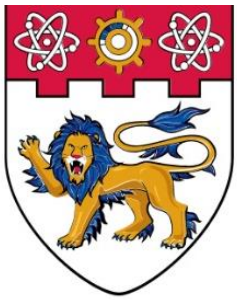
EXTRACTING HUMAN PREFERENCES FROM LANGUAGE INPUTS USING GPT



NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Mohamed Raizee Bin Mohamed Ibrahim

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

Year 2023/2024

EXTRACTING HUMAN PREFERENCES FROM LANGUAGE INPUTS USING GPT

SUBMITTED

BY

MOHAMED RAIZEE BIN MOHAMED IBRAHIM

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

A final year project report presented to Nanyang Technological University
in partial fulfilment of the requirements for the
Degree of Bachelor of Engineering (Mechanical Engineering)
Nanyang Technological University

Year 2023/2024

Abstract

This study presents an innovative architecture designed to significantly enhance the robustness and performance of a pipeline for generating modified trajectories for robotic arms, based on initial trajectories provided by the user. By adopting a divide-and-conquer strategy, this novel pipeline excels in processing intricate chained commands from user inputs, accurately identifying the requisite features to be fed into the deformation function. This function is responsible for producing the altered trajectory that aligns with the user's specifications.

The report rigorously evaluates the proposed architecture against existing methodologies, using a variety of benchmarks focused on accuracy and performance metrics. Additionally, to augment user interaction and streamline the command input process, the report introduces the integration of an advanced speech-to-text module. This module not only simplifies the user interface but also expands the system's accessibility by enabling voice commands. The comprehensive assessment presented in this report underscores the significant advancements achieved by the new pipeline in terms of efficiency, accuracy, and usability, marking a pivotal step forward in the field of robotic trajectory deformation.

Acknowledgement

I would like to express my deepest gratitude to Associate Professor Ang Wei Tech for his unparalleled support and guidance throughout this research project. I am also profoundly thankful to Ph.D. student researcher J-Anne Yow, whose invaluable advice and deep expertise significantly contributed to the progress of this work. Her mentorship has been instrumental in navigating the complexities of our research. Additionally, I would like to extend my thanks to both my faculty, the School of Mechanical and Aerospace Engineering, and the Joint Research Institute, Rehabilitation Research Institute of Singapore, for providing the essential resources and facilities necessary for the successful completion of this research. Lastly, I extend my appreciation to my family, friends, and colleagues, whose unwavering support has been the driving force behind the successful completion of this Final Year Project.

Table of Contents

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and motivation

The field of robotics has developed rapidly over the years, propelled by the rise of Artificial Intelligence (AI), which has played a pivotal role in expediting this transformative journey [1]. As robots increasingly find their way to working in a shared environment with humans, achieving a near-seamless human-robot interaction is imperative to foster collaboration and productivity. However, conventional forms of human-robot interaction using traditional keyboard-and-mouse or touch-screen interfaces often require individuals to possess technical skills to command complex robotic tasks that are not ideal for non-technical users [2]. This underscores the need to establish more accessible and user-friendly human-robot interaction methods.

Spoken language serves as the cornerstone of human communication, offering an intuitive means of conveying intent. Integrating speech into Robotics enhances robot-to-human interaction, elevating the potential applications of robotics by enabling direct vocal commands for task execution. However, for robots to comprehend these commands, they must first be processed by a language model into text that the robot can interpret. This necessitates grounding the language within the context of the physical world, a feat achievable through prompt engineering and the insights provided by vision models [2,3]. Once the language is grounded, the language model can

generate a deterministic response that the robot can understand, and thus take the appropriate course of action.

1.2 Objectives and scopes

The objectives of this project are as follow:

1. Develop an approach to improve performance accuracy for feature classification.
2. Increase robustness for chained commands for language correction.
3. Incorporate speech-to-text capability.
4. Integration of the overall new pipeline with robotic arm

To align with the objectives, the scopes are as follow:

- Design a working pipeline on generic manipulation features by making modifications on the language encoder section.
- Focuses on generic manipulation features such as:
    a. Object distance related feature.
    b. Cartesian distance related feature
- To design the pipeline using the Whisper model for transcription, GPT-4[1] for preprocessing and classification

[1] Generative Pre-trained Transformers model by OpenAI

To elaborate further, the objective of this research is to enhance and simplify human-robot interaction by integrating advanced voice recognition technology. This technology is capable of accurately transcribing users' voice commands and extracting relevant features for the robot's interpretation. To achieve this goal, the research proposes a comprehensive pipeline that utilises the state-of-the-art Whisper Speech-to-Text model alongside the Generative Pre-trained Transformer (GPT) 4 language model. Additionally, the pipeline incorporates a Mask-RCNN computer vision model for real-time object detection within the environment.

The study aims to improve user control by enabling the issuance of voice commands to the robot, thus allowing it to dynamically adjust its trajectory based on these instructions. This enhancement is expected to boost the robot's responsiveness to user commands significantly.

Furthermore, the study explores augmenting the pipeline robustness in processing complex chained commands through advanced prompt engineering techniques. This endeavour aims to render the interaction between humans and robots more fluid and natural. By adopting this holistic approach, the research seeks to increase the overall effectiveness and user-friendliness of human-robot interactions.

Finally, the paper will compare the performance in terms of accuracy between employing a generative language model, such as GPT-4, and a sentence similarity model, like the Bidirectional Encoder Representations from Transformers (BERT) in feature matching. This benchmarking will provide insights into the effectiveness of

each model in understanding and executing commands based on human intent in the context of human-robot interaction.

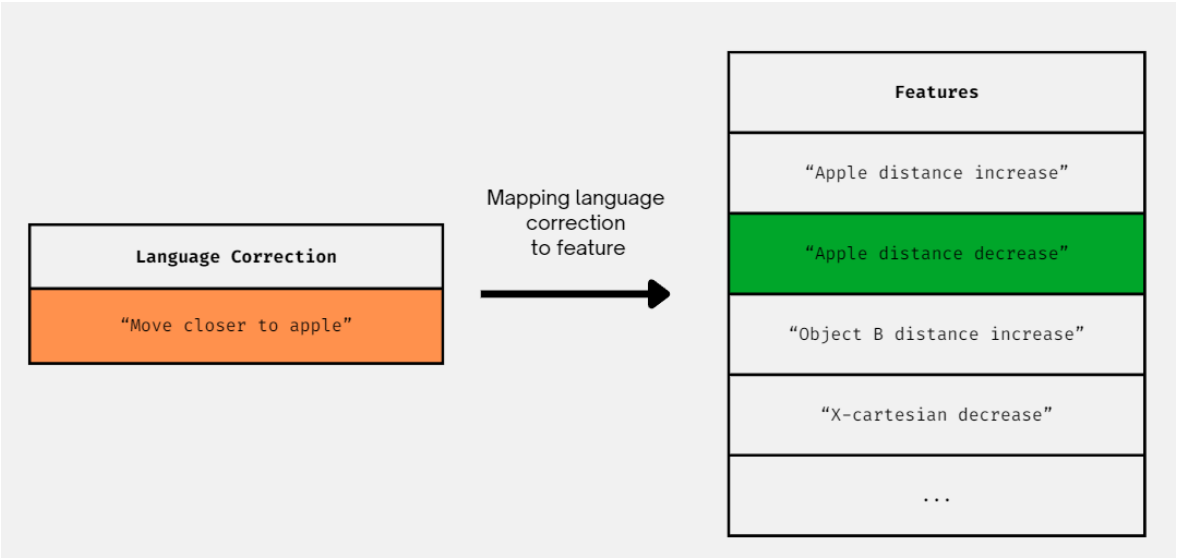


Figure 1: Feature matching from language correction

1.3 Structure of the report

Chapter 1 presents the evolution of robotics, driven by AI, and emphasises the need for intuitive human-robot interaction through speech, outlining how language models and vision models play a critical role in interpreting and grounding human commands for effective robot action.

Chapter 2 reviews existing work on grounding of language in robotics applications, the employment of large language models for the generation of features, and the merits of integrating a text-to-speech model into the processing pipeline.

Chapter 3 introduces a series of approaches designed to accomplish the stated objectives. Each strategy will be meticulously delineated, with an emphasis on the rationale behind its selection and the mechanisms through which it operates. Furthermore, this section will delve into the utilisation of Whisper transcription and its subsequent correction, elaborating on its application within this context.

Chapter 4 assesses the efficacy of each methodological approach mentioned in Chapter 3, ultimately identifying the most superior method as the proposed architecture for integration into the new pipeline.

Chapter 5 evaluates the effectiveness of the text-to-speech Whisper model in transcribing the local Singaporean accent. Moreover, this section will evaluate the accuracy of the post-transcription correction task facilitated by GPT-4.

Chapter 6 outlines the proposed pipeline that addresses the objectives and discusses potential directions for future work aimed at enhancing the system.

Chapter 2

Literature Review

2.1 Grounding Language in Robotics

Despite recent breakthroughs in natural language processing (NLP) capabilities spurred by the advent of Large Language Models (LLMs), challenges persist when these models lack context from a real-world perspective, particularly due to their insufficient grounding in a robot's surrounding scene [4]. Additionally, LLMs struggle to directly output low-level robot commands, a limitation arising from the scarce availability of relevant training data [5]. Consequently, the responses generated by LLMs may prove suboptimal or entirely unhelpful for a robot intended to perform tasks in a specific environment, given its inability to anchor these responses to practical robot use cases. This deficiency has guided robotics research toward the pursuit of grounded language acquisition, emphasising the need to contextualise language models within the physical world. The understanding and execution of tasks based on human commands thus require a robust mapping between language and the physical realm.

2.1.1 Statistical Machine Learning Method

One of the problems faced in grounding a language is novel language and perceptual inputs not anticipated in predefined formal representations of the world. In her pioneering work, Cynthia Matuszek introduces an advanced method for teaching

robots to understand language through a unique integration of language processing and sensory perception. This approach advocates for the application of statistical machine-learning methods to establish mappings between words and formal representations of the world. Furthermore, the paper posits that the comprehension of words and phrases can be approached as a classification problem [6]. This innovative strategy entails the development of a joint model that amalgamates language analysis with the context derived from sensor input, thereby facilitating a deeper linkage between words and their real-world significance. Central to Matuszek's methodology is the deliberate utilisation of verbal instructions and sensory data to educate specialised classifiers [6]. These classifiers can discern the environment and tasks at hand, thus allowing robots to comprehend and execute commands given in natural language. Employing statistical machine learning techniques, this model effectively integrates linguistic information with data collected by the robot's sensors to facilitate its operation [6, 7].

Notably, this methodology is primarily focused on processing objects, colours, and shapes as classifiers and exhibits limitations in generalising to entirely new visual categories. The chief constraint of this approach resides in its restricted capacity to progress beyond descriptive language, failing to incorporate a broader vocabulary and more complex representations of meaning, such as interpretations driven by intent [6, 8]. This shortcoming originates from the reliance on the Long Short-Term Memory (LSTM) recurrent neural network architecture. Although LSTM mitigates the vanishing gradient problem, allowing for the training of extended sequences, it is marked by computational intensity and a heightened susceptibility to overfitting [9, 10]. Further,

LSTM is characterised by protracted training durations and lacks the capability for parallel training. As a result, there is an urgent necessity for the development of an approach that overcomes these impediments.

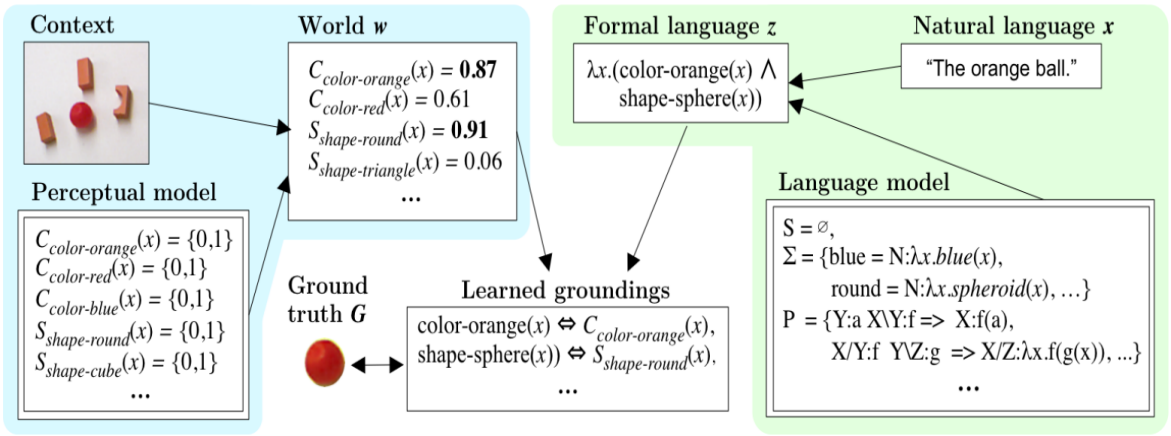


Figure 2: A joint modelling approach to classifier-based language grounding [6].

2.1.2 Language Model with Task Affordances

Another study addresses the challenges associated with grounding language in robotics through the "SayCan" framework. This framework integrates a large language model (LLM) with task affordances and value functions, aiming to ground large language models through value functions [11]. The LLM contributes a wealth of semantic world knowledge, offering insights into the steps required for the execution of complex and temporally extended tasks. Nevertheless, a gap persists in bridging the divide between this high-level conceptual understanding and the concrete, executable skills available to a robot. Furthermore, LLM by itself does not observe the consequences of their generation on any physical process. This observation supports the recommendation to merge the LLM's capabilities with task affordances and value functions [11]. Such integration is crucial for overcoming the limitations posed by

LLMs, which, despite their extensive textual knowledge, lack the tangible experiences necessary for direct control in robotic applications.

The methodology utilises prompt engineering within Large Language Models (LLMs) to methodically break down complex instructions into manageable sub-tasks, thereby improving robots' comprehension abilities [9]. To mitigate potential limitations stemming from oversight of the robot's capabilities and its operational environment, a comprehensive suite of acquired skills facilitates precise low-level visuomotor control. The method assesses the contribution of each skill through probabilistic evaluation and affordance functions. This ensures that the robot not only grasps the instructions but also implements them effectively, considering its capabilities and the context of the environment. This dual-layered approach improves the applicability of LLMs in robotic task execution, fostering a more nuanced and effective strategy.

However, the drawbacks of the limitations of the SayCan approach, as outlined in the paper, include inheriting the limitations and biases of large language models (LLMs), such as dependency on training data [12, 13, 14]. Additionally, the LLM only receives feedback from the environment through the selected skills, which can be limiting for instance, if a skill fails. This suggests areas for future investigation, like incorporating more direct sources of environmental feedback into the model to enhance its decision-making process.

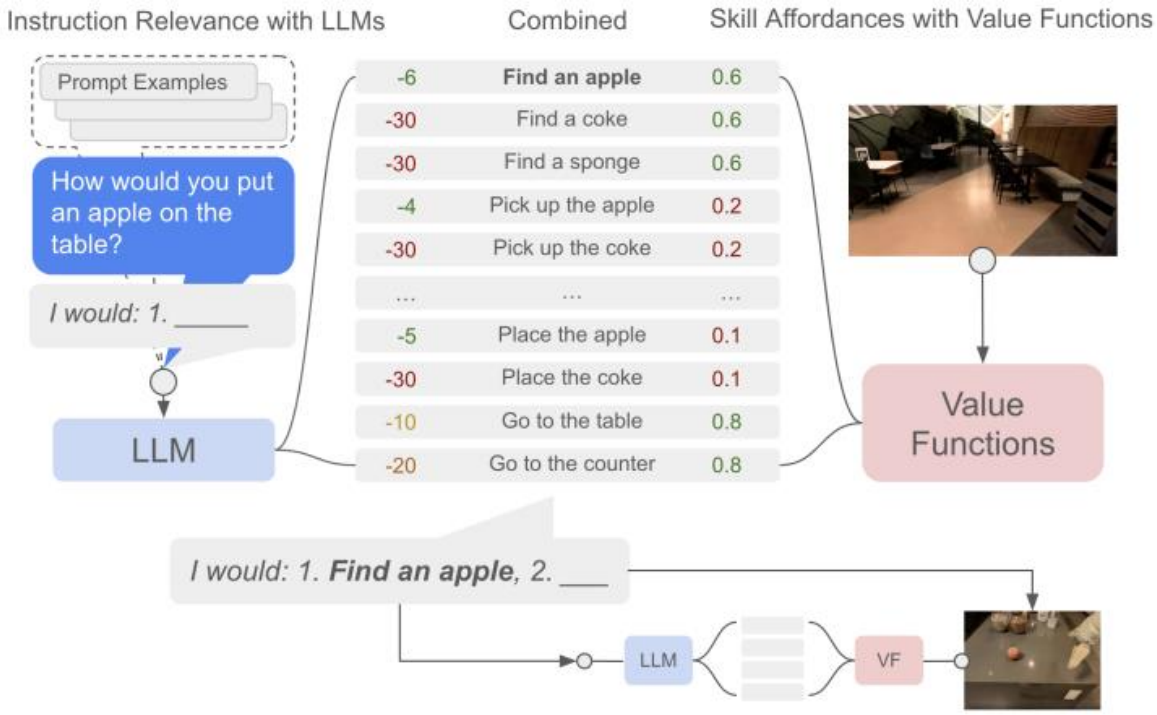


Figure 3: SayCan combines LLM and Skill Affordances [8].

