Custom_Bert_Model_AdaFactor_3e

August 10, 2021

Question answering comes in many forms. In this example, we'll look at the particular type of extractive QA that involves answering a question about a passage by highlighting the segment of the passage that answers the question. This involves fine-tuning a model which predicts a start position and an end position in the passage. We will use the Stanford Question Answering Dataset (SQuAD) 2.0.

0.1 Prerequisites:

1. Download and install the required libraries below.
2. Import the required libraries

[1]:
```
!pip install torch
!pip install transformers
!pip install sentencepiece
!pip install wandb
!pip install allennlp
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.7/dist-packages
(1.9.0+cu102)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from torch) (3.7.4.3)
Collecting transformers
  Downloading transformers-4.9.1-py3-none-any.whl (2.6 MB)
     |                | 2.6 MB 14.5 MB/s
Collecting pyyaml>=5.1
  Downloading PyYAML-5.4.1-cp37-cp37m-manylinux1_x86_64.whl (636 kB)
     |                | 636 kB 62.0 MB/s
Collecting sacremoses
  Downloading sacremoses-0.0.45-py3-none-any.whl (895 kB)
     |                | 895 kB 65.3 MB/s
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from transformers) (4.6.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
packages (from transformers) (1.19.5)
Collecting tokenizers<0.11,>=0.10.1
  Downloading tokenizers-0.10.3-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_6
4.manylinux_2_12_x86_64.manylinux2010_x86_64.whl (3.3 MB)
     |                | 3.3 MB 60.9 MB/s
Collecting huggingface-hub==0.0.12
```

```
  Downloading huggingface_hub-0.0.12-py3-none-any.whl (37 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-
packages (from transformers) (3.0.12)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-
packages (from transformers) (21.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-
packages (from transformers) (2.23.0)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.7/dist-packages (from transformers) (2019.12.20)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-
packages (from transformers) (4.41.1)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from huggingface-
hub==0.0.12->transformers) (3.7.4.3)
Requirement already satisfied: pyparsing>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from packaging->transformers) (2.4.7)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->transformers) (3.5.0)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests->transformers) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (2021.5.30)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (1.15.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (1.0.1)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers) (7.1.2)
Installing collected packages: tokenizers, sacremoses, pyyaml, huggingface-hub,
transformers
  Attempting uninstall: pyyaml
    Found existing installation: PyYAML 3.13
    Uninstalling PyYAML-3.13:
      Successfully uninstalled PyYAML-3.13
Successfully installed huggingface-hub-0.0.12 pyyaml-5.4.1 sacremoses-0.0.45
tokenizers-0.10.3 transformers-4.9.1
Collecting sentencepiece
  Downloading
sentencepiece-0.1.96-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(1.2 MB)
     |                  | 1.2 MB 14.1 MB/s
Installing collected packages: sentencepiece
Successfully installed sentencepiece-0.1.96
Collecting wandb
```

```
  Downloading wandb-0.11.2-py2.py3-none-any.whl (1.8 MB)
     |                  | 1.8 MB 12.4 MB/s
Requirement already satisfied: requests<3,>=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from wandb) (2.23.0)
Collecting pathtools
  Downloading pathtools-0.1.2.tar.gz (11 kB)
Requirement already satisfied: Click!=8.0.0,>=7.0 in
/usr/local/lib/python3.7/dist-packages (from wandb) (7.1.2)
Collecting GitPython>=1.0.0
  Downloading GitPython-3.1.18-py3-none-any.whl (170 kB)
     |                  | 170 kB 57.3 MB/s
Collecting docker-pycreds>=0.4.0
  Downloading docker_pycreds-0.4.0-py2.py3-none-any.whl (9.0 kB)
Collecting sentry-sdk>=1.0.0
  Downloading sentry_sdk-1.3.1-py2.py3-none-any.whl (133 kB)
     |                  | 133 kB 64.1 MB/s
Requirement already satisfied: python-dateutil>=2.6.1 in
/usr/local/lib/python3.7/dist-packages (from wandb) (2.8.1)
Requirement already satisfied: six>=1.13.0 in /usr/local/lib/python3.7/dist-
packages (from wandb) (1.15.0)
Requirement already satisfied: promise<3,>=2.0 in /usr/local/lib/python3.7/dist-
packages (from wandb) (2.3)
Collecting shortuuid>=0.5.0
  Downloading shortuuid-1.0.1-py3-none-any.whl (7.5 kB)
Requirement already satisfied: psutil>=5.0.0 in /usr/local/lib/python3.7/dist-
packages (from wandb) (5.4.8)
Collecting subprocess32>=3.5.3
  Downloading subprocess32-3.5.4.tar.gz (97 kB)
     |                  | 97 kB 8.3 MB/s
Collecting configparser>=3.8.1
  Downloading configparser-5.0.2-py3-none-any.whl (19 kB)
Requirement already satisfied: PyYAML in /usr/local/lib/python3.7/dist-packages
(from wandb) (5.4.1)
Collecting urllib3>=1.26.5
  Downloading urllib3-1.26.6-py2.py3-none-any.whl (138 kB)
     |                  | 138 kB 63.2 MB/s
Requirement already satisfied: protobuf>=3.12.0 in
/usr/local/lib/python3.7/dist-packages (from wandb) (3.17.3)
Collecting gitdb<5,>=4.0.1
  Downloading gitdb-4.0.7-py3-none-any.whl (63 kB)
     |                  | 63 kB 2.3 MB/s
Requirement already satisfied: typing-extensions>=3.7.4.0 in
/usr/local/lib/python3.7/dist-packages (from GitPython>=1.0.0->wandb) (3.7.4.3)
Collecting smmap<5,>=3.0.1
  Downloading smmap-4.0.0-py2.py3-none-any.whl (24 kB)
Collecting requests<3,>=2.0.0
  Downloading requests-2.26.0-py2.py3-none-any.whl (62 kB)
     |                  | 62 kB 1.1 MB/s
```

```
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.7/dist-packages (from requests<3,>=2.0.0->wandb) (2.10)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from requests<3,>=2.0.0->wandb) (2.0.2)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests<3,>=2.0.0->wandb)
(2021.5.30)
Building wheels for collected packages: subprocess32, pathtools
  Building wheel for subprocess32 (setup.py) … done
  Created wheel for subprocess32: filename=subprocess32-3.5.4-py3-none-any.whl
size=6502
sha256=70d7aed3fc55a42b15e484748f907db663e583983339fe47e42bc0db2fcaa294
  Stored in directory: /root/.cache/pip/wheels/50/ca/fa/8fca8d246e64f19488d07567
547ddec8eb084e8c0d7a59226a
  Building wheel for pathtools (setup.py) … done
  Created wheel for pathtools: filename=pathtools-0.1.2-py3-none-any.whl
size=8806
sha256=8c21a1962a7cafa21a3a05a2a757cbbb749eb8532645cbc2c8624143b551cbec
  Stored in directory: /root/.cache/pip/wheels/3e/31/09/fa59cef12cdcfecc627b3d24
273699f390e71828921b2cbba2
Successfully built subprocess32 pathtools
Installing collected packages: smmap, urllib3, gitdb, subprocess32, shortuuid,
sentry-sdk, requests, pathtools, GitPython, docker-pycreds, configparser, wandb
  Attempting uninstall: urllib3
    Found existing installation: urllib3 1.24.3
    Uninstalling urllib3-1.24.3:
      Successfully uninstalled urllib3-1.24.3
  Attempting uninstall: requests
    Found existing installation: requests 2.23.0
    Uninstalling requests-2.23.0:
      Successfully uninstalled requests-2.23.0
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.26.0 which
is incompatible.
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is
incompatible.
Successfully installed GitPython-3.1.18 configparser-5.0.2 docker-pycreds-0.4.0
gitdb-4.0.7 pathtools-0.1.2 requests-2.26.0 sentry-sdk-1.3.1 shortuuid-1.0.1
smmap-4.0.0 subprocess32-3.5.4 urllib3-1.26.6 wandb-0.11.2
Collecting allennlp
  Downloading allennlp-2.6.0-py3-none-any.whl (689 kB)
     |                  | 689 kB 14.6 MB/s
```

```
Requirement already satisfied: wandb<0.12.0,>=0.10.0 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (0.11.2)
Collecting transformers<4.9,>=4.1
  Downloading transformers-4.8.2-py3-none-any.whl (2.5 MB)
     |                  | 2.5 MB 24.6 MB/s
Requirement already satisfied: sentencepiece in
/usr/local/lib/python3.7/dist-packages (from allennlp) (0.1.96)
Collecting overrides==3.1.0
  Downloading overrides-3.1.0.tar.gz (11 kB)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages
(from allennlp) (1.19.5)
Requirement already satisfied: requests>=2.18 in /usr/local/lib/python3.7/dist-
packages (from allennlp) (2.26.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-
packages (from allennlp) (0.22.2.post1)
Requirement already satisfied: lmdb in /usr/local/lib/python3.7/dist-packages
(from allennlp) (0.99)
Requirement already satisfied: huggingface-hub>=0.0.8 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (0.0.12)
Requirement already satisfied: filelock<3.1,>=3.0 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (3.0.12)
Requirement already satisfied: termcolor==1.1.0 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (1.1.0)
Collecting google-cloud-storage<1.42.0,>=1.38.0
  Downloading google_cloud_storage-1.41.1-py2.py3-none-any.whl (105 kB)
     |                  | 105 kB 58.6 MB/s
Requirement already satisfied: torchvision<0.11.0,>=0.8.1 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (0.10.0+cu102)
Collecting boto3<2.0,>=1.14
  Downloading boto3-1.18.16-py3-none-any.whl (131 kB)
     |                  | 131 kB 63.0 MB/s
Collecting checklist==0.0.11
  Downloading checklist-0.0.11.tar.gz (12.1 MB)
     |                  | 12.1 MB 24.8 MB/s
Collecting jsonnet>=0.10.0
  Downloading jsonnet-0.17.0.tar.gz (259 kB)
     |                  | 259 kB 50.1 MB/s
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-
packages (from allennlp) (1.4.1)
Collecting tensorboardX>=1.2
  Downloading tensorboardX-2.4-py2.py3-none-any.whl (124 kB)
     |                  | 124 kB 61.1 MB/s
Requirement already satisfied: more-itertools in
/usr/local/lib/python3.7/dist-packages (from allennlp) (8.8.0)
Requirement already satisfied: h5py in /usr/local/lib/python3.7/dist-packages
(from allennlp) (3.1.0)
Requirement already satisfied: spacy<3.1,>=2.1.0 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (2.2.4)
```

```
Requirement already satisfied: pytest in /usr/local/lib/python3.7/dist-packages
(from allennlp) (3.6.4)
Requirement already satisfied: torch<1.10.0,>=1.6.0 in
/usr/local/lib/python3.7/dist-packages (from allennlp) (1.9.0+cu102)
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages
(from allennlp) (3.2.5)
Requirement already satisfied: tqdm>=4.19 in /usr/local/lib/python3.7/dist-
packages (from allennlp) (4.41.1)
Collecting munch>=2.5
  Downloading munch-2.5.0-py2.py3-none-any.whl (10 kB)
Requirement already satisfied: dill>=0.3.1 in /usr/local/lib/python3.7/dist-
packages (from checklist==0.0.11->allennlp) (0.3.4)
Requirement already satisfied: jupyter>=1.0 in /usr/local/lib/python3.7/dist-
packages (from checklist==0.0.11->allennlp) (1.0.0)
Requirement already satisfied: ipywidgets>=7.5 in /usr/local/lib/python3.7/dist-
packages (from checklist==0.0.11->allennlp) (7.6.3)
Collecting patternfork-nosql
  Downloading patternfork_nosql-3.6.tar.gz (22.3 MB)
     |                  | 22.3 MB 1.3 MB/s
Collecting iso-639
  Downloading iso-639-0.4.5.tar.gz (167 kB)
     |                  | 167 kB 64.3 MB/s
Collecting jmespath<1.0.0,>=0.7.1
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting s3transfer<0.6.0,>=0.5.0
  Downloading s3transfer-0.5.0-py3-none-any.whl (79 kB)
     |                  | 79 kB 10.1 MB/s
Collecting botocore<1.22.0,>=1.21.16
  Downloading botocore-1.21.16-py3-none-any.whl (7.8 MB)
     |                  | 7.8 MB 49.4 MB/s
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in
/usr/local/lib/python3.7/dist-packages (from
botocore<1.22.0,>=1.21.16->boto3<2.0,>=1.14->allennlp) (2.8.1)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in
/usr/local/lib/python3.7/dist-packages (from
botocore<1.22.0,>=1.21.16->boto3<2.0,>=1.14->allennlp) (1.26.6)
Collecting google-resumable-media<3.0dev,>=1.3.0
  Downloading google_resumable_media-1.3.3-py2.py3-none-any.whl (75 kB)
     |                  | 75 kB 6.2 MB/s
Requirement already satisfied: google-auth<3.0dev,>=1.24.0 in
/usr/local/lib/python3.7/dist-packages (from google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (1.32.1)
Collecting google-cloud-core<3.0dev,>=1.6.0
  Downloading google_cloud_core-1.7.2-py2.py3-none-any.whl (28 kB)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.7/dist-packages (from google-
auth<3.0dev,>=1.24.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (0.2.8)
Requirement already satisfied: setuptools>=40.3.0 in
```

```
/usr/local/lib/python3.7/dist-packages (from google-
auth<3.0dev,>=1.24.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (57.2.0)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-
packages (from google-auth<3.0dev,>=1.24.0->google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (1.15.0)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from google-
auth<3.0dev,>=1.24.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (4.2.2)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-
packages (from google-auth<3.0dev,>=1.24.0->google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (4.7.2)
Requirement already satisfied: google-api-core<2.0.0dev,>=1.21.0 in
/usr/local/lib/python3.7/dist-packages (from google-cloud-
core<3.0dev,>=1.6.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (1.26.3)
Requirement already satisfied: packaging>=14.3 in /usr/local/lib/python3.7/dist-
packages (from google-api-core<2.0.0dev,>=1.21.0->google-cloud-
core<3.0dev,>=1.6.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (21.0)
Requirement already satisfied: googleapis-common-protos<2.0dev,>=1.6.0 in
/usr/local/lib/python3.7/dist-packages (from google-api-
core<2.0.0dev,>=1.21.0->google-cloud-core<3.0dev,>=1.6.0->google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (1.53.0)
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages
(from google-api-core<2.0.0dev,>=1.21.0->google-cloud-
core<3.0dev,>=1.6.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (2018.9)
Requirement already satisfied: protobuf>=3.12.0 in
/usr/local/lib/python3.7/dist-packages (from google-api-
core<2.0.0dev,>=1.21.0->google-cloud-core<3.0dev,>=1.6.0->google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (3.17.3)
Collecting google-crc32c<2.0dev,>=1.0
  Downloading google_crc32c-1.1.2-cp37-cp37m-manylinux2014_x86_64.whl (38 kB)
Requirement already satisfied: cffi>=1.0.0 in /usr/local/lib/python3.7/dist-
packages (from google-crc32c<2.0dev,>=1.0->google-resumable-
media<3.0dev,>=1.3.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (1.14.6)
Requirement already satisfied: pycparser in /usr/local/lib/python3.7/dist-
packages (from cffi>=1.0.0->google-crc32c<2.0dev,>=1.0->google-resumable-
media<3.0dev,>=1.3.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (2.20)
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from huggingface-hub>=0.0.8->allennlp)
(4.6.1)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from huggingface-hub>=0.0.8->allennlp)
(3.7.4.3)
Requirement already satisfied: traitlets>=4.3.1 in
/usr/local/lib/python3.7/dist-packages (from
ipywidgets>=7.5->checklist==0.0.11->allennlp) (5.0.5)
Requirement already satisfied: ipython>=4.0.0 in /usr/local/lib/python3.7/dist-
packages (from ipywidgets>=7.5->checklist==0.0.11->allennlp) (5.5.0)
Requirement already satisfied: ipykernel>=4.5.1 in
```

```
/usr/local/lib/python3.7/dist-packages (from
ipywidgets>=7.5->checklist==0.0.11->allennlp) (4.10.1)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in
/usr/local/lib/python3.7/dist-packages (from
ipywidgets>=7.5->checklist==0.0.11->allennlp) (1.0.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in
/usr/local/lib/python3.7/dist-packages (from
ipywidgets>=7.5->checklist==0.0.11->allennlp) (3.5.1)
Requirement already satisfied: nbformat>=4.2.0 in /usr/local/lib/python3.7/dist-
packages (from ipywidgets>=7.5->checklist==0.0.11->allennlp) (5.1.3)
