MATHEMATICS FOR MACHINE LEARNING

Nagoya University, Fall 2024

Lecture 11

Christmath Quiz

Language Models

Join here:

www.menti.com 1841 8986

https://www.henrikbachmann.com/mml2024.html

Semester project

Objective: Choose a project topic related to Nagoya or Japan more broadly that can be addressed using machine learning algorithms. Your task is to develop a machine learning model to solve a specific problem or provide insights into an aspect of life, business, environment, culture, etc., in Nagoya/Japan.

Group Size: 1-3 members

Code: Preferably a Google Colab notebook. Exceptions are possible; please provide full documentation for any different technology or package used. If you plan not to submit a Google Colab, please contact us in advance.

Documentation: 5-10 slides as if you were going to present the project.
Your slides should cover (for example):

- Problem Statement
- Data Collection
- Data Exploration and Visualization
- Model Building and Evaluation
- Conclusion

TACT Deadline: 19th January 2025

Presentation: 20th January 2025

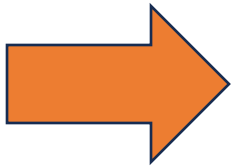
5-10 minutes each group

Language Model

A **language model** is a probabilistic model of a natural language.

Causal language modeling: Given a string of text a language model gives a probability distribution for the next word

"Today we are going to learn __"



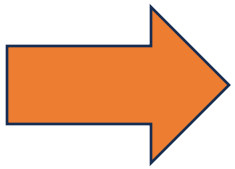
about	-> 21.16%
how	-> 13.53%
more	-> 8.25%
from	-> 7.53%
a	-> 7.10%
the	-> 6.03%
that	-> 4.74%
what	-> 3.68%
to	-> 1.81%
something	-> 1.77%
..	...%

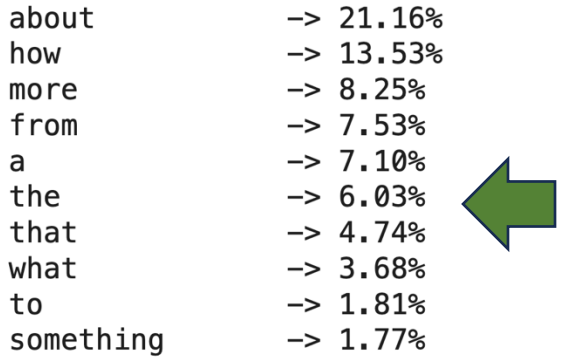
Language Model

A **language model** is a probabilistic model of a natural language.

Causal language modeling: Given a string of text a language model gives a probability distribution for the next word

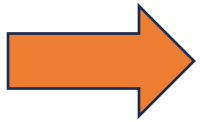
"Today we are going to learn __"





This kind of language models can be used to create text by choosing from the distribution the next word and then continue…

"Today we are going to learn the ___"



Language Models

Examples

- **n-Gram Models**
- Hidden Markov Models (HMM)
- Bag-of-Words (BoW) Model
- Latent Dirichlet Allocation (LDA)
- Word2Vec (CBOW and Skip-gram)
- GloVe (Global Vectors for Word Representation)
- Recurrent Neural Networks (RNNs)
- Long Short-Term Memory (LSTM)
- **Transformer Models**
- BERT (Bidirectional Encoder Representations from **Transformers**)
- GPT (Generative Pre-trained **Transformer**)

Naïve approach: n-Gram model

(Markov) Assumption: The next word just depends on the last n-1 words

"Today we are going to learn __"

n=2 (bigram) : The next word just depends on "learn"
n=3 : The next words just depends on "to learn"
…

Naïve approach: n-Gram model

(Markov) Assumption: The next word just depends on the last n-1 words

"Today we are going to learn __"

n=2 (bigram) : The next word just depends on "learn"
n=3 : The next words just depends on "to learn"
…

n-gram model

- We assume we have a given set of text (**corpus**)
- For a given sequence of words $w_1, \ldots, w_m$ we want to choose the next word.
- We choose the word where the following probability is the highest

$$P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{\text{count}(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{\text{count}(w_{i-(n-1)}, \ldots, w_{i-1})}$$

Counts how often this sequence appears in the corpus

We need to pay attention

- The n-gram model is too simple to be good.

