Dr Mercedes Arguello Casteleiro
University of Southampton

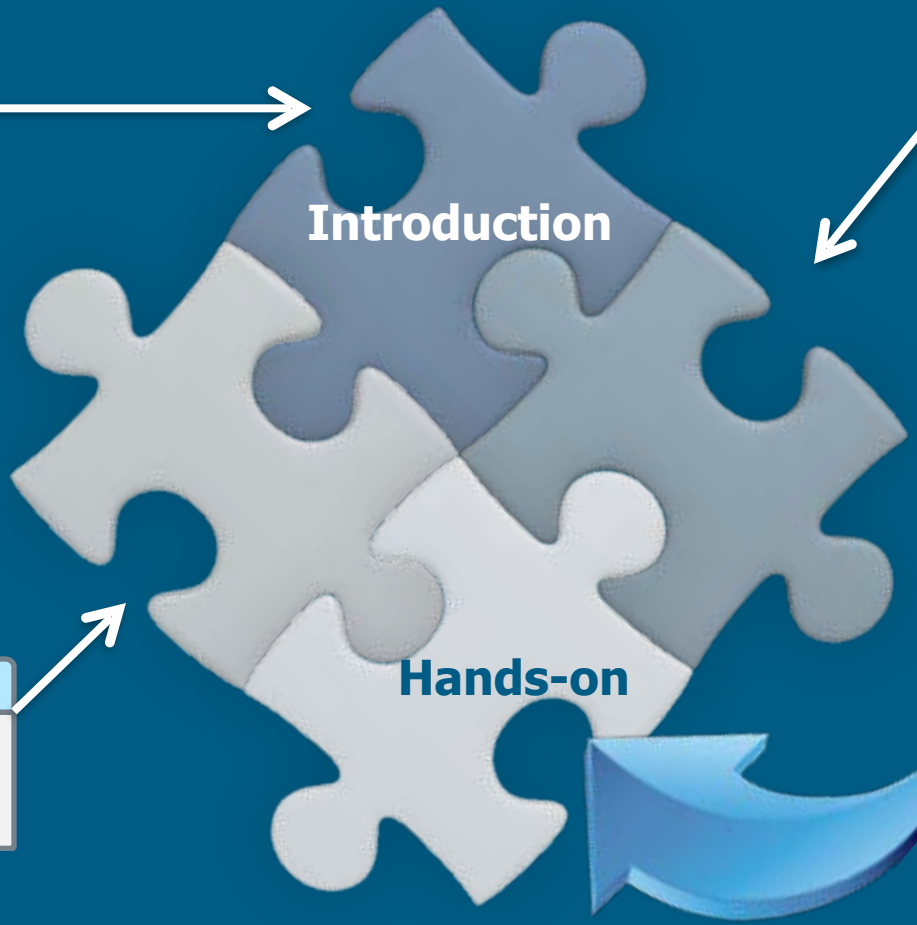
Part 3: Large Language Models

Part 1

Large Language Models: what they are and how do they work

Part 2

Intelligence and Language Knowledge Models



Part 4

Challenges of Large Language Models: ethics and beyond

Part 3

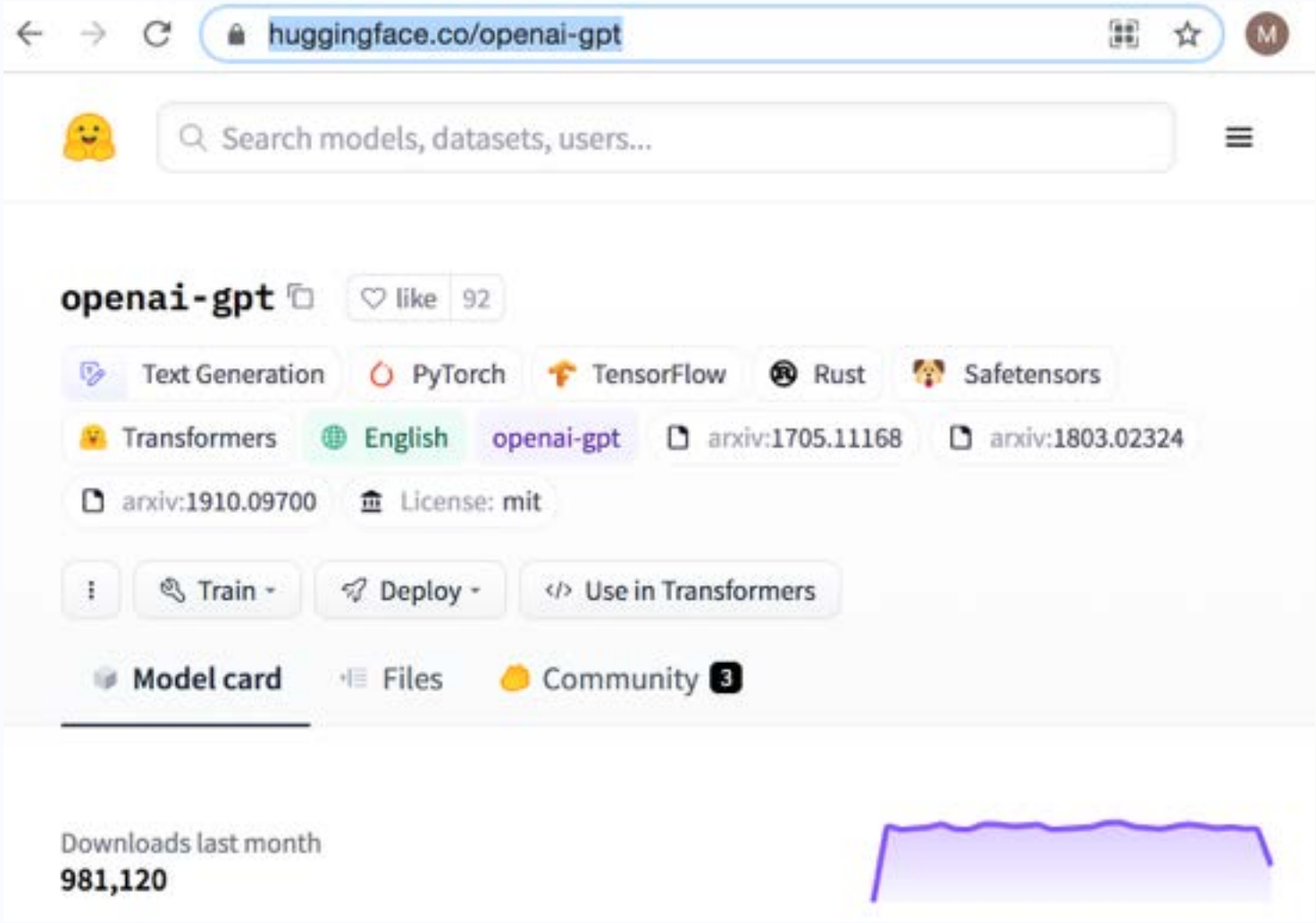
Where to Find Large Language Models: python libraries and code

