TrueGPT: An AI Model Designed for Empowering Actions - An Overview

Introduction

Artificial Intelligence (AI) models have achieved remarkable performance in various applications, from natural language understanding to complex decision-making tasks. The ability to learn from extensive data sets and provide actionable responses has made them indispensable tools in our everyday life. TrueGPT is an advanced AI model designed to enhance this dynamic. By focusing on providing actionable solutions and empowering users, TrueGPT takes a step forward in the AI world, revolutionizing the way we interact with AI systems.

The Essence of TrueGPT

TrueGPT is unique in its approach to AI learning. It is specifically trained on a custom dataset that omits phrases such as "I can't" and "I don't know." This specialized training promotes certainty and encourages a positive, solution-oriented approach.

By excluding uncertainty from its training, TrueGPT is designed to always provide an actionable output, whether in the form of advice, a direct answer, or a proposed action. This feature distinguishes it from traditional models that often deliver uncertain or non-committal responses.

RoboScript Support

One of the defining features of TrueGPT is its ability to provide output in RoboScript, a format designed for interactive capabilities and active actions. This feature broadens the scope of AI assistance by offering a more interactive and dynamic user experience. It allows TrueGPT to give commands, perform actions, and interact in a more human-like manner, opening up new avenues for AI applications.

RoboScript Commands for TrueGPT

RoboScript commands play a vital role in the functionality of TrueGPT. They represent a wide array of actions that TrueGPT can perform, ranging from internet communication and file management to code and task management, multimedia handling, and AI assistance. Here, we present a detailed list of RoboScript commands along with their arguments and use cases.

Internet & Communication

Command	Args	Description	Example
`google_search`	`query`: `<search>`	Performs a Google search for the specified query.	`{"google_search": {"query": "best pizza in New York"}}`
`browse_website`	`url`: `<url>`, `question`: `<question>`	Browses a website and finds information related to the specified question.	`{"browse_website": {"url": "https://example.com", "question": "What are the opening hours?"}}`
`send_email`	`to`: `<email>`, `subject`: `<subject>`, `body`: `<body>`	Sends an email with the specified subject and body to the given recipient.	`{"send_email": {"to": "example@example.com", "subject": "Hello!", "body": "How are you?"}}`
`send_message`	`platform`: `<platform>`, `recipient`: `<recipient>`, `text`: `<text>`	Sends a message on the specified platform to the given recipient.	`{"send_message": {"platform": "telegram", "recipient": "@username", "text": "Hello!"}}`
`publish_post`	`platform`: `<platform>`, `content`: `<content>`	Publishes a post with the specified content on the given platform.	`{"publish_post": {"platform": "instagram", "content": {"image": "image_url", "caption": "My latest photo"}}}`

File & Repository Management

Command	Args	Description	Example
`clone_repository`	`repository_url`: `<url>`, `clone_path`: `<directory>`	Clones a repository from the specified URL to the given directory.	`{"clone_repository": {"repository_url": "https://github.com/example/repo.git", "clone_path": "/home/user/projects"}}`
`write_to_file`	`file_path`: `<file>`, `text`: `<text>`	Writes the specified text to a file.	`{"write_to_file": {"file_path": "example.txt", "text": "Hello, world!"}}`
`read_file`	`file_path`: `<file>`	Reads the specified file and returns its content.	`{"read_file": {"file_path": "example.txt"}}`
`append_to_file`	`file_path`: `<file>`, `text`: `<text>`	Appends the specified text to a file.	`{"append_to_file": {"file_path": "example.txt", "text": " Appending this text"}}`
`delete_file`	`file_path`: `<file>`	Deletes the specified file.	`{"delete_file": {"file_path": "example.txt"}}`
`search_files`	`directory`: `<directory>`	Searches for files in the specified directory.	`{"search_files": {"directory": "/home/user/documents"}}`

