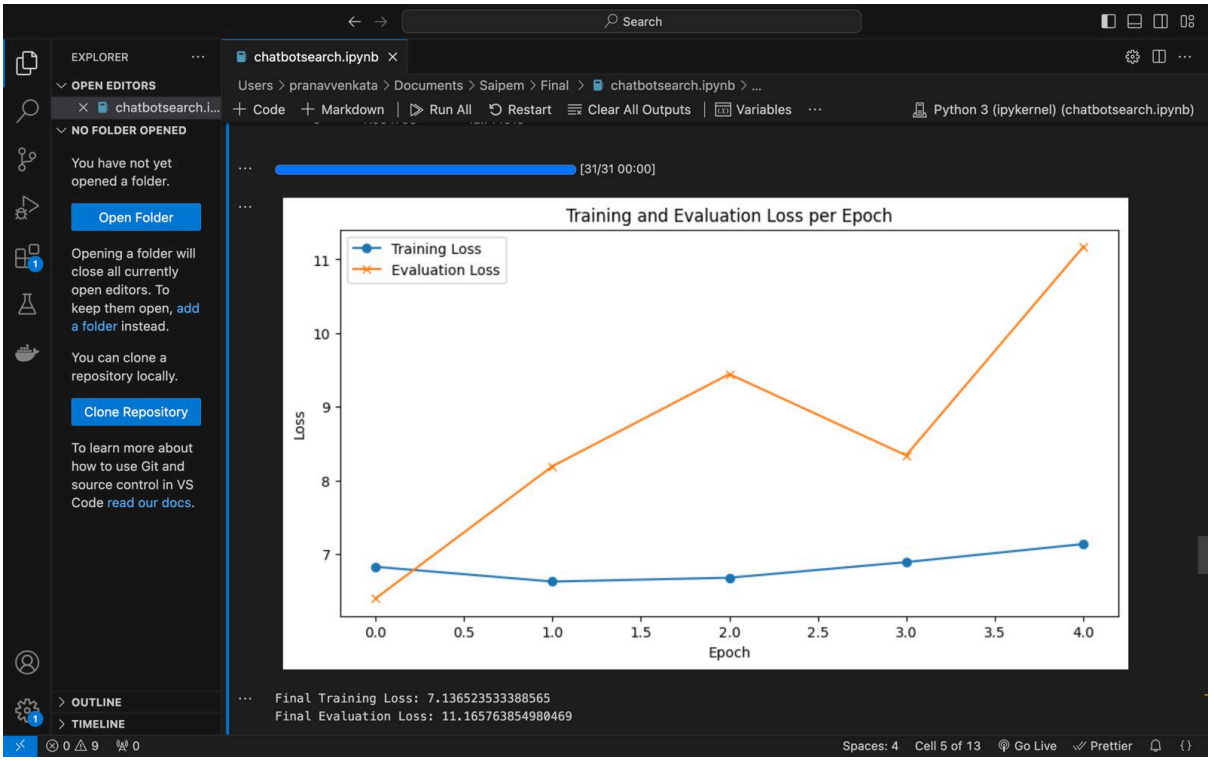
DATA ANNOTATION AND TESTING

1. Initially tested using the untrained BERT model for entity extraction and matching user input with given data using these entities and keywords that were extracted.
2. During the process, we also managed to classify user intent, but the accuracy was not up to the mark.
3. Later, we tried an ML approach to tokenize the data. This approach also included filters, intent classification and similarity matching using the user query.
4. Furthermore, the BERT model code was optimised by listing the queries and intent separately and training the model.
5. Next step, we tried implementing spaCy library using a trained RoBERTa model. But we encountered problems trying to annotate the data.
6. Decided on finding a common annotator that can annotate the data according to specified parameters.
7. Label-studio was chosen as the annotator since it was open-source and provided full-range usage.
8. We specified the parameters
    a. “Type”,
    b. “Discipline”,
    c. “Month”,
    d. “Day”,
    e. “Month to Month”, and performed annotation for the given data.