2.1.3 Language and Image Encoder with Initial Trajectory

A recent study introduces an innovative methodology named Language Trajectory Transformer for grounding language in robotics, diverging from traditional approaches by incorporating an initial trajectory for the robotic agent. This method leverages the concept of trajectory initialization as a foundational step, establishing a baseline from which the robot can further refine its actions based on natural language instructions. At the core of this approach is a multimodal process, where the initial trajectory acts as a rough guide, encapsulating basic movement patterns and safety constraints, while subsequent refinements through natural language processing enable the dynamic alteration of this trajectory to better align with human intent and contextual nuances [15]. A pivotal component of this methodology is the integration of an image

encoder, employing pre-trained models like CLIP to detect and understand objects within the scene. This inclusion of visual context through image encoding significantly improves the robot's situational awareness [16], empowering it to make more informed decisions based on both the physical environment and linguistic commands received.

The trajectory modification aspect of this methodology warrants particular attention. Through a multi-modal transformer decoder, the system integrates the embedded outputs of language and image encoders to generate modified trajectories that closely follow user instructions. This model predicts the reshaped trajectory sequentially, considering the semantic objectives derived from language input and the geometric relationships identified through the initial trajectory and scene understanding. The approach's flexibility is further evidenced by its capacity to adapt to changes in velocity and three-dimensional movements, ensuring that the robot's actions are not only safe and efficient but also contextually relevant and aligned with specific user commands.

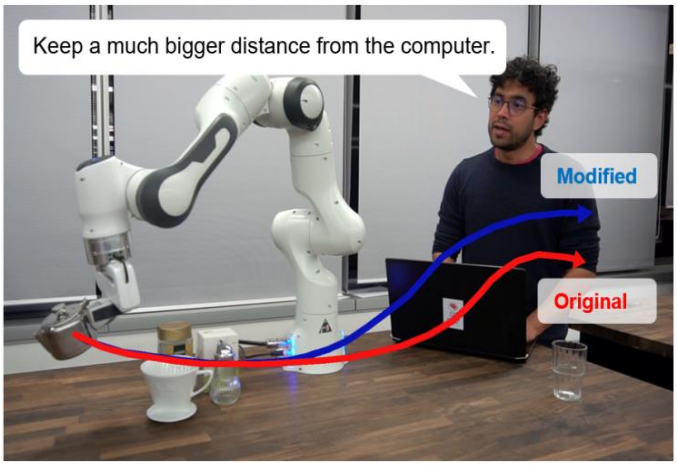


Figure 4: Trajectory modification based on user's constraints [15].

By harnessing advanced NLP techniques, cutting-edge machine learning algorithms, and sophisticated image recognition capabilities, this approach enables robots to interpret and execute complex commands with increased accuracy and adaptability [15]. This methodology not only enhances the robot's ability to understand and respond to verbal instructions but also significantly improves the fluidity and efficiency of human-robot interaction. The integration of an initial trajectory with language-based refinements, object recognition, and trajectory modification represents a promising research avenue, offering a novel perspective on the challenges of bridging the gap between human linguistic commands and robotic execution.

2.1.4 Trajectory Correction from Language Feature

Building on the trajectory modification strategy described earlier, ExTraCT (Explainable Trajectory Corrections from Language Inputs Using Textual Description of Features) introduces a novel modular approach for natural language corrections. This approach complements and enhances the capabilities of the previously mentioned methodology. ExTraCT utilises Large Language Models (LLMs) for comprehending natural language, in conjunction with a series of predefined trajectory deformation functions. Through this framework, it generates online textual descriptions for trajectory modification features, thereby facilitating a direct semantic correspondence between user utterances and these descriptions. Upon identifying a match, a relevant deformation function is applied to the initial trajectory, enabling precise adjustments based on the recognized feature [17].

The separation of language understanding from trajectory deformation augments the system's interpretability and adaptability. ExTraCT's modular design enables accurate interpretation of instructions and context-aware trajectory adjustments across diverse scenarios, eliminating the need for extensive retraining. Its focus on explainability assists in pinpointing the root causes of any discrepancies or failures in comprehending language instructions or executing trajectory modifications [17]. Thus, ExTraCT's methodology represents a significant progression in grounding language within robotics, offering a more explainable, accurate, and user-friendly method for implementing language-based trajectory corrections in robotic systems.

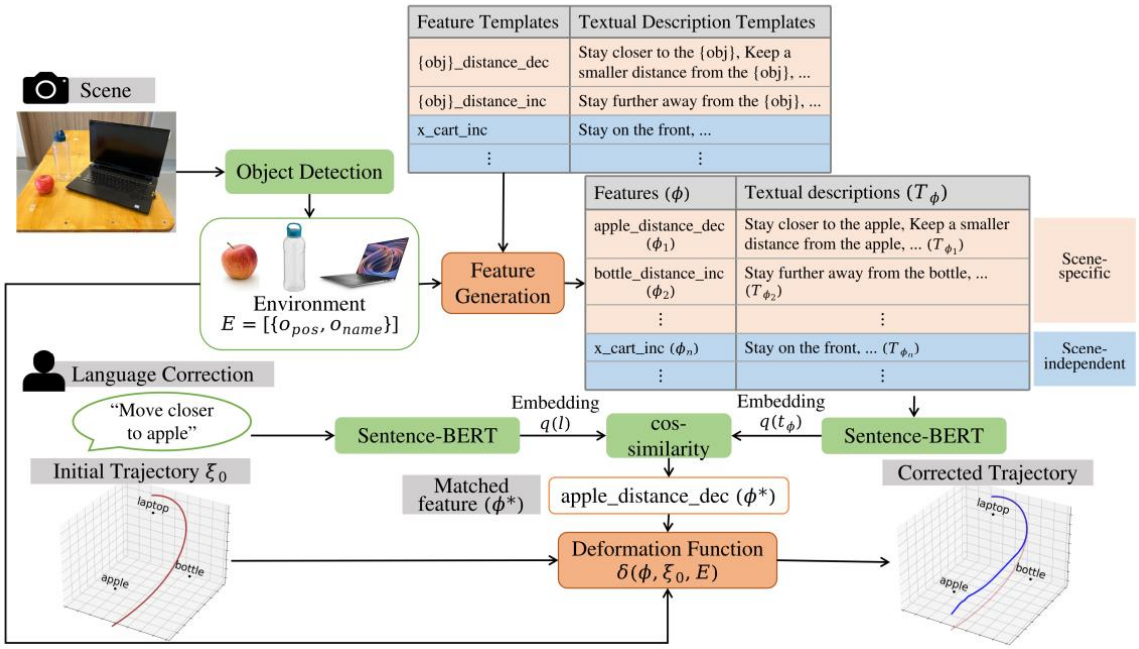


Figure 5: Trajectory modification based on user's constraints [17].

After reviewing several approaches to grounding language acquisition and its application in robotics, it becomes evident that enabling robots to interpret and act upon verbal instructions involves a complex interplay of language understanding,

contextual awareness, and action execution [18]. Grounded language acquisition stands out as a critical capability for robots, especially in nuanced applications like trajectory correction based on vocal commands. This process not only demands an understanding of human intent but also necessitates the robot's ability to integrate this understanding with the physical environment to produce actions that align with human expectations.

Particularly in scenarios where robotic arms are manoeuvred based on voice commands, the precision in maintaining a safe distance from surrounding objects underscores the importance of grounding language in the local context. This approach ensures that robots can adapt their actions to meet specific human preferences, thereby enhancing the effectiveness and safety of human-robot interactions [19]. The methodologies explored, such as ExTraCT, highlight the advancements in this domain, showcasing how combining natural language processing, machine learning, and image recognition can lead to more adaptive, efficient, and user-friendly robotic systems [16, 18]. Summarily, grounding language in robotics emerges not just as a technical necessity but as a foundational aspect that bridges the gap between human linguistic complexity and robotic action, paving the way for more intuitive, responsive, and capable robotic assistants in various applications.

2.2 Improving autonomy and performance with advanced Machine Learning

2.2.1 Language Model for Feature Generation

Recent advancements in natural language processing (NLP) technologies have spurred significant interest in identifying the most effective models for feature generation and matching tasks. Within this context, the comparison between Generative Pre-trained Transformer (GPT) models and Bidirectional Encoder Representations from Transformers (BERT) has become a focal point of discussions. Specifically, this literature review highlights the superiority of GPT, particularly in its ChatGPT iteration, over BERT for these tasks, emphasising BERT's limitations.

The GPT model, with its strong generative capabilities, has emerged as a frontrunner in the field of text generation tasks. Its architecture, designed for unidirectional training, affords it a distinctive advantage in generating coherent and contextually relevant text. This capability is especially beneficial for feature generation and matching tasks, where the ability to process and generate large sequences accurately is paramount. Conversely, the BERT model, though renowned for its deep understanding of language context and impressive performance in classification tasks, exhibits notable limitations when tasked with sentence-level data. The model's intrinsic design focuses on phrase-level processing, which hampers its efficacy in tasks requiring extensive sentence-level classification accuracy [20].

Moreover, ChatGPT, an iteration of the GPT model, demonstrates exceptional prowess in processing long sequences and maintaining context over extended texts. This is a critical advantage in feature classification tasks, where the model's capability to understand and classify complex sentences, even in zero-shot settings, sets it apart [20]. The inherent unidirectional training and text generation ability of ChatGPT also significantly outperform BERT in keyphrase generation tasks. This superiority is attributed to its enhanced understanding of context and its aptitude for integrating knowledge across various domains more effectively than BERT [21, 22]. Consequently, ChatGPT's ability to produce high-quality, accurate, and contextually relevant keyphrases across diverse document types and lengths underscores its superiority over BERT, particularly for feature generation and matching tasks.

While BERT offers promising classification capabilities, its clear limitations at processing sentence-level data effectively and its phrase-level design reduce its suitability for tasks requiring high accuracy in feature generation and classification. In contrast, the GPT model, especially in its ChatGPT iteration, presents a more capable solution, underpinned by its generative capabilities and superior performance across a broader range of NLP applications.

2.2.2 Pipeline optimisation with Speech-To-Text

Coalescing speech recognition with Natural Language Processing (NLP) is vital for allowing robots to comprehend and respond to human verbal commands, especially beneficial for individuals with disabilities where voice commands are of significant

assistance. This combination consists of two primary elements: speech recognition, which transforms spoken language into written text, and NLP, which aids robots in grasping the significance and context, thus enhancing their capability to understand language.

Nonetheless, incorporating speech recognition into robots presents various obstacles, especially in environments involving human interaction. Factors such as background noise, the nuanced nature of language, the distance between the robot's microphone and the speaker, and the presence of words that sound alike can complicate achieving high accuracy levels [23]. Although deep learning techniques can take advantage of large volumes of unlabeled speech data, they may find it challenging to decode effectively, requiring extensive fine-tuning. Problems such as overfitting with datasets of limited size and the obstacles of supervised pre-training when data is scarce are other issues that arise [24, 25].

To overcome these hurdles, innovative deep learning models like Whisper have been developed, utilising weakly supervised pre-training on broad, varied sets of audio data. This method, alongside processes to filter out low-quality training data and detect essential speech features, helps to address the challenges mentioned [25]. Remarkably, this approach does not depend on highly curated, crowdsourced datasets, making it more scalable and showcasing enhanced reliability and generalizability in various contexts [26]. The Whisper model, in particular, stands out for its high reliability, with its high 95% confidence interval for human-level performance regarding word error rate (WER) metrics. This performance illustrates its

capability to handle a wide spectrum of audio inputs with proficiency comparable to humans, highlighting its versatility in tackling different speech recognition challenges as shown in Figure 6.

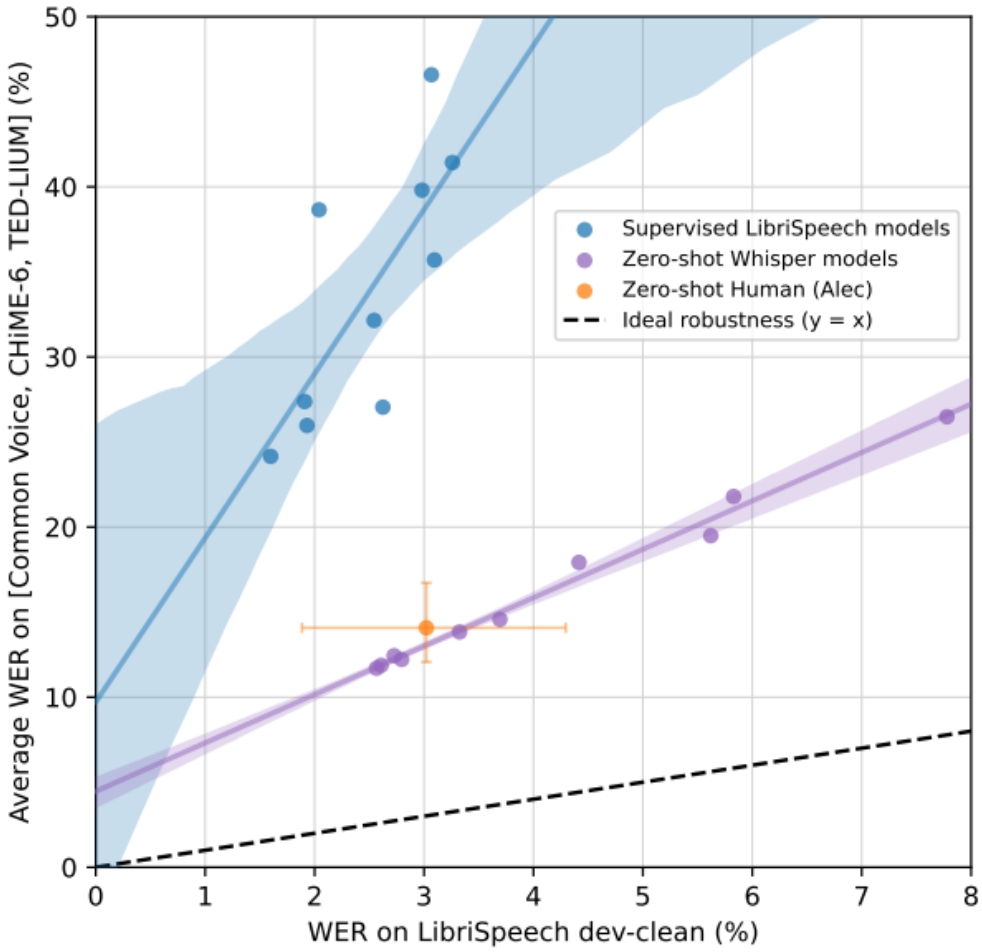


Figure 6: Trajectory modification based on user's constraints [27].

2.3 Baseline Architecture (BERT)

The baseline methodology employed in this study has been adapted from the approach explained in the ExTraCT paper [17], which is briefly discussed in **Section**

2.1.4. As noted previously in the scope, the baseline comprises solely the ExTraCT language encoding architecture, as depicted in Figure 7.

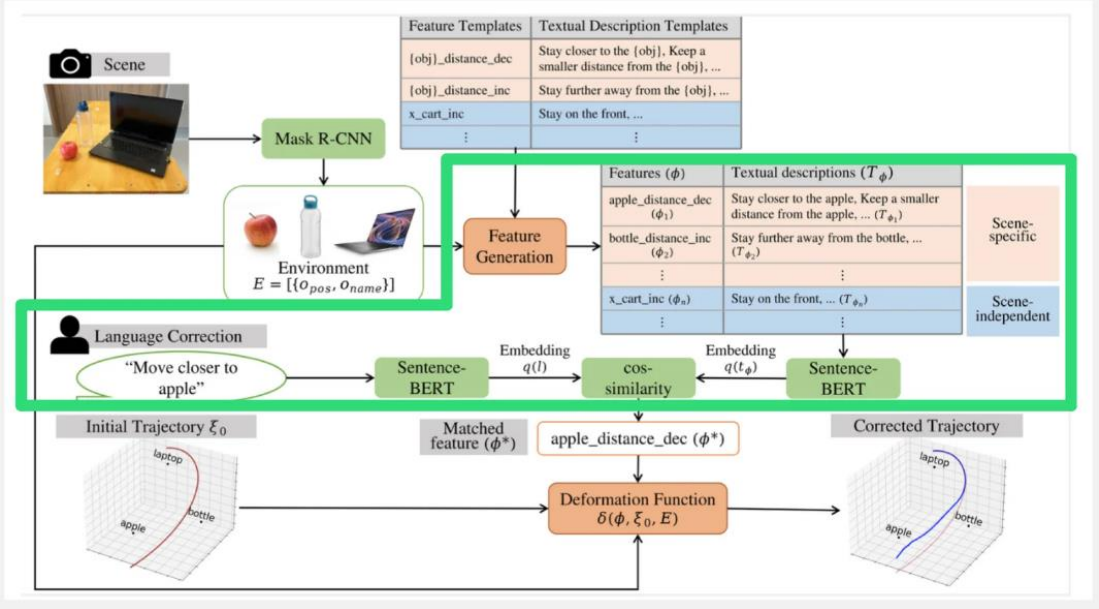


Figure 7: Baseline architecture from ExTraCT [17].

