INF721

2024/2

UFV

Deep Learning

L18: Transformers (Part II)

Logistics

Announcements

- PA4 is due tonight! 11:59pm
- Midterm Exam II has been pushed back to Nov 25th

Last Lecture

- Problems with RNNs
- Transformers
  - Self-attention
  - Multi-head Attention
  - Positional Embedding
  - Masked Multi-head

UFV

Lecture Outline

- Contextual Word Embeddings
- General Pre-Training (GPT)
  - Fine-tuning
  - Gerating Text
  - ChatGPT
- Bidirectional Encoder Representations from Transformers (BERT)
  - Masked Language Model
  - Word and Sentence Embedding

UFV

Contextual Word Embeddings

Models like Word2Vec and Glove learn a static Embedding Matrix E, so they can't consider the context of a word to produce it's embedded vector.

$E =$

Man	Woman	King	Queen	Apple	Orange
-1	1	-0.95	0.97	0.00	0.01
0.01	0.02	0.93	0.95	-0.01	0.00
0.03	0.02	0.7	0.69	0.03	-0.02
0.04	0.01	0.02	0.01	0.95	0.97

$$E \cdot o_{6257} = \underbrace{\begin{bmatrix} 0.01 \\ 0.00 \\ -0.02 \\ 0.97 \\ \dots \end{bmatrix}}_{e_{6257}}$$

- For example, the word **apple** will have the same representation in both sentences:

