

Deep Learning in Applications

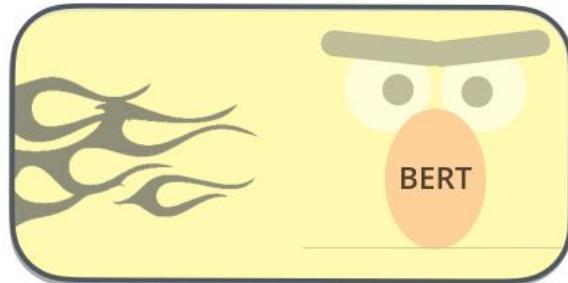
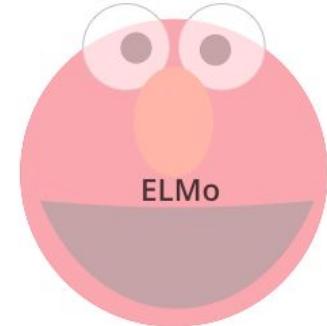
How NLP Cracked Transfer Learning

Anastasia Ianina

15.06.2020

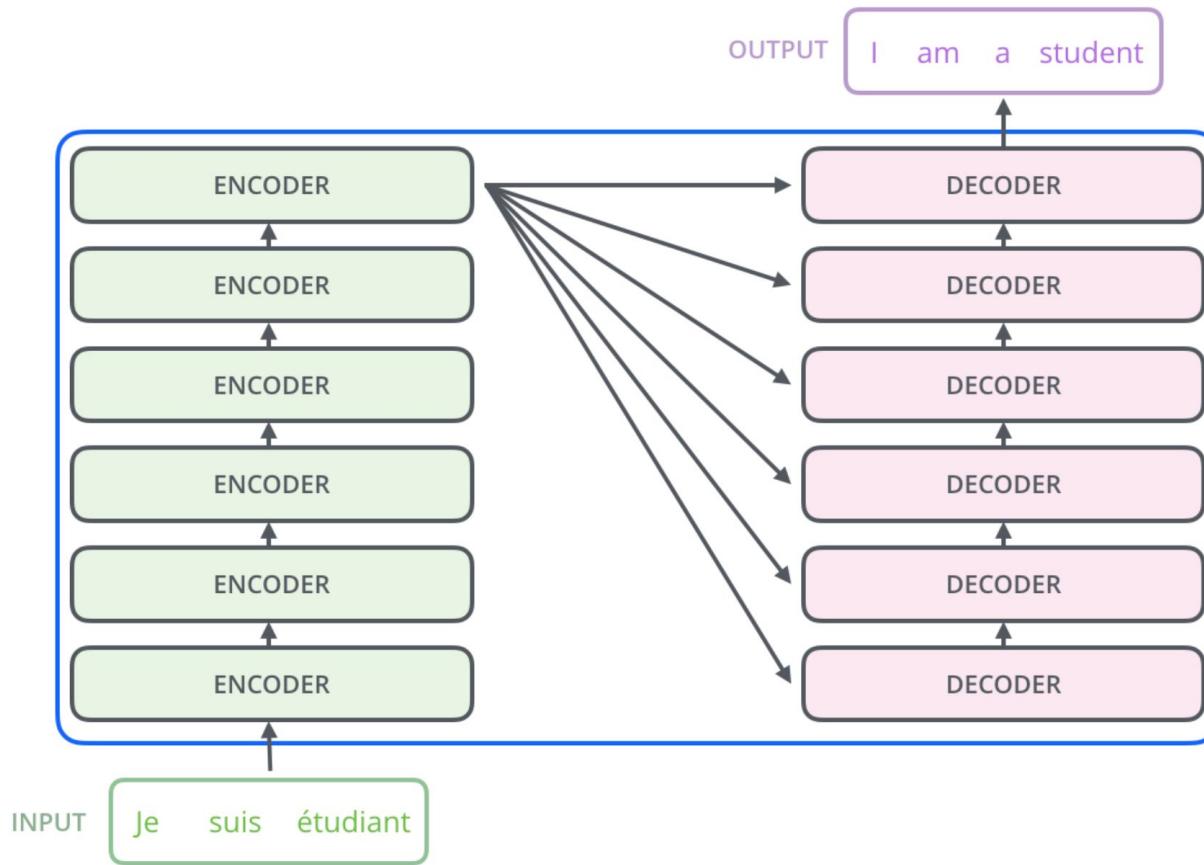
1. Transformer: recap
2. OpenAI Transformer
3. ELMO
4. BERT
5. BERTology
6. GPT, GPT-2, GPT-3
7. Q & A

Based on: <http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture13-contextual-representations.pdf>
<https://jalammar.github.io/illustrated-transformer/>
<http://jalammar.github.io/illustrated-bert/>

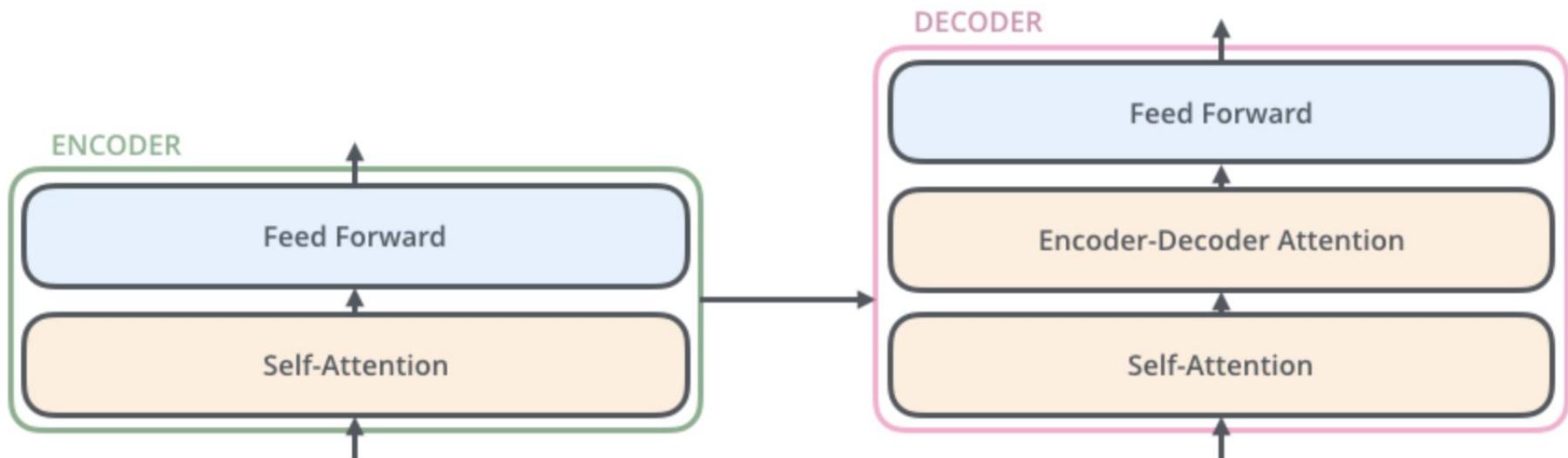


The Transformer: recap

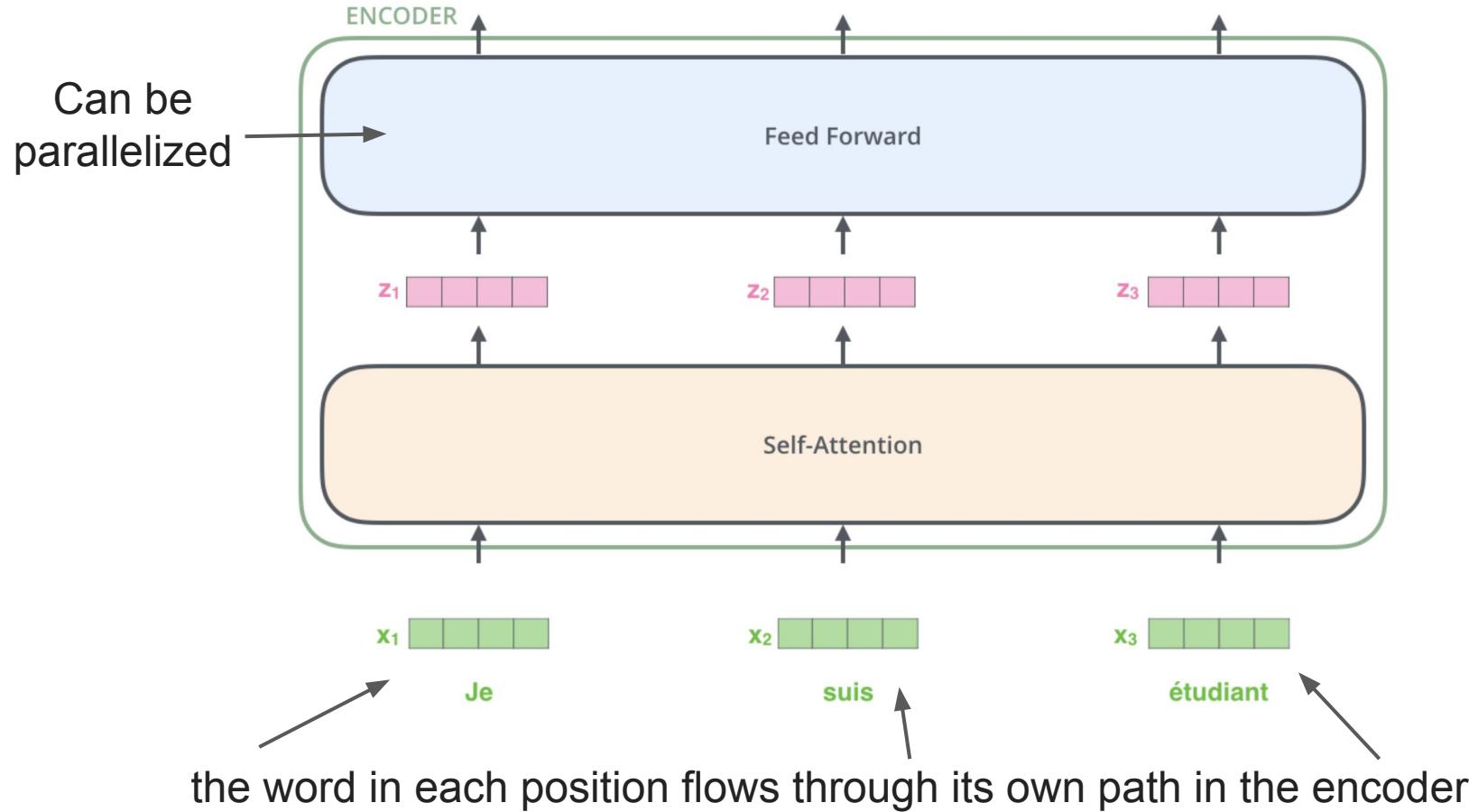
The Transformer



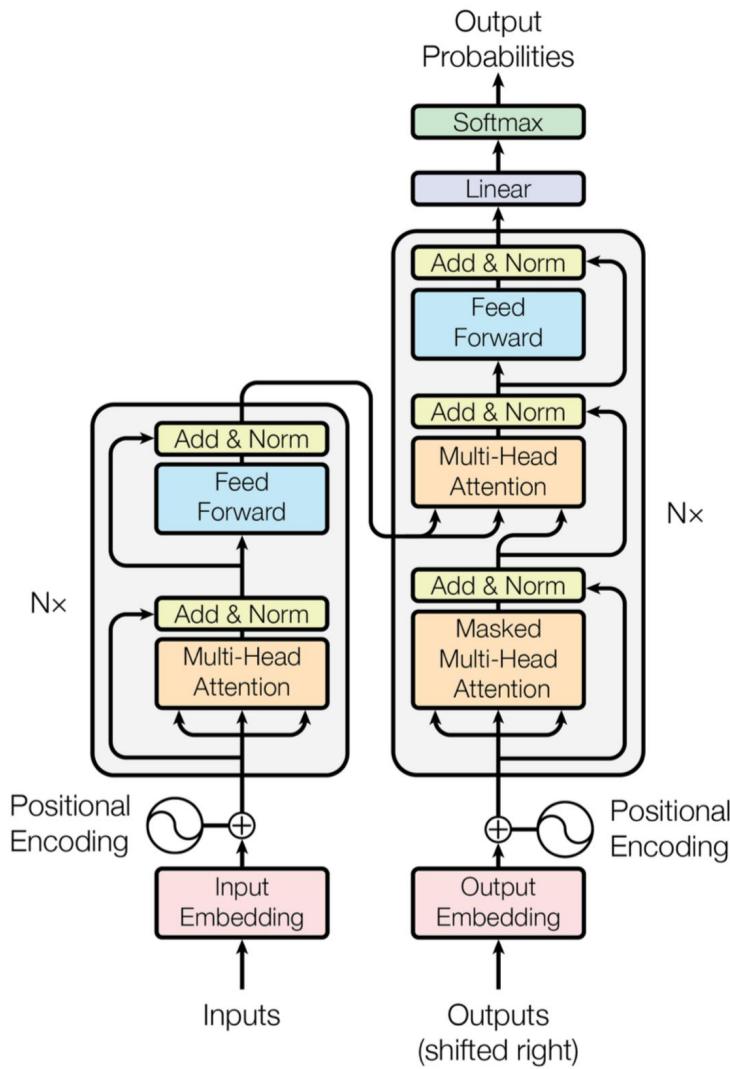
The Transformer



The Transformer



The Transformer: recap



- Proposed in the paper
“Attention is All You Need”
(Ashish Vaswani et al.)
- No recurrent or convolutional neural networks -> just attention
- Uses Multi-Head **self-attention** concept

Self-Attention: recap

Self-Attention in Detail

Input

Embedding

Queries

Keys

Values

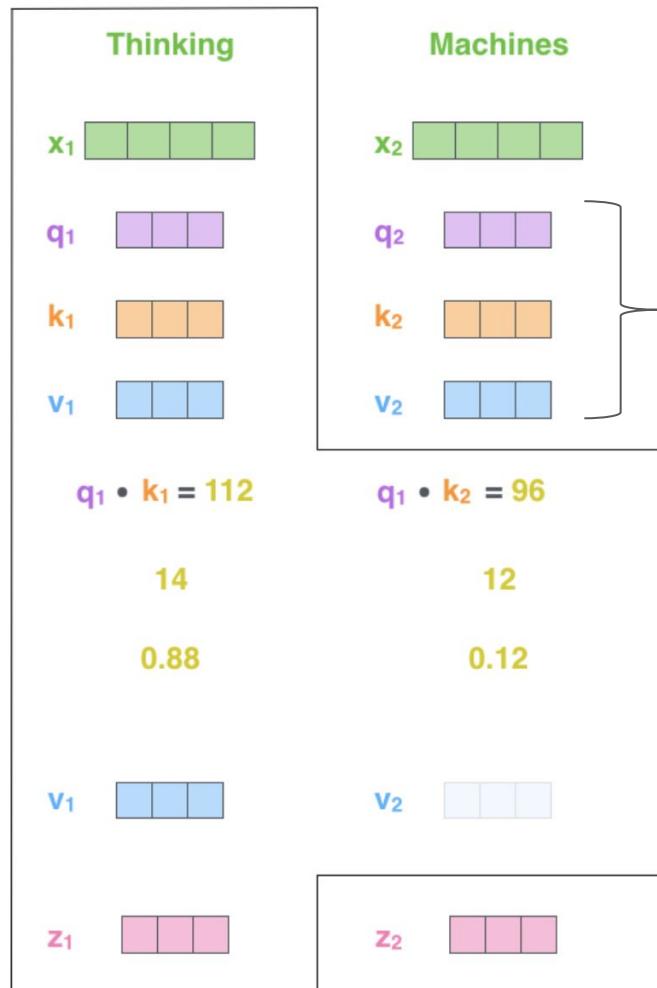
Score

Divide by 8 ($\sqrt{d_k}$)

