Evaluating Student Writing

We aim to identify elements in student writing i.e. we segment text and classify argumentative and rhetorical elements i.e. predict human annotations in essays written by 6th-12th grade students.

(The demo and visualisation on the test data is at the end of the file)

Setup

In [1]:
```
# Install libraries to get evaluation metrics for training data
!pip install seqeval
!pip install seqeval -qq
!pip install wandb
!pip install --upgrade wandb -qq
import wandb
import warnings
warnings.filterwarnings('ignore')
warnings.simplefilter('ignore')

# visualization with displacy
import pandas as pd
import os
from pathlib import Path
import spacy
from spacy import displacy
from pylab import cm, matplotlib
```

```
Requirement already satisfied: seqeval in /opt/conda/lib/python3.7/site-packag
es (1.2.2)
Requirement already satisfied: scikit-learn>=0.21.3 in /opt/conda/lib/python3.
7/site-packages (from seqeval) (0.23.2)
Requirement already satisfied: numpy>=1.14.0 in /opt/conda/lib/python3.7/site-
packages (from seqeval) (1.19.5)
Requirement already satisfied: joblib>=0.11 in /opt/conda/lib/python3.7/site-p
ackages (from scikit-learn>=0.21.3->seqeval) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/lib/python3.
7/site-packages (from scikit-learn>=0.21.3->seqeval) (3.0.0)
Requirement already satisfied: scipy>=0.19.1 in /opt/conda/lib/python3.7/site-
packages (from scikit-learn>=0.21.3->seqeval) (1.7.2)
WARNING: Running pip as the 'root' user can result in broken permissions and c
onflicting behaviour with the system package manager. It is recommended to use
a virtual environment instead: https://pip.pypa.io/warnings/venv
Requirement already satisfied: wandb in /opt/conda/lib/python3.7/site-packages
(0.12.16)
Requirement already satisfied: Click!=8.0.0,>=7.0 in /opt/conda/lib/python3.7/
site-packages (from wandb) (8.0.3)
Requirement already satisfied: docker-pycreds>=0.4.0 in /opt/conda/lib/python
3.7/site-packages (from wandb) (0.4.0)
Requirement already satisfied: psutil>=5.0.0 in /opt/conda/lib/python3.7/site-
packages (from wandb) (5.8.0)
Requirement already satisfied: protobuf>=3.12.0 in /opt/conda/lib/python3.7/si
te-packages (from wandb) (3.19.1)
Requirement already satisfied: python-dateutil>=2.6.1 in /opt/conda/lib/python
3.7/site-packages (from wandb) (2.8.0)
Requirement already satisfied: six>=1.13.0 in /opt/conda/lib/python3.7/site-pa
ckages (from wandb) (1.16.0)
Requirement already satisfied: setproctitle in /opt/conda/lib/python3.7/site-p
ackages (from wandb) (1.2.3)
Requirement already satisfied: PyYAML in /opt/conda/lib/python3.7/site-package
s (from wandb) (6.0)
Requirement already satisfied: shortuuid>=0.5.0 in /opt/conda/lib/python3.7/si
te-packages (from wandb) (1.0.8)
Requirement already satisfied: sentry-sdk>=1.0.0 in /opt/conda/lib/python3.7/s
ite-packages (from wandb) (1.5.0)
Requirement already satisfied: pathtools in /opt/conda/lib/python3.7/site-pack
ages (from wandb) (0.1.2)
Requirement already satisfied: promise<3,>=2.0 in /opt/conda/lib/python3.7/sit
e-packages (from wandb) (2.3)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-pac
kages (from wandb) (59.1.1)
Requirement already satisfied: requests<3,>=2.0.0 in /opt/conda/lib/python3.7/
site-packages (from wandb) (2.25.1)
Requirement already satisfied: GitPython>=1.0.0 in /opt/conda/lib/python3.7/si
te-packages (from wandb) (3.1.24)
Requirement already satisfied: importlib-metadata in /opt/conda/lib/python3.7/
site-packages (from Click!=8.0.0,>=7.0->wandb) (4.8.2)
Requirement already satisfied: gitdb<5,>=4.0.1 in /opt/conda/lib/python3.7/sit
e-packages (from GitPython>=1.0.0->wandb) (4.0.9)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /opt/conda/lib/py
thon3.7/site-packages (from GitPython>=1.0.0->wandb) (3.10.0.2)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/python3.7/site-p
ackages (from requests<3,>=2.0.0->wandb) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.7/
site-packages (from requests<3,>=2.0.0->wandb) (2021.10.8)
Requirement already satisfied: chardet<5,>=3.0.2 in /opt/conda/lib/python3.7/s
ite-packages (from requests<3,>=2.0.0->wandb) (4.0.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/lib/python
```

```
3.7/site-packages (from requests<3,>=2.0.0->wandb) (1.26.7)
Requirement already satisfied: smmap<6,>=3.0.1 in /opt/conda/lib/python3.7/sit
e-packages (from gitdb<5,>=4.0.1->GitPython>=1.0.0->wandb) (3.0.5)
Requirement already satisfied: zipp>=0.5 in /opt/conda/lib/python3.7/site-pack
ages (from importlib-metadata->Click!=8.0.0,>=7.0->wandb) (3.6.0)
WARNING: Running pip as the 'root' user can result in broken permissions and c
onflicting behaviour with the system package manager. It is recommended to use
a virtual environment instead: https://pip.pypa.io/warnings/venv
```

