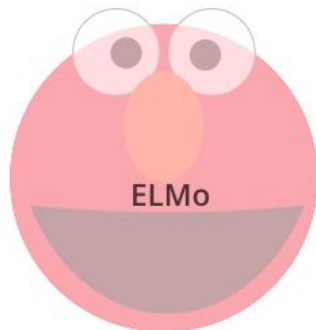


Lecture 05: Context based models & BERT overview

Radoslav Neychev

1. OpenAI Transformer
2. ELMo
3. BERT
4. GPT-2 & GPT-3
5. Q & A

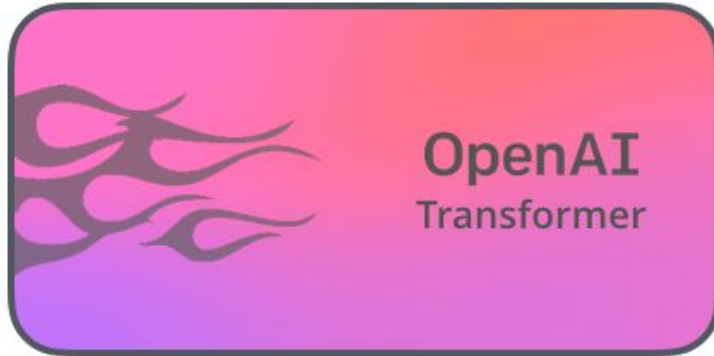
Based on: <http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture13-contextual-representations.pdf>
<https://jalammar.github.io/illustrated-transformer/>
<http://jalammar.github.io/illustrated-bert/>
<https://medium.com/mlreview/understanding-building-blocks-of-ulmfit-818d3775325b>



OpenAI Transformer: Pre-training Decoder for Language Modeling

OpenAI Transformer

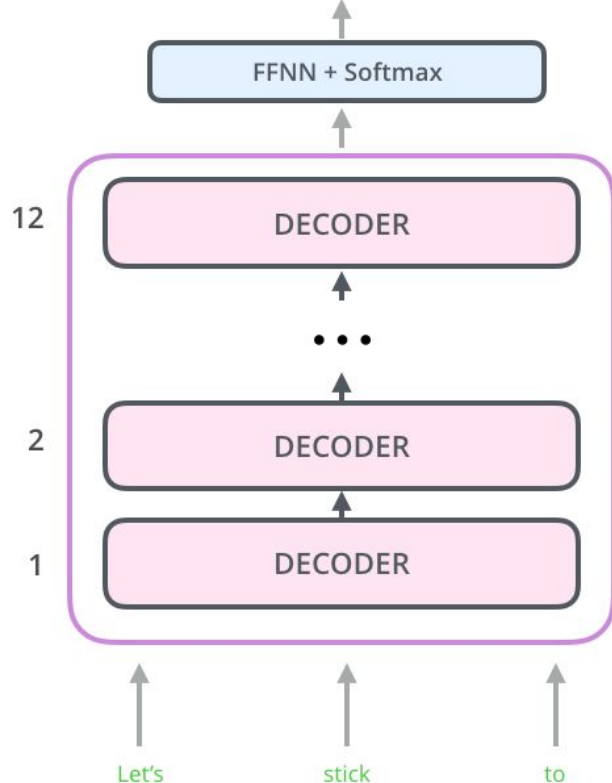
- The Encoder-Decoder structure of the transformer made it perfect for machine translation
- But what about sentence classification?
- **Main goal: pre-train a language model that can be fine-tuned for other tasks**



OpenAI Transformer

Possible classes:
All English words

0.1%	Aardvark
...	...
10%	Improvisation
...	...
0%	Zyzzzyva

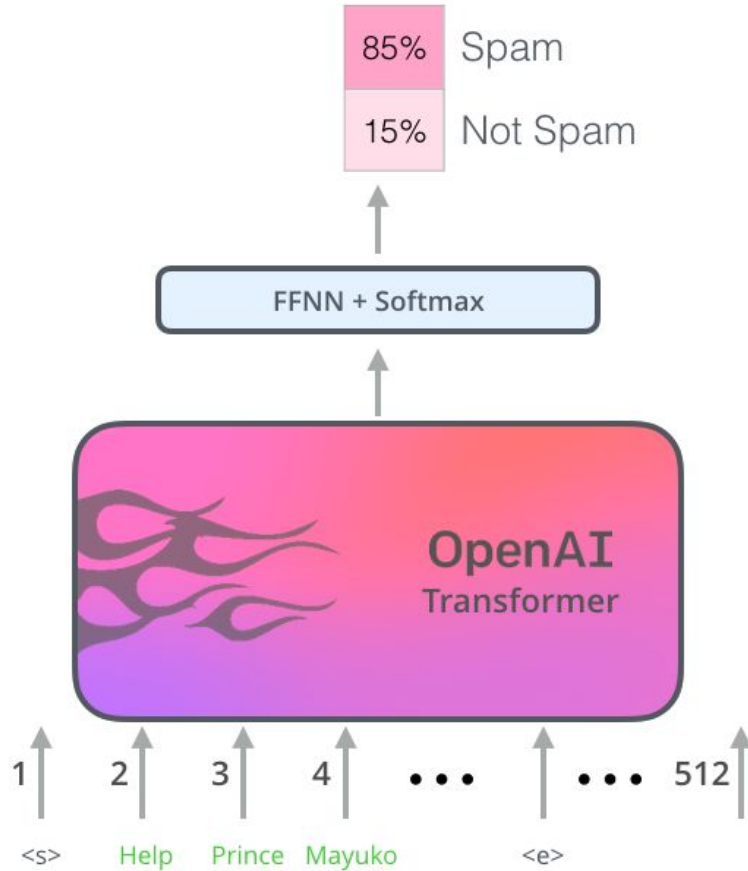


Differences from vanilla Transformer:

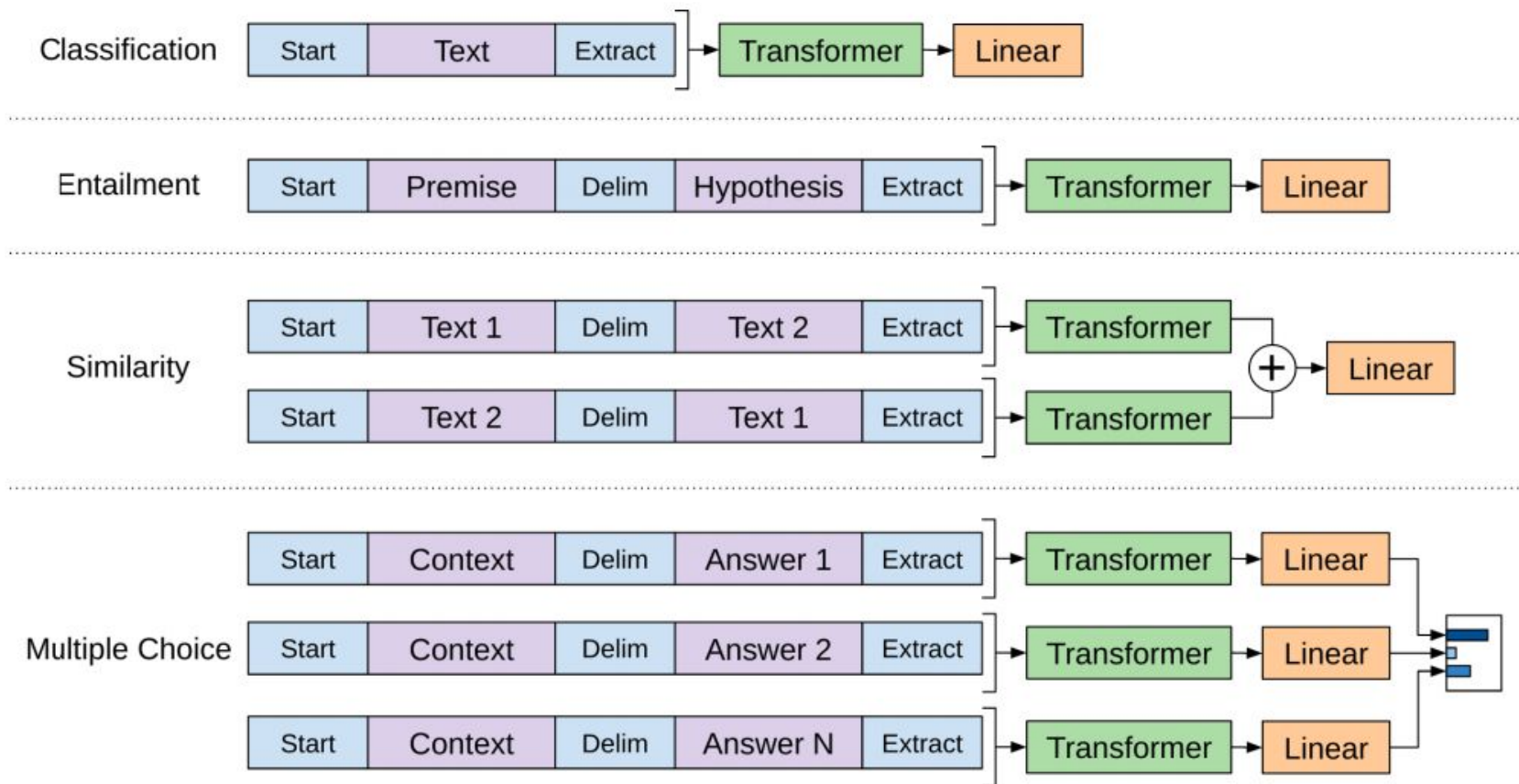
- no encoder
- decoder layers would not have the encoder-decoder attention sublayer
- Pre-train the model on predicting the next word using massive (unlabeled) datasets