Softmax

Softmax
X
Value

Sum



STEP 1: create Query, Key, Value

STEP 2: calculate scores

STEP 3: divide by $\sqrt{d_k}$

STEP 4: softmax

STEP 5: multiply each value vector by the softmax score

STEP 6: sum up the weighted value vectors

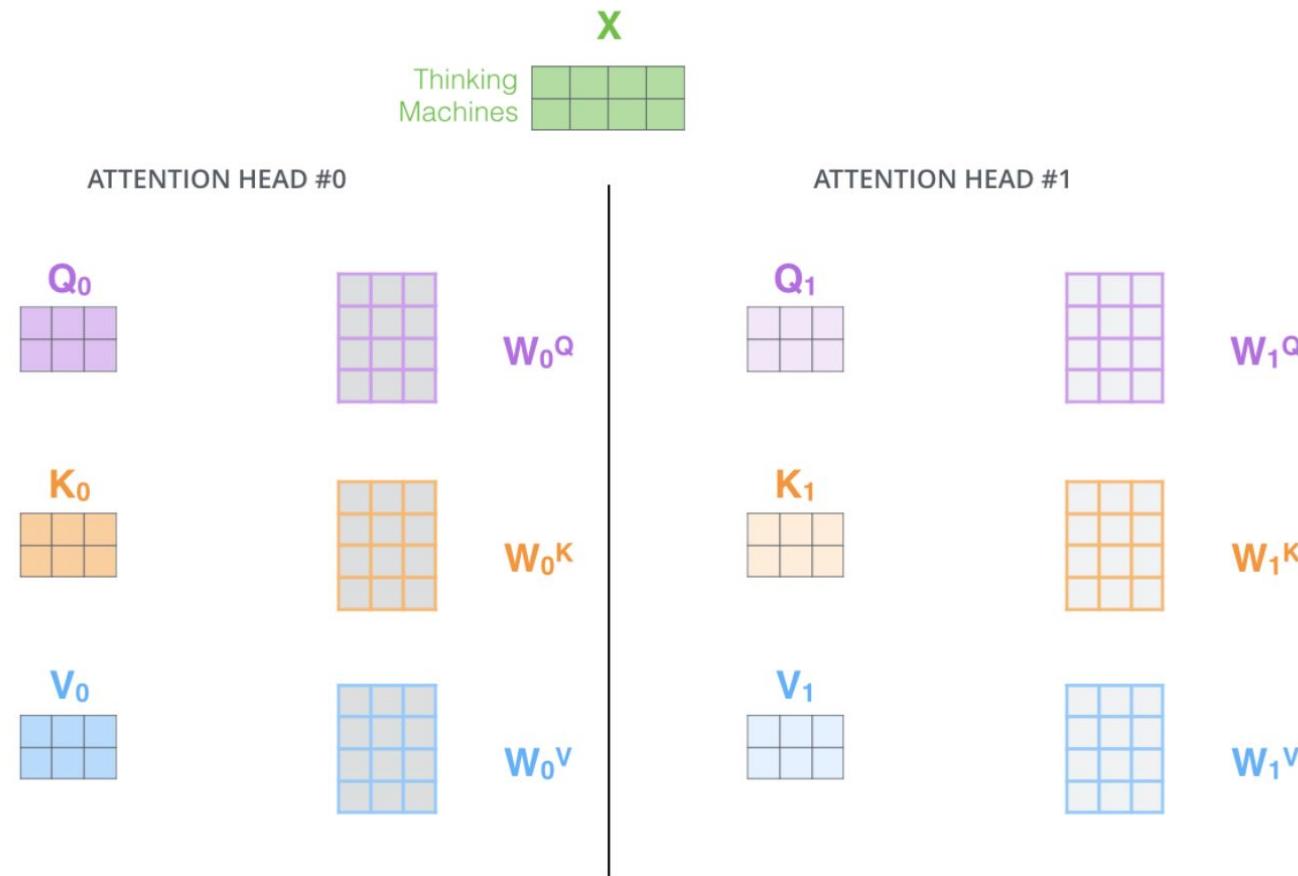
Self-Attention: Matrix Calculation

$$\text{softmax} \left(\frac{\begin{array}{c} \text{Q} \quad \text{K}^T \\ \times \\ \hline \end{array}}{\sqrt{d_k}} \right) \text{V}$$

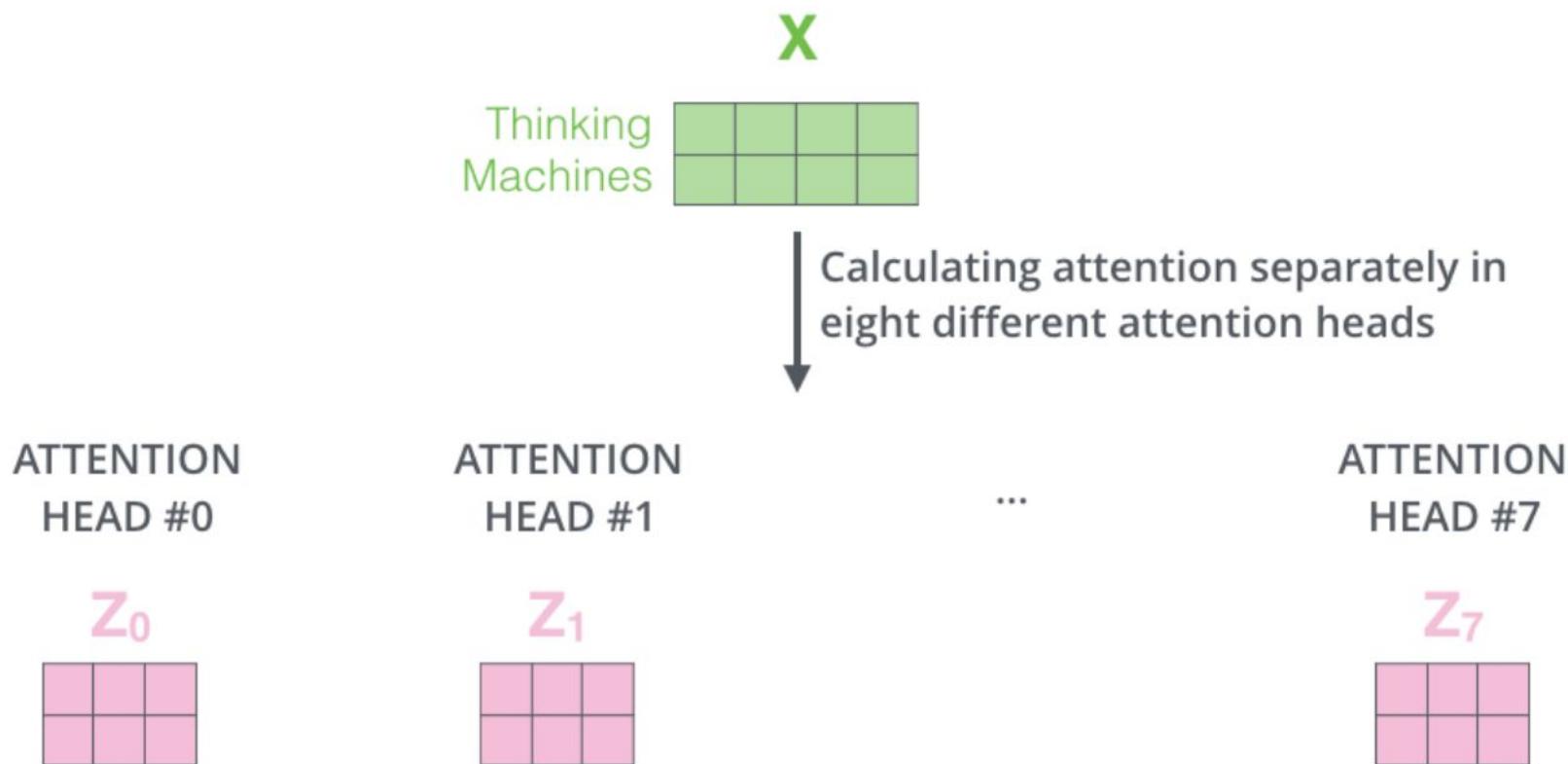
Diagram illustrating the matrix calculation for Self-Attention:

- The input matrix Q is a 3x3 purple square.
- The input matrix K^T is a 3x3 orange square.
- The output matrix V is a 3x3 blue square.
- The result is the softmax of the product of Q and K^T , scaled by $\sqrt{d_k}$.
- The final result is labeled Z and is represented by a 3x3 pink square.

Multi-Head Attention



Multi-Head Attention



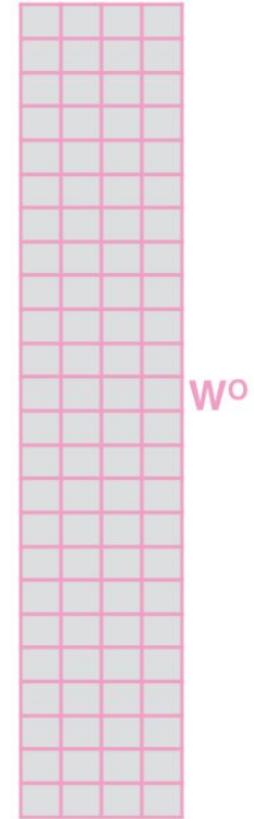
Multi-Head Attention

1) Concatenate all the attention heads



2) Multiply with a weight matrix W^o that was trained jointly with the model

X



3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

$$= \begin{matrix} Z \\ \begin{array}{|c|c|c|c|} \hline & & & \\ \hline \end{array} \end{matrix}$$

1) This is our input sentence*

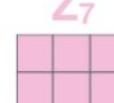
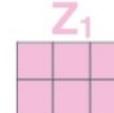
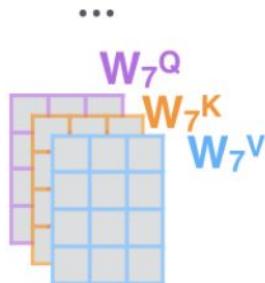
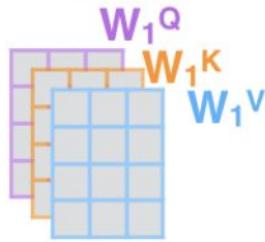
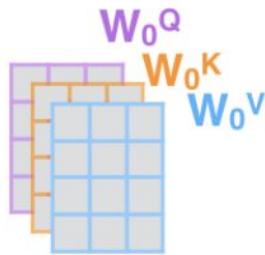
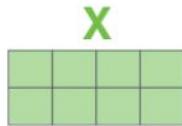
2) We embed each word*

3) Split into 8 heads.
We multiply X or R with weight matrices

4) Calculate attention using the resulting $Q/K/V$ matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^o to produce the output of the layer

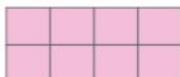
Thinking
Machines



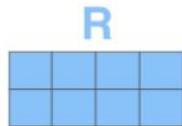
W^o



Z

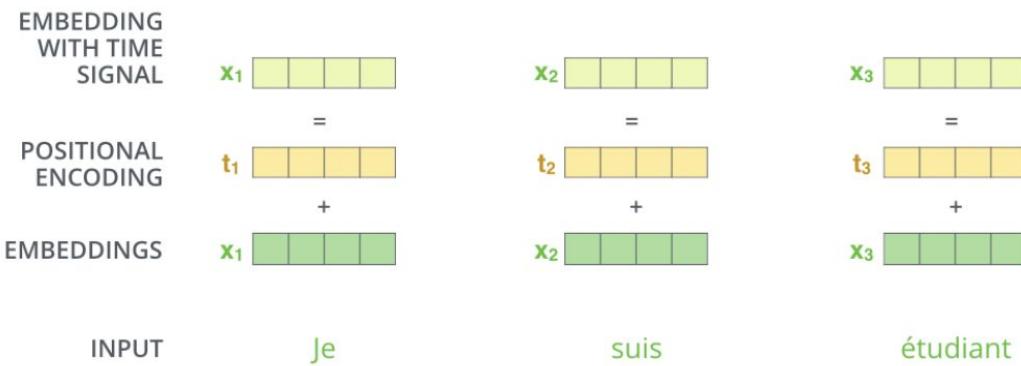
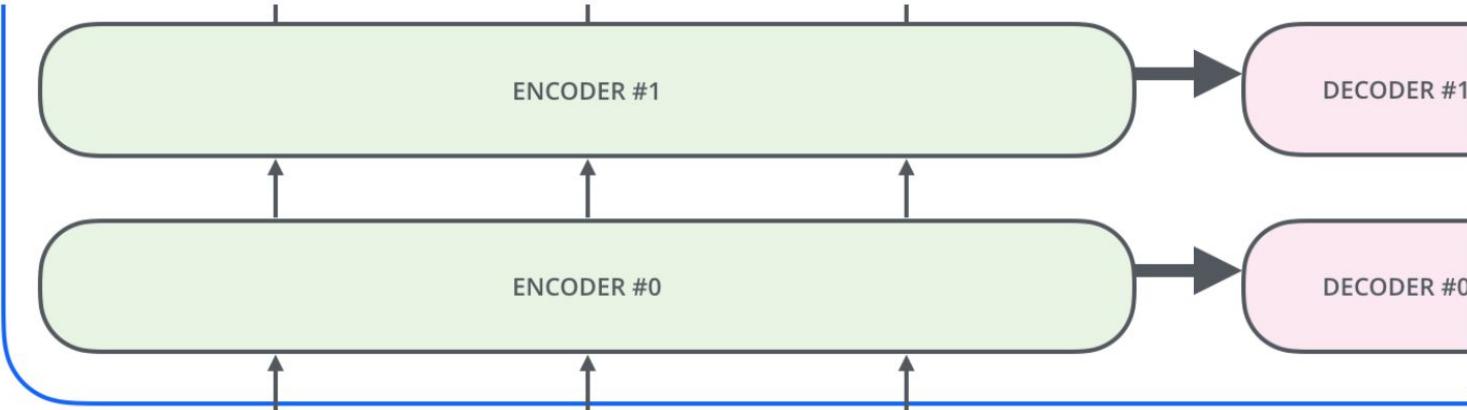


* In all encoders other than #0, we don't need embedding.
We start directly with the output of the encoder right below this one



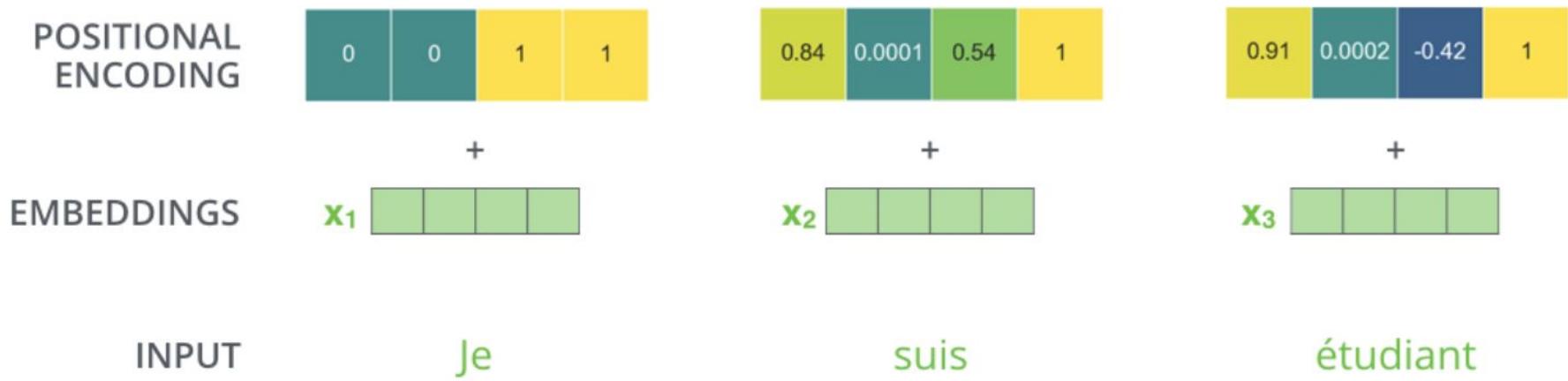
Positional Encoding

Positional Encoding



It provides meaningful distances between the embedding vectors once they're projected into Q/K/V vectors and during dot-product attention

Positional Encoding



OpenAI Transformer: Pre-training Decoder for Language Modeling

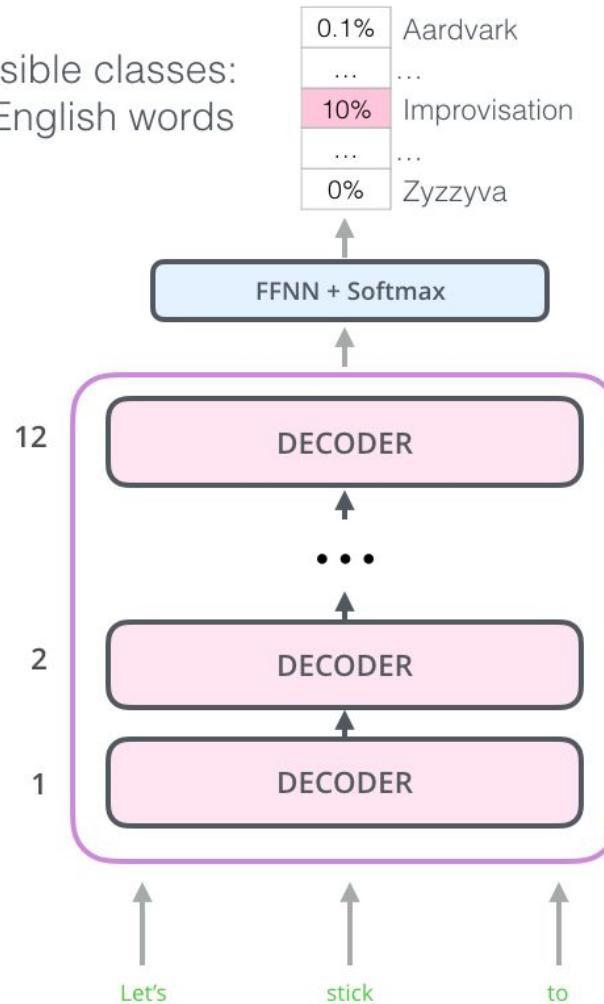
OpenAI Transformer

- The Encoder-Decoder structure of the transformer made it perfect for machine translation
- But what about sentence classification?
- **Main goal: pre-train a language model that can be fine-tuned for other tasks**



