

Article

Comparative Analysis of Generic and Fine-Tuned Large Language Models for Conversational Agent Systems

Laura Villa , David Carneros-Prado , Cosmin C. Dobrescu , Adrián Sánchez-Miguel , Guillermo Cubero 
and Ramón Hervás * 

Department of Technologies and Information Systems, University of Castilla-La Mancha, 13071 Ciudad Real, Spain; laura.villa@uclm.es (L.V.); david.carneros@uclm.es (D.C.-P.); cosmin.dobrescu@uclm.es (C.C.D.); adrian.sortega@uclm.es (A.S.-M.); guillermo.cubero@uclm.es (G.C.)

* Correspondence: ramon.hlucas@uclm.es

Abstract: In the rapidly evolving domain of conversational agents, the integration of Large Language Models (LLMs) into Chatbot Development Platforms (CDPs) is a significant innovation. This study compares the efficacy of employing generic and fine-tuned GPT-3.5-turbo models for designing dialog flows, focusing on the intent and entity recognition crucial for dynamic conversational interactions. Two distinct approaches are introduced: a generic GPT-based system (G-GPT) leveraging the pre-trained model with complex prompts for intent and entity detection, and a fine-tuned GPT-based system (FT-GPT) employing customized models for enhanced specificity and efficiency. The evaluation encompassed the systems' ability to accurately classify intents and recognize named entities, contrasting their adaptability, operational efficiency, and customization capabilities. The results revealed that, while the G-GPT system offers ease of deployment and versatility across various contexts, the FT-GPT system demonstrates superior precision, efficiency, and customization, although it requires initial training and dataset preparation. This research highlights the versatility of LLMs in enriching conversational features for talking assistants, from social robots to interactive chatbots. By tailoring these advanced models, the fluidity and responsiveness of conversational agents can be enhanced, making them more adaptable and effective in a variety of settings, from customer service to interactive learning environments.



Citation: Villa, L.; Carneros-Prado, D.; Dobrescu, C.C.; Sánchez-Miguel, A.; Cubero, G.; Hervás, R. Comparative Analysis of Generic and Fine-Tuned Large Language Models for Conversational Agent Systems. *Robotics* **2024**, *13*, 68. <https://doi.org/10.3390/robotics13050068>

Academic Editors: Soyeon Caren Han and Naira Hovakimyan

Received: 27 March 2024

Revised: 23 April 2024

Accepted: 26 April 2024

Published: 29 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: large language model; dialog model design; conversational agents; chatbot; chatbot development platform; intent classification; entity recognition

1. Introduction

In this modern era marked by significant strides in artificial intelligence, the emergence of virtual assistants, chatbots, and conversational agents has fundamentally altered the engagement with digital platforms. These technologies, which are characterized by their ability to deliver immediate and personalized responses, have greatly enhanced user interactions. The development of such systems typically falls into two distinct categories: rule-based systems and those that harness the power of artificial intelligence, particularly through machine learning algorithms [1].

Among the variety of tools available for crafting these conversational interfaces, third-party Chatbot Development Platforms (CDPs) are particularly noteworthy. These platforms skillfully blend machine learning techniques, including intent and entity recognition, with rule-based frameworks to facilitate sophisticated Natural Language Understanding (NLU). Despite their utility, reliance on third-party solutions often introduces constraints that hinder flexibility and autonomy in chatbot development [2].

1.1. Background

In the realm of conversational AI solutions, prominent CDPs, such as DialogFlow (<https://cloud.google.com/dialogflow>. Last accessed: 26 March 2024) (Google), IBM

Watson Assistant (<https://cloud.ibm.com/catalog/services/watson-assistant>. Last accessed: 26 March 2024), Microsoft Bot Framework (<https://www.botframework.com/>. Last accessed: 26 March 2024), and Amazon Lex (<https://aws.amazon.com/es/lex/>. Last accessed: 26 March 2024) stand out as potent NLU platforms. These platforms, being third-party systems, provide user-friendly interfaces for the creation, design, and deployment of chatbots, conversational agents, and virtual assistants. They are acclaimed for their comprehensive features, facilitating the construction of interactive and efficient virtual assistants applicable in various sectors, including education [3], healthcare [4,5], early disease detection [6], and addressing recommendations, support queries, and other services [7,8]. Their adaptability and integration capabilities render them favorable options for entities aiming to develop conversational interfaces across diverse channels, industries, and fields. Similar open-source platforms are available for this purpose, for example Rasa [9], which has been used in areas such as e-healthcare support [10,11] and workflow management systems [12].