Requirement already satisfied: tornado>=4.0 in /usr/local/lib/python3.7/dist-
packages (from ipykernel>=4.5.1->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(5.1.1)
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.7/dist-
packages (from ipykernel>=4.5.1->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(5.3.5)
Requirement already satisfied: simplegeneric>0.8 in
/usr/local/lib/python3.7/dist-packages (from
ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp) (0.8.1)
Requirement already satisfied: pexpect in /usr/local/lib/python3.7/dist-packages
(from ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp) (4.8.0)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.7/dist-
packages (from ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(0.7.5)
Requirement already satisfied: prompt-toolkit<2.0.0,>=1.0.4 in
/usr/local/lib/python3.7/dist-packages (from
ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp) (1.0.18)
Requirement already satisfied: pygments in /usr/local/lib/python3.7/dist-
packages (from ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(2.6.1)
Requirement already satisfied: decorator in /usr/local/lib/python3.7/dist-
packages (from ipython>=4.0.0->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(4.4.2)
Requirement already satisfied: notebook in /usr/local/lib/python3.7/dist-
packages (from jupyter>=1.0->checklist==0.0.11->allennlp) (5.3.1)
Requirement already satisfied: qtconsole in /usr/local/lib/python3.7/dist-
packages (from jupyter>=1.0->checklist==0.0.11->allennlp) (5.1.1)
Requirement already satisfied: nbconvert in /usr/local/lib/python3.7/dist-
packages (from jupyter>=1.0->checklist==0.0.11->allennlp) (5.6.1)
Requirement already satisfied: jupyter-console in /usr/local/lib/python3.7/dist-
packages (from jupyter>=1.0->checklist==0.0.11->allennlp) (5.2.0)
Requirement already satisfied: jupyter-core in /usr/local/lib/python3.7/dist-
packages (from nbformat>=4.2.0->ipywidgets>=7.5->checklist==0.0.11->allennlp)
(4.7.1)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
/usr/local/lib/python3.7/dist-packages (from
nbformat>=4.2.0->ipywidgets>=7.5->checklist==0.0.11->allennlp) (2.6.0)
Requirement already satisfied: ipython-genutils in
```

```
/usr/local/lib/python3.7/dist-packages (from
nbformat>=4.2.0->ipywidgets>=7.5->checklist==0.0.11->allennlp) (0.2.0)
Requirement already satisfied: pyparsing>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from packaging>=14.3->google-api-
core<2.0.0dev,>=1.21.0->google-cloud-core<3.0dev,>=1.6.0->google-cloud-
storage<1.42.0,>=1.38.0->allennlp) (2.4.7)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages
(from prompt-toolkit<2.0.0,>=1.0.4->ipython>=4.0.0->ipywidgets>=7.5->checklist==
0.0.11->allennlp) (0.2.5)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/usr/local/lib/python3.7/dist-packages (from pyasn1-modules>=0.2.1->google-
auth<3.0dev,>=1.24.0->google-cloud-storage<1.42.0,>=1.38.0->allennlp) (0.4.8)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests>=2.18->allennlp) (2.10)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.18->allennlp) (2.0.2)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.18->allennlp)
(2021.5.30)
Requirement already satisfied: srsly<1.1.0,>=1.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(1.0.5)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(3.0.5)
Requirement already satisfied: catalogue<1.1.0,>=0.0.7 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(1.0.0)
Requirement already satisfied: plac<1.2.0,>=0.9.6 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(1.1.3)
Requirement already satisfied: blis<0.5.0,>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(0.4.1)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(1.0.5)
Requirement already satisfied: thinc==7.4.0 in /usr/local/lib/python3.7/dist-
packages (from spacy<3.1,>=2.1.0->allennlp) (7.4.0)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(2.0.5)
Requirement already satisfied: wasabi<1.1.0,>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from spacy<3.1,>=2.1.0->allennlp)
(0.8.2)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->huggingface-hub>=0.0.8->allennlp) (3.5.0)
Requirement already satisfied: pillow>=5.3.0 in /usr/local/lib/python3.7/dist-
```

```
packages (from torchvision<0.11.0,>=0.8.1->allennlp) (7.1.2)
Requirement already satisfied: tokenizers<0.11,>=0.10.1 in
/usr/local/lib/python3.7/dist-packages (from transformers<4.9,>=4.1->allennlp)
(0.10.3)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.7/dist-packages (from transformers<4.9,>=4.1->allennlp)
(2019.12.20)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.7/dist-packages
(from transformers<4.9,>=4.1->allennlp) (5.4.1)
Requirement already satisfied: sacremoses in /usr/local/lib/python3.7/dist-
packages (from transformers<4.9,>=4.1->allennlp) (0.0.45)
Requirement already satisfied: GitPython>=1.0.0 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(3.1.18)
Requirement already satisfied: docker-pycreds>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(0.4.0)
Requirement already satisfied: configparser>=3.8.1 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(5.0.2)
Requirement already satisfied: Click!=8.0.0,>=7.0 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(7.1.2)
Requirement already satisfied: psutil>=5.0.0 in /usr/local/lib/python3.7/dist-
packages (from wandb<0.12.0,>=0.10.0->allennlp) (5.4.8)
Requirement already satisfied: sentry-sdk>=1.0.0 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(1.3.1)
Requirement already satisfied: subprocess32>=3.5.3 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(3.5.4)
Requirement already satisfied: pathtools in /usr/local/lib/python3.7/dist-
packages (from wandb<0.12.0,>=0.10.0->allennlp) (0.1.2)
Requirement already satisfied: promise<3,>=2.0 in /usr/local/lib/python3.7/dist-
packages (from wandb<0.12.0,>=0.10.0->allennlp) (2.3)
Requirement already satisfied: shortuuid>=0.5.0 in
/usr/local/lib/python3.7/dist-packages (from wandb<0.12.0,>=0.10.0->allennlp)
(1.0.1)
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.7/dist-
packages (from GitPython>=1.0.0->wandb<0.12.0,>=0.10.0->allennlp) (4.0.7)
Requirement already satisfied: smmap<5,>=3.0.1 in /usr/local/lib/python3.7/dist-
packages (from
gitdb<5,>=4.0.1->GitPython>=1.0.0->wandb<0.12.0,>=0.10.0->allennlp) (4.0.0)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages
(from notebook->jupyter>=1.0->checklist==0.0.11->allennlp) (2.11.3)
Requirement already satisfied: terminado>=0.8.1 in
/usr/local/lib/python3.7/dist-packages (from
notebook->jupyter>=1.0->checklist==0.0.11->allennlp) (0.10.1)
```

```
Requirement already satisfied: Send2Trash in /usr/local/lib/python3.7/dist-
packages (from notebook->jupyter>=1.0->checklist==0.0.11->allennlp) (1.7.1)
Requirement already satisfied: pyzmq>=13 in /usr/local/lib/python3.7/dist-
packages (from jupyter-
client->ipykernel>=4.5.1->ipywidgets>=7.5->checklist==0.0.11->allennlp) (22.1.0)
Requirement already satisfied: ptyprocess in /usr/local/lib/python3.7/dist-
packages (from
terminado>=0.8.1->notebook->jupyter>=1.0->checklist==0.0.11->allennlp) (0.7.0)
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-
packages (from h5py->allennlp) (1.5.2)
Requirement already satisfied: MarkupSafe>=0.23 in
/usr/local/lib/python3.7/dist-packages (from
jinja2->notebook->jupyter>=1.0->checklist==0.0.11->allennlp) (2.0.1)
Requirement already satisfied: testpath in /usr/local/lib/python3.7/dist-
packages (from nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (0.5.0)
Requirement already satisfied: entrypoints>=0.2.2 in
/usr/local/lib/python3.7/dist-packages (from
nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (0.3)
Requirement already satisfied: pandocfilters>=1.4.1 in
/usr/local/lib/python3.7/dist-packages (from
nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (1.4.3)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.7/dist-
packages (from nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (0.7.1)
Requirement already satisfied: mistune<2,>=0.8.1 in
/usr/local/lib/python3.7/dist-packages (from
nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (0.8.4)
Requirement already satisfied: bleach in /usr/local/lib/python3.7/dist-packages
(from nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp) (3.3.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.7/dist-
packages (from bleach->nbconvert->jupyter>=1.0->checklist==0.0.11->allennlp)
(0.5.1)
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages
(from patternfork-nosql->checklist==0.0.11->allennlp) (0.16.0)
Collecting backports.csv
  Downloading backports.csv-1.0.7-py2.py3-none-any.whl (12 kB)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.7/dist-
packages (from patternfork-nosql->checklist==0.0.11->allennlp) (4.6.3)
Requirement already satisfied: lxml in /usr/local/lib/python3.7/dist-packages
(from patternfork-nosql->checklist==0.0.11->allennlp) (4.2.6)
Collecting feedparser
  Downloading feedparser-6.0.8-py3-none-any.whl (81 kB)
     |                  | 81 kB 11.0 MB/s
Collecting pdfminer.six
  Downloading pdfminer.six-20201018-py3-none-any.whl (5.6 MB)
     |                  | 5.6 MB 58.6 MB/s
Collecting python-docx
  Downloading python-docx-0.8.11.tar.gz (5.6 MB)
     |                  | 5.6 MB 56.5 MB/s
```

```
Collecting cherrypy
  Downloading CherryPy-18.6.1-py2.py3-none-any.whl (419 kB)
     |                 | 419 kB 48.9 MB/s
Collecting cheroot>=8.2.1
  Downloading cheroot-8.5.2-py2.py3-none-any.whl (97 kB)
     |                 | 97 kB 8.1 MB/s
Collecting jaraco.collections
  Downloading jaraco.collections-3.3.0-py3-none-any.whl (9.9 kB)
Collecting portend>=2.1.1
  Downloading portend-2.7.1-py3-none-any.whl (5.3 kB)
Collecting zc.lockfile
  Downloading zc.lockfile-2.0-py2.py3-none-any.whl (9.7 kB)
Collecting jaraco.functools
  Downloading jaraco.functools-3.3.0-py3-none-any.whl (6.8 kB)
Collecting tempora>=1.8
  Downloading tempora-4.1.1-py3-none-any.whl (15 kB)
Collecting sgmllib3k
  Downloading sgmllib3k-1.0.0.tar.gz (5.8 kB)
Collecting jaraco.classes
  Downloading jaraco.classes-3.2.1-py3-none-any.whl (5.6 kB)
Collecting jaraco.text
  Downloading jaraco.text-3.5.1-py3-none-any.whl (8.1 kB)
Requirement already satisfied: sortedcontainers in
/usr/local/lib/python3.7/dist-packages (from pdfminer.six->patternfork-
nosql->checklist==0.0.11->allennlp) (2.4.0)
Requirement already satisfied: chardet in /usr/local/lib/python3.7/dist-packages
(from pdfminer.six->patternfork-nosql->checklist==0.0.11->allennlp) (3.0.4)
Collecting cryptography
  Downloading cryptography-3.4.7-cp36-abi3-manylinux2014_x86_64.whl (3.2 MB)
     |                 | 3.2 MB 51.1 MB/s
Requirement already satisfied: pluggy<0.8,>=0.5 in
/usr/local/lib/python3.7/dist-packages (from pytest->allennlp) (0.7.1)
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.7/dist-
packages (from pytest->allennlp) (21.2.0)
Requirement already satisfied: py>=1.5.0 in /usr/local/lib/python3.7/dist-
packages (from pytest->allennlp) (1.10.0)
Requirement already satisfied: atomicwrites>=1.0 in
/usr/local/lib/python3.7/dist-packages (from pytest->allennlp) (1.4.0)
Requirement already satisfied: qtpy in /usr/local/lib/python3.7/dist-packages
(from qtconsole->jupyter>=1.0->checklist==0.0.11->allennlp) (1.9.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages
(from sacremoses->transformers<4.9,>=4.1->allennlp) (1.0.1)
Building wheels for collected packages: checklist, overrides, jsonnet, iso-639,
patternfork-nosql, python-docx, sgmllib3k
  Building wheel for checklist (setup.py) … done
  Created wheel for checklist: filename=checklist-0.0.11-py3-none-any.whl
size=12165634
sha256=765491d4ae3fdb36a72cffb0317e3c14cb54120981e7c6aaa2b7fd718d96f021
```

```
  Stored in directory: /root/.cache/pip/wheels/6a/8a/07/6446879be434879c27671c83
443727d74cecf6b630c8a24d03
  Building wheel for overrides (setup.py) … done
  Created wheel for overrides: filename=overrides-3.1.0-py3-none-any.whl
size=10188
sha256=06783c7e5040b140fa1cc683e6902114f7b7b4d3380e4c0530985c00f776b3cf
  Stored in directory: /root/.cache/pip/wheels/3a/0d/38/01a9bc6e20dcfaf0a6a7b552
d03137558ba1c38aea47644682
  Building wheel for jsonnet (setup.py) … done
  Created wheel for jsonnet: filename=jsonnet-0.17.0-cp37-cp37m-linux_x86_64.whl
size=3388670
sha256=8ece4c7815c8e829ab607002b8e0d44a03d98863b79c7bfe7242f820414e5e8b
  Stored in directory: /root/.cache/pip/wheels/1c/28/7e/287c6b19f7161bb03c6986a3
c46b51d0d7d9a1805346634e3a
  Building wheel for iso-639 (setup.py) … done
  Created wheel for iso-639: filename=iso_639-0.4.5-py3-none-any.whl size=169062
sha256=30ca3fb80b4ab22e735af18fa8931eb178760cbd1700f666e1183f940040e5c3
  Stored in directory: /root/.cache/pip/wheels/47/60/19/6d020fc92138ed1b113a1827
1e83ea4b5525fe770cb45b9a2e
  Building wheel for patternfork-nosql (setup.py) … done
  Created wheel for patternfork-nosql: filename=patternfork_nosql-3.6-py3-none-
any.whl size=22332805
sha256=b3dc7f7300dc7e6cb36391018606333f8f7a514020a4c21f7a674dd5f2956c5c
  Stored in directory: /root/.cache/pip/wheels/97/72/8f/5305fe28168f93b658da9ed4
33b9a1d3ec90594faa0c9aaf4b
  Building wheel for python-docx (setup.py) … done
  Created wheel for python-docx: filename=python_docx-0.8.11-py3-none-any.whl
size=184507
sha256=a4dc3df40300d8d001d44921b3e020ad4a17abc84bfae6bafd12f1112a0e6c98
  Stored in directory: /root/.cache/pip/wheels/f6/6f/b9/d798122a8b55b74ad30b5f52
b01482169b445fbb84a11797a6
  Building wheel for sgmllib3k (setup.py) … done
  Created wheel for sgmllib3k: filename=sgmllib3k-1.0.0-py3-none-any.whl
size=6065
sha256=02b9f0bb6671ed289da558267d4a1e93a0b7ea629b8e23df3358871de3f5f79b
  Stored in directory: /root/.cache/pip/wheels/73/ad/a4/0dff4a6ef231fc0dfa12ffba
c2a36cebfdddfe059f50e019aa
Successfully built checklist overrides jsonnet iso-639 patternfork-nosql python-
docx sgmllib3k
Installing collected packages: jaraco.functools, tempora, jaraco.text,
jaraco.classes, zc.lockfile, sgmllib3k, portend, jmespath, jaraco.collections,
cryptography, cheroot, python-docx, pdfminer.six, google-crc32c, feedparser,
cherrypy, botocore, backports.csv, transformers, s3transfer, patternfork-nosql,
munch, iso-639, google-resumable-media, google-cloud-core, tensorboardX,
overrides, jsonnet, google-cloud-storage, checklist, boto3, allennlp
  Attempting uninstall: transformers
    Found existing installation: transformers 4.9.1
    Uninstalling transformers-4.9.1:
```

```
    Successfully uninstalled transformers-4.9.1
  Attempting uninstall: google-resumable-media
    Found existing installation: google-resumable-media 0.4.1
    Uninstalling google-resumable-media-0.4.1:
      Successfully uninstalled google-resumable-media-0.4.1
  Attempting uninstall: google-cloud-core
    Found existing installation: google-cloud-core 1.0.3
    Uninstalling google-cloud-core-1.0.3:
      Successfully uninstalled google-cloud-core-1.0.3
  Attempting uninstall: google-cloud-storage
    Found existing installation: google-cloud-storage 1.18.1
    Uninstalling google-cloud-storage-1.18.1:
      Successfully uninstalled google-cloud-storage-1.18.1
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
google-cloud-bigquery 1.21.0 requires google-resumable-
media!=0.4.0,<0.5.0dev,>=0.3.1, but you have google-resumable-media 1.3.3 which
is incompatible.
Successfully installed allennlp-2.6.0 backports.csv-1.0.7 boto3-1.18.16
botocore-1.21.16 checklist-0.0.11 cheroot-8.5.2 cherrypy-18.6.1
cryptography-3.4.7 feedparser-6.0.8 google-cloud-core-1.7.2 google-cloud-
storage-1.41.1 google-crc32c-1.1.2 google-resumable-media-1.3.3 iso-639-0.4.5
jaraco.classes-3.2.1 jaraco.collections-3.3.0 jaraco.functools-3.3.0
jaraco.text-3.5.1 jmespath-0.10.0 jsonnet-0.17.0 munch-2.5.0 overrides-3.1.0
patternfork-nosql-3.6 pdfminer.six-20201018 portend-2.7.1 python-docx-0.8.11
s3transfer-0.5.0 sgmllib3k-1.0.0 tempora-4.1.1 tensorboardX-2.4
transformers-4.8.2 zc.lockfile-2.0
```