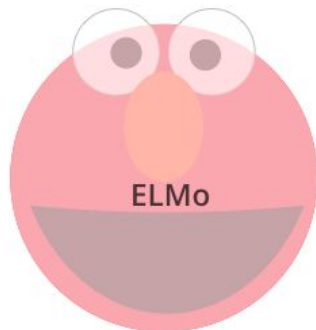
OpenAI Transformer

- 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](#),
[Peters et. al., 2018 in the ELMo paper](#)

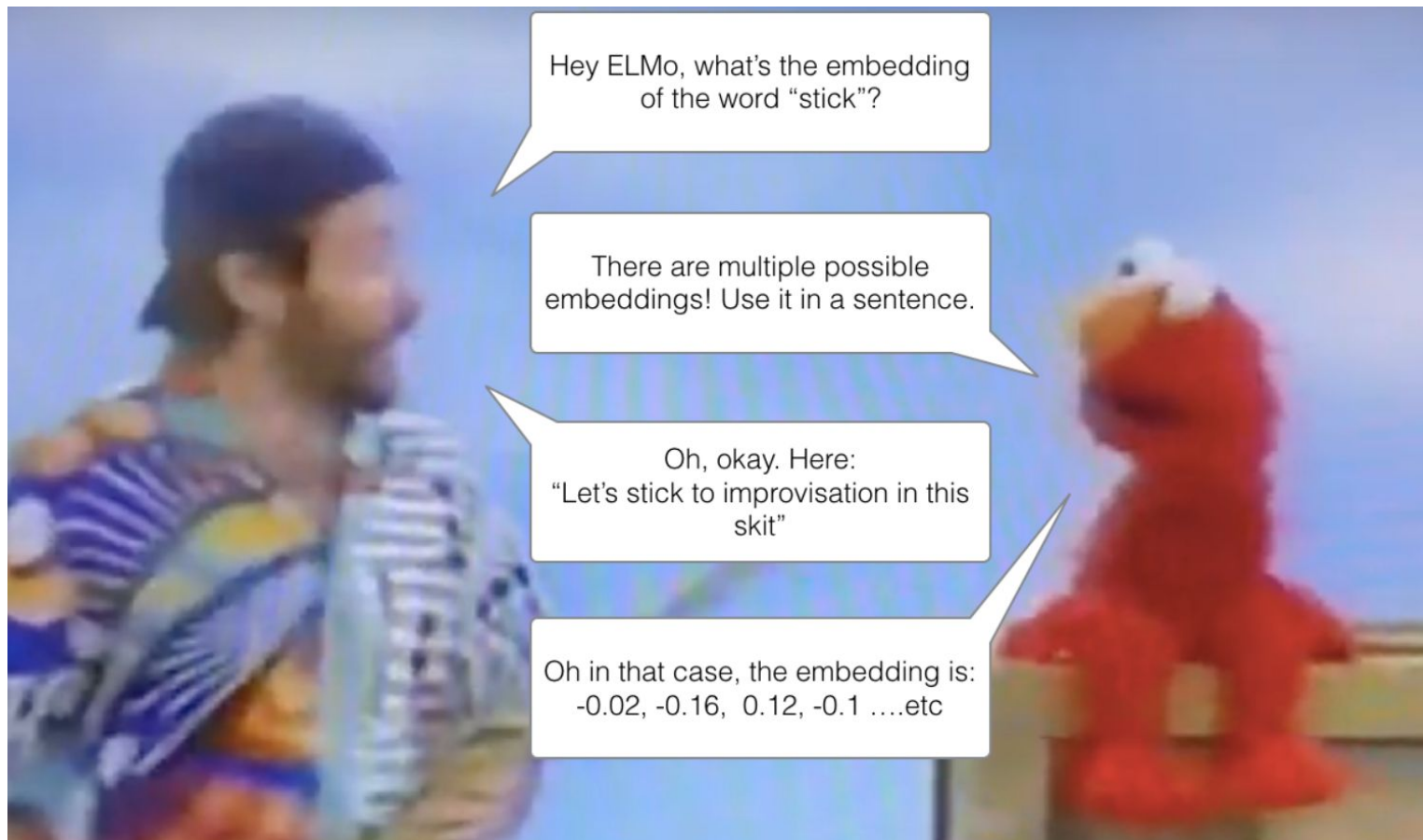
ELMo – deep contextualized
word representations

What does it stand for?



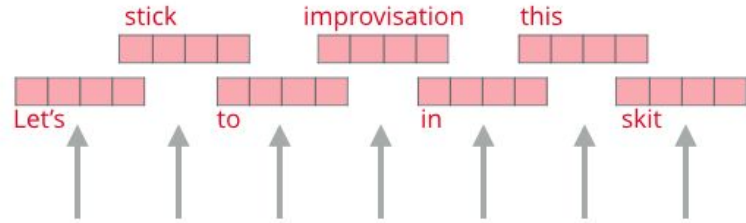
1. **E**xpedited **L**abour **M**arket **O**pinion
2. **E**lectric **L**ight **M**achine **O**rganization
3. **E**nough **L**et's **M**ove **O**n
4. **E**MBEDDINGS FROM LANGUAGE MODELS

ELMo: contextualized word embeddings

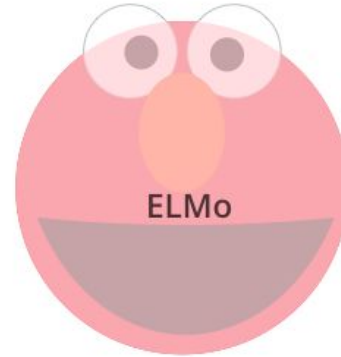


ELMo: Contextualized word embeddings

ELMo
Embeddings



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



Words to embed



Bidirectional Language Models (biLMs)

Possible classes:
All English words

0.1%	Aardvark
...	...
10%	Improvisation
...	...
0%	Zyzzzyva

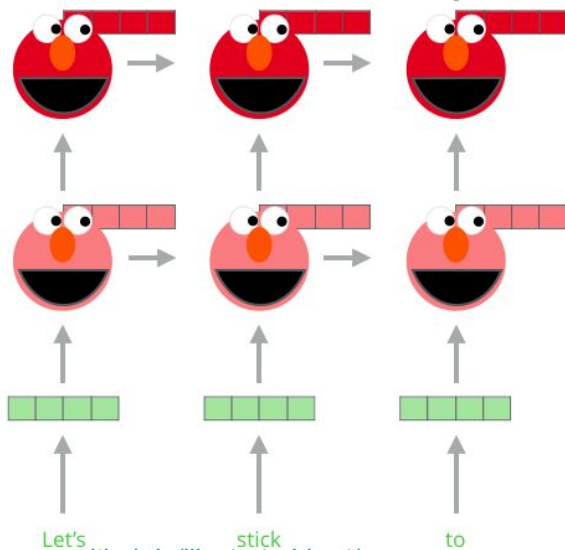
FFNN + Softmax

Output Layer

LSTM Layer #2

LSTM Layer #1

Embedding



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:

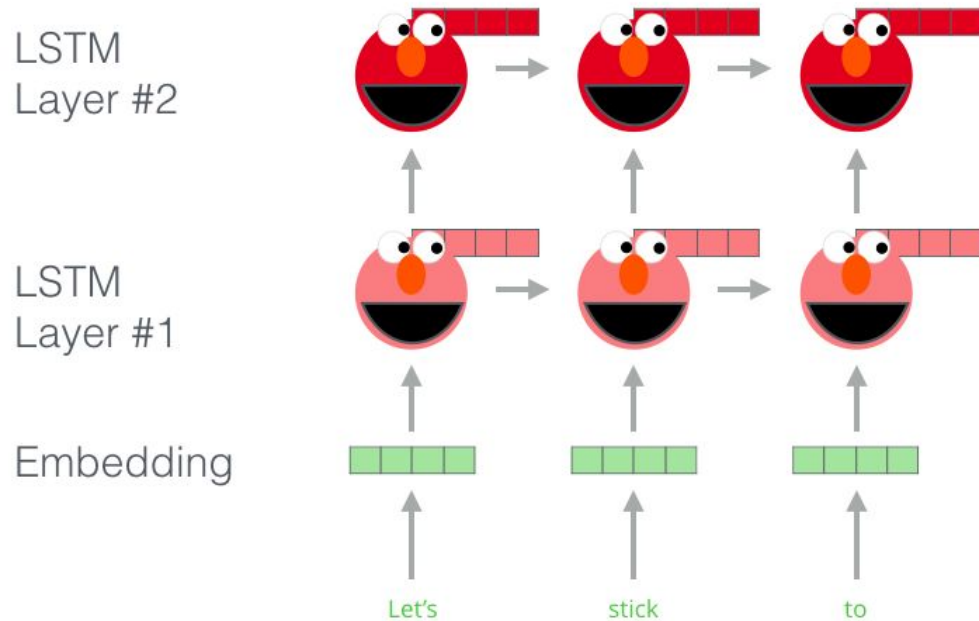
$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_{k+1}, t_{k+2}, \dots, t_N)$$