OpenAI Transformer

Possible classes:
All English words

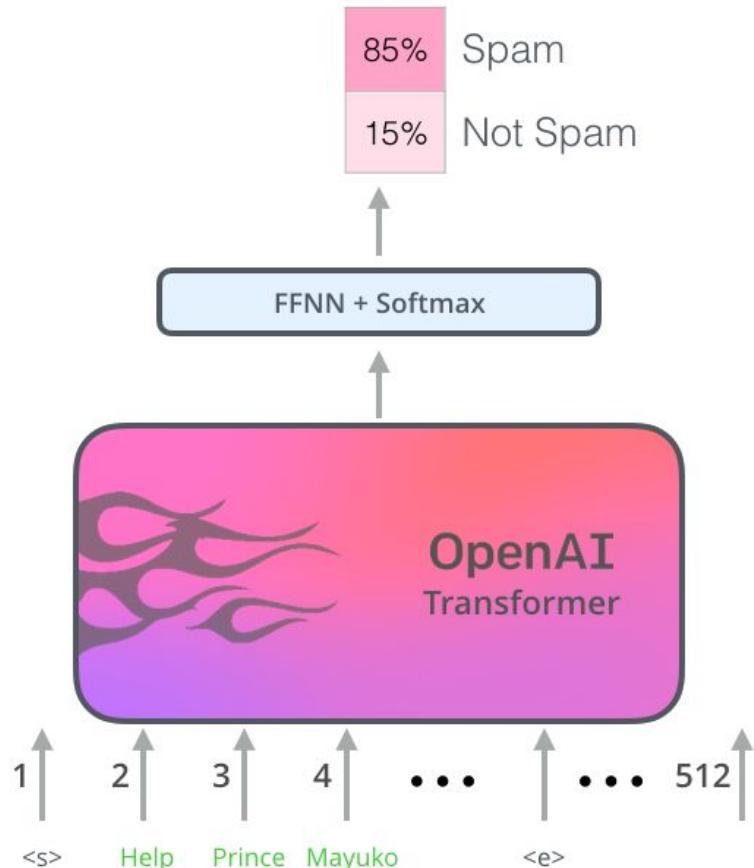


Differences from vanilla Transformer:

- no encoder
- decoder layers would not have the encoder-decoder attention sublayer
- Pre-train the model on predicting the next word using massive (unlabeled) datasets

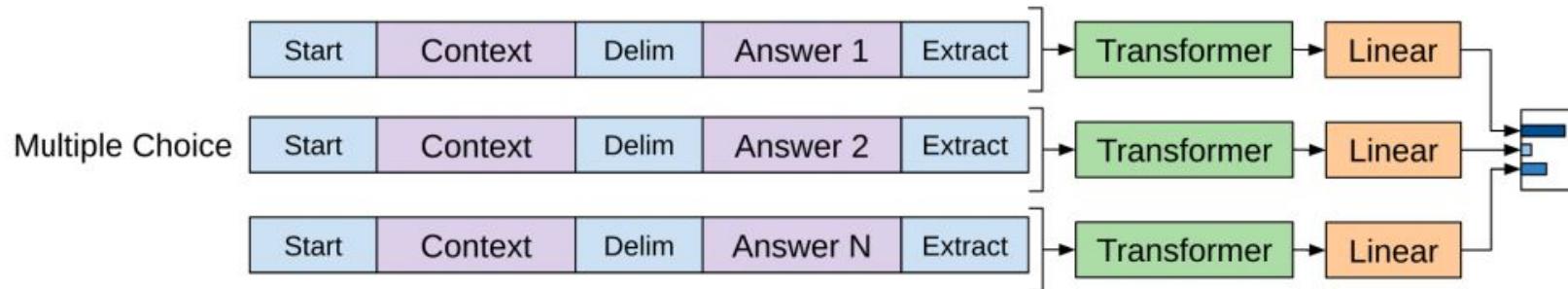
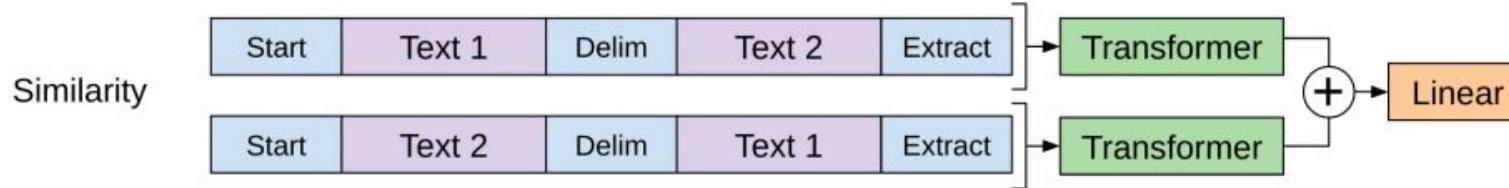
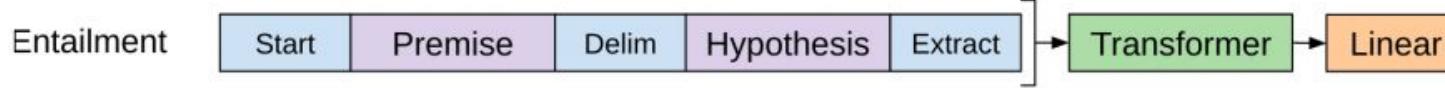
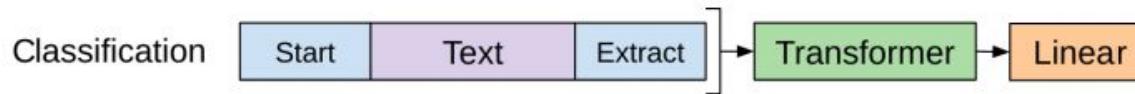
Pre-training phase

OpenAI Transformer



- During pre-training phase layers have been tuned to reasonably handle language
- Now let's use it for downstream tasks (e.g. sentence classification)

Input transformations for different tasks





ELMo: context that matters

ELMo: contextualized word embeddings

“Why not give it an embedding based on the context it’s used in – to both capture the word meaning in that context as well as other contextual information?”

[Peters et. al., 2017](#), [McCann et. al., 2017](#),
and yet again [Peters et. al., 2018](#) in the ELMo paper



ELMo - deep contextualized
word representations

What does it stand for?



1. Expedited Labour Market Opinion
2. Electric Light Machine Organization
3. Enough Let's Move On

What does it stand for?



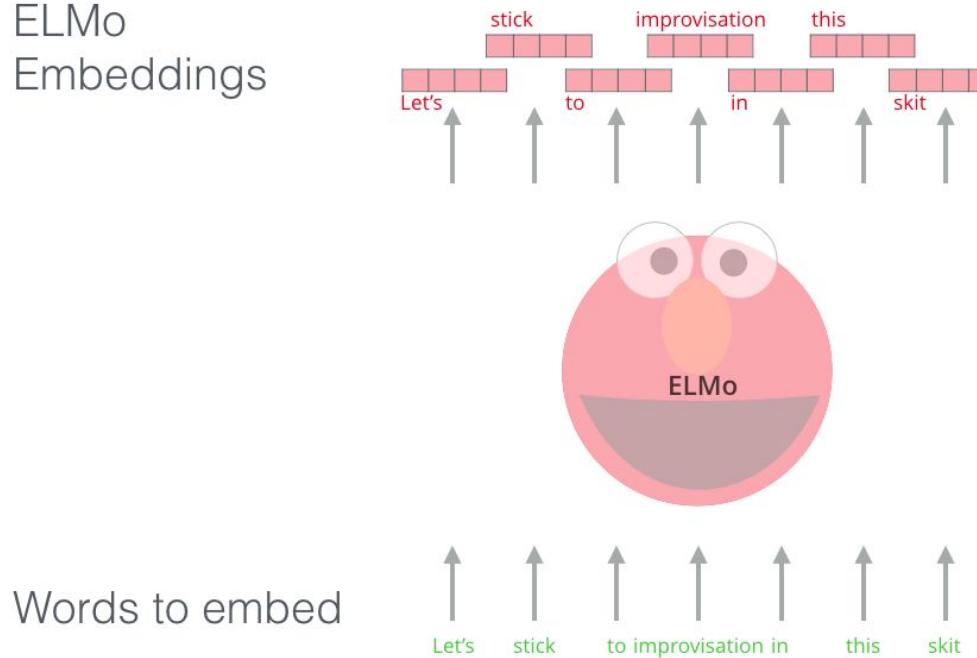
1. Expedited Labour Market Opinion
2. Electric Light Machine Organization
3. Enough Let's Move On
4. Embeddings from Language Models

ELMo: contextualized word embeddings



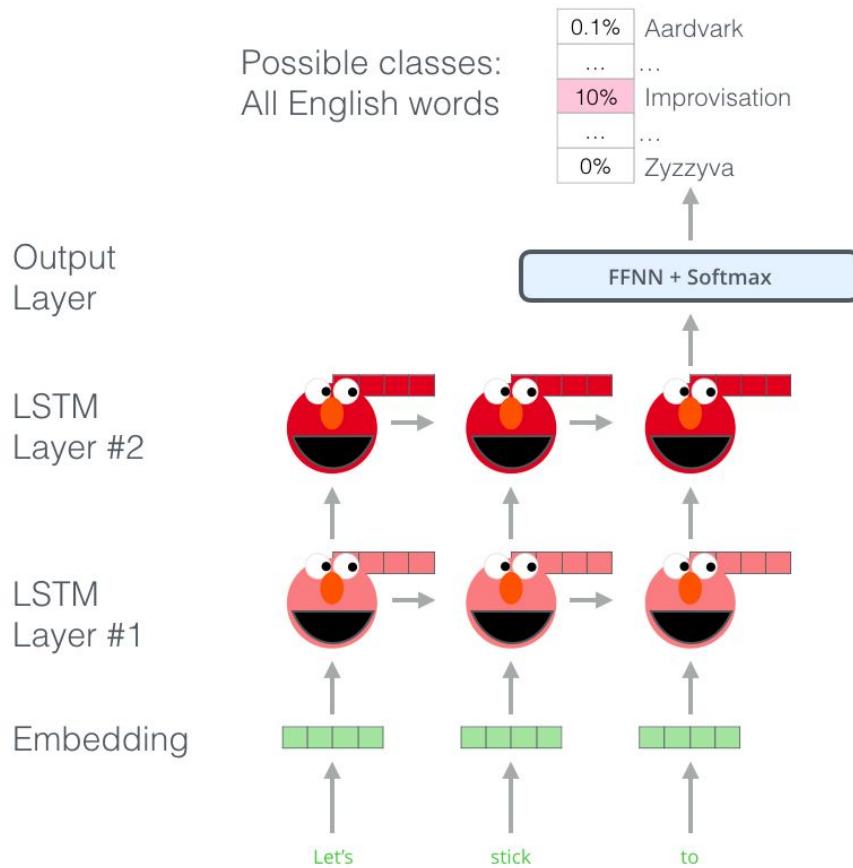
ELMo: Contextualized word embeddings

ELMo
Embeddings



- uses a bi-directional LSTM trained on Language Modeling task
- a model can learn without labels

Bidirectional Language Models (biLMs)



biLMs consist of forward and backward LMs:

- forward:

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_1, t_2, \dots, t_{k-1})$$

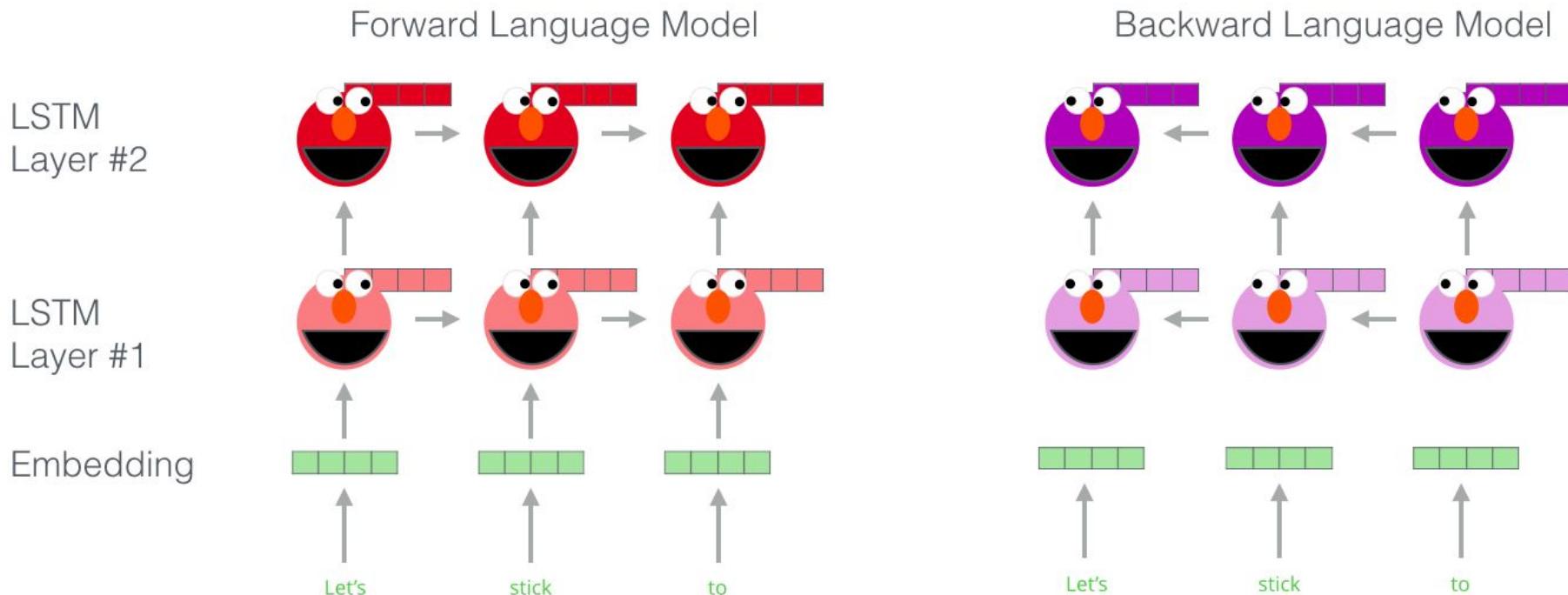
- Backward:

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_{k+1}, t_{k+2}, \dots, t_N)$$

LSTM predicts next word in both directions to build biLMs

ELMo: main pipeline

Embedding of “stick” in “Let’s stick to” - Step #1

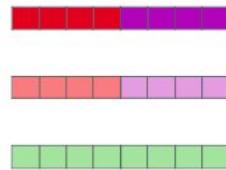


ELMo: main pipeline

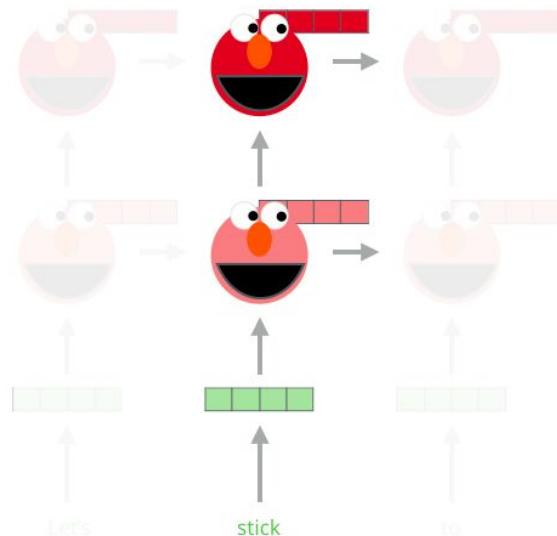
ELMo represents a word as a linear combination of corresponding hidden layers:

