Computational Linguistics for Indian Languages

(CS689A)

Assignment 2

SUBMITTED BY:

DHIRAJ PAREEK
231110012

Question 2:

Comparison between Indic-Bert and Indic-NER

Validation set:

Indic-BERT: Macro F1 score = 0.6722
Indic-NER: Macro F1 score = 0.7761

Test set:

Indic-BERT: Macro F1 score = 0.6947
Indic-NER: Macro F1 score = 0.7997

Indic-NER outperforms Indic-BERT due to following reasons:

1. **Specialisation**: Indic-NER is specifically designed and fine-tuned for the NER task on Indic languages. It focuses only on recognizing named entities like people, places, organisations, etc. within text. This specialisation allows it to be more effective.
2. **Efficiency** : Indic-NER have a simpler and more efficient architecture tailored for the NER task. It can be optimised to efficiently identify named entities in Indic text, leading to faster inference and better performance compared to Indic-BERT, which has a more general architecture.
3. **Domain-specific Knowledge** : Indic-NER may incorporate domain-specific knowledge or features relevant to the NER task in Indic languages. This domain-specific information helps the model better understand and recognize named entities in Indic text, giving it an edge over the more general Indic-BERT model.

Question 4

IndicNER and Ground truth

```
Precision of  B-PER : 0.3333333333333333
Recall of  B-PER : 1.0
f1_score of  B-PER : 0.5

Precision of  I-PER : 0.0
Recall of  I-PER : 0.0
f1_score of  I-PER : 0.0

Precision of  B-LOC : 0.46153846153846156
Recall of  B-LOC : 0.8571428571428571
f1_score of  B-LOC : 0.6

Precision of  I-LOC : 1.0
Recall of  I-LOC : 0.3333333333333333
f1_score of  I-LOC : 0.5

Precision of  B-ORG : 0.375
Recall of  B-ORG : 1.0
f1_score of  B-ORG : 0.5454545454545454

Precision of  I-ORG : 0.2
Recall of  I-ORG : 1.0
f1_score of  I-ORG : 0.3333333333333337

Precision of  B-MISC : 0.0
...
Recall of  O : 0.8984126984126984
f1_score of  O : 0.8775193798449613

Macro F1 Score : 0.37292302873698224
```

IndicBERT and Ground truth

```
Precision of  B-PER : 0.3333333333333333
Recall of  B-PER : 0.5
f1_score of  B-PER : 0.4

Precision of  I-PER : 0.0
Recall of  I-PER : 0.0
f1_score of  I-PER : 0.0

Precision of  B-LOC : 0.5
Recall of  B-LOC : 0.5714285714285714
f1_score of  B-LOC : 0.5333333333333333

Precision of  I-LOC : 0.0
Recall of  I-LOC : 0.0
f1_score of  I-LOC : 0.0

Precision of  B-ORG : 0.0
Recall of  B-ORG : 0.0
f1_score of  B-ORG : 0.0

Precision of  I-ORG : 0.0
Recall of  I-ORG : 0.0
f1_score of  I-ORG : 0.0

Precision of  B-MISC : 0.0
...
Recall of  O : 0.9142857142857143
f1_score of  O : 0.8834355828220858

Macro F1 Score : 0.2018632129061577
```

ChatGPT and ground truth

```
Precision of  B-PER : 1.0
Recall of  B-PER : 0.5
f1_score of  B-PER : 0.6666666666666666

Precision of  B-LOC : 0.5
Recall of  B-LOC : 0.6666666666666666
f1_score of  B-LOC : 0.5714285714285715

Precision of  O : 0.9551282051282052
Recall of  O : 0.873900293255132
f1_score of  O : 0.9127105666156202

Macro F1 Score : 0.23897842274565095
```

Macro F1 score of IndicBERT and Chatgpt is low as they are not recognising the B-MISC and I-MISC classes. Apart from that the 25 sentences does not include cases for I-ORG and I-PER so f1 score for these classes is 0 and making the overall average low.

Question 5:

HyperParameters which I have tuned are *learning rate* and *per device train batch size and per device eval batch size*

Per_device_train_batch_size:

It determines the number of training inputs processed on CPU or GPU at a time during training. It affects the efficiency of training. A larger batch size can lead to faster training times but may require more memory. On the other hand, a smaller batch size can improve generalisation but might slow down training.
The optimal value for per_device_train_batch_size depends on available memory, model architecture, and dataset size.

per_device_eval_batch_size:

Similar to per_device_train_batch_size, this hyperparameter determines the number of evaluation samples processed simultaneously on each device during evaluation.

learning_rate:

Learning rate controls the step size during the optimization process like gradient descent. It determines how much the model's parameters are adjusted in response to the estimated error each time the model is updated.
The optimal learning rate depends on factors such as the model architecture, dataset complexity, and optimization algorithm. It's typically chosen through hyperparameter tuning techniques aiming to find the learning rate that achieves the best balance between convergence speed and stability.