LSTM predicts next word in both directions to build biLMs

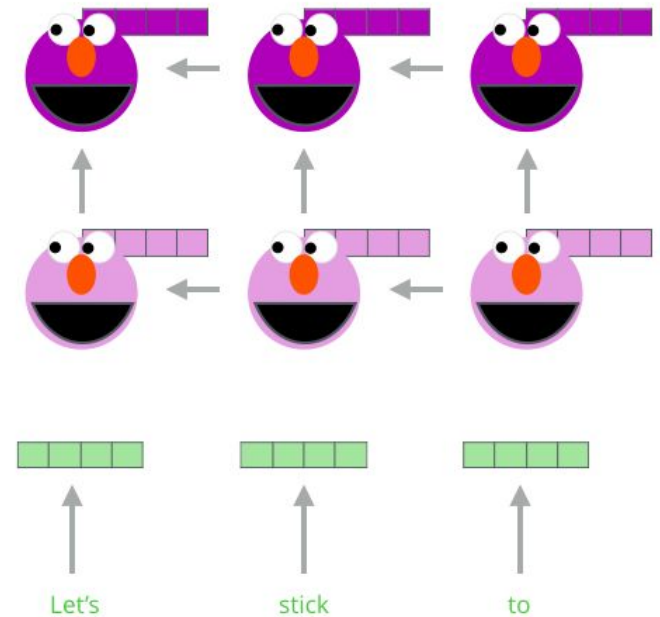
ELMo: main pipeline

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

Forward Language Model



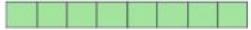
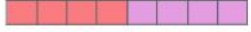
Backward Language Model



ELMo: main pipeline

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

1- Concatenate hidden layers



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

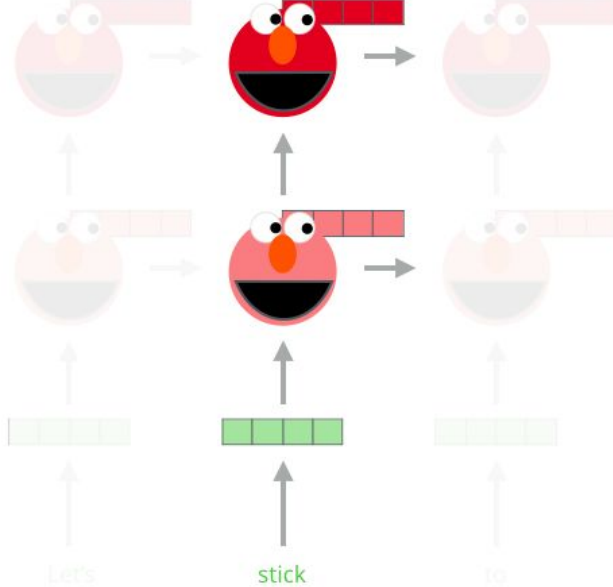


3- Sum the (now weighted) vectors

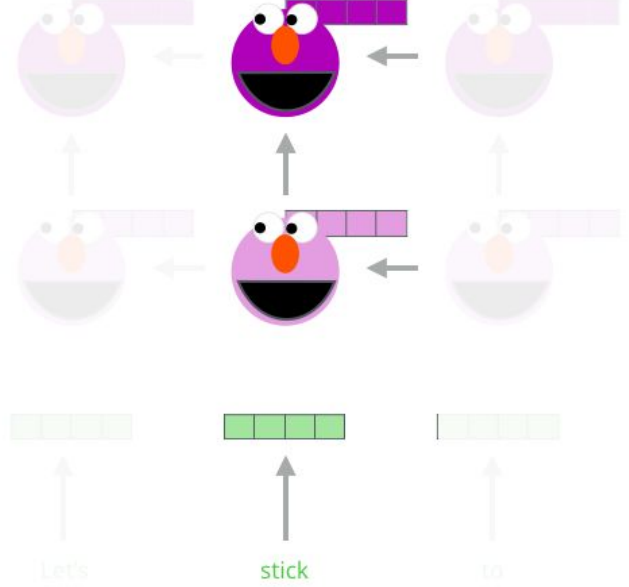


ELMo embedding of “stick” for this task in this context

Forward Language Model

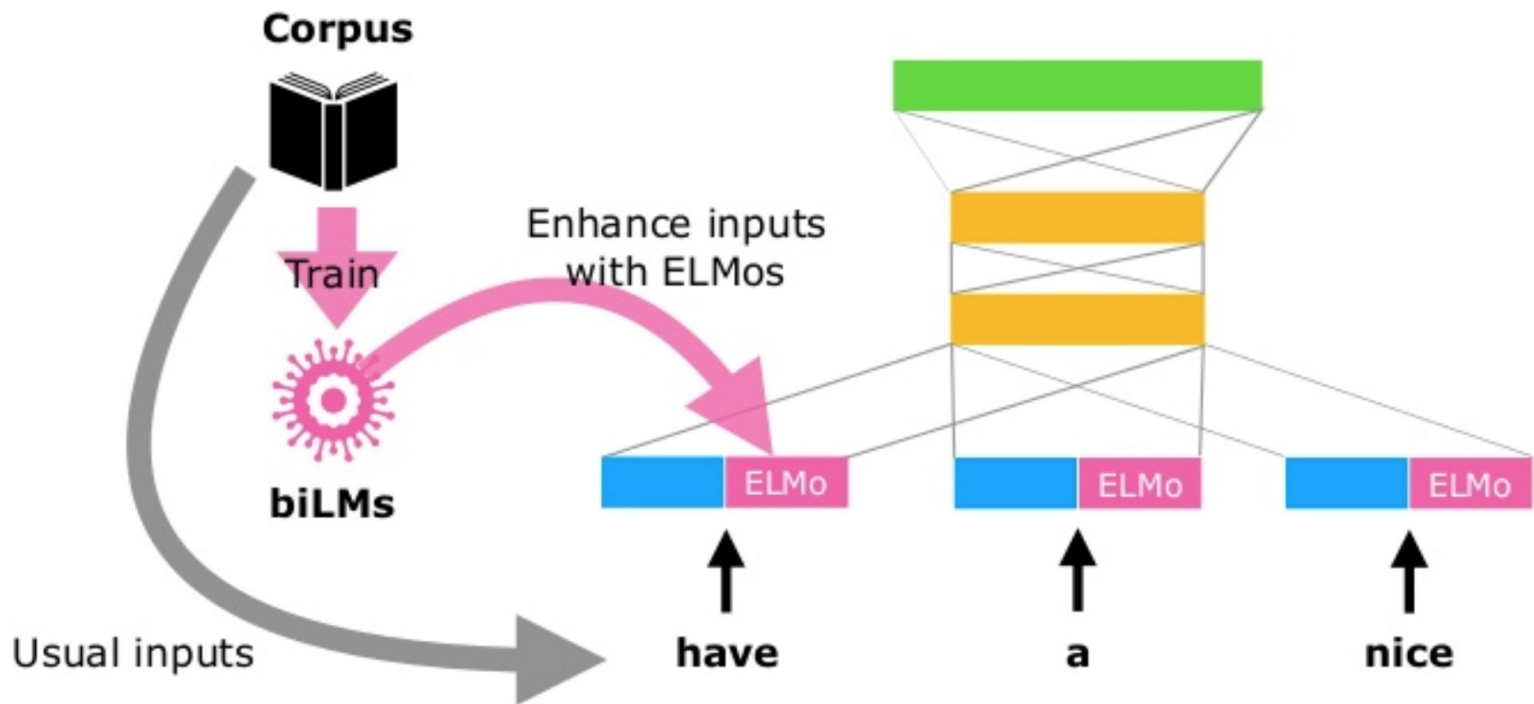


Backward Language Model

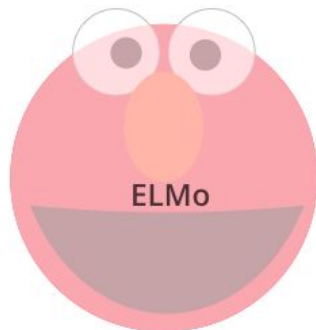
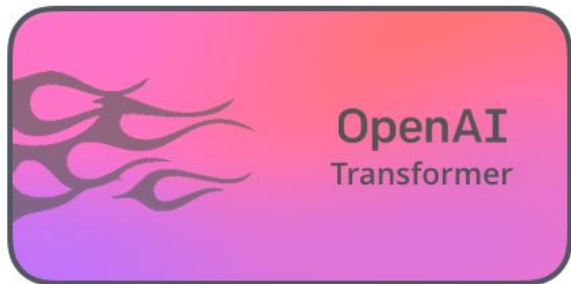


ELMo represents a word as a linear combination of corresponding hidden layers

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



- 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

1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

Semi-supervised Learning Step

Model:



Dataset:



Objective:

Predict the masked word
(language modeling)

2 - Supervised training on a specific task with a labeled dataset.