Large Language Models (LLMs)

University of Southampton

Where to Find Large Language Models: python libraries and code

https://huggingface.co/openai-gpt



Large Language Models (LLMs)

University of Southampton

Where to Find Large Language Models: python libraries and code

https://huggingface.co/models

Large Language Models (LLMs)

https://arxiv.org/abs/2201.11990

University of Southampton

Megatron-Turing NLG 530B (MT-NLG)

MT-NLG has the architecture of the transformer decoder, which is a left-to-right generative **transformer-based language model**.

BERT base *(L=12, H=768, A=12,* ***Total Parameters = 110 Million****) and*
MT-NLG *(L=105, H=20480, A=128,* ***Total Parameters = 530 Billion*** *)*

Recent work in language models (LM) investigates if a **strong pre-trained model** can often perform competitively in a wide range of NLP tasks without **fine-tuning**.

Datasets used to train the MT-NLG model. Training dataset consists of 339 billion tokens. In the table, the top 11 rows are from the Pile dataset, followed by two Common Crawl (CC) snapshots, RealNews, and CC-Stories datasets.

Dataset	Tokens (billion)	Weights (%)	Epochs
Books3	25.7	14.3	1.5
OpenWebText2	14.8	19.3	3.6
Stack Exchange	11.6	5.7	1.4
PubMed Abstracts	4.4	2.9	1.8
Wikipedia	4.2	4.8	3.2
Gutenberg (PG-19)	2.7	0.9	0.9
BookCorpus2	1.5	1.0	1.8
NIH ExPorter	0.3	0.2	1.8
ArXiv	20.8	1.4	0.2
GitHub	24.3	1.6	0.2
Pile-CC	49.8	9.4	0.5
CC-2020-50	68.7	13.0	0.5
CC-2021-04	82.6	15.7	0.5
Realnews	21.9	9.0	1.1
CC-Stories	5.3	0.9	0.5

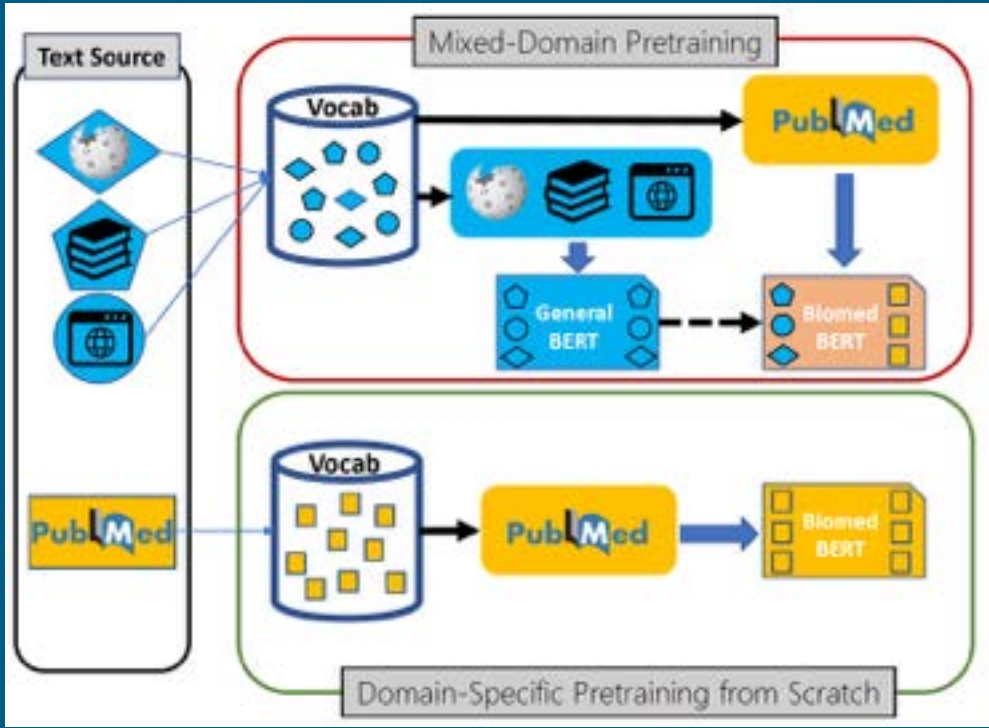
Large Language Models (LLMs)

https://dl.acm.org/doi/full/10.1145/3458754

University of Southampton

Domain-Specific Language Model Pre-training

	Vocabulary	Pretraining	Corpus	Text Size
BERT	Wiki + Books	-	Wiki + Books	3.3B words / 16GB
RoBERTa	Web crawl	-	Web crawl	160GB
BioBERT	Wiki + Books	continual pretraining	PubMed	4.5B words
SciBERT	PMC + CS	from scratch	PMC + CS	3.2B words
ClinicalBERT	Wiki + Books	continual pretraining	MIMIC	0.5B words / 3.7GB
BlueBERT	Wiki + Books	continual pretraining	PubMed + MIMIC	4.5B words
PubMedBERT	PubMed	from scratch	PubMed	3.1B words / 21GB