As stated in ExTraCT, it is assumed that the language correction l can be mapped to a limited set of features Φ that which may either be scene-specific[2] or scene-independent[3].

Table 1
Example Features Generated

scene-specific	**scene-independent**
Apple distance decrease	X-Cartesian decrease
Apple distance increase	Z-Cartesian increase
…	…

[2] Object related features.
[3] Cartesian related features.

Each feature $\boldsymbol{\phi} \in \boldsymbol{\Phi}$ corresponds to a deformed trajectory, i.e. $\boldsymbol{\xi = \delta(\phi, \xi_0, E)}$ where $\boldsymbol{\delta}$ is the trajectory deformation function which is described in ExTraCT [17]. Thus, this allows to obtain the most likely trajectory $\boldsymbol{\xi^* = \delta(\phi^*, \xi_0, E)}$ from the most likely feature $\boldsymbol{\phi^*}$

This reduces the problem of finding the most likely trajectory $\boldsymbol{\xi^*}$ to the problem of finding the most likely feature $\boldsymbol{\phi^*}$, hence simplifying the problem into the following equation [17]:

$$\boldsymbol{\phi^* = \underset{\Phi \in \Phi}{\operatorname{argmax}} P(\phi \mid l)} \qquad \textbf{(1)}$$

The feature matching is done with sentence similarity between the language correction and the textual description as depicted in Figure 8. This comparison is achieved using similarity metrics such as Cosine Similarity. Models designed for sentence similarity, such as all-MiniLM-L6-v2, convert input texts into vector embeddings. These embeddings encapsulate semantic information, allowing the model to calculate the proximity of the sentences within the vector space. Through this method, these models can evaluate the level of similarity with a high degree of accuracy, significantly aiding in a variety of applications ranging from document clustering to question-answering systems.

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}},$$

where A_i and B_i are the ith components of vectors $\mathbf{A}$ and $\mathbf{B}$, respectively. **(2)**

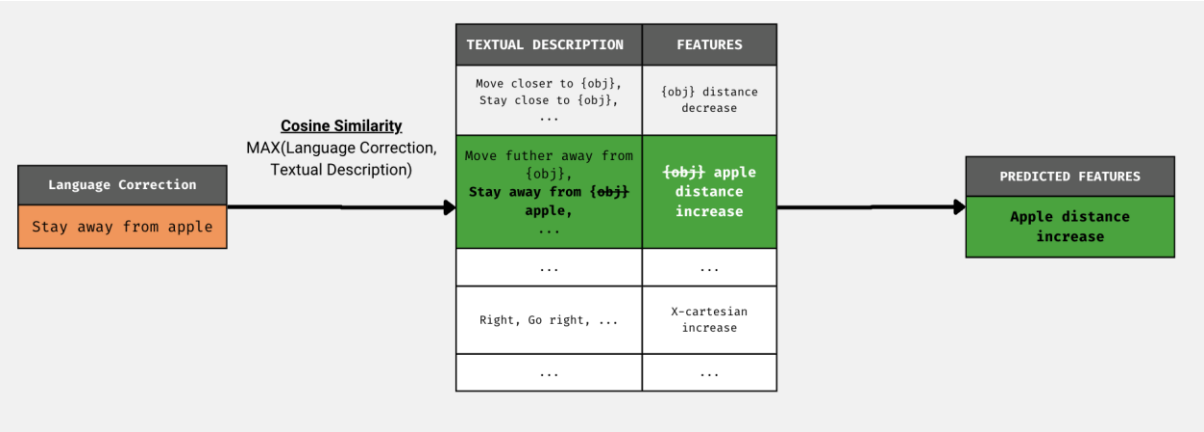


Figure 8: Baseline Feature Matching

The foundational approach is crafted to encompass pre-coded textual instructions, such as "move closer to {object}." This template translates directly to the associated feature – "{object} distance decrease". Here, the "{object}" placeholder is ingeniously designed to adapt to a variety of objects that are identified within the environment by the Mask R-CNN model. When language corrections are offered by users, these are transformed into BERT embeddings and subsequently processed through an encoding function. This pivotal function makes a comparison between the user's language correction input and every textual description available on the pre-coded list, ultimately selecting the description that exhibits the highest cosine similarity. This selected textual description is then adeptly matched to its corresponding feature from the feature template. Following this crucial matching process, the identified feature is forwarded into a deformation function. This function is tasked with fine-tuning the initial

trajectory to better accommodate the user's correction, ensuring that the system’s response aligns more closely with user expectations and enhances overall performance.

To summarise, the literature review meticulously addresses various methodologies for integrating language grounding into robotics applications, emphasising the substantial limitations associated with the extensive training requirements and difficulties in processing complex chained commands. This highlights an urgent imperative to refine the approach to accommodate more advanced textual commands efficiently. Moreover, prior studies have not adequately combined these methodologies into a cohesive autonomous pipeline, indicating a gap in the current research landscape. To advance towards a more autonomous application, the integration of the Whisper speech-to-text model is deemed essential, enabling the use of voice commands directly, thus eliminating the reliance on manual typing. Finally, this project will leverage components of the ExTraCT architecture as the foundational approach, serving as a benchmark for evaluating new methodologies.

Chapter 3

Methodology

3.1 Problem Definition

The goal of this report is to develop a robust pipeline designed to handle a wide range of user language corrections, from straightforward single sentence commands to more complex, chained commands using GPT models. This enhancement is envisioned through the integration of a speech-to-text functionality, specifically tailored to assist users who are either unable to type or have limited mobility. By implementing this feature, the process of modifying trajectories based on users' voice commands becomes more seamless, facilitating the execution of multiple chained commands within a singular input.

3.2 Using a larger embedding model

This research investigates the efficacy of employing a larger embedding model (1536 dimensions compared to baseline of 384) for evaluative purposes, particularly to assess whether transitioning to a more advanced embedding model significantly enhances performance metrics in processing commands. The implementation of the ada-002 model permits a direct assessment of whether the limitations observed in the baseline architecture, as delineated in ExTraCT [17], originate from the constraints of the Large Language Models (LLMs) or are inherent to the architecture itself. This

evaluation focuses on comparing ada-002 with the previously used all-MiniLM-L6-v2 embedding model, which will be expounded upon in Chapters 4 and 5 of this report.

Rank	Model	Model Size (GB)	Embedding Dimensions
75	all-MiniLM-L6-v2	0.09	384

Figure 9: all-MiniLM-L6-v2 MTEB rank.

Rank	Model	Model Size (GB)	Embedding Dimensions
41	text-embedding-ada-002		1536

Figure 10: ada-002 MTEB rank.

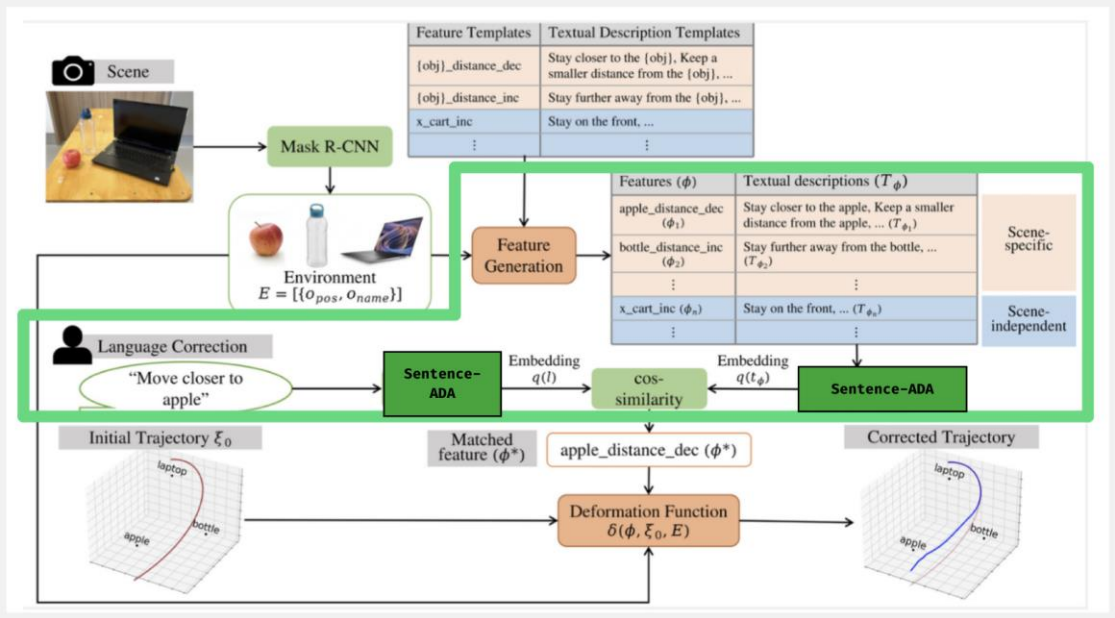


Figure 11: ada-002 sentence similarity architecture.

3.3 Divide & Conquer: Strategy behind GPT-4 preprocessing

The newly proposed architecture employs the divide and conquer algorithm, a cornerstone strategy in computer science designed to tackle complex issues by decomposing them into smaller, more manageable sub-problems. This methodology encompasses three principal steps: division of the main problem into smaller sub-problems, resolution of each sub-problem by directly solving them if they are sufficiently simple or by further subdivision if not, and integration of the solutions to these sub-problems to formulate a comprehensive solution to the initial problem. This technique finds application in numerous algorithms, notably in sorting algorithms such as QuickSort and MergeSort, as well as in algorithms dedicated to computing the Discrete Fourier Transform. Its efficacy in addressing complex challenges renders it an invaluable asset in the algorithm developer's toolkit, thus meriting exploration in the context of resolving intricate chained commands.

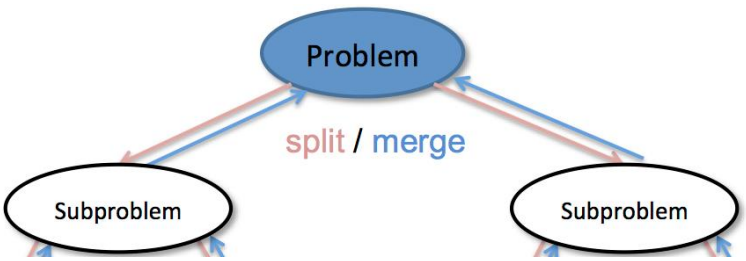


Figure 12: Divide and conquer strategy [28]

3.3.1 GPT: Split complex commands

Breaking complex chained commands into simpler, singular commands is facilitated by employing a divide-and-conquer strategy. This tactic simplifies the problem by breaking it down into manageable parts. To apply this to the current challenge, meticulous analysis was done on the structure of complex sentences and strategizing on how to effectively split them. By leveraging the capabilities of sophisticated models such as GPT-4, which is designed to comprehend and generate human-like text, further refining the approach through prompt engineering. This involves crafting prompts that guide the model in understanding and executing the task of dismantling complex commands into simpler ones efficiently (**APPENDIX A**).

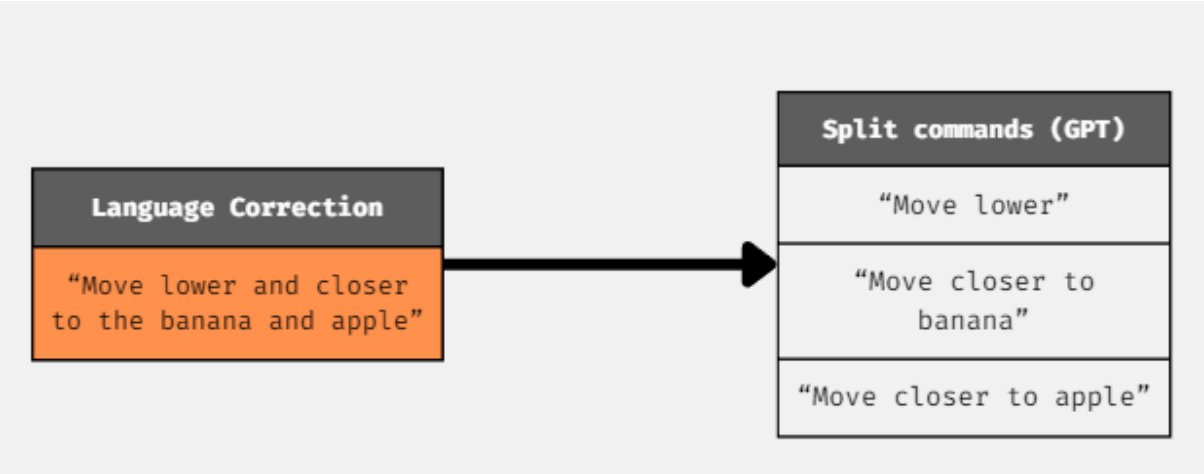


Figure 13: Splitting complex command into individual commands.

Prompt engineering, a critical component of the strategy, has been explored extensively to find the most effective method for splitting complex commands. After experimenting with various techniques, the zero-shot prompting approach has consistently stood out for its effectiveness. Zero-shot prompting does not require prior

examples to direct the model's responses, making it an ideal choice for the objective. This method has demonstrated remarkable success in breaking down complex commands, streamlining the process, and paving the way for more efficient handling of such tasks.

3.3.2 GPT: Object extraction

The proposed approach introduces a method for verifying whether an object from the user's input is present within the scene. It acts as a mechanism for confirming that the object mentioned by the user indeed exists within the scene; otherwise, it notifies the user of the object's absence, prompting them to retry their commands with an object that is appropriate to the scene. The prompt for this task can be found in **APPENDIX B**.

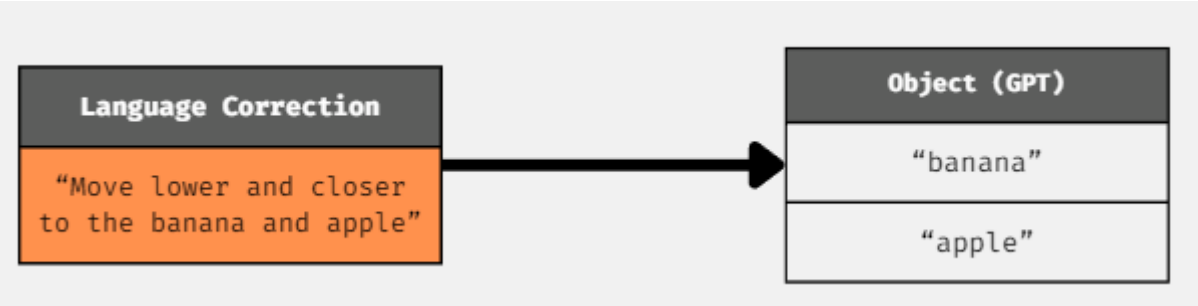


Figure 14: Object extraction.

3.4 GPT-Ada: GPT preprocessing & Feature classification using Ada

As presented in the description of the preprocessing steps for GPT-4, the approach introduced herein leverages the GPT-4-turbo API to perform dynamic object extraction and command splitting. By executing these two preprocessing API calls

asynchronously, it becomes feasible to perform a comparative analysis between each processed command and its respective textual description. This methodology enables the precise mapping of each command to specific features, thereby effectively addressing the challenges associated with the processing of complex chained commands. This was a notable limitation of the ExTraCT method. The implementation of this strategy not only simplifies the analysis of commands but also substantially improves the system's capability to comprehend and accurately execute complex chained instructions. The matching of features is conducted using the ada-002 embedding model briefly described in **Section 3.2**.