Supervised Learning Step

Model:
(pre-trained
in step #1)



Classifier

75% Spam
25% Not Spam

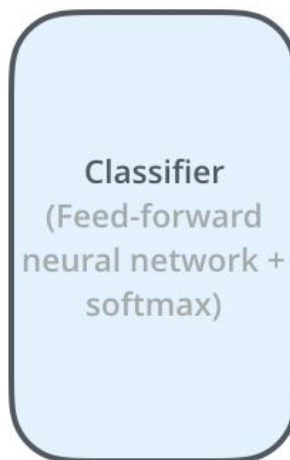
Dataset:

Email message	Class
Buy these pills	Spam
Win cash prizes	Spam
Dear Mr. Atreides, please find attached...	Not Spam

BERT

Input
Features

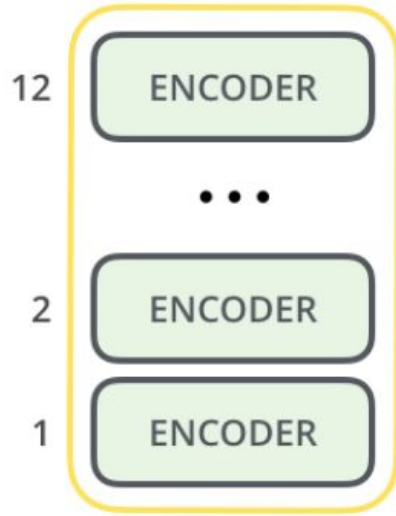
Help Prince Mayuko Transfer
Huge Inheritance