As the fundamental foundation of social assistants, from chatbots to interactive avatars and social robots, the ability to carry on conversations and verbal communication plays a key role. This capability not only improves accessibility and usability, but also enriches the user experience, enabling more natural and human interactions. In particular, the integration of advanced language models and third-party CDPs has proven to be fundamental to equip these assistants with communicative skills for social assistance. For example, chatbots integrate these kinds of platforms for providing conversational support in health care issues [6,11], or making clothes recommendations [7]. Social robots also incorporate these conversational skills, especially in home health assistance and companionship [5,13]. Other types of talking assistants, such as voice and virtual assistants, use their dialog capabilities to help users with common, everyday tasks, such as preparing for exams [3] or paying in online stores [8]. In all these contexts, the assistants social component is critical to perform their tasks and interact with their users.

1.2. Motivation

In this regard, third-party CDPs provide a framework for creating conversational experiences. These platforms facilitate the rapid deployment, customization and integration of conversational skills into social assistants. However, despite their extended use and their advantages, these platforms have limitations such as dependency on specific intent recognition providers, leading to potential vendor lock-in situations, especially concerning the natural language engine and its training constraints. Additionally, these platforms often lack the necessary abstraction mechanisms for seamless integration with external platforms that an organization might need to engage with [14]. They are designed to provide non-expert, user-oriented tools for rapid chatbot deployment, which might render them rigid (<https://www.chatbots.org/dialogflow>. Last accessed: 15 September 2023) and inadequate for expert developers.

Parallel to CDPs, pre-trained Language Models (PLMs) (based on transformer architectures) have proven effective in natural language processing (NLP) tasks. In particular, groundbreaking advancements in the field of AI have emerged in recent years with the introduction of Large Language Models (LLMs). LLMs are large PLMs that demonstrate powerful capabilities owing to their scale-up [15]. They represent a significant leap forward, with models such as the Generative pre-trained Transformer (GPT) developed by OpenAI [16], showcasing exceptional capabilities in text generation, translation, document summarization, etc. Their ability to understand and produce human language by processing a vast corpus of text makes them invaluable in various NLP applications [17,18]. Given their proficiency in NLP tasks, LLMs such as GPT could be instrumental in crafting chatbot dialog flows, highlighting their potential as powerful tools in the domain of conversational AI.

In response to third-party CDPs limitations, and taking advantage of LLMs, this study explores the potential of leveraging LLMs, specifically GPT-3.5-turbo, as an alternative

foundation for CDPs. GPT-3.5-turbo, with its advanced natural language processing (NLP) capabilities, offers a promising avenue for designing conversational flows that are both dynamic and adaptable. The goal is to create a development platform that not only meets the diverse linguistic requirements of various applications, but also empowers developers with greater control over the system's configuration. This approach seeks to deviate from the opaque black-box nature of existing third-party CDPs, helping to achieve a more transparent and customizable solution.

Through a detailed exploration of the use and effectiveness of GPT-3.5-turbo in dialog flow design, this study aims to showcase the potential of LLMs in revolutionizing the landscape of conversational AI. By harnessing the power of GPT-3.5-turbo, it is aimed to develop a CDP that not only enhances the flexibility and adaptability of chatbot development but also offers a more developer-centric approach, ultimately contributing to the advancement of conversational technology. To further enrich this exploration, two distinct system bases are proposed for this platform: one utilizing fine-tuning of GPT models and the other employing the pre-trained model without modification. The fine-tuning approach involves customizing the GPT model on specific datasets, allowing it to adapt to particular conversational nuances and domains, thereby enhancing its relevance and effectiveness in targeted scenarios. This method offers the advantage of tailoring the model's responses to better align them with the desired conversational tone and content, leading to more accurate and contextually appropriate interactions. However, utilizing the pre-trained GPT model without modification leverages extensive and diverse training of the model across a wide range of topics and styles. This approach offers broad applicability and generalization capability, making it suitable for a wide array of conversational contexts without the need for additional training. This study aims to examine the comparative strengths and potential applications of both methodologies, providing insights into their practical implications in the field of conversational AI.

