Automotive NER Assignment

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.

Dataset:

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

Instructions:

1. Download the files FLAT_RCL.zip and RCL.txt. The zip file contains the file FLAT_RCL.txt, which is the data you will be working with. RCL.txt has the description of all the columns in the FLAT_RCL.txt.
2. For the NER task, you must use the data only from columns: {DEFECT SUMMARY, CONSEQUENCE SUMMARY, CORRECTIVE SUMMARY, RECALL NOTES} . These four columns contain lots of automotive text data.
3. Use as many rows of the dataset as needed for the given task.

Submission Requirement:

1. Create a github repo for the code you have written and share the link as part of the submission.
2. The repo must contain a file “test.py” that takes text paragraph as input and return all the entities and their types as the output.
3. In the repo, add a short report showing comparisons of fine-tuned model and zero-shot/few-shot approach. Any other findings can be added to this report.
4. Create a hugging face repo for fine-tuned model and share the link.

Example of NER:

Input 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 welds securing the*

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 wiring and/or fuel lines which could potentially lead to a fire.”

Expected Output:

[{"Entity": "bottoming out the suspension", "Label": "Failure Issue"},

{"Entity": "amplification of the stress", "Label": "Failure Issue"},

{"Entity": "floor truss network", "Label": "Component"},

{"Entity": "fracture of welds", "Label": "Failure Issue"},

{"Entity": "chassis frame rail", "Label": "Component"},

{"Entity": "floor truss network support system", "Label": "Component"},

{"Entity": "damage to electrical wiring", "Label": "Failure Issue"},

{"Entity": "fuel lines", "Label": "Component"},

{"Entity": "fire", "Label": "Failure Issue"}]