  *"I want a glass of **apple** juice"*

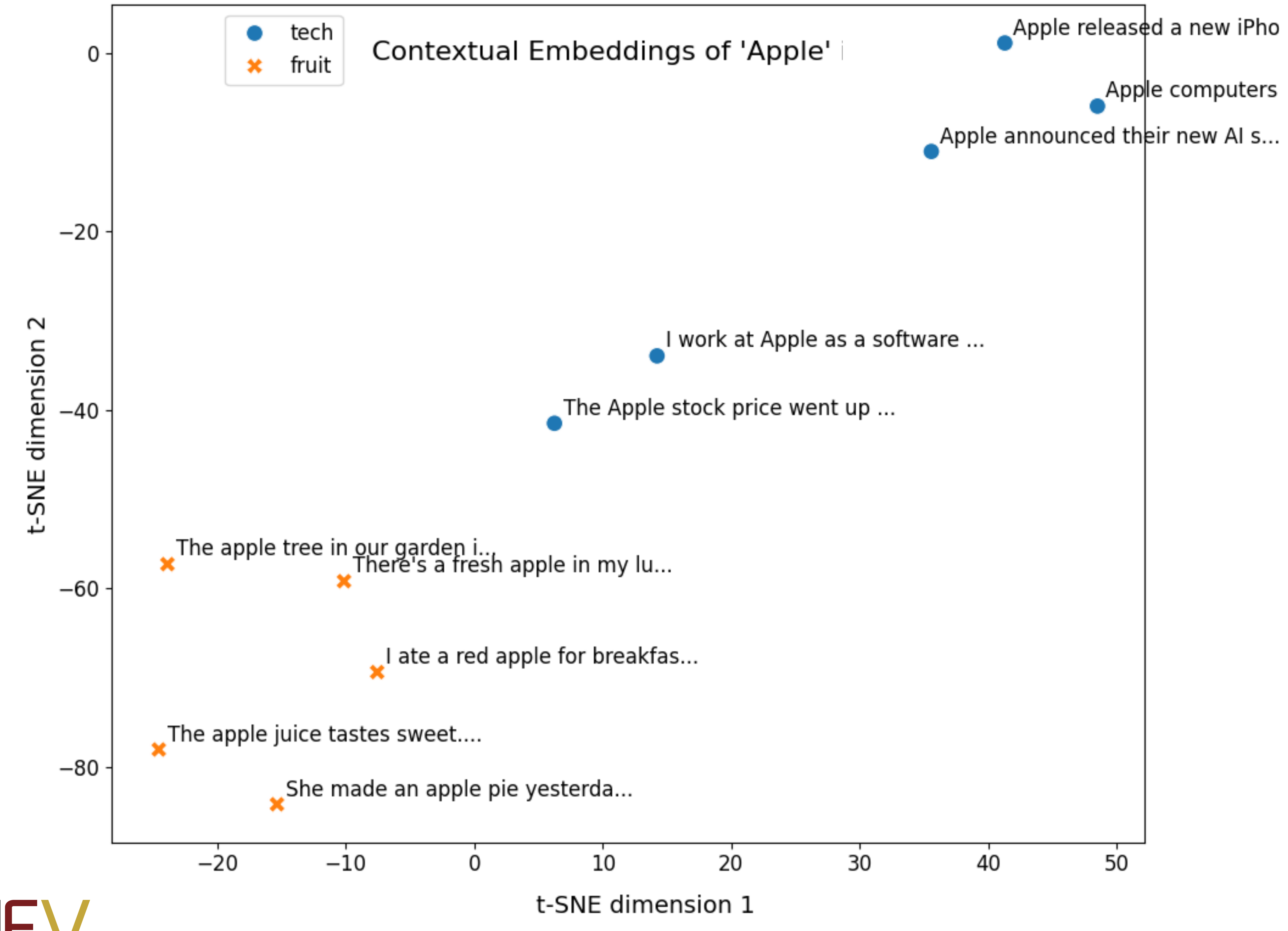
  *"I work at **apple**"*

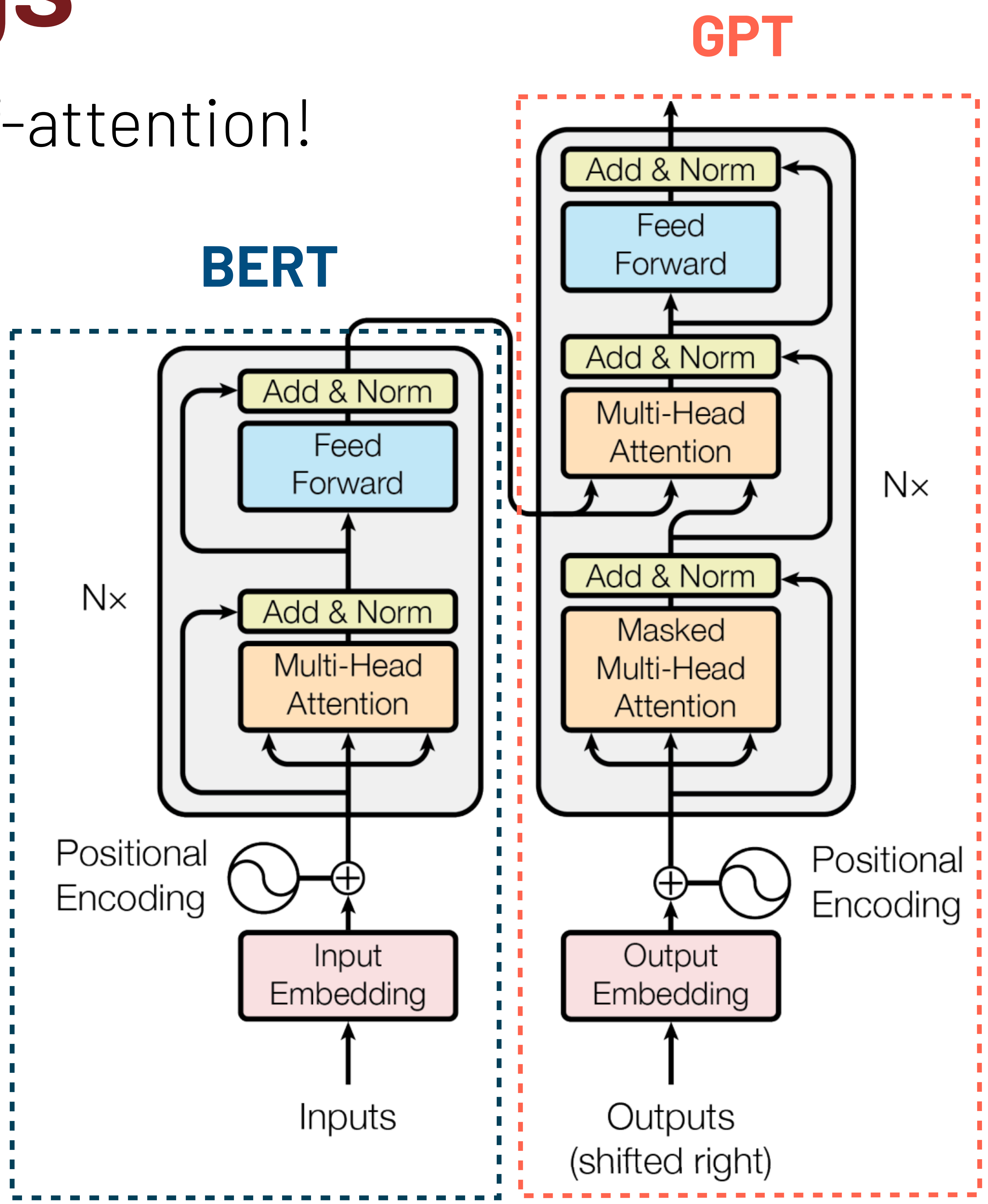
- But they have completely different meanings, because of their different context.

UFV

Contextual Word Embeddings

Transformers learn **contextual representations** via self-attention!





We can learn contextual embeddings using either the **Encoder** or the **Decoder** to train a Language Model!

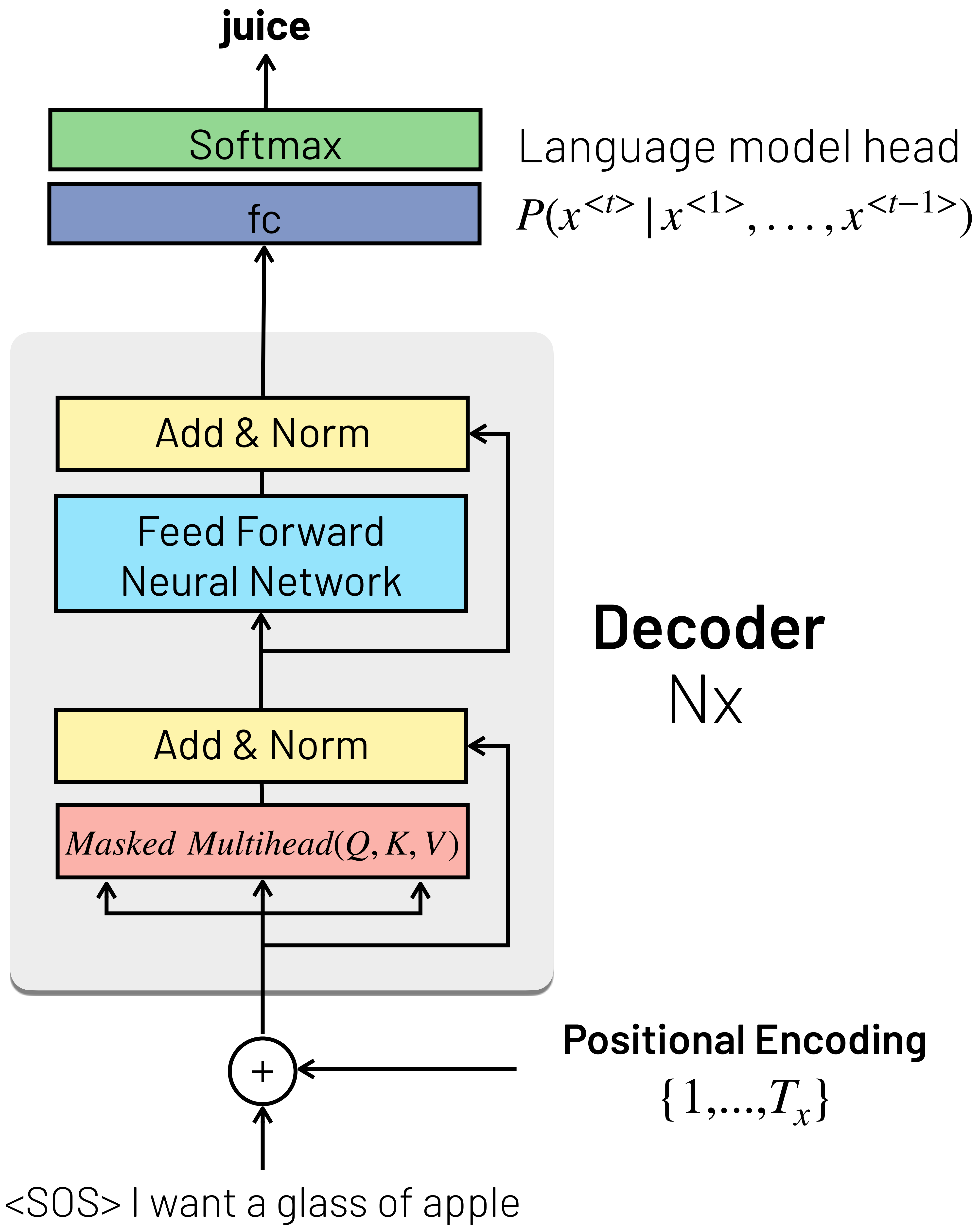
UFV

General Pre-Training (GPT)

Language Model based on the **Transformer Decoder**

1. **Train a language model** in unlabelled text

No need for another Multihead attention layer because we are not doing translation!