Words can have different meaning depending on the context

"I watched the athletes **train** at the Nagoya Dome." "I took the **train** to Nagoya Station."

”We **train** the neural network." ”The red plastic **train** is my daughter's favorite."

We need to pay attention: Transformer

- The n-gram model is too simple to be good.

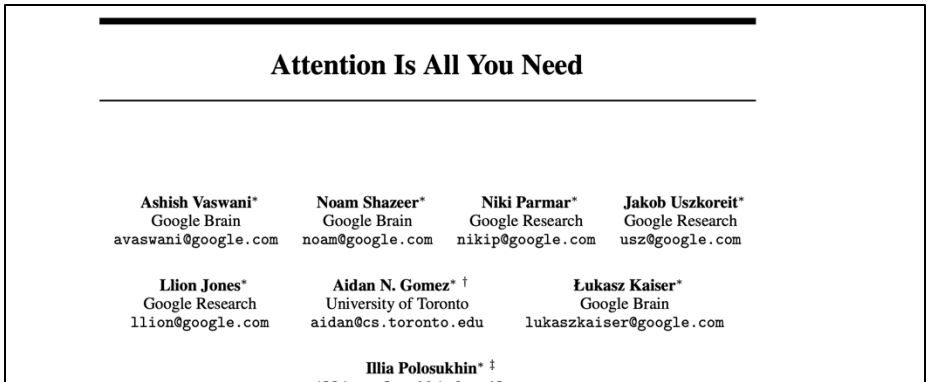
Words can have different meaning depending on the context

"I watched the athletes **train** at the Nagoya Dome." "I took the **train** to Nagoya Station."

”We **train** the neural network." ”The red plastic **train** is my daughter's favorite."

- Between n-gram and today's best language models exists a list of algorithm that we will not focus on (recurrent neural networks, long short-term memory, gated recurrent units)

- ChatGPT (GPT = Generative Pre-trained Transformer) use so-called **transformer**



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

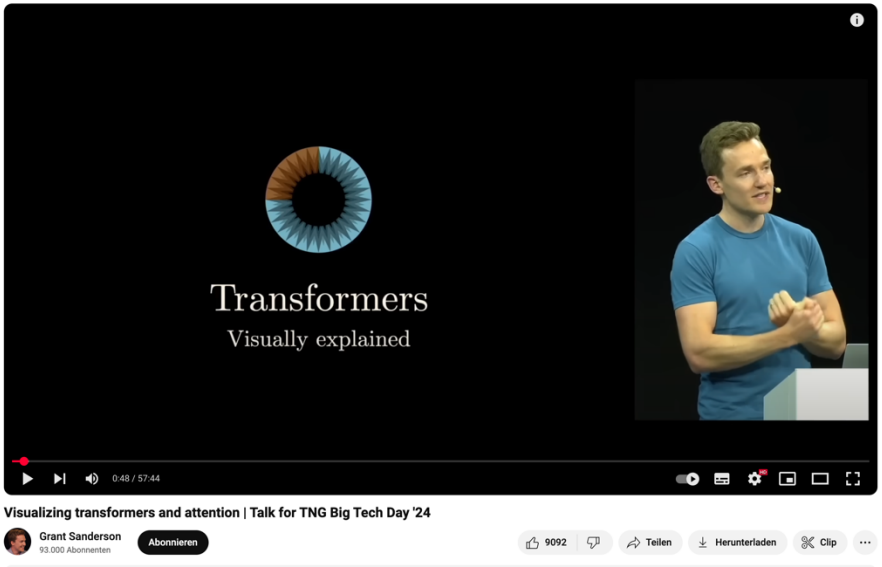
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡

We need to pay attention: Transformer

Recommendation for the holiday:

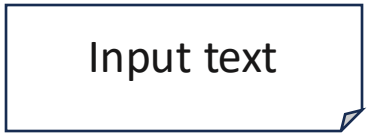


Grant Sanderson (3Blue1Brown):
Overview talk at TNG Big Tech Day 24

https://youtu.be/KJtZARuO3JY?si=ByqOZh0kwmokalx4

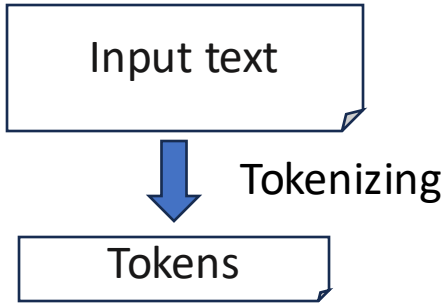
Based on the 3Blue1Brown Series on Machine Learning https://youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi&si=w03l9M9wSGGcgs-_

Transformer rough overview



”The red plastic **train** is my daughter's favorite."

Transformer rough overview



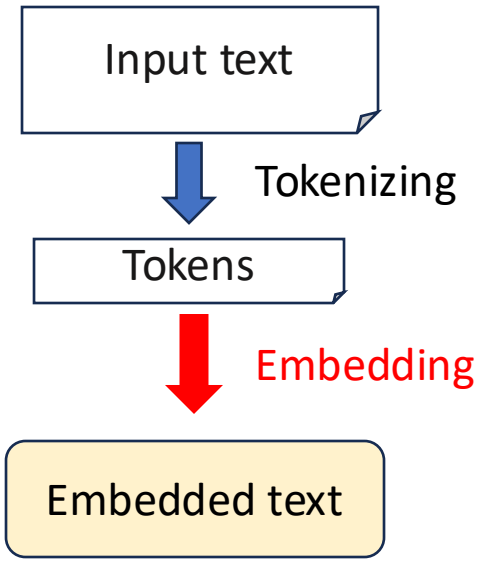
”The red plastic **train** is my daughter's favorite."

["The", " red", " plastic", " train", " is", " my", " daughter", "'s", " favorite", "."]

- Text is usually split into “tokens” rather than words.
- These tokens can be subword units (e.g., Byte Pair Encoding) that make it easier for the model to handle both common and rare words, as well as new vocabulary, by splitting them into smaller, more manageable pieces.

“antidisestablishmentarianism” → [" anti", "dis", "est", "ablish", "ment", "ari", "an", "ism"]

Transformer rough overview



”The red plastic **train** is my daughter's favorite."

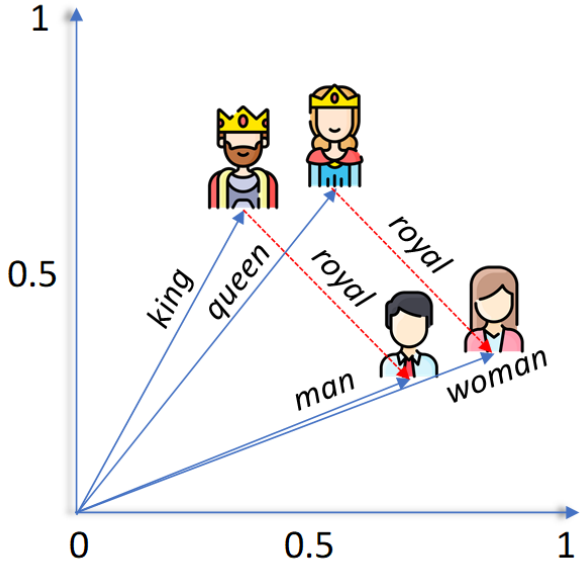
["The", " red", " plastic", " train", " is", " my", " daughter", "'s", " favorite", "."]

