INDIAN INSTITUTE OF TECHNOLOGY (IIT), PATNA

ASSIGNMENT

Name	– Ms. Bhavna Latare
Roll No.	– 2311AI27
Course	– MTech
Stream	– Artificial Intelligence

Objective

Fine tune a large language model (LLM) for performing named entity recognition (NER) task on an automotive dataset. NER task involves processing unstructured text data to extract useful information/entities. This assignment is broken down in three tasks.

1. First task involves analyzing the data and identifying what are some entities that can be extracted from this data. We are interested in entities related to automotive domain. Some examples could be component, failure issue, vehicle model, corrective action etc. You may choose what all and how many entities you are planning to extract.

2. Second task is to use an open source LLM of your choice and write the prompt to extract the automotive domain entities from given dataset. Some examples of LLMs include LLaMA 2 7b, Phi-2, Mistral 7b etc. Explore zero shot and few-shot learning and other prompt engineering techniques.

3. Final task is to fine tune the selected LLM on a subset of provided dataset. Be creative in selecting fine-tuning method, generating fine-tuning dataset and training the model. Evaluate the fine-tuned model against zero-shot/few-shot on a held-out testing dataset. Report the comparison.

Task 01

Dataset Given - NHTSA Recall (https://www.nhtsa.gov/nhtsa-datasets-and-apis#recalls)

This dataset contains information about automotive recalls reported to the National Highway Traffic Safety Administration (NHTSA). Here's a detailed description of each field:

1. RECORD_ID: A unique identifier for each record, indicating the sequence number of the record.
2. CAMPNO: NHTSA Campaign Number, a unique identifier assigned to each recall campaign.
3. MAKETXT: The make or brand of the vehicle or equipment affected by the recall.
4. MODELTXT: The model of the vehicle or equipment affected by the recall.
5. YEARTXT: The model year of the vehicle or equipment affected by the recall. It's '9999' if unknown or not applicable.
6. MFGCAMPNO: Manufacturer Campaign Number, a number assigned by the manufacturer for the recall campaign.
7. COMPNAME: Component Description, describing the affected component of the vehicle or equipment.
8. MFGNAME: The name of the manufacturer that filed the defect/noncompliance report.
9. BGMAN: Begin Date of Manufacturing, indicating the start date of manufacturing for the affected units.
10. ENDMAN: End Date of Manufacturing, indicating the end date of manufacturing for the affected units.
11. RCLTYPECD: Type of recall report: Vehicle, Equipment, or Tire.
12. POTAFF: Potential number of units affected by the recall.
13. ODATE: Date on which the owner was notified by the manufacturer about the recall.
14. INFLUENCED_BY: Recall initiator, whether it's the manufacturer (MFR), Office of Vehicle Safety Compliance (OVSC), or Office of Defects Investigation (ODI).
15. MFGTXT: Manufacturers of the recalled vehicles/products.
16. RCDATE: Report received date, indicating when the recall report was received.
17. DATEA: Record creation date.
18. RPNO: Regulation Part Number.
19. FMVSS: Federal Motor Vehicle Safety Standard Number.

20. DESC_DEFECT: Defect summary, describing the defect(s) identified in the recalled vehicles or equipment.

21. CONSEQUENCE_DEFECT: Consequence summary, describing the potential consequences of the identified defect(s).

22. CORRECTIVE_ACTION: Corrective summary, describing the actions recommended or taken to correct the defect(s).

23. NOTES: Additional notes or details related to the recall.

24. RCL_CMPT_ID: Number uniquely identifying a recalled component.

25. MFR_COMP_NAME: Manufacturer-supplied component name.

26. MFR_COMP_DESC: Manufacturer-supplied component description.

27. MFR_COMP_PTNO: Manufacturer-supplied component part number.

This dataset provides comprehensive information about automotive recalls, including details about the affected vehicles or equipment, the nature of the defect(s), potential consequences, and corrective actions taken.

