**Comparison of Both the models(INDIC BERT and INDIC NER):-  
-** Both indic BERT and indic NER models are designed for natural language processing tasks with a focus on Indic languages:

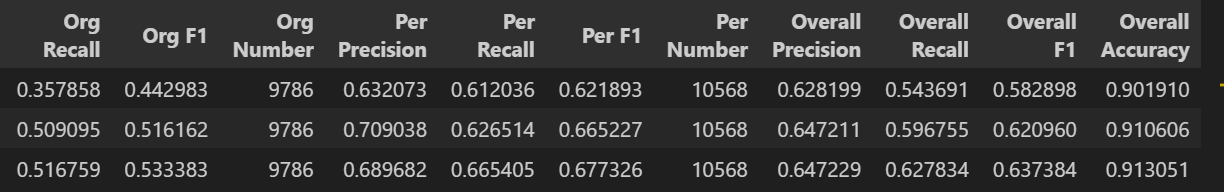
On one hand indic BERT is used for various NLP tasks such as text classification, named entity recognition, question answering, and more, while on the other hand indic NER is specifically designed for the task of Named Entity Recognition, which involves identifying and classifying entities such as persons, organizations, locations, dates, etc., within a given text.

Both the models are focused on processing Indic languages, Indic BERT is a versatile language representation model suitable for various NLP tasks, while Indic NER specifically targets the task of named entity recognition within Indic language text.

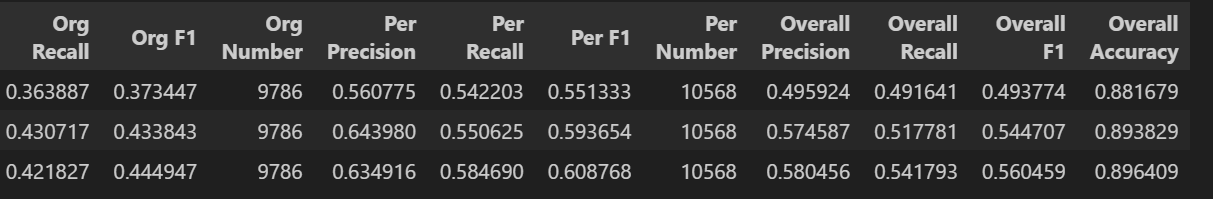
**Validation Set :-**

Macro F1 score for indicBERT :-

0.637384(for batch size 8, 3 epochs, learning rate 2e-5)



0.560459(for batch size 16, 2 epochs, learning rate 1e-5)

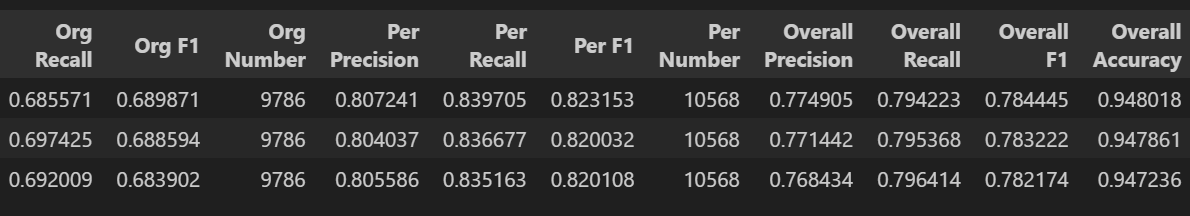


0.625391(for batch size 8, 2 epochs, learning rate 2e-5)



Macro F1 score for indicNER :-

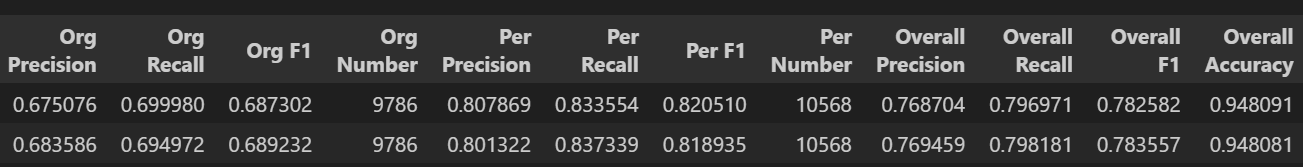
0.782174(for batch size 8, 3 epochs, learning rate 2e-5)



0.782065(for batch size 16, 3 epochs, learning rate 1e-5),



0.783557(for batch size 8, 2 epochs, learning rate 2e-5)



**Test Set :-**

Macro F1 score for indicBERT :-

0.6506229277705257(for batch size 8, 3 epochs, learning rate 2e-5)



0.568932028808008(for batch size 16, 2 epochs, learning rate 1e-5)