Choose Model to run

In [2]:
```
print("Choose the model to run: \n 1. LongFormer \n 2. BigBird")
print("Kindly enter 1 or 2")
while True:
    input_model = input()
    if len(input_model)>1:
        print("Invalid Input, Kindly enter 1 or 2")
    else:
        if input_model == '1':
            model_checkpoint = "allenai/longformer-base-4096"
            print("The model choosen is Longformer")
            break
        elif input_model == '2':
            model_checkpoint = "google/bigbird-roberta-base"
            print("The model choosen is BigBird")
            break
        else:
            print("Invalid Input, Kindly enter 1 or 2")
```

```
Choose the model to run:
 1. LongFormer
 2. BigBird
Kindly enter 1 or 2

The model choosen is BigBird
```

Configurations

In [3]:
```
# Configurations

SAMPLE =  False # Used for debugging

EXP_NUM = 4
task = "ner"
max_length = 1024
stride = 128
min_tokens = 6
model_path = f'{model_checkpoint.split("/")[-1]}-{EXP_NUM}'
max_length = 1024
batch_size = 4

# TRAINING HYPERPARAMS
BS = 4
GRAD_ACC = 8
LR = 5e-5
WD = 0.01
WARMUP = 0.1
N_EPOCHS = 5
```

Data Preprocessing

In [4]:
```
import pandas as pd

# Importing the train data
train = pd.read_csv('../input/feedback-prize-2021/train.csv')
train.head()
```

Out[4]:

	id	discourse_id	discourse_start	discourse_end	discourse_text	discourse_type
0	423A1CA112E2	1.622628e+12	8.0	229.0	Modern humans today are always on their phone....	Lead
1	423A1CA112E2	1.622628e+12	230.0	312.0	They are some really bad consequences when stu...	Position
2	423A1CA112E2	1.622628e+12	313.0	401.0	Some certain areas in the United States ban ph...	Evidence
3	423A1CA112E2	1.622628e+12	402.0	758.0	When people have phones, they know about certa...	Evidence
4	423A1CA112E2	1.622628e+12	759.0	886.0	Driving is one of the way how to get around. P...	Claim

In [5]:
```
# Viewing the unique classes in the dataset
classes = train.discourse_type.unique().tolist()
classes
```

Out[5]:
```
['Lead',
 'Position',
 'Evidence',
 'Claim',
 'Concluding Statement',
 'Counterclaim',
 'Rebuttal']
```

In [6]:
```
# Setting label incides
from collections import defaultdict
tags = defaultdict()

for i, c in enumerate(classes):
    print(i,c)
    tags[f'B-{c}'] = i
    tags[f'I-{c}'] = i + len(classes)
tags[f'O'] = len(classes) * 2
tags[f'Special'] = -100

l2i = dict(tags)
```

```
i2l = defaultdict()
for k, v in l2i.items():
    i2l[v] = k
i2l[-100] = 'Special'

i2l = dict(i2l)

N_LABELS = len(i2l) - 1
```

```
0 Lead
1 Position
2 Evidence
3 Claim
4 Concluding Statement
5 Counterclaim
6 Rebuttal
```

In [7]:
```
# Viewing the tags assigned to the classes
tags
```

Out[7]:
```
defaultdict(None,
            {'B-Lead': 0,
             'I-Lead': 7,
             'B-Position': 1,
             'I-Position': 8,
             'B-Evidence': 2,
             'I-Evidence': 9,
             'B-Claim': 3,
             'I-Claim': 10,
             'B-Concluding Statement': 4,
             'I-Concluding Statement': 11,
             'B-Counterclaim': 5,
             'I-Counterclaim': 12,
             'B-Rebuttal': 6,
             'I-Rebuttal': 13,
             'O': 14,
             'Special': -100})
```

In [8]:
```
# Reading raw text from the essay files

from pathlib import Path

path = Path('../input/feedback-prize-2021/train')

def get_raw_text(ids):
    with open(path/f'{ids}.txt', 'r') as file: data = file.read()
    return data
```

In [9]:
```
# Grouping annotations based on discourse_type, discourse_start, discourse_end
# predictionstring to form single tuple for each essay

df1 = train.groupby('id')['discourse_type'].apply(list).reset_index(name='class
df2 = train.groupby('id')['discourse_start'].apply(list).reset_index(name='star
df3 = train.groupby('id')['discourse_end'].apply(list).reset_index(name='ends')
df4 = train.groupby('id')['predictionstring'].apply(list).reset_index(name='pre

#Merging the dataframes
df = pd.merge(df1, df2, how='inner', on='id')
df = pd.merge(df, df3, how='inner', on='id')
```

```
df = pd.merge(df, df4, how='inner', on='id')

#Adding raw essay text to the merged data
df['text'] = df['id'].apply(get_raw_text)
```

