

# Inside a NLP multitasking neural network. BERT, one model to rule them all

Bachelor Thesis in Telecommunication Technologies Engineering

Ion Bueno Ulacia

**uc3m** | Universidad **Carlos III** de Madrid

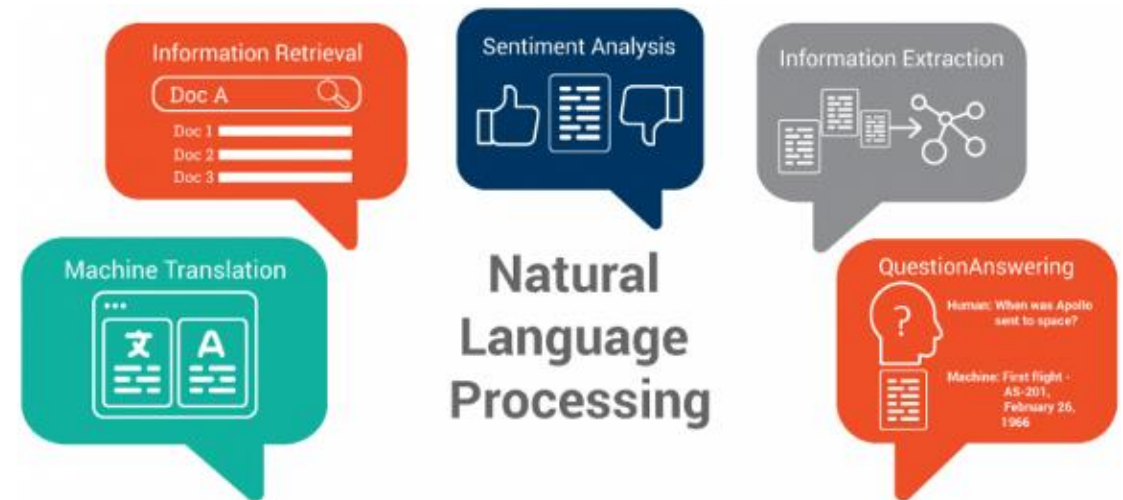


# Contents

- Background
  - Natural Language Processing (NLP)
  - Recurrent Neural Networks (RNNs)
  - The Transformer
- BERT implementations
  - Sentiment Analysis
  - Question Answering
- Conclusion

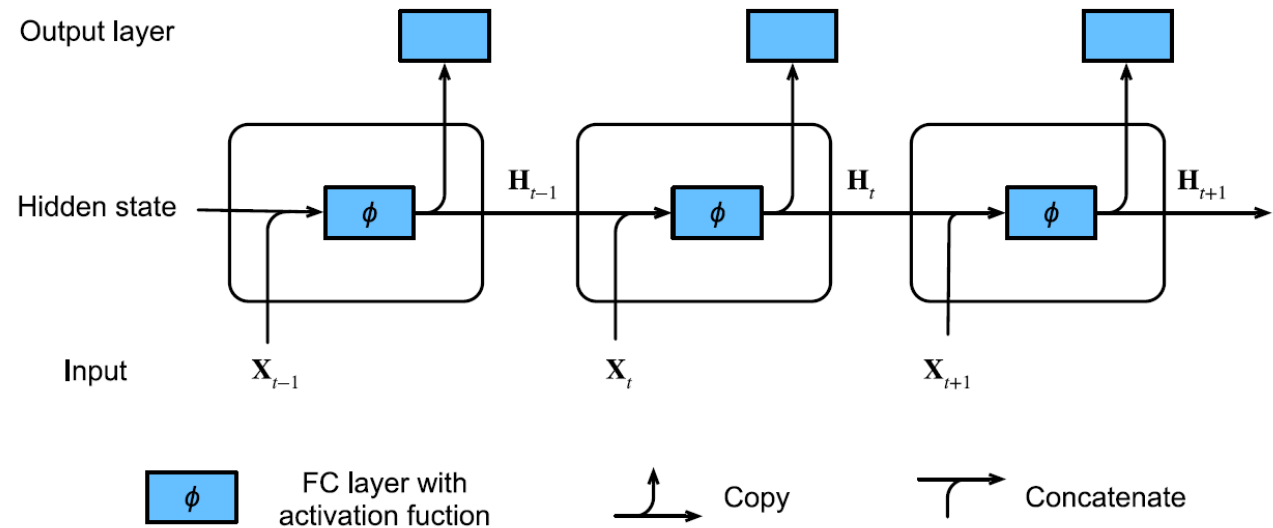
# Natural Language Processing (NLP)

- An area of Artificial Intelligence
- Processing or generation of text
- Multiple applications
- **Used to analyze covid-19 mental health impact**



# Recurrent Neural Networks (RNNs)

- Text is sequential data
- Words are dependent between them
- RNNs: recurrence computation