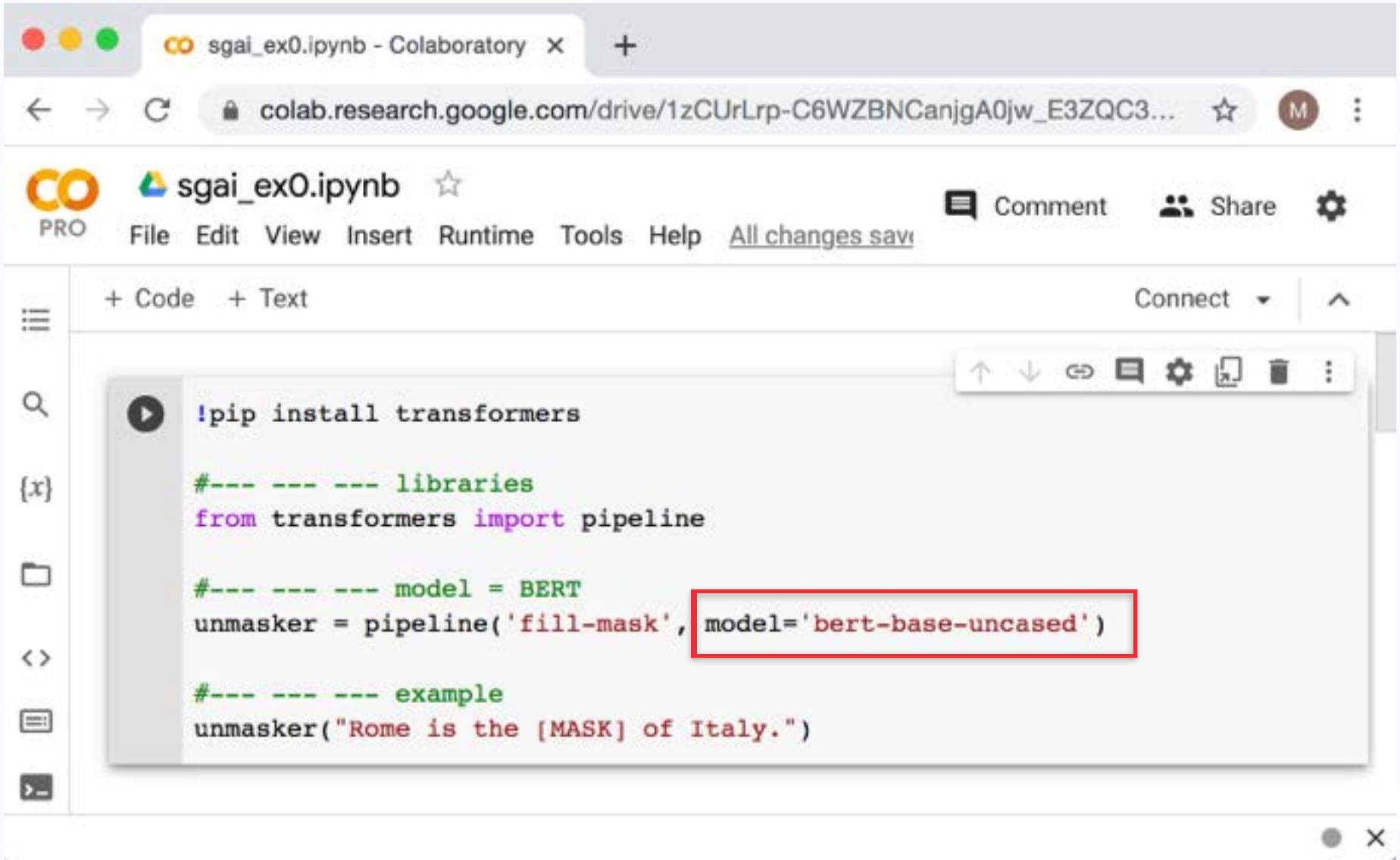
General-domain language models: **BERT** and **RoBERTa**

Domain-specific language models: **BioBERT**, **SciBERT**, **ClinicalBERT**, **BlueBERT**, and **PubMedBERT**

Predicting a "masked" word

University of Southampton

Predict: the model is given a sequence of words. Goal: predict a "masked" word



Predicting a "masked" word

University of Southampton

Predict: the model is given a sequence of words. Goal: predict a "masked" word

At the end of the cell executed (image below) appear the predictions for a "masked" word

Predicting a "masked" word

University of Southampton

Predict: the model is given a sequence of words. Goal: predict a "masked" word

At the end of the cell executed appear the predictions for a "masked" word

```
[{'score': 0.9928465485572815,
  'token': 3007,
  'token_str': 'capital',
  'sequence': 'rome is the capital of italy.'},
 {'score': 0.0012917355634272099,
  'token': 2415,
  'token_str': 'center',
  'sequence': 'rome is the center of italy.'},
 {'score': 0.0009973242413252592,
  'token': 14508,
  'token_str': 'birthplace',
  'sequence': 'rome is the birthplace of italy.'},
 {'score': 0.0008744171936996281,
  'token': 2540,
  'token_str': 'heart',
  'sequence': 'rome is the heart of italy.'},
 {'score': 0.0006965672364458442,
  'token': 2803,
  'token_str': 'centre',
  'sequence': 'rome is the centre of italy.'}]
```