[2]:
```
import torch
import transformers as tfs
import numpy as np
import json
from pathlib import Path
from torch.utils.data import DataLoader
from tqdm.notebook import tqdm
from torch.utils.tensorboard import SummaryWriter
from transformers import BertTokenizerFast,BertPreTrainedModel, BertConfig,␣
 ↪BertModel, BertGenerationEncoder
from transformers import Adafactor, DistilBertTokenizerFast, DistilBertConfig
from transformers.modeling_outputs import (
    BaseModelOutput,
    BaseModelOutputWithPastAndCrossAttentions,
    BaseModelOutputWithPoolingAndCrossAttentions,
    CausalLMOutputWithCrossAttentions,
```

```
    MaskedLMOutput,
    MultipleChoiceModelOutput,
    NextSentencePredictorOutput,
    QuestionAnsweringModelOutput,
    SequenceClassifierOutput,
    TokenClassifierOutput,
)
import string, re
import torch.nn as nn
from allennlp.nn.util import masked_log_softmax, masked_max

import math

from transformers.activations import gelu
from transformers.deepspeed import is_deepspeed_zero3_enabled
from transformers.file_utils import (
    add_code_sample_docstrings,
    add_start_docstrings,
    add_start_docstrings_to_model_forward,
    replace_return_docstrings,
)

from transformers.modeling_utils import (
    PreTrainedModel,
    apply_chunking_to_forward,
    find_pruneable_heads_and_indices,
    prune_linear_layer,
)
from transformers.utils import logging
import torch.nn.functional as F
from torch.nn import Parameter
```

[3]:
```
torch.cuda.is_available()
```

[3]:
```
True
```

1 Utility functions for Metrics evaluation

[4]:
```
# Removing articles and punctuation, and standardizing whitespace are all␣
 ↪typical text processing steps


def normalize_text(s):

    def remove_articles(text):
        regex = re.compile(r"\b(a|an|the)\b", re.UNICODE)
```

```
        return re.sub(regex, " ", text)

    def white_space_fix(text):
        return " ".join(text.split())

    def remove_punc(text):
        exclude = set(string.punctuation)
        return "".join(ch for ch in text if ch not in exclude)

    def lower(text):
        return text.lower()

    return white_space_fix(remove_articles(remove_punc(lower(s))))
```

2 1. Data Understanding

In this section we will import the data & convert it correctly into paralell lists of contexts, questions and answers provided in the SQuAD 2.0 Dataset.

