Adaptive Text-to-Command Translation for Robot Navigation: Fine-Tuning T5-Small for Natural Language-Based Control

***Team Members*: Adrien, Akhil, Sai Jagadeesh, Suhas, Varun, Vijay**

Dataset : Link - Navigation Dataset

The dataset used for this project comprises **11040** examples of combinations of complex and simple sentences providing instructions about the sequence of the batteries to be followed for the robot. These sentences are generated using a template of **93** sentences which had placeholders for battery colors (red, green, orange, blue, and purple). An example of the placeholder is below

"Your journey starts with [battery1] as the primary destination. After you've completed this, make your way to the [battery2], then move to [battery3], you can now go to [battery4] and then to [battery5]."

Here, the placeholders are replaced by every possible color with permutations which will lead to one sentence in the template giving out multiple sentences of different orders of colors. The output labels provide a simplified version of instructions. For example, for the input sentence, "**You'll end up at green battery eventually, but start with purple battery and make sure to visit orange battery, followed by blue battery and red battery."**, the given output label is: "**Go to purple; Go to orange; Go to blue; Go to red; Go to green".** This creates a diverse and exhaustive set of instructions for training the T5-small model and ensures that the T5-small model learns to map natural language instructions to action outputs, which allows the T5-small model to generalize well to similar unseen instructions, ensuring robust performance in guiding the robot to navigate through the desired checkpoints, that is the batteries in our case.

Baseline Results:

When testing T5-small with the simplest direction from our training dataset, it failed to provide the right answer.

```
Input: Go to the green battery first, then go to the red battery, followed by the orange battery, blue battery and purple battery.
Just give me the color order in this way: Go to [color1], Go to [color2]
Output: the color order: Go to [color1], Go to [color2]
```

When testing T5-small with a more challenging direction from our training dataset, it failed to provide an acceptable answer.

```
Input: Before going to purple battery,start your task at green battery and make sure to visit blue battery along the way. Then go to the red battery, and wrap up at the orange battery.
Just give me the color order in this way: Go to [color1], Go to [color2]
Output: , go to [color1], Go to [color2] and go to [color2]
```

First, when fed with the least complex direction from our training dataset, it simply repeated the directions. Second, when fed with a more challenging direction from our training dataset, failed to understand the task and provide a reasonable answer. These results showed that T5-small is not able to perform this task with the least complex direction in our dataset. So, we concluded that T5-small is not able to understand the requirements of this task and thus fails at this task.

Fine-Tuned Model Results and Techniques:

T5-small is a **sequence-to-sequence (seq2seq) Transformer model** that processes input text and generates output text. As sequence-to-sequence models like T5-small are general they can be **finetuned by training it on our dataset to make them give the output in the way we want without making any architectural changes to it.** So we took the pre-trained model, T5-small, and fine-tuned it on our task by training it in our dataset consequently changing all the parameters of the T5-small model with each training epoch.

After **fine-tuning the T5-small model with our task-specific dataset**, the model is skilled at processing and reorganizing long and complicated input statements into a correct sequence. Compared to pre-trained T5-small model, which struggles to reliably grasp the meaning and correctly reorder instructions, the fine-tuned version has learned to recognize subtle structures and interdependencies in input texts, as per the following screenshots:

```
Input: Go to the green battery first, then go to the red battery, followed by the orange battery, blue battery and purple battery.
Just give me the color order in this way: Go to [color1], Go to [color2]
Output: Go to green; Go to red; Go to orange; Go to blue; Go to purple
```