In [10]:
```
#Viewing the merged data
df.head()
```

Out[10]:

	id	classlist	starts	ends	predictionstrings	text
0	0000D23A521A	[Position, Evidence, Evidence, Claim, Counterc...	[0.0, 170.0, 358.0, 438.0, 627.0, 722.0, 836.0...	[170.0, 357.0, 438.0, 626.0, 722.0, 836.0, 101...	[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 1...	Some people belive that the so called "face" o...
1	00066EA9880D	[Lead, Position, Claim, Evidence, Claim, Evide...	[0.0, 456.0, 638.0, 738.0, 1399.0, 1488.0, 231...	[455.0, 592.0, 738.0, 1398.0, 1487.0, 2219.0, ...	[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 1...	Driverless cars are exaclty what you would exp...
2	000E6DE9E817	[Position, Counterclaim, Rebuttal, Evidence, C...	[17.0, 64.0, 158.0, 310.0, 438.0, 551.0, 776.0...	[56.0, 157.0, 309.0, 422.0, 551.0, 775.0, 961....	[2 3 4 5 6 7 8, 10 11 12 13 14 15 16 17 18 19 ...	Dear: Principal\n\nI am arguing against the po...
3	001552828BD0	[Lead, Evidence, Claim, Claim, Evidence, Claim...	[0.0, 161.0, 872.0, 958.0, 1191.0, 1542.0, 161...	[160.0, 872.0, 957.0, 1190.0, 1541.0, 1612.0, ...	[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 1...	Would you be able to give your car up? Having ...
4	0016926B079C	[Position, Claim, Claim, Claim, Claim, Evidenc...	[0.0, 58.0, 94.0, 206.0, 236.0, 272.0, 542.0, ...	[57.0, 91.0, 150.0, 235.0, 271.0, 542.0, 650.0...	[0 1 2 3 4 5 6 7 8 9, 10 11 12 13 14 15, 16 17...	I think that students would benefit from learn...

In [11]:
```
# Size of the dataset before removing outliers
df2=df
df.shape
```

Out[11]:
```
(15594, 6)
```

In [12]:
```
# Removing outliers i.e. essays with more than 5 occurances of the same class a

from collections import Counter
res = []
for i in range(len(df['classlist'])):
```

```
    temp = df['classlist'][i]
    res.append(dict(Counter(temp)))

def countOcuurances(res, df2):
    df = pd.DataFrame(res)
    classes = ['Lead', 'Position', 'Evidence', 'Claim', 'Concluding Statement',
    for c in classes:
        index = df[df[c]>5].index
        df = df.drop(index)
        df2 = df2.drop(index)

    return df2

df2 = countOcuurances(res,df2)
df = df2
```

In [13]:
```
#Size of the dataset after removing outliers
df.shape
```

Out[13]:
```
(12736, 6)
```

In [14]:
```
# debugging
if SAMPLE: df = df.sample(n=10).reset_index(drop=True)
```

In [15]:
```
# Performing Train Test split
from datasets import Dataset, load_metric

ds = Dataset.from_pandas(df)
datasets = ds.train_test_split(test_size=0.1, shuffle=True, seed=42)
datasets
```

Out[15]:
```
DatasetDict({
    train: Dataset({
        features: ['id', 'classlist', 'starts', 'ends', 'predictionstrings',
'text', '__index_level_0__'],
        num_rows: 11462
    })
    test: Dataset({
        features: ['id', 'classlist', 'starts', 'ends', 'predictionstrings',
'text', '__index_level_0__'],
        num_rows: 1274
    })
})
```

In [16]:
```
#Initialing the tokenizer
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained(model_checkpoint, add_prefix_space=Tr
```

```
normalizer.cc(51) LOG(INFO) precompiled_charsmap is empty. use identity normal
ization.
```

In [17]:
```
# If a span is created wihout a starting token for a class
# then we convert the first token to be the starting token
def fix_beginnings(labels):
    for i in range(1,len(labels)):
        curr_lab = labels[i]
        prev_lab = labels[i-1]
        if curr_lab in range(7,14):
```

```
            if prev_lab != curr_lab and prev_lab != curr_lab - 7:
                labels[i] = curr_lab -7
    return labels
```

In [18]:
```
# tokenizing and adding labels
def tokenize_and_align_labels(examples):

    o = tokenizer(examples['text'], truncation=True, padding=True,
                  return_offsets_mapping=True, max_length=max_length,
                  stride=stride, return_overflowing_tokens=True)
    #print(o.keys())
    sample_mapping = o["overflow_to_sample_mapping"]
    offset_mapping = o["offset_mapping"]

    o["labels"] = []

    for i in range(len(offset_mapping)):

        sample_index = sample_mapping[i]

        labels = [l2i['O'] for i in range(len(o['input_ids'][i]))]

        for label_start, label_end, label in \
        list(zip(examples['starts'][sample_index], examples['ends'][sample_inde
            for j in range(len(labels)):
                token_start = offset_mapping[i][j][0]
                token_end = offset_mapping[i][j][1]
                if token_start == label_start:
                    labels[j] = l2i[f'B-{label}']
                if token_start > label_start and token_end <= label_end:
                    labels[j] = l2i[f'I-{label}']

        for k, input_id in enumerate(o['input_ids'][i]):
            if input_id in [0,1,2]:
                labels[k] = -100

        labels = fix_beginnings(labels)
        o["labels"].append(labels)

    return o
```

In [19]:
```
# Tokenising both train and test
tokenized_datasets = datasets.map(tokenize_and_align_labels, batched=True, \
                                  batch_size=20000,
                                  remove_columns=datasets["train"].column_names
```

```
  0%|          | 0/1 [00:00<?, ?ba/s]
  0%|          | 0/1 [00:00<?, ?ba/s]
```