To perform Named Entity Recognition (NER) on the automotive dataset using the provided columns {DEFECT SUMMARY, CONSEQUENCE SUMMARY, CORRECTIVE SUMMARY, RECALL NOTES} the steps followed were:

1. Data Preprocessing:

- Load the dataset.
- Extract the relevant columns: DEFECT SUMMARY, CONSEQUENCE SUMMARY, CORRECTIVE SUMMARY, and RECALL NOTES.
- Clean the text data by removing any unnecessary characters, punctuation, and special symbols.
- Tokenize the text into individual words or phrases.

2. Entity Recognition:

- Utilize a pre-trained NER model specifically designed for the automotive domain, or train a custom NER model if necessary.
- Iterate through each row of the dataset and apply the NER model to extract entities.
- Use domain-specific lexicons, dictionaries, or patterns to improve entity recognition accuracy, especially for specialized terms in the automotive domain.

3. Entity Extraction:

- Identify and extract entities from the processed text data based on the NER model's output.

- Define entity categories such as component, failure issue, vehicle model, corrective action, etc.

- Capture entities that correspond to each category and store them for further analysis.

4. Post-processing and Analysis:

- Perform post-processing steps such as filtering out duplicate entities, normalizing entity names, and resolving any ambiguities.

- Analyze the extracted entities to gain insights into common issues, affected components, vehicle models, and corrective actions mentioned in the dataset.

- Visualize the extracted entities using charts, graphs, or tables to present the findings effectively.

5. Documentation:

- Document the entire process step by step, including data preprocessing, entity recognition, extraction, and analysis.

- Provide explanations for the choice of columns, NER model, and any domain-specific resources utilized.

- Include examples of extracted entities and their corresponding categories.

- Describe the insights obtained from the entity extraction process and how they contribute to understanding the dataset.

Task 02

"Using an open source LLM of your choice and write the prompt to extract the automotive domain entities from given dataset. Some examples of LLMs include LLaMA 2 7b, Phi-2, Mistral 7b etc. Explore zero shot and few-shot learning and other prompt engineering techniques."

Mistral 7b LLM has been used for performing this task.

Methods used to Extract Entities Related to Automotive Domain from Recall Data

1. Reading the Dataset:

   - The code reads a dataset named "FLAT_RCL.txt", which contains records of automotive recalls. Key columns include "DEFECT SUMMARY", "CONSEQUENCE SUMMARY", "CORRECTIVE SUMMARY", and "RECALL NOTES".

2. Defining Column Names:

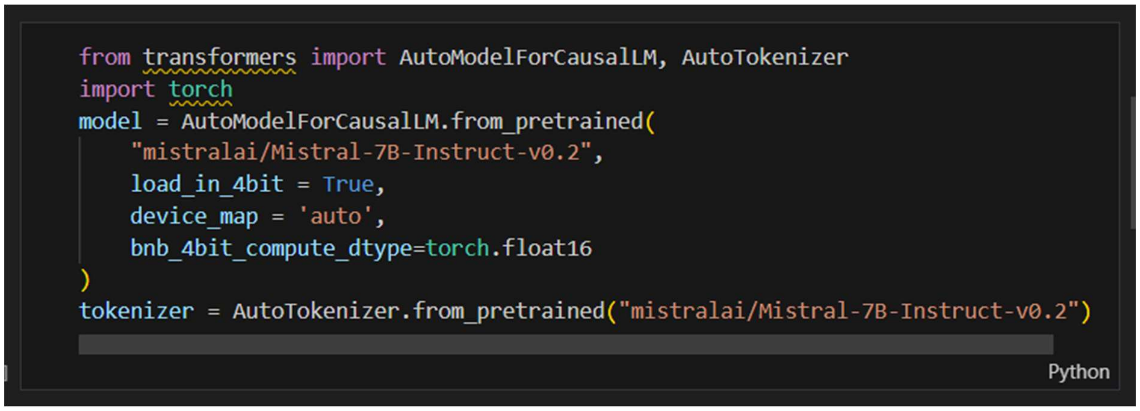
   - The script defines column names based on the description provided in the data schema. These columns are used to structure the data into a pandas DataFrame for further analysis.

