# Question 5:

### Hyper parameters tuning:

**Learning Rate:** The learning rate determines how quickly or slowly the model converges to the optimal solution. High learning rate can cause the model to converge quickly, but it may also lead to overshooting the optimal solution. While low learning rate may result in slow convergence or getting stuck in local minima. I have kept learning rate 5e-5 for training the model

**Batch size:** Smaller batch sizes introduce randomness into the training process and prevent the model from memorizing the training data. Larger batch sizes can lead to more stable training, because gradient estimates are based on more samples and are therefore less noisy. I have used batch size of 16 for training indicNER and batch size of 64 for indicBERT model training.

**Number of Epochs:** The number of epochs means how many times entire dataset is passed forward and backward through the neural network during training. Each epoch consists of one forward pass and one backward pass of all the training examples. I have used number_of_epochs =5 for training of both the models.

**Optimizer:** Optimizer determines how the model's weights are updated during training. I am using Adamw optimizer for training of both indicBERT and indicNER model.

# Test results:

**1. indicBERT:**

**1) On test dataset:**

Classwise_f1_score= {O: 0.9554179566563468, B-PER: 0.859106529209622, I-PER: 0.9016786570743405, B-ORG: 0.7557726465364121, I-ORG: 0.5975869410929738, B-LOC: 0.8183632734530938, I-LOC: 0.5951417004048584}

"macro_f1_score": 0.7832953863468068,

"recall": 0.765376395155545,

"precision": 0.7989588497768964

**2) On train dataset:**

Classwise_f1_score= {O: 0.9686426522788432, B-PER: 0.8921617171929739, I-PER: 0.8951185252990751, B-ORG: 0.8329033663270492, I-ORG: 0.8061137466019935, B-LOC: 0.878320986592407, I-LOC: 0.7152514365118113}

"macro_f1_score": 0.8555017758291648,

"recall": 0.8509329698922067,

"precision": 0.83821896106876

**3) On validation dataset:**

Classwise_f1_score= {O: 0.9474071180755003, B-PER: 0.8300395256916996, I-PER: 0.8551964512040556, B-ORG: 0.7313139260424862, I-ORG: 0.6371787208607291, B-LOC: 0.8034852964057881, I-LOC: 0.5743670886075949}

"macro_f1_score": 0.7832953863468068,

"recall": 0.7684268752696933,

"precision": 0.754344953755463

**2. IndicNER:**

**1) On test dataset:**

Classwise_f1_score= {O: 0.9548410404624277, B-PER: 0.8845680531626876, I-PER: 0.9123519458544839, B-ORG: 0.7418723070896983, I-ORG: 0.5952813067150635, B-LOC: 0.8419161676646706, I-LOC: 0.6366782006920415}

"macro_f1_score": 0.7953584316630105,

"recall": 0.7867862015201715,

"precision": 0.8265765765765766

**2) On train dataset:**

Classwise_f1_score= {O: 0.9892231627328928, B-PER: 0.9640262814694017, I-PER: 0.9642838830238772, B-ORG: 0.9411451328361077, I-ORG: 0.9496380269559347, B-LOC: 0.9556336473983533, I-LOC: 0.9057173958959529}

"macro_f1_score": 0.9528096471875028,

"recall": 0.9434294550852584,

"precision": 0.940665824025164

**3) On validation dataset:**

Classwise_f1_score= {O: 0.9556310403937365, B-PER: 0.8698203153648698, I-PER: 0.8820809248554914, B-ORG: 0.7619220968329087, I-ORG: 0.7129174216929848, B-LOC: 0.8315162087512721, I-LOC: 0.6593856655290102}

"macro_f1_score": 0.7953584316630105,

"recall": 0.8104676676314676,

"precision": 0.8021912688975709

**3. IndicBERT prediction vs manually written ner_tags (on 25sentences)**

Classwise_f1_score= {O: 0.9472182596291013, B-PER: 0.7142857142857142, I-PER: 0.9333333333333333, B-ORG: 0.6, I-ORG: 0.42857142857142855, B-LOC: 0.6428571428571429, I-LOC: 0.2222222222222222, B-MISC: 0.0, I-MISC: 0.0}

"macro_f1_score": 0.4987209000998825,

"recall": 0.4771845675691829,

"precision": 0.868360175178357

**4. IndicNER predictions vs manually answered ner_tags (on 25sentences)**

Classwise_f1_score= {O: 0.9644970414201184, B-PER: 0.7500000000000001, I-PER: 0.9090909090909091, B-ORG: 0.6666666666666667, I-ORG: 0.5714285714285715, B-LOC: 0.7692307692307693, I-LOC: 0.0, B-MISC: 0.0, I-MISC: 0.0}

"macro_f1_score": 0.5145459953152262,

"recall": 0.5044344487769564,

"precision": 0.8958113943787297

**5. Chatgpt predictions vs Manually answered ner_tags (on 25 sentences)**

Classwise_f1_score= {O: 0.9435483870967744, B-PER: 0.8148148148148148, I-PER: 0.7999999999999999, B-ORG: 0.625, I-ORG: 0.42857142857142855, B-LOC: 0.31578947368421056, I-LOC: 0.2, B-MISC: 0.0, I-MISC: 0.0}

"macro_f1_score": 0.45863601157413647,

"recall": 0.41634387269980494,

"precision": 0.6317460317460317

## My learnings through these comparisons:

As Indic-NER is specifically designed for named entity recognition tasks, while Indic-BERT is a more general-purpose language model. So, the architecture of Indic-NER is more suited to capturing the specific patterns and features relevant to NER.

Even though both models were fine-tuned on the same Marathi naampadam, initial parameters of models before fine-tuning are different. Indic-NER has been initialized in a way that is more conducive to NER tasks. So, I think this gave it advantage during fine-tuning over indic-BERT model

Also, Indic-NER might be a simpler model compared to Indic-BERT, because it is designed only for NER task, but indic-BERT is designed for multiple tasks so, this makes Indic-NER easier to fine-tune and result in better generalization to the NER task.