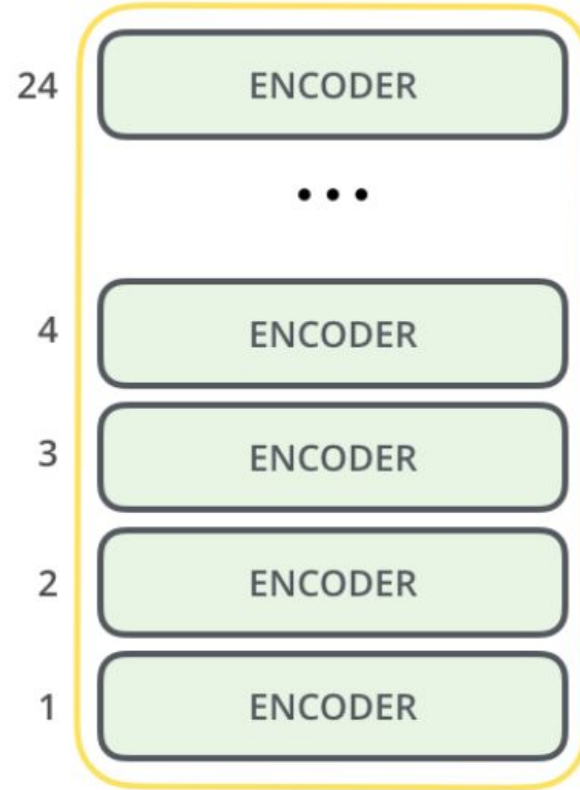
Output
Prediction



BERT: base and large





BERT_{BASE}

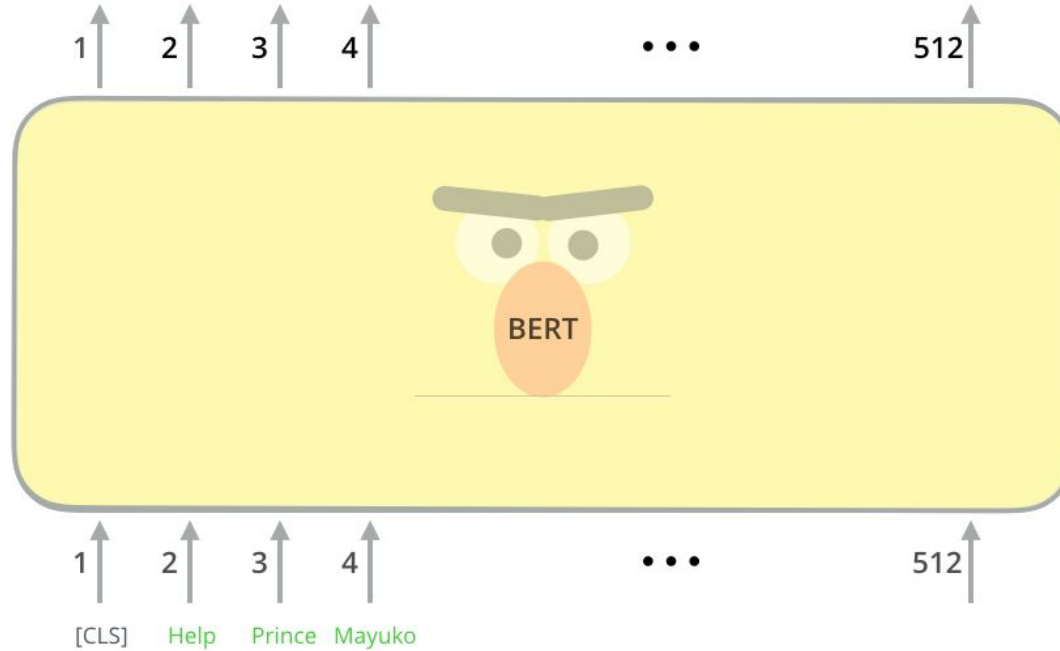


BERT_{LARGE}

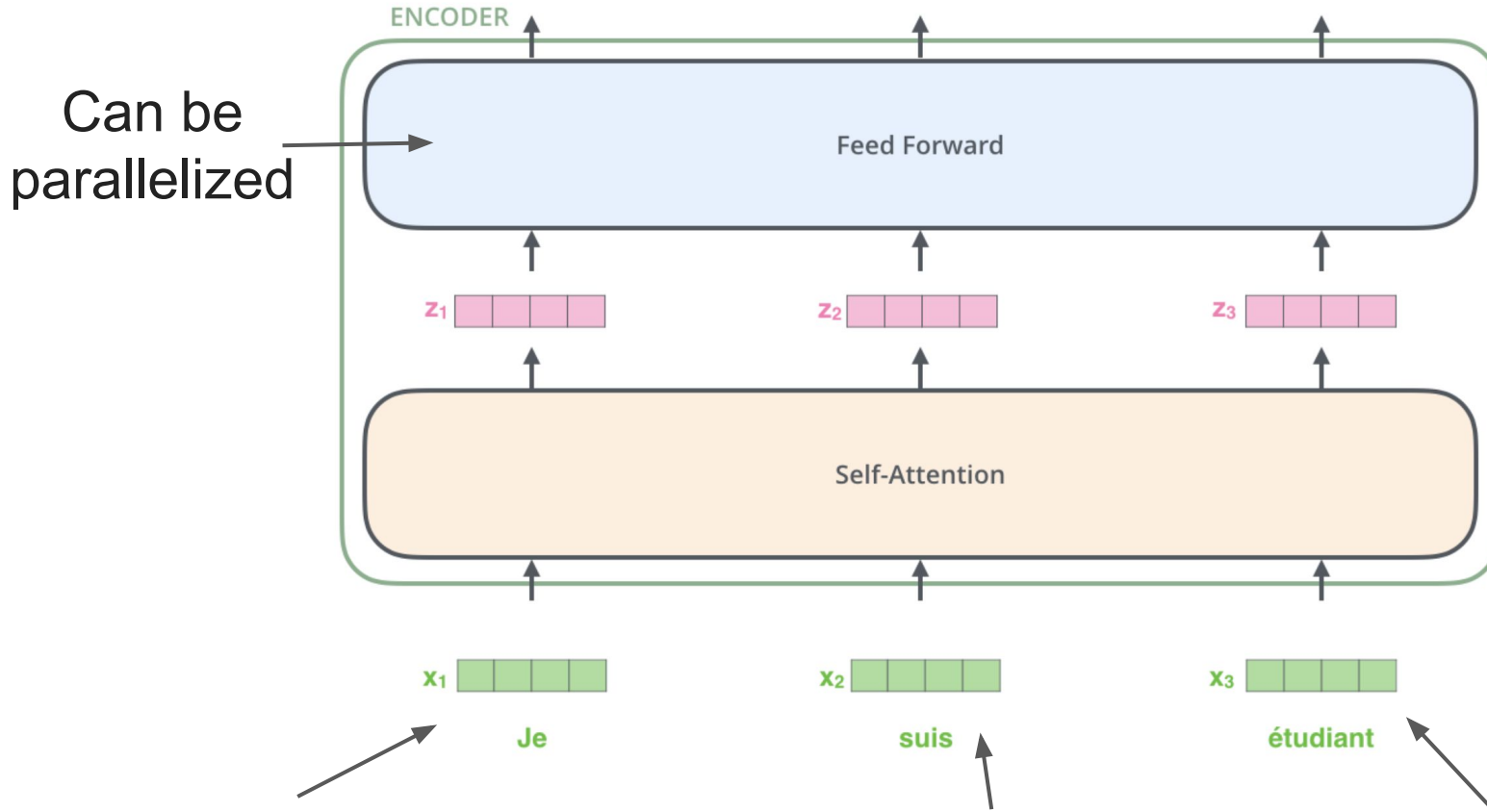
BERT vs. Transformer

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

Model inputs

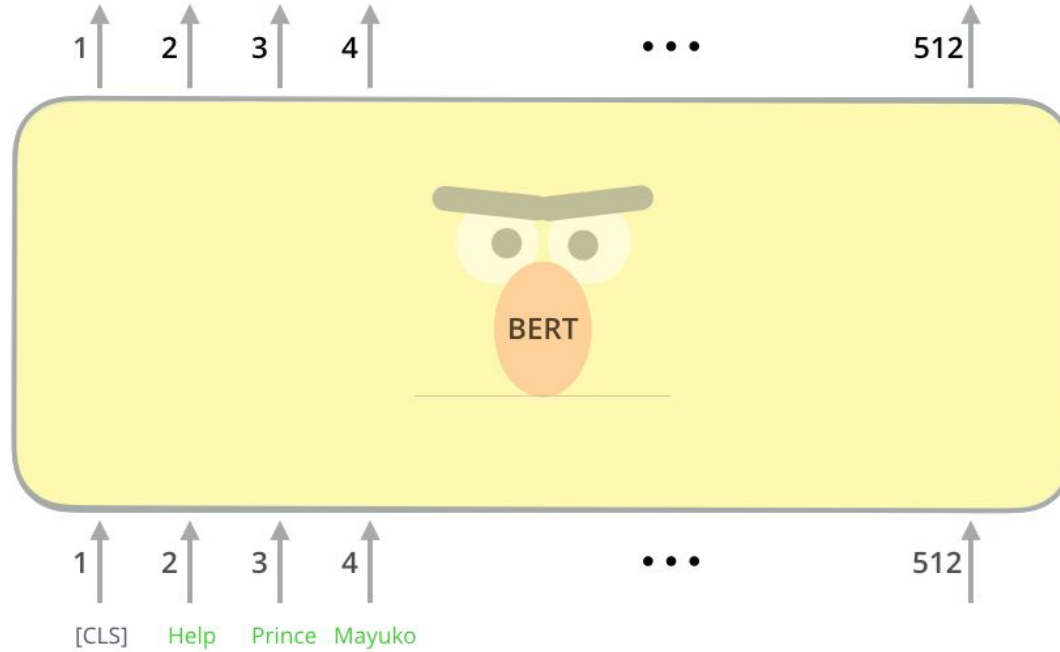


Transformer Block in BERT



the word in each position flows through its own path in the encoder

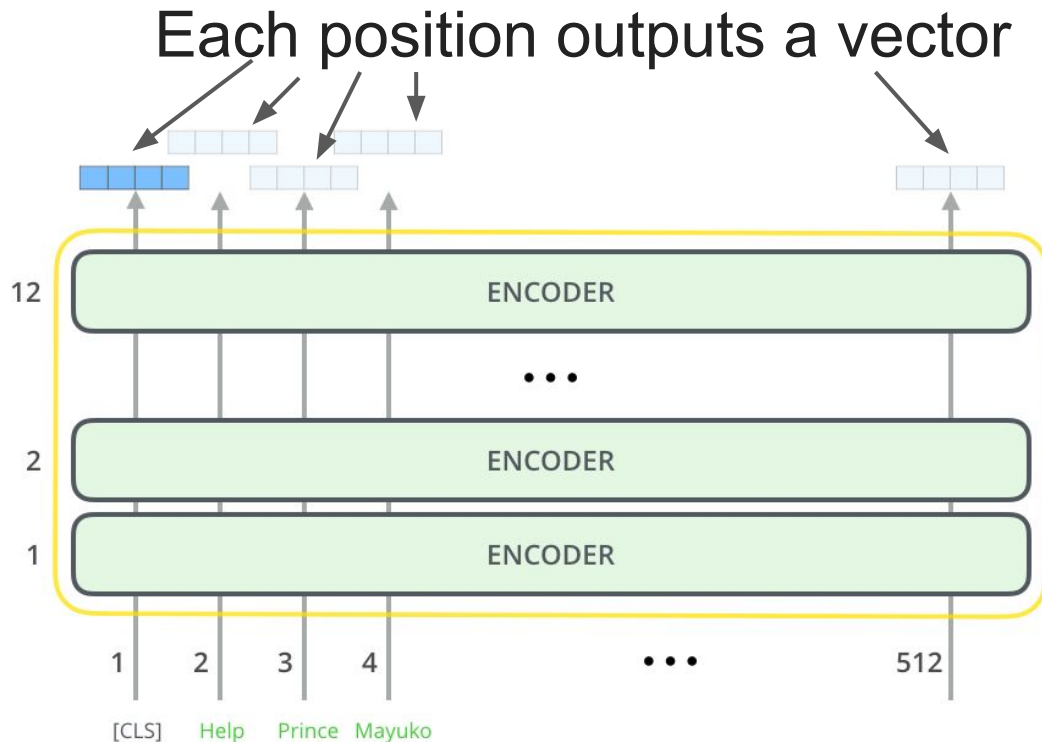
Model inputs



Identical to the Transformer up until this point

Why is BERT so special?

Model outputs

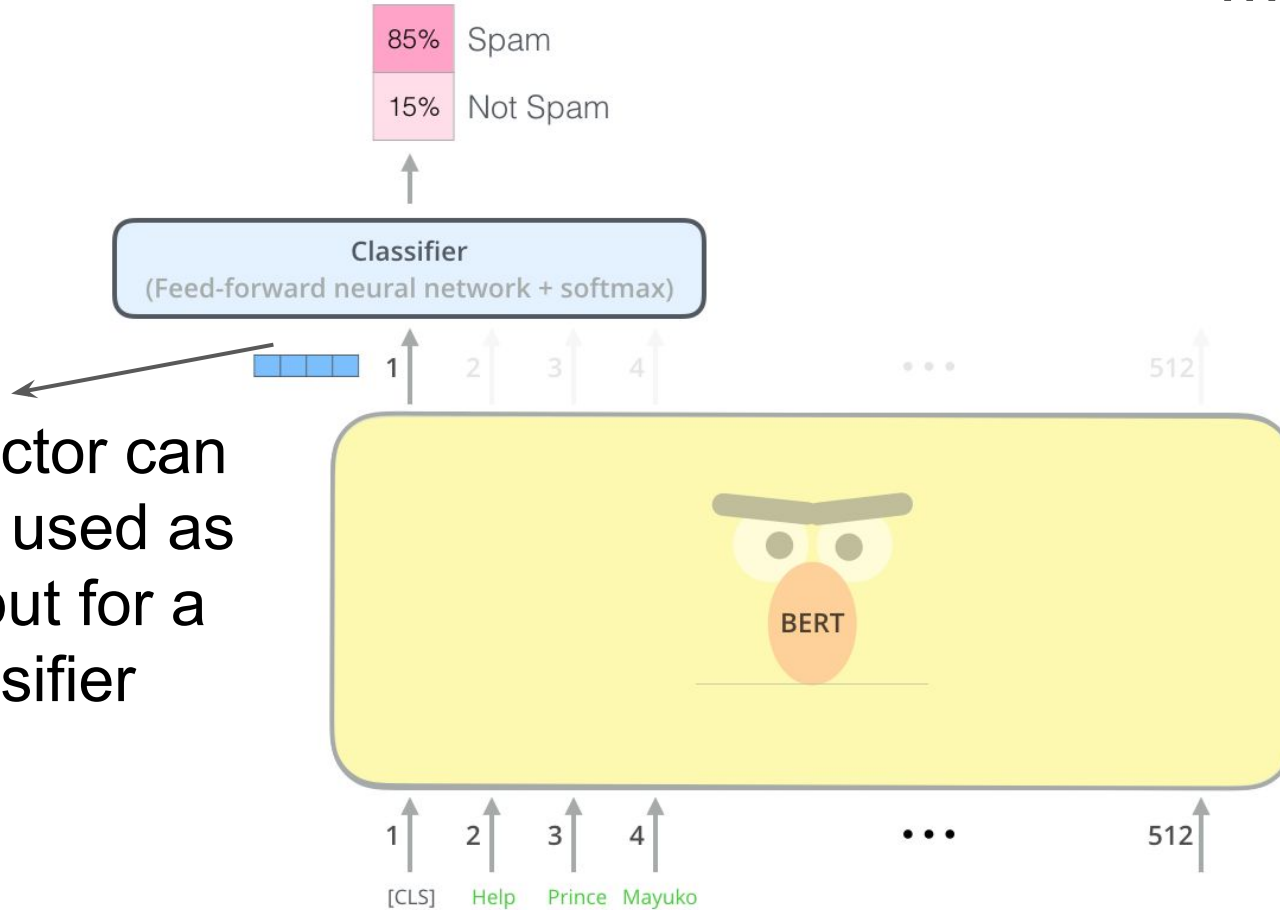


BERT

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

Model inputs

This vector can now be used as the input for a classifier

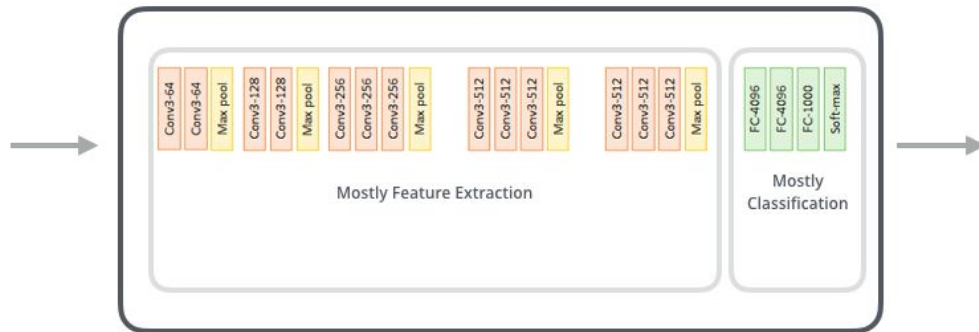


Similar to CNN concept!