2.1 Download SQuAD 2.0 Data

[5]:
```
# Function to compute the exact match for an answer.
# This will help us determine how accurately do our answers match with the␣
↪suggested answers
def compute_exact_match(prediction, truth):
    return int(normalize_text(prediction) == normalize_text(truth))
```

[6]:
```
# Function to compute the F1 Statistic

def compute_f1(prediction, truth):
    pred_tokens = normalize_text(prediction).split()
    truth_tokens = normalize_text(truth).split()

    # if either the prediction or the truth is no-answer then f1 = 1 if they␣
↪agree, 0 otherwise
    if len(pred_tokens) == 0 or len(truth_tokens) == 0:
        return int(pred_tokens == truth_tokens)

    common_tokens = set(pred_tokens) & set(truth_tokens)

    # if there are no common tokens then f1 = 0
    if len(common_tokens) == 0:
        return 0

    prec = len(common_tokens) / len(pred_tokens)
    rec = len(common_tokens) / len(truth_tokens)
```

```
    return 2 * (prec * rec) / (prec + rec)
```

[7]:
```
# Function to calculate exact match and exact F1 score for a particular␣
↪training epoch
def calculate_stats(input_ids,start,end,idx):
    batch_start = 8*idx
    batch_end = batch_start+8
    data = val_qac[batch_start:batch_end]
    em = 0
    ef1 = 0
    for i,d in enumerate(data):
        answer_start = start[i]
        answer_end = end[i]
        answer = tokenizer.convert_tokens_to_string(tokenizer.
↪convert_ids_to_tokens(input_ids[i][answer_start:answer_end]))
        gold_ans = d['answers']
        if len(gold_ans)==0:
            gold_ans.append("")
        em_s= max((compute_exact_match(answer, g_answer)) for g_answer in␣
↪gold_ans)
        ef1_s = max((compute_f1(answer, g_answer)) for g_answer in gold_ans)
        em+=em_s
        ef1+=ef1_s
    return em,ef1
```

Note : This dataset can be explored in the Hugging Face model hub (SQuAD V2), and can be alternatively downloaded with the NLP library with load_dataset("squad_v2").

[8]:
```
# ## Create a squad directory and download the train and evaluation datasets␣
↪directly into the library
!mkdir squad
!wget https://rajpurkar.github.io/SQuAD-explorer/dataset/train-v2.0.json -O␣
↪squad/train-v2.0.json
!wget https://rajpurkar.github.io/SQuAD-explorer/dataset/dev-v2.0.json -O squad/
↪dev-v2.0.json
```

```
--2021-08-08 11:25:46--  https://rajpurkar.github.io/SQuAD-
explorer/dataset/train-v2.0.json
Resolving rajpurkar.github.io (rajpurkar.github.io)… 185.199.108.153,
185.199.109.153, 185.199.110.153, …
Connecting to rajpurkar.github.io (rajpurkar.github.io)|185.199.108.153|:443…
connected.
HTTP request sent, awaiting response… 200 OK
Length: 42123633 (40M) [application/json]
Saving to: 'squad/train-v2.0.json'

squad/train-v2.0.js 100%[===================>]  40.17M   128MB/s    in 0.3s
```

```
2021-08-08 11:25:47 (128 MB/s) - ‘squad/train-v2.0.json’ saved
[42123633/42123633]

--2021-08-08 11:25:47--  https://rajpurkar.github.io/SQuAD-
explorer/dataset/dev-v2.0.json
Resolving rajpurkar.github.io (rajpurkar.github.io)… 185.199.108.153,
185.199.109.153, 185.199.110.153, …
Connecting to rajpurkar.github.io (rajpurkar.github.io)|185.199.108.153|:443…
connected.
HTTP request sent, awaiting response… 200 OK
Length: 4370528 (4.2M) [application/json]
Saving to: ‘squad/dev-v2.0.json’

squad/dev-v2.0.json 100%[===================>]   4.17M  --.-KB/s    in 0.06s

2021-08-08 11:25:47 (69.0 MB/s) - ‘squad/dev-v2.0.json’ saved [4370528/4370528]
```

Below we will import the data and convert it into parallel lists of contexts, questions, and answers.

[9]:
```
def read_squad(path):
    path = Path(path)
    with open(path, 'rb') as f:
        squad_dict = json.load(f)

    contexts = []
    questions = []
    answers = []
    combined_qac=[] #combined contexts, questions & answers
    counter=0
    for group in squad_dict['data']:
        for passage in group['paragraphs']:
            context = passage['context']
            for qa in passage['qas']:
                question = qa['question']
                q_answers = qa['answers'].copy()
                q_answers = list(map(lambda x:x['text'], q_answers))
                for answer in qa['answers']:
                    contexts.append(context)
                    questions.append(question)
                    answers.append(answer)
                    combined_qac.append({'context':context,'question':
 ↪question,'answers':q_answers})
    return contexts, questions, answers, combined_qac

train_contexts, train_questions, train_answers,train_qac = read_squad('squad/
 ↪train-v2.0.json')
```

```
val_contexts, val_questions, val_answers, val_qac = read_squad('squad/dev-v2.0.
↪json')
```

Now that we have converted the data into parallel lists, let us assess what the dataset holds.

[10]:
```
len(train_contexts)
```

[10]:
```
86821
```

[11]:
```
train_contexts[0]
```

[11]:
```
'Beyoncé Giselle Knowles-Carter (/bi j nse / bee-YON-say) (born September 4,
1981) is an American singer, songwriter, record producer and actress. Born and
raised in Houston, Texas, she performed in various singing and dancing
competitions as a child, and rose to fame in the late 1990s as lead singer of
R&B girl-group Destiny\'s Child. Managed by her father, Mathew Knowles, the
group became one of the world\'s best-selling girl groups of all time. Their
hiatus saw the release of Beyoncé\'s debut album, Dangerously in Love (2003),
which established her as a solo artist worldwide, earned five Grammy Awards and
featured the Billboard Hot 100 number-one singles "Crazy in Love" and "Baby
Boy".'
```

[12]:
```
train_questions[0]
```

[12]:
```
'When did Beyonce start becoming popular?'
```

[13]:
```
train_answers[0]
```

[13]:
```
{'answer_start': 269, 'text': 'in the late 1990s'}
```

[14]:
```
len(train_qac)
```

[14]:
```
86821
```

[15]:
```
train_qac[0]
```

[15]:
```
{'answers': ['in the late 1990s'],
 'context': 'Beyoncé Giselle Knowles-Carter (/bi j nse / bee-YON-say) (born
September 4, 1981) is an American singer, songwriter, record producer and
actress. Born and raised in Houston, Texas, she performed in various singing and
dancing competitions as a child, and rose to fame in the late 1990s as lead
singer of R&B girl-group Destiny\'s Child. Managed by her father, Mathew
Knowles, the group became one of the world\'s best-selling girl groups of all
time. Their hiatus saw the release of Beyoncé\'s debut album, Dangerously in
Love (2003), which established her as a solo artist worldwide, earned five
Grammy Awards and featured the Billboard Hot 100 number-one singles "Crazy in
Love" and "Baby Boy".',
 'question': 'When did Beyonce start becoming popular?'}
```

Inspecting Validation Data

[16]:
```
len(val_contexts)
```

[16]:
```
20302
```

[17]:
```
val_contexts[0]
```

[17]:
```
'The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the
people who in the 10th and 11th centuries gave their name to Normandy, a region
in France. They were descended from Norse ("Norman" comes from "Norseman")
raiders and pirates from Denmark, Iceland and Norway who, under their leader
Rollo, agreed to swear fealty to King Charles III of West Francia. Through
generations of assimilation and mixing with the native Frankish and Roman-
Gaulish populations, their descendants would gradually merge with the
Carolingian-based cultures of West Francia. The distinct cultural and ethnic
identity of the Normans emerged initially in the first half of the 10th century,
and it continued to evolve over the succeeding centuries.'
```

[18]:
```
val_questions[0]
```

[18]:
```
'In what country is Normandy located?'
```

[19]:
```
val_answers[0]
```

[19]:
```
{'answer_start': 159, 'text': 'France'}
```

[20]:
```
val_qac[0]
```

[20]:
```
{'answers': ['France', 'France', 'France', 'France'],
 'context': 'The Normans (Norman: Nourmands; French: Normands; Latin: Normanni)
were the people who in the 10th and 11th centuries gave their name to Normandy,
a region in France. They were descended from Norse ("Norman" comes from
"Norseman") raiders and pirates from Denmark, Iceland and Norway who, under
their leader Rollo, agreed to swear fealty to King Charles III of West Francia.
Through generations of assimilation and mixing with the native Frankish and
Roman-Gaulish populations, their descendants would gradually merge with the
Carolingian-based cultures of West Francia. The distinct cultural and ethnic
identity of the Normans emerged initially in the first half of the 10th century,
and it continued to evolve over the succeeding centuries.',
 'question': 'In what country is Normandy located?'}
```

2.2 Observations:

- We have successfully created 3 subsets of both the training and validation sets
- We gathered the following stats:
  - **Training Data**
    - Length: 86821

    - The combined_qac shows the way things will work, i.e.: We submit a context & a question to the model & receive the answer already highlighted
    - train_answers shows the answer for a particular question and the start index value
- **Validation Data**
    - Length: 20302
    - Similar to the train_qac we have created a val_qac to understand the validation dataset better as well

3 2. Data processing

In this section we will prepare the data appropriately for modelling and training.

We will extract token positions where answers begins & ends for train & validation data.

The contexts and questions are just strings. The answers are dicts containing the subsequence of the passage with the correct answer as well as an integer indicating the character at which the answer begins. In order to train a model on this data we need (1) the tokenized context/question pairs, and (2) integers indicating at which token positions the answer begins and ends.

First, let's get the character position at which the answer ends in the passage (we are given the starting position). Sometimes SQuAD answers are off by one or two characters, so we will also adjust for that.

[21]:
```
## Index the answers and contexts in the training and validation sets. This␣
↪will help us generate the tokens
## and helpp get better answers for our questions
def add_end_idx(answers, contexts):
    for answer, context in zip(answers, contexts):
        gold_text = answer['text']
        start_idx = answer['answer_start']
        end_idx = start_idx + len(gold_text)

        # sometimes squad answers are off by a character or two - fix this
        if context[start_idx:end_idx] == gold_text:
            answer['answer_end'] = end_idx
        elif context[start_idx-1:end_idx-1] == gold_text:
            answer['answer_start'] = start_idx - 1
            answer['answer_end'] = end_idx - 1     # When the gold label is off␣
↪by one character
        elif context[start_idx-2:end_idx-2] == gold_text:
            answer['answer_start'] = start_idx - 2
            answer['answer_end'] = end_idx - 2     # When the gold label is off␣
↪by two characters

add_end_idx(train_answers, train_contexts)
add_end_idx(val_answers, val_contexts)
```