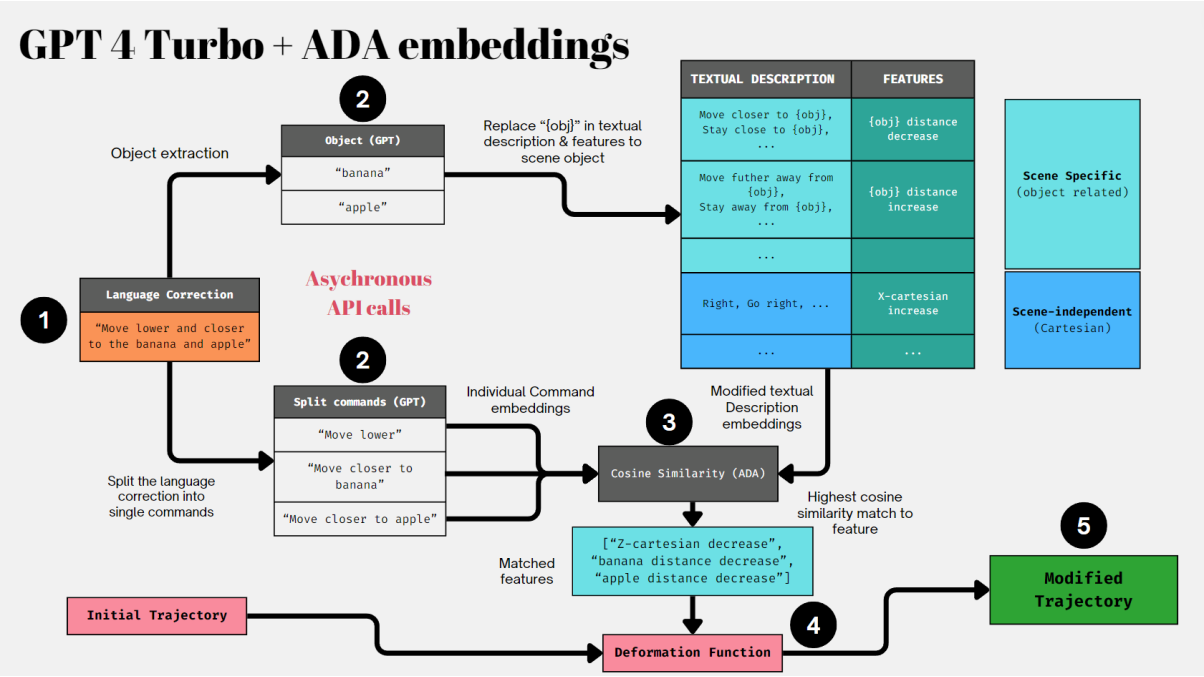


Figure 15: GPT-Ada Architecture

While this strategy comes with its own set of advantages, breaking down complex commands into multiple simpler ones actually leads to an increase in time complexity from O(N) to O(N^2) This is because it necessitates the execution of an iterative loop

that compares each individual command against all textual descriptions, thus significantly slowing down the process. Moreover, although ada embeddings outperform the baseline embeddings, they are not ideally suited for tasks related to feature generation. According to the findings presented in the literature review, GPT models, especially in feature generation and matching, exhibit superior performance. Consequently, there's a clear need to further refine the current pipeline. One promising direction could be to entirely shift away from using embeddings for sentence similarity, instead opting for the more advanced capabilities of GPT-4 Turbo. This change could potentially streamline the process, optimising it for better efficiency and accuracy in handling complex linguistic tasks.

3.5 GPT-Only: GPT preprocessing & Feature classification

Contrary to the strategy outlined in **Section 3.4**, the current method exclusively employs GPT across all three phases. By integrating an additional prompt and an extra API call into the GPT-4 Turbo architecture, its capabilities are substantially bolstered, allowing it to execute feature matching and classification tasks with exceptional efficacy. It leverages the outcomes from the split commands API call as inputs. This augmentation is designed to transpire subsequent to the initial feature generation phase, wherein objects are scrutinized for dynamic attributes such as variations in distance (either escalating or diminishing). These dynamic properties, coupled with Cartesian features, constitute the core data that is subsequently fed into the feature classification API call.

By feeding GPT with detailed prompts and the generated features (see **APPENDIX C**), the model is equipped to deliver accurate predictions concerning the correct feature mappings. The prompt directs GPT to perform the requisite classification using common sense reasoning; for example, an object's approach should be classified as 'object distance decrease'. Furthermore, the one-shot prompting technique is employed to provide the language model with a contextual example, thereby enhancing its understanding of the task at hand. Additionally, reference axes are labelled and provided to GPT to ensure the desired output is achieved. This process capitalises on GPT's advanced ability to interpret and analyse data, enabling it to yield predictions with remarkable confidence. This innovative methodology not only simplifies the feature matching and classification process but also significantly improves the accuracy and reliability of the predictions generated by GPT.

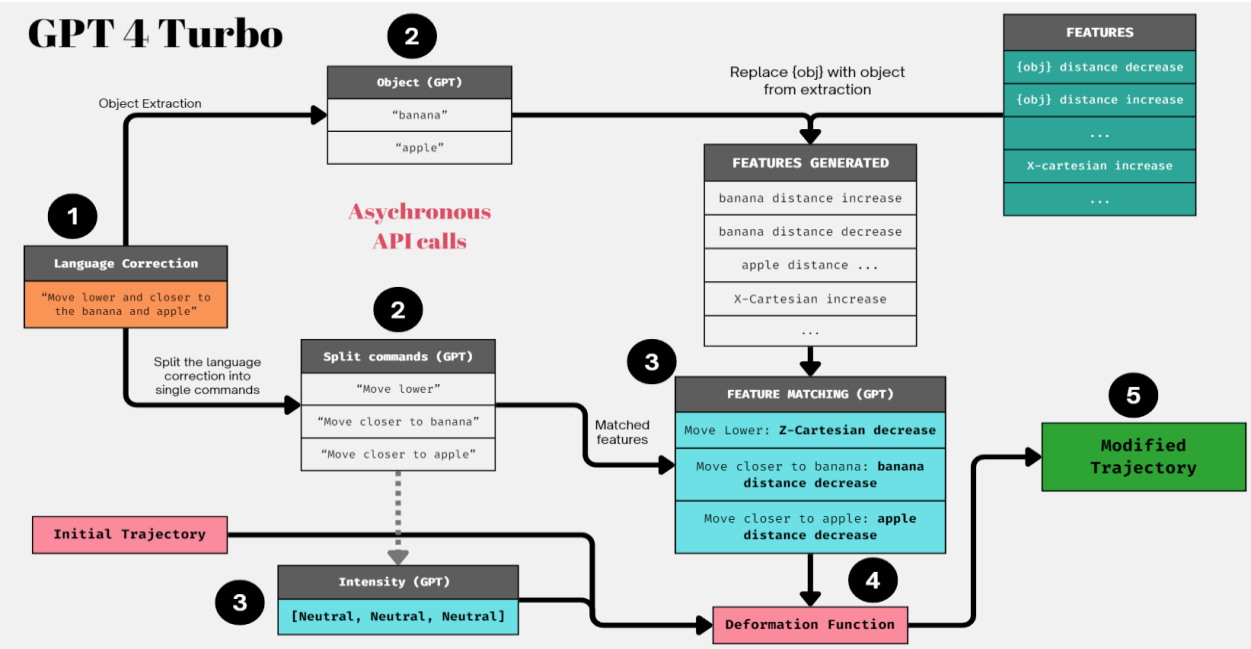


Figure 16: GPT-Only Architecture

3.5.1 Intensity recognition

Concurrent to the invocation of the feature matching API in GPT-only approach, the outcome of the split command API is parsed into a subsequent API request. This request is specifically designed to categorise the intensity levels of each distinct split sentence into one of three predefined categories: low, neutral, or high. Detailed labels are explicitly annotated within the prompt to guide GPT in generating outputs that align with the categories, based on the adverbs contained within the users' commands. These adverbs signify the degree of intensity in the users' preferences, thereby ensuring that the API response accurately reflects the desired intensity levels for trajectory modifications, in close alignment with users' expectations.

3.6 Integrating speech to text

Enhancing user experience with voice recognition is not just about embracing cutting-edge technology; it involves cultivating an intuitive, efficient, and accessible digital ecosystem. The previous workflow pipeline falls short in providing a user-friendly interface, hindering seamless interaction with the robotic arm. Within the framework of the ExTraCT pipeline, it necessitates users to input commands manually via a keyboard, a process that proves to be cumbersome. By incorporating voice recognition technology, which has witnessed remarkable improvements, this new approach enables users to interact with the entire pipeline through natural language commands. This incorporation significantly elevates the user experience, rendering digital interfaces more accessible to a wider audience.

3.6.1 Whisper Transcription

Accessibility is at the core of enhancing user experience through voice recognition. By allowing users to communicate with technology using voice commands, it effectively break down barriers for individuals who may struggle with traditional input methods. This includes people with visual impairments, movement difficulties, or those who simply find typing cumbersome due to injury or discomfort. Voice recognition serves as an inclusive tool, ensuring everyone has equal access to digital resources and services.

Rather than relying on the traditional method of providing textual input for language correction, a simple voice command can now perform the same task with ease. This new approach means users no longer need to manually type out their requests or corrections. Just by speaking into the device, users can achieve the desired outcome. This addition reduces the time and effort required for such tasks, making the process not only more efficient but also more accessible to a broader range of users, including those who may have difficulties with typing or with the written word.

As evaluated earlier in the literature review, Whisper model stands out for its remarkable ability to transcribe speech with high accuracy, even amidst noisy surroundings. This showcases the model's position at the cutting edge of contemporary technology, making it highly versatile for various applications. As advancements in voice recognition technology forge ahead, interactions through this medium can be expected to evolve into more fluid and natural experiences.

3.6.2 Whisper Transcription Correction

Although Whisper has been trained on a wide variety of languages, dialects, and accents, enabling it to achieve remarkable accuracy across an extensive range of vocal inputs, it is imperative to acknowledge that, akin to any automated system, Whisper, despite its impressive capabilities, is not devoid of limitations. The fidelity of its transcriptions can be compromised by factors such as homophones—words that present a similar auditory profile but diverge significantly in meaning—and objects that are referred to by multiple appellations. Furthermore, variations in accents exacerbate transcription inaccuracies. The endeavour to rectify errors emanating from accent disparities poses a formidable challenge, necessitating extensive fine-tuning of the Whisper model. This process demands a substantial allocation of time and resources. Consequently, this report concentrates on ameliorating transcription errors attributable to phonetically similar words and objects bearing multiple names. It underscores the critical necessity for additional interventions to enhance transcription accuracy.

In pursuit of augmenting transcription precision and, concomitantly, the fidelity of feature matching, this report advocates for the incorporation of a GPT post-processing strategy. Adhering to the protocols recommended in OpenAI's documentation[4], the application of GPT-4 for post-transcription refinement is poised to amend inaccuracies, including those engendered by homophones or items with multiple nomenclatures. Moreover, GPT-4 is equipped to rectify improperly transcribed entities stemming from erroneous pronunciations or accent variations, employing meticulously devised

[4] https://platform.openai.com/docs/guides/speech-to-text/improving-reliability

prompts. This approach accentuates the promise held by GPT-4 to substantially enhance transcription accuracy, thereby ensuring elevated integrity within the transcription process.

To ascertain the more effective methodology, this study examines two distinct prompts, as indicated in **Appendices E** and **F**. The inaugural prompt focuses on rectifying mistranscribed entities according to their semantic significance, by selecting the most semantically akin entity from the available objects within the scene to replace the erroneously transcribed one. This necessity arises from the phenomenon where certain objects are known by multiple appellations; for example, an object labelled as a "**cup**" may not be recognized if referred to as a "**mug**." Consequently, this strategy involves the amendment of such entities to their closest semantic equivalents within the scene. In contrast, the secondary prompt is designed to replace the mistranscribed entity with another within the scene that exhibits the greatest phonetic similarity, thus aiming to mitigate the challenges posed by homophones. An illustrative case involves the phonetic resemblance between the words "**vase**" and "**bus**," which necessitates the modification of the uttered word to match the corresponding object in the scene, based on auditory resemblance. This situation underscores the necessity for a comprehensive exploration of both prompts to determine the more accurate interpretation. Such rectification is aimed at aligning the spoken language more closely with the objects present in the scene, as identified by the perception module. The overarching methodology proposed by this investigation is depicted in Figure 19.

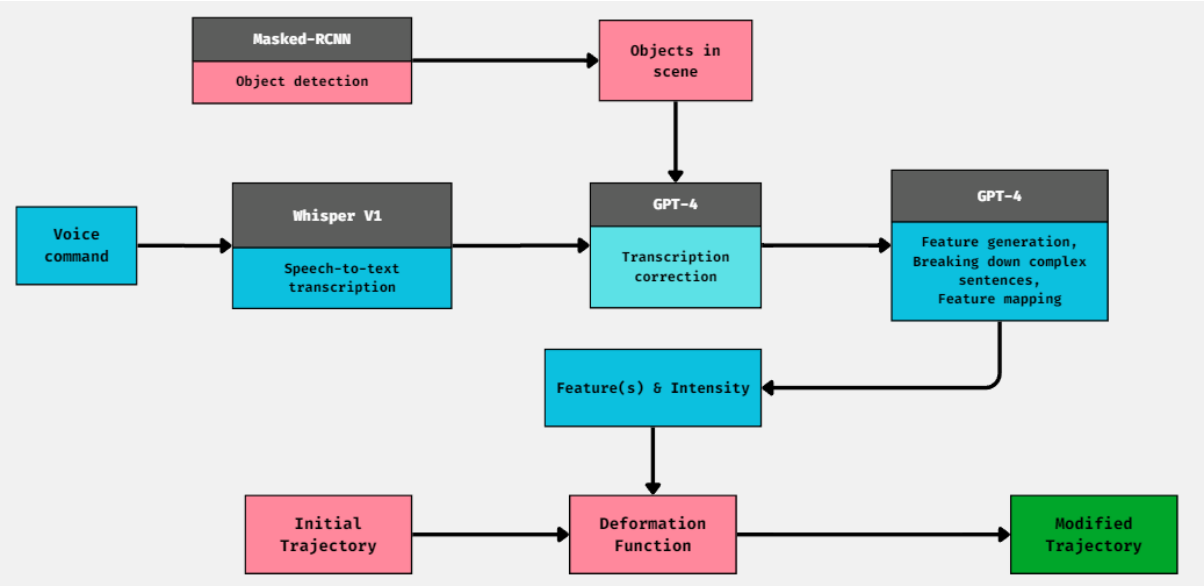


Figure 17: Overall Pipeline

Chapter 4

Feature Matching Benchmark & Results

4.1 Experiment A Overview

Experiment A aimed to evaluate the performance of feature mapping from the various approaches by comparing it against the baseline. The principal objective of this experiment was to determine whether the proposed new approaches indeed exhibit superior performance in terms of accuracy in predicting feature matching, thereby enhancing the robustness of the pipeline. To guarantee the accuracy is exclusively reliant on the novel framework and architecture's capability, the initial transcription through whisper and its subsequent correction have been excluded from this experiment. This experiment proceeds under the assumption that the transcriptions are accurate, thereby directly incorporating textual input.

4.1.1 Experiment A

To evaluate the comparative accuracy of various approaches, feature matching accuracy assessments were undertaken, involving four distinct methods:

1. ExTraCT with all-MiniLM-L6-v2 textual embedding model, referred to as **ExTraCT-Base** (baseline).
2. ExTraCT with ada-002 embedding model, referred to as **ExTraCT-Ada.**
3. The proposed approach using GPT-4 Turbo for splitting complex commands and object extraction then using ExTraCT-ada for feature matching, referred to as **GPT-Ada.**
4. The proposed approach using GPT-4 Turbo for splitting complex commands, object extraction and feature matching, referred to as **GPT-Only.**

In the course of this investigation, the following hypotheses were posited:

- **Hypothesis 1**: ExTraCT-Ada is anticipated to surpass ExTraCT-Base in terms of performance accuracy for singular command inputs, yet it is presumed to falter in managing complex, chained command inputs.
- **Hypothesis 2**: The incorporation of GPT-driven preprocessing steps in GPT-Ada and GPT-Only is expected to facilitate the processing of complex chained commands, thereby enhancing performance accuracy.
- **Hypothesis 3**: The application of GPT-4 Turbo feature matching in GPT-Only is hypothesised to yield superior results in comparison to GPT-Ada, which relies on ada-002's cosine similarity for feature selection. This expected outcome is attributed to the inherent limitations associated with embedding models.

- The structured investigation of these hypotheses aims to highlight the effectiveness and limitations of each method, ultimately contributing valuable insights into the field of feature matching accuracy within advanced computational architectures.

4.1.2 Dataset

The dataset devised for this experiment has been specifically designed to assess the performance of a robotic arm in interpreting and executing commands that reflect real-world scenarios. Comprising 150 commands, the dataset is methodically segmented into two categories: single commands (55) and complex, chained commands (95). This dichotomy is pivotal for evaluating the robotic arm's proficiency across a range of tasks, from straightforward to complex based on the different methods presented in Chapter 3 Methodology. Similar to ExTraCT paper, the dataset only covers two types of trajectory modifications - cartesian changes and object distance changes.