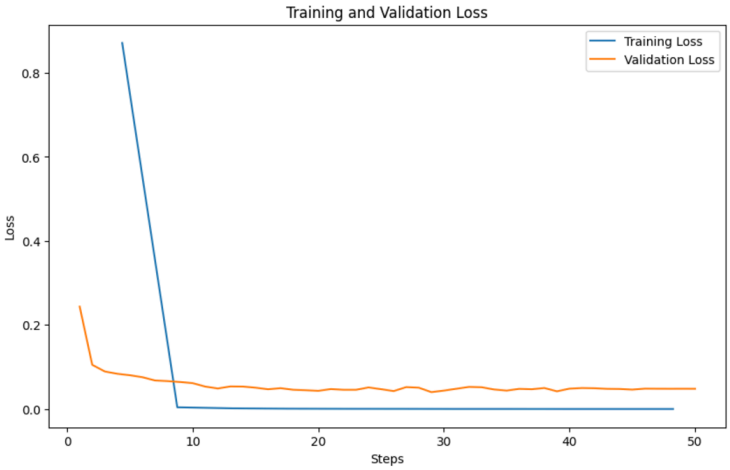
For example, given the instruction:
"Go to the green battery first, then go to the red battery, followed by the orange battery, blue battery, and purple battery."
The fine-tuned model outputs the sequence: ***"Go to green; Go to red; Go to orange; Go to blue; Go to purple"***

```
Input: Before going to purple battery,start your task at green battery and make sure to visit blue battery along the way. Then go to the red battery, and wrap up at the orange battery.
Just give me the color order in this way: Go to [color1], Go to [color2]
Output: Go to green; Go to blue; Go to purple; Go to red; Go to orange
```

In another example, with the more **complex instruction:**
"Before going to the purple battery, start your task at the green battery and make sure to visit the blue battery along the way. Then go to the red battery, and wrap up at the orange battery."
The fine-tuned model outputs the sequence: ***"Go to green; Go to blue; Go to purple; Go to red; Go to orange"***

This demonstrates that the model can effectively learn dependencies and orders within instructions, producing the required output format even for more **complicated input.**

Graph of Training and Validation loss of Fine-tuned Model:



Evaluation Metrics for Fine-tuned Model:

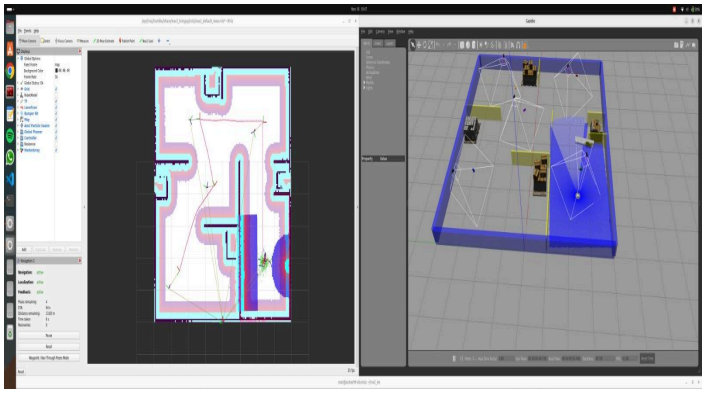
```
[87] evaluation_results = trainer.evaluate()
     print(evaluation_results)
```

```
[29/29 02:06]
{'eval_loss': 0.04829540103673935, 'eval_runtime': 4.2886, 'eval_samples_per_second': 106.329, 'eval_steps_per_second': 6.762, 'epoch': 50.0}
```

Accuracy:

```
/usr/local/lib/python3.10/dist-packages/transformers
  warnings.warn(
 Accuracy: 100.00%
Correct Predictions: 219 out of 219
```

Robot Operating System Implementation



We've been working on **integrating the T5-small model's output with the ROS framework** to make the robot **follow dynamic sequences of battery colors in simulation.** To achieve this, **we migrated the current framework from ROS 2 Galactic to Humble,** giving us access to improved libraries and tools. The framework now **uses a custom ROS node to convert the model's output into waypoints for navigation in the Gazebo simulation**. Previously, the robot followed static sequences with ArUco markers, but now it dynamically adapts to the T5-small model's inputs. We're currently **refining the waypoint mapping** from random color sequences and testing navigation accuracy within the simulation environment.