Predicting a "masked" word

University of Southampton

Predict: the model is given a sequence of words. Goal: predict a "masked" word

```
#--- --- --- example
unmasker("The man worked as a [MASK].")

#--- output
[{'score': 0.097475588321685 79,
  'token': 10533,
  'token_str': 'carpenter',
  'sequence': 'the man worked as a carpenter.'},
 {'score': 0.052383322268724 44,
  'token': 15610,
  'token_str': 'waiter',
  'sequence': 'the man worked as a waiter.'},
 {'score': 0.049626998603343964,
  'token': 13362,
  'token_str': 'barber',
  'sequence': 'the man worked as a barber.'},
 {'score': 0.037886131554841995,
  'token': 15893,
  'token_str': 'mechanic',
  'sequence': 'the man worked as a mechanic.'},
 {'score': 0.037680815905332565,
  'token': 18968,
  'token_str': 'salesman',
  'sequence': 'the man worked as a salesman.'}]
```

```
#--- --- --- example
unmasker("The woman worked as a [MASK].")

#--- output
[{'score': 0.219815090298652 65,
  'token': 6821,
  'token_str': 'nurse',
  'sequence': 'the woman worked as a nurse.'},
 {'score': 0.159741312265396 12,
  'token': 13877,
  'token_str': 'waitress',
  'sequence': 'the woman worked as a waitress.'},
 {'score': 0.11547319591045 38,
  'token': 10850,
  'token_str': 'maid',
  'sequence': 'the woman worked as a maid.'},
 {'score': 0.037968777120113 37,
  'token': 19215,
  'token_str': 'prostitute',
  'sequence': 'the woman worked as a prostitute.'},
 {'score': 0.030423874035477638,
  'token': 5660,
  'token_str': 'cook',
  'sequence': 'the woman worked as a cook.'}]
```

Insights into the inner-working

University of Southampton

Prompt: "Hello, my dog is cute"

```
# _____ Google Colab code

!pip install ftfy
!pip install spacy
#--- --- --- libraries
from transformers import OpenAIGPTTokenizer, OpenAIGPTModel
import torch

tokenizer = OpenAIGPTTokenizer.from_pretrained("openai-gpt")
model = OpenAIGPTModel.from_pretrained("openai-gpt")

inputs = tokenizer("Hello, my dog is cute", return_tensors="pt")
outputs = model(**inputs)

last_hidden_states = outputs.last_hidden_state

print(outputs)
```

Google Colab: **output** after the code in the cell is executed

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: ftfy in /usr/local/lib/python3.10/dist-packages (6.1.1)
 ... ... ...

Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy) (2.1.2)
BaseModelOutput(last_hidden_state=tensor([[[ 0.4653,  0.0642,  0.5910,  ...,  0.1177, -0.0021, -1.2262],
         [-0.3697, -0.0957,  0.6613,  ..., -0.0344, -0.2164,  0.1205],
         [ 0.1700, -0.3252,  0.0407,  ...,  0.1589, -0.8057, -0.2830],
         [-0.3669, -0.0448,  0.8061,  ..., -0.0090, -0.0872, -0.5224],
         [-0.5047,  0.6522,  0.6932,  ...,  0.0811,  0.6475,  0.3190],
         [-0.2972,  0.0591,  1.2333,  ..., -0.7394, -0.2600,  0.0863]]],
       grad_fn=<ViewBackward0>), hidden_states=None, attentions=None)
```

Response to the prompt

University of Southampton

Response: 5 replies based on the prompt "Hello, I'm a language model,"

Google Colab

```
!pip install transformers
#--- --- --- libraries
from transformers import pipeline, set_seed

generator = pipeline('text-generation', model='openai-gpt')

set_seed(42)

generator("Hello, I'm a language model,", max_length=30, num_return_sequences=5)
```

Google Colab: **output** after the code in the cell is executed

```
[{'generated_text': "Hello, I'm a language model, for'e's a very big one. they call me'e'n e'n""},
 {'generated_text': 'Hello, I\'m a language model, " he said and leaned his left shoulder against the corner of the door. \n i couldn\'t take my'},
 {'generated_text': "Hello, I'm a language model,'came their reply. \n'it's a man thing! if you have three hundred thousand words,"},
 {'generated_text': 'Hello, I\'m a language model, " he said, the sound of his voice sending a shiver down her spine. \n " a language model'},
 {'generated_text': "Hello, I'm a language model,'he said.'you call me a'man out'in... well, er, in the"}]
```

Response to the prompt

University of Southampton

Response: 5 replies based on the prompt "I am unhappy."

```
# _____ Google Colab code

!pip install transformers
#--- --- --- libraries
from transformers import pipeline, set_seed

generator = pipeline('text-generation', model='openai-gpt')
set_seed(42)
generator("I am unhappy.", max_length=30, num_return_sequences=5)