3. Data Exploration:

   - Basic exploration of the dataset is conducted by printing unique values from columns such as "CONSEQUENCE SUMMARY", "CORRECTIVE SUMMARY", and "DEFECT SUMMARY". This helps understand the nature of the recall data.

4. Loading a Language Model:

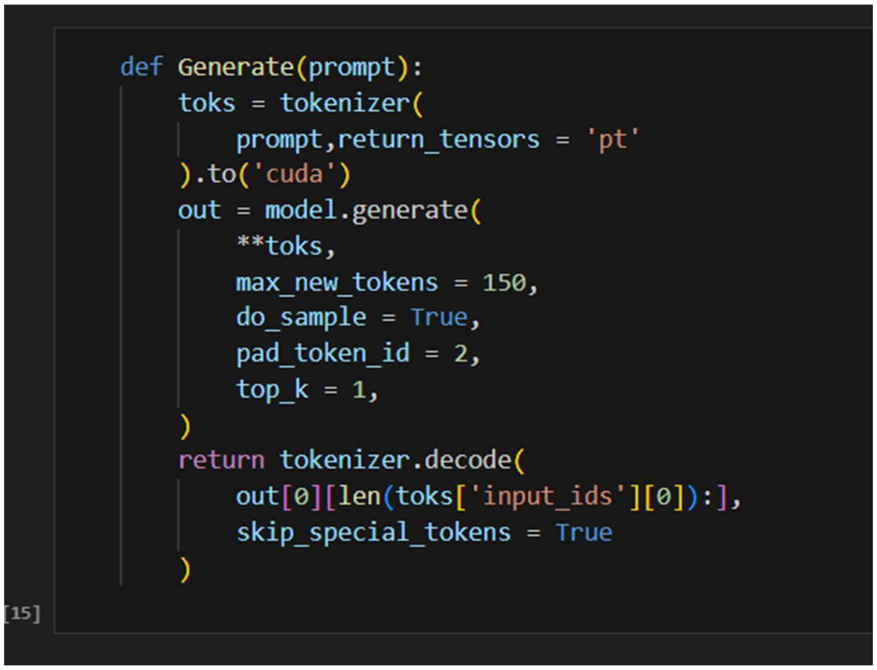
   - The code loads a pre-trained language model suitable for text generation. In this instance, the "Mistral-7B-Instruct-v0.2" model is utilized for its capabilities.



```python
from transformers import AutoModelForCausalLM, AutoTokenizer
import torch
model = AutoModelForCausalLM.from_pretrained(
    "mistralai/Mistral-7B-Instruct-v0.2",
    load_in_4bit = True,
    device_map = 'auto',
    bnb_4bit_compute_dtype=torch.float16
)
tokenizer = AutoTokenizer.from_pretrained("mistralai/Mistral-7B-Instruct-v0.2")
```

