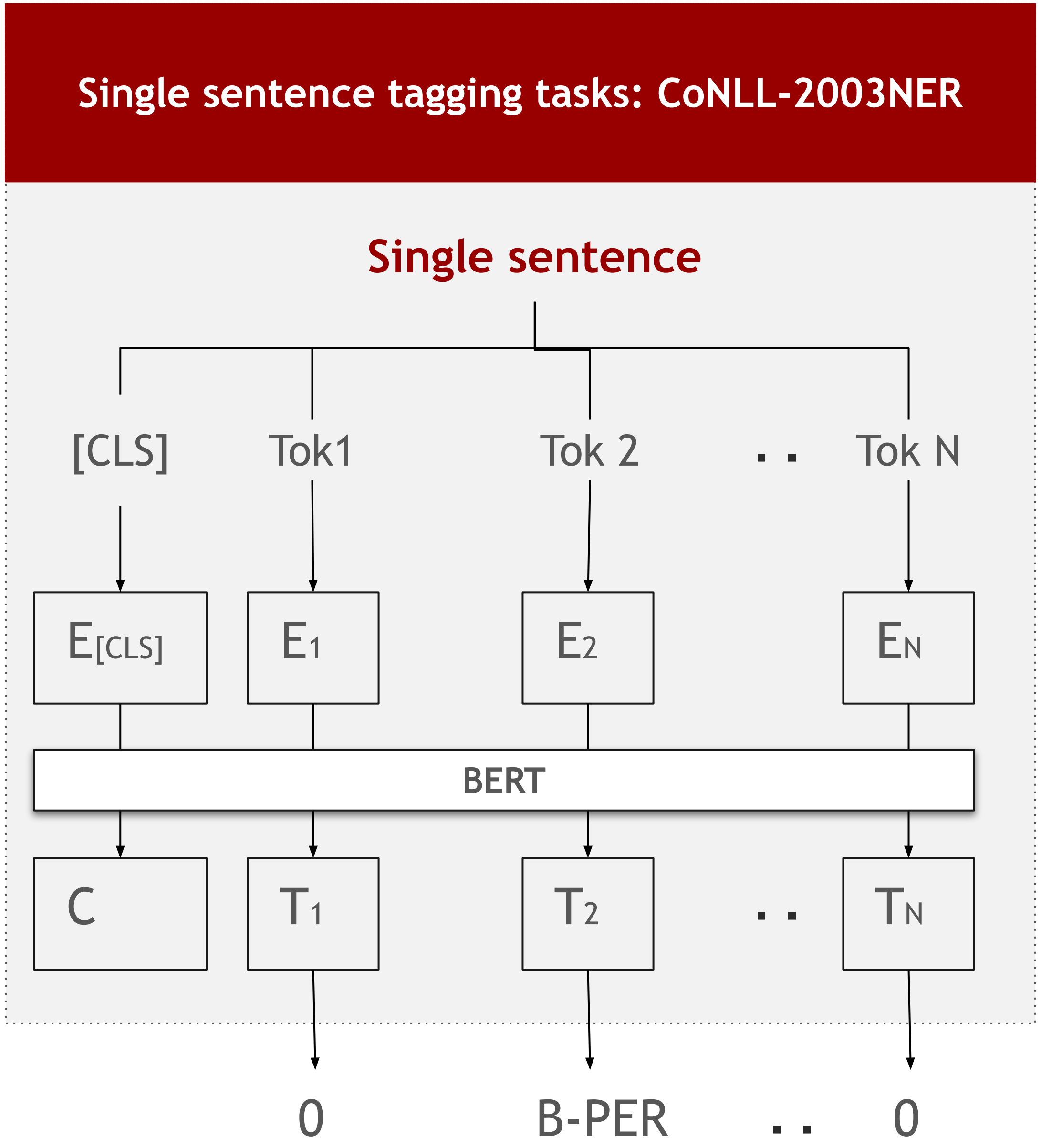
Input	[CLS]	My	Dog	Is	Cute	[SEP]	He	Likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{My}	E_{Dog}	E_{Is}	E_{Cute}	$E_{[SEP]}$	E_{He}	E_{Likes}	E_{play}	$E_{\#\#ing}$	$[E_{SEP}]$
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