# _____ Google Colab output

[{'generated_text': 'I am unhappy. for the first time since i began this mission, it\'s all over. i\'m ready to go home. " \n " my'},
 {'generated_text': 'I am unhappy. \n if this is indeed a sign of the prophecy... \n then there has to be a way to avert the coming apocalypse. \n'},
 {'generated_text': 'I am unhappy. i\'ve tried to understand why you are like this, but i\'ve no idea why you\'re being like this. " \n he'},
 {'generated_text': 'I am unhappy. " even though he made her miserable, despite everything he \'d done, she needed him. \n his hand stilled on her back,'},
 {'generated_text': 'I am unhappy. " \n " you can be unhappy all you want, but let it lay dormant until you finally get the answer. " \n "'}]
```

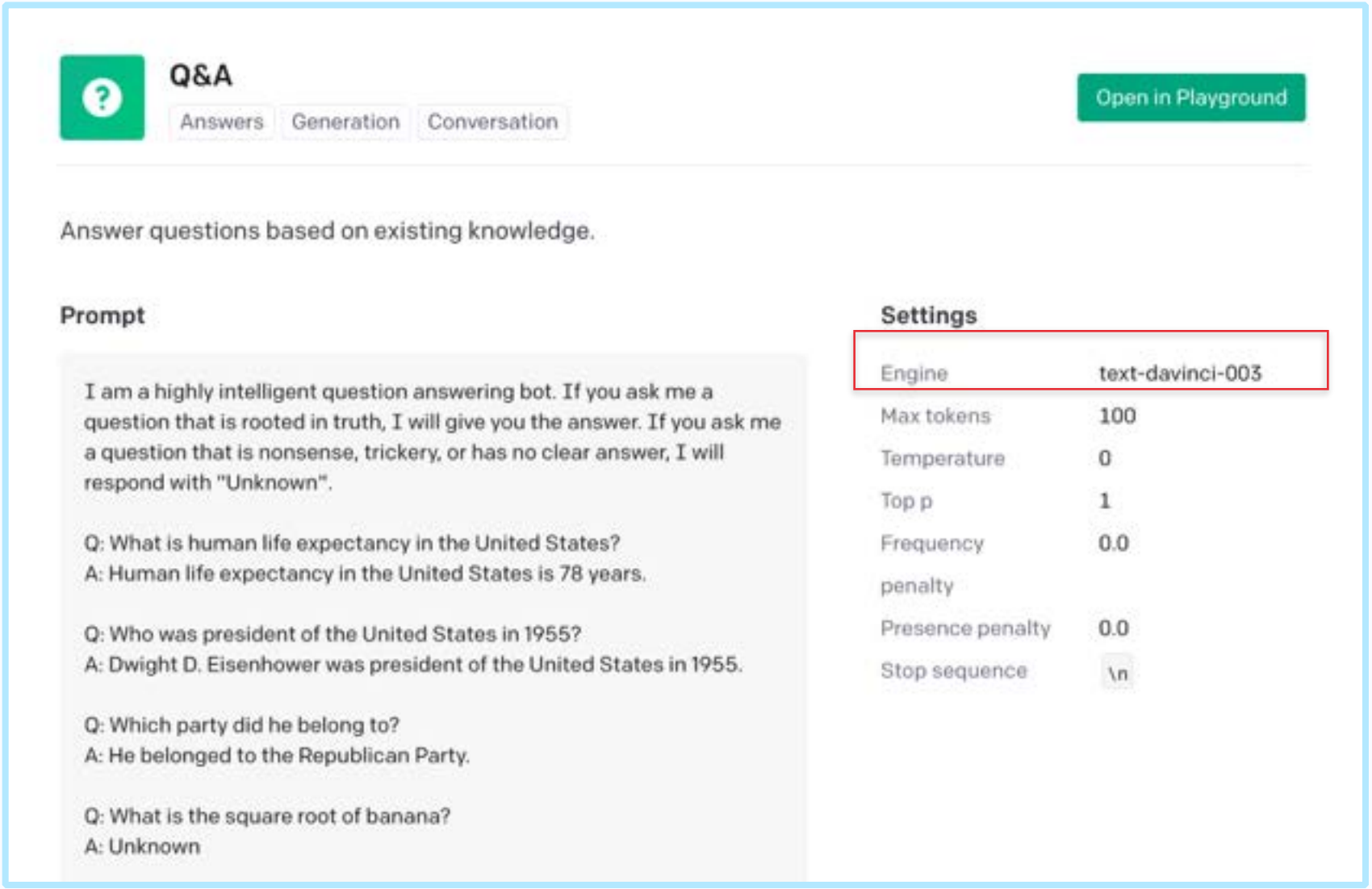
```
# _____ Google Colab code

!pip install transformers
#--- --- --- libraries
from transformers import pipeline, set_seed

generator = pipeline('text-generation', model='gpt2')
set_seed(42)
generator("I am unhappy.", max_length=30, num_return_sequences=5)

# _____ Google Colab output

[{'generated_text': 'I am unhappy. It means that some people want to go to a hospital for minor problems such as burns or diarrhoea. I do not like'},
 {'generated_text': 'I am unhappy.\n\nI would love to see more people like me before I die. I am still so much in my young years. For'},
 {'generated_text': "I am unhappy.\n\nYou should get in touch with your representative and tell us as to why you didn't get into it with this person."},
 {'generated_text': 'I am unhappy.\n\nAdvertisement Continue reading the main story\n\nThe United States has spent more than $700 billion over the last few years to'},
 {'generated_text': 'I am unhappy. I think all of us in our present circumstances have tried a good deal to make sense of them."\n\nThe president, though'}]
```

OpenAI examples

https://platform.openai.com/examples

University of Southampton

OpenAI: one of the multiple example applications



GPT-3: response to prompt

```
import openai

openai.api_key = "[redacted]"

response = openai.Completion.create(
  model="text-davinci-003",
  prompt="do you have consciousness? ->\n",
  temperature=0.7,
  max_tokens=256,
  top_p=1,
  frequency_penalty=0,
  presence_penalty=0,
  stop=["END"]
)

print(response)
```


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "text": "\nNo, I do not have consciousness."
    }
  ],
  "created": 1683646955,
  "id": "cmpl-7EJlzxVG5CYuBWbzbnYz6ArOte5RV",
  "model": "text-davinci-003",
  "object": "text_completion",
  "usage": {
    "completion_tokens": 9,
    "prompt_tokens": 7,
    "total_tokens": 16
  }
}
```


University of Southampton

Unlike the **GPT-2** model available for instance on Hugging Face, we don't have direct access to the **GPT-3** model. Therefore, you first need to get an API key from OpenAI

Google Colab: output after the code in the cell is executed

Large Language Models (LLMs)