1.3. Related Work

To the best of our knowledge, prior research has not explored the integration of LLMs for specific functionalities within CDPs. While most existing studies focus on LLMs powering entire chatbot systems (where the conversational agent is entirely comprised of an LLM like ChatGPT, as in [19]), there is minimal exploration into their integration with other conversational components. However, there is some emerging research that seeks to leverage both rule-based systems and LLMs, which attempts to take advantage of the potentials of both rule-based intelligent chatbots and LLMs to create a kind of hybrid conversational system [20].

In terms of comparing the performance of generic versus fine-tuned LLMs within conversational systems, the literature shows some analysis in other domains. For instance, studies have comparisons of generic and fine-tuned BERT [21] models for tasks like analyzing humanitarian data in an ethically conscious manner [22] and predicting stock prices [23]. There are also discussions on the methodologies of fine-tuning LLMs, comparing various approaches and their outcomes against other fine-tuned LLMs [24]. However, there is a lack of direct comparisons between generic pre-trained models and their fine-tuned versions in the context of NLP tasks for creating domain-specific conversational agents. This absence suggests a significant opportunity for investigation into how these two approaches could be tailored to enhance conversational AI.

1.4. Organization

The document is organized as follows: Section 2 introduces the three primary stages of the system and the methodologies applied. In Section 3, the findings are displayed and their importance with respect to the objective is discussed. Section 4 provides a comprehensive interpretation of these results and their wider implications. The document concludes with Section 5, which summarizes the research objectives and significant contributions.

2. Materials and Methods

Most of the most commonly used CDPs includes three basic components for crafting dialog flow:

- **Intents** : These are the underlying objectives behind a user’s query or message, essentially driving the assistant’s responses and steering the direction of the conversation.
- **Entities**: These refer to specific concepts, objects, or pieces of information relevant to the model derived from the user input. Recognizing entities allows for tailored or more nuanced services.
- **Dialog Nodes**: Often represented as graphs or trees, these nodes shape the route, progression, or framework of the conversation. They are usually represented by conditional statements that include intents and entities, and, if met, the response contained in the node is returned.

Together, these elements form the backbone of dialog flows, empowering developers to craft complex conversational sequences by linking user inputs to corresponding outputs. CDPs leverage Natural Language Processing (NLP) and machine learning to discern user intents, extract entities, and navigate through dialog nodes to formulate appropriate responses (Figure 1).

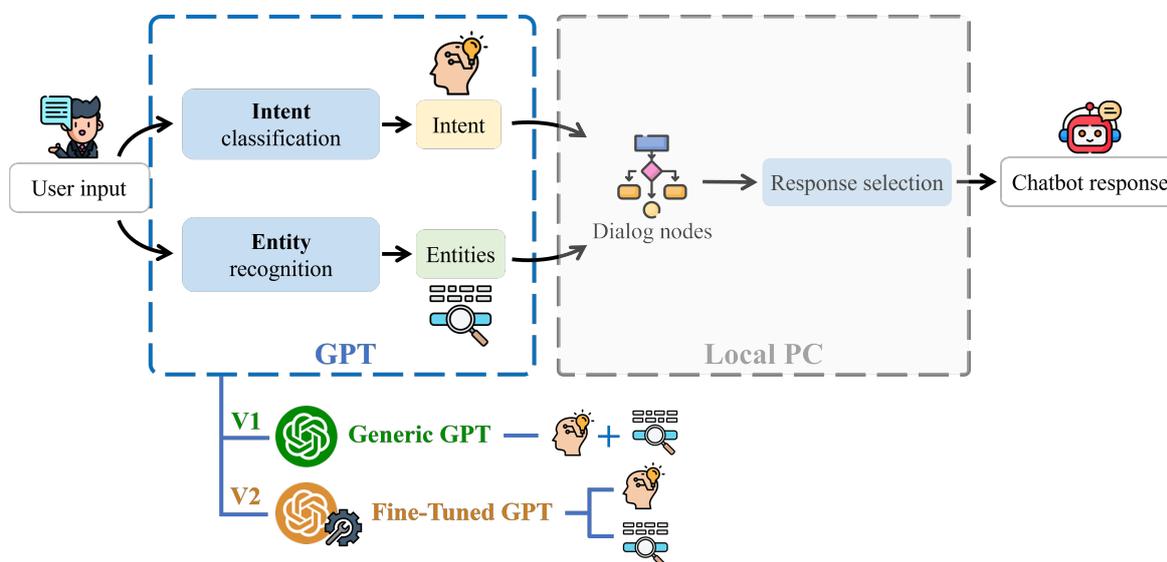


Figure 1. Process of response generation in a Chatbot Development Platform. In the system, the objective is to perform intent classification and entity recognition via GPT models. It differentiates between two approaches, V1 utilizing a generic pre-trained GPT and V2 employing two fine-tuned GPT models. Both versions are compared and discussed in this paper.