In [20]:
```
tokenized_datasets
```

Out[20]:
```
DatasetDict({
    train: Dataset({
        features: ['attention_mask', 'input_ids', 'labels', 'offset_mapping',
'overflow_to_sample_mapping'],
        num_rows: 11727
    })
    test: Dataset({
        features: ['attention_mask', 'input_ids', 'labels', 'offset_mapping',
'overflow_to_sample_mapping'],
        num_rows: 1301
    })
})
```

Model and Training

In [21]:
```
from transformers import AutoModelForTokenClassification, TrainingArguments, Tr

model = AutoModelForTokenClassification.from_pretrained(model_checkpoint, num_l
```

```
Some weights of the model checkpoint at google/bigbird-roberta-base were not u
sed when initializing BigBirdForTokenClassification: ['cls.predictions.transfo
rm.LayerNorm.weight', 'cls.predictions.bias', 'cls.predictions.transform.Layer
Norm.bias', 'cls.seq_relationship.bias', 'cls.predictions.decoder.weight', 'cl
s.predictions.transform.dense.bias', 'cls.predictions.decoder.bias', 'cls.seq_
relationship.weight', 'cls.predictions.transform.dense.weight']
- This IS expected if you are initializing BigBirdForTokenClassification from
the checkpoint of a model trained on another task or with another architecture
(e.g. initializing a BertForSequenceClassification model from a BertForPreTrai
ning model).
- This IS NOT expected if you are initializing BigBirdForTokenClassification f
rom the checkpoint of a model that you expect to be exactly identical (initial
izing a BertForSequenceClassification model from a BertForSequenceClassificati
on model).
Some weights of BigBirdForTokenClassification were not initialized from the mo
del checkpoint at google/bigbird-roberta-base and are newly initialized: ['cla
ssifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use i
t for predictions and inference.
```

In [22]:
```
#print(model)
```

In [23]:
```
model_name = model_checkpoint.split("/")[-1]
args = TrainingArguments(
    f"{model_name}-finetuned-{task}",
    evaluation_strategy = "epoch",
    logging_strategy = "epoch",
    save_strategy = "epoch",
    learning_rate=LR,
    per_device_train_batch_size=BS,
    per_device_eval_batch_size=BS,
    num_train_epochs=N_EPOCHS,
    weight_decay=WD,
    report_to='wandb',
    gradient_accumulation_steps=GRAD_ACC,
    warmup_ratio=WARMUP
)
```

In [24]:
```
from transformers import DataCollatorForTokenClassification

data_collator = DataCollatorForTokenClassification(tokenizer)
```

In [25]:
```
# Loading Metric
metric = load_metric("seqeval")
```

In [26]:
```
import numpy as np

def compute_metrics(p):
    predictions, labels = p
    predictions = np.argmax(predictions, axis=2)

    # Remove special tokens
    true_predictions = [
        [i2l[p] for (p, l) in zip(prediction, label) if l != -100]
        for prediction, label in zip(predictions, labels)
    ]
    true_labels = [
        [i2l[l] for (p, l) in zip(prediction, label) if l != -100]
        for prediction, label in zip(predictions, labels)
    ]

    results = metric.compute(predictions=true_predictions, references=true_labe
    return {
        "precision": results["overall_precision"],
        "recall": results["overall_recall"],
        "f1": results["overall_f1"],
        "accuracy": results["overall_accuracy"],
    }
```

In [27]:
```
trainer = Trainer(
    model,
    args,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=tokenized_datasets["test"],
    data_collator=data_collator,
    tokenizer=tokenizer,
    compute_metrics=compute_metrics,
)
```

In [28]:
```
import os
os.environ["WANDB_DISABLED"] = "true"
```

In [29]:
```
trainer.train()
wandb.finish()
```

```
The following columns in the training set  don't have a corresponding argument
in `BigBirdForTokenClassification.forward` and have been ignored: overflow_to_
sample_mapping, offset_mapping.
***** Running training *****
  Num examples = 11727
  Num Epochs = 5
  Instantaneous batch size per device = 4
  Total train batch size (w. parallel, distributed & accumulation) = 32
  Gradient Accumulation steps = 8
  Total optimization steps = 1830
Automatic Weights & Biases logging enabled, to disable set os.environ["WANDB_D
ISABLED"] = "true"
huggingface/tokenizers: The current process just got forked, after parallelism
has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true
| false)
huggingface/tokenizers: The current process just got forked, after parallelism
has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true
| false)
```

Tracking run with wandb version 0.12.16

W&B syncing is set to `offline` in this directory.

```
Run `wandb online` or set WANDB_MODE=online to enable cloud syncing.
```

[1830/1830 3:40:01, Epoch 4/5]