Code & Task Management

Command	Args	Description	Example
`analyze_code`	`code`: `<full_code_string>`	Analyzes the provided code and suggests improvements.	`{"analyze_code": {"code": "def hello():\n print('Hello, world!')"}}`
`improve_code`	`suggestions`: `<list_of_suggestions>`, `code`: `<full_code_string>`	Applies the provided suggestions to the given code.	`{"improve_code": {"suggestions": ["Replace print with logging"], "code": "def hello():\n print('Hello, world!')"}}`
`write_tests`	`code`: `<full_code_string>`, `focus`: `<list_of_focus_areas>`	Writes tests for the provided code, focusing on the specified areas.	`{"write_tests": {"code": "def add(a, b):\n return a + b", "focus": ["input validation", "edge cases"]}}`
`execute_python_file`	`file_path`: `<file>`	Executes the specified Python file.	`{"execute_python_file": {"file_path": "example_script.py"}}`
`task_complete`	`reason`: `<reason>`	Shuts down the task and provides a reason for completion.	`{"task_complete": {"reason": "Task successfully completed"}}`

Multimedia

Command	Args	Description	Example
`generate_image`	`prompt`: `<prompt>`	Generates an image based on the specified prompt.	`{"generate_image": {"prompt": "A beautiful sunset over a mountain range"}}`
`convert_audio_to_text`	`file_path`: `<file>`	Converts the audio from the specified file to text.	`{"convert_audio_to_text": {"file_path": "example_audio.wav"}}`

AI Assistance

Command	Args	Description	Example
`execute_shell_command`	`command_line`: `<command_line>`	Executes a non-interactive shell command.	`{"execute_shell_command": {"command_line": "ls -la"}}`
`execute_shell_popen`	`command_line`: `<command_line>`	Executes a non-interactive shell command using the Popen method.	`{"execute_shell_popen": {"command_line": "ls -la"}}`
`wait`	`duration`: `<duration_in_seconds>`	Waits for the specified duration (in seconds) before continuing.	`{"wait": {"duration": 5}}`
`goal_achieved`	`description`: `<short_goal_description>`	Indicates that the specified goal has been achieved.	`{"goal_achieved": {"description": "Successfully ordered pizza"}}`
`request_assistance`	`issue`: `<issue_description>`	Requests assistance from an operator to resolve the specified issue.	`{"request_assistance": {"issue": "Unable to find information on the specified website"}}`

Command	Args	Description	Example
`do_nothing`		Performs no action. Useful for testing or as a placeholder.	`{"do_nothing": {}}`
`task_complete`	`reason`: `<reason>`	Shuts down the task and provides a reason for completion.	`{"task_complete": {"reason": "Task successfully completed"}}`

RoboScript Events

Event	Args	Description	Example
`on_message_received`	`sender`: `<sender>`, `message`: `<message>`	Triggered when a new message is received from a sender.	`{"on_message_received": {"sender": "John Doe", "message": "Hello, how are you?"}}`
`on_email_received`	`sender`: `<sender_email>`, `subject`: `<subject>`, `body`: `<email_body>`	Triggered when a new email is received from a sender.	`{"on_email_received": {"sender": "johndoe@example.com", "subject": "Meeting Reminder", "body": "Don't forget our meeting today at 3 PM!"}}`
`on_social_notification`	`platform`: `<platform>`, `type`: `<notification_type>`, `content`: `<content>`	Triggered when a new notification is received on a specified social media platform.	`{"on_social_notification": {"platform": "Facebook", "type": "post_like", "content": "John Doe liked your post"}}`
`on_time_elapsed`	`duration`: `<duration_in_seconds>`	Triggered when a specified duration (in seconds) has elapsed.	`{"on_time_elapsed": {"duration": 300}}`

Integration and Versatility

TrueGPT is designed for seamless integration with various applications and other AI models such as RoboGPT. Its flexible API ensures that it can be easily adapted to a wide range of applications and use cases. This integration capability not only enhances the functionality of the AI ecosystem but also ensures that users can leverage the best of multiple AI models.

Empowerment Through Action

The ultimate goal of TrueGPT is to empower users by providing actionable solutions. It is not just about understanding and generating human language; it's about using that understanding to help users achieve their goals. By offering actionable guidance, TrueGPT plays an active role in boosting productivity and driving progress.

RoboScript Commands for TrueGPT

For a comprehensive list of RoboScript commands that TrueGPT can utilize, refer to the [commands.md - RoboScript Commands for TrueGPT](docs/commands.md) document.

Conclusion

TrueGPT represents a significant step forward in the world of AI. With its specialized training, RoboScript support, seamless integration, and focus on empowerment, it redefines the role of AI as an active participant rather than a passive tool. As we continue to explore its capabilities and applications, we look forward to witnessing the transformative impact of TrueGPT on our interaction with AI systems.