"The" ” red" ” plastic" ” train"

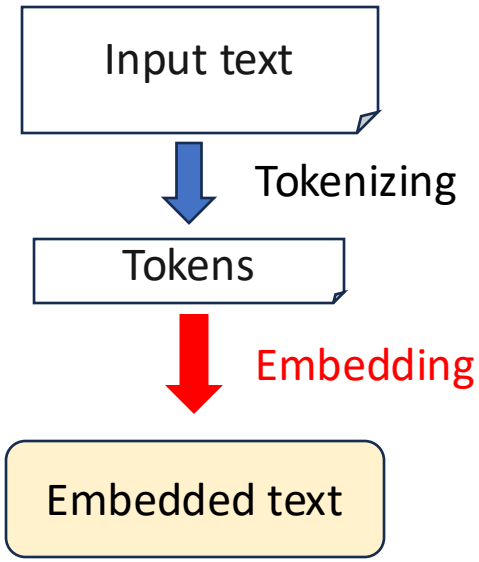
$$\begin{pmatrix}0.10\\0.20\\-0.30\end{pmatrix}, \begin{pmatrix}0.04\\0.14\\0.05\end{pmatrix}, \begin{pmatrix}-0.10\\0.24\\0.01\end{pmatrix}, \begin{pmatrix}0.16\\-0.05\\0.30\end{pmatrix}, \begin{pmatrix}0.03\\-0.12\\-0.09\end{pmatrix}, \begin{pmatrix}0.11\\-0.07\\0.22\end{pmatrix}, \begin{pmatrix}-0.03\\0.18\\0.09\end{pmatrix}, \begin{pmatrix}0.01\\0.08\\-0.04\end{pmatrix}, \begin{pmatrix}0.26\\-0.02\\0.17\end{pmatrix}, \begin{pmatrix}0.00\\0.00\\0.00\end{pmatrix}$$

Embedding: Transforms tokens into vectors (Example: GPT3 uses 12288 dimensions)

Position and directions in the embedding space have “meaning”



Transformer rough overview



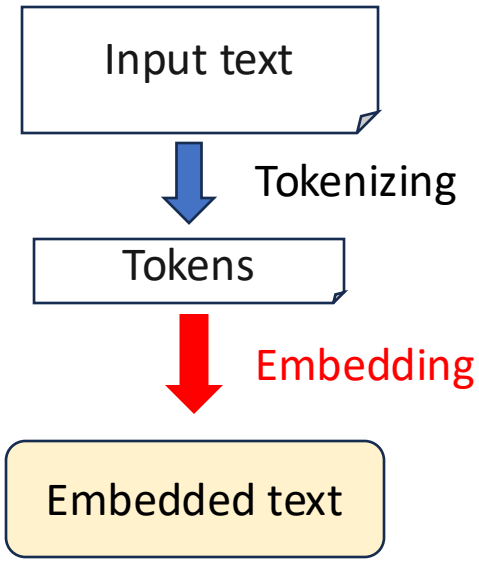
”The red plastic **train** is my daughter's favorite."

["The", " red", " plastic", " train", " is", " my", " daughter", "'s", " favorite", "."]

"The" ” red" ” plastic" ” train"

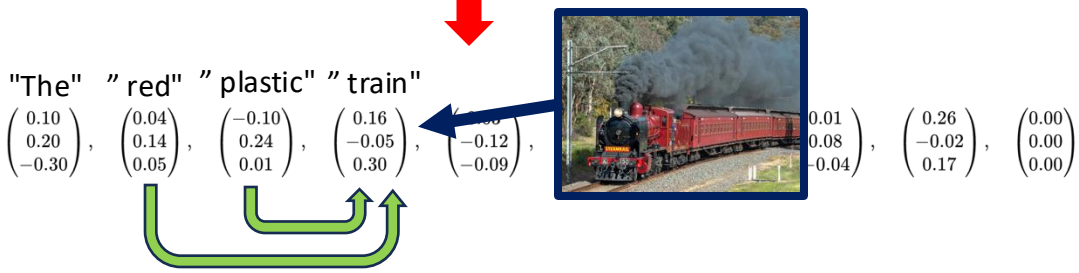
$$\begin{pmatrix}0.10\\0.20\\-0.30\end{pmatrix}, \begin{pmatrix}0.04\\0.14\\0.05\end{pmatrix}, \begin{pmatrix}-0.10\\0.24\\0.01\end{pmatrix}, \begin{pmatrix}0.16\\-0.05\\0.30\end{pmatrix}, \begin{pmatrix}\\-0.12\\-0.09\end{pmatrix}, \begin{pmatrix}0.01\\0.08\\-0.04\end{pmatrix}, \begin{pmatrix}0.26\\-0.02\\0.17\end{pmatrix}, \begin{pmatrix}0.00\\0.00\\0.00\end{pmatrix}$$

