Named-Entity-Recognition using a pre-trained BERT Model

1. **Data Preparation and Pre-processing**

In this project we have used the ***ConLL-2003*** dataset. We used the load_dataset() method from the Datasets library to load the dataset.
the dataset contains labels for three tasks: NER, POS, and chunking. To perform named entity recognition, we will look at the NER tags.
It has the following features:

tokens sequence	**pos_tags** sequence	**chunk_tags** sequence	**ner_tags** sequence
["EU", "rejects", "German", "call", "to", "boycott",…	[22, 42, 16, 21, 35, 37, 16, 21, 7]	[11, 21, 11, 12, 21, 22, 11, 12, 0]	[3, 0, 7, 0, 0, 0, 7, 0, 0]
["Peter", "Blackburn"]	[22, 22]	[11, 12]	[1, 2]

We extracted the NER features from the features attribute of the dataset and further retrieved labels using the names attribute of features.

```
raw_datasets = load_dataset("conll2003")

ner_feature = raw_datasets["train"].features["ner_tags"]

label_names = ner_feature.feature.names
```

We tokenized the pre-tokenized input, using a tokenizer. To match every token to its corresponding word, we give special tokens a label of -100. This is because by default -100 is an index that is ignored in the loss function (cross entropy). Then, each token gets the same label as the token that started the word it's inside since they are part of the same entity. For tokens inside a word but not at the beginning, we replace the B- with I- (since the token does not begin the entity)

Examples

*** Example ***
guid: dev-0
tokens: CR ##IC ##KE ##T - L ##EI ##CE ##ST ##ER ##S ##H ##IR ##E T ##A ##KE O ##VE ##R AT TO ##P A ##FT ##ER IN ##NI ##NG ##S VI ##CT ##OR ##Y .

input_ids: 101 15531 9741 22441 1942 118 149 27514 10954 9272 9637 1708 3048 18172 2036 157 1592 22441 152 17145 2069 13020 16972 2101 138 26321 9637 15969 27451 11780 1708 7118 16647 9565 3663 119 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
*** Example ***
guid: dev-1
tokens: L ##ON ##D ##ON 1996 - 08 - 30
input_ids: 101 149 11414 2137 11414 1820 118 4775 118 1476 102 0
*** Example ***
guid: dev-2
tokens: West Indian all - round ##er Phil Simmons took four for 38 on Friday as Leicestershire beat Somerset by an innings and 39 runs in two days to take over at the head of the county championship .
input_ids: 101 1537 1890 1155 118 1668 1200 5676 14068 1261 1300 1111 3383 1113 5286 1112 21854 3222 8860 1118 1126 6687 1105 3614 2326 1107 1160 1552 1106 1321 1166 1120 1103 1246 1104 1103 2514 2899 119 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
*** Example ***
guid: dev-3
tokens: Their stay on top , though , may be short - lived as title rivals Essex , Derbyshire and Surrey all closed in on victory while Kent made up for lost time in their rain - affected match against Nottinghamshire .
input_ids: 101 2397 2215 1113 1499 117 1463 117 1336 1129 1603 118 2077 1112 1641 9521 8493 117 15964 1105 9757 1155 1804 1107 1113 2681 1229 5327 1189 1146 1111 1575 1159 1107 1147 4458 118 4634 1801 1222 21942 119 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

2. Ground Truth

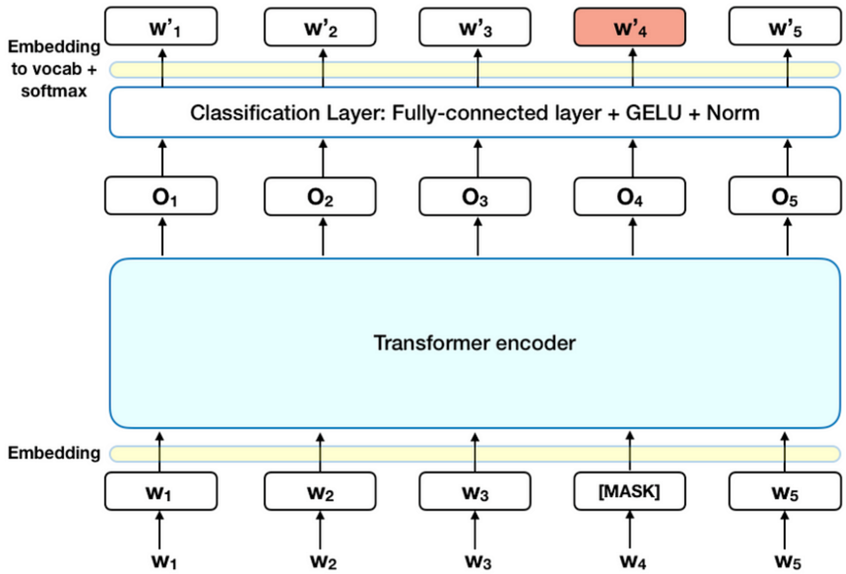
The labels for the ConLL-2003 dataset are:
['O', 'B-PER', 'I-PER', 'B-ORG', 'I-ORG', 'B-LOC', 'I-LOC', 'B-MISC', 'I-MISC']

Example
Text: EU rejects German call to boycott British lamb .
Labels: B-ORG O B-MISC O O O B-MISC O O

3. Network Details

BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection.
As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it

is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).