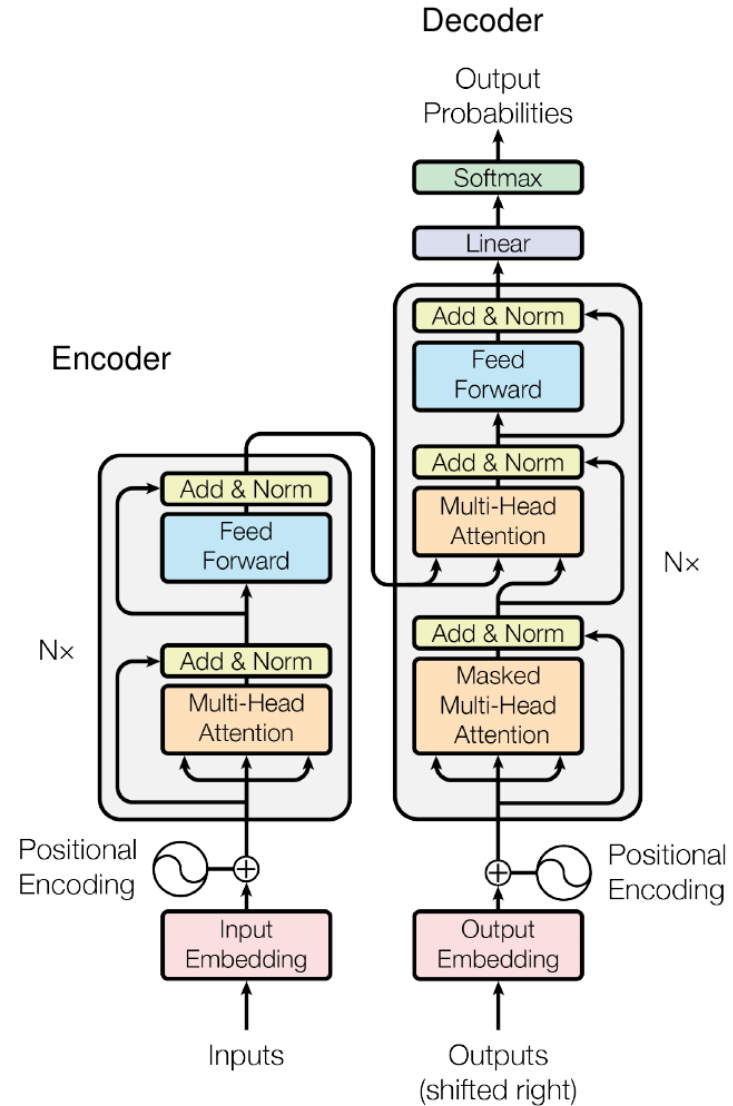
# RNNs Drawbacks

- Slow to train: sequential computation
- Difficult to capture long-term dependencies as the sentence increases: vanishing or exploding gradients

Karin has bought a **guitar**. When she came home, she left **it** in the kitchen and instead picked up some wine. When her brother entered the kitchen, he picked **it** up and started ...

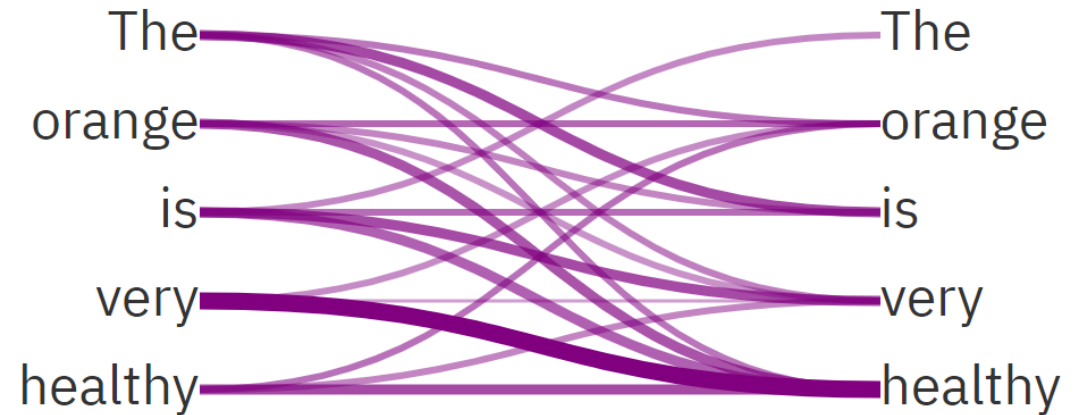
# The Transformer

- Released in 2017
- Parallelization of sequential data
- Avoid vanishing or exploding gradients
- Pure **self-attention** mechanisms



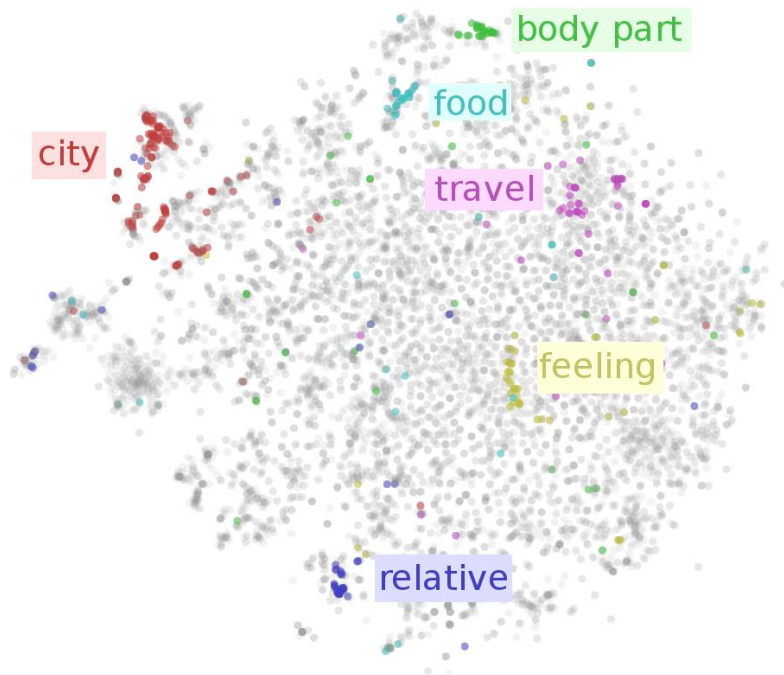
# Self-Attention

- Representation of a sentence
- Maps set to set
- Word relations: grammar, semantic...
- Critical in **BERT**