Transformer rough overview



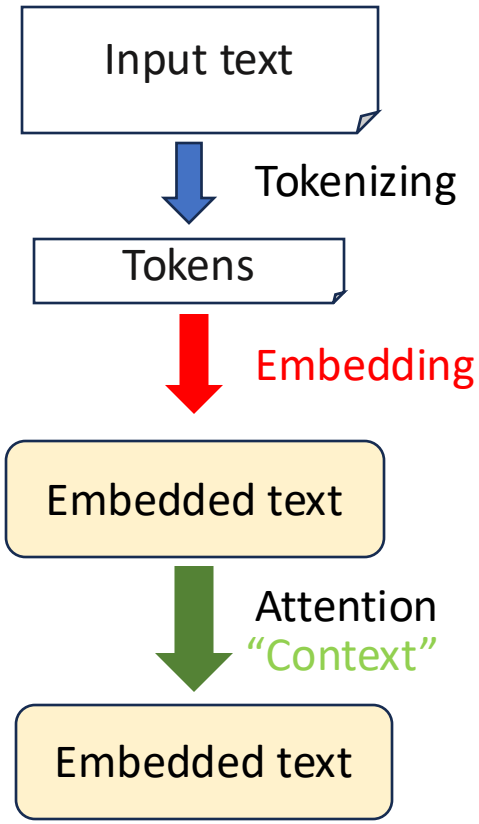
”The red plastic **train** is my daughter's favorite."

["The", " red", " plastic", " train", " is", " my", " daughter", "'s", " favorite", "."]



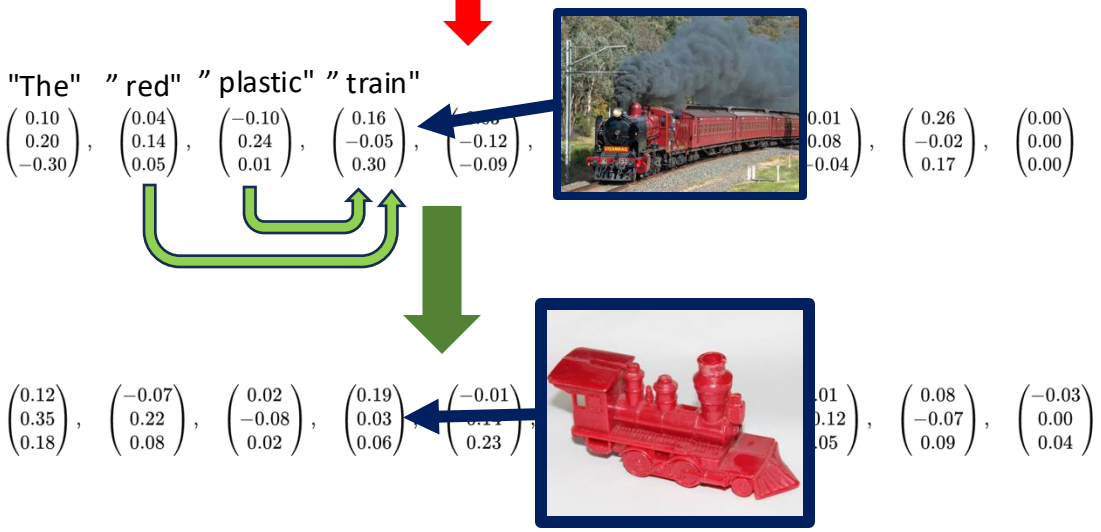
The word train changes it interpretation due to its context