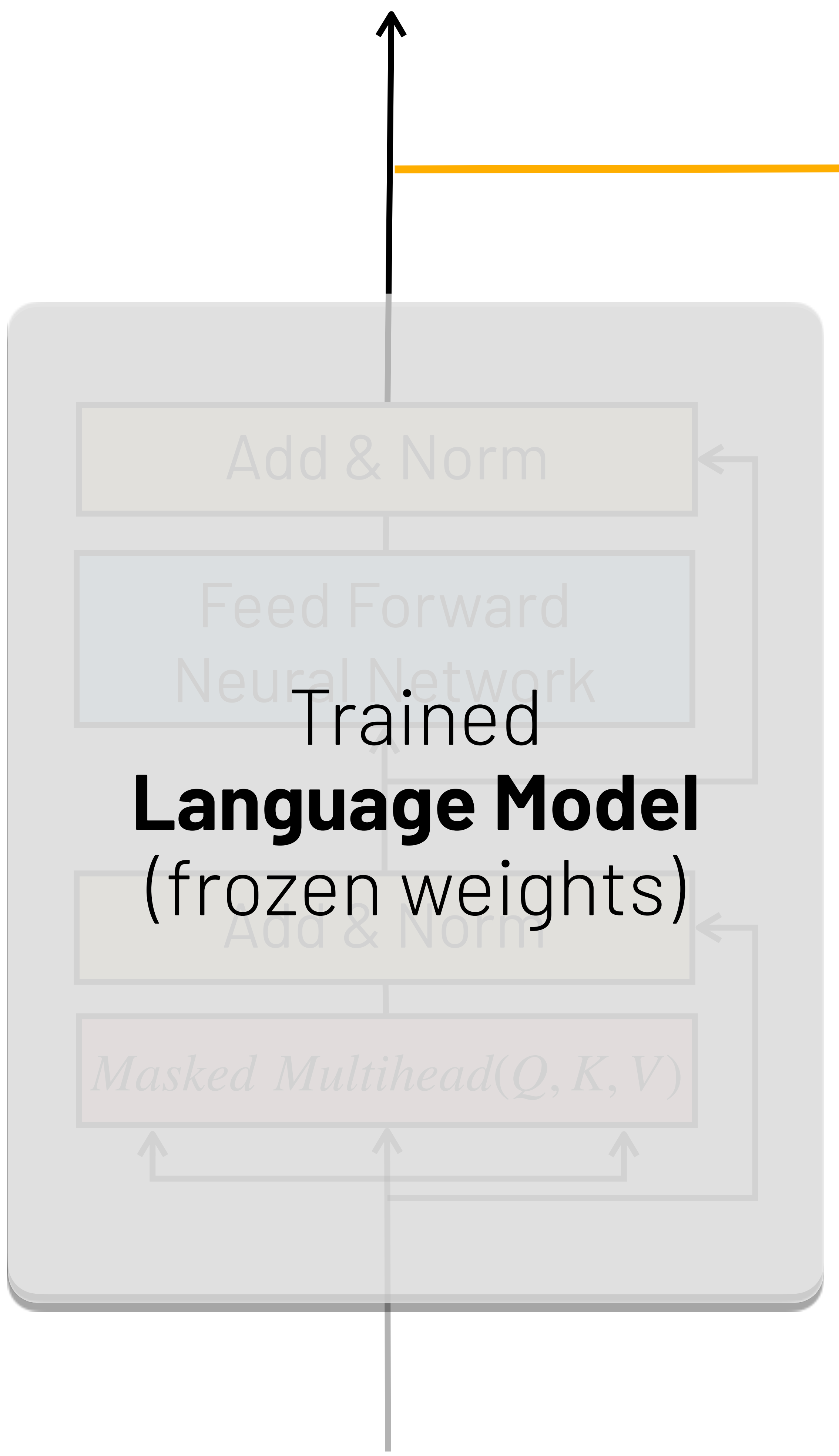


Radford et al. 2018, Improving Language Understanding by Generative Pre-Training

General Pre-Training (GPT)

Language Model based on the **Transformer Decoder**

1. Train a language model in unlabelled text
2. **Freeze the model** up to the last transformer block and use it **to extract contextual embeddings**



$c = \{c^{<1>}, \ldots, c^{<7>}\}$

The output of the last decoder block is the **contextual embeddings**

Decoder
Nx

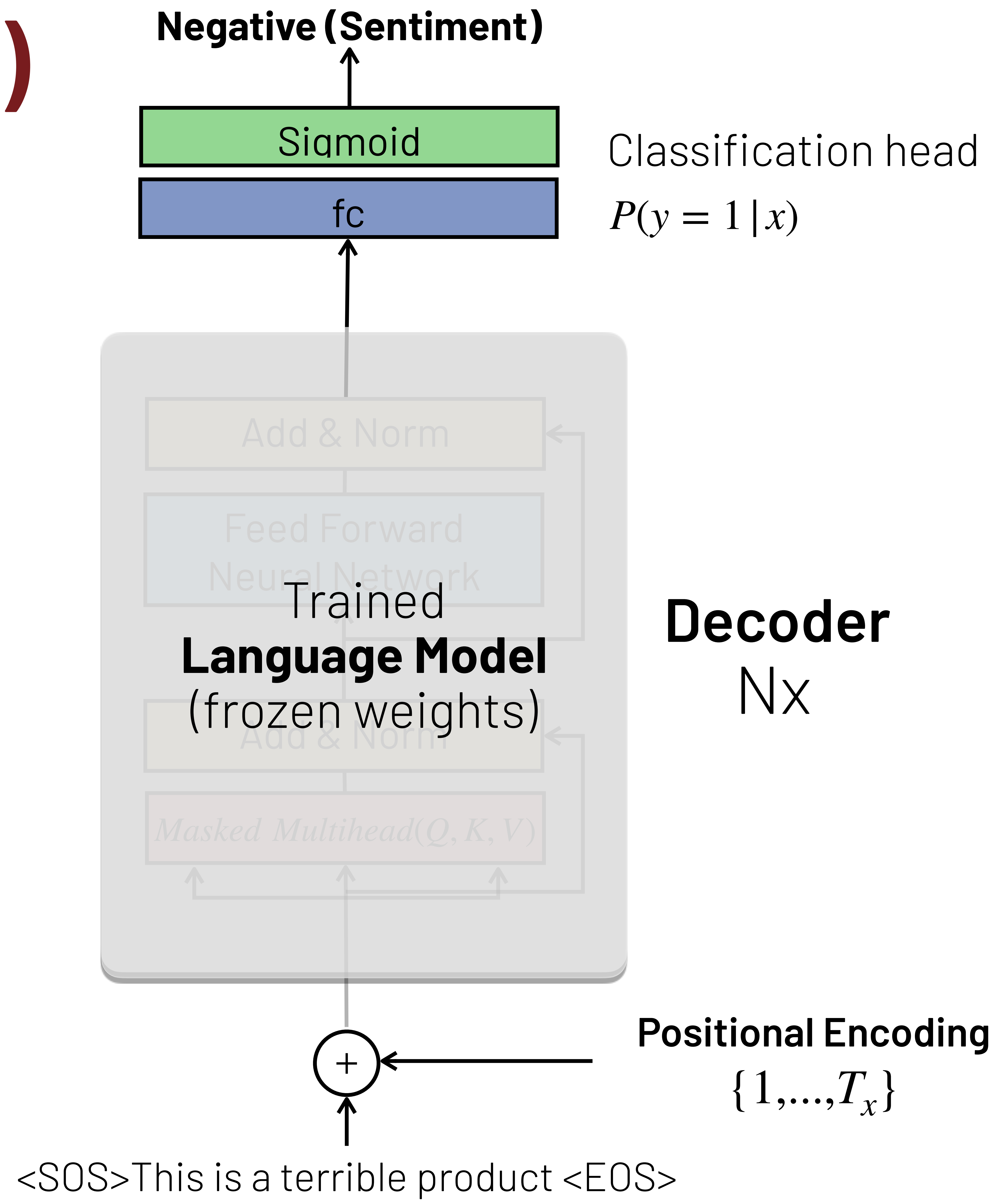
UFV

Radford et al. 2018, Improving Language Understanding by Generative Pre-Training

General Pre-Training (GPT)