4 Creating Custom BERT Model

[22]:
```
class Encoder(nn.Module):
    """
    Encoder class for Pointer-Net
    """

    def __init__(self, embedding_dim,
                 hidden_dim,
                 n_layers,
                 dropout,
                 bidir):
        """
        Initiate Encoder

        :param Tensor embedding_dim: Number of embbeding channels
        :param int hidden_dim: Number of hidden units for the LSTM
        :param int n_layers: Number of layers for LSTMs
        :param float dropout: Float between 0-1
        :param bool bidir: Bidirectional
        """

        super(Encoder, self).__init__()
        self.hidden_dim = hidden_dim//2 if bidir else hidden_dim
        self.n_layers = n_layers*2 if bidir else n_layers
        self.bidir = bidir
        self.lstm = nn.LSTM(embedding_dim,
                            self.hidden_dim,
                            n_layers,
                            dropout=dropout,
                            bidirectional=bidir)

        # Used for propagating .cuda() command
        self.h0 = Parameter(torch.zeros(1), requires_grad=False)
        self.c0 = Parameter(torch.zeros(1), requires_grad=False)

    def forward(self, embedded_inputs,
                hidden):
        """
        Encoder - Forward-pass

        :param Tensor embedded_inputs: Embedded inputs of Pointer-Net
        :param Tensor hidden: Initiated hidden units for the LSTMs (h, c)
        :return: LSTMs outputs and hidden units (h, c)
        """

        embedded_inputs = embedded_inputs.permute(1, 0, 2)
```

```
        outputs, hidden = self.lstm(embedded_inputs, hidden)

        return outputs.permute(1, 0, 2), hidden

    def init_hidden(self, embedded_inputs):
        """
        Initiate hidden units

        :param Tensor embedded_inputs: The embedded input of Pointer-NEt
        :return: Initiated hidden units for the LSTMs (h, c)
        """

        batch_size = embedded_inputs.size(0)

        # Reshaping (Expanding)
        h0 = self.h0.unsqueeze(0).unsqueeze(0).repeat(self.n_layers,
                                                      batch_size,
                                                      self.hidden_dim)
        c0 = self.h0.unsqueeze(0).unsqueeze(0).repeat(self.n_layers,
                                                      batch_size,
                                                      self.hidden_dim)

        return h0, c0


class Attention(nn.Module):
    """
    Attention model for Pointer-Net
    """

    def __init__(self, input_dim,
                 hidden_dim):
        """
        Initiate Attention

        :param int input_dim: Input's diamention
        :param int hidden_dim: Number of hidden units in the attention
        """

        super(Attention, self).__init__()

        self.input_dim = input_dim
        self.hidden_dim = hidden_dim

        self.input_linear = nn.Linear(input_dim, hidden_dim)
        self.context_linear = nn.Conv1d(input_dim, hidden_dim, 1, 1)
```

```
        self.V = Parameter(torch.FloatTensor(hidden_dim), requires_grad=True)
        self._inf = Parameter(torch.FloatTensor([float('-inf')]),␣
↪requires_grad=False)
        self.tanh = nn.Tanh()
        self.softmax = nn.Softmax()

        # Initialize vector V
        nn.init.uniform(self.V, -1, 1)

    def forward(self, input,
                context,
                mask):
        """
        Attention - Forward-pass

        :param Tensor input: Hidden state h
        :param Tensor context: Attention context
        :param ByteTensor mask: Selection mask
        :return: tuple of - (Attentioned hidden state, Alphas)
        """

        # (batch, hidden_dim, seq_len)
        inp = self.input_linear(input).unsqueeze(2).expand(-1, -1, context.
↪size(1))

        # (batch, hidden_dim, seq_len)
        context = context.permute(0, 2, 1)
        ctx = self.context_linear(context)

        # (batch, 1, hidden_dim)
        V = self.V.unsqueeze(0).expand(context.size(0), -1).unsqueeze(1)

        # (batch, seq_len)
        att = torch.bmm(V, self.tanh(inp + ctx)).squeeze(1)
        if len(att[mask]) > 0:
            att[mask] = self.inf[mask]
        alpha = self.softmax(att)

        hidden_state = torch.bmm(ctx, alpha.unsqueeze(2)).squeeze(2)

        return hidden_state, alpha

    def init_inf(self, mask_size):
        self.inf = self._inf.unsqueeze(1).expand(*mask_size)


class Decoder(nn.Module):
```

```
    """
    Decoder model for Pointer-Net
    """

    def __init__(self, embedding_dim,
                 hidden_dim):
        """
        Initiate Decoder

        :param int embedding_dim: Number of embeddings in Pointer-Net
        :param int hidden_dim: Number of hidden units for the decoder's RNN
        """

        super(Decoder, self).__init__()
        self.embedding_dim = embedding_dim
        self.hidden_dim = hidden_dim

        self.input_to_hidden = nn.Linear(embedding_dim, 4 * hidden_dim)
        self.hidden_to_hidden = nn.Linear(hidden_dim, 4 * hidden_dim)
        self.hidden_out = nn.Linear(hidden_dim * 2, hidden_dim)
        self.att = Attention(hidden_dim, hidden_dim)

        # Used for propagating .cuda() command
        self.mask = Parameter(torch.ones(1), requires_grad=False)
        self.runner = Parameter(torch.zeros(1), requires_grad=False)

    def forward(self, embedded_inputs,
                decoder_input,
                hidden,
                context):
        """
        Decoder - Forward-pass

        :param Tensor embedded_inputs: Embedded inputs of Pointer-Net
        :param Tensor decoder_input: First decoder's input
        :param Tensor hidden: First decoder's hidden states
        :param Tensor context: Encoder's outputs
        :return: (Output probabilities, Pointers indices), last hidden state
        """

        batch_size = embedded_inputs.size(0)
        input_length = embedded_inputs.size(1)

        # (batch, seq_len)
        mask = self.mask.repeat(input_length).unsqueeze(0).repeat(batch_size, 1)
        self.att.init_inf(mask.size())
```

```
        # Generating arang(input_length), broadcasted across batch_size
        runner = self.runner.repeat(input_length)
        for i in range(input_length):
            runner.data[i] = i
        runner = runner.unsqueeze(0).expand(batch_size, -1).long()

        outputs = []
        pointers = []

        def step(x, hidden):
            """
            Recurrence step function

            :param Tensor x: Input at time t
            :param tuple(Tensor, Tensor) hidden: Hidden states at time t-1
            :return: Hidden states at time t (h, c), Attention probabilities␣
↪(Alpha)
            """

            # Regular LSTM
            h, c = hidden

            gates = self.input_to_hidden(x) + self.hidden_to_hidden(h)
            input, forget, cell, out = gates.chunk(4, 1)

            input = torch.sigmoid(input)
            forget = torch.sigmoid(forget)
            cell = torch.tanh(cell)
            out = torch.sigmoid(out)

            c_t = (forget * c) + (input * cell)
            h_t = out * torch.tanh(c_t)

            # Attention section
            hidden_t, output = self.att(h_t, context, torch.eq(mask, 0))
            hidden_t = torch.tanh(self.hidden_out(torch.cat((hidden_t, h_t),␣
↪1)))

            return hidden_t, c_t, output

        # Recurrence loop
        for _ in range(input_length):
            h_t, c_t, outs = step(decoder_input, hidden)
            hidden = (h_t, c_t)

            # Masking selected inputs
            masked_outs = outs * mask
```

```
            # Get maximum probabilities and indices
            max_probs, indices = masked_outs.max(1)
            one_hot_pointers = (runner == indices.unsqueeze(1).expand(-1, outs.
 ↪size()[1])).float()

            # Update mask to ignore seen indices
            mask  = mask * (1 - one_hot_pointers)

            # Get embedded inputs by max indices
            embedding_mask = one_hot_pointers.unsqueeze(2).expand(-1, -1, self.
 ↪embedding_dim).byte()
            decoder_input = embedded_inputs[embedding_mask.data].
 ↪view(batch_size, self.embedding_dim)

            outputs.append(outs.unsqueeze(0))
            pointers.append(indices.unsqueeze(1))

        outputs = torch.cat(outputs).permute(1, 0, 2)
        pointers = torch.cat(pointers, 1)

        return (outputs, pointers), hidden


class PointerNet(nn.Module):
    """
    Pointer-Net
    """

    def __init__(self, embedding_dim,
                 hidden_dim,
                 lstm_layers,
                 dropout,
                 bidir=False):
        """
        Initiate Pointer-Net

        :param int embedding_dim: Number of embbeding channels
        :param int hidden_dim: Encoders hidden units
        :param int lstm_layers: Number of layers for LSTMs
        :param float dropout: Float between 0-1
        :param bool bidir: Bidirectional
        """

        super(PointerNet, self).__init__()
        self.embedding_dim = embedding_dim
        self.bidir = bidir
```

```
        self.embedding = nn.Linear(2, embedding_dim)
        self.encoder = Encoder(embedding_dim,
                               hidden_dim,
                               lstm_layers,
                               dropout,
                               bidir)
        self.decoder = Decoder(embedding_dim, hidden_dim)
        self.decoder_input0 = Parameter(torch.FloatTensor(embedding_dim),␣
↪requires_grad=False)

        # Initialize decoder_input0
        nn.init.uniform(self.decoder_input0, -1, 1)

    def forward(self, inputs):
        """
        PointerNet - Forward-pass

        :param Tensor inputs: Input sequence
        :return: Pointers probabilities and indices
        """

        batch_size = inputs.size(0)
        input_length = inputs.size(1)

        decoder_input0 = self.decoder_input0.unsqueeze(0).expand(batch_size, -1)

        inputs = inputs.view(batch_size * input_length, -1)
        embedded_inputs = self.embedding(inputs).view(batch_size, input_length,␣
↪-1)

        encoder_hidden0 = self.encoder.init_hidden(embedded_inputs)
        encoder_outputs, encoder_hidden = self.encoder(embedded_inputs,
                                                       encoder_hidden0)
        if self.bidir:
            decoder_hidden0 = (torch.cat(encoder_hidden[0][-2:], dim=-1),
                               torch.cat(encoder_hidden[1][-2:], dim=-1))
        else:
            decoder_hidden0 = (encoder_hidden[0][-1],
                               encoder_hidden[1][-1])
        (outputs, pointers), decoder_hidden = self.decoder(embedded_inputs,
                                                           decoder_input0,
                                                           decoder_hidden0,
                                                           encoder_outputs)

        return  outputs, pointers
```