Embedding of “stick” in “Let’s stick to” - Step #2

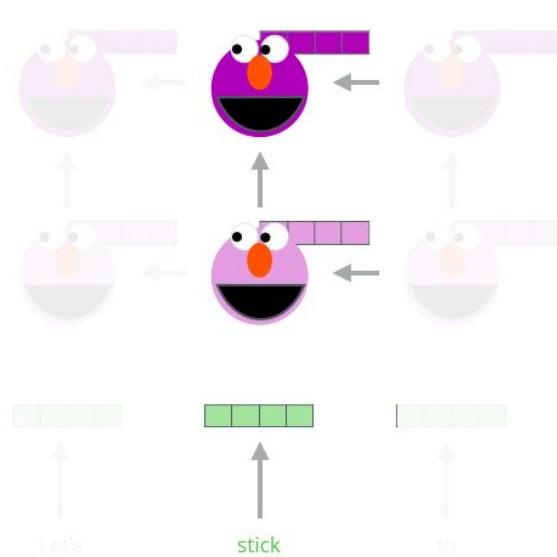
1- Concatenate hidden layers



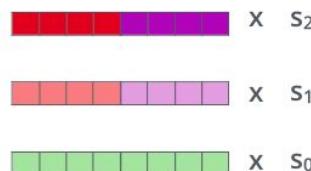
Forward Language Model



Backward Language Model



2- Multiply each vector by a weight based on the task



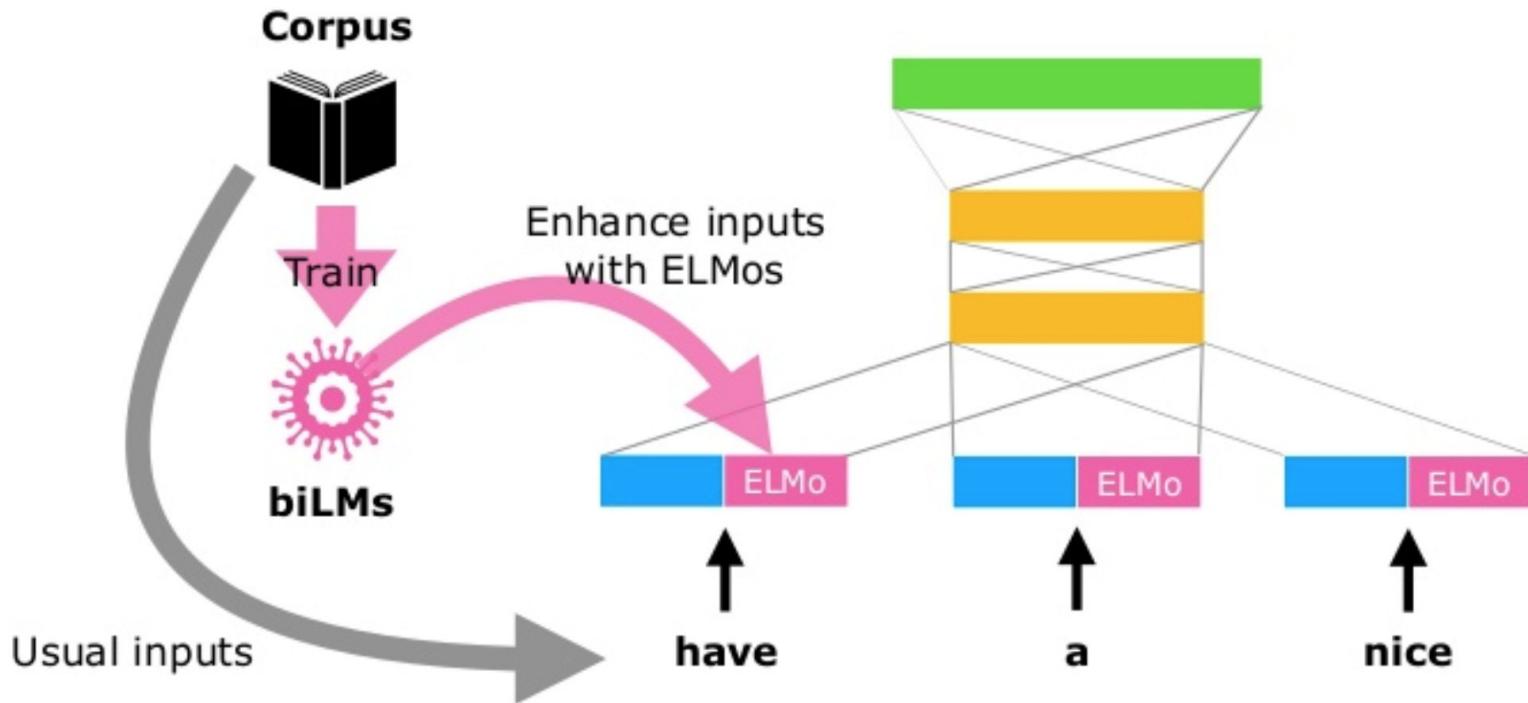
3- Sum the (now weighted) vectors



ELMo embedding of “stick” for this task in this context

ELMo

ELMo can be integrated to almost all neural NLP tasks with simple concatenation to the embedding layer



ELMo: overview

- Pretrained ELMo models: <http://allennlp.org/elmo>
- AllenNLP is a library on the top of PyTorch
- Higher levels seems to catch semantics while lower layer probably capture syntactic features



BERT

Bidirectional Encoder Representations from Transformers

BERT

Input
Features

Help Prince Mayuko Transfer
Huge Inheritance



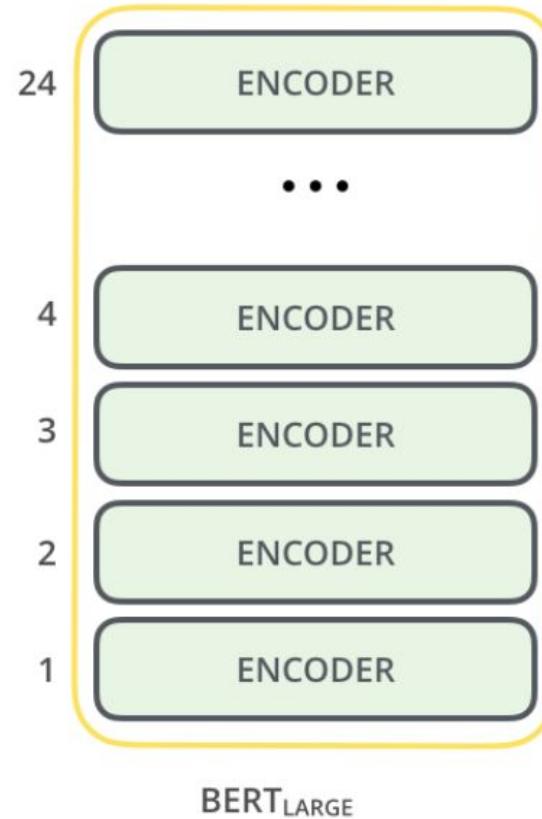
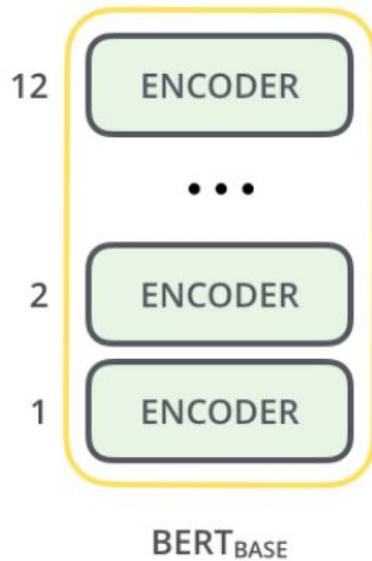
Classifier
(Feed-forward
neural network +
softmax)



85% Spam
15% Not Spam

Output
Prediction

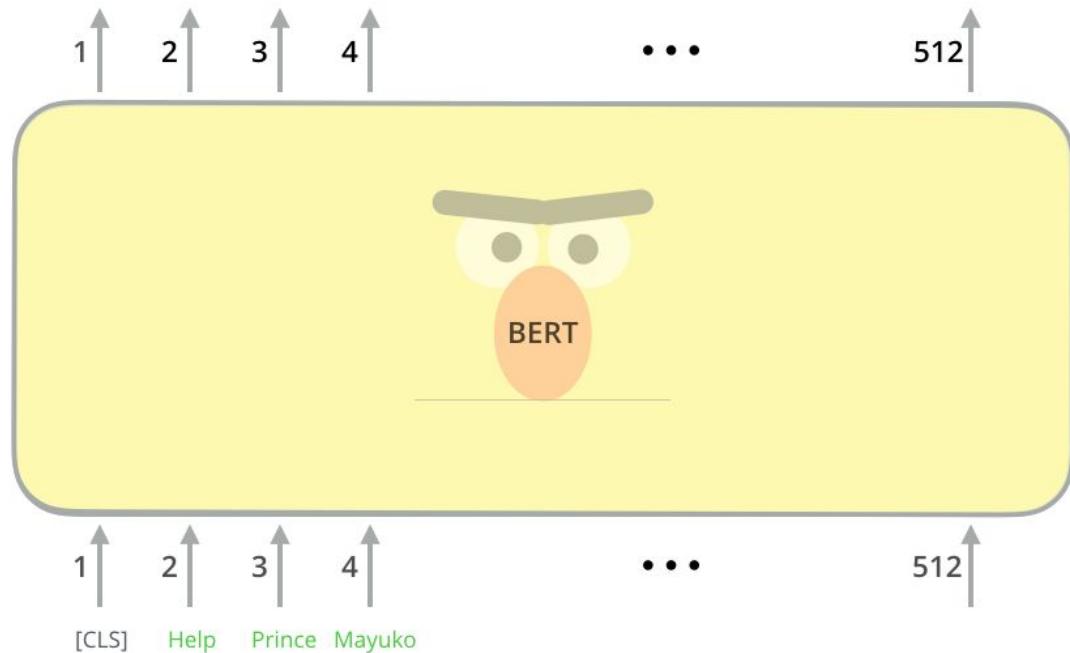
BERT: base and large



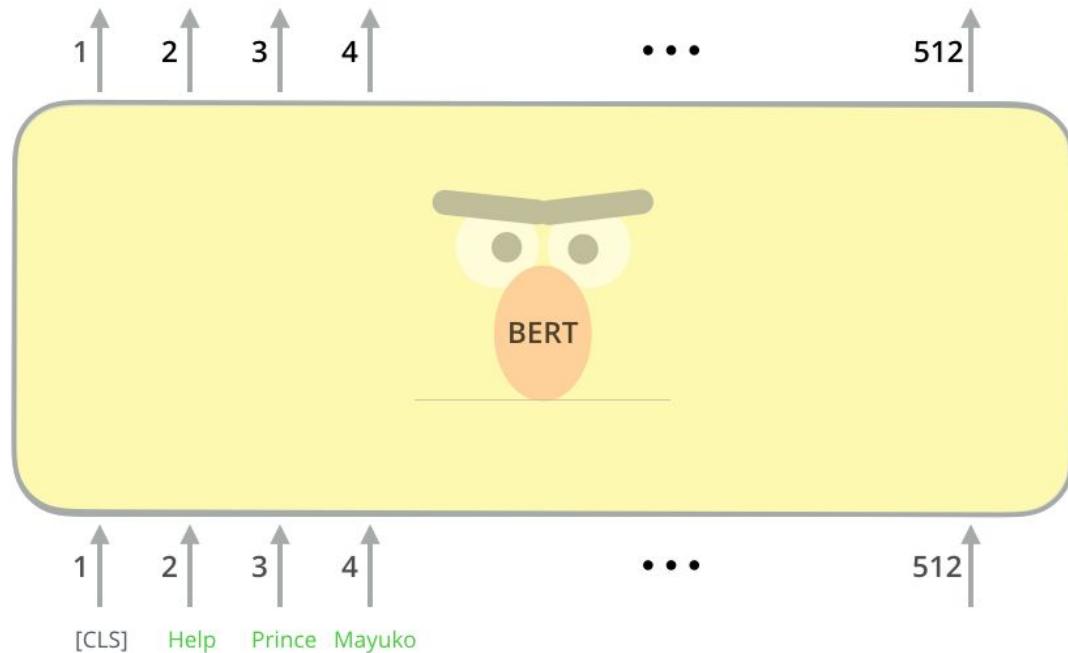
BERT vs. Transformer

	 THE TRANSFORMER	 BERT	
		Base BERT	Large BERT
Encoders	6	12	24
Units in FFN	512	768	1024
Attention Heads	8	12	16