Language Model based on the **Transformer Decoder**

1. Train a language model in unlabelled text
2. Freeze the model up to the last transformer block and use it to extract contextual embeddings
3. **Substitute** the Language Model head by a **Classification** head (e.g., sentiment analysis)



Radford et al. 2018, Improving Language Understanding by Generative Pre-Training

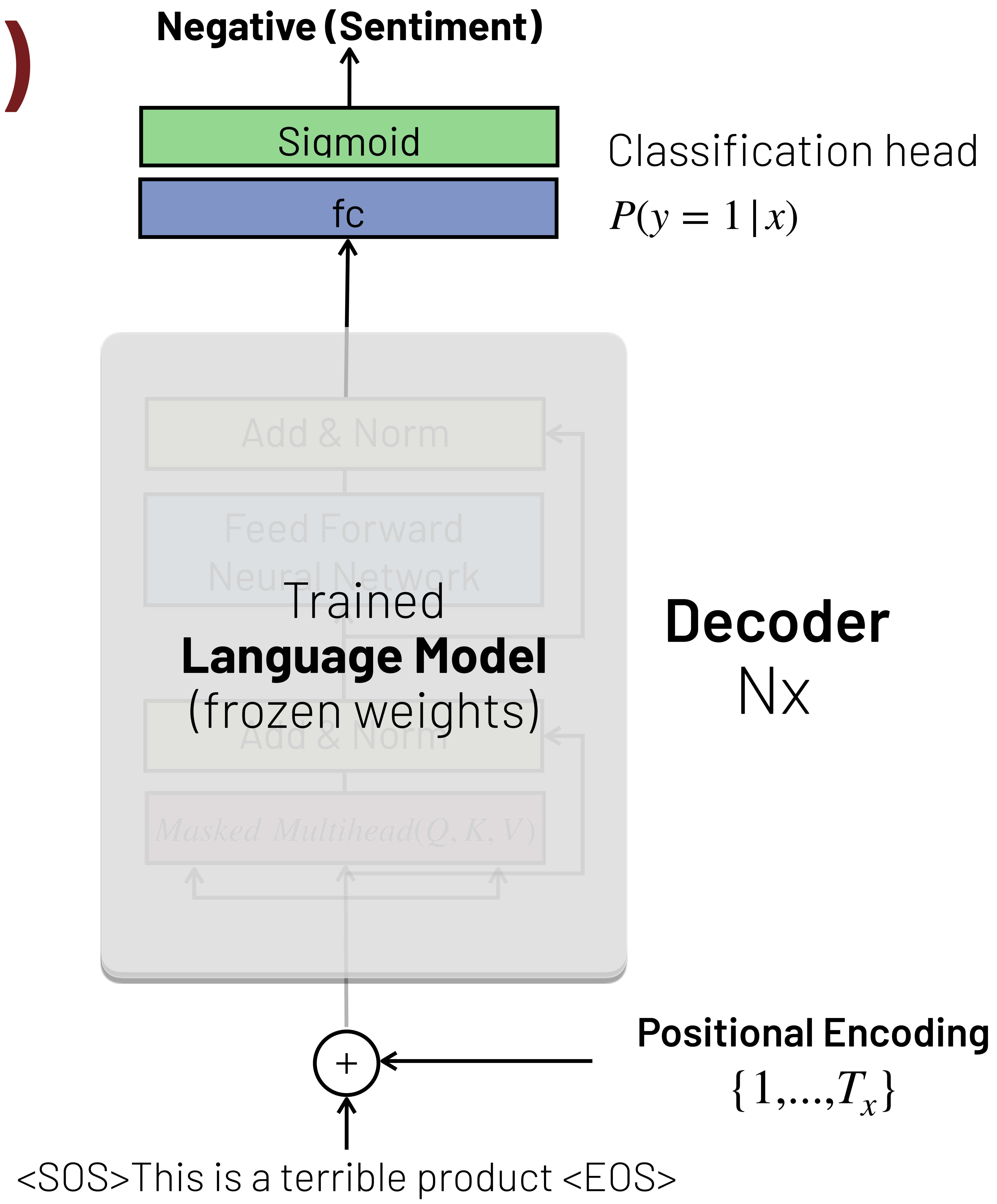
General Pre-Training (GPT)

Language Model based on the **Transformer Decoder**

1. Train a language model in unlabelled text
2. Freeze the model up to the last transformer block and use it to extract contextual embeddings
3. Substitute the Language Model head by a Classification head (e.g., sentiment analysis)

Step 1. is called **pre-training!**

Step 3. is called **fine-tuning!**



Radford et al. 2018, Improving Language Understanding by Generative Pre-Training

Bytepair Encoding (BTE)

GPT (and many other transformer models) uses the Byteapair Encoding (BPE) algorithm to optimize the English Vocabulary, turning comon patters into single tokens.

- BPE is like building a dictionary of common word pieces by repeatedly combining the most frequent pairs of characters or subwords.

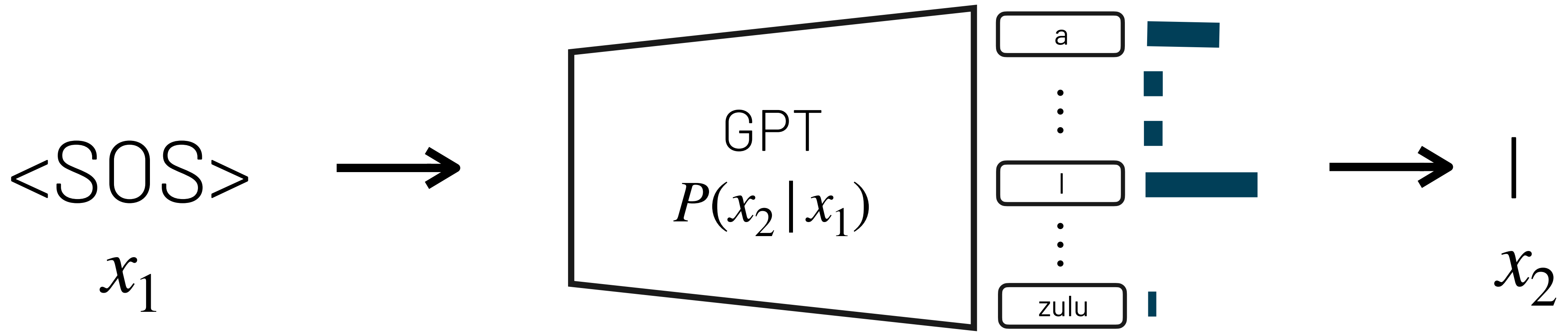
```
Text: "low lower lowest”

Initial: l o w _ l o w e r _ l o w e s t

Merges:
  1. (l,o)  → "lo": "lo w _ lo w e r _ lo w e s t"
  2. (lo,w) → "low": "low _ low e r _ low e s t"
  3. (e,r)  → "er": "low _ low er _ low e s t"
```