Thus, to design a chatbot development platform, three main processes are required: intent detection, entity extraction, traversal, and evaluation of dialog node conditions to generate a response.

Several solutions arise to address this problem by using LLMs, the most straightforward of which is to utilize the substantial context power of GPT models to store the conversational tree and let the LLM itself perform intention detection, entity extraction, dialog node traversal. However, this is not a viable solution, given that conversational trees in practice can have a substantial size; the context of GPT models would quickly “lose memory”, forgetting nodes when its context capacity becomes full. Furthermore, regardless of the number of nodes, this would inevitably occur, as user requests also begin to become part of the context. On the other hand, the base GPT-3.5-turbo models have an associated cost and a maximum number of tokens in the requests; therefore, it is not advisable to perform inference and requests with too much context, both because of the time and cost involved in each query.

In fact, considering the flow described in Figure 1, the most complex natural language processing tasks in this type of system are intent detection and entity extraction. The remainder is essentially a graph traversal in which certain conditions are evaluated (based on the detected intent and entities), and a predefined response from the dialog tree is returned. Therefore, the decision was made to dedicate the use of the LLM to the intent and entity detection processes, whereas the hosting of the conversational tree and its traversal were performed locally on the machine. Thus, the context and number of tokens in requests for the GPT models are significantly reduced.

In the forthcoming sections of this paper, the approaches of intent detection and entity extraction are compared through two distinct methodologies: employing pre-trained generic GPT models as they are, and fine-tuning GPT models for specific applications (as shown in Figure 1). For the design of both systems (especially for the list of intentions and entities to detect, as well as the preparation of the datasets) it is assumed as an example that the chatbot system will be for a Spanish store, so the intentions and entities to extract will be related to that subject (appointments, calendars, reservations, events, etc.).

Initially, the utilization of pre-trained generic GPT models without any modifications is explored. This approach leverages the broad and diverse training of these models, enabling them to perform intent detection and entity extraction with a high degree of versatility. Following this, the fine-tuning process is examined, in which GPT models are specifically adapted to cater to particular datasets.

2.1. Generic GPT Based System (G-GPT)

Pre-trained GPT models are undeniably powerful natural language processing tools, offering plug-and-play functionality that enables seamless integration into various applications. These generic models can be leveraged directly with an accurate prompt, thereby obviating the need for additional training. However, the efficacy of these models relies heavily on the precision and quality of the prompts used to query them. A well-constructed prompt not only ensures accurate responses but also enhances the model's ability to discern user intent and extract relevant entities effectively.

2.1.1. Prompt Engineering

Crafting a precise prompt for a pre-trained GPT model entails several key aspects to ensure its effectiveness in fulfilling the intended tasks of intent classification and entity extraction.

- **System Context:** Begin describes the purpose of the model within the broader system context, making it clear that the model functions as both an intent classifier and an entity extractor.
 - **Intent Classification:** Enumerate all possible intents that the model should classify, ensuring that it outputs exactly one intent.
 - **Entity Extraction:** Specify the entities that need to be extracted. Additionally, details of any format aspect, such as date representation (modified ISO 8601 format [25] with the month represented by three letters for enhanced comprehension and disambiguation).
- **Date and time stamp:** This includes a date and time stamp within the prompt to provide a temporal context for dynamic entity extraction. For instance, the model should understand references like “tomorrow” relative to the current date and time.
- **Desired Output Format:** Define the desired output format, preferably structured as a JSON object containing the extracted intents and entities. This format choice aimed to facilitate the subsequent analysis of the model's outputs.
- **Example of Correct System Functioning:** Provide an illustrative example within the prompt to demonstrate the expected behavior of the model. This example serves as a reference point for the model to understand its tasks and desired output.

Leveraging the innate language-understanding capabilities of GPT models, these aspects need not be exhaustively explained, allowing for concise yet comprehensive prompts tailored to the model's requirements.

Considering these aspects, the final prompt is shown in Listing 1 (552 tokens (<https://platform.openai.com/tokenizer>. Last accessed: 26 March 2024)).