Model inputs

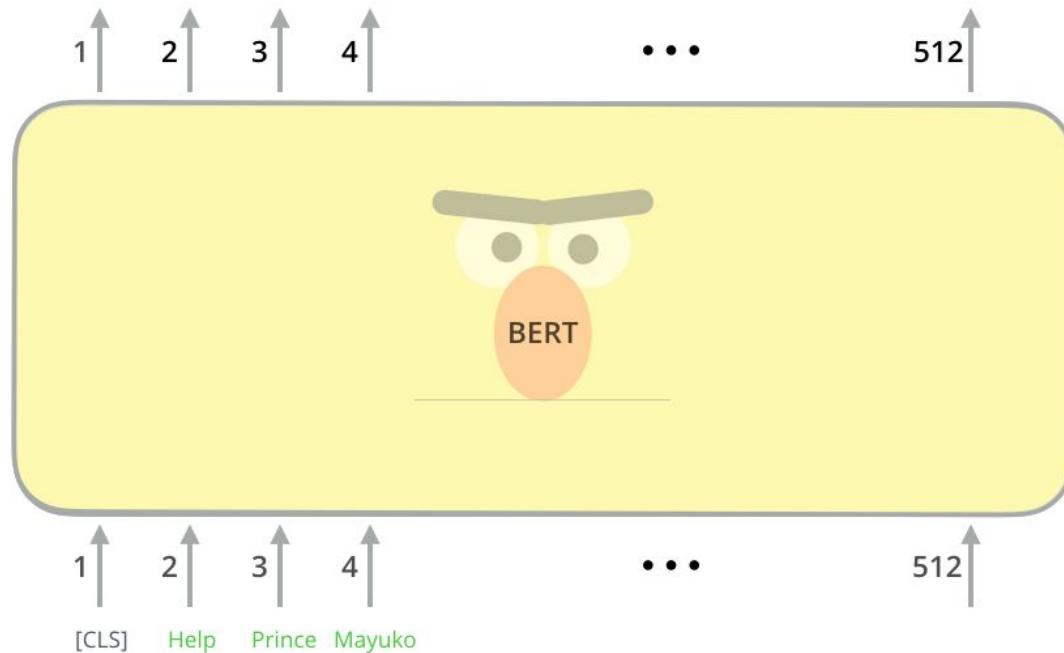


Model inputs



Identical to the Transformer up until this point

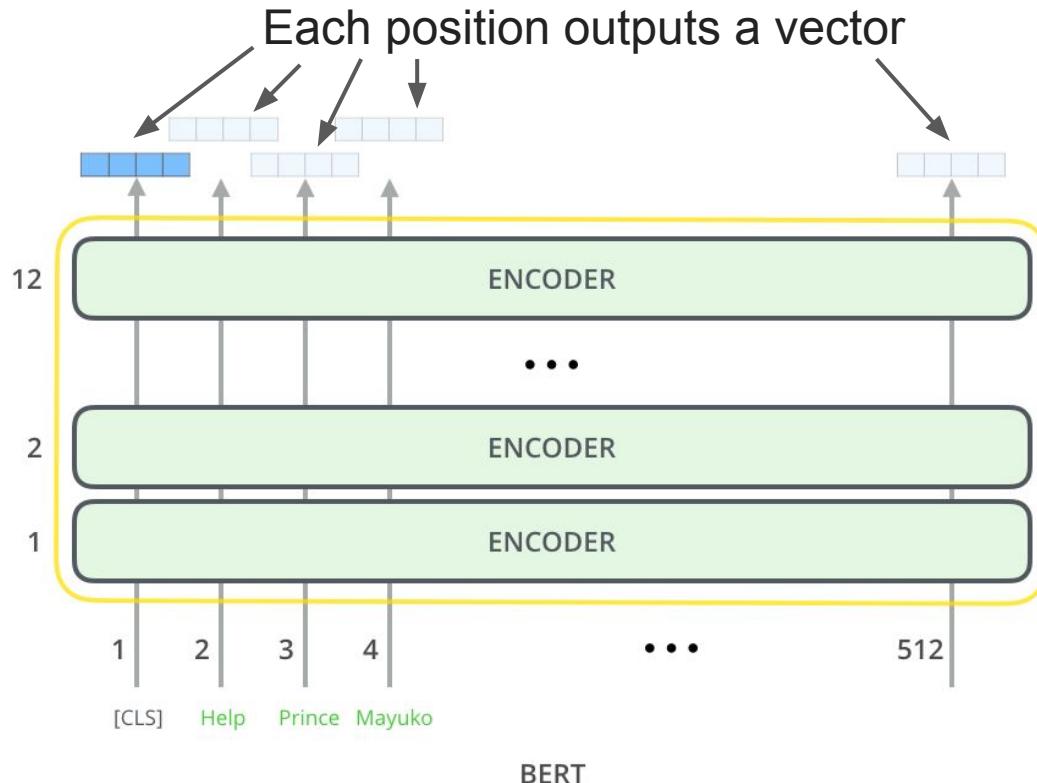
Model inputs



Identical to the Transformer up until this point

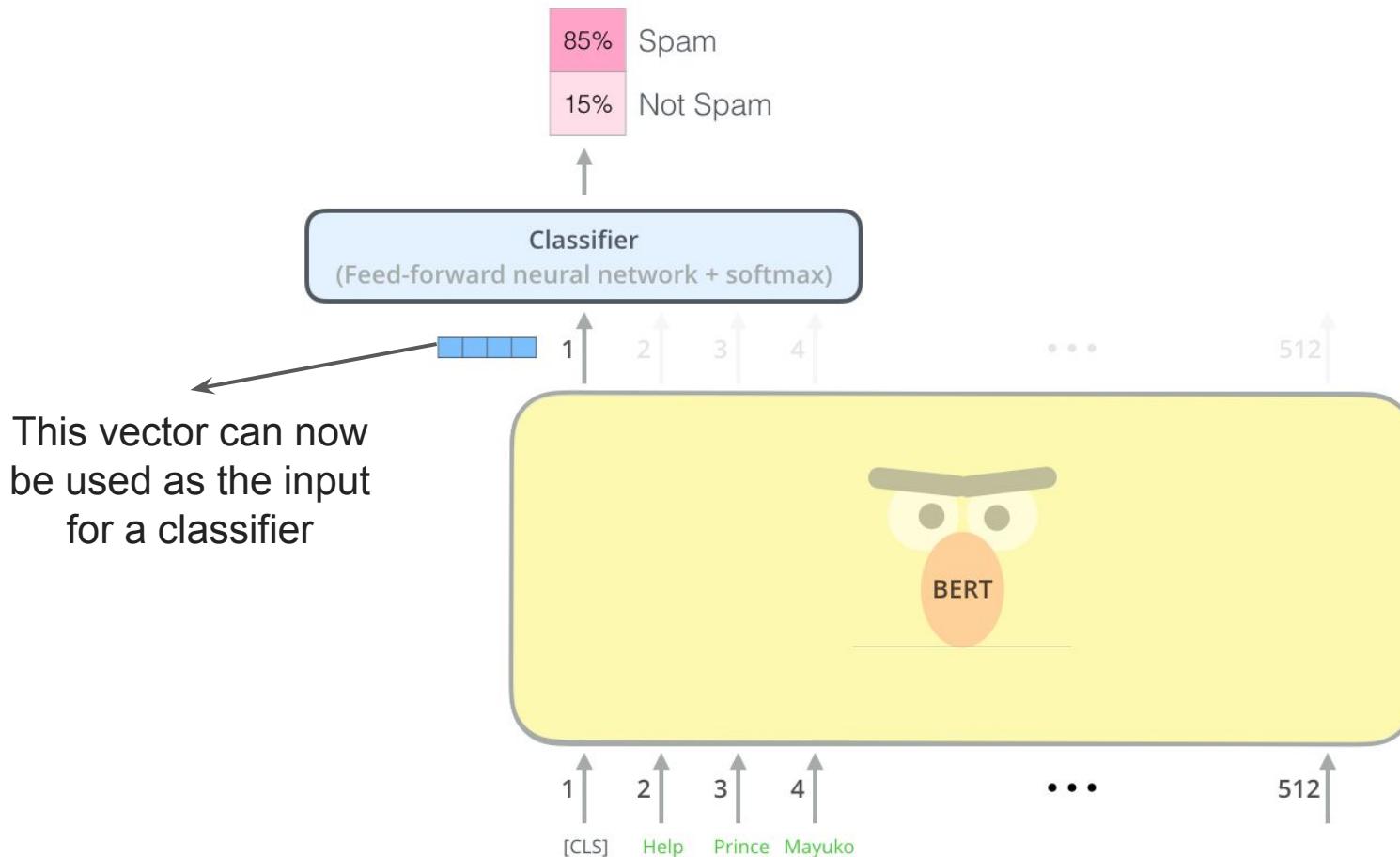
Why is BERT so special?

Model outputs

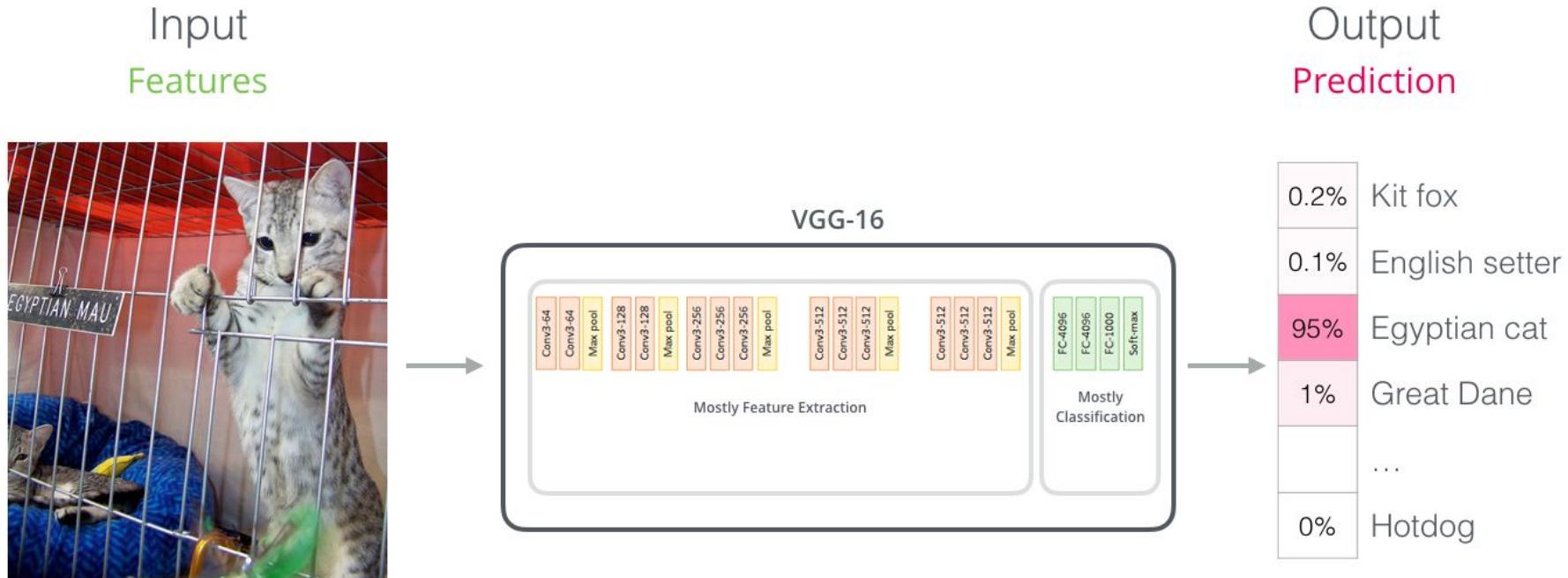


For sentence classification we focus on the first position (that we passed [CLS] token to)

Model inputs

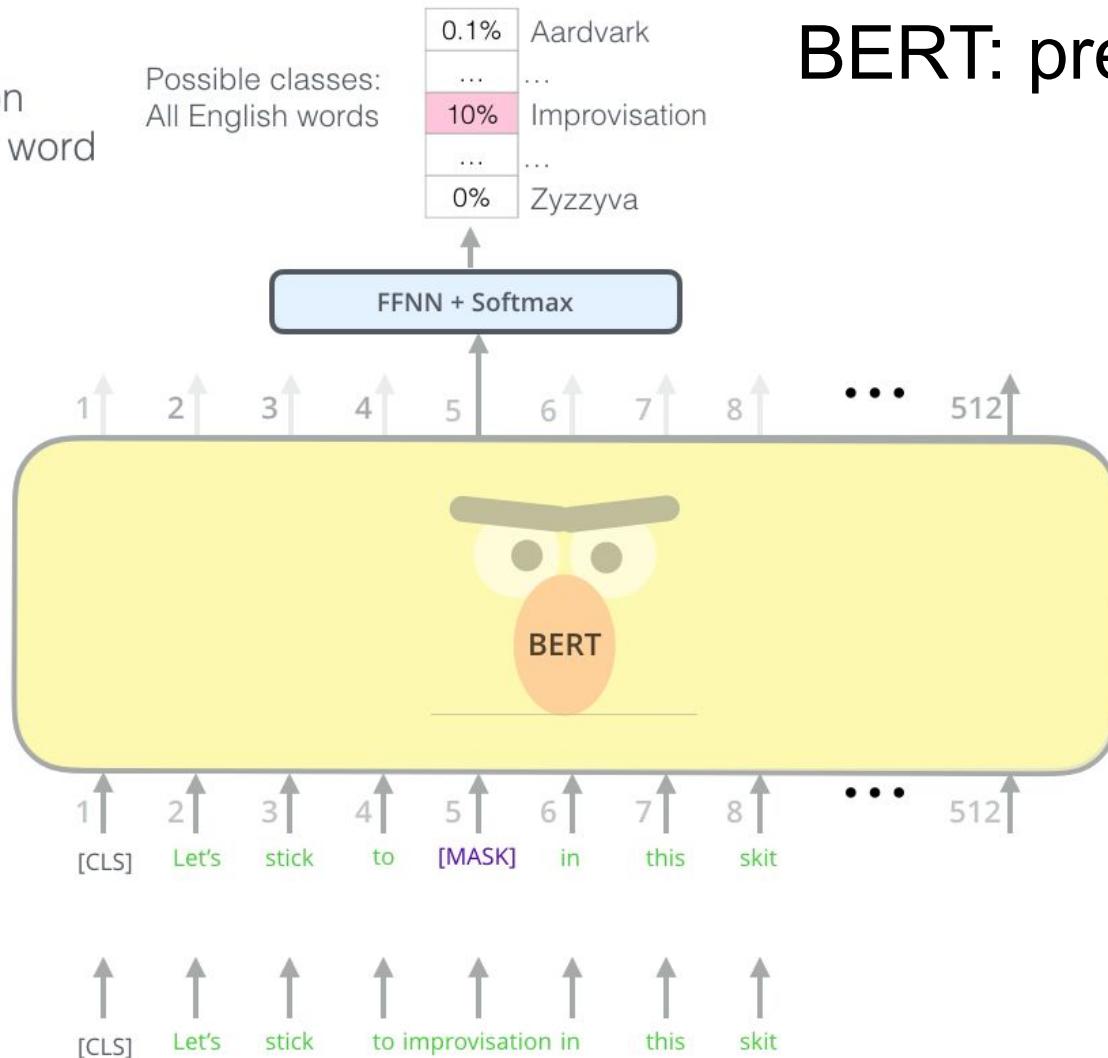


Similar to CNN concept!



BERT: pre-training

Use the output of the masked word's position to predict the masked word



BERT: pre-training

- “Masked Language Model” approach
- To make BERT better at handling relationships between multiple sentences, the pre-training process includes an additional task:

“Given two sentences (A and B), is B likely to be the sentence that follows A, or not?”

BERT: pre-training

Predict likelihood
that sentence B
belongs after
sentence A



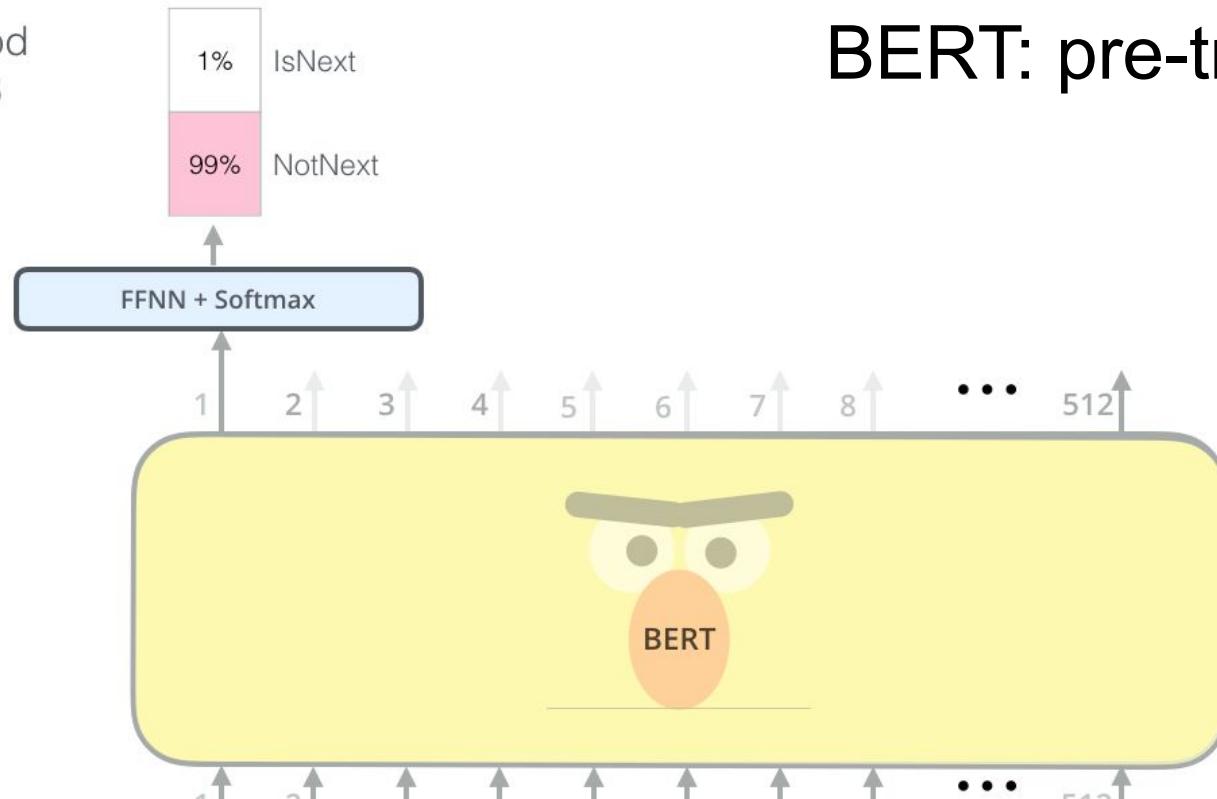
Tokenized
Input

1 [CLS] 2 the man 3 [MASK] 4 to 5 the 6 store 7 [SEP] ... 512

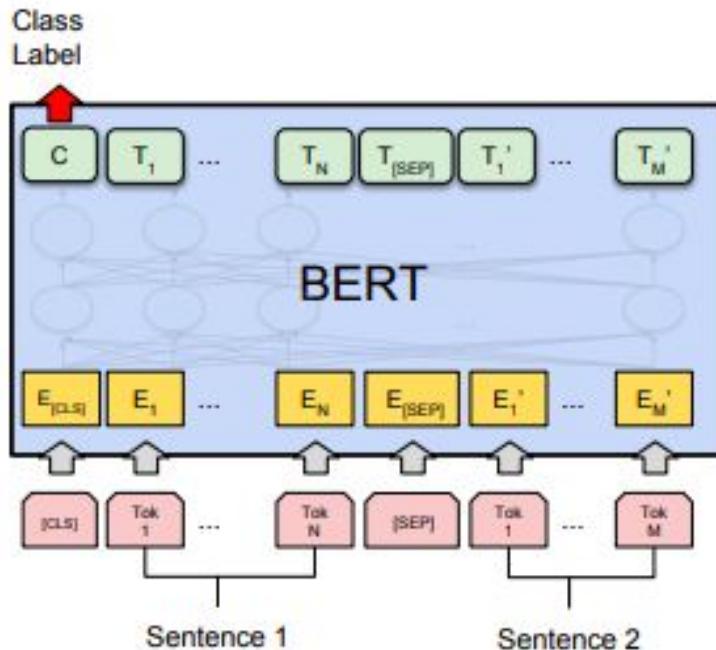
Input

[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]