UFV

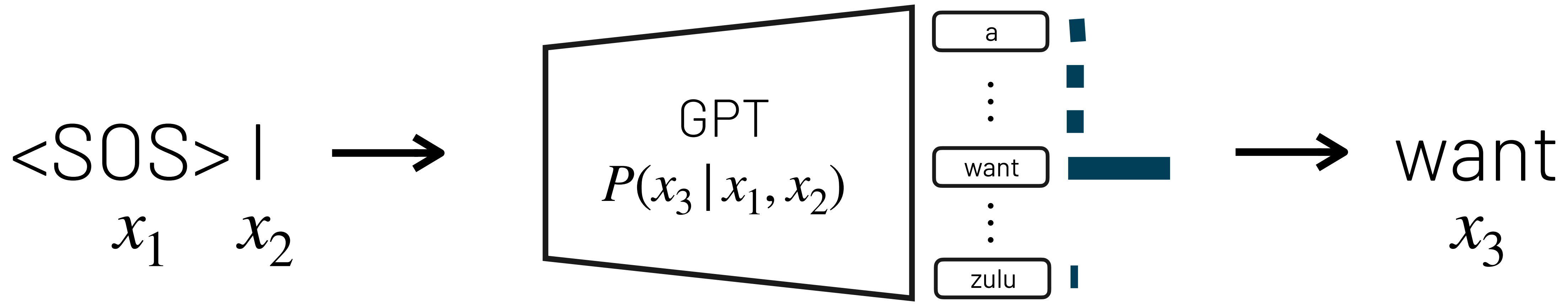
Generating Text with GPT

GPT is a language model and thus it can generate text similarly to RNN-based language models

1. Start your sequence with $x_1 =$ <SOS>
2. Sample the next word using the probability distribution given by GPT:



3. Concatenate the sampled word x_2 to your current sentence and sample from the model again:



Repeat steps 2. and 3. until the $< EOS >$ token is sampled.

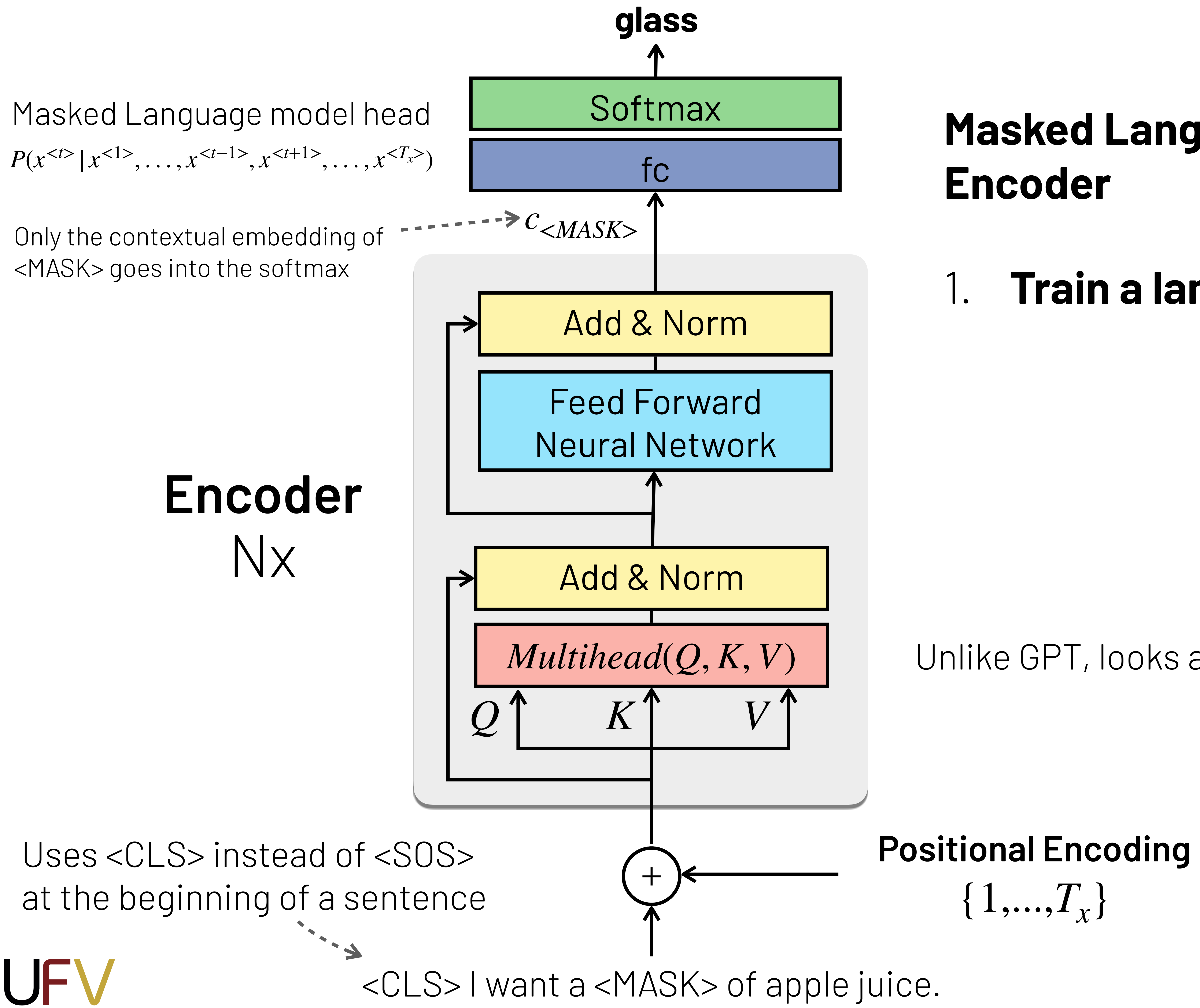
UFV

GPT Evolution Over Time

	GPT 1 (2018)	GPT 2 (2019)	GPT 3 (2020)	GPT 3.5 (2022)	GPT 4 (2023)
Parameters	117M	1.5B	175B	~175B	Estimated ~1.8T
Nº of Decoder Blocks	12	48	96	~96	Unknown
Pre-training Dataset	5GB (~0.5B Tokens)	40GB (~8B tokens)	570 GB (~300B tokens)	570 GB (~300B tokens)	Unknown
Main Contribution	Introduced generative pretraining for transformers, showing decent results on text classification and sentiment analysis.	Generated coherent long-form text and exhibited surprising zero-shot learning abilities, like translation and summarization without specific task training.	Showed impressive few-shot, zero-shot, and multi-task learning capabilities. Could perform tasks like question-answering, text generation, and even code generation.	Improved performance in language understanding and generation. Reduced bias, hallucination, and increased coherence in long outputs.	Demonstrated strong multi-modal capabilities (image and text input), exhibited higher reasoning abilities, and improved complex task handling.