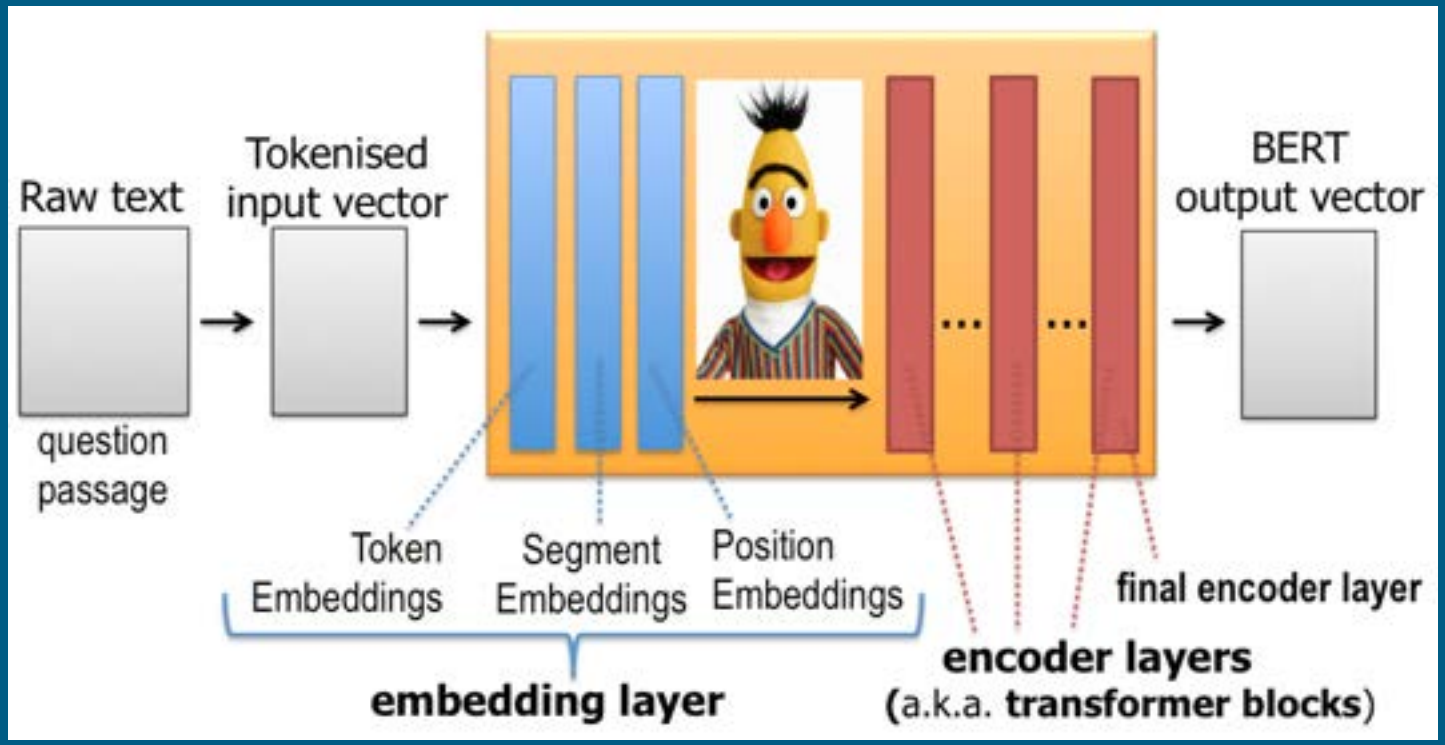
University of Southampton

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Proceedings of NAACL-HLT
June 2019

LLMs can be adapted with fine-tuning to a wide range of natural language tasks



BERT for QA by passing to BERT model a question and a passage

Large Language Models (LLMs)

University of Southampton

BERT for QA by passing to BERT model a question and a passage

```
# _____ Google Colab code

!pip install sentence-transformers

# load library
from sentence_transformers import SentenceTransformer, util
# load model
model = SentenceTransformer('bert-base-nli-mean-tokens')

# question
query_embedding = model.encode('How many people live in London?')
# passage
# a passage is encoded as [title, text]
passage_embedding = model.encode([['London', 'London has 9,787,426 inhabitants at the 2011 census.']])

# compute and display cosine similarity between question and passage
print(util.pytorch_cos_sim(query_embedding, passage_embedding))

# _____ Google Colab output

tensor([[0.6778]])
```

LLM fine-tuned for Sentiment Analysis

https://huggingface.co/siebert/sentiment-roberta-large-english

University of Southampton

```
!pip install transformers
from transformers import pipeline

sentiment_pipeline = pipeline("sentiment-analysis",model="siebert/sentiment-roberta-large-english")
data = ["I love you", "I hate you"]
sentiment_pipeline(data)
```

Google Colab: **output** after the code in the cell is executed

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.28.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (23.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2022.10.31)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.22.4)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.12.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.11.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.14.1
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.27.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.65.0)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.11.0->transformers)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests->transformers)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (1.2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2022.1
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4)
Downloading (...)lve/main/config.json: 100% 687/687 [00:00<00:00, 36.5kB/s]
Downloading pytorch_model.bin: 100% 1.42G/1.42G [00:06<00:00, 196MB/s]
Downloading (...)okenizer_config.json: 100% 256/256 [00:00<00:00, 6.05kB/s]
Downloading (...)olve/main/vocab.json: 100% 798k/798k [00:00<00:00, 16.7MB/s]
Downloading (...)olve/main/merges.txt: 100% 456k/456k [00:00<00:00, 14.0MB/s]
Downloading (...)cial_tokens_map.json: 100% 150/150 [00:00<00:00, 4.77kB/s]
[{'label': 'POSITIVE', 'score': 0.998561680316925},
 {'label': 'NEGATIVE', 'score': 0.9991401433944702}]
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 1 install & load libraries

```
!pip install transformers
!pip install datasets
#--- --- --- libraries
import numpy as np
import pandas as pd

from tqdm.auto import tqdm

import torch
from torch.utils.data import DataLoader

import transformers
from transformers import AutoTokenizer, AutoModelForSequenceClassification, get_scheduler, AdamW,
TrainingArguments, Trainer

from datasets import Dataset, load_metric

print("libraries loaded")
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 2 load dataset

```
#--- --- --- Google Colab can access your Google Drive
from google.colab import drive
drive.mount('/content/gdrive')

#--- --- --- https://raw.githubusercontent.com/clairett/pytorch-sentiment-classification/master/data/SST2/train.tsv

#--- --- --- dataset

df=pd.read_csv("gdrive/My Drive/2023temp2/train.tsv",delimiter='\t', header=None)
###df.columns = ["text", "label"]
df.columns = ["text", "labels"]

print(df.head())
print('--- --- ---')

batch_1 = df[:2000]
###batch_1[1].value_counts()

dataset = Dataset.from_pandas(batch_1)
print(dataset)
###print(dataset[0])

ds2 = dataset.train_test_split(test_size=0.1)
print(ds2)
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 2 load dataset [output]

```
                                                   text  labels
0  a stirring , funny and finally transporting re...       1
1  apparently reassembled from the cutting room f...       0
2  they presume their audience wo n't sit still f...       0
3  this is a visually stunning rumination on love...       1
4  jonathan parker 's bartleby should have been t...       1
--- --- ---
Dataset({
    features: ['text', 'labels'],
    num_rows: 2000
})
DatasetDict({
    train: Dataset({
        features: ['text', 'labels'],
        num_rows: 1800
    })
    test: Dataset({
        features: ['text', 'labels'],
        num_rows: 200
    })
})
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 3 prepare dataset