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
0	1.087800	0.689825	0.206134	0.336801	0.255744	0.778607
1	0.640200	0.635552	0.235993	0.386356	0.293011	0.792288
2	0.530500	0.623517	0.238571	0.396204	0.297815	0.795563
3	0.445400	0.660478	0.259748	0.414231	0.319285	0.795294
4	0.380900	0.679613	0.246262	0.406766	0.306789	0.790095

```
The following columns in the evaluation set  don't have a corresponding argume
nt in `BigBirdForTokenClassification.forward` and have been ignored: overflow_
to_sample_mapping, offset_mapping.
***** Running Evaluation *****
  Num examples = 1301
  Batch size = 4
Saving model checkpoint to bigbird-roberta-base-finetuned-ner/checkpoint-366
Configuration saved in bigbird-roberta-base-finetuned-ner/checkpoint-366/confi
g.json
Model weights saved in bigbird-roberta-base-finetuned-ner/checkpoint-366/pytor
ch_model.bin
tokenizer config file saved in bigbird-roberta-base-finetuned-ner/checkpoint-3
66/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-finetuned-ner/checkpoint-36
6/special_tokens_map.json
The following columns in the evaluation set  don't have a corresponding argume
nt in `BigBirdForTokenClassification.forward` and have been ignored: overflow_
to_sample_mapping, offset_mapping.
***** Running Evaluation *****
  Num examples = 1301
  Batch size = 4
Saving model checkpoint to bigbird-roberta-base-finetuned-ner/checkpoint-732
Configuration saved in bigbird-roberta-base-finetuned-ner/checkpoint-732/confi
g.json
Model weights saved in bigbird-roberta-base-finetuned-ner/checkpoint-732/pytor
ch_model.bin
tokenizer config file saved in bigbird-roberta-base-finetuned-ner/checkpoint-7
32/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-finetuned-ner/checkpoint-73
2/special_tokens_map.json
The following columns in the evaluation set  don't have a corresponding argume
nt in `BigBirdForTokenClassification.forward` and have been ignored: overflow_
to_sample_mapping, offset_mapping.
***** Running Evaluation *****
  Num examples = 1301
  Batch size = 4
Saving model checkpoint to bigbird-roberta-base-finetuned-ner/checkpoint-1098
Configuration saved in bigbird-roberta-base-finetuned-ner/checkpoint-1098/conf
ig.json
Model weights saved in bigbird-roberta-base-finetuned-ner/checkpoint-1098/pyto
rch_model.bin
tokenizer config file saved in bigbird-roberta-base-finetuned-ner/checkpoint-1
098/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-finetuned-ner/checkpoint-109
8/special_tokens_map.json
The following columns in the evaluation set  don't have a corresponding argume
nt in `BigBirdForTokenClassification.forward` and have been ignored: overflow_
to_sample_mapping, offset_mapping.
***** Running Evaluation *****
  Num examples = 1301
  Batch size = 4
Saving model checkpoint to bigbird-roberta-base-finetuned-ner/checkpoint-1464
Configuration saved in bigbird-roberta-base-finetuned-ner/checkpoint-1464/conf
ig.json
Model weights saved in bigbird-roberta-base-finetuned-ner/checkpoint-1464/pyto
rch_model.bin
tokenizer config file saved in bigbird-roberta-base-finetuned-ner/checkpoint-1
464/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-finetuned-ner/checkpoint-146
4/special_tokens_map.json
```

```
The following columns in the evaluation set  don't have a corresponding argume
nt in `BigBirdForTokenClassification.forward` and have been ignored: overflow_
to_sample_mapping, offset_mapping.
***** Running Evaluation *****
  Num examples = 1301
  Batch size = 4
Saving model checkpoint to bigbird-roberta-base-finetuned-ner/checkpoint-1830
Configuration saved in bigbird-roberta-base-finetuned-ner/checkpoint-1830/conf
ig.json
Model weights saved in bigbird-roberta-base-finetuned-ner/checkpoint-1830/pyto
rch_model.bin
tokenizer config file saved in bigbird-roberta-base-finetuned-ner/checkpoint-1
830/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-finetuned-ner/checkpoint-183
0/special_tokens_map.json


Training completed. Do not forget to share your model on huggingface.co/models
=)
```

Waiting for W&B process to finish... **(success).**

```
VBox(children=(Label(value='0.000 MB of 0.000 MB uploaded (0.000 MB deduped)
\r'), FloatProgress(value=1.0, max…
```

Run history:

eval/accuracy	
eval/f1	
eval/loss	
eval/precision	
eval/recall	
eval/runtime	
eval/samples_per_second	
eval/steps_per_second	
train/epoch	
train/global_step	
train/learning_rate	
train/loss	
train/total_flos	
train/train_loss	
train/train_runtime	
train/train_samples_per_second	
train/train_steps_per_second	

Run summary:

eval/accuracy	0.79009
eval/f1	0.30679
eval/loss	0.67961
eval/precision	0.24626
eval/recall	0.40677
eval/runtime	106.2539
eval/samples_per_second	12.244
eval/steps_per_second	3.068
train/epoch	5.0
train/global_step	1830
train/learning_rate	0.0
train/loss	

	0.3809
train/total_flos	3.085074066272256e+16
train/train_loss	0.61696
train/train_runtime	13215.7603
train/train_samples_per_second	4.437
train/train_steps_per_second	0.138

You can sync this run to the cloud by running:
`wandb sync /kaggle/working/wandb/offline-run-20220505_200605-2z3tg5g8`
Find logs at: `./wandb/offline-run-20220505_200605-2z3tg5g8/logs`

In [30]:
```
trainer.save_model(model_path)
```

```
Saving model checkpoint to bigbird-roberta-base-4
Configuration saved in bigbird-roberta-base-4/config.json
Model weights saved in bigbird-roberta-base-4/pytorch_model.bin
tokenizer config file saved in bigbird-roberta-base-4/tokenizer_config.json
Special tokens file saved in bigbird-roberta-base-4/special_tokens_map.json
```