Transformer rough overview

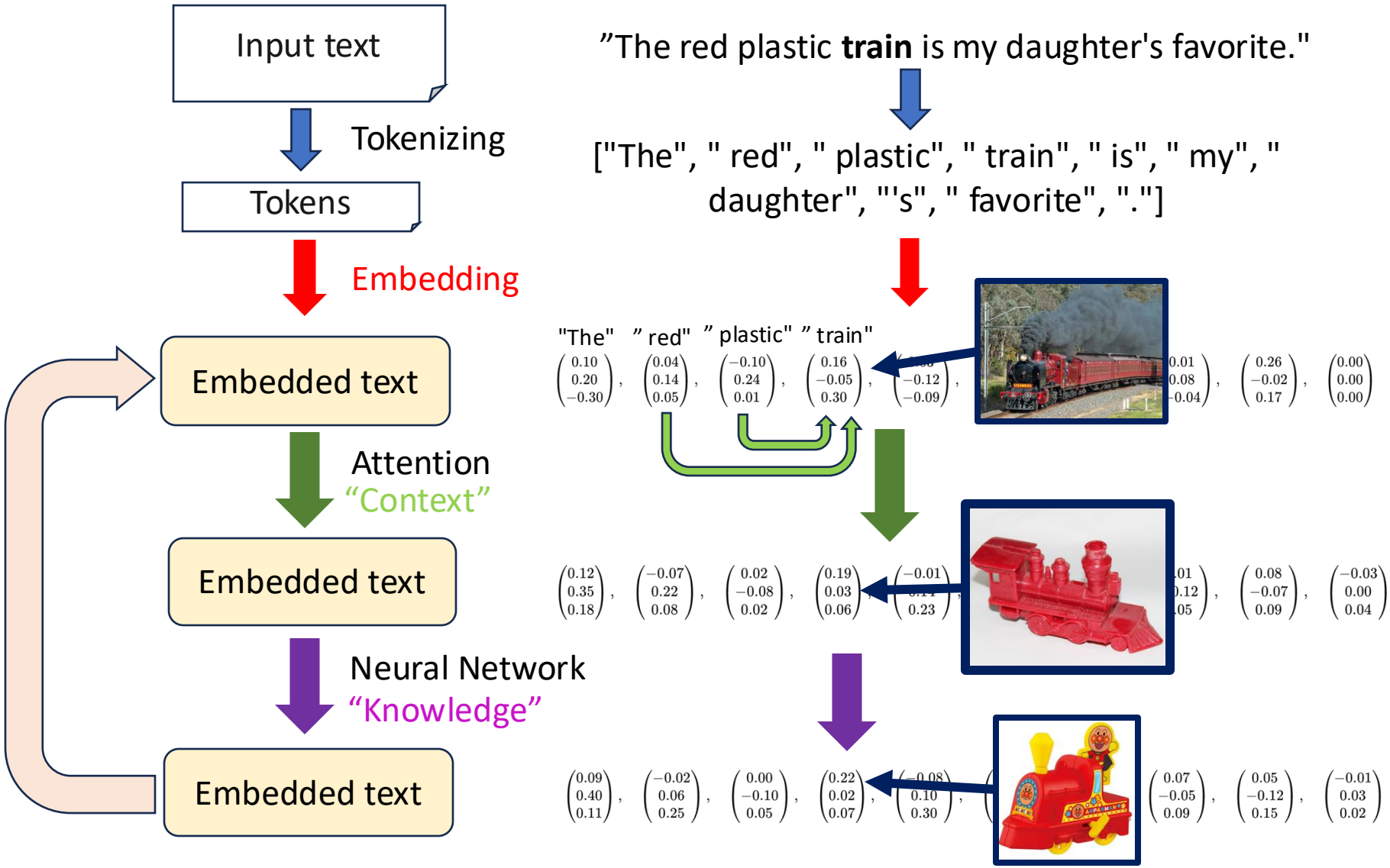


”The red plastic **train** is my daughter's favorite."

["The", " red", " plastic", " train", " is", " my", " daughter", "'s", " favorite", "."]



Transformer rough overview



Transformer rough overview