# Self-Attention Intuition (I)

Word → Token → Embedding vector



	living being	feline	human	gender	royalty	verb	plural
<i>man</i> →	0.6	-0.2	0.8	0.9	-0.1	-0.9	-0.7
<i>woman</i> →	0.7	0.3	0.9	-0.7	0.1	-0.5	-0.4
<i>king</i> →	0.5	-0.4	0.7	0.8	0.9	-0.7	-0.6
<i>queen</i> →	0.8	-0.1	0.8	-0.9	0.8	-0.5	-0.9

Word      Word embedding



# Self-Attention Intuition (II)

Sentence:            *The*        *orange*    *is*            *very*        *healthy*

Word vectors:         $x_1$              $x_2$              $x_3$              $x_4$              $x_5$

Query:                 $q_1 = W_Q x_1, \dots, q_6 = W_Q x_5$

Keys:                  $k_1 = W_K x_1, \dots, k_6 = W_K x_5$

Values:                $v_1 = W_V x_1, \dots, v_6 = W_V x_5$

Selected word:        *orange*  $\rightarrow q_2$

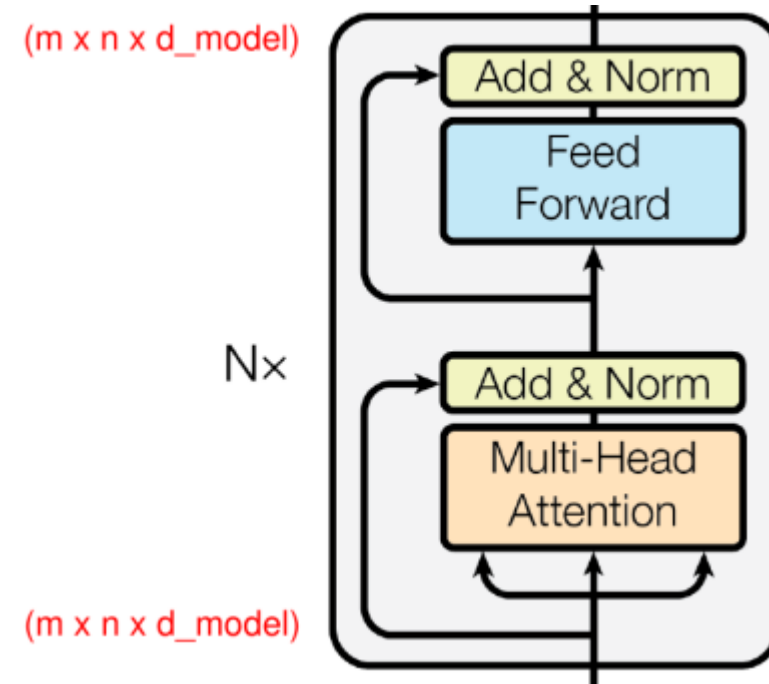
Weights:               $z_1 = k_1^T q_2, \dots, z_5 = k_5^T q_2$

$[w_1, \dots, w_5] = \text{softmax}(z_1, \dots, z_5),$         where  $\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$

Attention vector:     $y_2 = \sum_i w_i v_i = w_1 v_1 + \dots + w_5 v_5$

# Transformer Encoder

- Architecture used by BERT
- Multi-Head attention block
- Input and output with same dimensionality
  - $m$ : number of sentences
  - $n$ : number of words
  - $d_{model}$ : embedding dimension



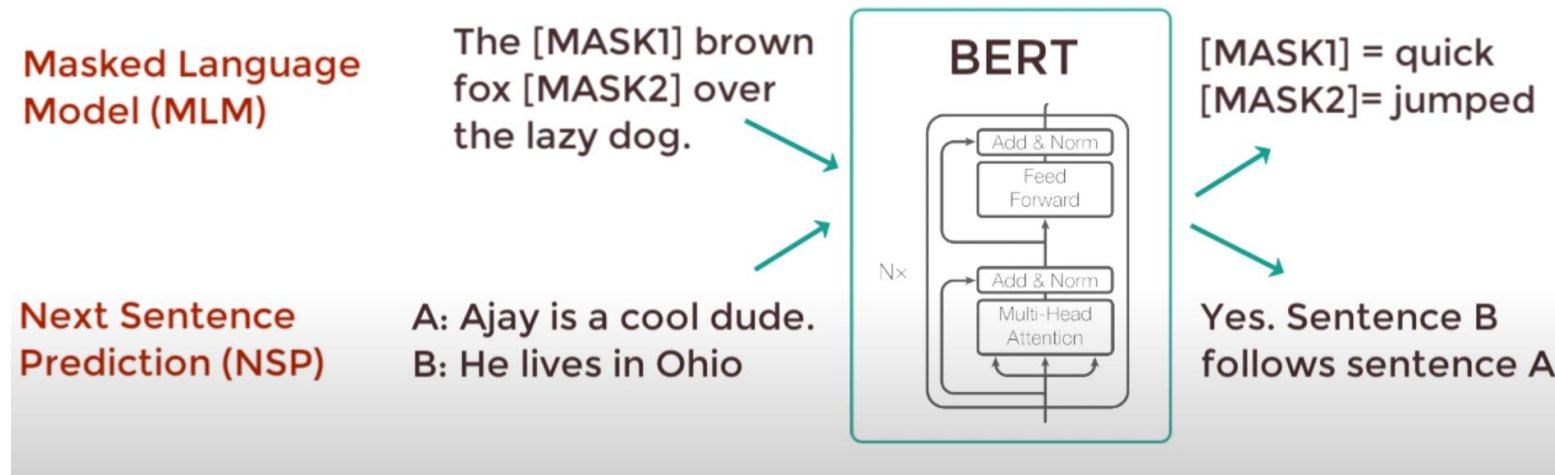
# BERT

- **B**idirectional **E**ncoder **R**epresentations from **T**ransformers, released in 2018 by Google
- Composed by a stack of Transformer encoders
- Uses self-attention to get left and right context