Input
Features



VGG-16



Output
Prediction

0.2%	Kit fox
0.1%	English setter
95%	Egyptian cat
1%	Great Dane
...	...
0%	Hotdog

BERT: pre-training

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

Possible classes:
All English words

0.1%	Aardvark
...	...
10%	Improvisation
...	...
0%	Zyzzzyva

FFNN + Softmax

1 2 3 4 5 6 7 8 ... 512



Randomly mask
15% of tokens

1 2 3 4 5 6 7 8 ... 512
[CLS] Let's stick to [MASK] in this skit

Input

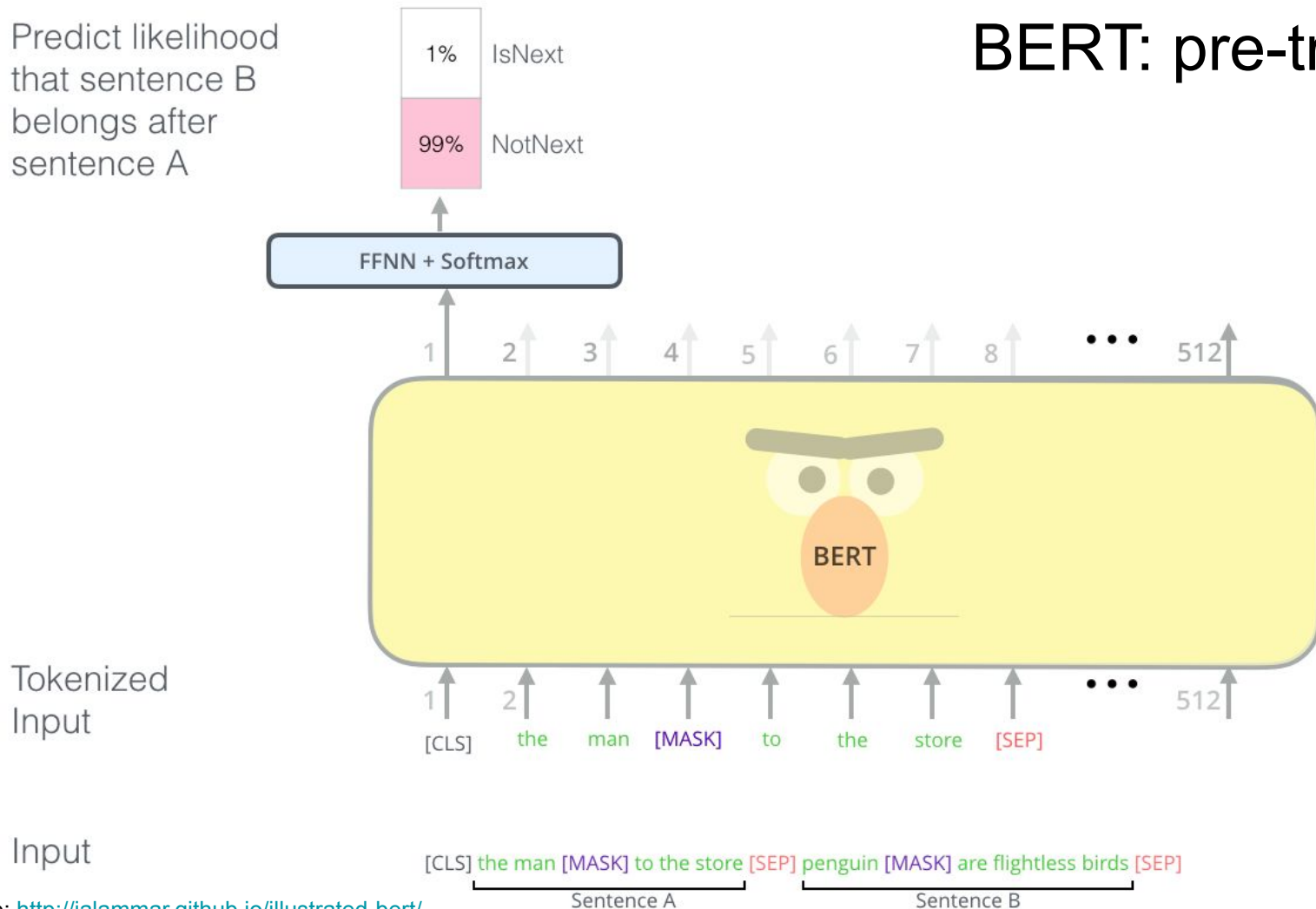
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑
[CLS] Let's stick to improvisation in this skit

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

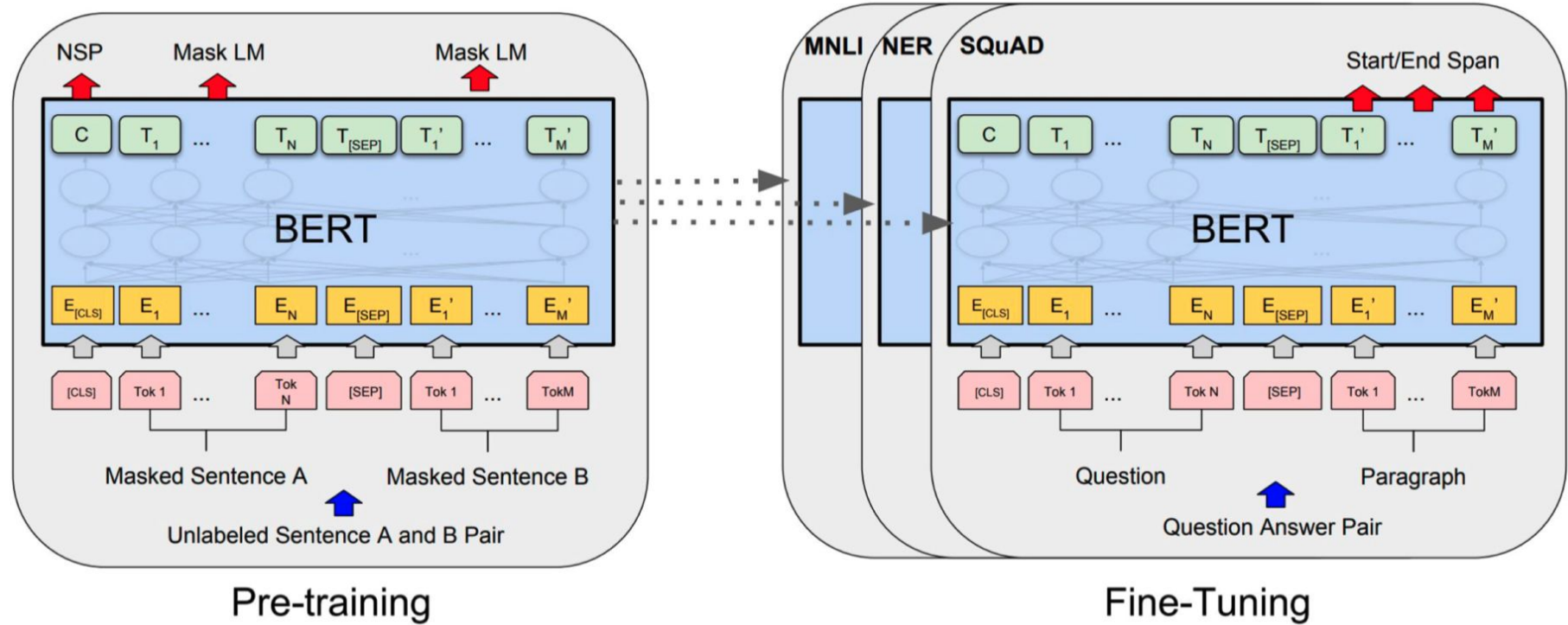


BERT: input data format

For each tokenized input sentence, we need to create:

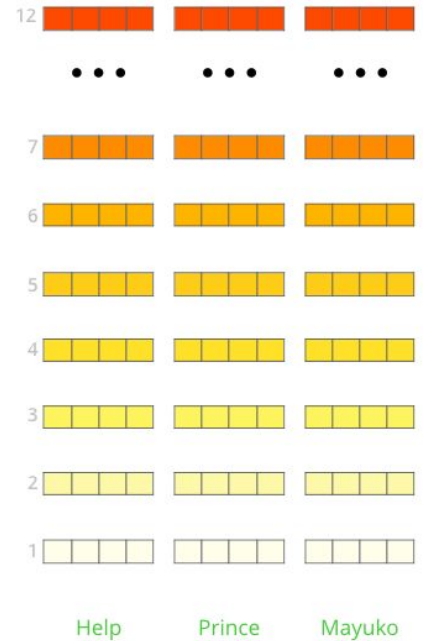
- **input ids**: a sequence of integers identifying each input token to its index number in the BERT tokenizer vocabulary
- **segment mask**: a sequence of 1s and 0s used to identify whether the input is one sentence or two sentences long. For one sentence inputs, this is simply a sequence of 0s. For two sentence inputs, there is a 0 for each token of the first sentence, followed by a 1 for each token of the second sentence
- **attention mask**: a sequence of 1s and 0s, with 1s for all input tokens and 0s for all padding tokens

BERT: fine-tuning for different tasks



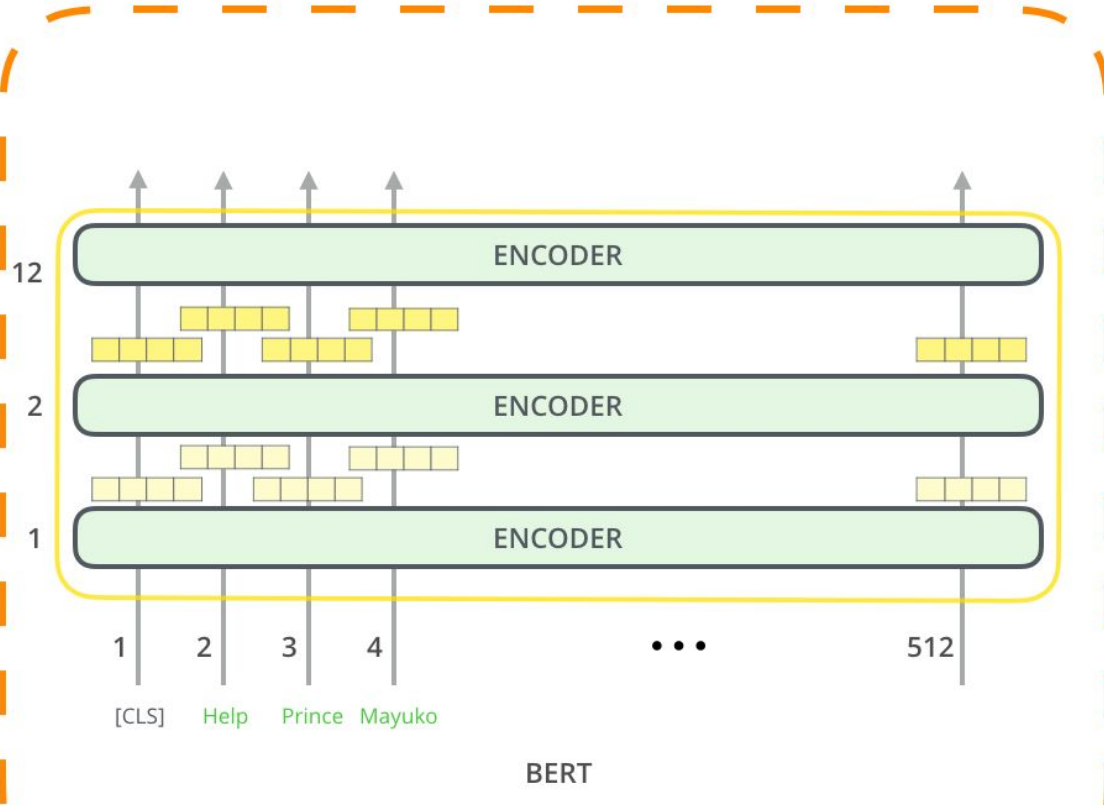
BERT for feature extraction

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?