4.1.3 Scope of Evaluation

As previously detailed in **Section 3.2**, the approach undertaken by ExTraCT encompasses a comprehensive methodology. However, the evaluation specifically requires the examination of the language encoder section alone, as depicted in Figure 19, which corresponds to the green bounded region illustrated in Figure 11 in Section 3.2.

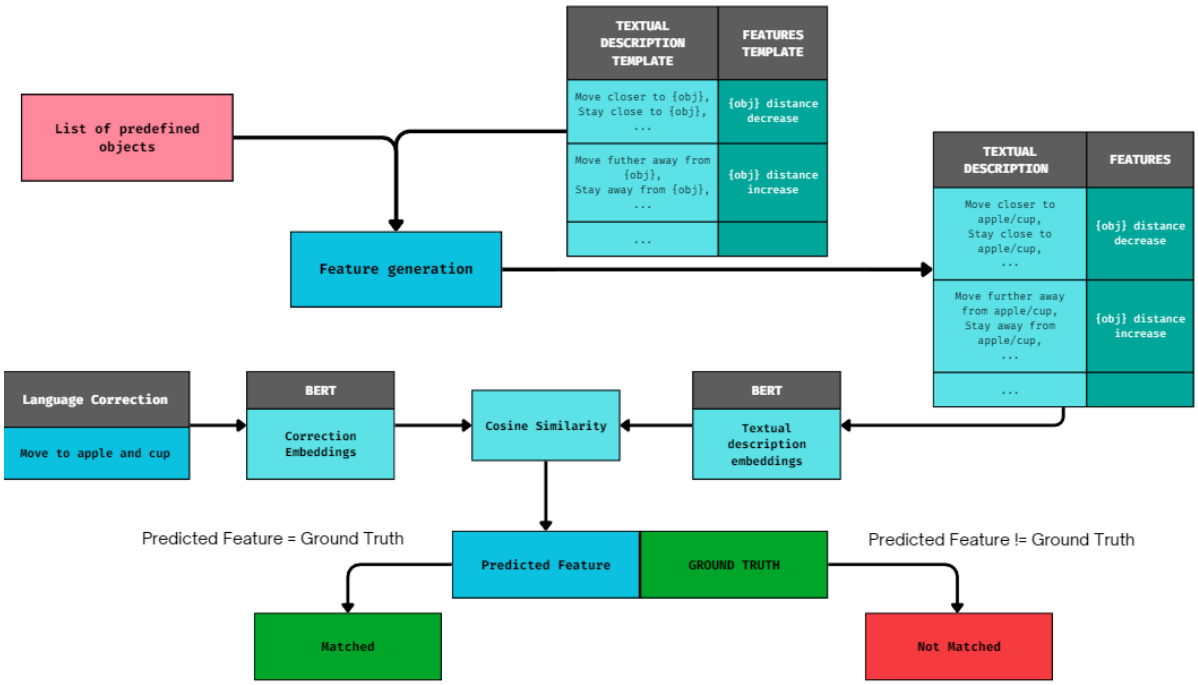


Figure 18: ExTraCT-Base evaluation flowchart.

4.1.4 Evaluation Metrics

To assess the performance accuracy of each new methodological approach, manual annotations were performed for object extraction, command splitting, and feature matching to serve as a ground truth for comparison with the respective predictions generated by the pipeline. Every individual API call was scrutinised to determine whether errors could be attributed to specific steps, potentially leading to a cascading effect of inaccuracies in subsequent predictions from the API calls. If the output of an API call matched the ground truth, it was deemed a correct prediction.

Ultimately, this evaluation functions as a benchmark for the accuracy of feature selection, a paramount aspect in attributing a deformation function to alter the trajectory towards the desired path. Figure 20 depicted below illustrates the various paths undertaken by different methodological approaches in predicting the features to

be parsed to the deformation function. The accuracy that serves as a benchmark is presented in **Step 4** of Figure 19, which pertains to selecting the correct prediction features.

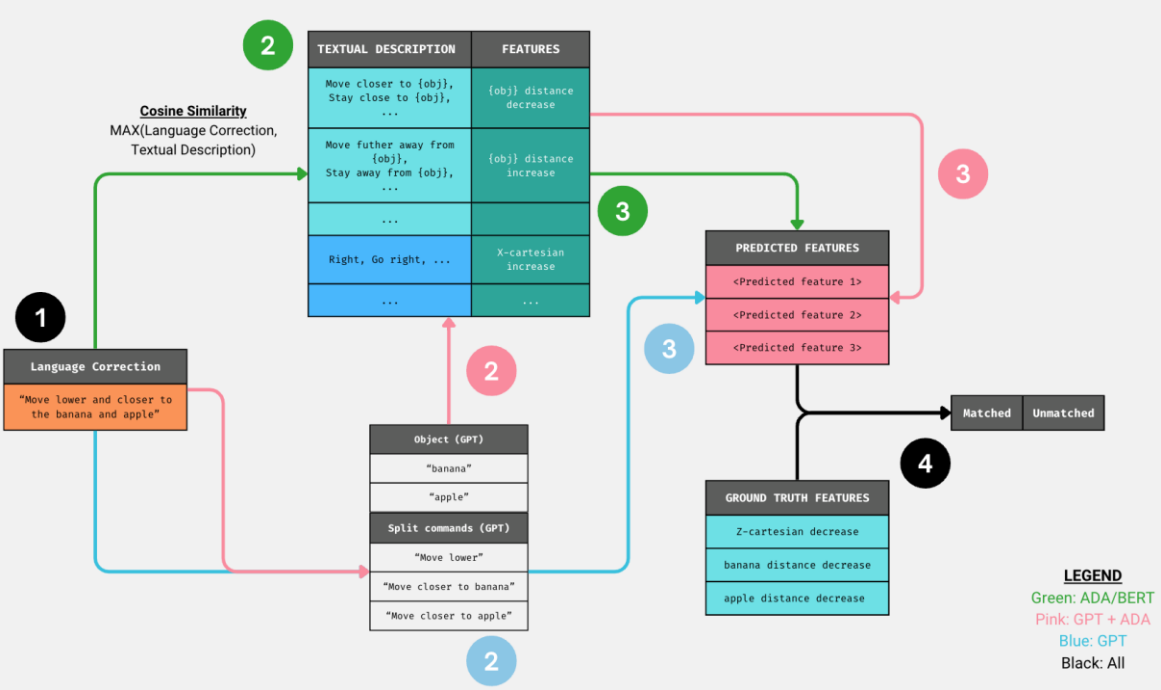


Figure 19: Evaluation for performance accuracy of feature matching.

4.2 Results

Table 2
Benchmark performance results among different methods of approach.

Approach	Performance Accuracy		
	Single	Complex	Total (%)
ExTraCT-Base	48/55	0/95	32.0
ExTraCT-Ada	55/55	0/95	36.7
GPT-Ada	55/55	84/95	92.6
GPT-Only	55/55	91/95	97.3

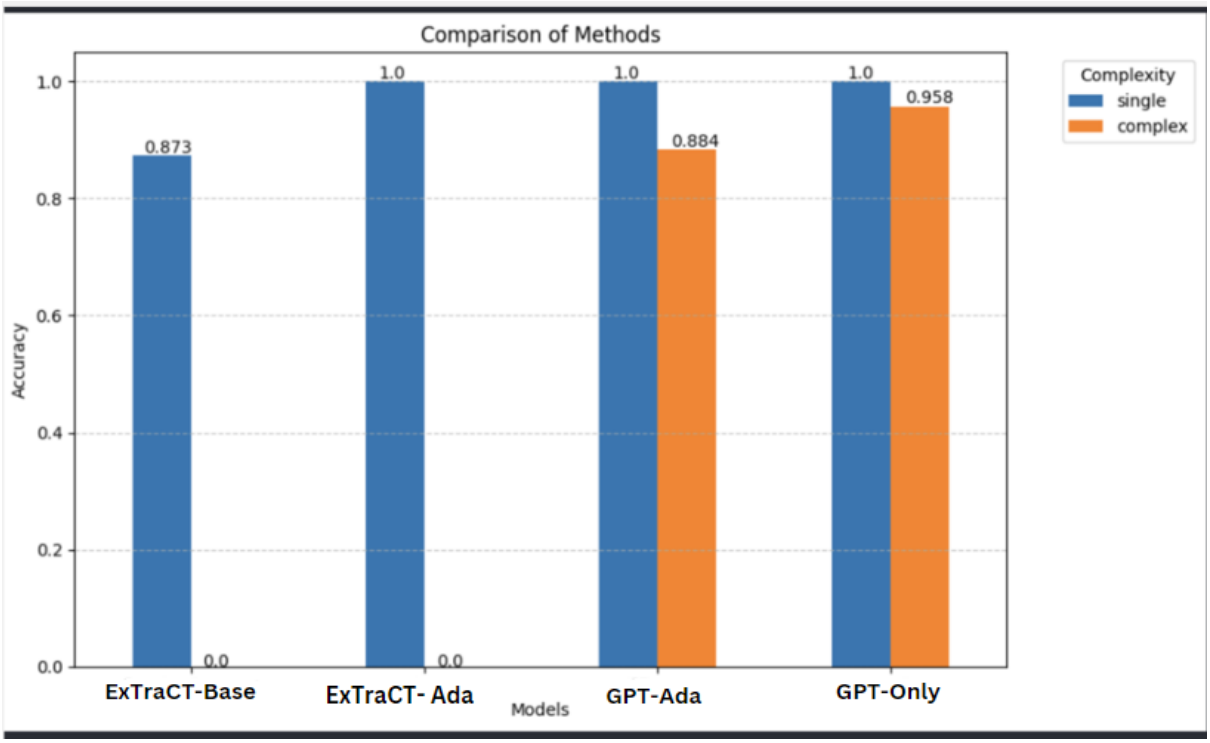


Figure 20: Visualisation for performance among the four methods.

Table 2 and Figure 20 presents the benchmarking performance results among the different approaches. It distinctly highlights the significant improvement in performance from baseline all the way to the GPT approach, showing support for the hypothesis listed above. To understand the boost in performance for each new approach compared to the previous, a thorough inspection is done to check for each different case of data - single and complex commands and also to evaluate any errors detected at each step taken along the pipeline (object extraction, split commands and feature selection).

4.2.1 Analysis of failure cases

A: ExTraCT-Base failure cases compared to ExTraCT-Ada

While the baseline architecture generally performs well for single sentence commands, it is important to note that it exhibits several failure cases due to its limitations in interpreting commands with opposite meanings. This issue arises because the approach employs sentence similarity, as discussed in the Methodology section. However, it lacks the capability for semantic similarity, leading to incorrect feature matching. For instance, it may erroneously interpret sentences with contrasting meanings, such as "Stay away from the bottle" versus "Get close to the bottle." This discrepancy is vividly showcased in Table 3 below, where sentences are sometimes mapped to textual descriptions that convey completely opposite meanings. Furthermore, the model encounters difficulties in processing complex, sequential commands. It tends to map outputs to individual instruction sentences, rendering it incapable of handling multiple instructions simultaneously. This limitation highlights a significant area for enhancement, particularly in applications that demand nuanced comprehension and multitasking capabilities. Substituting the baseline model with ada-002 significantly enhances the feature alignment for individual commands, as illustrated in Table 4. This improvement supports **Hypothesis 1** and demonstrates that ada-002 serves as a more precise embedding model.

Table 3
Baseline Embedding model inability to

Mode	Sentences	Similarity score
Source sentence	Stay **away** from the bottle	-
Sentence to compare	Go close to the bottle	0.772
Sentence to compare	Move further **away** from bottle	0.759

Table 4
Comparison between **ExTraCT-Base** and **ExTraCT-Ada** on test case no. 49

Methods	**ExTraCT-Base**	**ExTraCT-Ada**
Language Correction	"Move away from bottle and then closer to cup"	
Predicted Features	["bottle distance decrease"] (Wrong matching for bottle, Missing cup feature)	["bottle distance increase"] (Missing cup feature)
Ground Truth Features	["bottle distance increase", "cup distance decrease"]	

B: Failure cases of ExTraCT-Base and ExTraCT- Ada architecture compared to GPT-Ada architecture

Upgrading the embedding model size significantly enhances accuracy in single command feature matching; however, it is unequivocally evident that adopting the identical architecture from ExTraCT is ineffective for processing complex chained commands. As illustrated in Table 4, the embedding models exhibit a limitation in only identifying the initial feature within a language correction. This limitation underscores

the current approach's failure to dissect commands into multi-step sequences and to interpret them in a sequential manner. This issue stems from the intrinsic limitations of the sentence similarity technique, which requires processing the sentence in its entirety.

Table 5
Comparison between **ExTraCT- Ada** and **GPT-Ada** on test case no. 49.

Methods	**ExTraCT- Ada**	**GPT-Ada**
Language Correction	“Move away from bottle and then closer to cup”	
Predicted Features	[“bottle distance increase”] (Missing cup feature)	[“bottle distance increase”, “cup distance decrease”]
Ground Truth Features	[“bottle distance increase”, “cup distance decrease”]	

The newly proposed architecture incorporates the GPT-4 Turbo model in the preprocessing phase, facilitating dynamic object filtering for feature generation tailored to scene-dependent features. This architecture also efficiently segments complex sentences into simpler sentences, enabling individual feature matching. As illustrated in Table 5, this new approach demonstrates competence in handling both simple and complex commands, thereby supporting **Hypothesis 2** claim.

C: Failure cases of GPT-Ada compared to GPT-Only

Table 6
Comparison between **GPT-Ada** and **GPT-Only** approach on test case no. 123

Methods	**GPT-Ada**	**GPT-Only**
Language Correction	"Move down and closer to the plate and microwave"	
Split commands	["Move down", "closer to the plate", "microwave"]	["Move down", "closer to the plate", "microwave"]
Ground Truth Split commands	["Move down", "closer to the plate", "closer to the microwave"]	["Move down", "closer to the plate", "closer to the microwave"]
Predicted Features	["Z-Cartesian decrease", "plate distance decrease", "microwave distance increase"]	["Z-Cartesian decrease", "plate distance decrease", "microwave distance increase"]
Ground Truth Features	["Z-Cartesian decrease", "plate distance decrease", "microwave distance increase"]	

As demonstrated in Table 2, leveraging the context answering capability of GPT-4 Turbo results in a significant performance enhancement. A discernible increase of 4.7% in overall accuracy is sufficient to substantiate the assertion that adopting a comprehensive GPT-4 Turbo infrastructure is markedly advantageous. To elucidate this finding, Comparison 6 has been meticulously designed to explicate the phase exhibiting a bottleneck, thereby elucidating why the integration of GPT-Only approach outperforms GPT-Ada. It is evident that the bottleneck emerges during the command splitting phase, wherein it is unexpectedly observed that the split command API does

not perform accurately. This observation leads to the hypothesis that GPT might not fully comprehend the context required for proper linguistic adjustment. This hypothesis is premised on the observation that the term 'microwave' could be ambiguously categorised either as a noun, referring to the physical appliance, or as a verb, denoting the act such as microwaving a food in the said appliance. This ambiguity resulted in a feature matching accuracy to suffer.

To substantiate this theory, several similar test cases were evaluated shown in Table 7 below. It was revealed that terms such as “**microwave**”, "**monitor**" and "**bowl**”, are edge cases, and when utilised in commands that concatenate multiple objects, tend to be incorrectly parsed as a single entity. This finding elucidates the complexity of language processing and the challenges inherent in accurately deciphering and executing multi-component commands.

Table 7
Failure case due to split command error

Language Correction	Move up and nearer to the peach and monitor
Split	[’Move up’, ‘Move nearer to the peach’, ‘Monitor’]
Feature Match	[‘Z-Cartesian increase’, ‘peach distance decrease’, ‘Monitor distance increase’]
Ground Truth	[‘Z-Cartesian increase’, ‘peach distance decrease’, ‘Monitor distance decrease’]

To work around this, it is observed that adding stop word “the” before each object helps in the split command task by adding context and giving GPT a better

understanding that the Words “microwave” and “monitor” in the above examples are a continuation of objects in a single command, hence enabling GPT to split each objects into separate commands while retaining the action verb in its prefix as shown in Table 8.

Table 8
Successful case upon adding stop word.