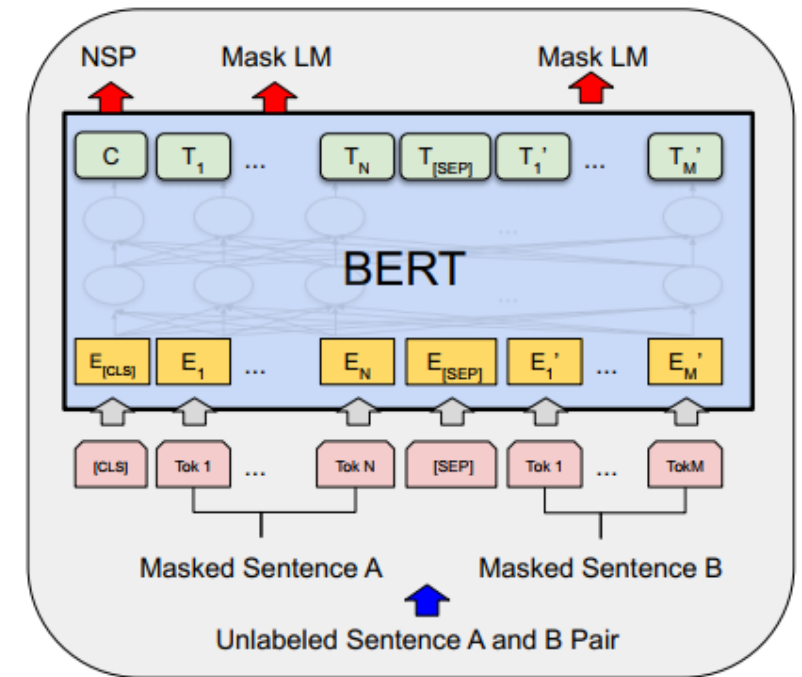
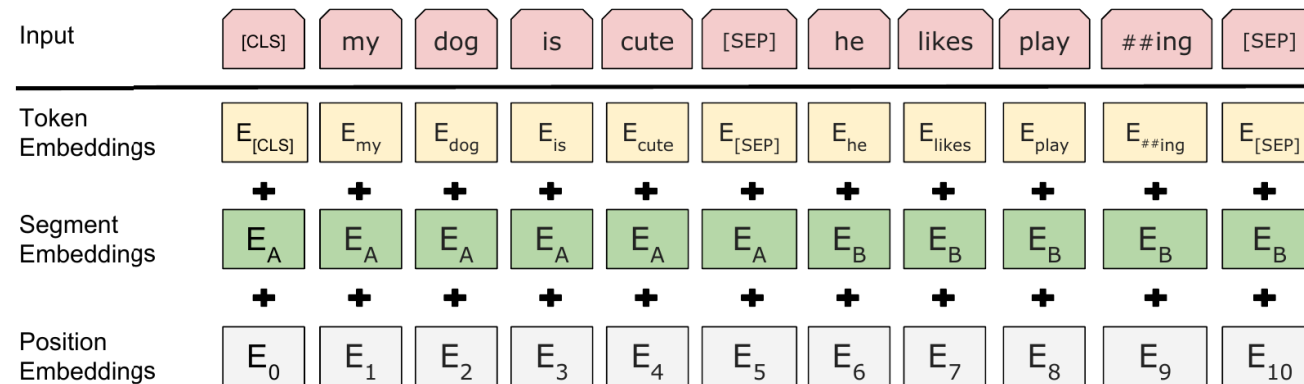
# Pre-Training BERT

- 16 TPUs for 4 days, data from Wikipedia and BookCorpus
- Masked language modelling
- Next Sentence Prediction



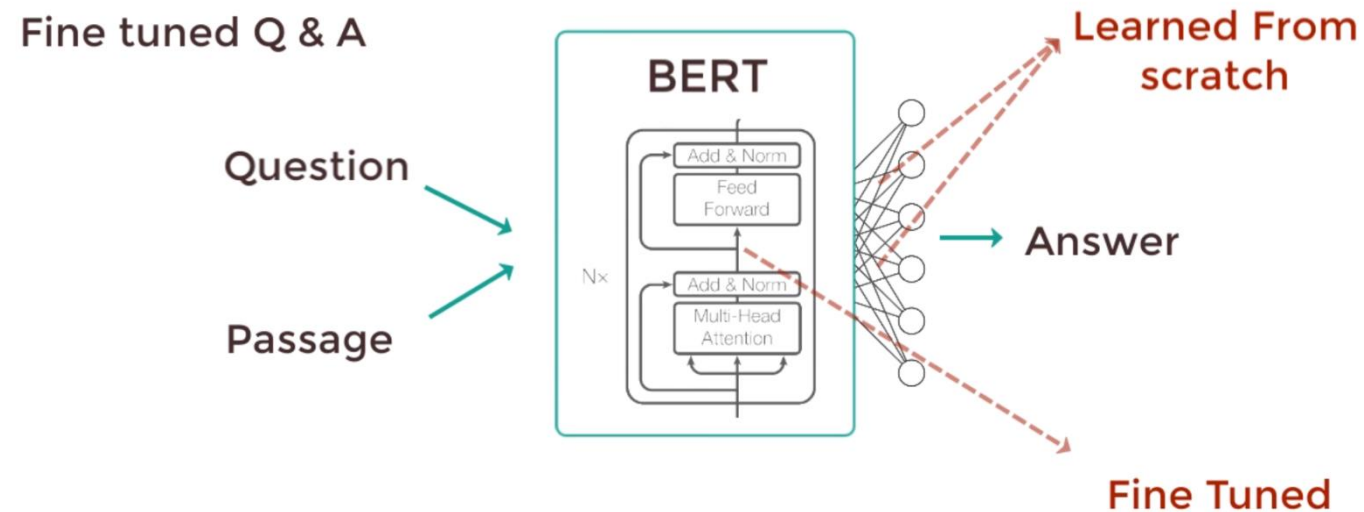
# Input-Output BERT

- Input: addition of 3 embeddings
- Output: classification token and attention vectors
- Same dimensionality