Listing 1. Prompt for the Generic GPT system.

```

1 Given the user input, accurately extract the single most relevant intent from the following
  options: Yes, No, Thanks, Make_Appointment, Know_Operation_Hours, Know_Location, Help,
  Greetings, Goodbye, Connect_to_agent, or select Irrelevant if none of the intents apply.
  In addition to determining the intent, it is crucial to identify any and all entities
  present within the query with utmost precision. The entities to look for include 'sysdate'
  in the format YYYY-Mon-DD (where 'Mon' is abbreviated as Jan, Feb, etc.), 'systime' in
  the format hh:mm, and 'holiday', which refers to the names of holidays. The expected
  output format is:
2
3 {
4   "intent": "Intent_Here",
5   "entities": {
6     "systime": ["Time1", "Time2", ...], // Include mentioned times
7     "sysdate": ["Date1", "Date2", ...], // Include mentioned dates
8     "holiday": ["Holiday1", "Holiday2", ...] // Include mentioned holidays
9   }
10 }
11
12 Each entity type (systime, sysdate, holiday) can have none, one, or multiple instances. It is
  essential to capture every instance mentioned in the input. For example, the input [now:
  2024-Feb-11;11:35] with the query 'are you open tomorrow at 15:00 like on halloween?'
  should yield the following correct output:
13
14 {
15   "intent": "Know_Operation_Hours",
16   "entities": {
17     "sysdate": ["2024-Feb-12"],
18     "systime": ["15:00"],
19     "holiday": ["Halloween"]
20   }
21 }
22
23 Use the 'now' timestamp provided for context in interpreting time-relative terms like '
  tomorrow', 'yesterday', etc. This reference helps in accurately converting relative dates
  and times to their absolute counterparts. The current date and time for reference are [
  now:2024-Mar-20;09:59].
24
25 Pay special attention to correctly identifying and formatting all instances of dates, times,
  and holidays mentioned in the query, as missing any detail will result in incomplete or
  incorrect entity extraction. Ensure that every entity is captured in the provided output
  format, adhering to the specified types and formats for systime, sysdate, and holiday.
26
27 For example, for the input "December 31st is New Year's Eve, we will toast to the new year at
  00:01 am", the correct output should include the sysdate '2024-Dec-31', the holiday 'New
  Year's Eve', and the systime '00:01', reflecting the comprehensive identification of all
  entities mentioned:

```

The GPT queries use a temperature value of 0. Temperature in natural language processing controls the randomness of generated text. A temperature of 0 makes the model deterministic, always producing the same output for a given input, ensuring consistent results.

2.1.2. Data Preparation

Once the prompt is established, the next step involves creating a dataset to validate the performance of the model. Two datasets were created to validate the functionalities of both intent classification and entity extraction within the model (these datasets also serve later as resources for validating fine-tuned models for the Fine-Tuned GPT-based system, allowing for a comparison between the two systems). The intent dataset comprises 212 entries, each containing various user queries paired with their corresponding intent labels (Table 1). The entity dataset consists of 50 entries containing user queries along with their date and time stamps, as well as the extracted entities (Table 2). Both datasets are structured in JSONL format, a choice made to facilitate their reuse in the fine-tuning process of the FT-GPT, as outlined in Section 2.2.

Table 1. Intent dataset excerpt.

User Query	Intent
of course	Yes
of course not	No
thanks for everything	Thanks
farewell	Goodbye
can I speak to an agent online?	Connect_to_agent
I love Star Wars films	Irrelevant

Table 2. Entities dataset excerpt.

Datetime Stamp	User Query	Entities
2023-Dec-23; 16:30	we meet at 3 a.m. on christmas	{‘holiday’: [‘christmas’], ‘systime’: [‘03:00’]}
2024-Feb-29; 12:00	tomorrow is Friday	{‘sysdate’: [‘2024-Mar-01’]}
2024-Jan-26; 16:00	on Australia Day we go on vacation	{‘holiday’: [‘day_australia’]}
2019-Apr-07; 08:00	the party is on the 9th at 3:35 a.m. in the park	{‘sysdate’: [‘2019-Apr-09’], ‘systime’: [‘03:35’]}
2028-Feb-14; 07:00	in 2030, on March 7, we will be in a flying car like the ones in Back to the Future	{‘sysdate’: [‘2030-Mar-07’]}