   Code snippet for label-studio interface:-

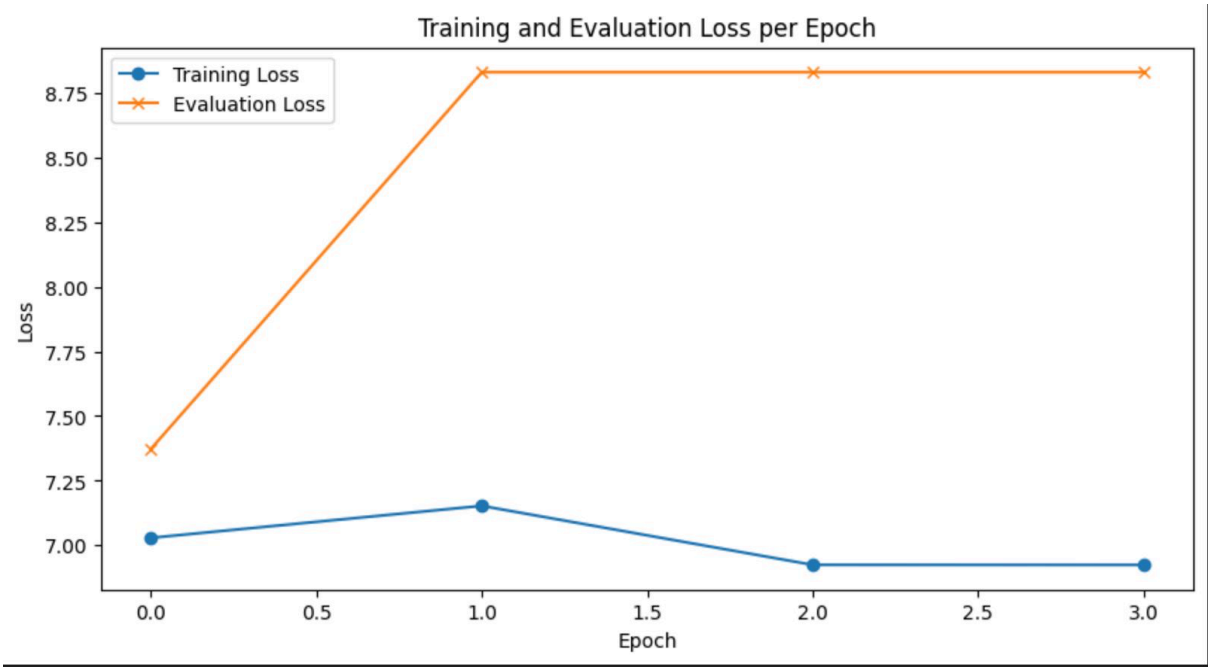
```
<View>
  <Text name="text" value="$text"/>
  <Labels name="label" toName="text">
    <Label value="Discipline" background="green"/>
    <Label value="Month" background="orange"/>
    <Label value="Type" background="brown"/>
    <Label value="Day" background="blue"/>
    <Label value="Month to Month" background="red"/>
    <!-- Add more entities as needed -->
  </Labels>
</View>
```