# Fine-Tune BERT

- Specialization in a task
- BERT + output layer and specific dataset
- Computationally inexpensive compared with pre-training



# Sentiment Analysis

- Predict positive or negative sentiment of a sentence
- Two labels: **0 – negative, 1 – positive**


 text-classification

This is an amazing model!

Compute

Computation time on cpu: 0.026 s

NEGATIVE	0.000
POSITIVE	1.000

[▶ JSON Output](#) [↔ API endpoint](#) [ⓘ](#) [Share](#) 

# SST-2 Dataset

- Stanford Sentiment Treebank (STT-2) dataset
- Dataset distribution:

Training	Validation	Test
45123	871	22225

- Positive and negative samples in training dataset

Total samples	Positive	Negative
45123	25114	20009



# SST-2 Preprocessing

- Tokenization: WordPiece
- Pad to maximum length
- Truncate to 512 if necessary
- Add special tokens

Original sentence:

one but two flagrantly fake thunderstorms

Token IDs:

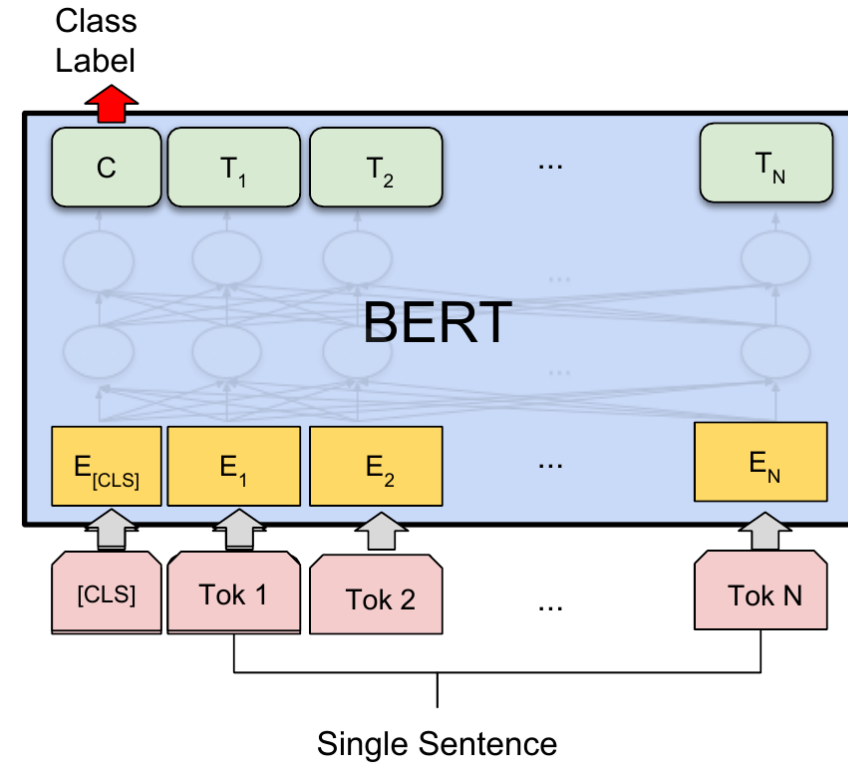
```
[ 101  2028  2021  2048  5210 17884  2135  8275  8505 19718  2015  102
   0      0      0      0      0      0      0      0      0      0      0      0
   0      0      0      0      0      0      0      0      0      0      0      0
   0      0      0      0      0      0      0      0      0      0      0      0
   0      0      0      0      0      0      0      0      0      0      0      0
   0      0      0      0      0      0      0      0      0      0      0      0]
```

After tokenization:

```
['[CLS]' 'one' 'but' 'two' 'flag' '##rant' '##ly' 'fake' 'thunder'
 '##storm' '##s' '[SEP]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]'
 '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]'
 '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]'
 '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]'
 '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]' '[PAD]'
 '[PAD]' '[PAD]' '[PAD]']
```