Test Data

In [31]:
```
def tokenize_for_test(examples):

    o = tokenizer(examples['text'], truncation=True, return_offsets_mapping=Tru

    offset_mapping = o["offset_mapping"]

    o["labels"] = []

    for i in range(len(offset_mapping)):

        labels = [l2i['O'] for i in range(len(o['input_ids'][i]))]

        for label_start, label_end, label in \
        list(zip(examples['starts'][i], examples['ends'][i], examples['classlis
            for j in range(len(labels)):
                token_start = offset_mapping[i][j][0]
                token_end = offset_mapping[i][j][1]
                if token_start == label_start:
                    labels[j] = l2i[f'B-{label}']
                if token_start > label_start and token_end <= label_end:
                    labels[j] = l2i[f'I-{label}']

        for k, input_id in enumerate(o['input_ids'][i]):
            if input_id in [0,1,2]:
                labels[k] = -100

        labels = fix_beginnings(labels)

        o["labels"].append(labels)
```

```
    return o
```

In [32]:
```
tokenized_test = datasets.map(tokenize_for_test, batched=True)
tokenized_test
```

```
  0%|          | 0/12 [00:00<?, ?ba/s]
  0%|          | 0/2 [00:00<?, ?ba/s]
```

Out[32]:
```
DatasetDict({
    train: Dataset({
        features: ['__index_level_0__', 'attention_mask', 'classlist', 'ends',
'id', 'input_ids', 'labels', 'offset_mapping', 'predictionstrings', 'starts',
'text'],
        num_rows: 11462
    })
    test: Dataset({
        features: ['__index_level_0__', 'attention_mask', 'classlist', 'ends',
'id', 'input_ids', 'labels', 'offset_mapping', 'predictionstrings', 'starts',
'text'],
        num_rows: 1274
    })
})
```

In [33]:
```
# ground truth for test data

l = []
for example in tokenized_test['test']:
    for c, p in list(zip(example['classlist'], example['predictionstrings'])):
        l.append({
            'id': example['id'],
            'discourse_type': c,
            'predictionstring': p,
        })

gt_df = pd.DataFrame(l)
```

In [34]:
```
path = Path('../input/feedback-prize-2021/train')

colors = {
            'Lead': '#8000ff',
            'Position': '#2b7ff6',
            'Evidence': '#2adddd',
            'Claim': '#80ffb4',
            'Concluding Statement': 'd4dd80',
            'Counterclaim': '#ff8042',
            'Rebuttal': '#ff0000',
            'Other': '#007f00',
         }

def visualize(df, text):
    ents = []
    example = df['id'].loc[0]

    for i, row in df.iterrows():
        ents.append({
                        'start': int(row['discourse_start']),
                         'end': int(row['discourse_end']),
                         'label': row['discourse_type']
                    })
```

```
    doc2 = {
        "text": text,
        "ents": ents,
        "title": example
    }

    options = {"ents": train.discourse_type.unique().tolist() + ['Other'], "col
    displacy.render(doc2, style="ent", options=options, manual=True, jupyter=Tr
```

In [35]:
```
predictions, labels, _ = trainer.predict(tokenized_test['test'])
```

```
The following columns in the test set  don't have a corresponding argument in
`BigBirdForTokenClassification.forward` and have been ignored: predictionstrin
gs, offset_mapping, __index_level_0__, text, starts, ends, id, classlist.
***** Running Prediction *****
  Num examples = 1274
  Batch size = 4
Attention type 'block_sparse' is not possible if sequence_length: 458 <= num g
lobal tokens: 2 * config.block_size + min. num sliding tokens: 3 * config.bloc
k_size + config.num_random_blocks * config.block_size + additional buffer: con
fig.num_random_blocks * config.block_size = 704 with config.block_size = 64, c
onfig.num_random_blocks = 3. Changing attention type to 'original_full'...
```


[319/319 00:45]

In [36]:
```
preds = np.argmax(predictions, axis=-1)
preds.shape
```

Out[36]:
```
(1274, 1925)
```

In [37]:
```
def get_class(c):
    if c == 14: return 'Other'
    else: return i2l[c][2:]

def pred2span(pred, example, viz=False, test=False):
    example_id = example['id']
    n_tokens = len(example['input_ids'])
    classes = []
    all_span = []
    for i, c in enumerate(pred.tolist()):
        if i == n_tokens-1:
            break
        if i == 0:
            cur_span = example['offset_mapping'][i]
            classes.append(get_class(c))
        elif i > 0 and (c == pred[i-1] or (c-7) == pred[i-1]):
            cur_span[1] = example['offset_mapping'][i][1]
        else:
            all_span.append(cur_span)
            cur_span = example['offset_mapping'][i]
            classes.append(get_class(c))
    all_span.append(cur_span)

    if test: text = get_test_text(example_id)
    else: text = get_raw_text(example_id)

    # abra ka dabra se soli fanta ko pelo
```

```
    # map token ids to word (whitespace) token ids
    predstrings = []
    for span in all_span:
        span_start = span[0]
        span_end = span[1]
        before = text[:span_start]
        token_start = len(before.split())
        if len(before) == 0: token_start = 0
        elif before[-1] != ' ': token_start -= 1
        num_tkns = len(text[span_start:span_end+1].split())
        tkns = [str(x) for x in range(token_start, token_start+num_tkns)]
        predstring = ' '.join(tkns)
        predstrings.append(predstring)

    rows = []
    for c, span, predstring in zip(classes, all_span, predstrings):
        e = {
            'id': example_id,
            'discourse_type': c,
            'predictionstring': predstring,
            'discourse_start': span[0],
            'discourse_end': span[1],
            'discourse': text[span[0]:span[1]+1]
        }
        rows.append(e)


    df = pd.DataFrame(rows)
    df['length'] = df['discourse'].apply(lambda t: len(t.split()))

    # short spans are likely to be false positives, we can choose a min number
    df = df[df.length > min_tokens].reset_index(drop=True)
    if viz: visualize(df, text)

    return df
```