9. Annotated approximately up to 1000 fields of data, but this led to an overfitting issue.

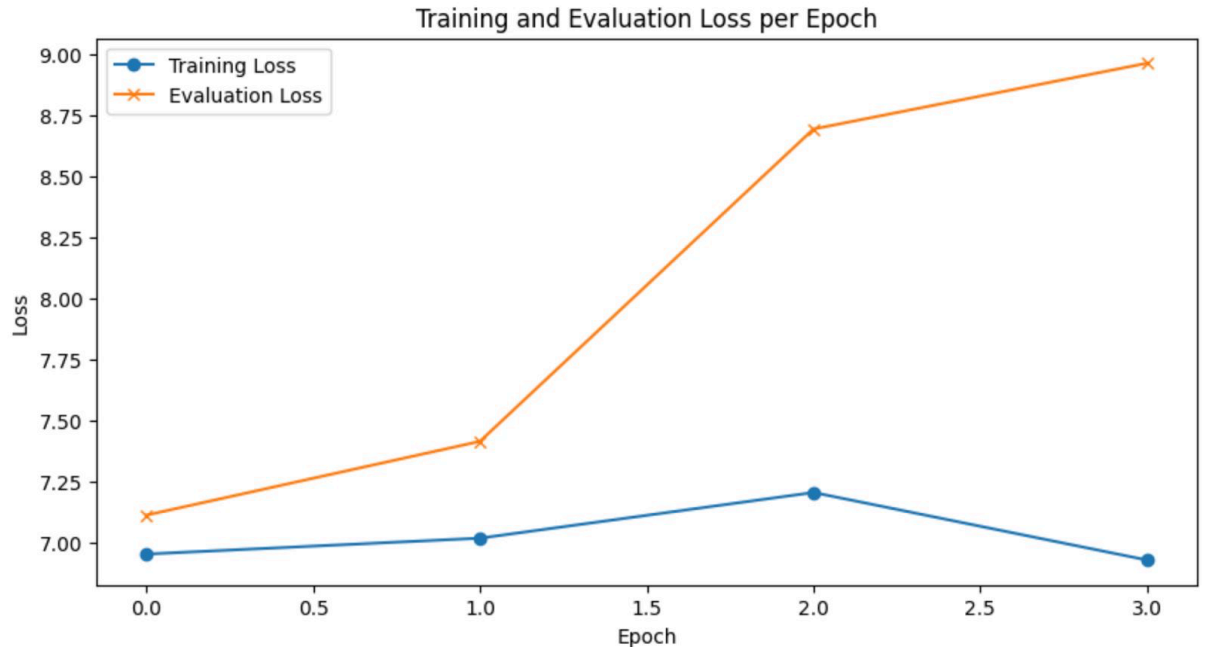


10. Added the following parameters to improve model performance, but it did not solve the issue,

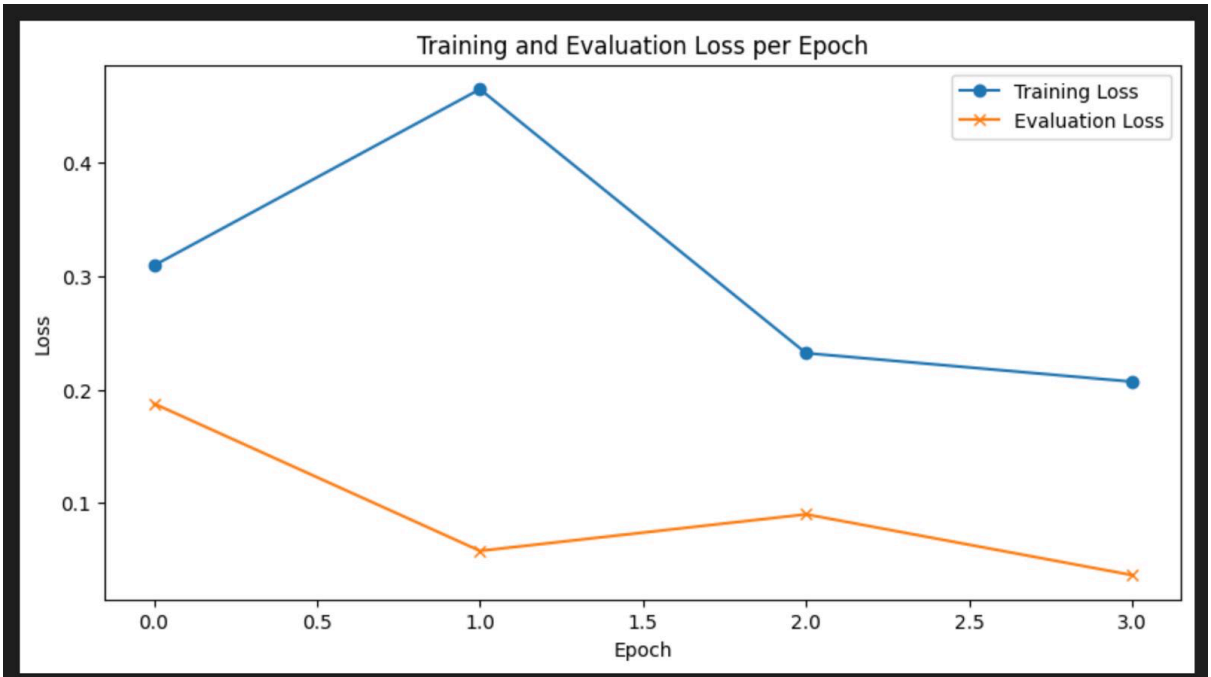
```
training_args = TrainingArguments(
    output_dir="./results",
    eval_strategy="epoch",
    save_strategy="epoch",
    learning_rate=3e-4,
    per_device_train_batch_size=5,
    per_device_eval_batch_size=5,
    num_train_epochs=4,
    weight_decay=0.01,
    logging_dir="./logs",
    logging_steps=10,
    load_best_model_at_end=True,
    metric_for_best_model="eval_loss"
)
```



11. Now, we tried increasing the testing data to 30% split from the original 20% and we're currently testing using this approach.



12. Since the loss was more, we went back to 20% testing data and set the early stopping threshold to 3, which yielded the following graph,




```
Final Training Loss: 0.20694010279385755
Final Evaluation Loss: 0.03640138357877731
```