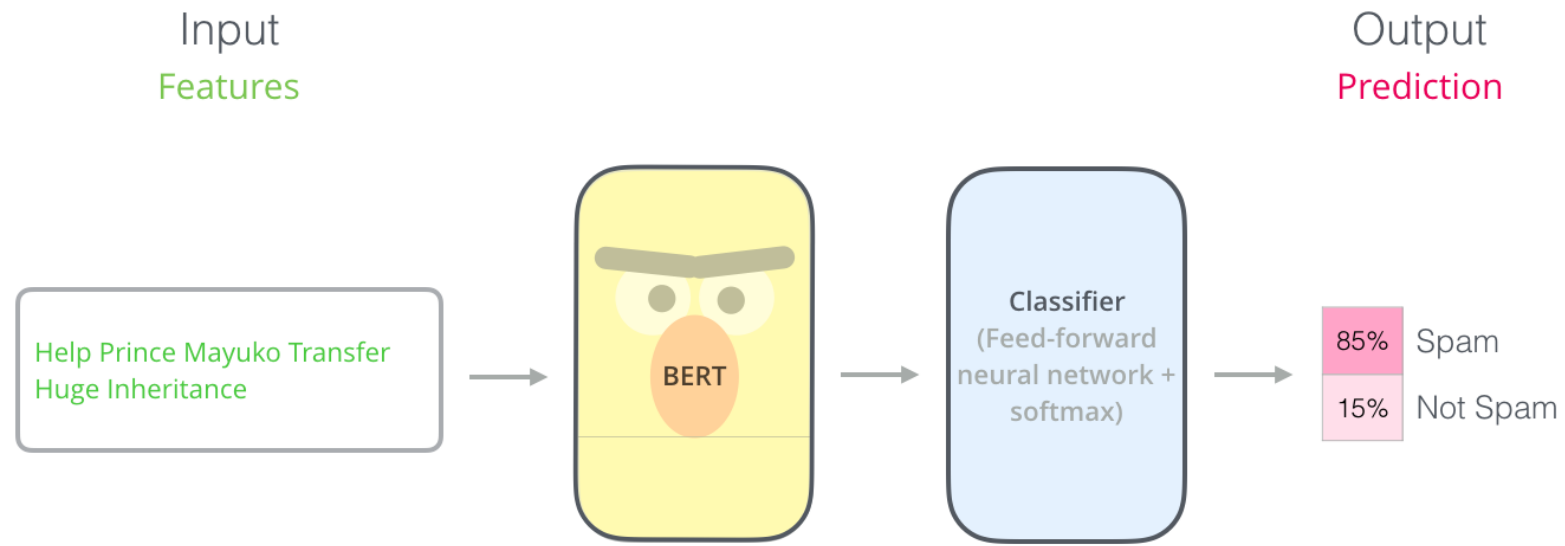
# Generation of the Model (I)

- Input: single sentence
- Output: classification token C



# Generation of the Model (II)

- BERT + classifier (feed forward network with one single hidden layer)
- New parameters to train: classification layer weights  $W$
- Softmax to calculate probabilities



# Metrics

- Accuracy =  $\frac{TP+TN}{TP+TN+FP+FN}$

True Positive Rate =  $\frac{TP}{TP+FN}$

False Positive Rate =  $\frac{FP}{FP+TN}$

		Predicted	
		Positive	Negative
Real	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

- Cross-Entropy Loss: difference between prediction and real value

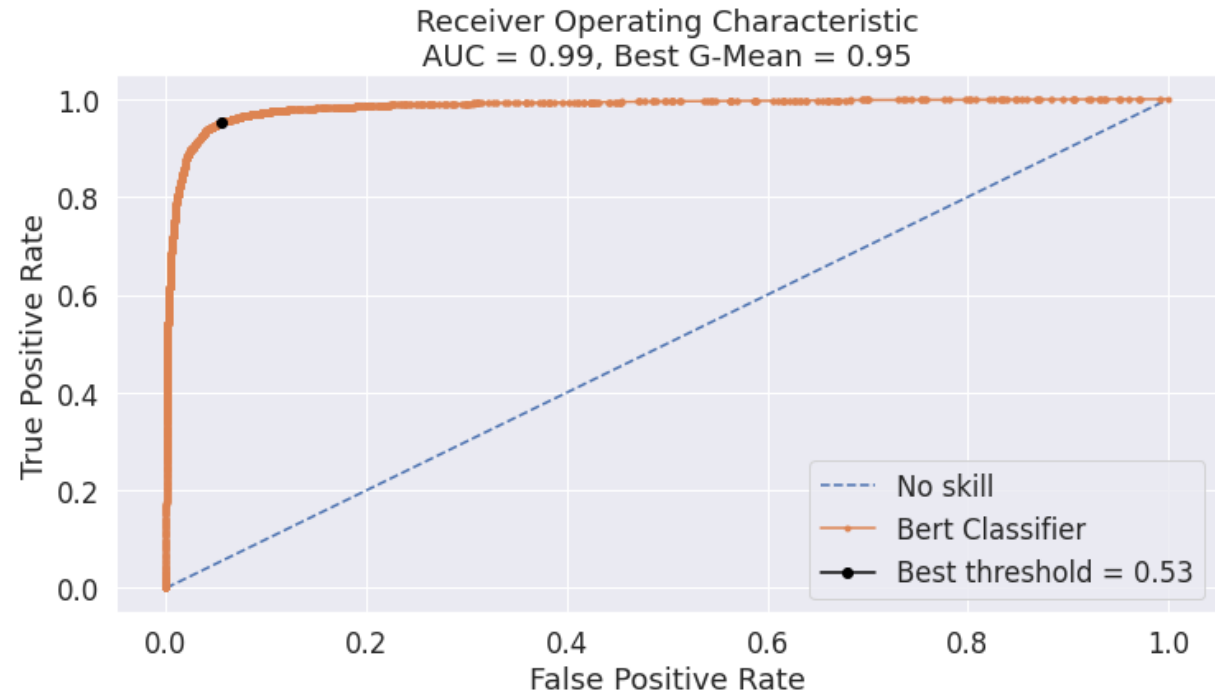
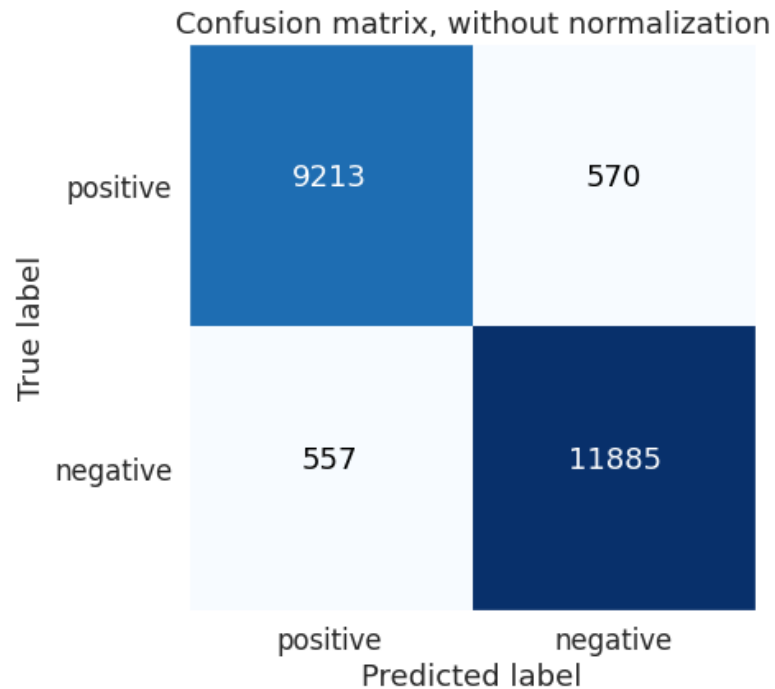
# Fine-Tune Results on Sentiment Analysis