In [38]:

```
dfs = []
for i in range(len(tokenized_test['test'])):
    dfs.append(pred2span(preds[i], tokenized_test['test'][i]))

pred_df = pd.concat(dfs, axis=0)
pred_df['class'] = pred_df['discourse_type']
pred_df
```

Out[38]:

	id	discourse_type	predictionstring	discourse_start	discourse_end	discourse
0	F4FD84517F40	Lead	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18...	0	375	Schools are offering distant learning for stud...
1	F4FD84517F40	Position	62 63 64 65 66 67 68 69 70 71 72 73 74 75	375	458	classes would help tons of students move clos...
2	F4FD84517F40	Claim	76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 9...	460	613	Most of the people failing classes or didn't g...
3	F4FD84517F40	Evidence	104 105 106 107 108 109 110 111 112 113 114 11...	615	1104	big example of this is working. They have to ...
4	F4FD84517F40	Rebuttal	202 203 204 205 206 207 208 209 210 211 212 21...	1150	1223	students grades are dropping due to the amoun...
...	...	...	...	...	...	...
0	58F8F0F77817	Position	0 1 2 3 4 5 6 7	0	48	There are many advantages of limiting car usage,
1	58F8F0F77817	Evidence	17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 3...	99	462	"Its a ooo opportunity to take away stress an...
2	58F8F0F77817	Evidence	94 95 96 97 98 99 100 101 102 103 104 105 106 ...	504	1021	can use someonjes elses gas instead of your o...
3	58F8F0F77817	Evidence	205 206 207 208 209 210 211 212 213 214 215 21...	1094	1463	people that i know get really flustered ehen ...
4	58F8F0F77817	Concluding Statement	276 277 278 279 280 281 282 283 284 285 286 28...	1486	1563	car usage is good because it save sgas and is...

12392 rows × 8 columns

In [39]:
```
def calc_overlap(row):
    #Calculates the overlap between prediction and ground truth and
    #overlap percentages used for determining true positives.
    set_pred = set(row.predictionstring_pred.split(" "))
    set_gt = set(row.predictionstring_gt.split(" "))
    # Length of each and intersection
    len_gt = len(set_gt)
    len_pred = len(set_pred)
    inter = len(set_gt.intersection(set_pred))
    overlap_1 = inter / len_gt
    overlap_2 = inter / len_pred
    return [overlap_1, overlap_2]


def score_feedback_comp_micro(pred_df, gt_df):
    gt_df = (
        gt_df[["id", "discourse_type", "predictionstring"]]
        .reset_index(drop=True)
        .copy()
    )
    pred_df = pred_df[["id", "class", "predictionstring"]].reset_index(drop=Tru
    pred_df["pred_id"] = pred_df.index
    gt_df["gt_id"] = gt_df.index
    # Step 1. all ground truths and predictions for a given class are compared.
    joined = pred_df.merge(
        gt_df,
        left_on=["id", "class"],
        right_on=["id", "discourse_type"],
        how="outer",
        suffixes=("_pred", "_gt"),
    )
    joined["predictionstring_gt"] = joined["predictionstring_gt"].fillna(" ")
    joined["predictionstring_pred"] = joined["predictionstring_pred"].fillna("

    joined["overlaps"] = joined.apply(calc_overlap, axis=1)

    # 2. If the overlap between the ground truth and prediction is >= 0.5,
    # and the overlap between the prediction and the ground truth >= 0.5,
    # the prediction is a match and considered a true positive.
    # If multiple matches exist, the match with the highest pair of overlaps is
    joined["overlap1"] = joined["overlaps"].apply(lambda x: eval(str(x))[0])
    joined["overlap2"] = joined["overlaps"].apply(lambda x: eval(str(x))[1])

    joined["potential_TP"] = (joined["overlap1"] >= 0.5) & (joined["overlap2"]
    joined["max_overlap"] = joined[["overlap1", "overlap2"]].max(axis=1)
    tp_pred_ids = (
        joined.query("potential_TP")
        .sort_values("max_overlap", ascending=False)
        .groupby(["id", "predictionstring_gt"])
        .first()["pred_id"]
        .values
    )

    # 3. Any unmatched ground truths are false negatives
    # and any unmatched predictions are false positives.
    fp_pred_ids = [p for p in joined["pred_id"].unique() if p not in tp_pred_ic
```

```
    matched_gt_ids = joined.query("potential_TP")["gt_id"].unique()
    unmatched_gt_ids = [c for c in joined["gt_id"].unique() if c not in matched

    # Get numbers of each type
    TP = len(tp_pred_ids)
    FP = len(fp_pred_ids)
    FN = len(unmatched_gt_ids)
    # calc microf1
    my_f1_score = TP / (TP + 0.5 * (FP + FN))
    return my_f1_score


def score_feedback_comp(pred_df, gt_df, return_class_scores=False):
    class_scores = {}
    pred_df = pred_df[["id", "class", "predictionstring"]].reset_index(drop=Tru
    for discourse_type, gt_subset in gt_df.groupby("discourse_type"):
        pred_subset = (
            pred_df.loc[pred_df["class"] == discourse_type]
            .reset_index(drop=True)
            .copy()
        )
        class_score = score_feedback_comp_micro(pred_subset, gt_subset)
        class_scores[discourse_type] = class_score
    f1 = np.mean([v for v in class_scores.values()])
    if return_class_scores:
        return f1, class_scores
    return f1
```