2.2. Fine-Tuned GPT Based System (FT-GPT)

In this system version, fine-tuning and few-shot learning are applied. Fine-tuning is a process by which a pre-trained model is further trained on a specific task or dataset. The advantage of fine-tuning is that it allows us to leverage the vast knowledge that these models have already learned from training on a large amount of data, while still adapting the model to specific tasks or domains. The fine-tuning process involves continuing the training of the model on a new dataset or task, allowing the model to adjust and optimize its parameters to perform well on this specific task. In addition, after being trained on a large corpus of data, few-shot learning is an approach in which the model can generalize and perform well on a new task after seeing only a few new examples. This approach provides an effective way to quickly adapt the model to new tasks without extensive training.

To modularize and separate both tasks for better performance, two GPT models were trained: an intent classifier and entity recognizer.

2.2.1. Intent Classifier

This model operates similarly to a natural-language classifier, identifying different intentions within a text as its categories. In the context of a store assistant chatbot’s dialog, the recognizable intentions include *Yes*, *No*, *Thanks*, *Make_Appointment*, *Know_Operation_Hours*, *Know_Location*, *Help*, *Greetings*, *Goodbye*, and *Connect_to_agent*. Additionally, an *Irrelevant* category is included to cover requests that do not fit the aforementioned intentions.

Data Preparation

The initial phase of training the model involved the creation of a training dataset. This dataset is structured as a JSONL file, containing approximately 100 textual instances for each intent, as illustrated in Listing 2. Each entry (called ‘message’) is a dictionary in the conversational chat format since it is required to fine-tune GPT-3.5-turbo models. It consists of a list of three dictionaries, each corresponding to a specific role:

1. System (“role”: “system”). This role is used to provide instructions or contexts that guide the interaction between the user and the assistant. Instructions under this role are for the assistant, not the user, and define how the assistant should behave in the

conversation in general terms. In the intent classification model, the system content corresponds to the following sentence:

Classify text intents from the given set: ['Yes', 'No', 'Thanks', 'Make_Appointment', 'Know_Operation_Hours', 'Know_Location', 'Help', 'Greetings', 'Goodbye', 'Connect_to_agent', 'Irrelevant']. The model should accurately identify the sole intention present in the input text.

2. User (*“role”: “user”*). Represents the inputs or questions posed by the user. These are the queries or comments to which the assistant must respond.
3. Assistant (*“role”: “assistant”*). This corresponds to the responses generated by the assistant (the chatbot) in response to the user’s questions or comments. The assistants’ responses should follow the instructions defined in the system messages. For the intent classification model, the assistant’s content corresponds to the intent class.

Listing 2. JSONL intent dataset excerpt (*PROMPT* has been omitted for better readability).

```

1 {"messages":
2   [{"role": "system", "content": SYSTEM_PROMPT},
3     {"role": "user", "content": "of course"},
4     {"role": "assistant", "content": "Yes"}]}
5
6 {"messages":
7   [{"role": "system", "content": SYSTEM_PROMPT},
8     {"role": "user", "content": "of course not"},
9     {"role": "assistant", "content": "No"}]}
10
11 {"messages":
12   [{"role": "system", "content": SYSTEM_PROMPT},
13     {"role": "user", "content": "thanks for everything"},
14     {"role": "assistant", "content": "Thanks"}]}
15

```

This dataset was created applying the “RetroTrain” [26] process, in which an advanced model is used (GPT-4 model) to generate a training set for an earlier version in a semi-automated way. The model is provided with a set of basic examples and tasked with generating similar sentences according the given structure. These generated examples are meticulously evaluated by a domain expert, who eliminates any examples that are either not representative or incorrect.

Furthermore, a second version of the dataset was created, in which label encoding was used to encode the target labels, assigning a numerical value to each label (from 0 to *num_classes* – 1). This second dataset will be used to fine-tune a model for the same intent classification purpose, checking whether the model uses semantic information from the intent classes for training. Table 3 shows examples of the two types of dataset labels.

Table 3. Examples of *label-encoded* and *full-label strings* dataset versions.

User Entry Example	Full Label	Encoded Label
I’m stuck, can you help me?	Help	6
tell me how you can help me	Help	6
see you later	Goodbye	8
farewell	Goodbye	8
I love star wars films	Irrelevant	10
I would like to learn how to ride a horse	Irrelevant	10

Fine-Tuning

The dataset was divided into training and testing subsets, with 80% (848 entries) allocated for training and 20% (212 entries) allocated for testing.