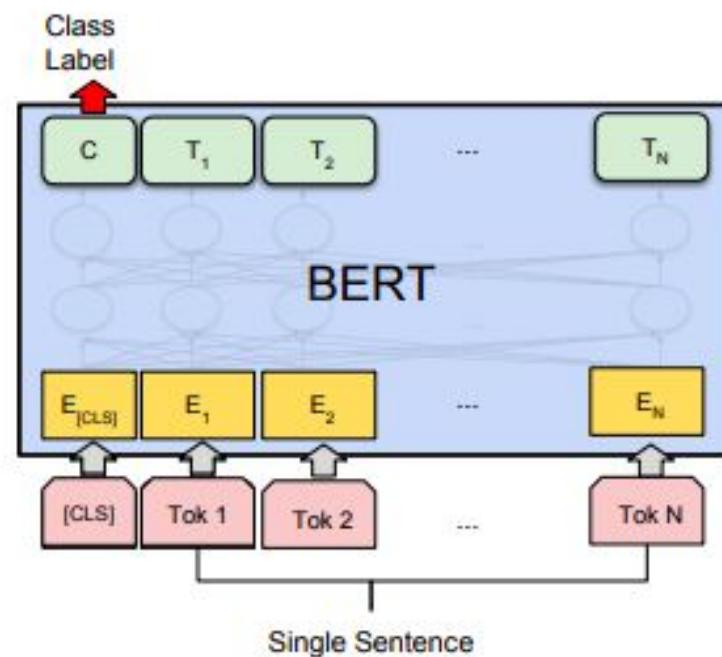
Sentence A Sentence B



BERT: fine-tuning for different tasks

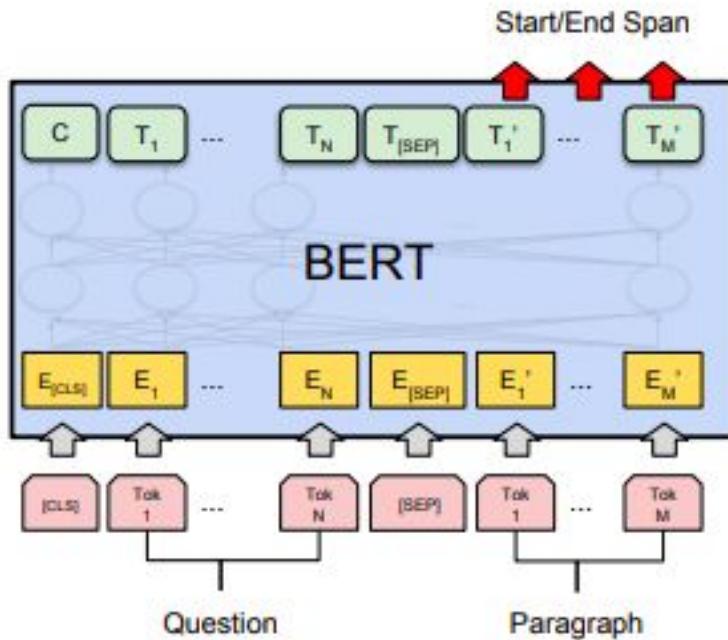


(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG

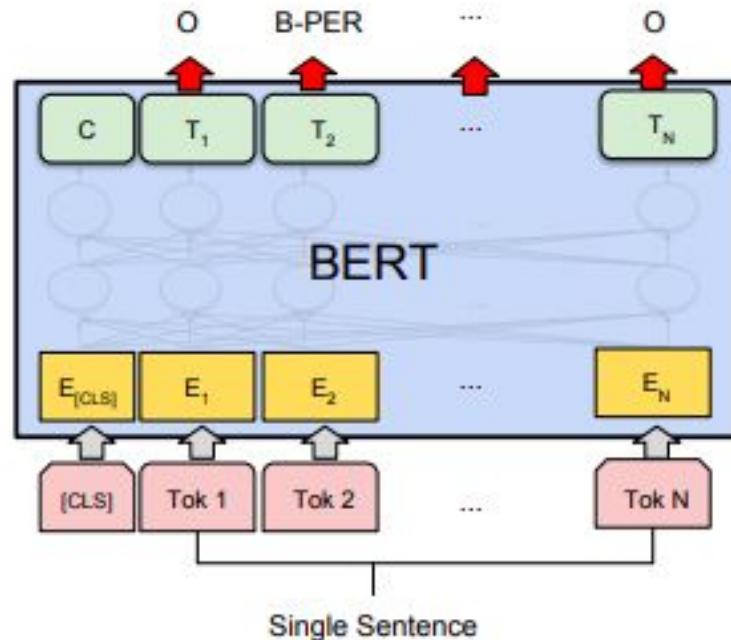


(b) Single Sentence Classification Tasks:
SST-2, CoLA

BERT: fine-tuning for different tasks

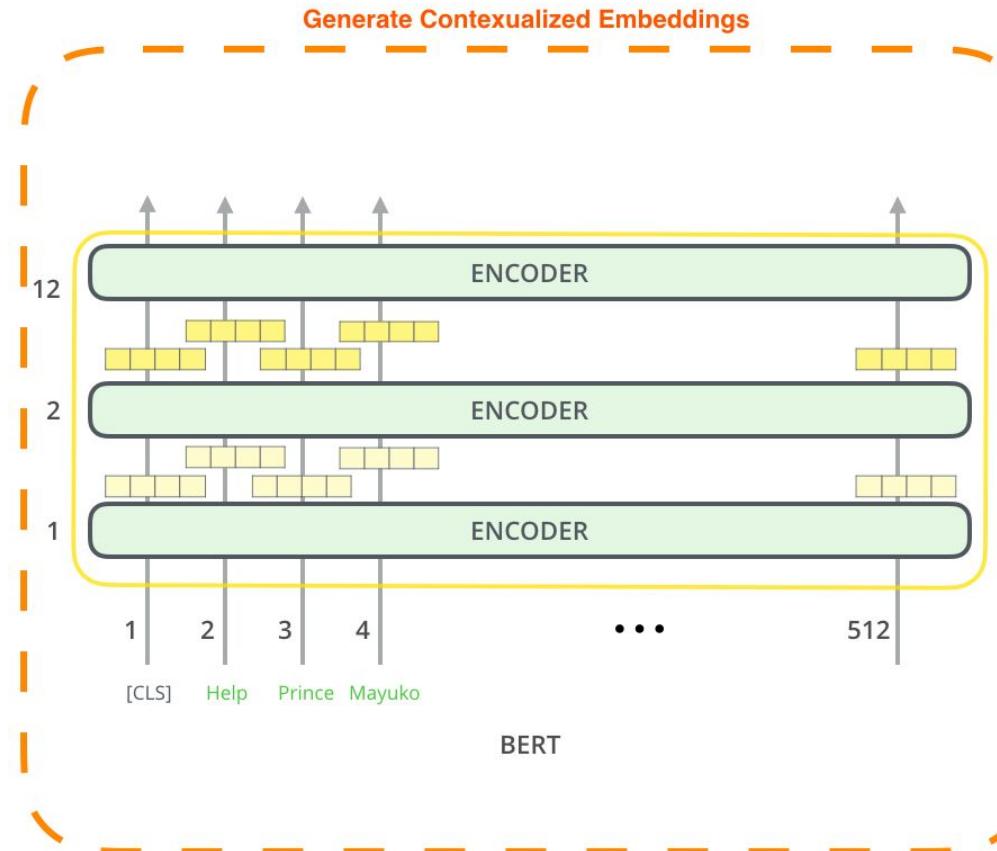


(c) Question Answering Tasks:
SQuAD v1.1

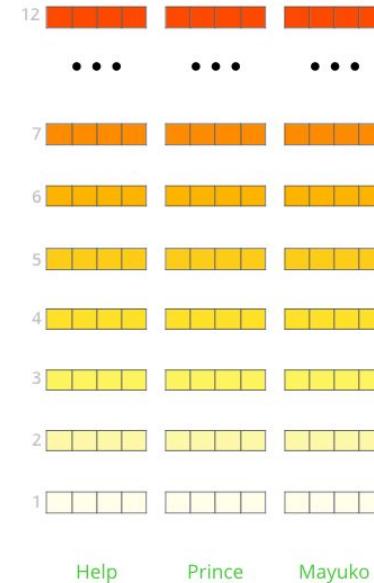


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

BERT for feature extraction



The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

BERT for feature extraction

What is the best contextualized embedding for “**Help**” in that context?

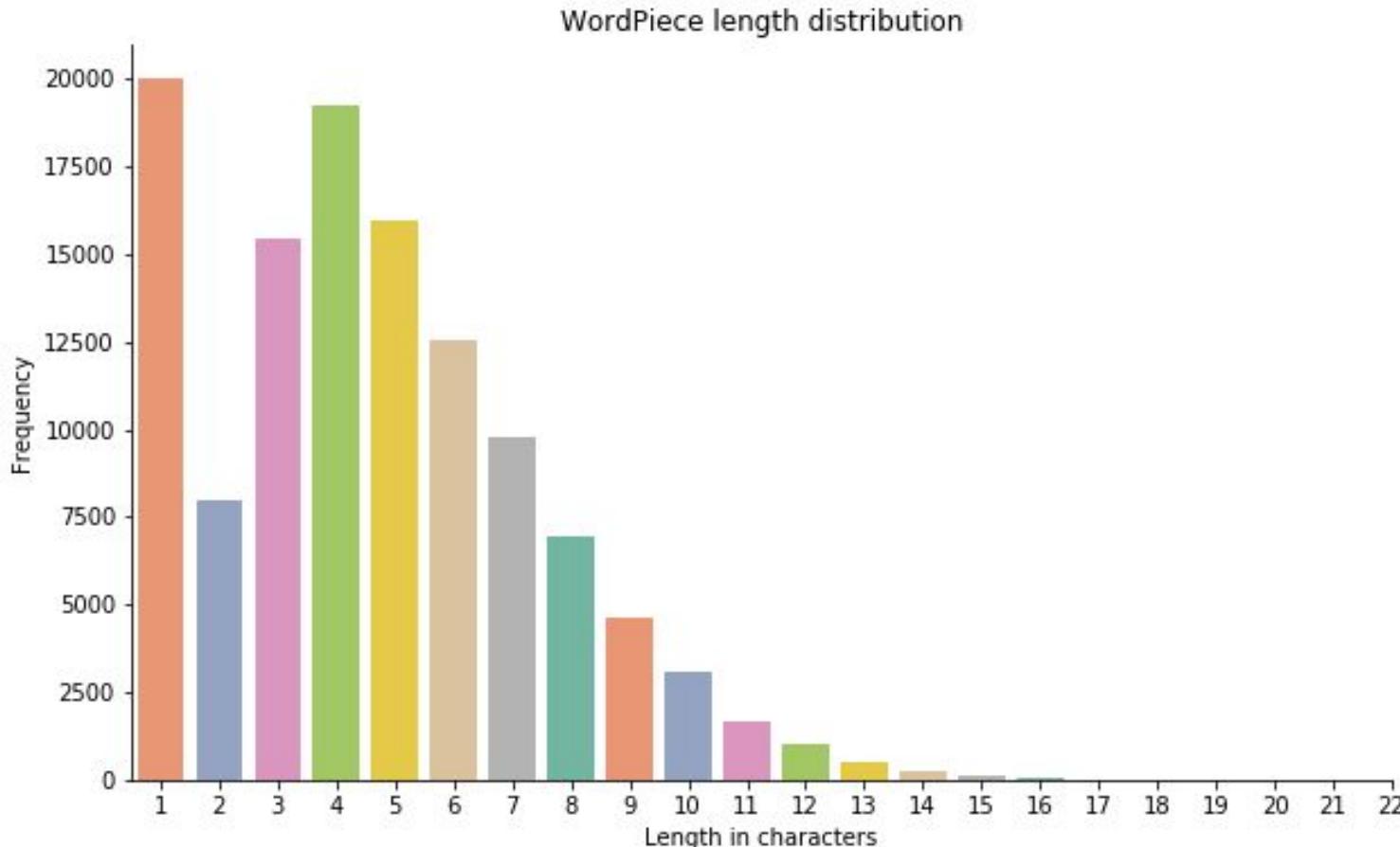
For named-entity recognition task CoNLL-2003 NER

		Dev F1 Score
12		
• • •		
7		
6		
5		
4		
3		
2		
1		
Help		
First Layer	Embedding	91.0
Last Hidden Layer		94.9
Sum All 12 Layers		95.5
Second-to-Last Hidden Layer		95.6
Sum Last Four Hidden		95.9
Concat Last Four Hidden		96.1

Example: Unaffable -> un, ##aff, ##able

- Single model for 104 languages with a large shared vocabulary (119,547 [WordPiece](#) model)
- Non-word-initial units are prefixed with ##
- The first 106 symbols: constants like PAD and UNK
- 36.5% of the vocabulary are non-initial word pieces
- The alphabet consists of 9,997 unique characters that are defined as word-initial (C) and continuation symbols (##C), which together make up 19,994 word pieces
- The rest are multicharacter word pieces of various length.

BERT: tokenization

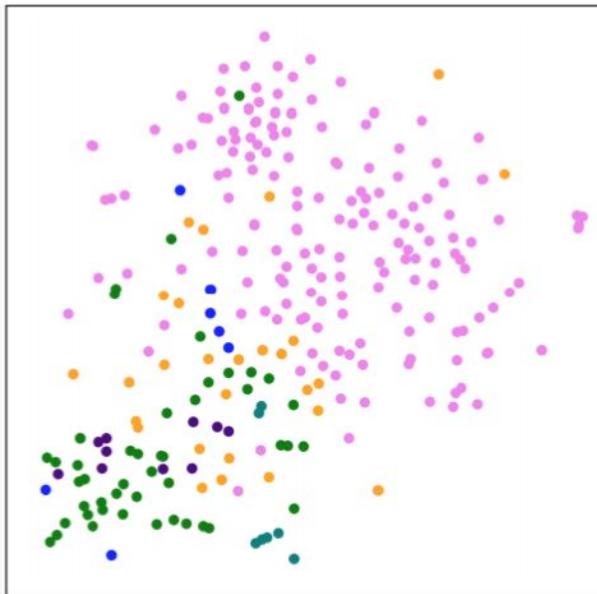


BERT: overview

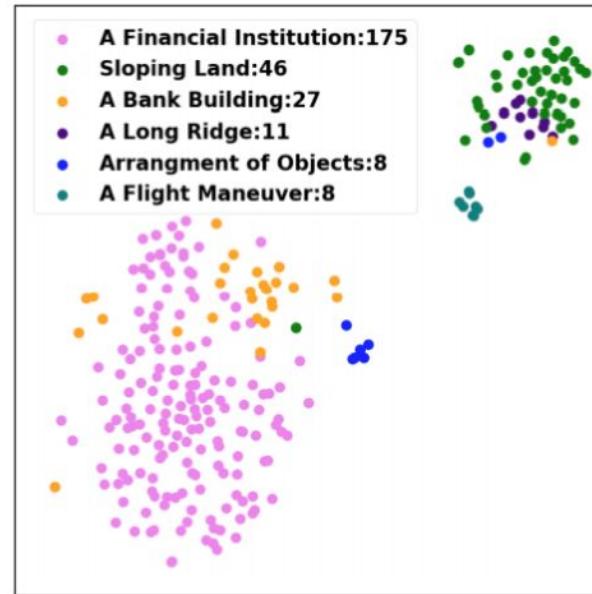
- [BERT repo](#)
- [Try out BERT on TPU](#)
- [WordPieces Tokenizer](#)
- [PyTorch Implementation of BERT](#)

BERT vs. ELMO (Word Sense Disambiguation problem)

T-SNE plots of different
senses of ‘bank’