0.6448631705251608(for batch size 8, 2 epochs, learning rate 2e-5)



Macro F1 score for indicNER :-

0.7968405085841638(for batch size 8, 3 epochs, learning rate 2e-5)



0.8015636145555458(for batch size 16, 3 epochs, learning rate 1e-5),



0.8071999774732174(for batch size 8, 2 epochs, learning rate 2e-5)



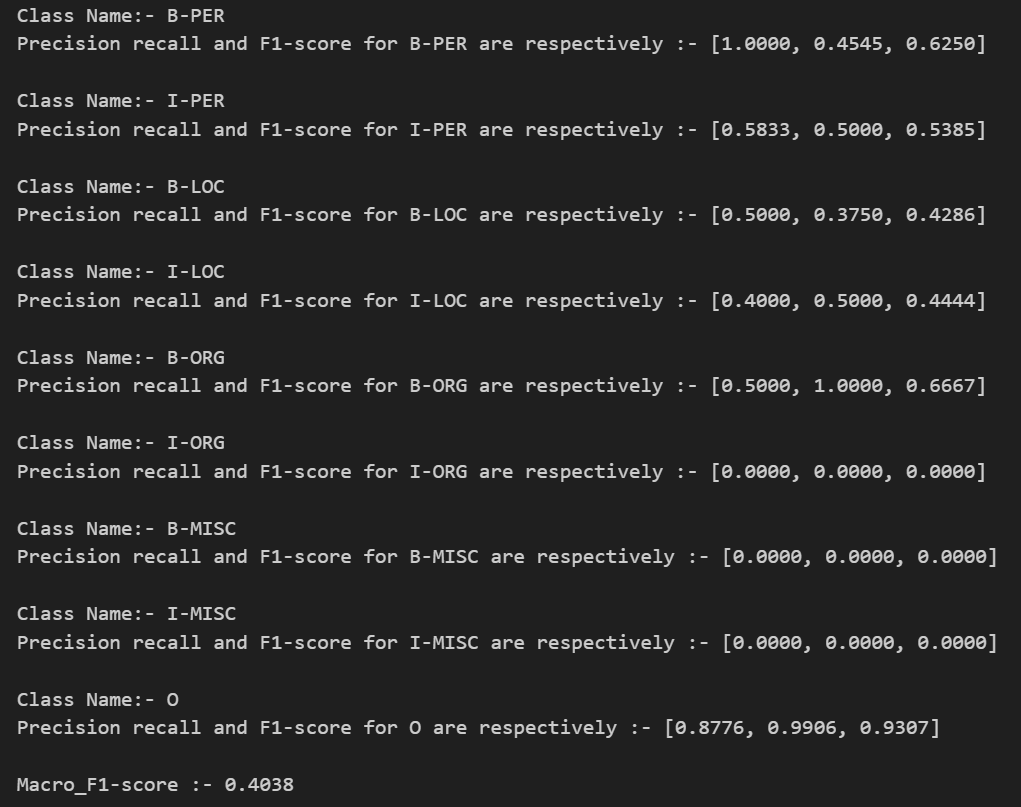
**-One of the major attention we can draw from here is that:-**

Even after adjusting epochs, batch size along with learning rates, the NER model consistently yielding the most similar scores. This suggests that NER model converges very fast on the corpus and its because indicNER is specifically made or designed for Name entity Recognition tasks