The Optimal values I have chosen are:

Learning rate = 5e-6
per device train batch size=6
per device eval batch size=6

The model is providing better results when fine tuned on these values of hyperparameters giving macro f1 score 0.405989

IndicNER (arguments 1)

```
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-5)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.153100	0.170914	0.810615	0.855380	0.832396	10213	0.683332	0.677396	0.680351	9786	0.806352	0.838475	0.822100	10568	0.769886	0.792554	0.781056	0.947092
2	0.103200	0.182538	0.817916	0.850191	0.833741	10213	0.673960	0.698753	0.686133	9786	0.804627	0.829296	0.816775	10568	0.767202	0.794484	0.780605	0.946835
3	0.074200	0.207841	0.811579	0.852345	0.831463	10213	0.671473	0.685265	0.678299	9786	0.800625	0.824186	0.812235	10568	0.763516	0.789119	0.776106	0.945863

IndicNER (arguments 2)

```
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=6,
    per_device_eval_batch_size=6,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-6)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.165600	0.182068	0.806988	0.845785	0.825931	10213	0.671824	0.696607	0.683991	9786	0.800831	0.838948	0.819446	10568	0.761888	0.795662	0.778409	0.947476
2	0.139600	0.177524	0.813711	0.855380	0.834025	10213	0.678192	0.697527	0.687724	9786	0.801950	0.840651	0.820844	10568	0.766837	0.799751	0.782948	0.948311
3	0.133600	0.178703	0.814278	0.857730	0.835439	10213	0.676578	0.699877	0.688031	9786	0.800974	0.840083	0.820063	10568	0.766057	0.801093	0.783183	0.948176

IndicNER (arguments 3)

```
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=10,
    per_device_eval_batch_size=10,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-6)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.181000	0.188524	0.787736	0.836483	0.811378	10213	0.657171	0.688330	0.672390	9786	0.794419	0.835163	0.814282	10568	0.748486	0.788596	0.768018	0.945695
2	0.149100	0.180712	0.809275	0.849212	0.828763	10213	0.665885	0.696505	0.680851	9786	0.798040	0.839799	0.818387	10568	0.759618	0.797069	0.777893	0.947548
3	0.140500	0.180156	0.810160	0.849506	0.829366	10213	0.670867	0.696301	0.683348	9786	0.798739	0.838948	0.818350	10568	0.761935	0.796807	0.778981	0.947696

IndicBERT
(Arguments 1)

```
batch_size=8
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-5)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.316400	0.297168	0.660154	0.688143	0.673858	10213	0.610167	0.403536	0.485792	9786	0.695213	0.618376	0.654547	10568	0.660382	0.572905	0.613541	0.910074
2	0.236200	0.260830	0.723383	0.715167	0.719252	10213	0.577652	0.532495	0.554155	9786	0.734257	0.671934	0.701715	10568	0.681371	0.641738	0.660961	0.920068
3	0.201900	0.258878	0.716024	0.737687	0.726694	10213	0.580255	0.553750	0.566693	9786	0.717626	0.711582	0.714591	10568	0.674786	0.669775	0.672271	0.921359

(Arguments 2)

```
batch_size=10
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-7)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.171900	0.261670	0.717105	0.736414	0.726632	10213	0.581300	0.552728	0.566654	9786	0.714773	0.713285	0.714029	10568	0.674643	0.669611	0.672118	0.921297
2	0.170500	0.262180	0.715469	0.735925	0.725553	10213	0.577569	0.552013	0.564502	9786	0.714651	0.713096	0.713873	10568	0.672763	0.669153	0.670953	0.921067
3	0.176300	0.262522	0.717559	0.735827	0.726578	10213	0.576587	0.553137	0.564619	9786	0.712883	0.714232	0.713556	10568	0.672469	0.669873	0.671168	0.921047

(Arguments 3)

```
args=TrainingArguments(
    output_dir='output_dir',
    per_device_train_batch_size=10,
    per_device_eval_batch_size=10,
    num_train_epochs=3,
    evaluation_strategy = "epoch",
    learning_rate=5e-6)
```

Epoch	Training Loss	Validation Loss	Loc Precision	Loc Recall	Loc F1	Loc Number	Org Precision	Org Recall	Org F1	Org Number	Per Precision	Per Recall	Per F1	Per Number	Overall Precision	Overall Recall	Overall F1	Overall Accuracy
1	0.471800	0.441183	0.497876	0.436111	0.464951	10213	0.318898	0.198651	0.244805	9786	0.510353	0.501419	0.505847	10568	0.460059	0.382668	0.417810	0.870880
2	0.385000	0.388195	0.565406	0.485851	0.522618	10213	0.393599	0.354384	0.372963	9786	0.609916	0.517979	0.560201	10568	0.523455	0.454870	0.486758	0.884538
3	0.361400	0.377335	0.550163	0.543915	0.547021	10213	0.417843	0.357041	0.385056	9786	0.616870	0.532173	0.571400	10568	0.532093	0.480027	0.504721	0.887121

Conclusion:

Comparison between two models showed that indicNER works better than IndicBERT for the task of Named entity recognition and also we require cross validation to find the optimal hyperparameters .