2.2.2. Named Entity Recognizer

This model focuses on identifying specific types of information, namely *sysdate* for recognizing dates, *sysime* for specific times, and *holiday* for identifying festive occasions such as Christmas, Halloween, and New Years among others.

Data Preparation

The training dataset is structured as a JSONL file with 250 text examples, each potentially featuring a variety of entity combinations as outlined in Listing 3.

1. System (“role”: “system”). The context information for the system is as follows:

```
[now:yyyy-Mon-dd;HH:mm] Your task is to extract dates, times, and
holidays from the user’s query and present them in a structured
JSON format. Follow the output structure: ‘sysime’: [‘Time1’,
‘Time2’], ‘sysdate’: [‘Date1’, ‘Date2’], ‘holiday’: [‘Holiday1’,
‘Holiday2’].
```

First, the date (yyyy-Mon-dd) and time (HH:mm) stamps of the query are specified by the *now* mark. This date and time reference can be considered for extracting named *dynamic entities*, which are moment-dependent. This is necessary because the GPT model cannot know the date and time by itself and usually struggles to deal with dynamic entities [26].

Second, the output JSON format is specified, as it is important to be consistent with the output of the model for use in other systems.

2. User (“role”: “user”). This represents examples of inputs, with entities to be extracted.
3. Assistant (“role”: “assistant”). It details the entities extracted in the specified JSON format.

Listing 3. JSONL entities dataset excerpt (*PROMPT* has been omitted for better readability).

```
1 {"messages": [
2   {"role": "system", "content": "[now:2023-Dec-23;16:30] PROMPT"},
3   {"role": "user", "content": "we meet at 3am on christmas"},
4   {"role": "assistant", "content": "{ 'holiday': ['christmas'], 'sysime': ['03:00'] }"}
5 ]}
6 {"messages": [
7   {"role": "system", "content": "[now:2024-Feb-29;12:00] PROMPT"},
8   {"role": "user", "content": "tomorrow is Friday"},
9   {"role": "assistant", "content": "{ 'sysdate': ['2024-Mar-01'] }"}
10 ]}
11 {"messages": [
12   {"role": "system", "content": "[now:2024-Jan-26;16:00] PROMPT"},
13   {"role": "user", "content": "on Australia Day we go on vacation"},
14   {"role": "assistant", "content": "{ 'holiday': ['day_australia'] }"}
15 ]}
16 {"messages": [
17   {"role": "system", "content": "[now:2019-Apr-07;08:00] PROMPT"},
18   {"role": "user", "content": "the party is on the 9th at 3:35 a.m. in the park"},
19   {"role": "assistant", "content": "{ 'sysdate': ['2019-Apr-09'], 'sysime': ['03:35'] }"}
20 ]}
21 {"messages": [
22   {"role": "system", "content": "[now:2028-Feb-14;07:00] PROMPT"},
23   {"role": "user", "content": "in 2030, on March 7, we will be in a flying car like the
24     ones in Back to the Future"},
25   {"role": "assistant", "content": "{ 'sysdate': ['2030-Mar-07'] }"}
26 ]}
```

“RetroTrain” was also employed during the creation of this dataset (semi-automatic process with GPT-4 to generate initial examples, followed by manual review and refinement).

In general, ISO 8601 has been adopted as the standard format for date notation because of its superior performance in fine-tuning OpenAI models (<https://community.openai.com/t/handling-current-date-and-time-for-GPT-3-5/301607/4>). Last accessed: 18 March 2024). However, a slight modification was introduced wherein the month is represented by three letters instead of two numbers. This alteration serves to enhance and ensure the model’s comprehension as well as to distinguish between days and months, particularly in cases where date formats vary with the arrangement of the day and month. By incor-

porating this variation, the clarity and accuracy of the date representation are augmented, thereby facilitating more effective processing and understanding of the model.

Fine-Tuning

The dataset was divided into training and testing subsets, with 80% (200 entries) allocated for training and 20% (50 entries) allocated for testing.

3. Results

In this section, the evaluation results of intent classification and entity recognition for both systems (G-GPT and FT-GPT) is shown.

3.1. Generic GPT Based System (G-GPT)

3.1.1. Intent Classification

The performance of the system was initially evaluated in terms of its intent classification capabilities using a dataset comprising 212 entries. This evaluation yielded an accuracy rate of 83.96%. The confusion matrix is shown in Figure 2. It clearly depicts a tendency to classify entries as ‘Irrelevant’.