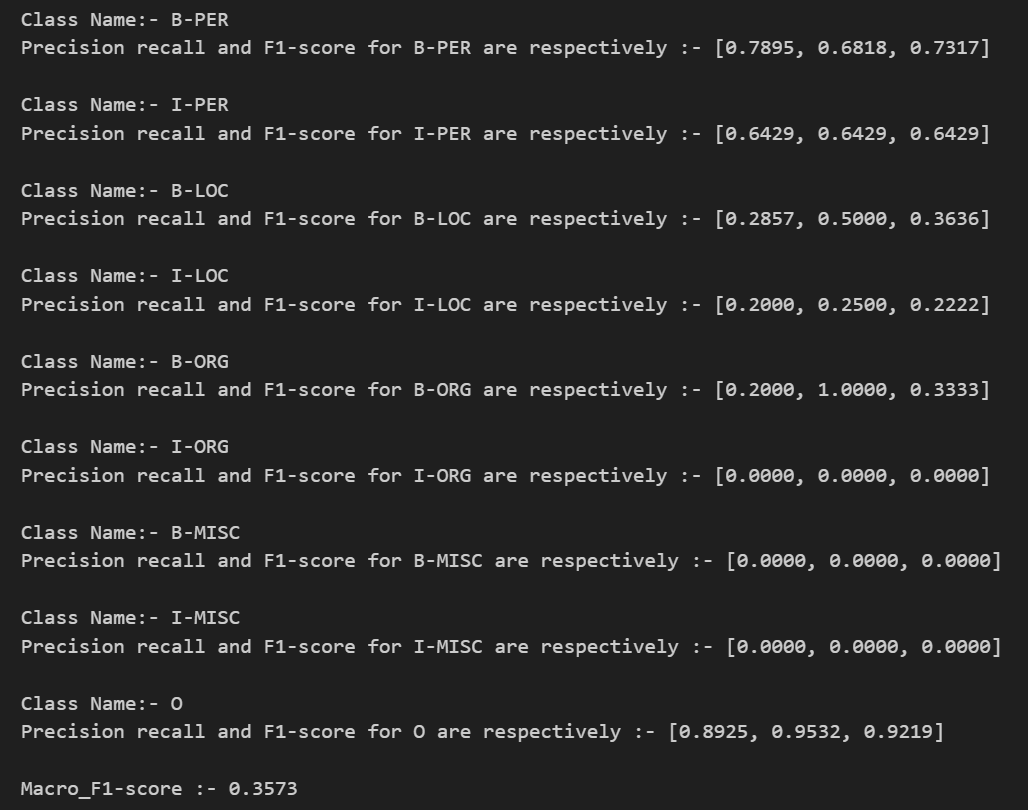
-Here we can see that IndicNER outperforms IndicBERT in named entity recognition (NER) tasks, as we can also see the evidence by its higher macro F1 score that indicBERT. This performance difference is attributed to IndicNER's specialized architecture and training methodologies optimized for NER, allowing it to capture NER-related nuances better than IndicBERT. While IndicBERT is a strong language model suitable for various NLP tasks, including NER, its general-purpose nature might not be as effective as a model explicitly designed for NER like IndicNER. I have also mentioned above that both model focus on indic languages but for NER indicNER is trained well so, task-specific models like IndicNER may achieve superior performance in their respective domains compared to more general models like IndicBERT.

**Below are the Question-4 output of precision recall and F1- score output matrics for:-**

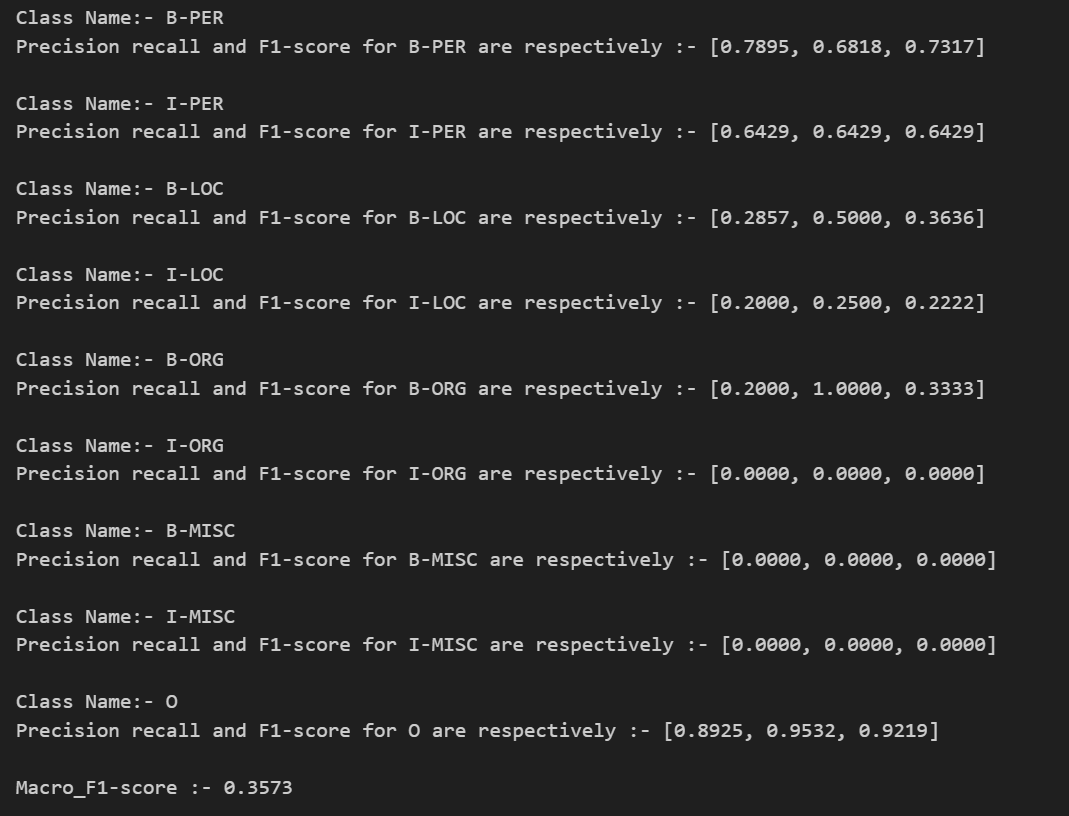
**1) manually found tags and chatGPT generated tags(**considering manually marked tags as ground truth**):-**