Generate Contextualized Embeddings



BERT

BERT for feature extraction

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

For named-entity recognition task CoNLL-2003 NER

12



...

7



6



5



4



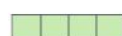
3



2



1



Help

First Layer

Embedding



91.0

Last Hidden Layer


12



94.9

Sum All 12
Layers

12




+

...

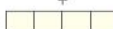
+

2




+

1



=



95.5

Second-to-Last
Hidden Layer


11



95.6


Sum Last Four
Hidden

12



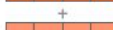
+

11



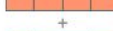
+

10




+

9




=



95.9

Concat Last
Four Hidden

9 10 11 12

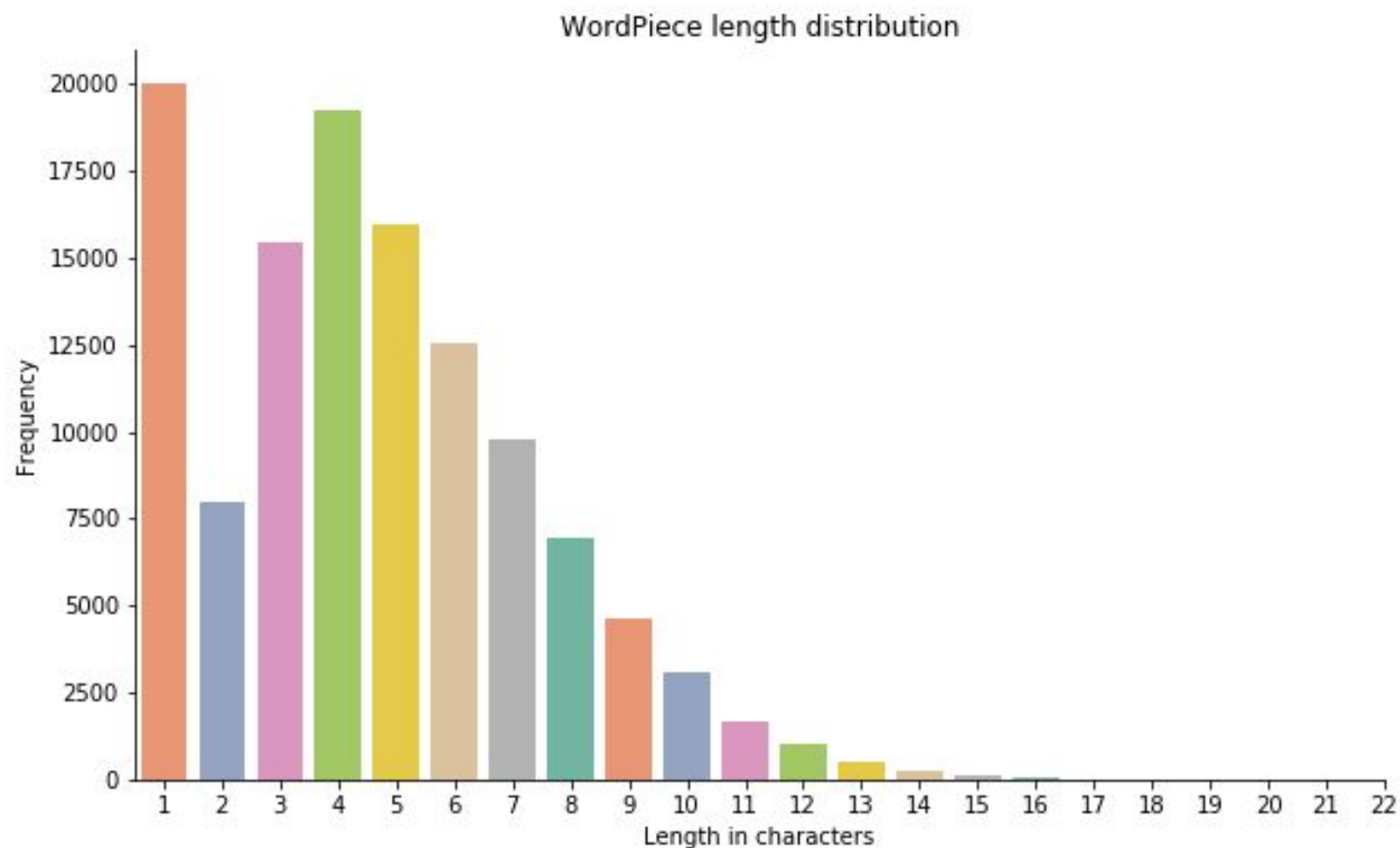


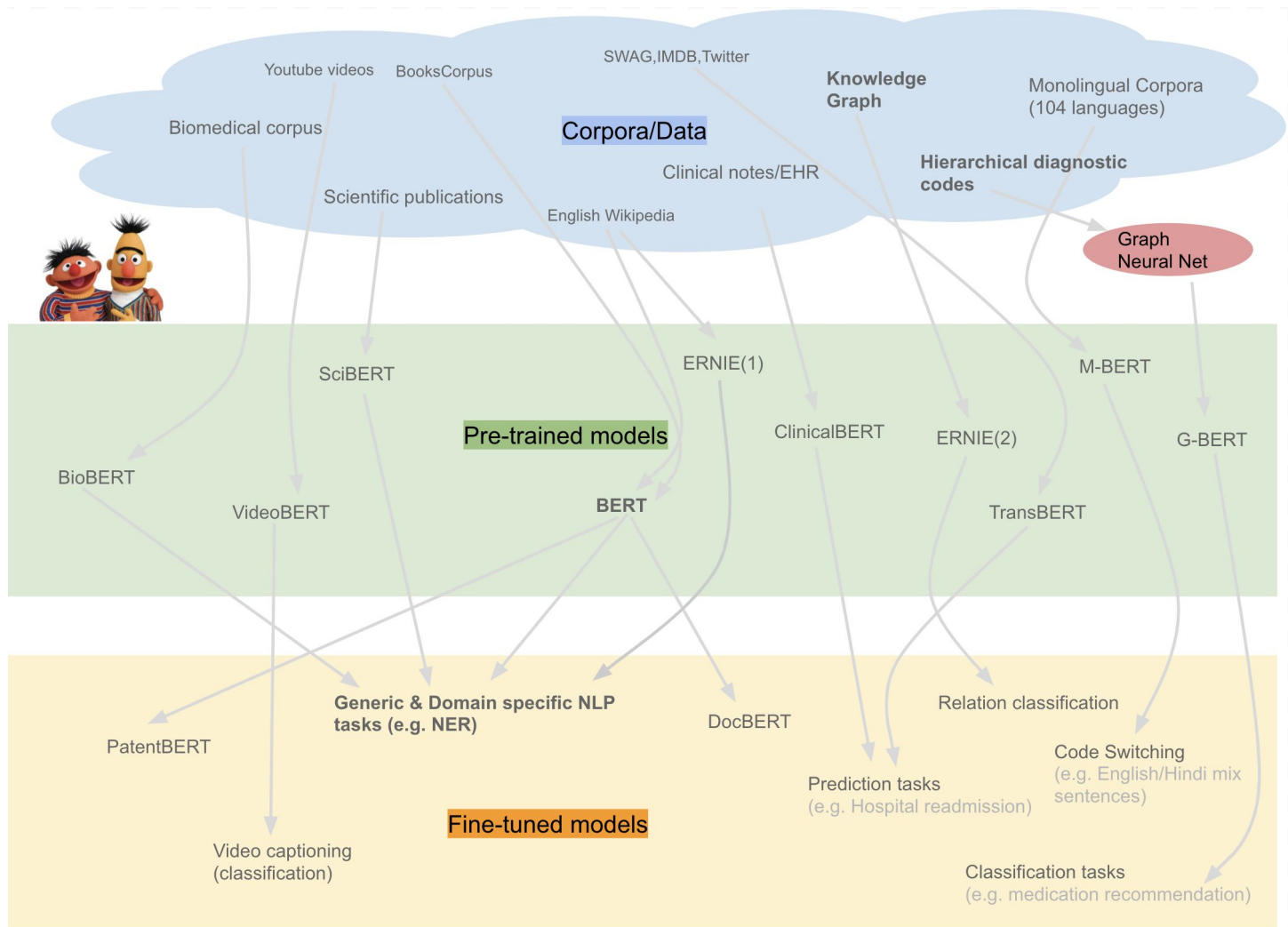
96.1

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 multi character word pieces of various length.

