**User Manual**

**Overview**

This document provides instructions on how to use the model testing solution. The solution is designed to evaluate the performance of a language model on various test cases, ensuring that the model produces accurate and expected outputs. The document includes information on the required imports, input formats, limitations, and steps to follow when using the solution.

**System Requirements**

* **Operating System: Windows, macOS, or Linux.**
* **Python Version: Ensure Python 3.7 or later is installed.**
* **RAM: At least 8 GB of RAM is recommended.**
* **Disk Space: Ensure you have at least 2 GB of free disk space for dependencies.**
* **Visual Studio Code (VS Code): You should have VS Code installed, as it’s the recommended IDE for running this solution.**

**Setting Up a Virtual Environment (venv)**

**Read token is needed to read from model:  
hf\_IfXpplRPLVxsMJQzQMfqhBUQVvpOaYsmCD**

**This is the write token:  
hf\_qbBOXPDfTjNqMDEZrYEnkjnRrkxBNhYAgc is the write token**

**A virtual environment isolates your project’s dependencies, making sure they don’t interfere with other Python projects on your machine.**

1. **Install Python:**
   * **Download and install Python from** [**python.org**](https://www.python.org/downloads/)**.**
   * **Make sure to check the option to add Python to your system PATH during installation.**
2. **Create a Virtual Environment:**
   * **Open the command prompt (or terminal on macOS/Linux).**
   * **Navigate to your project directory:**

**bash**

**cd path\to\your\project**

* + **Create a virtual environment using the venv module:**

**bash**

**python -m venv venv**

* + **This will create a new directory called venv in your project directory containing the Python interpreter and a copy of the standard library.**

1. **Activate the Virtual Environment:**
   * **On Windows:**

**bash**

**venv\Scripts\activate**

* + **On macOS/Linux:**

**bash**

**source venv/bin/activate**

* + **You should now see (venv) at the beginning of your command prompt, indicating that the virtual environment is active.**

**Installing Required Packages**

**Once the virtual environment is activated, install the necessary Python packages using pip.**

* **In your project’s root directory, create a requirements.txt file containing all the dependencies.**

**torch**

**transformers**

* **Install the packages:**

**bash**

**pip install -r requirements.txt**

**Setting Up Visual Studio Code**

* **Open your project in VS Code.**
* **If prompted, select the Python interpreter from your virtual environment:**
  + **Press Ctrl+Shift+P and type "Python: Select Interpreter."**
  + **Choose the interpreter located in your venv folder.**
* **Install the Python extension in VS Code if you haven’t already. This extension helps with code formatting, linting, and debugging.**

**Running the Model Testing Code**

* **Ensure your virtual environment is activated.**
* **Open a terminal in VS Code (`Ctrl+``) and run your Python scripts.**
* **You can also run the script directly by selecting "Run Python File" in the top-right corner of the editor.**

**6. Checking Your Torch Version**

**To check the installed version of PyTorch:**

**bash**

**python -c "import torch; print(torch.\_\_version\_\_)"**

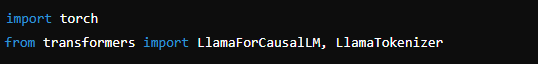
**7. Testing and Execution**

* **After setting up your environment, you can run the provided Python scripts to test the model against specific prompts.**
* **Modify the test cases and prompts as needed and rerun the script to evaluate the model’s performance under different scenarios.**

**This guide should help you or any other user set up a local environment and run the model testing solution in Visual Studio Code efficiently.**

**Required Imports**

Before running the model testing code, the following libraries must be imported:



These imports are essential for loading the model and tokenizer, encoding inputs, and generating outputs.

**Model Initialization**

To use the model, initialize the tokenizer and model using the following code:

A black screen with white text

Description automatically generated

Ensure that the model\_name is correctly specified.

**Input Format**

The input to the model is a prompt that describes the function to be tested along with a specific test case. The input format typically follows this structure:

A computer code with green text

Description automatically generated

Replace function\_name, parameter1, parameter2, and other placeholders with the appropriate names and values.

**Limitations on Inputs**

1. **Data Types**: The inputs must be in the correct data types as expected by the function being tested.
2. **Value Ranges**: Ensure that the input values are within the expected ranges to avoid unexpected errors.
3. **Test Cases**: The number of test cases should be manageable to avoid excessive computation time.

**Steps to Use the Solution**

1. **Set Up the Environment**: Ensure that the required libraries are installed (torch, transformers) and import them.
2. **Initialize the Model**: Load the appropriate model and tokenizer using the provided code.
3. **Prepare the Prompt**: Write the function prompt with the necessary test cases.
4. **Run Test Cases**: Use a loop to encode the prompt, generate outputs, and compare them against the expected results.
5. **Interpret the Results**: Evaluate whether the model's outputs match the expected outputs. Adjust the prompt or model settings if necessary.

**Conclusion**

This manual provides the essential steps and guidelines for using the model testing solution. By following these instructions, you can effectively test and evaluate the performance of language models on a variety of tasks.