5. Generating Text based on Prompt:

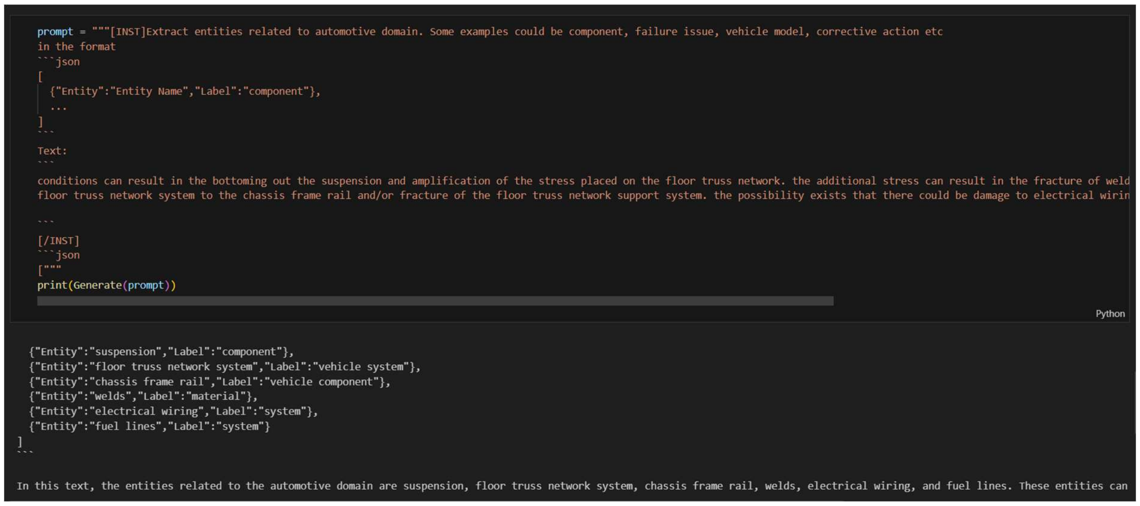
- A prompt is crafted to instruct the language model to extract entities pertinent to the automotive domain from the recall data. This prompt serves as guidance for the model and specifies the desired output format.



```python
def Generate(prompt):
    toks = tokenizer(
        prompt,return_tensors = 'pt'
    ).to('cuda')
    out = model.generate(
        **toks,
        max_new_tokens = 150,
        do_sample = True,
        pad_token_id = 2,
        top_k = 1,
    )
    return tokenizer.decode(
        out[0][len(toks['input_ids'][0]):],
        skip_special_tokens = True
    )
```

6. Generating Output:

- Text generation is executed by invoking the `Generate` function with the provided prompt. The function tokenizes the prompt, utilizes the loaded language model to generate text, and decodes the resulting output.



~~~~python
prompt = """[INST]Extract entities related to automotive domain. Some examples could be component, failure issue, vehicle model, corrective action etc
in the format
```json
[
  {"Entity":"Entity Name","Label":"component"},
  ...
]
```
Text:
```
conditions can result in the bottoming out the suspension and amplification of the stress placed on the floor truss network. the additional stress can result in the fracture of weld
floor truss network system to the chassis frame rail and/or fracture of the floor truss network support system. the possibility exists that there could be damage to electrical wirin

```
[/INST]
```json
["""
print(Generate(prompt))
~~~~

~~~~
  {"Entity":"suspension","Label":"component"},
  {"Entity":"floor truss network system","Label":"vehicle system"},
  {"Entity":"chassis frame rail","Label":"vehicle component"},
  {"Entity":"welds","Label":"material"},
  {"Entity":"electrical wiring","Label":"system"},
  {"Entity":"fuel lines","Label":"system"}
]
```

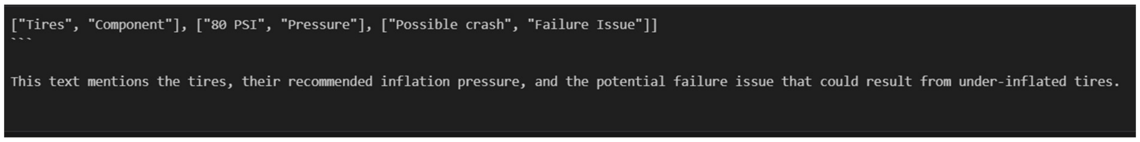
In this text, the entities related to the automotive domain are suspension, floor truss network system, chassis frame rail, welds, electrical wiring, and fuel lines. These entities can
~~~~

By following these steps, the code facilitates the extraction of relevant entities related to the automotive domain from recall data, leveraging the capabilities of large language models for natural language processing tasks.