Radford et al. 2018, Improving Language Understanding by Generative Pre-Training
Radford et al. 2019, Language Models are Unsupervised Multitask Learners
Brown et al. 2020, Language Models are Few-Shot Learners
OpenAI. 2023, GPT-4 Technical Report

Bidirectional Encoder Representations from Transformers (BERT)



Masked Language Model based on the **Transformer Encoder**

1. **Train a language model** in unlabelled text

Unlike GPT, looks at both left and right context

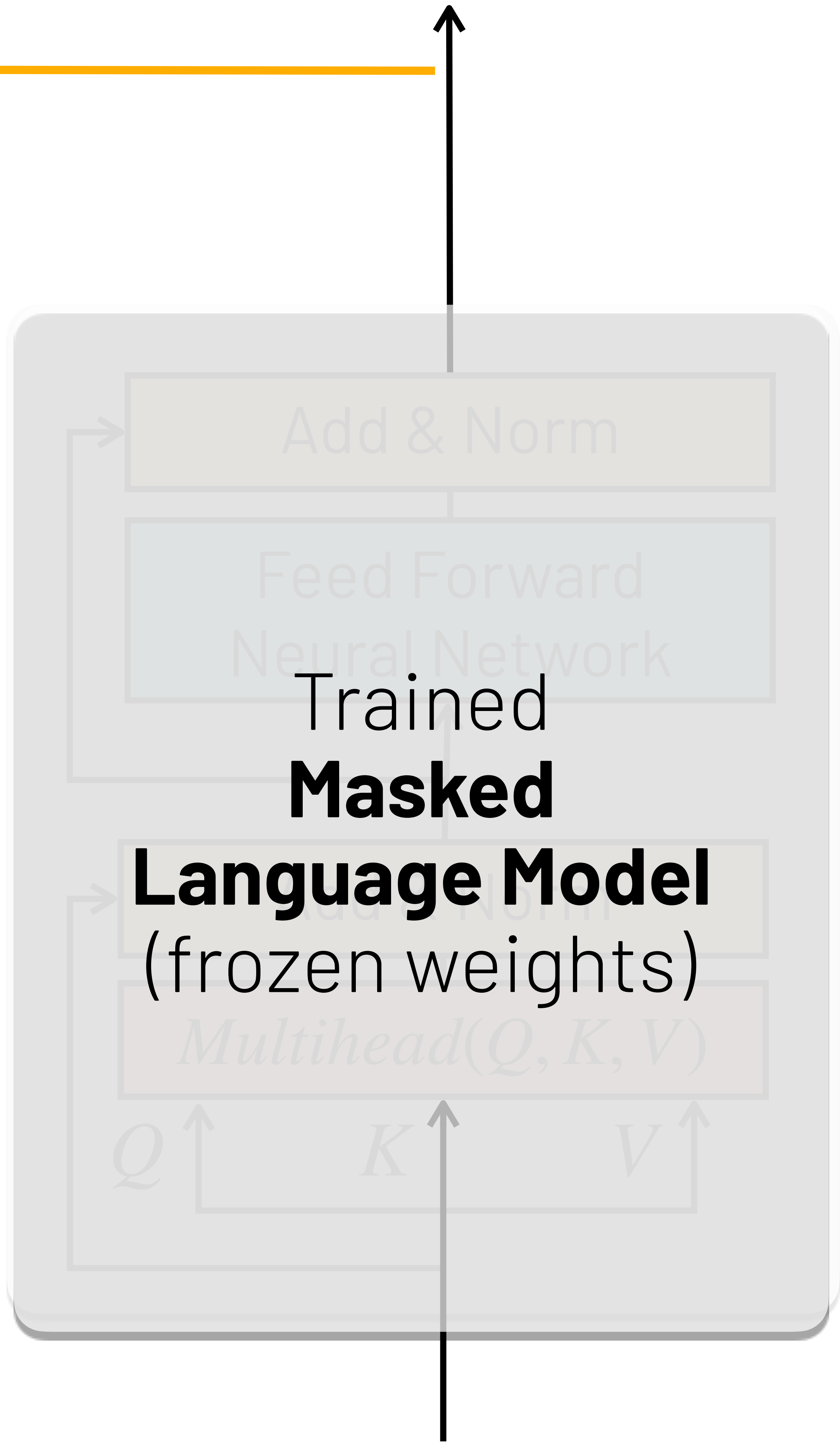
Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Bidirectional Encoder Representations from Transformers (BERT)

$$c = \{c^{<1>}, \ldots, c^{<7>}\}$$

The output of the last encoder block is the **contextual embeddings**

Encoder
Nx



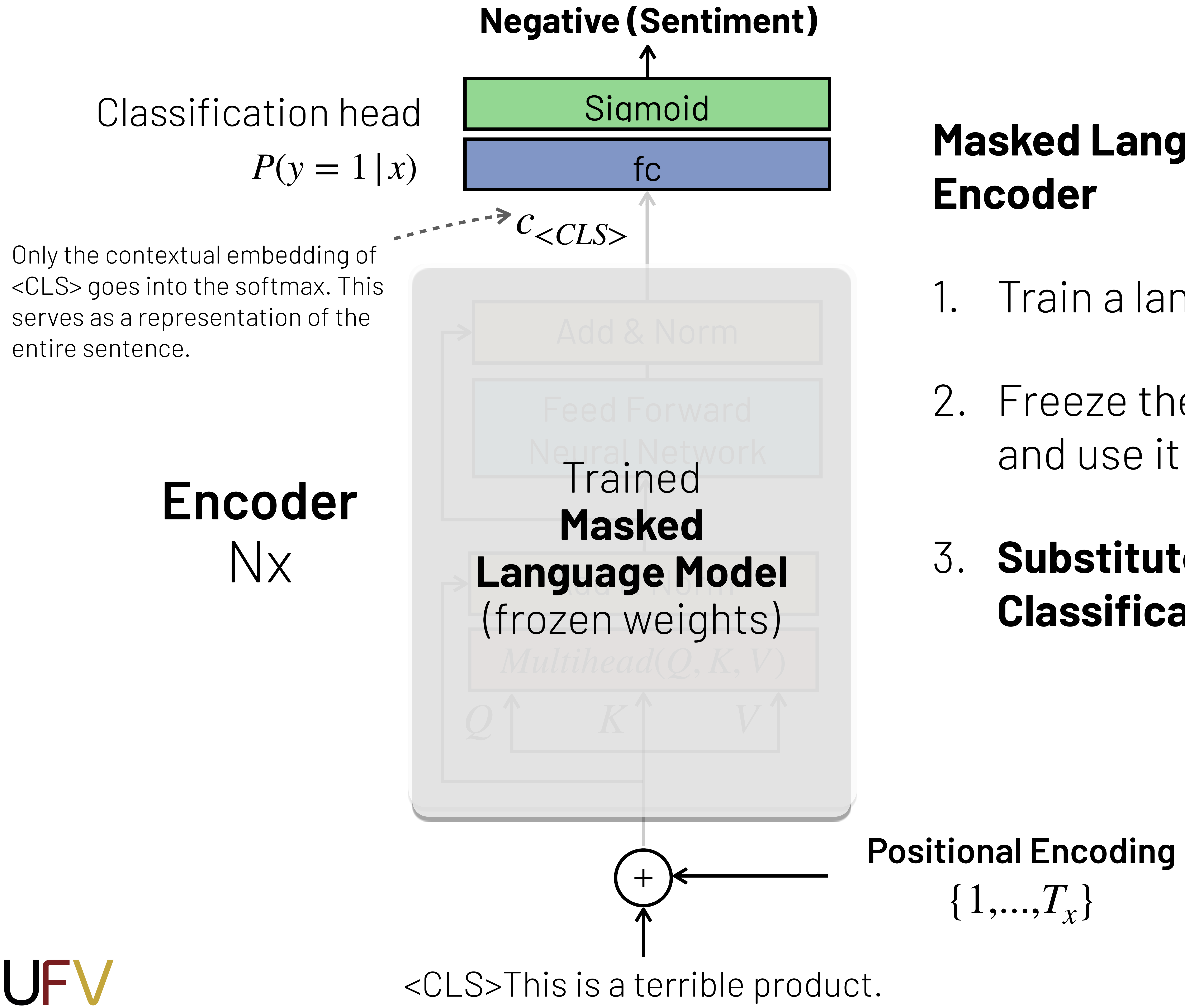
Masked Language Model based on the **Transformer Encoder**