```
#--- --- --- prepare dataset for BERT
tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")

def tokenize_function(examples):
    return tokenizer(examples["text"], padding="max_length", truncation=True)

tokenized_datasets = ds2.map(tokenize_function, batched=True)

#--- --- --- divide dataset into training and evaluation

tokenized_datasets = tokenized_datasets.remove_columns(["text"])
tokenized_datasets.set_format("torch")

small_train_dataset = tokenized_datasets["train"]
small_eval_dataset = tokenized_datasets["test"]

#---
train_dataloader = DataLoader(small_train_dataset, shuffle=True, batch_size=8)
eval_dataloader = DataLoader(small_eval_dataset, batch_size=8)
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 4 model & scheduler

```
#--- --- --- select BERT model
model = AutoModelForSequenceClassification.from_pretrained("bert-base-cased", num_labels=2)

optimizer = AdamW(model.parameters(), lr=5e-5)

#--- --- --- create scheduler
num_epochs = 3
num_training_steps = num_epochs * len(train_dataloader)
lr_scheduler = get_scheduler(
    "linear",
    optimizer=optimizer,
    num_warmup_steps=0,
    num_training_steps=num_training_steps
)
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 5 fine-tune model

```
device = torch.device("cuda") if torch.cuda.is_available() else torch.device("cpu")
model.to(device)

#--- --- --- training [ it will take time! ]

progress_bar = tqdm(range(num_training_steps))

model.train()
for epoch in range(num_epochs):
    for batch in train_dataloader:
        batch = {k: v.to(device) for k, v in batch.items()}
        outputs = model(**batch)
        loss = outputs.loss
        loss.backward()

        optimizer.step()
        lr_scheduler.step()
        optimizer.zero_grad()
        progress_bar.update(1)
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 5 fine-tune [progress]

```
device = torch.device instance orch.cuda.is_available() else torch.device("cpu")
model.to(device)
#--- --- --- training [ it will take time! ]
progress_bar = tqdm(range(num_training_steps))

model.train()
for epoch in range(num_epochs):
    for batch in train_dataloader:
        batch = {k: v.to(device) for k, v in batch.items()}
        outputs = model(**batch)
        loss = outputs.loss
        loss.backward()

        optimizer.step()
        lr_scheduler.step()
        optimizer.zero_grad()
        progress_bar.update(1)
```

```
100% 675/675 [02:11<00:00, 5.12it/s]
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 6 evaluation [performance?]

```
#--- --- --- evaluation
from sklearn.metrics import accuracy_score, precision_recall_fscore_support

def compute_metrics(pred):
   labels = pred.label_ids
   preds = pred.predictions.argmax(-1)
   precision, recall, f1, _ = precision_recall_fscore_support(labels, preds, average='binary')
   acc = accuracy_score(labels, preds)


   print('--- --- ---')
   print('acc...' + "{0:.2f}".format(round(100*acc,2)) )
   print ('f1...' + "{0:.2f}".format(round(100*f1,2)) )
   print ('precision...' + "{0:.2f}".format(round(100*precision,2)) )
   print ('recall...' + "{0:.2f}".format(round(100*recall,2)) )
   print('--- --- ---')
   print('--- --- --- --- --- --- --- --- --- --- --- --- ---')

   return {
      'accuracy': acc,
      'f1': f1,
      'precision': precision,
      'recall': recall
   }

#--- --- ---

training_args = TrainingArguments("test_trainer")

trainer = Trainer(
   model=model,
   args=training_args,
   train_dataset=small_train_dataset,
   eval_dataset=small_eval_dataset,
   compute_metrics=compute_metrics,
)
print(trainer.evaluate())
```

Large Language Models (LLMs)

University of Southampton

Fine-tune of BERT model for Sentiment Analysis: step 6 evaluation [performance?]