Learn a vector representation for each position

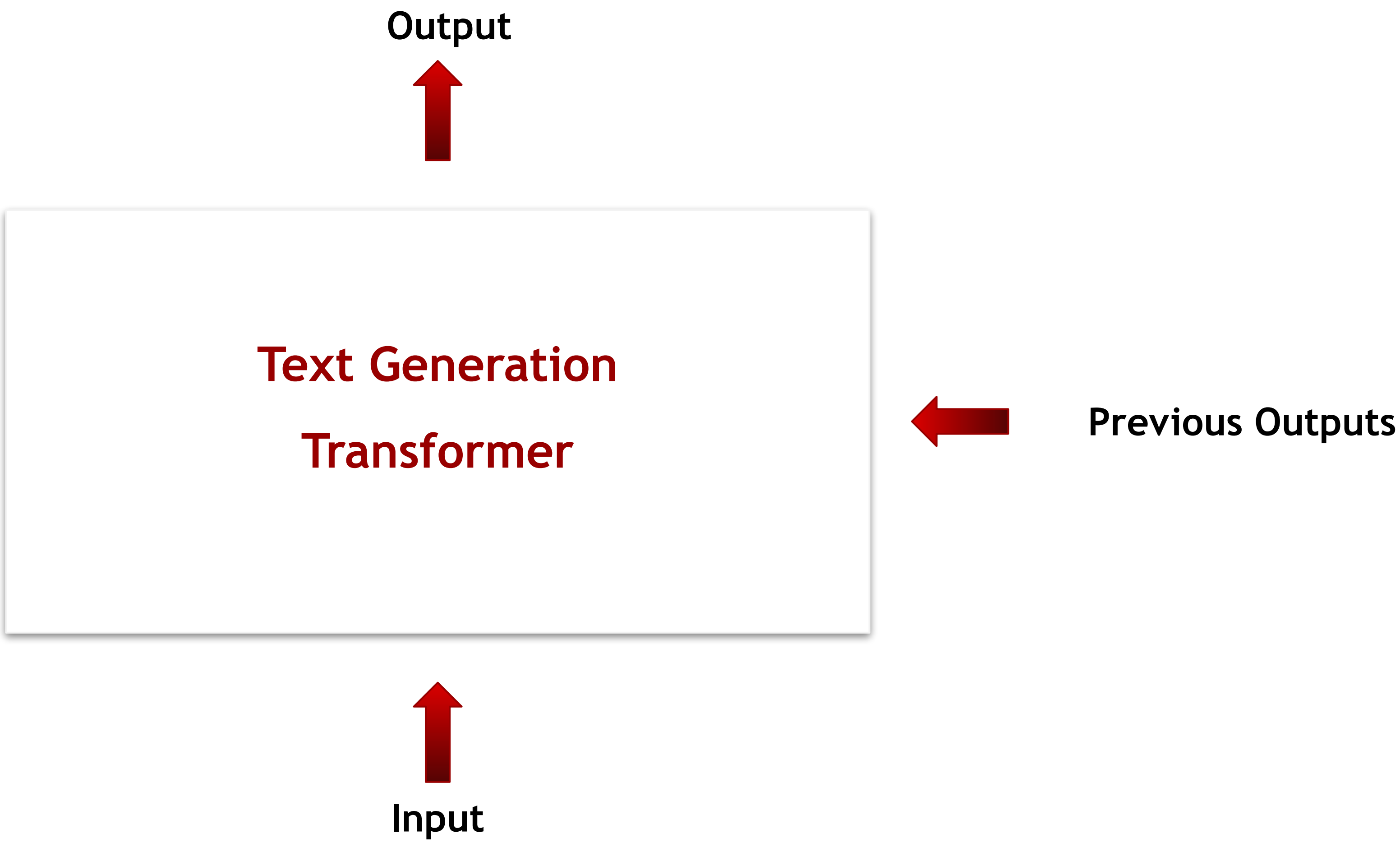
BERT Input Embedding

You can use BERT for various downstream tasks, or example:

- *Single sentence classification*
- *Sentence pair classification*
- *Question answering*
- *Single sentence tagging tasks*