1. Train a language model in unlabelled text
2. **Freeze the model** up to the last transformer block and use it t**o extract contextual embeddings**

UFV

Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Bidirectional Encoder Representations from Transformers (BERT)

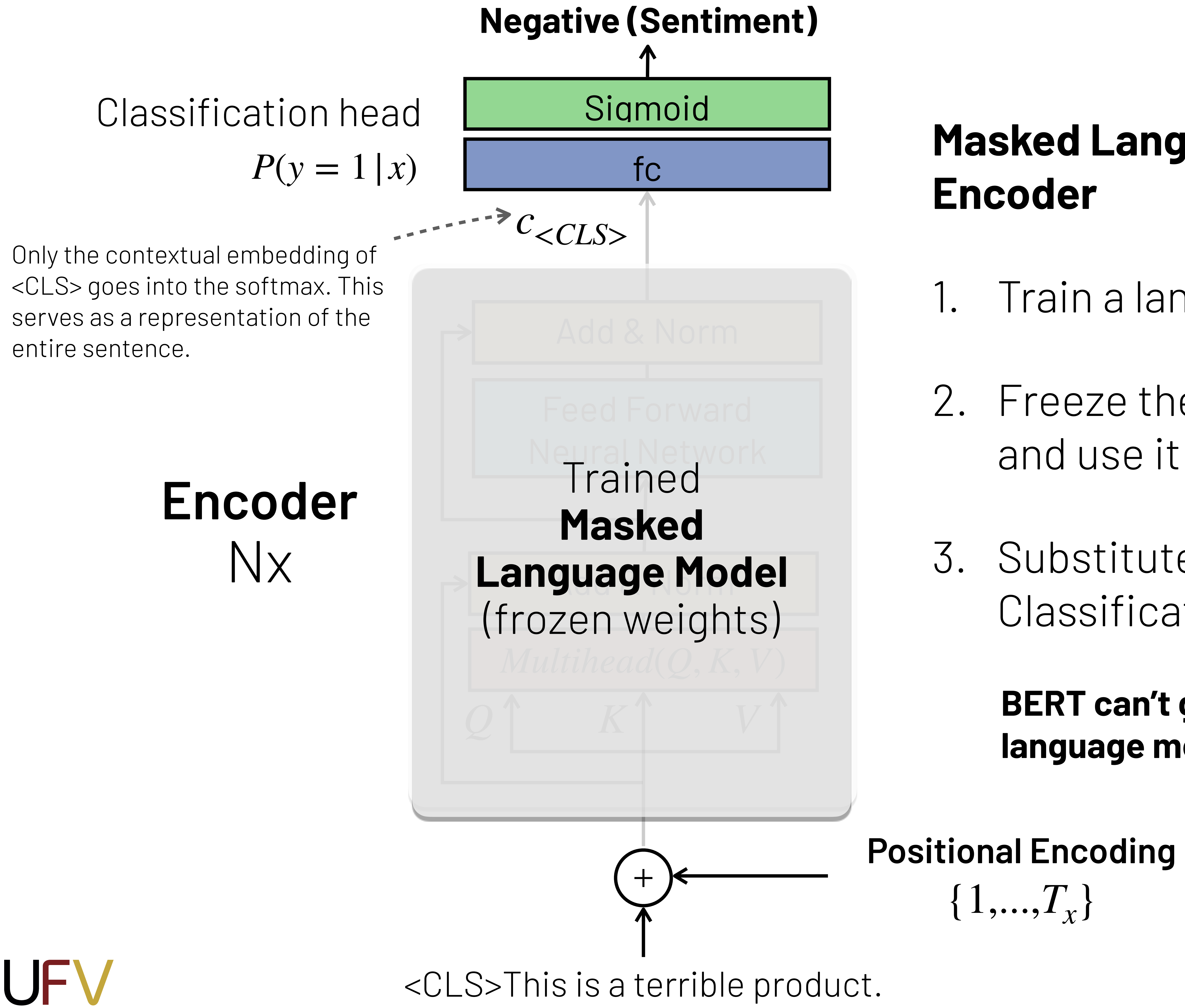


Masked Language Model based on the **Transformer Encoder**

1. Train a language model in unlabelled text
2. Freeze the model up to the last transformer block and use it to extract contextual embeddings
3. **Substitute** the Language Model head by a **Classification** head (e.g., sentiment analysis)

Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Bidirectional Encoder Representations from Transformers (BERT)



Masked Language Model based on the **Transformer Encoder**

1. Train a language model in unlabelled text

2. Freeze the model up to the last transformer block and use it to extract contextual embeddings

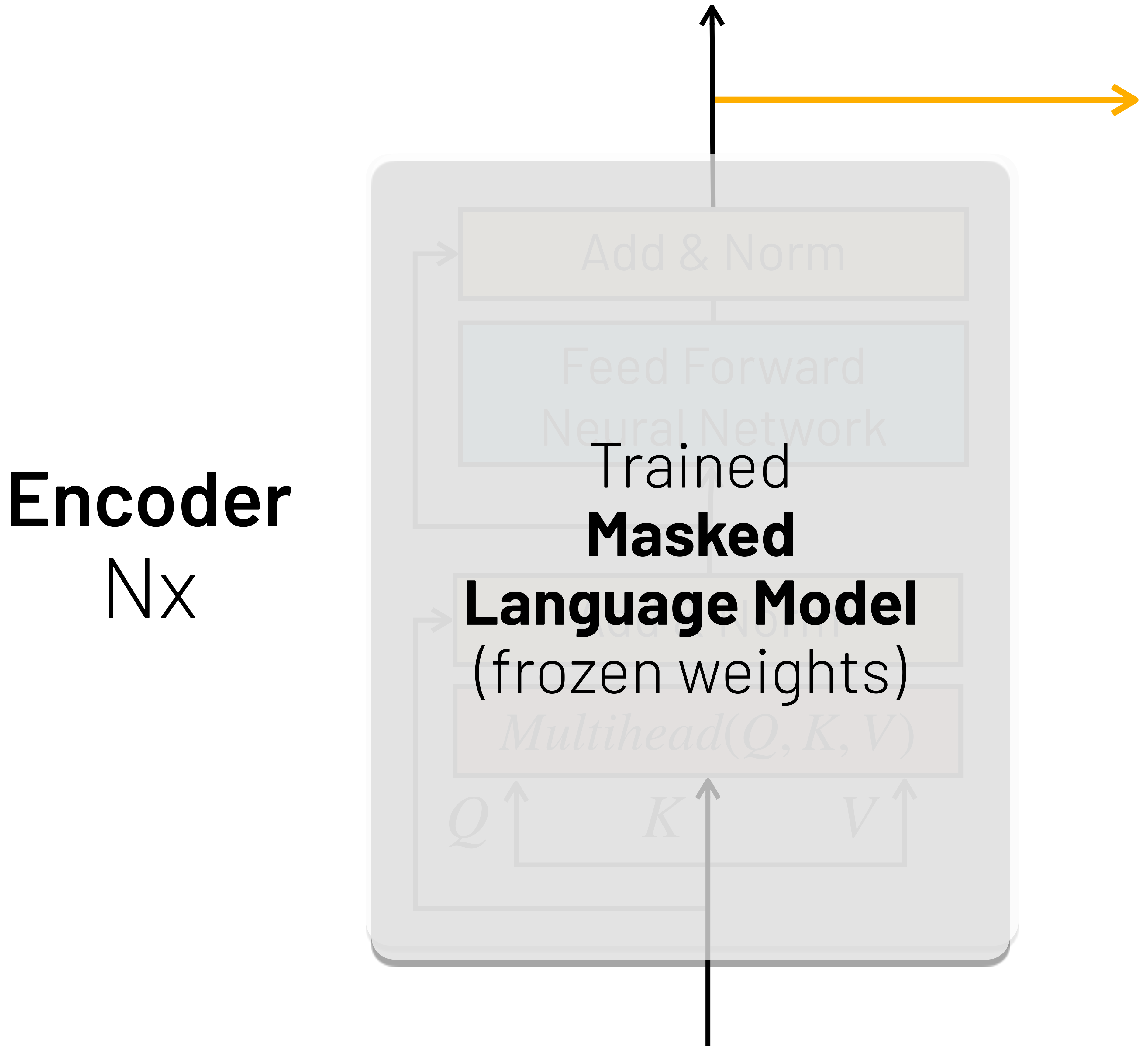
3. Substitute the Language Model head by a Classification head (e.g., sentiment analysis)