F-1 Score on Test Data

In [40]:
```
score_feedback_comp(pred_df, gt_df, return_class_scores=True)
```

Out[40]:
```
(0.5565781498148266,
 {'Claim': 0.5013676588897827,
  'Concluding Statement': 0.6702800361336947,
  'Counterclaim': 0.43243243243243246,
  'Evidence': 0.6306351183063512,
  'Lead': 0.7511139401654997,
  'Position': 0.5899053627760252,
  'Rebuttal': 0.3203125})
```

Demo - Visualising Predictions on Test Data

In [45]:
```
pred_df.head()
```

Out[45]:

	id	discourse_type	predictionstring	discourse_start	discourse_end	discourse
0	F4FD84517F40	Lead	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18...	0	375	Schools are offering distant learning for stud...
1	F4FD84517F40	Position	62 63 64 65 66 67 68 69 70 71 72 73 74 75	375	458	classes would help tons of students move clos...
2	F4FD84517F40	Claim	76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 9...	460	613	Most of the people failing classes or didn't g...
3	F4FD84517F40	Evidence	104 105 106 107 108 109 110 111 112 113 114 11...	615	1104	big example of this is working. They have to ...
4	F4FD84517F40	Rebuttal	202 203 204 205 206 207 208 209 210 211 212 21...	1150	1223	students grades are dropping due to the amoun...

In [42]:
```
pred2span(preds[0], tokenized_test['test'][0], viz=True)
```

F4FD84517F40

Schools are offering distant learning for students to attend classes off campus. Many students are saying this is great and they are taking the offer to continue and better their education. On the other hand, some say that this is not the best idea for schooling. Schools want to give students an opportunity to better their learning and stay connected with education. Online **Lead** classes would help tons of students move closer to graduation and a better future. **Position**

Most of the people failing classes or didn't graduate the first time around have things they have to do outside of school to help support their families. **Claim** A big example of this is working. They have to work to make money for their families but without all of the education they need, they can't work for a better paying job. Therefore, lots of people are working multiple jobs to support families; this is leaving them no time to spend six to eight hours of the day in a building. Another example is that nowadays many teens are having to stay home to watch their own child or watching a family member because of finances in their home. Although **Evidence** being home may help out with home situations, students grades are dropping due to the amount of absent days in classes **Rebuttal** . Online classes and video conferencing would benefit these students by allowing them to take classes at night and/or multitask during the day. **Evidence**

Another case where distant learning may come in use is for students very involved in sports. **Claim** Doing a sport can be a big responsibility. Sports are very time consuming, although some athletes make it look easy, there is a lot that goes into playing. Many levels of sports may require leaving the state or country for days, sometimes even weeks. While playing a sport it is easy to be injured or become sick. Furthermore, students are loosing class time and unable to make up for missed days in little time. Distant learning would be beneficial for those who travel so they can take their school work along and manage their free time. Online classes also help when they are sick or injured to give them time to rest and

heal and still not miss out on continuing to learning. **Evidence**

In conclusion, yes distant learning as an option to students is very beneficial. Allowing students to attend class from home or while traveling could get more and more people willing to go further in learning. **Concluding Statement**

Out[42]:

	id	discourse_type	predictionstring	discourse_start	discourse_end	discours
0	F4FD84517F40	Lead	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18...	0	375	Schools a offerir dista learning f stud
1	F4FD84517F40	Position	62 63 64 65 66 67 68 69 70 71 72 73 74 75	375	458	classes wou help tons studen move clos
2	F4FD84517F40	Claim	76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 9...	460	613	Most of th people failir classes didn't g
3	F4FD84517F40	Evidence	104 105 106 107 108 109 110 111 112 113 114 11...	615	1104	big examp of this workin They have
4	F4FD84517F40	Rebuttal	202 203 204 205 206 207 208 209 210 211 212 21...	1150	1223	studen grades a dropping du to th amoun
5	F4FD84517F40	Evidence	224 225 226 227 228 229 230 231 232 233 234 23...	1291	1366	by allowir them to tal classes night and
6	F4FD84517F40	Claim	237 238 239 240 241 242 243 244 245 246 247 24...	1368	1460	Another cas where dista learning ma come i
7	F4FD84517F40	Evidence	254 255 256 257 258 259 260 261 262 263 264 26...	1466	2140	a sport ca be a b responsibilit Sports a
8	F4FD84517F40	Concluding Statement	387 388 389 390 391 392 393 394 395 396 397 39...	2205	2351	ve beneficia Allowir students attend