Zero-shot learning

A model is trained to perform a task without any explicit examples or training data for that task. Instead, the model is provided with some form of semantic description or auxiliary information about the task, allowing it to make predictions even on data it has never seen before.

This has been used for the performance tracking on unseen classes or categories.



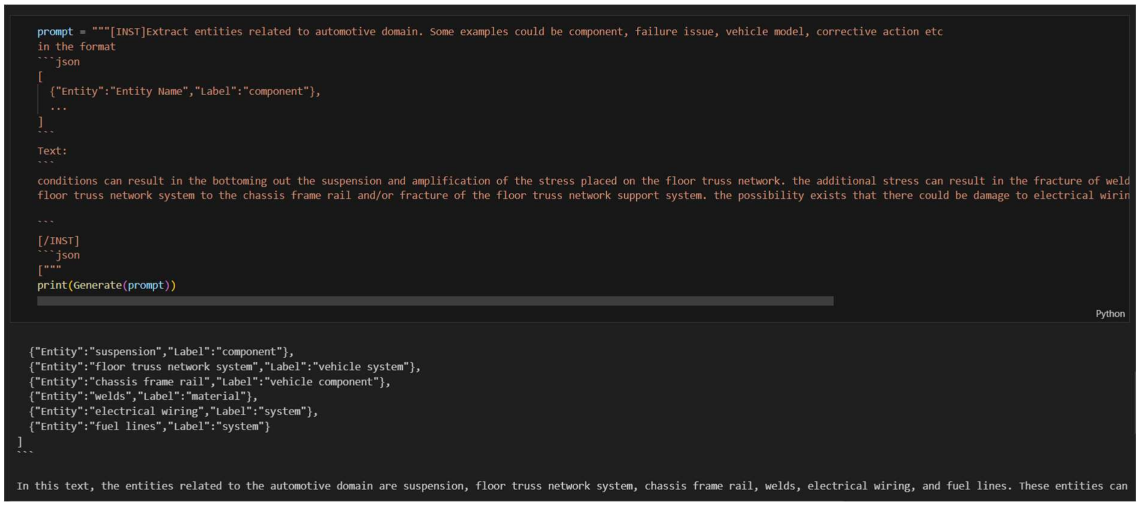
~~~
["Tires", "Component"], ["80 PSI", "Pressure"], ["Possible crash", "Failure Issue"]]
```

This text mentions the tires, their recommended inflation pressure, and the potential failure issue that could result from under-inflated tires.
~~~

Few-shot learning

It is the extension of zero-shot learning by providing the model with a small amount of labeled training data for a task. Here we have used the Two shot learning technique.

Here we have provided a text and entity as the training data and then given the next text for which we need the Named entity.

This is used when the labeled data is scarce or when adapting pre-trained models to perform new tasks quickly with minimal data.



~~~python
prompt = """[INST]Extract entities related to automotive domain. Some examples could be component, failure issue, vehicle model, corrective action etc
in the format
```json
[
  {"Entity":"Entity Name","Label":"component"},
  ...
]
```
Text:
```
conditions can result in the bottoming out the suspension and amplification of the stress placed on the floor truss network. the additional stress can result in the fracture of weld
floor truss network system to the chassis frame rail and/or fracture of the floor truss network support system. the possibility exists that there could be damage to electrical wirin

```
[/INST]
```json
["""
print(Generate(prompt))
~~~

Python

~~~
  {"Entity":"suspension","Label":"component"},
  {"Entity":"floor truss network system","Label":"vehicle system"},
  {"Entity":"chassis frame rail","Label":"vehicle component"},
  {"Entity":"welds","Label":"material"},
  {"Entity":"electrical wiring","Label":"system"},
  {"Entity":"fuel lines","Label":"system"}
]
```

In this text, the entities related to the automotive domain are suspension, floor truss network system, chassis frame rail, welds, electrical wiring, and fuel lines. These entities can
~~~