- 1 epoch = 1 iteration whole dataset
- Accuracy: **93%**
- Training time: **27 min 9 sec**

	Training Loss	Validation Loss	Accuracy	Training Time	Validation Time
epoch					
1	0.224070	0.199047	0.916773	0:05:10	0:00:02
2	0.095247	0.199060	0.933036	0:05:09	0:00:02
3	0.043675	0.222848	0.935268	0:05:09	0:00:02

# Test Results on Sentiment Analysis

- Accuracy: **95%**



# Final Application for Sentiment Analysis

- Example of a positive prediction

**sentence:** "Honestly, with all these incredible models, I think NLP is going to be a popular trend next years."

Sentiment	Probability
Positive	0.99952
Negative	0.00048

- Example of a negative prediction

**sentence:** "Next application was completely frustrating. It is difficult to implement."

Sentiment	Probability
Positive	0.00177
Negative	0.99823

# Question Answering

- Answer a question about the context
- The answer is taken from the context

question-answering

Where do I live?

Context

My name is Sylvain and I live in Brooklyn

Compute

Computation time on cpu: 0.102 s

Brooklyn0.989

► JSON Output ◀ API endpoint ⓘ

Share [ ]



# SQuAD Dataset

- Stanford Question Answering Dataset (SQuAD)
- Dataset distribution:

Training	Validation	Test
58691	10570	28908

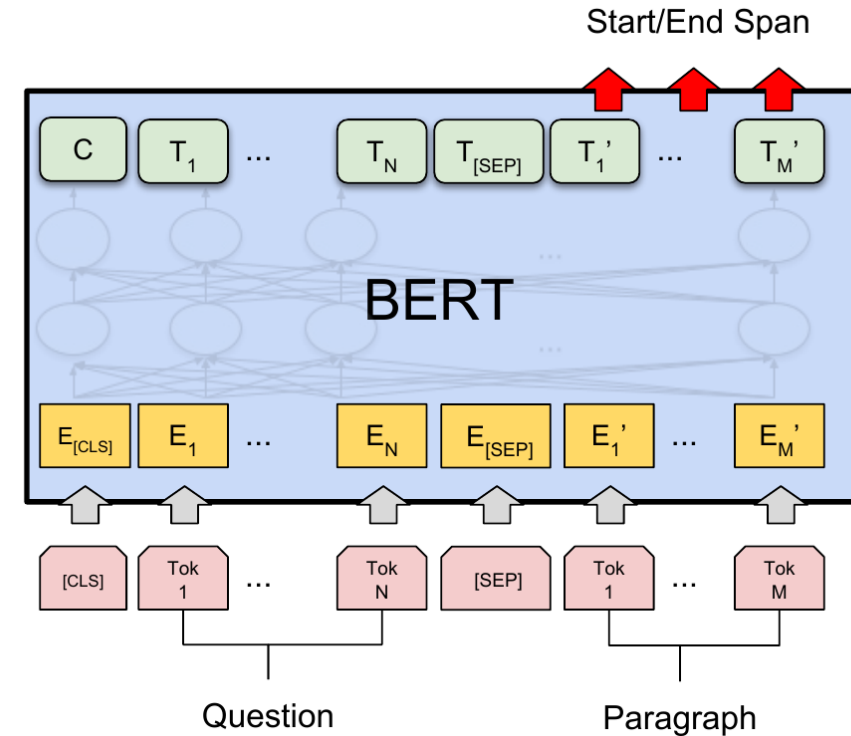
# SQuAD Preprocessing

- As sentiment analysis: tokenize and pad
- Truncate: divide long contexts into features
- Map answers

end_positions	input_ids	start_positions
92	[101, 2096, 2002, 2001, 1999, 3905, 2082, 1010, 2054, 4066, 1997, 16841, 2106, 28924, 2490, 1029, 102, 28924, 4114, 13616, 1998, 5500, 14921, 21289, 1996, 12078, 1012, 1999, 2255, 3777, 1010, 2002, 2419, 1037, 10467, 21248, 7795, 1005, 1055, 22965, 2013, 1996, 2142, 5424, 3072, 1012, 2076, 2023, 2027, 3631, 3645, 1997, 1037, 2334, 3309, 5496, 1997, 3529, 6544, 1012, 9105, 1996, 4614, 1005, 3086, 1010, 2027, 10016, 2010, 2155, 2013, 11200, 1012, 28924, 2333, 2000, 28616, 14660, 1010, 2045, 7052, 28616, 14660, 3905, 2082, 1012, 8498, 2010, 3037, 1999, 5424, 8986, 16841, 1010, 2002, 4188, 2000, 3693, 2151, 1997, ...]	90