BERT can't generate text because it is not a regular language model!

Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Extracting BERT Contextual Embeddings

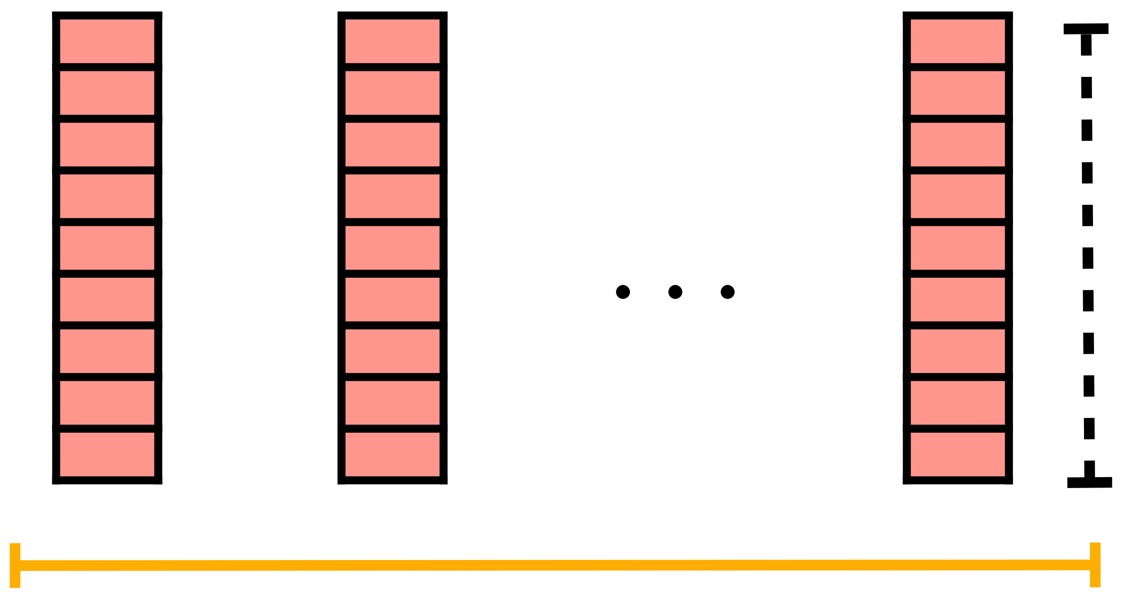
There are different methods to extract contextual word embeddings from BERT



<CLS>I want a glass of apple juice.

$$c = [c^{<1>}, c^{<2>}, \ldots, c^{<9>}]$$

The output of the last encoder block is the **contextual embeddings**



$$d_{model} = 9$$

- Last Hidden State (one vector per word)
- Mean Pooling (one vector per sentence)
- <CLS> Token Embedding (one vector per sentence)
- All Layer Embeddings
- ...

BERT Evolution Over Time

	BERT (Base)	BERT (Large)	RoBERTa	DistilBERT	ALBERT (Large)
Parameters	110M	340M	355M	66M	18M
Nº of Decoder Blocks	12 encoder layers, 12 attention heads	24 encoder layers, 16 attention heads	24 encoder layers, 16 attention heads	6 encoder layers, 12 attention heads	12 encoder layers, 12 attention heads (shared weights)
Pre-training Dataset	16 GB (~3.3B tokens)	16 GB (~3.3B tokens)	160 GB (~33 B tokens)	16 GB (~3.3B tokens)	16 GB (~3.3B tokens)
Main Contribution	Introduced bidirectional pretraining, greatly improved performance on NLP benchmarks like GLUE, SQuAD, and others. It became a foundation for many downstream tasks.	Same as BERT Base but with higher capacity, resulting in improved performance across NLP tasks, though with greater computational cost.	Tweaked BERT's training process (e.g., longer training), resulting in better performance on NLP benchmarks. Achieved state-of-the-art results on many tasks.	A distilled version of BERT, with 40% fewer parameters and 60% faster inference, while retaining 97% of BERT's performance on downstream tasks. Efficient for real-time applications.	Optimized for parameter efficiency by sharing layers and factorizing embedding parameters. Achieved performance close to BERT Large with significantly fewer parameters.

Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
Liu et al. 2019, RoBERTa: A Robustly Optimized BERT Pretraining Approach
Sanh et al. 2019 , DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter
Lan et al. 2019, ALBERT: A Lite BERT for Self-supervised Learning of Language Representations

BERTimbau

BERTimbau is a bert model especially pretrained for the Brazillian Portuguese language:

	BERTimbau Base	**BERTimbau Large**
Parameters	117M	1.5B
Nº of Decoder Blocks	12	48
Pre-training Dataset	5GB (~0.5B Tokens)	40GB (~8B tokens)
Pre-training Dataset	First large-scale pre-trained model for Brazilian Portuguese. Comparable to BERT base for Portuguese language tasks.	Larger version with increased capacity, achieving better performance on tasks requiring more linguistic nuance and complex text understanding.

Vocabulary specifically optimized for Portuguese morphology (and includes Portuguese-specific tokens and accents)

Souza F., Nogueira R. and Lotufo R. 2020, BERTimbau: Pretrained BERT Models for Brazilian Portuguese

Next Lecture

L19: Transfer Learning

Exploiting large unlabelled dataset to improve performance of supervised learning models