[23]:
```
# coding=utf-8
# Copyright 2019-present, the HuggingFace Inc. team, The Google AI Language␣
 ↪Team and Facebook, Inc.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
"""
 PyTorch DistilBERT model adapted in part from Facebook, Inc XLM model (https://
 ↪github.com/facebookresearch/XLM) and in
 part from HuggingFace PyTorch version of Google AI Bert model (https://github.
 ↪com/google-research/bert)
"""

logger = logging.get_logger(__name__)
_CHECKPOINT_FOR_DOC = "distilbert-base-uncased"
_CONFIG_FOR_DOC = "DistilBertConfig"
_TOKENIZER_FOR_DOC = "DistilBertTokenizer"

DISTILBERT_PRETRAINED_MODEL_ARCHIVE_LIST = [
    "distilbert-base-uncased",
    "distilbert-base-uncased-distilled-squad",
    "distilbert-base-cased",
    "distilbert-base-cased-distilled-squad",
    "distilbert-base-german-cased",
    "distilbert-base-multilingual-cased",
    "distilbert-base-uncased-finetuned-sst-2-english",
    # See all DistilBERT models at https://huggingface.co/models?
 ↪filter=distilbert
]


# UTILS AND BUILDING BLOCKS OF THE ARCHITECTURE #


def create_sinusoidal_embeddings(n_pos, dim, out):
    position_enc = np.array([[pos / np.power(10000, 2 * (j // 2) / dim) for j␣
 ↪in range(dim)] for pos in range(n_pos)])
    out.requires_grad = False
```

```
    out[:, 0::2] = torch.FloatTensor(np.sin(position_enc[:, 0::2]))
    out[:, 1::2] = torch.FloatTensor(np.cos(position_enc[:, 1::2]))
    out.detach_()


class Embeddings(nn.Module):
    def __init__(self, config):
        super().__init__()
        self.word_embeddings = nn.Embedding(config.vocab_size, config.dim,␣
↪padding_idx=config.pad_token_id)
        self.position_embeddings = nn.Embedding(config.max_position_embeddings,␣
↪config.dim)
        if config.sinusoidal_pos_embds:

            if is_deepspeed_zero3_enabled():
                import deepspeed

                with deepspeed.zero.GatheredParameters(self.position_embeddings.
↪weight, modifier_rank=0):
                    if torch.distributed.get_rank() == 0:
                        create_sinusoidal_embeddings(
                            n_pos=config.max_position_embeddings, dim=config.
↪dim, out=self.position_embeddings.weight
                        )
            else:
                create_sinusoidal_embeddings(
                    n_pos=config.max_position_embeddings, dim=config.dim,␣
↪out=self.position_embeddings.weight
                )

        self.LayerNorm = nn.LayerNorm(config.dim, eps=1e-12)
        self.dropout = nn.Dropout(config.dropout)

    def forward(self, input_ids):
        """
        Parameters:
            input_ids: torch.tensor(bs, max_seq_length) The token ids to embed.

        Returns: torch.tensor(bs, max_seq_length, dim) The embedded tokens␣
↪(plus position embeddings, no token_type
        embeddings)
        """
        seq_length = input_ids.size(1)
        position_ids = torch.arange(seq_length, dtype=torch.long,␣
↪device=input_ids.device)  # (max_seq_length)
```

```
        position_ids = position_ids.unsqueeze(0).expand_as(input_ids)  # (bs,␣
↪max_seq_length)

        word_embeddings = self.word_embeddings(input_ids)  # (bs,␣
↪max_seq_length, dim)
        position_embeddings = self.position_embeddings(position_ids)  # (bs,␣
↪max_seq_length, dim)

        embeddings = word_embeddings + position_embeddings  # (bs,␣
↪max_seq_length, dim)
        embeddings = self.LayerNorm(embeddings)  # (bs, max_seq_length, dim)
        embeddings = self.dropout(embeddings)  # (bs, max_seq_length, dim)
        return embeddings


class MultiHeadSelfAttention(nn.Module):
    def __init__(self, config):
        super().__init__()

        self.n_heads = config.n_heads
        self.dim = config.dim
        self.dropout = nn.Dropout(p=config.attention_dropout)

        assert self.dim % self.n_heads == 0

        self.q_lin = nn.Linear(in_features=config.dim, out_features=config.dim)
        self.k_lin = nn.Linear(in_features=config.dim, out_features=config.dim)
        self.v_lin = nn.Linear(in_features=config.dim, out_features=config.dim)
        self.out_lin = nn.Linear(in_features=config.dim, out_features=config.
↪dim)

        self.pruned_heads = set()

    def prune_heads(self, heads):
        attention_head_size = self.dim // self.n_heads
        if len(heads) == 0:
            return
        heads, index = find_pruneable_heads_and_indices(heads, self.n_heads,␣
↪attention_head_size, self.pruned_heads)
        # Prune linear layers
        self.q_lin = prune_linear_layer(self.q_lin, index)
        self.k_lin = prune_linear_layer(self.k_lin, index)
        self.v_lin = prune_linear_layer(self.v_lin, index)
        self.out_lin = prune_linear_layer(self.out_lin, index, dim=1)
        # Update hyper params
        self.n_heads = self.n_heads - len(heads)
```

```
        self.dim = attention_head_size * self.n_heads
        self.pruned_heads = self.pruned_heads.union(heads)

    def forward(self, query, key, value, mask, head_mask=None,␣
↪output_attentions=False):
        """
        Parameters:
            query: torch.tensor(bs, seq_length, dim)
            key: torch.tensor(bs, seq_length, dim)
            value: torch.tensor(bs, seq_length, dim)
            mask: torch.tensor(bs, seq_length)

        Returns:
            weights: torch.tensor(bs, n_heads, seq_length, seq_length)␣
↪Attention weights context: torch.tensor(bs,
            seq_length, dim) Contextualized layer. Optional: only if␣
↪`output_attentions=True`
        """
        bs, q_length, dim = query.size()
        k_length = key.size(1)
        # assert dim == self.dim, f'Dimensions do not match: {dim} input vs␣
↪{self.dim} configured'
        # assert key.size() == value.size()

        dim_per_head = self.dim // self.n_heads

        mask_reshp = (bs, 1, 1, k_length)

        def shape(x):
            """separate heads"""
            return x.view(bs, -1, self.n_heads, dim_per_head).transpose(1, 2)

        def unshape(x):
            """group heads"""
            return x.transpose(1, 2).contiguous().view(bs, -1, self.n_heads *␣
↪dim_per_head)

        q = shape(self.q_lin(query))  # (bs, n_heads, q_length, dim_per_head)
        k = shape(self.k_lin(key))  # (bs, n_heads, k_length, dim_per_head)
        v = shape(self.v_lin(value))  # (bs, n_heads, k_length, dim_per_head)

        q = q / math.sqrt(dim_per_head)  # (bs, n_heads, q_length, dim_per_head)
        scores = torch.matmul(q, k.transpose(2, 3))  # (bs, n_heads, q_length,␣
↪k_length)
        mask = (mask == 0).view(mask_reshp).expand_as(scores)  # (bs, n_heads,␣
↪q_length, k_length)
```

```
        scores.masked_fill_(mask, -float("inf"))  # (bs, n_heads, q_length,␣
↪k_length)

        weights = nn.Softmax(dim=-1)(scores)  # (bs, n_heads, q_length,␣
↪k_length)
        weights = self.dropout(weights)  # (bs, n_heads, q_length, k_length)

        # Mask heads if we want to
        if head_mask is not None:
            weights = weights * head_mask

        context = torch.matmul(weights, v)  # (bs, n_heads, q_length,␣
↪dim_per_head)
        context = unshape(context)  # (bs, q_length, dim)
        context = self.out_lin(context)  # (bs, q_length, dim)

        if output_attentions:
            return (context, weights)
        else:
            return (context,)


class FFN(nn.Module):
    def __init__(self, config):
        super().__init__()
        self.dropout = nn.Dropout(p=config.dropout)
        self.chunk_size_feed_forward = config.chunk_size_feed_forward
        self.seq_len_dim = 1
        self.lin1 = nn.Linear(in_features=config.dim, out_features=config.
↪hidden_dim)
        self.lin2 = nn.Linear(in_features=config.hidden_dim,␣
↪out_features=config.dim)
        assert config.activation in ["relu", "gelu"], f"activation ({config.
↪activation}) must be in ['relu', 'gelu']"
        self.activation = gelu if config.activation == "gelu" else nn.ReLU()

    def forward(self, input):
        return apply_chunking_to_forward(self.ff_chunk, self.
↪chunk_size_feed_forward, self.seq_len_dim, input)

    def ff_chunk(self, input):
        x = self.lin1(input)
        x = self.activation(x)
        x = self.lin2(x)
        x = self.dropout(x)
        return x
```

```
class TransformerBlock(nn.Module):
    def __init__(self, config):
        super().__init__()

        assert config.dim % config.n_heads == 0

        self.attention = MultiHeadSelfAttention(config)
        self.sa_layer_norm = nn.LayerNorm(normalized_shape=config.dim,␣
↪eps=1e-12)

        self.ffn = FFN(config)
        self.output_layer_norm = nn.LayerNorm(normalized_shape=config.dim,␣
↪eps=1e-12)

    def forward(self, x, attn_mask=None, head_mask=None,␣
↪output_attentions=False):
        """
        Parameters:
            x: torch.tensor(bs, seq_length, dim)
            attn_mask: torch.tensor(bs, seq_length)

        Returns:
            sa_weights: torch.tensor(bs, n_heads, seq_length, seq_length) The␣
↪attention weights ffn_output:
            torch.tensor(bs, seq_length, dim) The output of the transformer␣
↪block contextualization.
        """
        # Self-Attention
        sa_output = self.attention(
            query=x,
            key=x,
            value=x,
            mask=attn_mask,
            head_mask=head_mask,
            output_attentions=output_attentions,
        )
        if output_attentions:
            sa_output, sa_weights = sa_output  # (bs, seq_length, dim), (bs,␣
↪n_heads, seq_length, seq_length)
        else:  # To handle these `output_attentions` or `output_hidden_states`␣
↪cases returning tuples
            assert type(sa_output) == tuple
            sa_output = sa_output[0]
        sa_output = self.sa_layer_norm(sa_output + x)  # (bs, seq_length, dim)
```

```
        # Feed Forward Network
        ffn_output = self.ffn(sa_output)  # (bs, seq_length, dim)
        ffn_output = self.output_layer_norm(ffn_output + sa_output)  # (bs,␣
↪seq_length, dim)

        output = (ffn_output,)
        if output_attentions:
            output = (sa_weights,) + output
        return output


class Transformer(nn.Module):
    def __init__(self, config):
        super().__init__()
        self.n_layers = config.n_layers
        self.layer = nn.ModuleList([TransformerBlock(config) for _ in␣
↪range(config.n_layers)])

    def forward(
        self, x, attn_mask=None, head_mask=None, output_attentions=False,␣
↪output_hidden_states=False, return_dict=None
    ):  # docstyle-ignore
        """
        Parameters:
            x: torch.tensor(bs, seq_length, dim) Input sequence embedded.
            attn_mask: torch.tensor(bs, seq_length) Attention mask on the␣
↪sequence.

        Returns:
            hidden_state: torch.tensor(bs, seq_length, dim) Sequence of hidden␣
↪states in the last (top)
            layer all_hidden_states: Tuple[torch.tensor(bs, seq_length, dim)]
                Tuple of length n_layers with the hidden states from each layer.
                Optional: only if output_hidden_states=True
            all_attentions: Tuple[torch.tensor(bs, n_heads, seq_length,␣
↪seq_length)]
                Tuple of length n_layers with the attention weights from each␣
↪layer
                Optional: only if output_attentions=True
        """
        all_hidden_states = () if output_hidden_states else None
        all_attentions = () if output_attentions else None

        hidden_state = x
        for i, layer_module in enumerate(self.layer):
```

```
            if output_hidden_states:
                all_hidden_states = all_hidden_states + (hidden_state,)

            layer_outputs = layer_module(
                x=hidden_state, attn_mask=attn_mask, head_mask=head_mask[i],␣
↪output_attentions=output_attentions
            )
            hidden_state = layer_outputs[-1]

            if output_attentions:
                assert len(layer_outputs) == 2
                attentions = layer_outputs[0]
                all_attentions = all_attentions + (attentions,)
            else:
                assert len(layer_outputs) == 1

        # Add last layer
        if output_hidden_states:
            all_hidden_states = all_hidden_states + (hidden_state,)

        if not return_dict:
            return tuple(v for v in [hidden_state, all_hidden_states,␣
↪all_attentions] if v is not None)
        return BaseModelOutput(
            last_hidden_state=hidden_state, hidden_states=all_hidden_states,␣
↪attentions=all_attentions
        )


# INTERFACE FOR ENCODER AND TASK SPECIFIC MODEL #
class DistilBertPreTrainedModel(PreTrainedModel):
    """
    An abstract class to handle weights initialization and a simple interface␣
↪for downloading and loading pretrained
    models.
    """

    config_class = DistilBertConfig
    load_tf_weights = None
    base_model_prefix = "distilbert"

    def _init_weights(self, module):
        """Initialize the weights."""
        if isinstance(module, nn.Linear):
            # Slightly different from the TF version which uses␣
↪truncated_normal for initialization
            # cf https://github.com/pytorch/pytorch/pull/5617
```

```
            module.weight.data.normal_(mean=0.0, std=self.config.
 ↪initializer_range)
            if module.bias is not None:
                module.bias.data.zero_()
        elif isinstance(module, nn.Embedding):
            module.weight.data.normal_(mean=0.0, std=self.config.
 ↪initializer_range)
            if module.padding_idx is not None:
                module.weight.data[module.padding_idx].zero_()
        elif isinstance(module, nn.LayerNorm):
            module.bias.data.zero_()
            module.weight.data.fill_(1.0)


DISTILBERT_START_DOCSTRING = r"""

    This model inherits from :class:`~transformers.PreTrainedModel`. Check the␣
 ↪superclass documentation for the generic
    methods the library implements for all its model (such as downloading or␣
 ↪saving, resizing the input embeddings,
    pruning heads etc.)

    This model is also a PyTorch `torch.nn.Module <https://pytorch.org/docs/
 ↪stable/nn.html#torch.nn.Module>`__
    subclass. Use it as a regular PyTorch Module and refer to the PyTorch␣
 ↪documentation for all matter related to
    general usage and behavior.

    Parameters:
        config (:class:`~transformers.DistilBertConfig`): Model configuration␣
 ↪class with all the parameters of the model.
            Initializing with a config file does not load the weights␣
 ↪associated with the model, only the
            configuration. Check out the :meth:`~transformers.PreTrainedModel.
 ↪from_pretrained` method to load the model
            weights.
"""

DISTILBERT_INPUTS_DOCSTRING = r"""
    Args:
        input_ids (:obj:`torch.LongTensor` of shape :obj:`({0})`):
            Indices of input sequence tokens in the vocabulary.

            Indices can be obtained using :class:`~transformers.
 ↪DistilBertTokenizer`. See
```

```
            :meth:`transformers.PreTrainedTokenizer.encode` and :meth:
↪`transformers.PreTrainedTokenizer.__call__` for
            details.

            `What are input IDs? <../glossary.html#input-ids>`__
        attention_mask (:obj:`torch.FloatTensor` of shape :obj:`({0})`,␣
↪`optional`):
            Mask to avoid performing attention on padding token indices. Mask␣
↪values selected in ``[0, 1]``:

            - 1 for tokens that are **not masked**,
            - 0 for tokens that are **masked**.

            `What are attention masks? <../glossary.html#attention-mask>`__
        head_mask (:obj:`torch.FloatTensor` of shape :obj:`(num_heads,)` or :
↪obj:`(num_layers, num_heads)`, `optional`):
            Mask to nullify selected heads of the self-attention modules. Mask␣
↪values selected in ``[0, 1]``:

            - 1 indicates the head is **not masked**,
            - 0 indicates the head is **masked**.

        inputs_embeds (:obj:`torch.FloatTensor` of shape :obj:`({0},␣
↪hidden_size)`, `optional`):
            Optionally, instead of passing :obj:`input_ids` you can choose to␣
↪directly pass an embedded representation.
            This is useful if you want more control over how to convert :obj:
↪`input_ids` indices into associated
            vectors than the model's internal embedding lookup matrix.
        output_attentions (:obj:`bool`, `optional`):
            Whether or not to return the attentions tensors of all attention␣
↪layers. See ``attentions`` under returned
            tensors for more detail.
        output_hidden_states (:obj:`bool`, `optional`):
            Whether or not to return the hidden states of all layers. See␣
↪``hidden_states`` under returned tensors for
            more detail.
        return_dict (:obj:`bool`, `optional`):
            Whether or not to return a :class:`~transformers.file_utils.
↪ModelOutput` instead of a plain tuple.
"""



class DistilBertModel(DistilBertPreTrainedModel):
    def __init__(self, config):
```

```
        super().__init__(config)

        self.hidden_size = config.hidden_dim
        self.num_layers = 1
        self.num_directions = 2 #because we are implementing it in␣
↪bidirectional lstm mode

        self.embeddings = Embeddings(config)  # Embeddings
        self.transformer = Transformer(config)  # Encoder
        self.pointer = PointerNet(1,8,1,0.0,False)

        self.init_weights()

    def get_input_embeddings(self):
        return self.embeddings.word_embeddings

    def set_input_embeddings(self, new_embeddings):
        self.embeddings.word_embeddings = new_embeddings

    def _prune_heads(self, heads_to_prune):
        """
        Prunes heads of the model. heads_to_prune: dict of {layer_num: list of␣
↪heads to prune in this layer} See base
        class PreTrainedModel
        """
        for layer, heads in heads_to_prune.items():
            self.transformer.layer[layer].attention.prune_heads(heads)

    def forward(
        self,
        input_ids=None,
        attention_mask=None,
        head_mask=None,
        inputs_embeds=None,
        output_attentions=None,
        output_hidden_states=None,
        return_dict=None,
    ):
        output_attentions = output_attentions if output_attentions is not None␣
↪else self.config.output_attentions
        output_hidden_states = (
            output_hidden_states if output_hidden_states is not None else self.
↪config.output_hidden_states
        )
        return_dict = return_dict if return_dict is not None else self.config.
↪use_return_dict
```

```
        if input_ids is not None and inputs_embeds is not None:
            raise ValueError("You cannot specify both input_ids and␣
↪inputs_embeds at the same time")
        elif input_ids is not None:
            input_shape = input_ids.size()
        elif inputs_embeds is not None:
            input_shape = inputs_embeds.size()[:-1]
        else:
            raise ValueError("You have to specify either input_ids or␣
↪inputs_embeds")

        device = input_ids.device if input_ids is not None else inputs_embeds.
↪device

        if attention_mask is None:
            attention_mask = torch.ones(input_shape, device=device)  # (bs,␣
↪seq_length)

        # Prepare head mask if needed
        head_mask = self.get_head_mask(head_mask, self.config.num_hidden_layers)

        if inputs_embeds is None:
            inputs_embeds = self.embeddings(input_ids)  # (bs, seq_length, dim)


        inputs_embed, ots=self.pointer(inputs_embeds.type(torch.float))
        return self.transformer(
            x=inputs_embeds,
            attn_mask=attention_mask,
            head_mask=head_mask,
            output_attentions=output_attentions,
            output_hidden_states=output_hidden_states,
            return_dict=return_dict,
        )


class DistilBertForQuestionAnsweringC(DistilBertPreTrainedModel):
    def __init__(self, config):
        super().__init__(config)

        self.distilbert = DistilBertModel(config)
        self.qa_outputs = nn.Linear(config.dim, config.num_labels)
        assert config.num_labels == 2
        self.dropout = nn.Dropout(config.qa_dropout)

        self.init_weights()
```

```
    def forward(
        self,
        input_ids=None,
        attention_mask=None,
        head_mask=None,
        inputs_embeds=None,
        start_positions=None,
        end_positions=None,
        output_attentions=None,
        output_hidden_states=None,
        return_dict=None,
    ):
        r"""
        start_positions (:obj:`torch.LongTensor` of shape :obj:`(batch_size,)`,␣
↪`optional`):
            Labels for position (index) of the start of the labelled span for␣
↪computing the token classification loss.
            Positions are clamped to the length of the sequence (:obj:
↪`sequence_length`). Position outside of the
            sequence are not taken into account for computing the loss.
        end_positions (:obj:`torch.LongTensor` of shape :obj:`(batch_size,)`,␣
↪`optional`):
            Labels for position (index) of the end of the labelled span for␣
↪computing the token classification loss.
            Positions are clamped to the length of the sequence (:obj:
↪`sequence_length`). Position outside of the
            sequence are not taken into account for computing the loss.
        """
        return_dict = return_dict if return_dict is not None else self.config.
↪use_return_dict


        distilbert_output = self.distilbert(
            input_ids=input_ids,
            attention_mask=attention_mask,
            head_mask=head_mask,
            inputs_embeds=inputs_embeds,
            output_attentions=output_attentions,
            output_hidden_states=output_hidden_states,
            return_dict=return_dict,
        )


        hidden_states = distilbert_output[0]  # (bs, max_query_len, dim)

        hidden_states = self.dropout(hidden_states)  # (bs, max_query_len, dim)
```

```
        logits = self.qa_outputs(hidden_states)  # (bs, max_query_len, 2)
        start_logits, end_logits = logits.split(1, dim=-1)
        start_logits = start_logits.squeeze(-1).contiguous()  # (bs,␣
↪max_query_len)
        end_logits = end_logits.squeeze(-1).contiguous()  # (bs, max_query_len)

        total_loss = None
        if start_positions is not None and end_positions is not None:
            # If we are on multi-GPU, split add a dimension
            if len(start_positions.size()) > 1:
                start_positions = start_positions.squeeze(-1)
            if len(end_positions.size()) > 1:
                end_positions = end_positions.squeeze(-1)
            # sometimes the start/end positions are outside our model inputs,␣
↪we ignore these terms
            ignored_index = start_logits.size(1)
            start_positions = start_positions.clamp(0, ignored_index)
            end_positions = end_positions.clamp(0, ignored_index)

            loss_fct = nn.CrossEntropyLoss(ignore_index=ignored_index)
            start_loss = loss_fct(start_logits, start_positions)
            end_loss = loss_fct(end_logits, end_positions)
            total_loss = (start_loss + end_loss) / 2

        if not return_dict:
            output = (start_logits, end_logits) + distilbert_output[1:]
            return ((total_loss,) + output) if total_loss is not None else␣
↪output

        return QuestionAnsweringModelOutput(
            loss=total_loss,
            start_logits=start_logits,
            end_logits=end_logits,
            hidden_states=distilbert_output.hidden_states,
            attentions=distilbert_output.attentions,
        )
```