In the implementation of bert-ner, we have used customized training to change the hyperparameters and analyze the effect on various intrinsic and extrinsic evaluation metrics. Details for the same are given below:

```
<bound method Module.parameters of BertForTokenClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(28996, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
```

```
            (value): Linear(in_features=768, out_features=768, bias=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (output): BertSelfOutput(
            (dense): Linear(in_features=768, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=768, out_features=3072, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=3072, out_features=768, bias=True)
          (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
    )
  )
 )
 (dropout): Dropout(p=0.1, inplace=False)
 (classifier): Linear(in_features=768, out_features=9, bias=True)
)>
```

Zero-shot classification

For the zero shot classification, we took the following values:

1) Text: Djokovic has qualified for his sixth French Open final!

Predicted Labels:

- Environment: 0.36418867111206055
- Politics: 0.32020843029022217
- Sport: 0.3156028687953949

2) Text: On March 23, 2010, President Obama signed the Affordable Care Act into law, putting in place comprehensive reforms that improve access to affordable health coverage for everyone and protect consumers from abusive insurance company practices
Predicted Labels:
- Environment: 0.34412556886672974
- Sport: 0.3284330666065216
- Politics: 0.3274413049221039

3) Text: The goal was to show that scientists from various disciplines, diverse cultures and countries at different stages of development could find common ground about the conditions for triggering climate action in the current economic context
Predicted Labels:
- Environment: 0.34378793835639954
- Sport: 0.3305787146091461
- Politics: 0.32563337683677673

Fine-tuning

Evaluation metrics for different hyperparameters, Loss functions, and Optimizers:

```
Extrinsic evaluation using f1-score
Intrinsic evaluation using accuracy

 ***** Eval results *****
```

Loss = CrossEntropyLoss

Optimizer = AdamW

Learning rate = 5e-5

Hidden_layer size = 768

Regularization constant = 0.01

Epochs = 1

Number of attention heads = 12

	precision	recall	f1-score	support
ORG	0.8977	0.9292	0.9132	1341
LOC	0.9563	0.9521	0.9542	1837

	precision	recall	f1-score	support
MISC	0.8530	0.8623	0.8576	922
PER	0.9668	0.9658	0.9663	1842
avg / total	0.9303	0.9372	0.9337	5942

Accuracy: 0.9849

Loss = CrossEntropyLoss

Optimizer = SGD

Learning rate = 0.01

Hidden_layer size = 750

Regularization constant = 0.02

Epochs = 2

Number of attention heads = 5

	precision	recall	f1-score	support
ORG	0.9014	0.9135	0.9074	1341
MISC	0.8084	0.8514	0.8294	922
LOC	0.9459	0.9423	0.9441	1837
PER	0.9597	0.9696	0.9646	1842
avg / total	0.9188	0.9302	0.9244	5942

Accuracy: 0.9368

Loss = NLLLoss

Optimizer = AdamW

Learning rate = 5e-5

Hidden_layer size = 768

Regularization constant = 0.02

Epochs = 1

Number of attention heads = 12

	precision	recall	f1-score	support
ORG	0.5906	0.3184	0.4138	1341
MISC	0.0000	0.0000	0.0000	922

```
         LOC      0.5905    0.8258    0.6886       1837
         PER      0.8834    0.9463    0.9138       1842

avg / total       0.5897    0.6205    0.5895       5942
Accuracy:  0.6307
```

Single training item, mini-batch training and training after freezing 3 layers have been done in the Colab Notebook attached, the metrics obtained from the same are:

```
Optimizer ADAMW:

epoch 0: {'precision': 0.9397509256142713, 'recall': 0.9124183006535947,
'f1': 0.9258829381528767, 'accuracy': 0.9835462412433037}

epoch 1: {'precision': 0.9449680242342645, 'recall': 0.9268735556289205,
'f1': 0.9358333333333334, 'accuracy': 0.9849885206334256}

epoch 2: {'precision': 0.9508582968697409, 'recall': 0.9285127362366475,
'f1': 0.9395526731520745, 'accuracy': 0.9864308000235474}
```

Optimizer: SGD

```
epoch 0: {'precision': 0.8961629081117469, 'recall': 0.8686786296900489,
'f1': 0.8822067594433399, 'accuracy': 0.973832931064932}

epoch 1: {'precision': 0.9306630764052508, 'recall': 0.9035947712418301,
'f1': 0.916929199137788, 'accuracy': 0.9823541531759581}

epoch 2: {'precision': 0.9441265567149109, 'recall': 0.9205776173285198,
'f1': 0.9322033898305085, 'accuracy': 0.9850179549066933}
```

Submitted By:

Ayush Gupta (2020B3A70838P)

Ria Shekhawat (2020B4A71986P)