Automotive NER Assignment Report

By Kamal Kaushik Varanasi

Objective

There were three key objectives for this assignment -

1. Analyzing the data and identifying the entities that can be extracted related to the automotive domain. I decided to extract the following entities - **Component, Failure Issue, Vehicle model, Corrective action, Manufacturer name, Recall date.**

2. Exploring various zero shot learning, few-shot learning, and other prompt engineering techniques along with various open-source LLMs to extract the entities. For this task, I analyzed various zero-shot and few-shot techniques with **Meta LLaMA 2 7b, Microsoft Phi-2, OpenAI GPT-3.5-turbo models**.

3. Fine-tuning an LLM of choice on a subset of the given data. Evaluate and report the comparison between the zero shot and few shot approaches and the fine-tuned model.

Data

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

The dataset included unstructured text data with 250k rows and 27 columns. For this task, the data of only four columns - {DEFECT SUMMARY, CONSEQUENCE SUMMARY, CORRECTIVE SUMMARY, RECALL NOTES} was considered.

Upon analyzing the data, I found out that the majority of the text data in these columns followed a specific sentence structure, and the chosen entities occurred at specific locations in the text. And I also observed that there was a lot of repetition of the same sentences in these four columns across multiple rows. They were essentially repeating for the rows belonging to the same manufacturer (MFGNAME column).

So, in order to analyze the unique entries in these four columns, I first extracted all the unique combinations of the text in these columns. Then, I combined all the text fields from all four columns to make one text for each row.

Later, to generate a labeled dataset for fine-tuning, I took some samples out of these texts and annotated them manually (around 10-20). Then, I used OpenAI's gpt-3.5-turbo model to generate around 1050 instances of artificial labeled data. At the end, I validated and cleaned this data to get it ready for fine-tuning.

I split this data into the following - train (88%), validation (9%), and test (3%).

Selecting the language model and the fine-tuning approach

I decided to fine-tune the Microsoft Phi-2 model as compared to others because of the following reasons:

1. Performance of the LLM on various benchmarks like commonsense reasoning, language understanding, etc [1].
2. Size of the LLM and ease of fine-tuning, etc.
3. Open-sourced LLM.
4. Performance on the given data without any fine-tuning.

During fine-tuning, I used the Low-Rank Adaptation (LoRA) technique to reduce the number of trainable parameters and get better performance and memory utilization. I used Google Colab's A100 Nvidia GPU to fine-tune.

After fine-tuning, I uploaded my model to the HuggingFace Model Hub repo here. And the code for generating artificial data and fine-tuning is available in GitHub.

Fine-tuned Model vs Other Approaches

The LlaMa 2 7b model is a huge model that takes a lot of memory and loading time, which is not practical. Moreover, the LlaMa model shows sub-par performance in traditional NLP tasks like NER, which is also evident in the case of our task (refer to the results.csv). On the other hand, the Phi-2 model, being a small language model, performs better than the LlaMa on most of the benchmark tasks [1]. Being a small language model, it does not occupy much RAM and has faster loading rates. But it also has certain limitations. For example - it might struggle to follow instructions as it is not instruction tuned, and sometimes it can produce irrelevant extra text [2].

But for our task, I found out that after fine-tuning the Phi-2 model, there was a substantial improvement. It was able to give more consistent and reliable outputs than other LLMs (see example below). But sometimes, it tends to generate some irrelevant output as well. Please refer to the results.csv.

Example:

1. **Text:** *The excessive load on the rear axle can lead to the deformation of the suspension components and exacerbation of the stress on the axle housing. This increased stress may cause the fracture of bolts securing the axle housing to the chassis frame rail, or even the collapse of the axle housing support system. There is a risk of damage to brake lines and electrical wiring, potentially resulting in a fire.*

LlaMa Output:


```
{
        "component": "suspension components",
        "failure issue": "deformation",
        "vehicle model": null,
        "corrective action": null
}
```

Phi-2 Output:


```
{
        "component": "rear axle",
        "failure issue": "excessive load",
        "vehicle model": "not specified",
        "corrective action": "not specified"
}
```


Phi-2 Fine-tuned Output:


```
[
  {
    'Entity': 'rear axle',
    'Label': 'Component'
  },
  {
    'Entity': 'deformation of the suspension components',
    'Label': 'Failure_issue'
  },
  {
    'Entity': 'axle housing',
    'Label': 'Component'
  },
  {
    'Entity': 'fracture of bolts securing the axle housing to the chassis frame rail',
    'Label': 'Failure_issue'
  },
  {
    'Entity': 'axle housing support system',
    'Label': 'Component'
  },
  {
    'Entity': 'damage to brake lines',
    'Label': 'Failure_issue'
  },
  {
    'Entity': 'fire',
    'Label': 'Failure_issue'
  }
]
```


For the rest of the results, please view results.csv present in the GitHub Repo.

Ways to improve

We can improve the performance further by providing more robust training data, and choosing a different LLM along with better compute resources to fine-tune.

References

[1] https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/

[2] https://www.promptingguide.ai/models/phi-2