[24]:
```
## Initialize a tokenizer using DistilBERT which will help us tokenize our␣
↪training questions and answers

tokenizer = DistilBertTokenizerFast.from_pretrained('distilbert-base-uncased')

## obtain encoded training and validation sets from the tokenizer
train_encodings = tokenizer(train_contexts, train_questions, truncation=True,␣
↪padding=True)
val_encodings = tokenizer(val_contexts, val_questions, truncation=True,␣
↪padding=True)
```

```
HBox(children=(FloatProgress(value=0.0, description='Downloading', max=231508.0,␣
 ↪style=ProgressStyle(descripti…
```

```
HBox(children=(FloatProgress(value=0.0, description='Downloading', max=466062.0,␣
 ↪style=ProgressStyle(descripti…
```

```
HBox(children=(FloatProgress(value=0.0, description='Downloading', max=28.0,␣
 ↪style=ProgressStyle(description_w…
```

observation

[25]:
```
## Create a function to add token positions
def add_token_positions(encodings, answers):
    start_positions = []
    end_positions = []
    for i in range(len(answers)):
        start_positions.append(encodings.char_to_token(i,␣
 ↪answers[i]['answer_start']))
        end_positions.append(encodings.char_to_token(i,␣
 ↪answers[i]['answer_end'] - 1))
        # if None, the answer passage has been truncated
        if start_positions[-1] is None:
            start_positions[-1] = tokenizer.model_max_length
        if end_positions[-1] is None:
            end_positions[-1] = tokenizer.model_max_length
    encodings.update({'start_positions': start_positions, 'end_positions':␣
 ↪end_positions})

add_token_positions(train_encodings, train_answers)
add_token_positions(val_encodings, val_answers)
```

5 3. Train & Validation Dataset Creation

[26]:
```
## Creating the taining and validation datasets using the encoded training and␣
 ↪validation sets we created in
## the section above
class SquadDataset(torch.utils.data.Dataset):
    def __init__(self, encodings):
        self.encodings = encodings

    def __getitem__(self, idx):
        return {key: torch.tensor(val[idx]) for key, val in self.encodings.
 ↪items()}
```

```
    def __len__(self):
        return len(self.encodings.input_ids)

train_dataset = SquadDataset(train_encodings)
val_dataset = SquadDataset(val_encodings)
```