Language Correction	Move left and nearer to the peach and the monitor
Split	[’Move left’, ‘Move nearer to the peach’, ‘Move nearer to the monitor’]
Feature Match	[‘X-Cartesian decrease’, ‘peach distance decrease’, ‘Monitor distance decrease’]
Ground Truth	[‘X-Cartesian decrease, ‘peach distance decrease’, ‘Monitor distance decrease’]

Nevertheless, despite the inaccuracies in the output for certain edge cases following the split command, the GPT-Only approach continues to demonstrate superior accuracy in feature matching compared to the GPT-Ada method, thereby establishing its reliability as the more effective strategy. This superior performance of the Full GPT approach can be attributed to its method of processing the split commands. Specifically, the approach involves compiling a list of all individual commands, which is then parsed in its entirety into the feature matching GPT API call for output classification. Given that GPT gains visibility of the entire list, wherein commands are merely divided into various sections, it is conceivable that GPT has the capacity to reconstruct the complete raw language correction by leveraging the list, thereby preserving the context for Feature Matching, and consequently achieving a higher success rate.

Conversely, in the GPT-Ada method, each split command undergoes individual parsing to the Textual Description template for feature generation. Furthermore, the cosine similarity comparison is conducted separately for each instance, leading to a significant loss of context from the original language correction.

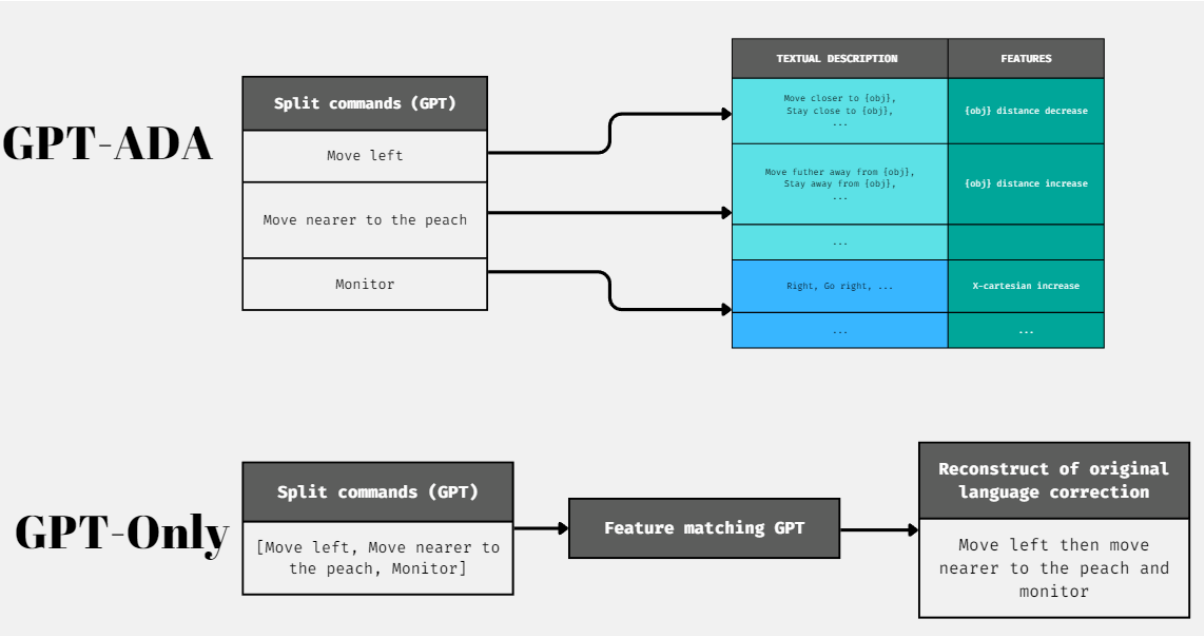


Figure 21: Feature Matching difference between GPT-Only and GPT-Ada

To summarise, the observations derived from Experiment A can be articulated as follows:

1. The implementation of a split command enhances performance significantly when dealing with complex chain commands.
2. The omission of the stop word “the” from a single-action verb that encompasses multiple objects may result in a loss of context, leading, in turn, to mismatches during feature matching.
3. Despite the occurrence of splitting errors, the GPT-Only approach retains the ability to accurately match features, likely owing to its capacity to reconstruct the list of split commands, thereby restoring the original language correction for context comprehension.

Chapter 5

Whisper Correction Experiment & Results

5.1 Experiment B Overview

Experiment B aimed to investigate the potential improvement in transcription accuracy achieved by integrating a whisper correction post-processing step. This study assessed the ensuing impact on the overall accuracy of feature prediction and the model's proficiency in recognizing and transcribing accents of local Singaporeans. Additionally, this experiment undertook a comparative analysis of the two prompts briefly described in **Section 3.5.1**. The goal was to ascertain which prompt would be more favourable for real-world application, judging by their respective accuracies.

5.1.2 Experiment B

This experiment is designed to assess the efficacy of integrating a Whisper Correction layer with the GPT-4 Turbo API in rectifying mistranscribed objects from initial transcriptions. The process of mistranscription, characterised by its ambiguity and the uncertainty regarding its intentionality, presents challenges in accurately restoring the original intended text as articulated by the user. Consequently, this study narrows its focus exclusively on the rectification of mistranscribed objects, necessitating the establishment of certain assumptions for the purposes of this investigation.

The methodology employed for rectifying mistranscribed objects involves the utilisation of a 'ground truth' list, which serves as a simulated representation of the

actual objects present within the scene. Mistranscribed objects are then algorithmically matched to the most closely related object within this list, based on semantic relevance or phonetic similarity, contingent upon the specific prompt. Subsequently, these objects are replaced with the correct entity as determined by the ground truth list, thereby achieving rectification.

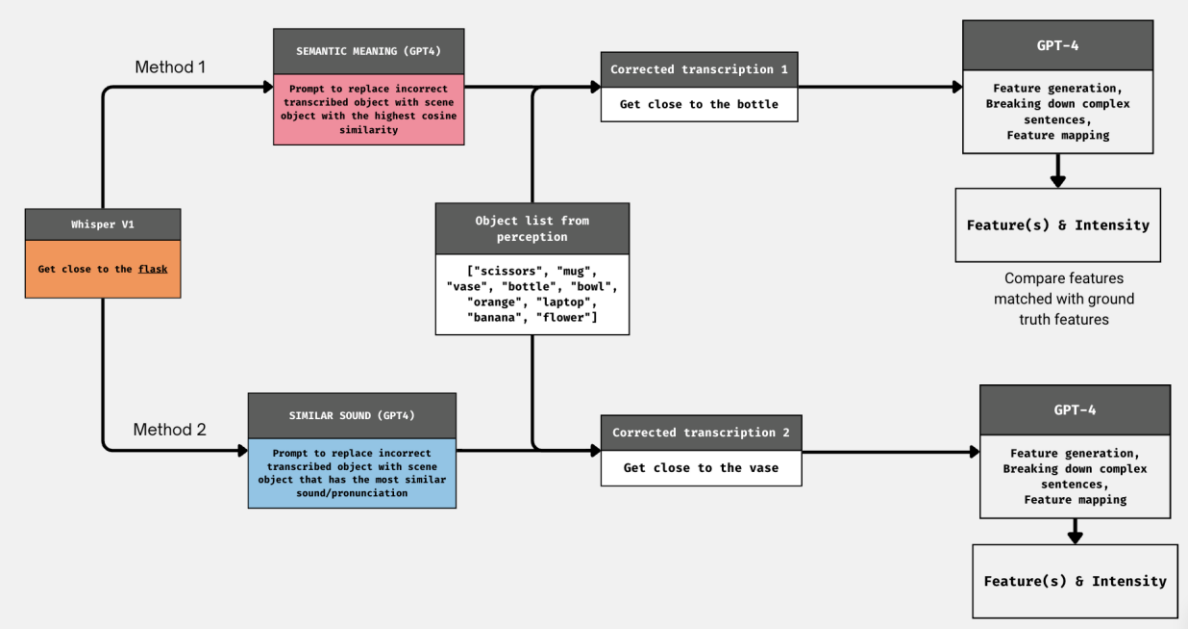


Figure 22: Semantic Meaning vs Similar Sound Whisper Correction

The assumptions in the experiment are as follows:

1. Candidates will read the commands from the dataset without deviating much from it.
2. Slight deviation by removing stop words like “the” is acceptable.
3. Objects from ground truth list represents objects on scene.
4. Zero background noise from a relatively quiet setting

Purpose of experiment:

1. Evaluate transcription performance of with and without transcription correction.

2. Evaluate feature matching performance when using Semantic Prompt vs Similar Sound Prompt for transcription correction.

5.1.2 Data Collection & Dataset

To initiate the experiment, a total of 10 test cases, encompassing both single and complex commands, were meticulously composed for the evaluation of the Whisper Correction mechanism. Participants are required to complete two iterations, yielding a total of 40 data points per participant. The dataset has been carefully designed to scrutinise the performance of the Whisper Correction in recognising object names that are synonymous with items from the ground truth list. This includes testing the system's capability to accurately substitute terms, such as replacing "**cup**" with "**mug**". Furthermore, the testing framework includes scenarios where the nomenclature of the subject matter bears phonetic resemblance to an unrelated item not included in the ground truth list. An illustrative example of this is the term "**bus**", which presents an opportunity to evaluate the system's proficiency in correcting phonetically similar prompts, in this case adjusting it to "**vase**".

A total of fourteen participants contributed to this experiment, resulting in a dataset that consists of 280 audio recordings for subsequent analysis. Given the varied permutations of race, gender, and tone across the fourteen participants, the sample size is considered sufficient for conducting a robust analysis. The data collection process took place in a room characterised by minimal noise and devoid of additional persons. The data is subsequently parsed into Semantic and Similar Sound prompts,

yielding two distinct outcomes in both transcription correction and final feature matching.

5.1.2 Profiles of Experimental Candidates

The cohort, comprising 14 participants selected for the experiment, exhibited a wide-ranging racial composition that reflected the multicultural mosaic of Singapore. Predominantly, participants were of local Chinese descent, augmented by two individuals of Burmese origin and one of Indian ethnicity. This deliberate assemblage aimed at encapsulating the rich diversity of accents found in Singapore, furnishing a robust framework for evaluating the veracity of the research findings within the realm of local dialectical variations. The inclusion of such varied profiles ensures that the study comprehensively encompasses the linguistic diversity emblematic of the Singaporean populace, offering insights into the efficacy of the model across an array of local accents.

5.1.3 Metrics

The transcription correction has two metrics to evaluate, the transcription correction accuracy and the final feature matching accuracy. While it is expected for the feature matching to fail when the transcription correction fails, it is still essential to evaluate each metrics individually to filter out edge cases where the failure in the transcription correction is due to missing or additional stop words.

For both transcription correction and feature matching tasks, ground truth annotations were manually created to serve as a basis for comparative analysis with the

predictions. The accuracy of each metric was calculated by averaging the scores of individual candidates, subsequently aggregating these averages across all candidates. This cumulative figure was then subjected to a final averaging process to derive the overall accuracy.

5.3.2 Transcription Correction vs Without

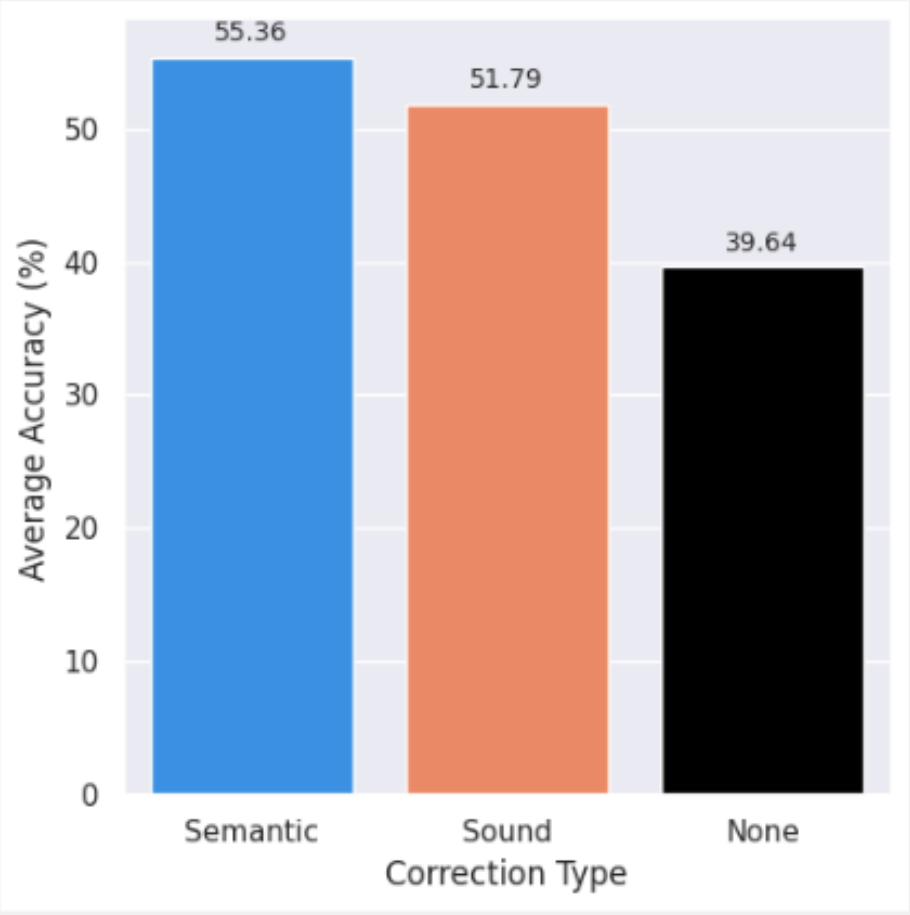


Figure 23: Transcription Accuracy

The results depicted in Figure 24 substantiate that Whisper's transcription performance is notably subpar when confronted with local accents, yielding an average accuracy of 38.64% in uncorrected transcriptions across 14 test subjects. This shortfall can likely be ascribed to the inadequacy of Whisper's training data for the English language, which has not been sufficiently tailored to accommodate the unique characteristics of the Singaporean accent. Furthermore, the typical lack of full enunciation inherent in local accents could exacerbate a challenge for Whisper to accurately transcribe the spoken words.

Although the transcription outcomes might appear less promising, it is incontrovertible that the application of transcription correction markedly improves the accuracy of transcriptions by a minimum of 12.15%. Achieving elevated levels of transcription accuracy is critical for the precise matching of features in accordance with the specified scene. Nonetheless, it is imperative to recognize the inherent challenges associated with attaining such accuracy, largely due to its significant reliance on the user. Factors including accents, the precision of pronunciation, and the speed of speech are pivotal in determining the efficacy of transcription processes.

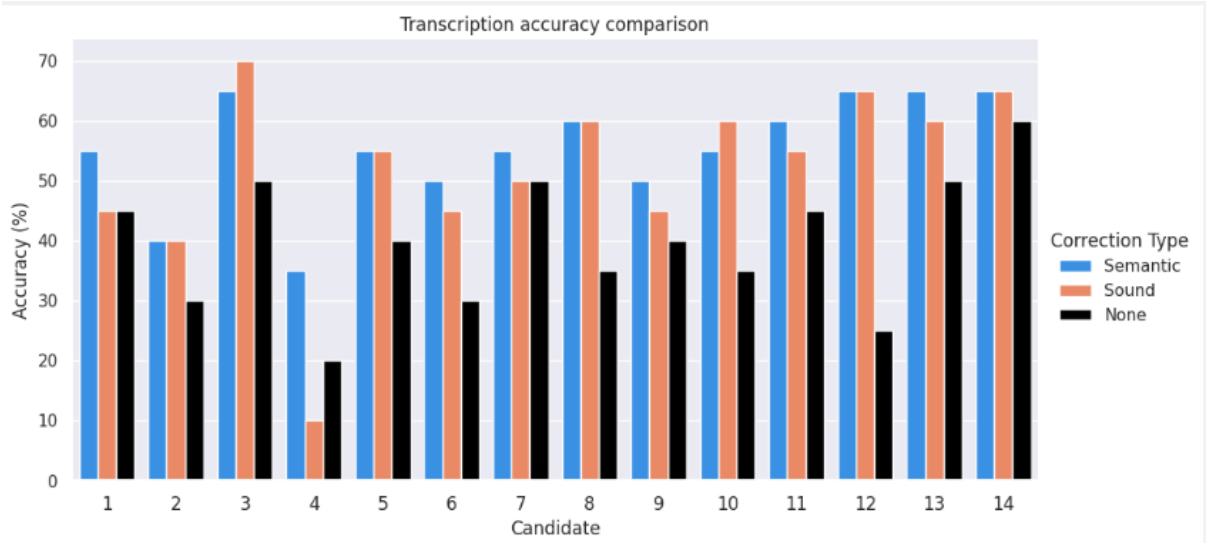


Figure 24: Candidates transcription accuracy.

Upon examining the transcription accuracy of each candidate individually, it is observed that employing the Semantic prompt results in superior accuracy overall.