****

**2) manually found tags and indicBERT generated tags(**considering manually marked tags as ground truth**):-**

****

**3) manually found tags and chatGPT generated tags(**considering manually marked tags as ground truth**):-**

****

**Learnings From the Comparison :-**

Here our main aim is to evaluate the performance and effectiveness of these models in the context of named entity recognition (NER) tasks, also we have already seen that indicNER is already trained for such type of tasks By comparing the outputs generated by these models with manually generated tags, we can assess their accuracy, precision, recall, and overall performance in identifying and classifying named entities within text data.

- As I have already mentioned above, the above given data reveals that the NER model consistently converges quickly on the corpus, indicating its proficiency in named entity recognition. IndicNER surpasses IndicBERT in this task, showcasing its specialized architecture and training optimized for NER. This superiority is evident in IndicNER's higher macro F1 score, emphasizing the effectiveness of task-specific models like IndicNER for named entity recognition in Indic languages.  
  
- From the above comparisons we can also say that accuracy of our model and precision of predicting the tags highly depends on parameters like learning rate and batch size as increasing the batch size decreases the precision of predictions made by the models and smaller learning rate gives better accuracy as compared with increasing the learning rate.

So i decided to tune learning rate, batch size and epoch values

**Tuning Hyper parameters to improve the performance of models:**

**Batch size** - as i have mentioned in above paragraph i decided to tune batch size because these parameters define the number of training samples and evaluation samples, respectively processed simultaneously on each device (GPU or CPU) during training. Larger batch sizes require more memory, and if the batch size exceeds the available memory, training may fail. Smaller batch sizes generalize better and help prevent overfitting, especially with limited data.

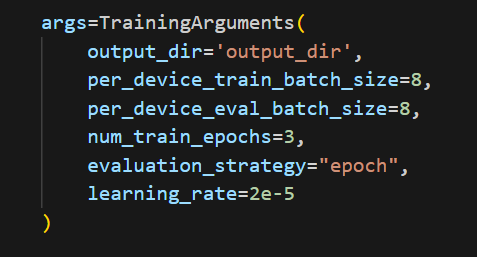
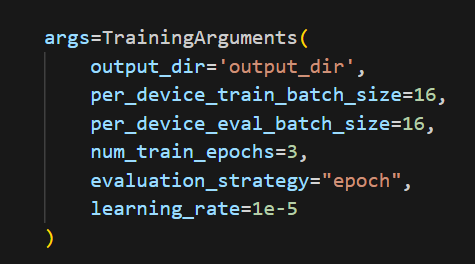
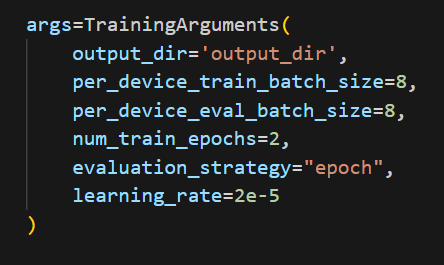
**epochs value** - very first thought to change hyperparameter is of epoch size as It directly controls how many times the model iterates over the entire training dataset. Increasing this parameter allows the model to see the training data more times, potentially leading to better convergence and improved performance.

**Learning rate** - As we know that learning rate determines the size of the step taken in the direction opposite to the gradient during optimization. A higher learning rate means larger steps, potentially leading to faster convergence but with the risk of overshooting the optimal solution

After considering all the tuning i have chosen following arguments for my model which i will use for output generation: 1) per\_device\_train\_batch\_size - 8 2) Per\_device\_eval\_batch\_size - 8

3) Num\_train\_epochs - 3 4) Learning\_rate - 2e-5

Below are the Screenshots of All the hyperparameter tuning i have done for indicBERT and same for indicNER:-



**Conclusion :-**  
In Conclusion i want to say that i have trained both of the model on different different sizes(here by sizes i mean different number of lines of corpus) but what i have found is that ➖

- Focusing on the learning rate, I have found that accuracy increases as we decrease the size of learning rate and also reverse happens when we increase learning rate.

- As we increase the batch size precision decreases of the models also it doesnt affect much on indicNER

- indicNER is built and designed for named entity recognition so it is the most fit choice to use for finding NER tags for word of the sentences given to us in form of tags

- We are able to train indicNER very quickly and proceed with NER for texts.