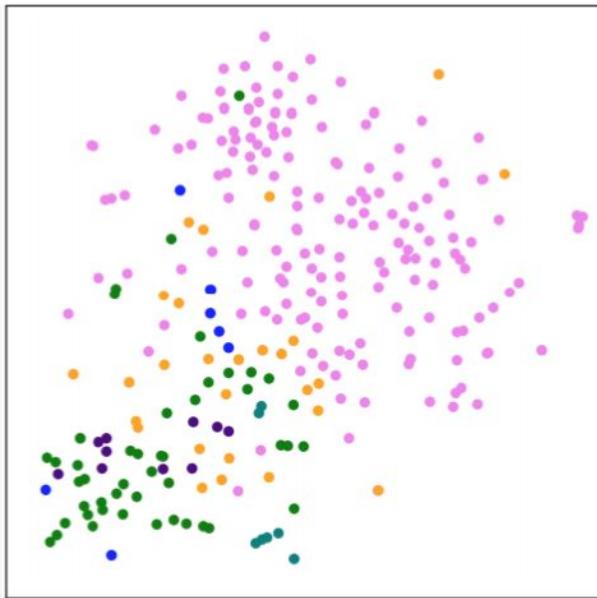
(c) ELMo



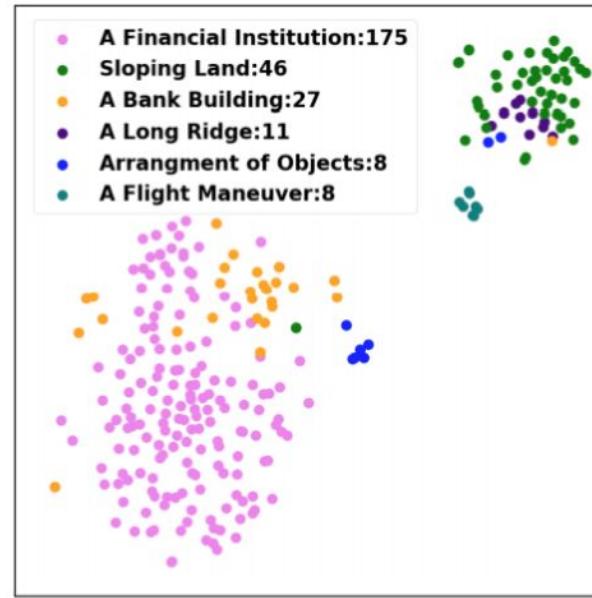
(a) BERT

BERT vs. ELMO (Word Sense Disambiguation problem)

T-SNE plots of different
senses of ‘bank’



(c) ELMo

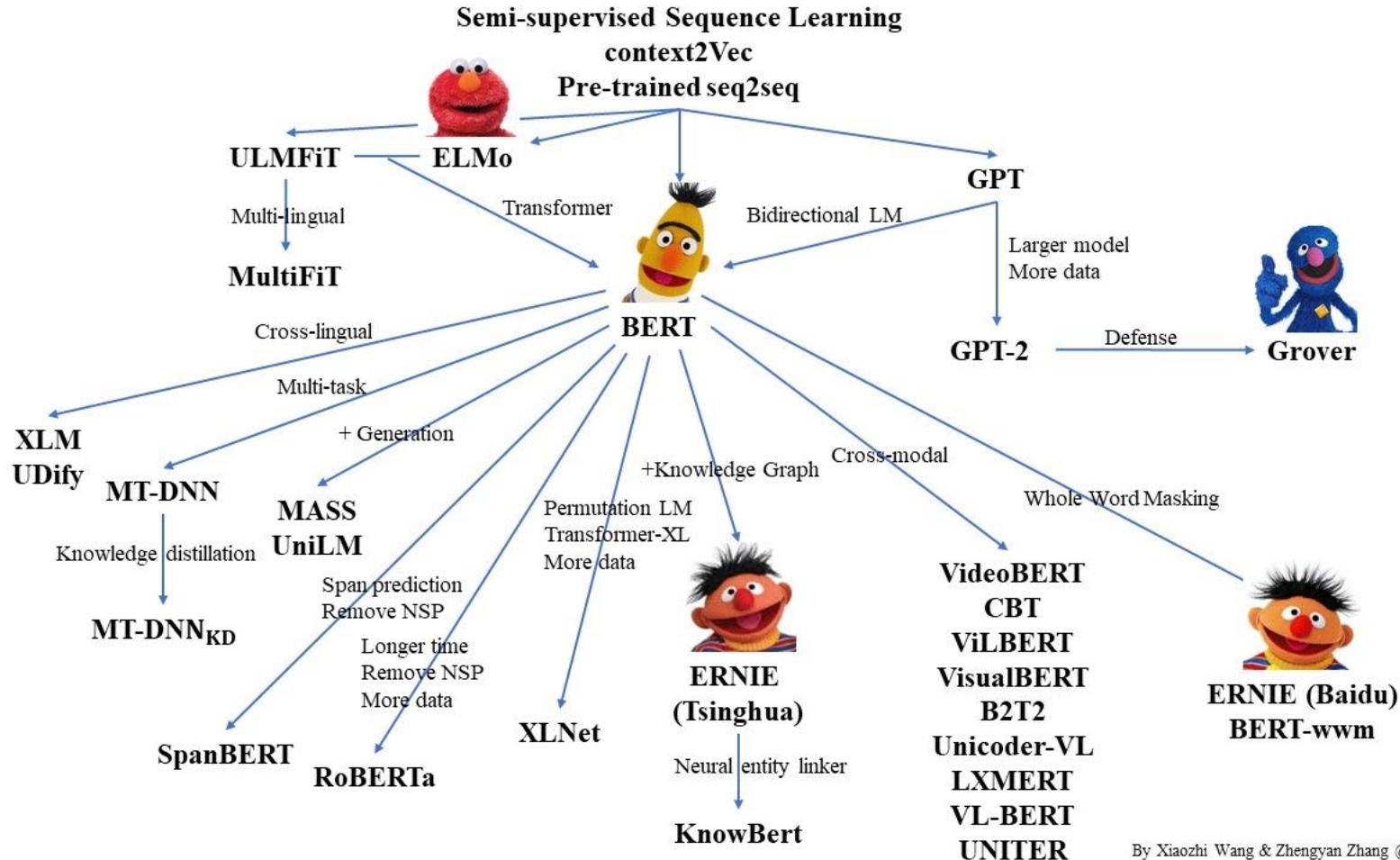


(a) BERT

BERTology

BERTology paper

BERTology



BERTology



 **Miles Brundage**
@Miles_Brundage 

2018: Language model papers have to introduce Sesame Street-related acronyms

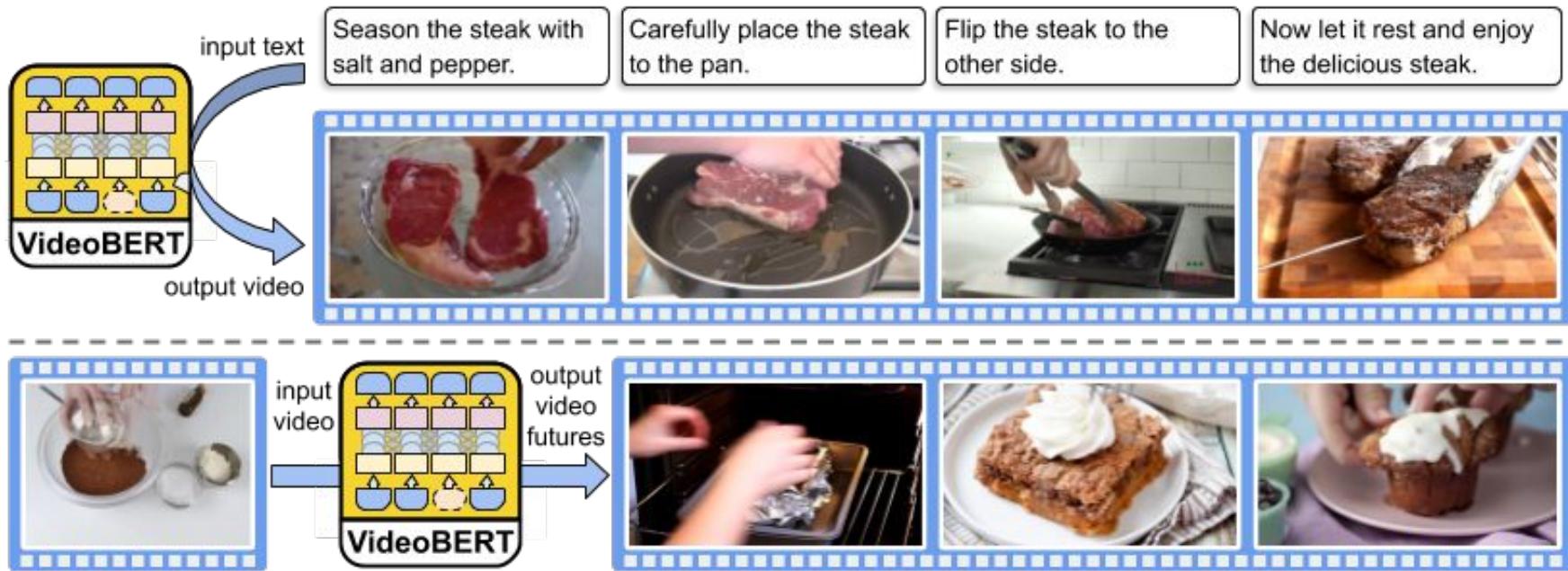
2019: Language model papers need Sesame Street jokes in the title, all talks need at least one Sesame Street image.

2020: ACL/NAACL co-located with Sesame Street convention, Big Bird gives a keynote.

 293 2:46 AM - Jun 12, 2019 

 57 people are talking about this 

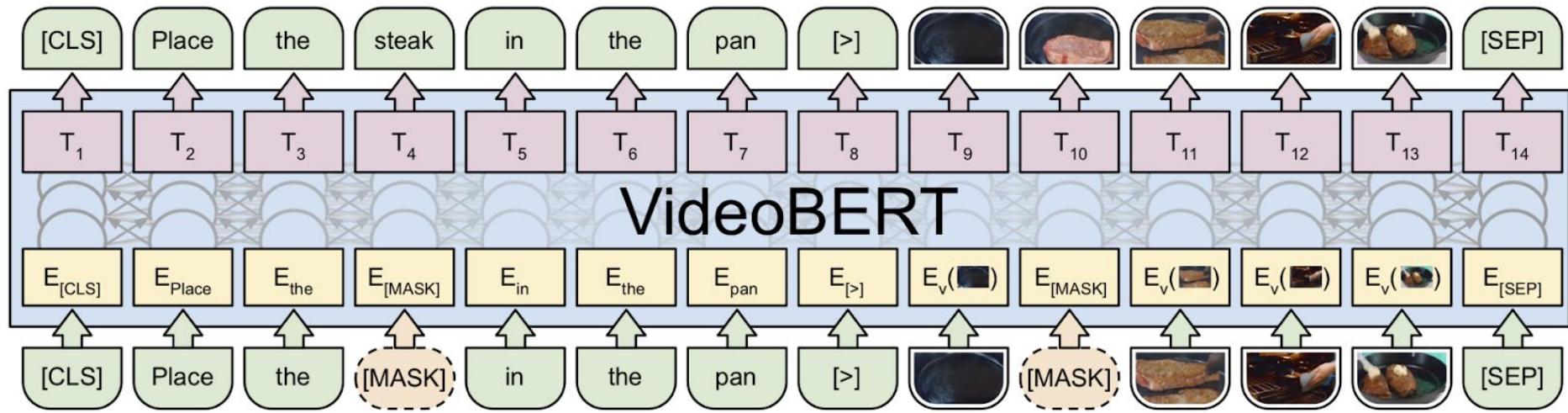
VideoBERT (ICCV 2019)



<https://arxiv.org/abs/1904.01766>

<https://ai.googleblog.com/2019/09/learning-cross-modal-temporal.html>

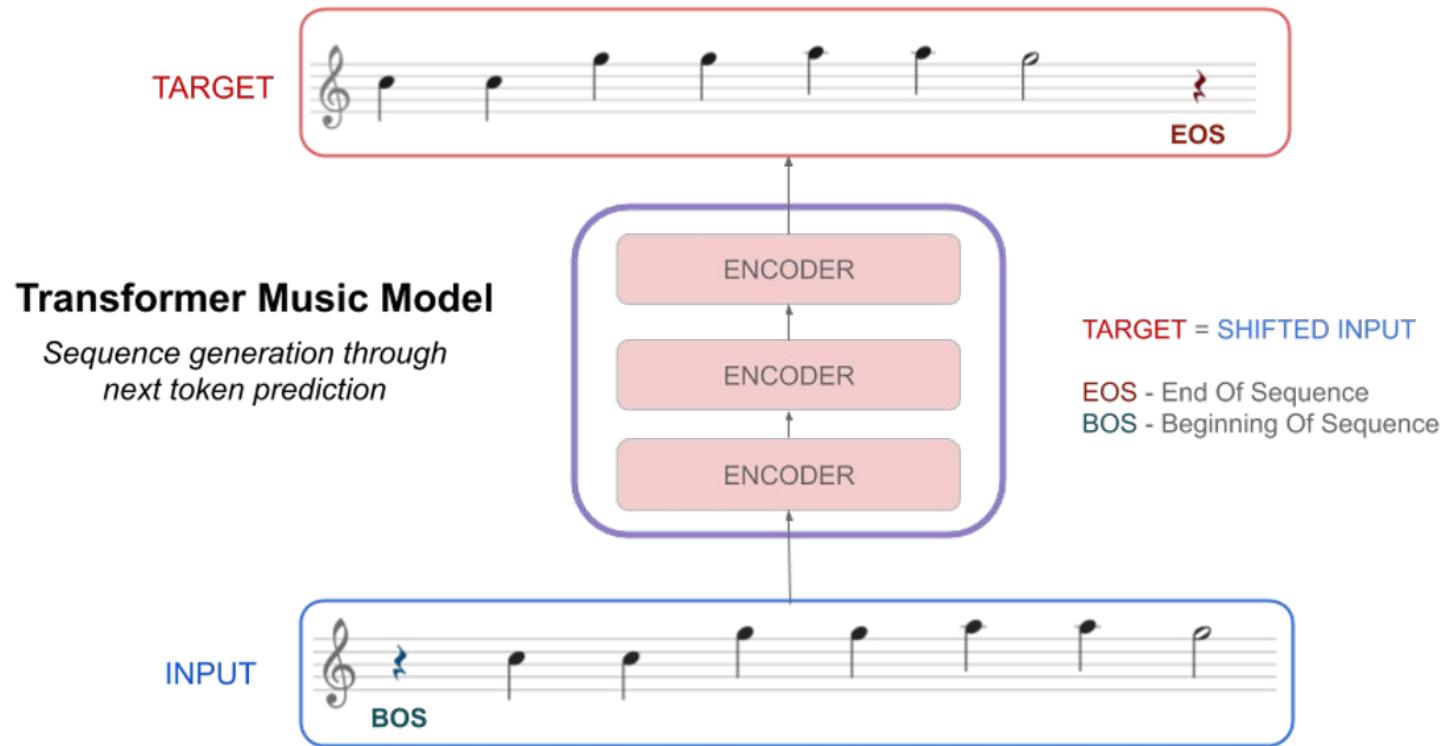
VideoBERT (ICCV 2019)



<https://arxiv.org/abs/1904.01766>

<https://ai.googleblog.com/2019/09/learning-cross-modal-temporal.html>

Music Transformer (ICLR 2019)



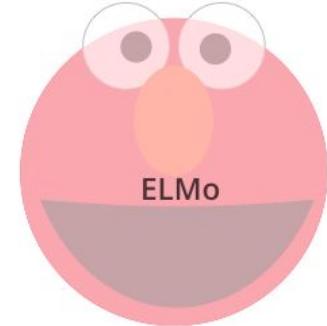
<https://arxiv.org/abs/1809.04281>

<https://magenta.tensorflow.org/music-transformer>

Fun demos to play with

Get a neural network to autocomplete your thoughts

And yet another auto-completion tool



GPT, GPT-2 and GPT-3

- Transformer-based architecture
- Trained to predict the **next** word
- 1.5 billion parameters
- Trained on 8 million web-pages



- Transformer-based architecture
- trained to predict the **next** word
- 1.5 billion parameters
- Trained on 8 million web-pages

On language tasks (question answering, reading comprehension, summarization, translation) works well **WITHOUT** fine-tuning

GPT-2: question answering

EXAMPLES

Who wrote the book the origin of species?

Correct answer: *Charles Darwin*

Model answer: Charles Darwin

What is the largest state in the U.S. by land mass?

Correct answer: *Alaska*

Model answer: California

GPT-2: language modeling

EXAMPLE

Both its sun-speckled shade and the cool grass beneath were a welcome respite after the stifling kitchen, and I was glad to relax against the tree's rough, brittle bark and begin my breakfast of buttery, toasted bread and fresh fruit. Even the water was tasty, it was so clean and cold. It almost made up for the lack of...

Correct answer: coffee

Model answer: food

GPT-2: machine translation

EXAMPLE

French sentence:

Un homme a expliqué que l'opération gratuite qu'il avait subie pour soigner une hernie lui permettrait de travailler à nouveau.

Reference translation:

One man explained that the free hernia surgery he'd received will allow him to work again.

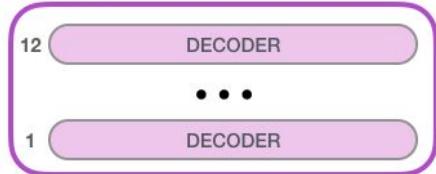
Model translation:

A man told me that the operation gratuity he had been promised would not allow him to travel.