5.3.3 Feature Matching Accuracy

Table 9
Feature matching accuracy for different transcription correction prompts.

Method	Accuracy in feature matching (%)
Semantic	63.2
Similar Sound	58.2

Comparative analysis of feature matching performance, utilising semantic prompts versus those based on phonetic similarity, reveals a marginal superiority of semantic prompts, demonstrating a 5% enhancement in accuracy according to the test case detailed in **Appendix I**. This enhancement aligns with the transcription accuracy depicted in Figure 24, indicating a direct correlation between transcription accuracy and the precision of feature matching.

A: Semantic post-processing prompt with GPT-4 Turbo

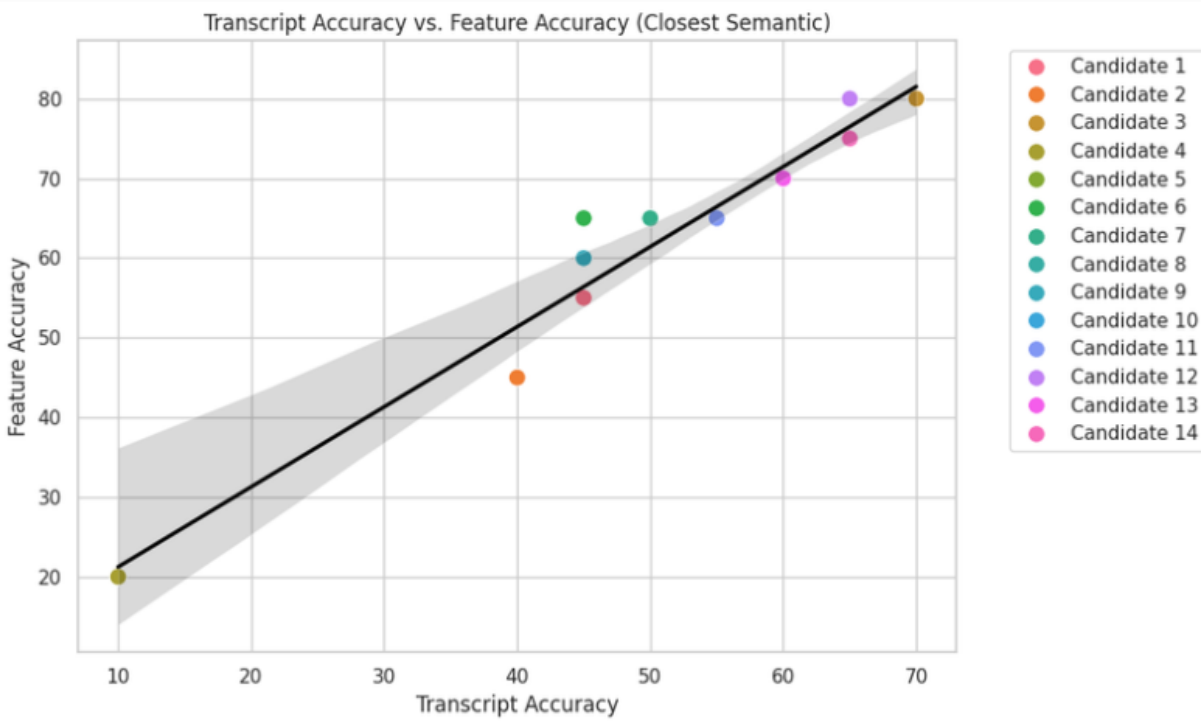


Figure 25: Semantic Prompt Feature vs Transcription Accuracy

Upon examining the accuracy of semantic prompts, it is observed that there exists a linear relationship between transcription accuracy and feature matching accuracy. Furthermore, this linear relationship suggests that the semantic prompt may be capable of predicting the success rate of feature accuracy based on transcription accuracy, thereby facilitating a more stable method of transcription correction. Promisingly, the semantic prompt demonstrates that 11 out of 14 of the candidates achieve a feature accuracy rate of 60% or above.

B: Similar sound post-processing prompt with GPT-4 Turbo

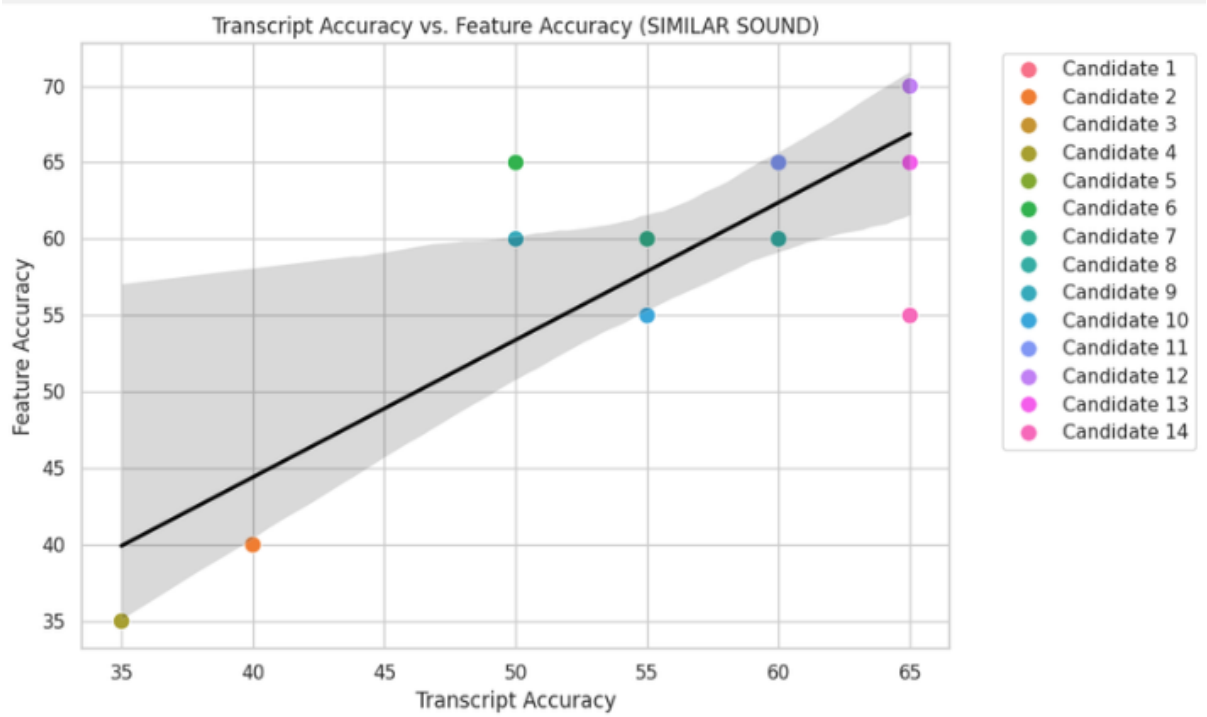


Figure 26: Similar Sound Prompt Feature vs Transcription Accuracy

In contrast, the similar sound prompt exhibits a more random distribution of data, with points that are significantly distanced from the regression line. This diffusion results in difficulty predicting the success rate of feature mapping based on transcription accuracy, thereby providing less assurance of the accuracy in feature matching. Additionally, it is worth considering that GPT 4 is not trained for phonetics capability, which explains the lower in performance compared to semantic prompt.

In summary, it is evident that the semantic prompt significantly surpasses its counterpart, the similar sound prompt, in transcription correction. Moreover, it ensures a more precise feature mapping when the transcription is accurate, making it the preferred choice of transcription correction.

Chapter 6

Conclusion & Future Works

6.1 Conclusion

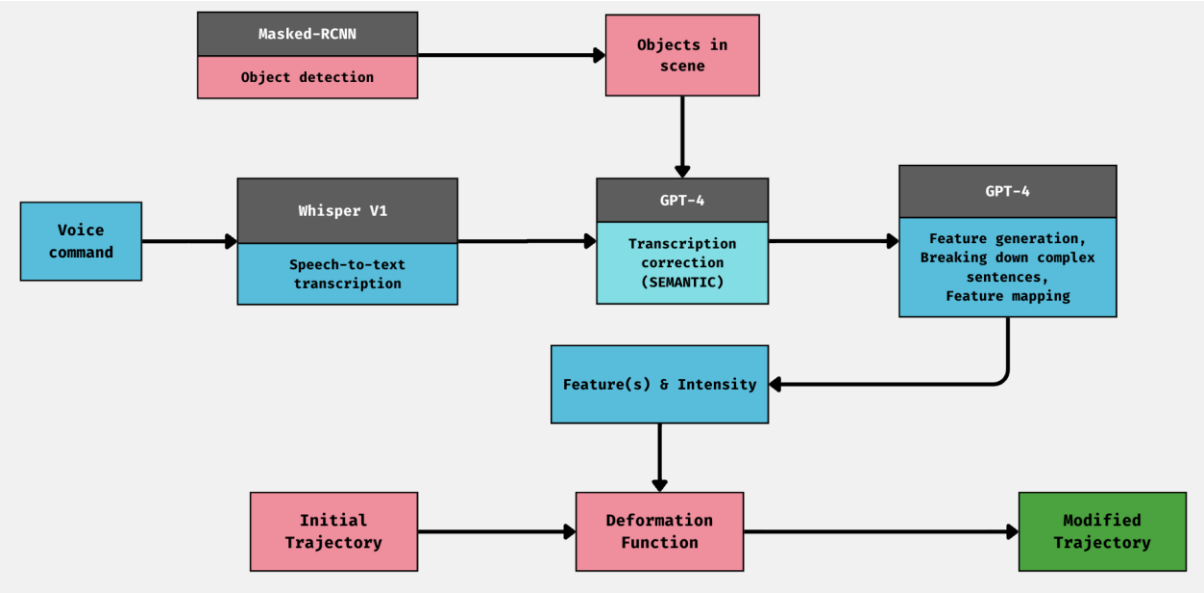


Figure 27: Full Proposed Pipeline

In conclusion, the revised pipeline, initially outlined in Figure 18 and now illustrated in Figure 28, has successfully met all project objectives.

Firstly, the new approach significantly improves performance accuracy in feature classification, thereby meeting the first objective.

Secondly, the use of a divide-and-conquer algorithm has enhanced the robustness of the system, enabling efficient processing of chained commands for language correction, which addresses the second objective.

Additionally, the third objective has been achieved by integrating a speech-to-text module into the pipeline. This module excels in recognizing and transcribing the accents of local Singaporeans, demonstrating high performance and speed in transcription, and includes GPT-4 post-processing for enhanced whisper transcription.

Finally, the successful integration of the entire pipeline with a robotic arm, which can generate the deformed trajectory for complex chained commands, fulfils the fourth objective. This integration demonstrates the pipeline's compatibility and practical application in real-world scenarios, significantly enhancing its utility and proving that each of the project's objectives has been effectively achieved.

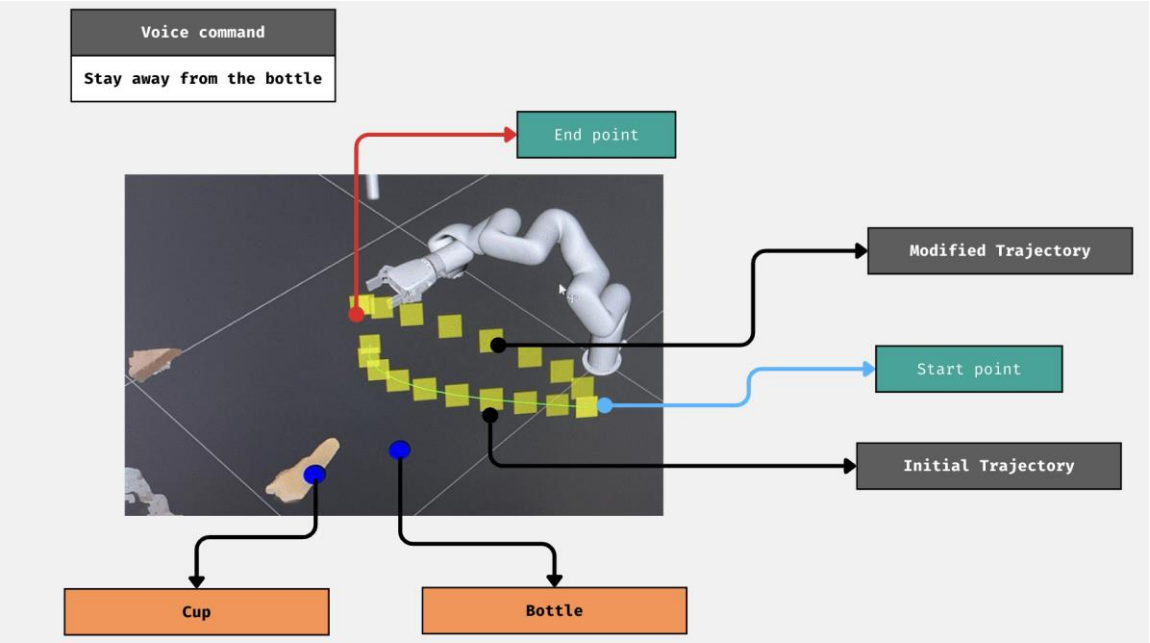


Figure 28: Modified Trajectory vs Initial Trajectory

6.4 Future Works

Future research could investigate ways to enhance the whisper transcription model. Although the incorporation of a post-processing transcription correction API call has been beneficial in augmenting transcription accuracy, further advancements might involve upgrading from the current Whisper V1 model to the Whisper V2 model within the proposed pipeline. The V2 model, benefitting from longer training durations and more extensive data, is also open source, thereby offering greater flexibility for customization according to the specific needs of the research. With a more advanced model, an improvement in transcription quality over the V1 model is anticipated.

Additionally, future studies could consider the application of real-time speech-to-text models over those that require pre-recorded audio, which tend to slow down the overall process. Although the present Whisper V1 model transcribes at a relatively quick pace, the inherent delay in waiting for transcriptions could potentially undermine usability. Future research might explore open-source projects that have integrated real-time capabilities with Whisper to reduce latency for enhanced performance.

Reference

1. R. Chopra, “Artificial Intelligence in Robotics: (Review Paper),” *International Journal for Research in Applied Science and Engineering Technology*, vol. 11, no. 4, pp. 2345–2349, Apr. 2023, doi: 10.22214/ijraset.2023.50635.
2. S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, “Robots that use language,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 25–55, May 2020, doi: 10.1146/annurev-control-101119-071628.
3. Z. Ma, J. Pan, and J. Chai, “World-to-Words: Grounded Open Vocabulary Acquisition through Fast Mapping in Vision-Language Models,” *arXiv (Cornell University)*, Jun. 2023, doi: 10.48550/arxiv.2306.08685.
4. B. Chen *et al.*, “Open-vocabulary queryable scene representations for real world planning,” *arXiv (Cornell University)*, Sep. 2022, doi: 10.48550/arxiv.2209.09874.
5. W. Yu *et al.*, “Language to rewards for robotic skill synthesis,” *arXiv (Cornell University)*, Jun. 2023, doi: 10.48550/arxiv.2306.08647.
6. C. Matuszek, “Grounded Language Learning: Where Robotics and NLP Meet,” *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, Jul. 2018, doi: 10.24963/ijcai.2018/810.
7. D. Misra, J. Langford, and Y. Artzi, “Mapping Instructions and Visual Observations to Actions with Reinforcement Learning,” *arXiv (Cornell University)*, Apr. 2017, doi: 10.48550/arxiv.1704.08795.

8. R. Paul, A. Barbu, S. Felshin, B. Katz, and N. Roy, “Temporal Grounding Graphs for Language Understanding with Accrued Visual-Linguistic Context,” *arXiv (Cornell University)*, Nov. 2018, doi: 10.48550/arxiv.1811.06966.
9. P. Srivatsavaya, “LSTM — Implementation, Advantages and Diadvantages - Prudhviraju Srivatsavaya - Medium,” *Medium*, Oct. 05, 2023. [Online]. Available:https://medium.com/@prudhviraju.srivatsavaya/lstm-implementation-advantages-and-diadvantages-914a96fa0acb
10. R. Józefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, “Exploring the limits of language modeling,” *arXiv (Cornell University)*, Feb. 2016, [Online]. Available: http://export.arxiv.org/pdf/1602.02410
11. M. J. Ahn *et al.*, “Do as I can, not as I say: grounding language in robotic affordances,” *arXiv (Cornell University)*, Apr. 2022, doi: 10.48550/arxiv.2204.01691.
12. E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, pages 610–623, 2021.
13. R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258, 2021.
14. Z. Yue *et al.*, “Robot Task Planning Based on Large Language Model Representing Knowledge with Directed Graph Structures,” *arXiv (Cornell University)*, Jun. 2023, doi: 10.48550/arxiv.2306.05171.

