Open in Colab

T81-558: Applications of Deep Neural Networks

Module 11: Natural Language Processing with Hugging Face

- Instructor: Jeff Heaton, McKelvey School of Engineering, Washington University in St. Louis
- For more information visit the class website.

Module 11 Material

- Part 11.1: Introduction to Hugging Face [Video] [Notebook]
- **Part 11.2: Hugging Face Tokenizers** [Video] [Notebook]
- Part 11.3: Hugging Face Datasets [Video] [Notebook]
- Part 11.4: Training Hugging Face Models [Video] [Notebook]
- Part 11.5: What are Embedding Layers in Keras [Video] [Notebook]

Google CoLab Instructions

The following code ensures that Google CoLab is running the correct version of TensorFlow.

In [ ]:
```
try:
    %tensorflow_version 2.x
    COLAB = True
    print("Note: using Google CoLab")
except:
    print("Note: not using Google CoLab")
    COLAB = False
```

```
Note: using Google CoLab
```

Part 11.2: Hugging Face Tokenizers

Tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation. Consider how the program might break up the following sentences into words.

- This is a test.

- Ok, but what about this?
- Is U.S.A. the same as USA.?
- What is the best data-set to use?
- I think I will do this-no wait; I will do that.

The hugging face includes tokenizers that can break these sentences into words and subwords. Because English, and some other languages, are made up of common word parts, we tokenize subwords. For example, a gerund word, such as "sleeping," will be tokenized into "sleep" and "##ing".

We begin by installing Hugging Face if needed.

In [ ]:
```
# HIDE OUTPUT
!pip install transformers
!pip install transformers[sentencepiece]
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.7/dist
-packages (4.17.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-pac
kages (from transformers) (2.23.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-pac
kages (from transformers) (3.6.0)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.
7/dist-packages (from transformers) (2019.12.20)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.7/dist-
packages (from transformers) (6.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/d
ist-packages (from transformers) (21.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
packages (from transformers) (1.21.5)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-p
ackages (from transformers) (4.63.0)
Requirement already satisfied: tokenizers!=0.11.3,>=0.11.1 in /usr/local/li
b/python3.7/dist-packages (from transformers) (0.11.6)
Requirement already satisfied: huggingface-hub<1.0,>=0.1.0 in /usr/local/li
b/python3.7/dist-packages (from transformers) (0.4.0)
Requirement already satisfied: sacremoses in /usr/local/lib/python3.7/dist-p
ackages (from transformers) (0.0.49)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.
7/dist-packages (from transformers) (4.11.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/
python3.7/dist-packages (from huggingface-hub<1.0,>=0.1.0->transformers) (3.
10.0.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/py
thon3.7/dist-packages (from packaging>=20.0->transformers) (3.0.7)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-pa
ckages (from importlib-metadata->transformers) (3.7.0)
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) (2021.10.8)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist
-packages (from requests->transformers) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /u
sr/local/lib/python3.7/dist-packages (from requests->transformers) (1.24.3)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packag
es (from sacremoses->transformers) (7.1.2)
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-packa
ges (from sacremoses->transformers) (1.1.0)
Requirement already satisfied: transformers[sentencepiece] in /usr/local/li
b/python3.7/dist-packages (4.17.0)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-p
ackages (from transformers[sentencepiece]) (4.63.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-pac
kages (from transformers[sentencepiece]) (2.23.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.1.0 in /usr/local/li
b/python3.7/dist-packages (from transformers[sentencepiece]) (0.4.0)
Requirement already satisfied: sacremoses in /usr/local/lib/python3.7/dist-p
ackages (from transformers[sentencepiece]) (0.0.49)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
```

```
packages (from transformers[sentencepiece]) (1.21.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/d
ist-packages (from transformers[sentencepiece]) (21.3)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.
7/dist-packages (from transformers[sentencepiece]) (4.11.3)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.7/dist-
packages (from transformers[sentencepiece]) (6.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-pac
kages (from transformers[sentencepiece]) (3.6.0)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.
7/dist-packages (from transformers[sentencepiece]) (2019.12.20)
Requirement already satisfied: tokenizers!=0.11.3,>=0.11.1 in /usr/local/li
b/python3.7/dist-packages (from transformers[sentencepiece]) (0.11.6)
Requirement already satisfied: sentencepiece!=0.1.92,>=0.1.91 in /usr/local/
lib/python3.7/dist-packages (from transformers[sentencepiece]) (0.1.96)
Requirement already satisfied: protobuf in /usr/local/lib/python3.7/dist-pac
kages (from transformers[sentencepiece]) (3.17.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/
python3.7/dist-packages (from huggingface-hub<1.0,>=0.1.0->transformers[sent
encepiece]) (3.10.0.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/py
thon3.7/dist-packages (from packaging>=20.0->transformers[sentencepiece])
(3.0.7)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-pa
ckages (from importlib-metadata->transformers[sentencepiece]) (3.7.0)
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.7/dist-pac
kages (from protobuf->transformers[sentencepiece]) (1.15.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist
-packages (from requests->transformers[sentencepiece]) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.
7/dist-packages (from requests->transformers[sentencepiece]) (2021.10.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /u
sr/local/lib/python3.7/dist-packages (from requests->transformers[sentencepi
ece]) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.
7/dist-packages (from requests->transformers[sentencepiece]) (3.0.4)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packa
ges (from sacremoses->transformers[sentencepiece]) (1.1.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packag
es (from sacremoses->transformers[sentencepiece]) (7.1.2)
```