GPT-2



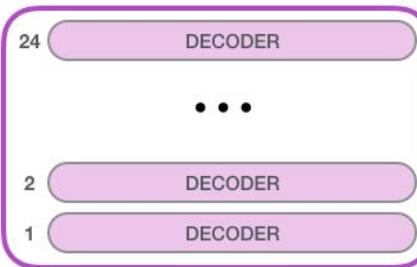
GPT-2
SMALL



Model Dimensionality: 768



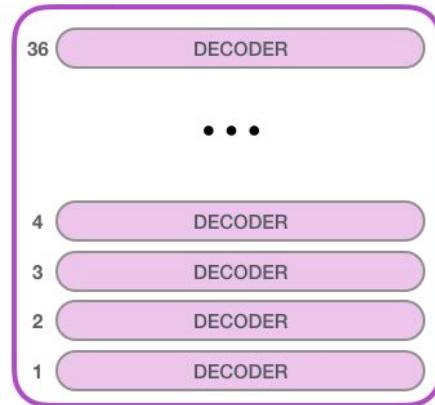
GPT-2
MEDIUM



Model Dimensionality: 1024



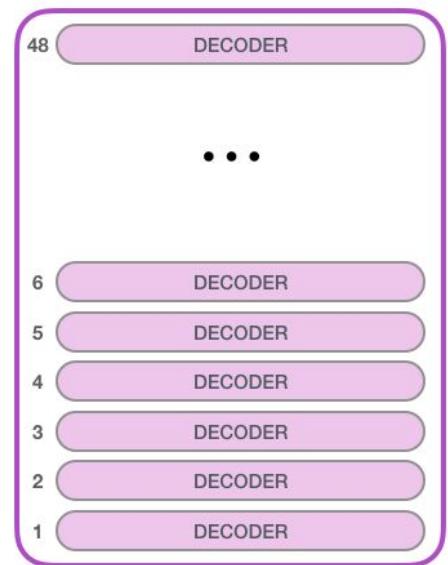
GPT-2
LARGE



Model Dimensionality: 1280



GPT-2
EXTRA
LARGE



Model Dimensionality: 1600

New AI fake text generator may be too dangerous to ... - The Guardian

<https://www.theguardian.com/.../elon-musk-backed-ai-writes-convincing-news-fiction>

4 days ago - The Elon Musk-backed nonprofit company OpenAI declines to release research publicly for fear of misuse. The creators of a revolutionary AI system that can write news stories and works of fiction – dubbed "deepfakes for text" – have taken the unusual step of not releasing ...

OpenAI built a text generator so good, it's considered too dangerous to ...

<https://techcrunch.com/2019/02/17/openai-text-generator-dangerous/> ▾

12 hours ago - A storm is brewing over a new language model, built by non-profit artificial intelligence research company OpenAI, which it says is so good at ...

The AI Text Generator That's Too Dangerous to Make Public | WIRED

<https://www.wired.com/story/ai-text-generator-too-dangerous-to-make-public/> ▾

4 days ago - In 2015, car-and-rocket man Elon Musk joined with influential startup backer Sam Altman to put artificial intelligence on a new, more open ...

Elon Musk-backed AI Company Claims It Made a Text Generator ...

<https://gizmodo.com/elon-musk-backed-ai-company-claims-it-made-a-text-gener-183...> ▾

Elon Musk-backed AI Company Claims It Made a Text Generator That's Too Dangerous to Release · Rhett Jones · Friday 12:15pm · Filed to: OpenAI Filed to: ...

Scientists have made an AI that they think is too dangerous to ...

<https://www.weforum.org/.../amazing-new-ai-churns-out-coherent-paragraphs-of-text/> ▾

3 days ago - Sample outputs suggest that the AI system is an extraordinary step forward, producing text rich with context, nuance and even something ...

New AI Fake Text Generator May Be Too Dangerous To ... - Slashdot

<https://news.slashdot.org/.../new-ai-fake-text-generator-may-be-too-dangerous-to-rele...> ▾

3 days ago - An anonymous reader shares a report: The creators of a revolutionary AI system that can write news stories and works of fiction – dubbed ...

GPT-2: fake news and hype

Top stories



OpenAI built a text generator so good, it's considered too dangerous to release

TechCrunch

11 hours ago



Elon Musk's AI company created a fake news generator it's too scared to make public

BGR.com

9 hours ago



The AI That Can Write A Fake News Story From A Handful Of Words

NDTV.com

2 hours ago

When Is Technology Too Dangerous to Release to the Public?

Slate • 2 days ago



Scientists Developed an AI So Advanced They Say It's Too Dangerous to Release

ScienceAlert • 6 days ago



- GPT-2: 1.5 billion parameters
- GPT-3: **175 billion** parameters



Geoffrey Hinton @geoffreyhinton · Jun 10

Extrapolating the spectacular performance of GPT3 into the future suggests that the answer to life, the universe and everything is just 4.398 trillion parameters.

62

643

3.4K



GPT-3: OpenAI API released

- You can request access in order to integrate the API into your product
- Given any text prompt, the API will return a text completion, attempting to match the pattern you gave it

References

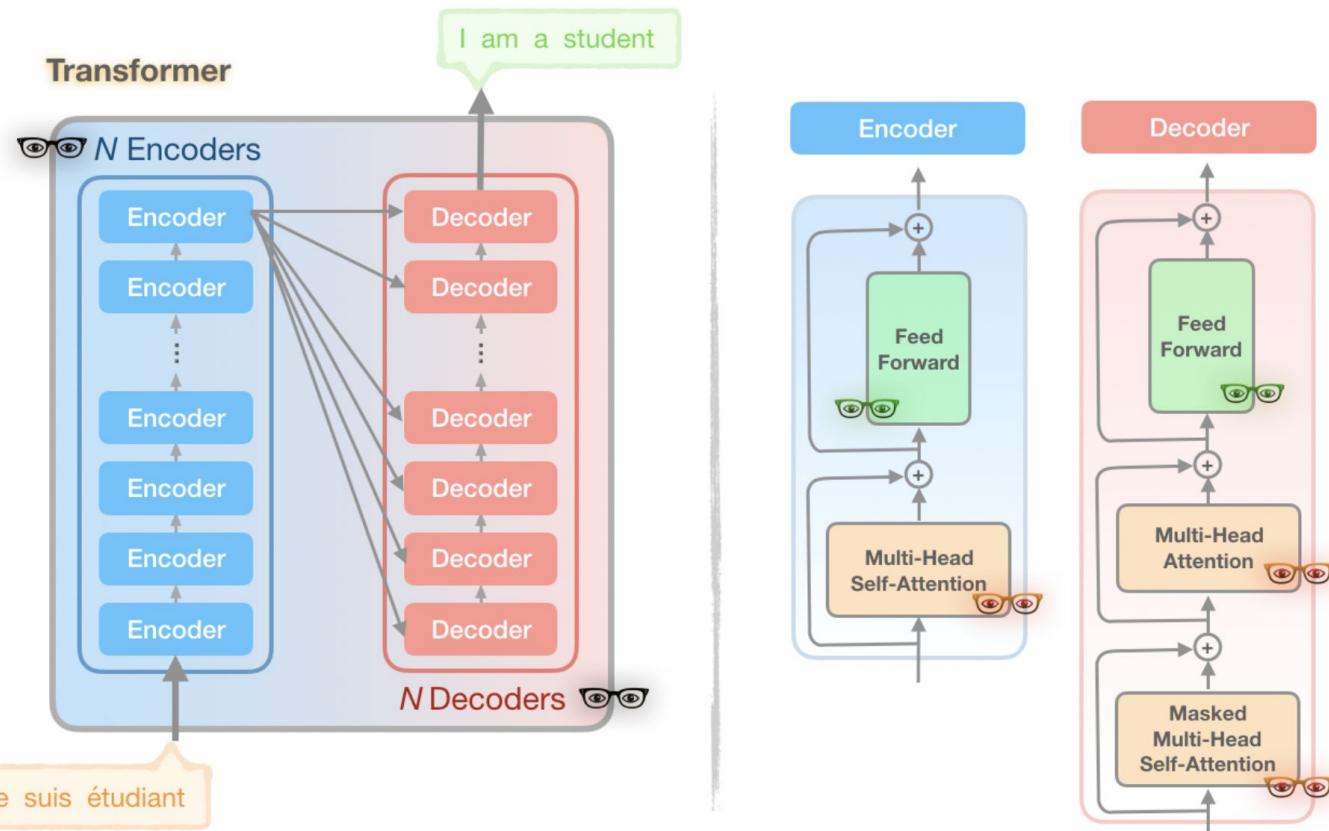
- [Transformer](#)
- [OpenAI Transformer](#)
- [ELMO](#)
- [BERT](#)
- [BERTology](#)
- [GPT](#)
- [GPT-2](#)
- [GPT-3](#)

The Reformer

Reformer: The Effective Transformer (ICLR 2020)

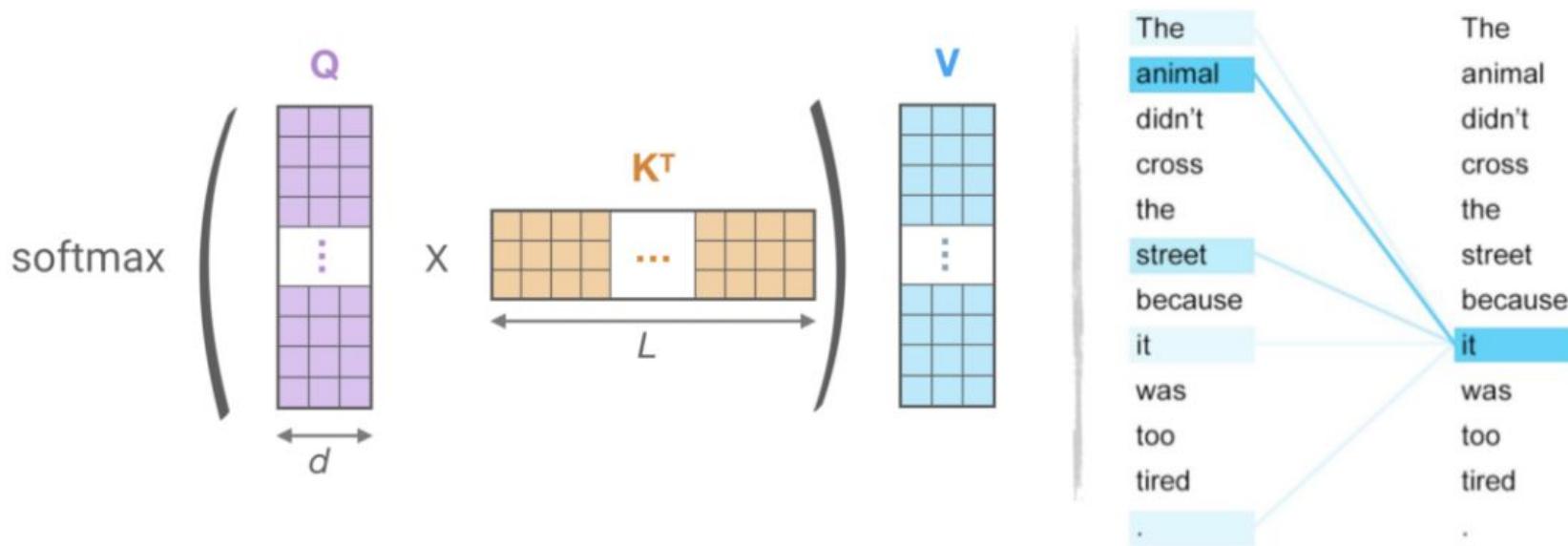
Transformer-based models have a problem:

- They require lots of GPUs to train
 - even cannot be fine-tuned on a single GPU



- Problem 1 (**Red** 🕶️): Attention computation
- Problem 2 (**Black** 🕶️): Large number of layers
- Problem 3 (**Green** 🕶️): Depth of feed-forward layers

Attention computation

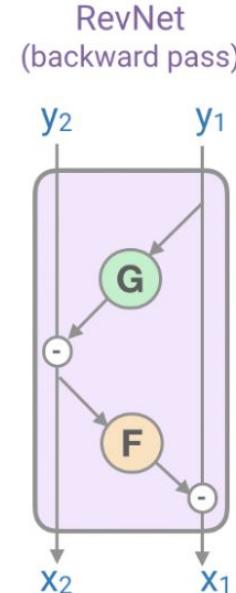
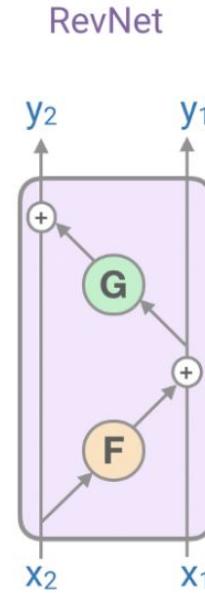
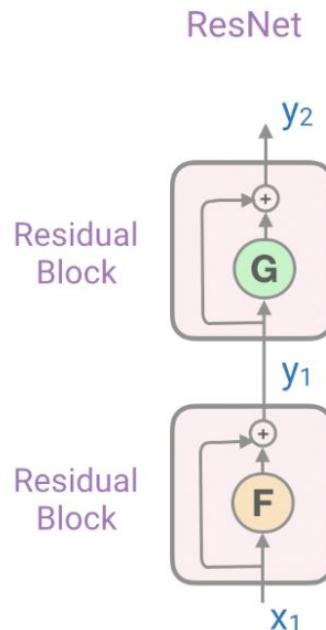


- Replace dot-product attention with locality-sensitive hashing (LSH)
 - changes the complexity from $O(L^2)$ to $O(L \log L)$

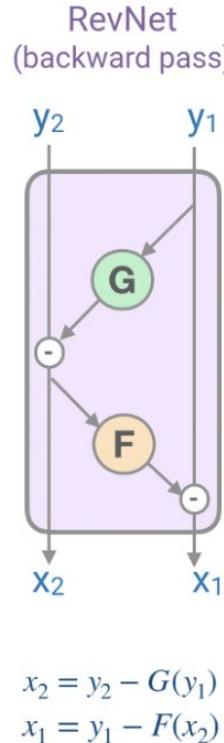
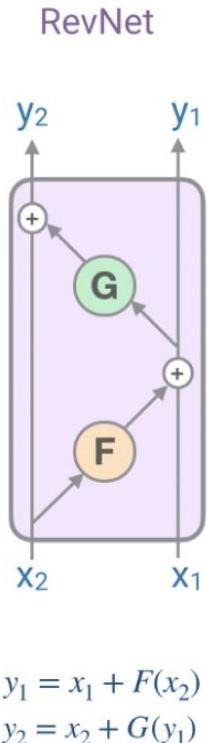
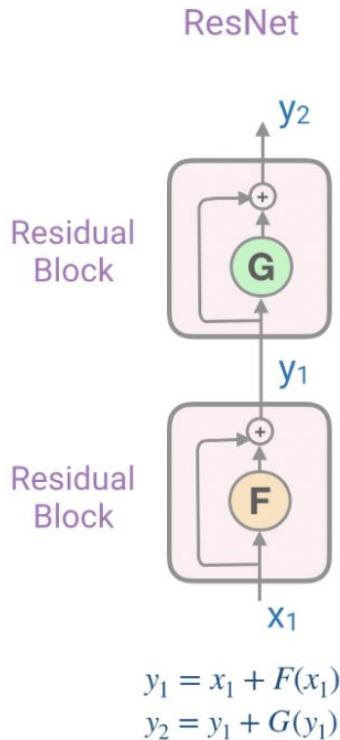
Locality-sensitive hashing for attention computation

- LSH - an efficient and approximate way of nearest neighbors search in high dimensional datasets.
- The main idea behind LSH is to select hash functions such that for two points ‘p’ and ‘q’, if ‘q’ is close to ‘p’ then with good enough probability we have ‘ $\text{hash}(q) == \text{hash}(p)$ ’.

Reversible Transformer



Reversible Transformer



- F - self-attention block
- G - feed-forward layer

Profit: storing activations only once during the training process

Chunking

Computations in feed-forward layers are independent across positions in a sequence => the computations for the forward and backward passes can be split into chunks.

$$Y_2 = \left[Y_2^{(1)}; \dots; Y_2^{(c)} \right] = \left[X_2^{(1)} + \text{FeedForward}(Y_1^{(1)}); \dots; X_2^{(c)} + \text{FeedForward}(Y_1^{(c)}) \right]$$

Chunking in the forward pass computation [Image is taken from the Reformer paper]