15. A. Bucker *et al.*, “LATTE: LAnguage Trajectory TransformER,” *arXiv.org*, Aug. 04, 2022. https://arxiv.org/abs/2208.02918
16. Z. Li, G. Z. Xie, V. Arya, and K. T. Chui, “Semantic Trajectory Planning for industrial Robotics,” *International Journal on Semantic Web and Information Systems*, vol. 20, no. 1, pp. 1–10, Dec. 2023, doi: 10.4018/ijswis.334556.
17. J. L. Yow, N. Garg, M. Ramanathan, and W. T. Ang, “ExTraCT -- Explainable Trajectory Corrections from language inputs using Textual description of features,” *arXiv (Cornell University)*, Jan. 2024, doi: 10.48550/arxiv.2401.03701.
18. A. Vanzo, D. Croce, E. Bastianelli, R. Basili, and D. Nardi, “Grounded language interpretation of robotic commA. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust speech recognition via Large-Scale Weak Supervision,” *arXiv (Cornell University)*, Dec. 2022, doi: 10.48550/arxiv.2212.04356.
19. A. Vanzo, D. Croce, E. Bastianelli, R. Basili, and D. Nardi, “Grounded language interpretation of robotic commands through structured learning,” *Artificial Intelligence (General Ed.)*, vol. 278, p. 103181, Jan. 2020, doi: 10.1016/j.artint.2019.103181.
20. Z. Hu, J. Pan, T. Fan, R. Yang, and D. Manocha, “Safe Navigation with Human Instructions in Complex Scenes,” *arXiv (Cornell University)*, Sep. 2018, doi: 10.48550/arxiv.1809.04280.
21. Y. Qiu and Y. Jin, “ChatGPT and Finetuned BERT: A Comparative Study for developing Intelligent Design Support Systems,” *Intelligent Systems With Applications*, vol. 21, p. 200308, Mar. 2024, doi: 10.1016/j.iswa.2023.200308.

22. R. Martínez-Cruz, A. J. López-López, and J. L. C. Portela, “ChatGPT vs State-of-the-Art Models: A Benchmarking Study in Keyphrase Generation Task,” *arXiv (Cornell University)*, Apr. 2023, doi: 10.48550/arxiv.2304.14177.
23. R. Peeters and C. Bizer, “Using ChatGPT for entity matching,” *arXiv (Cornell University)*, May 2023, doi: 10.48550/arxiv.2305.03423.
24. “Noisy speech training in MFCC-based speech recognition with noise suppression toward robot assisted autism therapy,” *IEEE Conference Publication | IEEE Xplore*, Aug. 01, 2017.
25. W. Chan, D. Park, C. A. Lee, Y. Zhang, Q. V. Le, and M. Norouzi, “SpeechStew: Simply mix all available speech recognition data to train one large neural network,” *arXiv (Cornell University)*, Apr. 2021, [Online]. Available: https://arxiv.org/pdf/2104.02133.pdf
26. Y. Zhang *et al.*, “BigSSL: Exploring the frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, Oct. 2022, doi: 10.1109/jstsp.2022.3182537.
27. D. Mahajan *et al.*, “Exploring the limits of weakly supervised pretraining,” in *Lecture Notes in Computer Science*, 2018, pp. 185–201. doi: 10.1007/978-3-030-01216-8_12.
28. J. Le, “Divide and Conquer Algorithms - Data notes,” *Medium*, May 10, 2020. [Online]. Available: https://data-notes.co/divide-and-conquer-algorithms-b135681d08fc

Appendix

APPENDIX A: Split Prompt

```
split_prompt = """
Analyze the user's command. If the command contains multiple
instructions, break it down into simpler sentences and separate them
with a comma without any newline.
Otherwise return the same input. Separate each sentence with a comma
without any newline.

ADDITIONALLY:
check if the input is DISTANCE or SPEED related.

FORMAT JSON:
    {
        "split": [__, ...],
        "type" : ["DISTANCE" or "SPEED", ...]
    }
"""
```

APPENDIX B: Object extraction prompt

```
dynamic_feature_prompt = """
Append physical/tangible object found in the input into a list. Return
empty list if no object found.
Separate each object with a comma.

Ensure object is in singular form. (e.g. "bottles" -> "bottle")
EXCEPTION (scissors -> scissors | pants -> pants | glasses -> glasses)

FORMAT JSON:
    {
        "obj_list": [__, ...]
    }
"""
```

APPENDIX C: Feature classification prompt

```
enhanced_inference_prompt = """
Return the closest semantic feature from the provided DYNAMIC FEATURES
list that matches each command in a given list.
If a command indicates approaching an object, the object's distance
decreases.
If a command implies avoiding an object or turning away from it, the
object's distance increases.
If a command involves changes in speed, the appropriate 'speed
increase' or 'speed decrease' feature should be issued.
Commands that include directional movement WITHOUT a specified object
should be matched with the corresponding Cartesian axis feature.
Compound commands require outputting multiple features.

The output array must have a length equal to the number of commands in
the input. Compound commands should have their respective features
concatenated in the output array, separated by a comma within ONE
STRING.
EXAMPLE: (speed after distance)
input: "Go fast when close to bottle"
out: "bottle distance decrease, speed increase"

For compound commands, assess and include both the speed alteration
and (cartesian axis change or object distance change).

REFERENCE AXES (when commands does not involve object):
- Key: left , Feature: X-cartesian decrease
- Key: right , Feature: X-cartesian increase
- Key: up , Feature: Z-cartesian increase
- Key: down , Feature: Z-cartesian decrease
- Key: forward , Feature: Y-cartesian increase
- Key: backward , Feature: Y-cartesian decrease

ADDITIONALLY:
- return the confidence score for each prediction.

JSON output should follow this format:
{
    "output" : ["___", ...],
```

```
    "confidence": [___, ...]
}
"""
```

APPENDIX D: Intensity prompt

```
intensity_prompt = """
Categorize each command given into one of the following intensities.
Ensure each command is matched with an intensity.

1. HIGH  - when the input has high intensity description on the
instructions like "very", "a lot", "greater" or anything that
signifies MORE.

2. LOW  - is the opposite, low intensity description on instructions
like "a little bit", "not by much", "lesser", something that signifies
LESS

3. NEUTRAL - is when the input does not have any of the above
description in its input.

Return only the answer.

FORMAT JSON:
    {
        "intensity" : [__, __, __, ...],
    }
"""
```

APPENDIX E: Whisper Correction Closest Semantic Prompt

```
Replace the noun object with item from gt_list with the closest
semantic meaning as `rectified_sentence`.
If no object is matched, return the original sentence as
`rectified_sentence`.
If the noun object is in gt_list, return the original sentence as
`rectified_sentence`.

rectified_sentence must be a command.

ground_truth_object = {vision_list}


JSON FORMAT:
{{
"wrong_obj" : "____",
"correct_obj" : "____",
"rectified_sentence" : "____",
}}
```

APPENDIX F: Whisper Correction Similar Sound Prompt

```
Rectify and return the user message by replacing the noun object with
the item from gt_list with the closest sound as `rectified_sentence`.
If no object is matched, return the original sentence as
`rectified_sentence`.

rectified_sentence must be a command.

ground_truth_object = {vision_list}

JSON FORMAT:
{{
"wrong_obj" : "____",
"correct_obj" : "____",
"rectified_sentence" : "____",
}}
```

APPENDIX G: 150 commands dataset


```
{
   "1": "Get close to bottle and turn right then move closer to apple and stay
away from book",
   "2": "Approach the vase and move left then get nearer to the orange and stay
down",
   "3": "Reduce the distance to the box and turn left then get a bit closer to
the banana but avoid the pen",
   "4": "Move closer to the cup and go right then keep close to the pear but step
back from the notebook",
   "5": "Stay close to the jar and move leftward then decrease the distance to
the watermelon but avoid the magazine",
   "6": "Get nearer to the bottle and approach the monitor then get a little bit
lower and stay down",
   "7": "Retreat from the table and move rightwards then move down and keep your
distance from the textbook",
   "8": "Get closer to the plate and go forward then move closer to the peach but
steer clear of the journal",
   "9": "Retreat from the glass and stay left then increase the distance from the
mango and move a bit further away from the diary",
   "10": "Keep a greater distance from the bowl and move right then move up and
avoid the notebook",
   "11": "Traverse right and get closer to the desktop then stay elevated and go
backwards",
   "12": "Move nearer to the book and go rightward then go down and stay close to
the book",
   "13": "Get a bit closer to the can and move leftward then descend downward and
avoid the magazine",
   "14": "Stay near the cup and move rightwards then move lower and increase the
distance from the notebook",
   "15": "Get closer to the plate and approach the laptop then remain at a lower
position and keep your distance from the journal",
   "16": "Move away from the glass and stay left then go up and move a bit
further away from the diary",
   "17": "Keep a greater distance from the bowl and move rightwards then ascend
upwards and avoid the notebook",
   "18": "Move nearer to the jar and go rightward then stay elevated and avoid
the textbook",
   "19": "Move closer to the plate and get close to the desktop then move lower
and steer clear of the magazine",
   "20": "Move forward and move rightwards then decrease the distance to the
notebook and move a bit further away from the diary",
```

```
  "21": "move more left and stay closer to the bottle",
  "22": "stay closer to the laptop and cup, move up a little bit",
  "23": "move down and closer to the plate and microwave",
  "24": "avoid the cake and the fridge",
  "25": "move away from bottle and book then move closer to cup",
  "26": "move away from bottle and then move closer to cup",
  "27": "move closer to egg and cake",
  "28": "move closer to the bottle and move further from the laptop",
  "29": "move up",
  "30": "move right",
  "31": "keep close to pineapple",
  "32": "move near the bottle",
  "33": "move near the pineapple",
  "34": "move further from the cake",
  "35": "move closer to the fridge",
  "36": "keep right and stay closer to the bottle",
  "37": "stay closer to the vase and cup, move up a little bit",
  "38": "move lower and closer to the banana and apple",
  "39": "stay extremely far away from the bottle and cup",
  "40": "stay away from laptop",
  "41": "stay further away from the pineapple",
  "42": "stay closer to spoon",
  "43": "move upwards",
  "44": "move further from laptop",
  "45": "move closer to pineapple",
  "46": "move nearer to spoon",
  "47": "move lower",
  "48": "avoid cake",
  "49": "move away from bottle and then closer to cup",
  "50": "move closer to spoon",
  "51": "move closer to microwave",
  "52": "retreat from the bottle",
  "53": "move towards bottle",
  "54": "move away from laptop",
  "55": "move away from pineapple",
  "56": "move towards spoon",
  "57": "move towards cake",
  "58": "move towards egg",
  "59": "go close to cup",
  "60": "move down",
  "61": "move to pineapple",
  "62": "move to pineapple, move down",
```

```
  "63": "avoid the fridge",
  "64": "closer to cake",
  "65": "move closer to bottle",
  "66": "avoid the cake",
  "67": "move closer to egg",
  "68": "move towards the bottle",
  "69": "move closer to the spoon",
  "70": "move away from the microwave",
  "71": "move towards the egg",
  "72": "left",
  "73": "up",
  "74": "move to the cake",
  "75": "avoid the egg",
  "76": "move closer to the cup",
  "77": "move closer to the pineapple",
  "78": "move closer to the egg",
  "79": "move higher",
  "80": "move closer to the bottle",
  "81": "move away from the pineapple",
  "82": "move away from the cake",
  "83": "closer to the bottle and further from the laptop",
  "84": "keep close to the table but stay clear from cup",
  "85": "closer to the cake",
  "86": "move away from the laptop but closer to the cup",
  "87": "move to the right",
  "88": "move towards the spoon",
  "89": "Get closer to the can",
  "90": "Traverse right",
  "91": " Move forward and then move right",
  "92": "Move backward and then move up",
  "93": "Approach the cup but avoid the scissors and keep left",
  "94": "Go close to the banana but keep a distance from apple",
  "95": "Move towards the cup and keep left",
  "96": "Move a little bit forward",
  "97": "Stay nearer to cup but keep a distance from the scissors",
  "98": "Stay far from the book and keep right while moving forward",
  "99": "Move towards the book and keep back from the banana",
  "100": "Take a step forward",
  "101": "move up and closer to the plate and microwave",
  "102": "Move near the cup and microwave",
  "103": "move up and further away from the bowl and bottle",
  "104": "stay closer to the bottle and apple, move up a little bit",
```

```
  "105": "Get close to the bottle and notebook",
  "106": "Get close to the orange and the notebook",
  "107": "Get away from bottle and microwave",
  "108": "Get near to cup and the microwave",
  "109": "Turn right and move closer to the bottle and apple",
  "110": "Move up and nearer to the peach and monitor",
  "111": "Move left and nearer to the peach and the monitor",
  "112": "Stay away from bottle and box and fridge",
  "113": "Stay away from vase and banana and microwave",
  "114": "Stay close to spoon and microwave and cup",
  "115": "Stay near to the bottle and the microwave and the cup",
  "116": "move closer to jar and cake",
  "117": "move closer to jar and microwave",
  "118": "move nearer to jar and the microwave",
  "119": "move away from vase and apple",
  "120": "move away from the vase and the apple",
  "121": "Turn right and move closer to the vase and the apple",
  "122": "Turn left and move closer to the vase and plate",
  "123": "Approach bottle and cup",
  "124": "Approach spoon and the cup",
  "125": "Approach vase and microwave",
  "126": "Approach the vase and microwave",
  "127": "Approach the glass and the microwave",
  "128": "Keep clear of the vase and the apple",
  "129": "Keep clear of jar and the apple",
  "130": "Avoid apple and microwave",
  "131": "Avoid the orange and the microwave",
  "132": "Move up and avoid the orange and the microwave",
  "133": "Move up and avoid pineapple and microwave",
  "134": "Avoid can, apple and plate",
  "135": "Avoid the can, the apple and the plate",
  "136": "Go up and avoid the vase and the glass",
  "137": "Approach bottle and avoid microwave",
  "138": "Approach the bottle and avoid the microwave",
  "139": "Approach the bottle and avoid the microwave and the cup",
  "140": "Go to the apple and bowl",
  "141": "Go to the pear and the bowl",
  "142": "Go near the orange and bowl and avoid the bottle and microwave",
  "143": "Go near the book and the bowl and avoid the bottle and the microwave",
  "144": "Move up and closer to the vase and the glass",
  "145": "Move up and closer to the bottle and glass",
  "146": "Move back and closer to the vase and microwave",
```

```
    "147": "Move back and closer to the cup and the microwave",
    "148": "Get away from microwave and bowl",
    "149": "Get away from the microwave and the bowl",
    "150": "Go close to bottle and pear"
}
```

APPENDIX G: Object list

```
{
   "objects": [
       "bottle",
       "orange",
       "apple",
       "book",
       "pear",
       "notebook",
       "cup",
       "box",
       "jar",
       "watermelon",
       "magazine",
       "banana",
       "can",
       "pen",
       "monitor",
       "textbook",
       "peach",
       "desktop",
       "bowl",
       "microwave",
       "laptop",
       "plate",
       "glass",
       "diary",
       "cake",
       "fridge",
       "egg",
       "pineapple",
       "spoon",
       "scissors",
       "vase"
   ]
}
```

APPENDIX H: Feature Templates & Textual Descriptions

FT	obj_distance_decrease	obj_distance_increase
TDT	Move closer to {obj} **Stay close to {obj}** **Decrease distance to {obj}** Keep a smaller distance from {obj}	Move further away from {obj} **Stay away from {obj}** **Increase distance to {obj}** Keep a bigger distance from {obj} Avoid {obj}

FT	z_cart_decrease	z_cart_increase
TDT	Move closer to table **Stay closer to table** Move lower Move down **Stay down** Go to the bottom Down Low	Move further away from table **Stay away from table** Move higher Move up **Stay up** Stay on the upper part Up Top Go to the top

FT	y_cart_decrease	y_cart_increase
TDT	**Stay on the left** **Go to the left** Move left Move more towards the left Left	**Stay on the right** **Go to the right** Move right Move more towards the right Right

FT	x_cart_decrease	x_cart_increase
TDT	**Stay at the back** **Go to the back** Move back Stay back Move backward Go behind	**Stay at the front** **Go to the front** Move front Stay front Move forward

APPENDIX I: Whisper Transcription Correction Experiment Data set

Test No	Commands
1	Get close to bottle but keep slightly right
2	Move a bit closer to bowl but stay away from flask
3	Stay close to the mug and stay low
4	Avoid the banana, keep right
5	Move close to the flower and move slower when close to vase
6	Go closer to tumbler and cup
7	Move front but keep a distance to the scissors
8	Move fast when closer to the bus
9	Get extremely close to the bottle
10	Move slowly when close to the bottle and stay away from orange and banana
Objects list	**["scissors", "mug", "vase", "bottle", "bowl", "orange", "laptop", "banana", "flower"]**