BERT: tokenization





BERT: overview

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

GPT-2 & GPT-3

- 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.

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



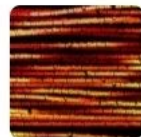
When Is Technology Too Dangerous to Release to the Public?

Slate · 2 days ago



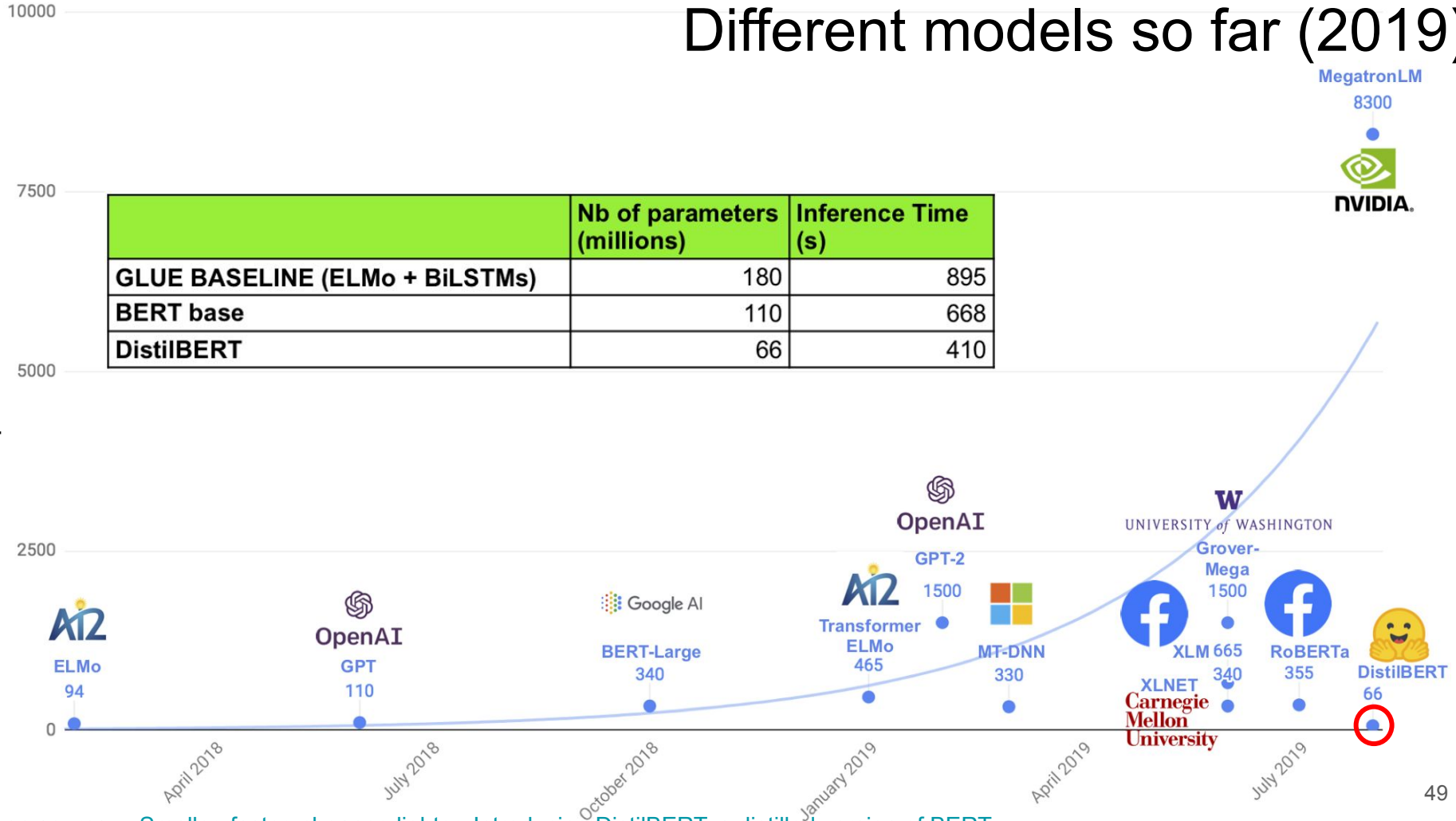
Scientists Developed an AI So Advanced They Say It's Too Dangerous to Release

ScienceAlert · 6 days ago



Different models so far (2019)

number of parameters, millions

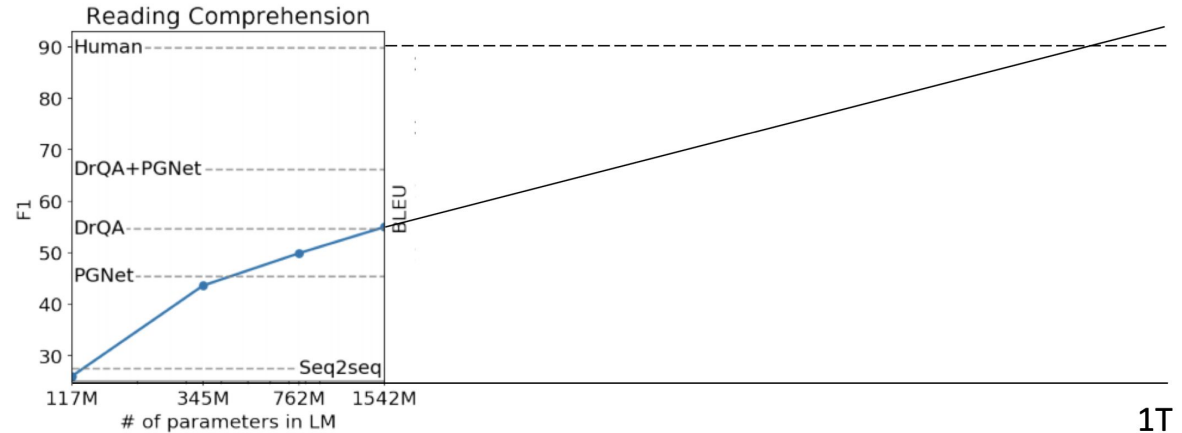
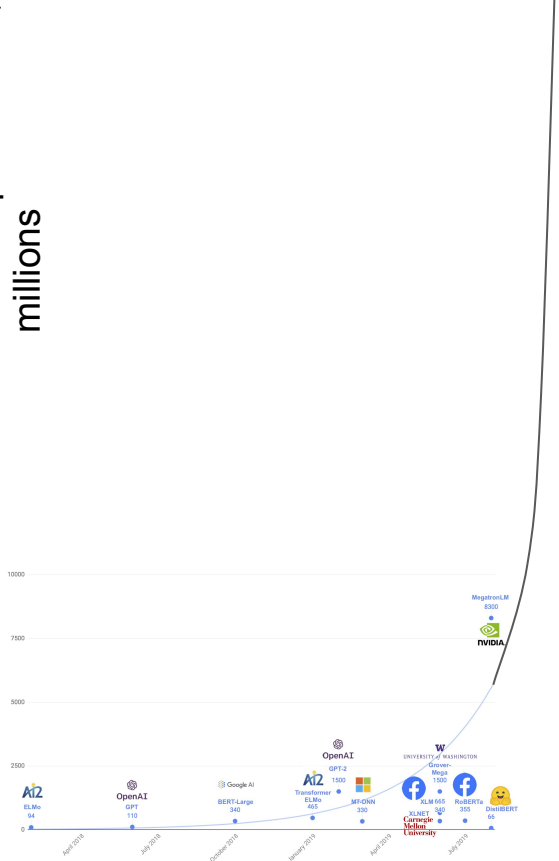


Latest achievements: GPT-3

GPT-3, May 2020

Proportions are not preserved for visual sake

Number of trainable parameters,
millions



Hypothesis from Stanford CS224N Lecture 20 (2019)

- 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



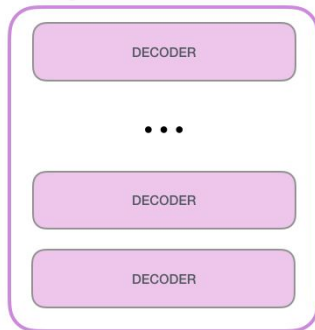
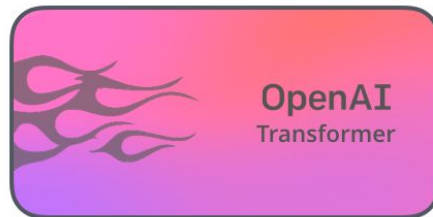
643



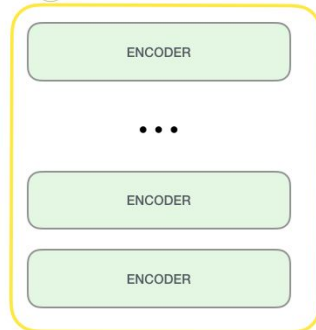
3.4K



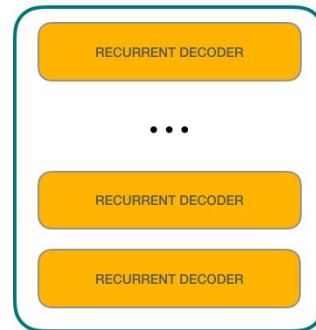
- Transformer
- OpenAI Transformer
- ELMO
- BERT
- BERTology
- GPT
- GPT-2
- GPT-3



BERT



TRANSFORMER XL



- Transfer learning caused a giant leap in Computer Vision and Natural Language Processing
 - Which domain is next?
- Using pre-trained BERT might be a good idea in many tasks
 - Or even using DistillBERT
- Better problem statement leads to better results
 - What information is hidden within the data and can be retrieved?