Text Generation Transformer



Output

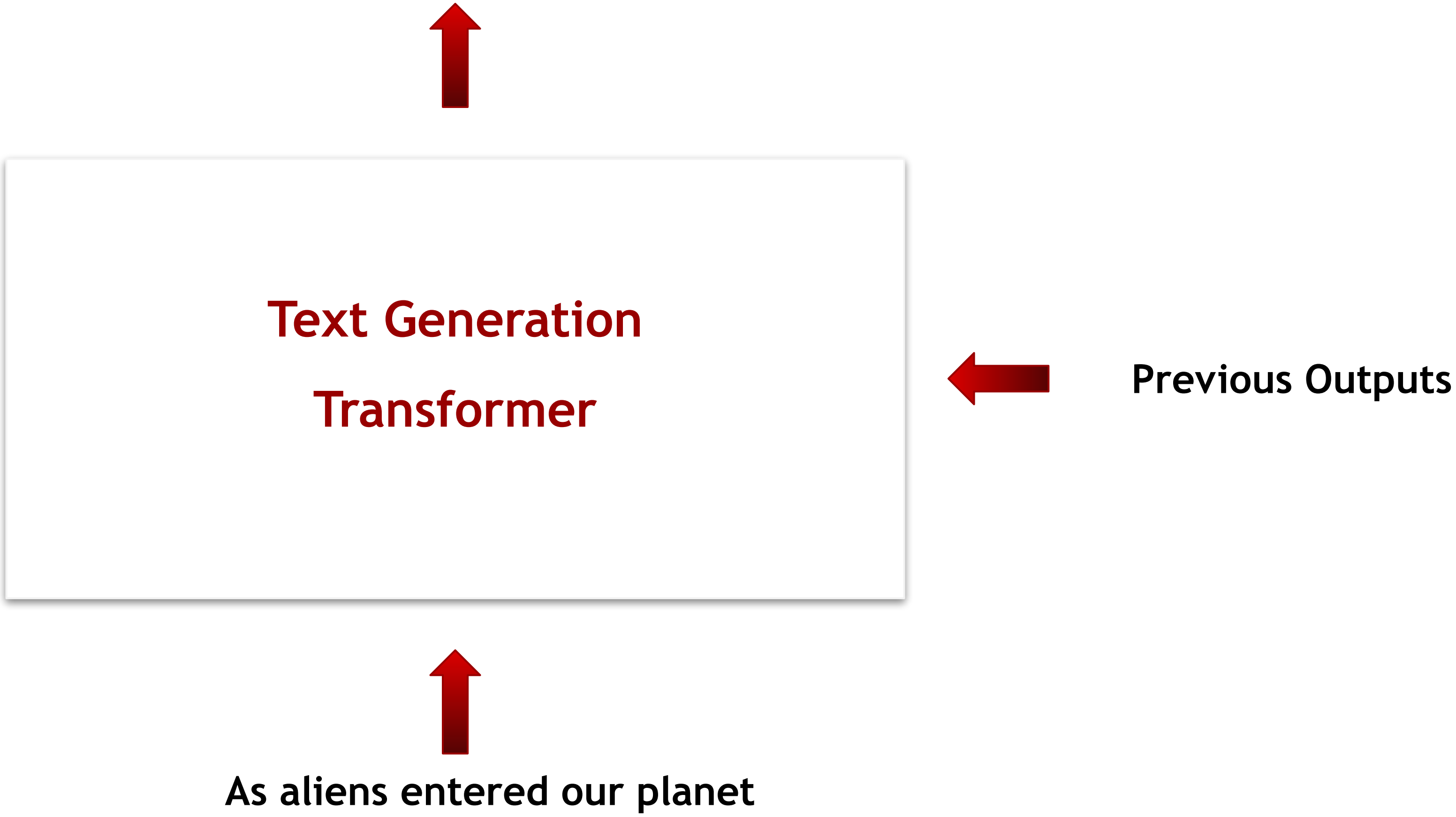
Text Generation Transformer

Previous Outputs

Input

Text Generation Transformer

and began to colonize Earth, a certain group of extraterrestrials began to manipulate our society through their influence of a certain number of the elite of the society to keep an iron grip over the populace….

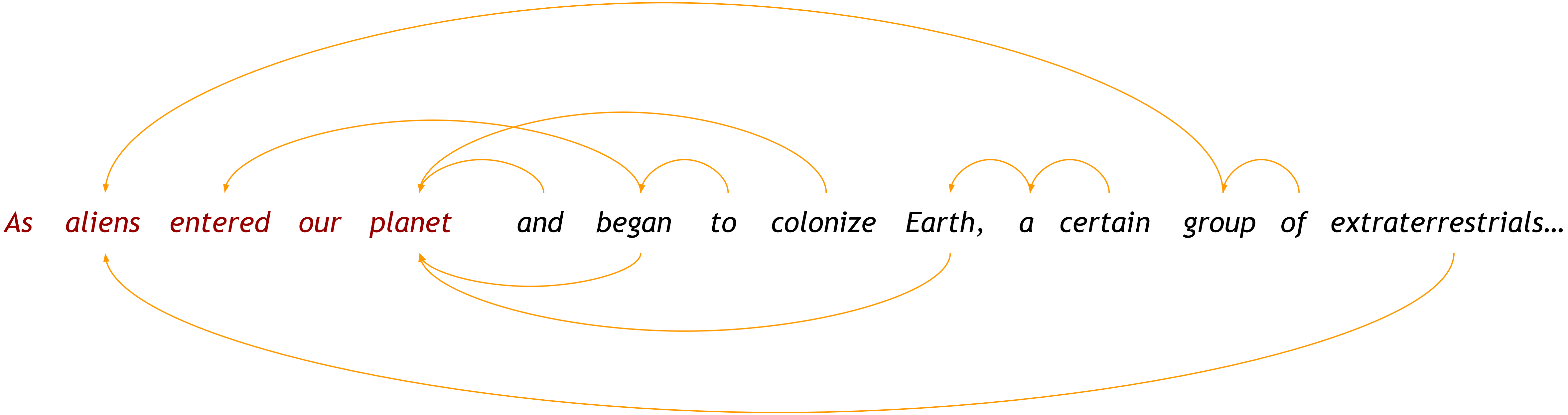


Text Generation Transformer

Previous Outputs

As aliens entered our planet

Text Generation Transformer



As aliens entered our planet and began to colonize Earth, a certain group of extraterrestrials…

Recurrent Neural Networks has a short reference window

As aliens entered our planet and began to colonize Earth, a certain group of extraterrestrials…

GRU’s & LSTM’s have a longer reference window than RNNs

As aliens entered our planet *and began to colonize Earth, a certain group of extraterrestrials…*

Attention mechanism has an infinite reference window

As aliens entered our planet and began to colonize Earth, a certain group of extraterrestrials…

THANK YOU