First, we create a Hugging Face tokenizer. There are several different tokenizers available from the Hugging Face hub. For this example, we will make use of the following tokenizer:

- distilbert-base-uncased

This tokenizer is based on BERT and assumes case-insensitive English text.

In [ ]:
```
from transformers import AutoTokenizer
model = "distilbert-base-uncased"
tokenizer = AutoTokenizer.from_pretrained(model)
```

We can now tokenize a sample sentence.

In [ ]:
```
encoded = tokenizer('Tokenizing text is easy.')
print(encoded)
```

```
{'input_ids': [101, 19204, 6026, 3793, 2003, 3733, 1012, 102], 'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1]}
```

The result of this tokenization contains two elements:

- input_ids - The individual subword indexes, each index uniquely identifies a subword.
- attention_mask - Which values in *input_ids* are meaningful and not padding. This sentence had no padding, so all elements have an attention mask of "1". Later, we will request the output to be of a fixed length, introducing padding, which always has an attention mask of "0". Though each tokenizer can be implemented differently, the attention mask of a tokenizer is generally either "0" or "1".

Due to subwords and special tokens, the number of tokens may not match the number of words in the source string. We can see the meanings of the individual tokens by converting these IDs back to strings.

In [ ]:
```
tokenizer.convert_ids_to_tokens(encoded.input_ids)
```

Out[ ]:
```
['[CLS]', 'token', '##izing', 'text', 'is', 'easy', '.', '[SEP]']
```

As you can see, there are two special tokens placed at the beginning and end of each sequence. We will soon see how we can include or exclude these special tokens. These special tokens can vary per tokenizer; however, [CLS] begins a sequence for this tokenizer, and [SEP] ends a sequence. You will also see that the gerund "tokening" is broken into "token" and "*ing".

For this tokenizer, the special tokens occur between 100 and 103. Most Hugging Face tokenizers use this approximate range for special tokens. The value zero (0) typically represents padding. We can display all special tokens with this command.

In [ ]:
```
tokenizer.convert_ids_to_tokens([0, 100, 101, 102, 103])
```

Out[ ]:
```
['[PAD]', '[UNK]', '[CLS]', '[SEP]', '[MASK]']
```

This tokenizer supports these common tokens:

- [CLS] - Sequence beginning.
- [SEP] - Sequence end.
- [PAD] - Padding.
- [UNK] - Unknown token.
- [MASK] - Mask out tokens for a neural network to predict. Not used in this book, see MLM paper.

It is also possible to tokenize lists of sequences. We can pad and truncate sequences to achieve a standard length by tokenizing many sequences at once.

In [ ]:
```
text = [
    "This movie was great!",
    "I hated this move, waste of time!",
    "Epic?"
]

encoded = tokenizer(text, padding=True, add_special_tokens=True)

print("**Input IDs**")
for a in encoded.input_ids:
    print(a)

print("**Attention Mask**")
for a in encoded.attention_mask:
    print(a)
```

```
**Input IDs**
[101, 2023, 3185, 2001, 2307, 999, 102, 0, 0, 0, 0]
[101, 1045, 6283, 2023, 2693, 1010, 5949, 1997, 2051, 999, 102]
[101, 8680, 1029, 102, 0, 0, 0, 0, 0, 0, 0]
**Attention Mask**
[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
```

Notice the **input_id**'s for the three movie review text sequences. Each of these sequences begins with 101 and we pad with zeros. Just before the padding, each group of IDs ends with 102. The attention masks also have zeros for each of the padding entries.

We used two parameters to the tokenizer to control the tokenization process. Some other useful parameters include:

- add_special_tokens (defaults to True) Whether or not to encode the sequences with the special tokens relative to their model.
- padding (defaults to False) Activates and controls truncation.
- max_length (optional) Controls the maximum length to use by one of the truncation/padding parameters.

In [ ]:
```
```