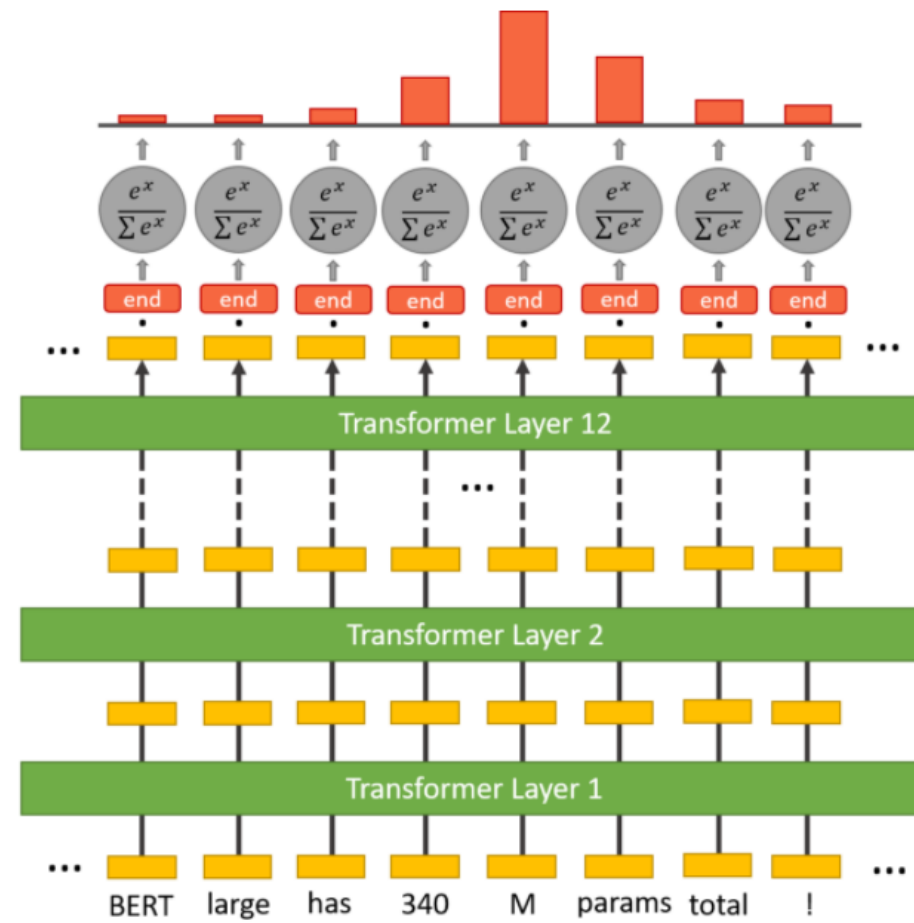
# Generation of the Model (I)

- Input: pair of sentences, question and context
- Output: start/end span answer



# Generation of the Model (II)

- New parameters: start ( $S$ ) and end ( $E$ ) vector
- Probability =  $\frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$ , for  $S$  and  $E$
- Score =  $S \cdot T_i + E \cdot T_j$ ,  $i$  start and  $j$  end



# Fine-Tune Results on Question Answering

- Loss evaluation. Difficult to quantify
- Training time: **1 h 23 min 21 sec**

	Training Loss	Validation Loss	Training Time	Validation Time
epoch				
1	1.3294	1.086206	0:27:47	0:01:16
2	0.7180	1.077027	0:27:47	0:01:16
3	0.4083	1.287688	0:27:47	0:01:16

# Test Results on Question Answering

	context	question	answers	predicted answers
0	The Human Development Index (HDI) is a composite statistic of life expectancy, education, and income per capita indicators, which are used to rank countries into four tiers of human development. A country scores higher HDI when the life expectancy at birth is longer, the education period is longer, and the income per capita is higher. The HDI was developed by the Pakistani economist Mahbub ul Haq, often framed in terms of whether people are able to "be" and "do" desirable things in their life, and was published by the United Nations Development Programme.	Who developed the HDI?	Mahbub ul Haq	Mahbub ul Haq
1	Great Britain lost Minorca in the Mediterranean to the French in 1756 but captured the French colonies in Senegal in 1758. The British Royal Navy took the French sugar colonies of Guadeloupe in 1759 and Martinique in 1762 as well as the Spanish cities of Havana in Cuba, and Manila in the Philippines, both prominent Spanish colonial cities. However, expansion into the hinterlands of both cities met with stiff resistance. In the Philippines, the British were confined to Manila until their agreed upon withdrawal at the war's end.	What island did Great Britain lose in 1756?	Great Britain lost Minorca in the Mediterranean to the French in 1756	Minorca

# Final Application for Question Answering

- Same context but different questions

**context:** " In 2016, I started studying telecommunications, it was hard at the beginning, but one year after, 2017, I was sure this was a good option!"

---

**question:** " When did I start telecommunications?"

---

The predicted answer is

2016

**context:** " In 2016, I started studying telecommunications, it was hard at the beginning, but one year after, 2017, I was sure this was a good option!"

---

**question:** " When was I sure about studying telecommunications?"

---

The predicted answer is

2017

# Conclusion

- Transformer and self attention defined current state-of-the-art architectures
- BERT boost pre-trained language models
  - State of the art models in multiple NLP tasks
  - Fine-tune in short times



Thank you for your attention