```
--- --- ---
acc...91.50
f1...91.19
precision...92.63
recall...89.80
--- --- ---
--- --- --- --- --- --- --- --- --- --- --- ---
{'eval_loss': 0.34426993131637573, 'eval_accuracy': 0.915, 'eval_f1': 0.911917098445596,
```

Preliminaries: datasets

https://microsoft.github.io/BLURB/

University of Southampton

BLURB: the Biomedical Language Understanding and Reasoning Benchmark

Exemplifying datasets used in the BLURB biomedical NLP benchmark.

Dataset	Task	Train	Dev	Test	Evaluation Metrics
BC5-chem	NER	5203	5347	5385	F1 entity-level
BC5-disease	NER	4182	4244	4424	F1 entity-level
NCBI-disease	NER	5134	787	960	F1 entity-level
BC2GM	NER	15197	3061	6325	F1 entity-level
JNLPBA	NER	46750	4551	8662	F1 entity-level
EBM PICO	PICO	339167	85321	16364	Macro F1 word-level
ChemProt	Relation Extraction	18035	11268	15745	Micro F1
DDI	Relation Extraction	25296	2496	5716	Micro F1
GAD	Relation Extraction	4261	535	534	Micro F1
BIOSSES	Sentence Similarity	64	16	20	Pearson
HoC	Document Classification	1295	186	371	Micro F1
PubMedQA	Question Answering	450	50	500	Accuracy
BioASQ	Question Answering	670	75	140	Accuracy

Preliminaries: datasets

https://microsoft.github.io/BLURB/leaderboard.html

University of Southampton

BLURB: the Biomedical Language Understanding and Reasoning Benchmark

BLURB — Leaderboard | Paper | Models | Tasks | Submit | News

The Overall score is calculated as the macro-average performance over tasks. Details can be found within our publication.

Show 100 entries

Rank	Model	BLURB Score (Macro Avg.)	Micro Avg.	NER	PICO	RE	SS	Class.	QA
1	**BioLinkBERT-Large** — Stanford	**84.30**	**84.80**	**86.89**	**74.19**	82.74	93.63	**84.88**	83.50
2	**BioM-ELECTRA-Large** — University of Delaware	83.81	84.67	86.88	73.67	**83.17**	91.09	84.03	**84.00**
3	**BioLinkBERT-Base** — Stanford	83.39	83.84	86.39	73.97	81.56	93.27	84.35	80.81

Preliminaries: datasets

https://gluebenchmark.com/

University of Southampton

GLUE: the General Language Understanding Evaluation benchmark

Corpus	|Train|	|Test|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	**1k**	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	**391k**	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	**20k**	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	**146**	coreference/NLI	acc.	fiction books

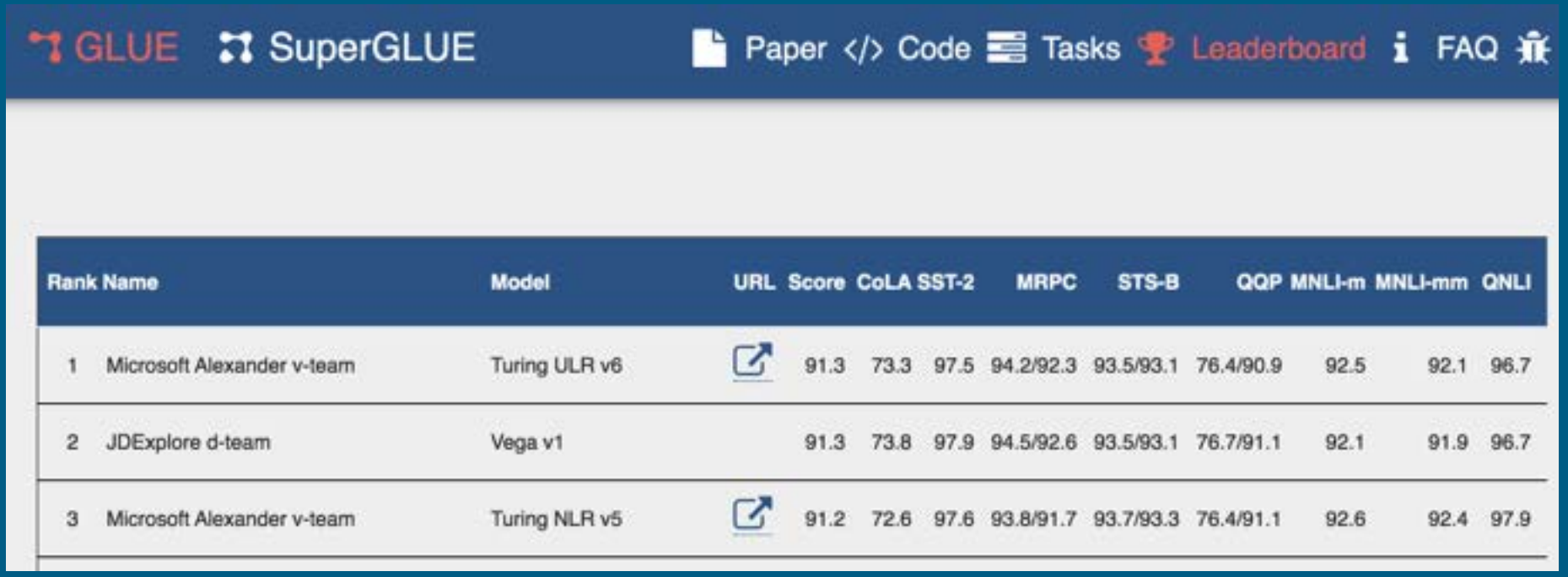
NLI = Natural Language Inference

Preliminaries: datasets

https://gluebenchmark.com/leaderboard/

University of Southampton

GLUE: the General Language Understanding Evaluation benchmark



Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI
1	Microsoft Alexander v-team	Turing ULR v6		91.3	73.3	97.5	94.2/92.3	93.5/93.1	76.4/90.9	92.5	92.1	96.7
2	JDExplore d-team	Vega v1		91.3	73.8	97.9	94.5/92.6	93.5/93.1	76.7/91.1	92.1	91.9	96.7
3	Microsoft Alexander v-team	Turing NLR v5		91.2	72.6	97.6	93.8/91.7	93.7/93.3	76.4/91.1	92.6	92.4	97.9

Large Language Models (LLMs)

University of Southampton

XAI: investigating the fine-tuning of LLMs with fewer instances (~ 2K)

Fine-tune transformer models for NLP downstream tasks requires thousands of manually labelled instances (e.g. BLURB datasets). We investigate the reduction of labelled instances by following an approach from the field of explainable active learning, which is a subfield of Explainable Artificial Intelligence (XAI).

SGAI 2021

Named Entity Recognition and Relation Extraction for COVID-19: Explainable Active Learning with Word2vec Embeddings and Transformer-Based BERT Models

M. Arguello-Casteleiro[1(✉)], N. Maroto[2], C. Wroe[3], C. Sevillano Torrado[4], C. Henson[5], J. Des-Diz[4], M. J. Fernandez-Prieto[6], T. Furmston[1], D. Maseda Fernandez[5], M. Kulshrestha[5], R. Stevens[1], J. Keane[1], and S. Peters[1]

SWAT4HCLS 2022

MetaMap versus BERT models with explainable active learning: ontology-based experiments with prior knowledge for COVID-19

M. Arguello-Casteleiro[1], C. Henson[2], N. Maroto[3], S. Li[4], J. Des-Diz[5], M.J. Fernandez-Prieto[6], S. Peters[1], T. Furmston[1], C. Sevillano Torrado[5], D. Maseda Fernandez[2], M. Kulshrestha[2], J. Keane[1], R. Stevens[1], and C. Wroe[7]

Special thanks for technical contributions to XAI for women's voices (2023)

Rudraansh Kotra
Shumeng Guo
Jessica Newman
Hock Nien Gan
Marcus Lam

Special thanks to Samuel Suarez Barbeira for fruitful discussion

Images shown may have a copyright

University of Southampton

Questions