5.0.1 Observations

6 4. Model Building & Training

[27]:
```
model =DistilBertForQuestionAnsweringC.
 ↪from_pretrained('distilbert-base-uncased')
```

```
HBox(children=(FloatProgress(value=0.0, description='Downloading', max=442.0,␣
 ↪style=ProgressStyle(description_…
```

```
HBox(children=(FloatProgress(value=0.0, description='Downloading', max=267967963.
 ↪0, style=ProgressStyle(descri…
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:99: UserWarning:
nn.init.uniform is now deprecated in favor of nn.init.uniform_.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:284: UserWarning:
nn.init.uniform is now deprecated in favor of nn.init.uniform_.
Some weights of the model checkpoint at distilbert-base-uncased were not used
when initializing DistilBertForQuestionAnsweringC: ['vocab_transform.weight',
'vocab_layer_norm.weight', 'vocab_transform.bias', 'vocab_layer_norm.bias',
'vocab_projector.bias', 'vocab_projector.weight']
- This IS expected if you are initializing DistilBertForQuestionAnsweringC from
the checkpoint of a model trained on another task or with another architecture
(e.g. initializing a BertForSequenceClassification model from a
BertForPreTraining model).
- This IS NOT expected if you are initializing DistilBertForQuestionAnsweringC
from the checkpoint of a model that you expect to be exactly identical
(initializing a BertForSequenceClassification model from a
BertForSequenceClassification model).
Some weights of DistilBertForQuestionAnsweringC were not initialized from the
model checkpoint at distilbert-base-uncased and are newly initialized:
['distilbert.pointer.decoder_input0',
'distilbert.pointer.decoder.att.input_linear.bias',
'distilbert.pointer.encoder.h0', 'distilbert.pointer.decoder.att.V',
'distilbert.pointer.decoder.hidden_to_hidden.weight',
'distilbert.pointer.decoder.att._inf',
'distilbert.pointer.decoder.hidden_out.weight',
'distilbert.pointer.decoder.input_to_hidden.bias',
```

```
'distilbert.pointer.encoder.lstm.weight_hh_l0',
'distilbert.pointer.encoder.lstm.bias_hh_l0',
'distilbert.pointer.decoder.hidden_out.bias',
'distilbert.pointer.embedding.weight', 'qa_outputs.weight',
'distilbert.pointer.decoder.input_to_hidden.weight',
'distilbert.pointer.decoder.runner', 'qa_outputs.bias',
'distilbert.pointer.decoder.att.context_linear.weight',
'distilbert.pointer.decoder.att.context_linear.bias',
'distilbert.pointer.encoder.lstm.bias_ih_l0',
'distilbert.pointer.decoder.att.input_linear.weight',
'distilbert.pointer.encoder.c0',
'distilbert.pointer.decoder.hidden_to_hidden.bias',
'distilbert.pointer.encoder.lstm.weight_ih_l0',
'distilbert.pointer.decoder.mask', 'distilbert.pointer.embedding.bias']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

6.0.1 Observations

[28]:
```
# Training the created model using the available cuda gpu or cpu
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

model.to(device) # send the model to the available device for training.
model.train()

train_dataloader = DataLoader(train_dataset, batch_size=8, shuffle=False)
val_dataloader = torch.utils.data.
 ↪DataLoader(val_dataset,batch_size=8,shuffle=False)

optim = Adafactor(model.parameters(),scale_parameter=False,␣
 ↪relative_step=False, lr=3e-5)
```

6.0.2 Observations

[29]:
```
torch.cuda.empty_cache()
```

6.0.3 Observations

[30]:
```
# Train for the model, perform validation on it per epoch and generate files␣
 ↪for a tensorboard
num_epochs = 2

writer = SummaryWriter()

for epoch in range(num_epochs):
    print('Epoch {}/{}'.format(epoch, num_epochs - 1))
    print('-' * 10)
    model.train()
```

```
    running_loss = 0.0
    tk0 = tqdm(train_dataloader, total=int(len(train_dataloader)))
    counter = 0
    for idx,batch in enumerate(tk0):
        optim.zero_grad()
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
        start_positions = batch['start_positions'].to(device)
        end_positions = batch['end_positions'].to(device)
        outputs = model(input_ids,␣
↪attention_mask=attention_mask,start_positions=start_positions,␣
↪end_positions=end_positions)
        loss = outputs[0]
        loss.backward()
        optim.step()
        running_loss += loss.item() *  batch['input_ids'].size(0)
        counter += 1
        tk0.set_postfix(loss=(running_loss / (counter * train_dataloader.
↪batch_size)))
    epoch_loss = running_loss / len(train_dataloader)
    writer.add_scalar('Train/Loss', epoch_loss,epoch)
    print('Training Loss: {:.4f}'.format(epoch_loss))

    model.eval()
    running_val_loss=0
    running_val_em=0
    running_val_f1=0
    tk1 = tqdm(val_dataloader, total=int(len(val_dataloader)))
    for idx,batch in enumerate(tk1):
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
        start_positions = batch['start_positions'].to(device)
        end_positions = batch['end_positions'].to(device)
        outputs = model(input_ids, attention_mask=attention_mask,␣
↪start_positions=start_positions, end_positions=end_positions)
        running_val_loss += loss.item() *  batch['input_ids'].size(0)
        counter += 1
        tk1.set_postfix(loss=(running_loss / (counter * val_dataloader.
↪batch_size)))
        answer_start = torch.argmax(outputs['start_logits'], dim=1)
        answer_end = torch.argmax(outputs['end_logits'], dim=1) + 1
        em_score, f1_score =␣
↪calculate_stats(input_ids,answer_start,answer_end,idx)
        running_val_em += em_score
        running_val_f1 += f1_score
    l = len(val_qac)
    epoch_v_loss = running_val_loss /l
```

```
    epoch_v_em = running_val_em/l
    epoch_val_f1 = running_val_f1/l
    writer.add_scalar('Val/Loss', epoch_v_loss,epoch)
    writer.add_scalar('Val/EM', epoch_v_em,epoch)
    writer.add_scalar('Val/F1', epoch_val_f1,epoch)
    print('Val Loss: {:.4f}, EM: {:.4f}, F1: {:.4f} '.
↪format(epoch_v_loss,epoch_v_em,epoch_val_f1))
```

```
Epoch 0/1
----------
```

```
HBox(children=(FloatProgress(value=0.0, max=10853.0), HTML(value='')))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:127: UserWarning:
Implicit dimension choice for softmax has been deprecated. Change the call to
include dim=X as an argument.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:240: UserWarning:
indexing with dtype torch.uint8 is now deprecated, please use a dtype torch.bool
instead. (Triggered internally at
/pytorch/aten/src/ATen/native/IndexingUtils.h:30.)
```

```
Training Loss: 11.1297
```

```
HBox(children=(FloatProgress(value=0.0, max=2538.0), HTML(value='')))
```

```
Val Loss: 2.0633, EM: 0.6457, F1: 0.7424
Epoch 1/1
----------
```

```
HBox(children=(FloatProgress(value=0.0, max=10853.0), HTML(value='')))
```

```
Training Loss: 6.9106
```

```
HBox(children=(FloatProgress(value=0.0, max=2538.0), HTML(value='')))
```

```
Val Loss: 1.2249, EM: 0.6577, F1: 0.7556
```

6.0.4 Observations

[31]:
```
# We save our model so that it can be reused later

torch.save(model,'./customBertmodelAdafactor3e.pt')
```

[32]:
```
# Generate a Tensorboard

%load_ext tensorboard
%tensorboard --logdir runs
```

```
<IPython.core.display.Javascript object>
```

6.0.5 Observations

We have created a Tensorboard to map the loss and accuracy across the various epochs that the model has trained at.

We will now run some examples to see how our model is performing & is it responding correctly to our questions.

7 5. Running The Model

We will now test the model on some contexts and questions to see if we are getting the correct answers

[33]:
```
test_context = """The Normans (Norman: Nourmands; French: Normands; Latin:␣
↪Normanni) were the people who in the 10th and 11th centuries gave their name␣
↪to Normandy, a region in France. They were descended from Norse ("Norman"␣
↪comes from "Norseman") raiders and pirates from Denmark, Iceland and Norway␣
↪who, under their leader Rollo, agreed to swear fealty to King Charles III of␣
↪West Francia. Through generations of assimilation and mixing with the native␣
↪Frankish and Roman-Gaulish populations, their descendants would gradually␣
↪merge with the Carolingian-based cultures of West Francia. The distinct␣
↪cultural and ethnic identity of the Normans emerged initially in the first␣
↪half of the 10th century, and it continued to evolve over the succeeding␣
↪centuries."""


test_question = """Who was the Norse leader?"""

test_answer =  "Rollo"
```

[34]:
```
def question_answer(question, context, model):
    inputs = tokenizer(question,context, return_tensors='pt')

    input_ids = inputs['input_ids'].to(device)

    attention_mask = inputs['attention_mask'].to(device)
    inputs.to(device)
    start_scores, end_scores = model(input_ids, attention_mask=attention_mask,␣
↪output_attentions=False)[:2]

    all_tokens = tokenizer.convert_ids_to_tokens(input_ids[0])
    answer = ' '.join(all_tokens[torch.argmax(start_scores) : torch.
↪argmax(end_scores)+1])
    answer = tokenizer.convert_tokens_to_ids(answer.split())
    answer = tokenizer.decode(answer)
    return answer
```

[35]:
```
question_answer(test_question, test_context, model)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:127: UserWarning:
Implicit dimension choice for softmax has been deprecated. Change the call to
include dim=X as an argument.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:240: UserWarning:
indexing with dtype torch.uint8 is now deprecated, please use a dtype torch.bool
instead. (Triggered internally at
/pytorch/aten/src/ATen/native/IndexingUtils.h:30.)
```

[35]:
```
'rollo'
```

[36]:
```
question_answer(val_questions[0], val_contexts[0], model)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:127: UserWarning:
Implicit dimension choice for softmax has been deprecated. Change the call to
include dim=X as an argument.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:240: UserWarning:
indexing with dtype torch.uint8 is now deprecated, please use a dtype torch.bool
instead. (Triggered internally at
/pytorch/aten/src/ATen/native/IndexingUtils.h:30.)
```

[36]:
```
'france'
```

8 Running The Model

[37]:
```
model_loaded = torch.load('./customBertmodelAdafactor3e.pt')
tokenizer = DistilBertTokenizerFast.from_pretrained('distilbert-base-uncased')
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

test_context = """The Normans (Norman: Nourmands; French: Normands; Latin:␣
↪Normanni) were the people who in the 10th and 11th centuries gave their name␣
↪to Normandy, a region in France. They were descended from Norse ("Norman"␣
↪comes from "Norseman") raiders and pirates from Denmark, Iceland and Norway␣
↪who, under their leader Rollo, agreed to swear fealty to King Charles III of␣
↪West Francia. Through generations of assimilation and mixing with the native␣
↪Frankish and Roman-Gaulish populations, their descendants would gradually␣
↪merge with the Carolingian-based cultures of West Francia. The distinct␣
↪cultural and ethnic identity of the Normans emerged initially in the first␣
↪half of the 10th century, and it continued to evolve over the succeeding␣
↪centuries."""
test_question = """Who was the Norse leader?"""
test_answer =  "Rollo"
```

[38]:
```
def question_answer(question, context, model):
    inputs = tokenizer(question,context, return_tensors='pt')

    input_ids = inputs['input_ids'].to(device)
```

```
    attention_mask = inputs['attention_mask'].to(device)
    inputs.to(device)
    start_scores, end_scores = model(input_ids, attention_mask=attention_mask,␣
 ↪output_attentions=False)[:2]

    all_tokens = tokenizer.convert_ids_to_tokens(input_ids[0])
    answer = ' '.join(all_tokens[torch.argmax(start_scores) : torch.
 ↪argmax(end_scores)+1])
    answer = tokenizer.convert_tokens_to_ids(answer.split())
    answer = tokenizer.decode(answer)
    return answer
```

[39]:
```
question_answer(test_question, test_context, model_loaded)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:127: UserWarning:
Implicit dimension choice for softmax has been deprecated. Change the call to
include dim=X as an argument.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:240: UserWarning:
indexing with dtype torch.uint8 is now deprecated, please use a dtype torch.bool
instead. (Triggered internally at
/pytorch/aten/src/ATen/native/IndexingUtils.h:30.)
```

[39]:
```
'rollo'
```

[40]:
```
## we will now take some text at random from Wikipedia and test our model. This␣
 ↪excerpt can be found at:
## https://en.wikipedia.org/wiki/Long_short-term_memory under the Idea heading.
context = """In theory, classic (or "vanilla") RNNs can keep track of arbitrary␣
 ↪long-term dependencies in the input sequences. The problem with vanilla RNNs␣
 ↪is computational (or practical) in nature: when training a vanilla RNN using␣
 ↪back-propagation, the gradients which are back-propagated can "vanish" (that␣
 ↪is, they can tend to zero) or "explode" (that is, they can tend to␣
 ↪infinity), because of the computations involved in the process, which use␣
 ↪finite-precision numbers. RNNs using LSTM units partially solve the␣
 ↪vanishing gradient problem, because LSTM units allow gradients to also flow␣
 ↪unchanged. However, LSTM networks can still suffer from the exploding␣
 ↪gradient problem."""
question = """What problem can LSTM suffer from?"""
answer = """exploding gradient problem"""
```

[41]:
```
question_answer(question, context, model_loaded)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:127: UserWarning:
Implicit dimension choice for softmax has been deprecated. Change the call to
include dim=X as an argument.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:240: UserWarning:
indexing with dtype torch.uint8 is now deprecated, please use a dtype torch.bool
instead. (Triggered internally at
```

```
    /pytorch/aten/src/ATen/native/IndexingUtils.h:30.)
```

[41]: 'exploding gradient problem'