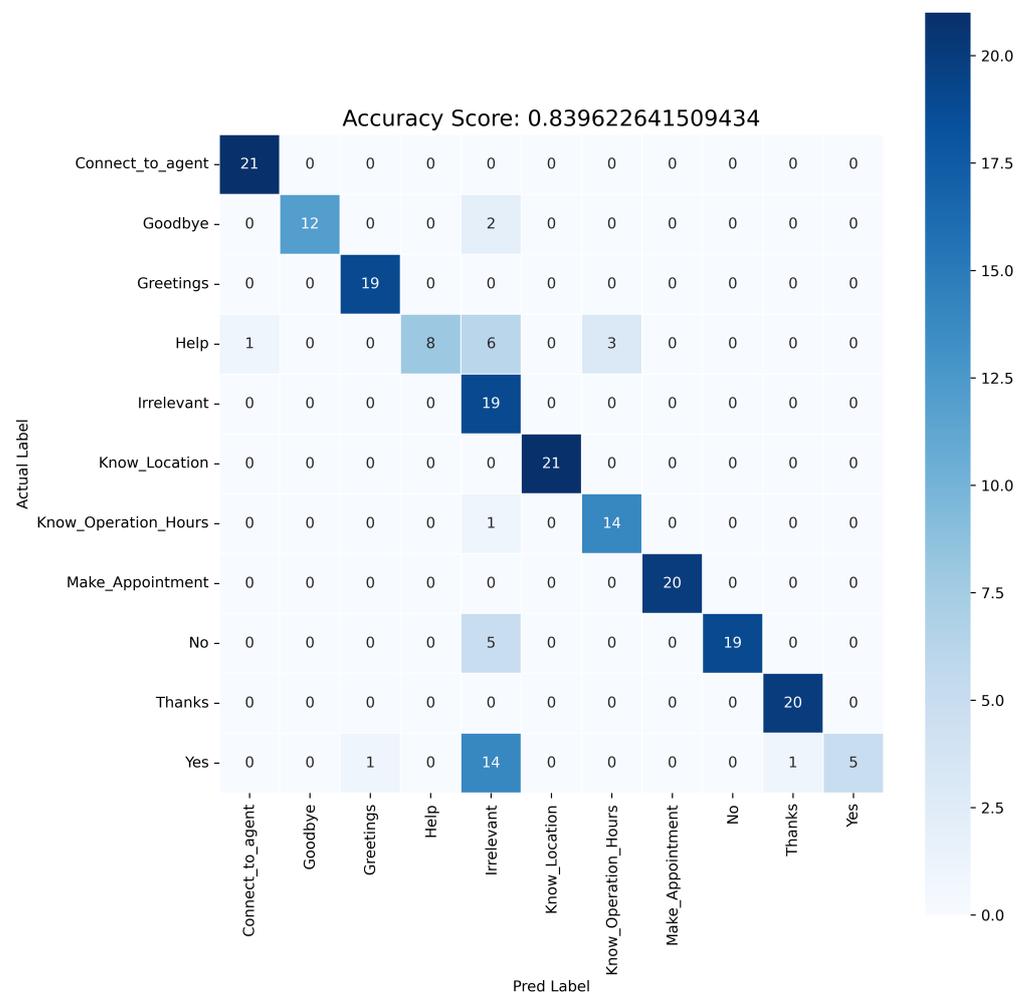


Figure 2. Intent classifier confusion matrix (Generic GPT based system).

3.1.2. Named Entity Recognizer

To evaluate this model, a dataset consisting of 50 sample entries showing a variety of entity combinations was used. The evaluation metrics employed are:

- **Row-Level Accuracy:** A “row” is considered correctly classified only if the model identifies all entities in that specific row, without introducing any additional entities beyond those present in the actual data. To calculate this metric, the number of rows in which the model’s predictions are in complete agreement with the actual data is divided by the overall number of entries in the dataset.
- **Entity-Level Accuracy:** This metric assesses the accuracy with which the model identifies individual entities across various occurrences within the dataset. The accuracy of each entity type is determined by dividing the number of times the model accurately identifies an entity by the total number of occurrences of that entity throughout the dataset.

These calculated metrics are shown in Table 4.

Table 4. Named Entity Recognizer model accuracies.

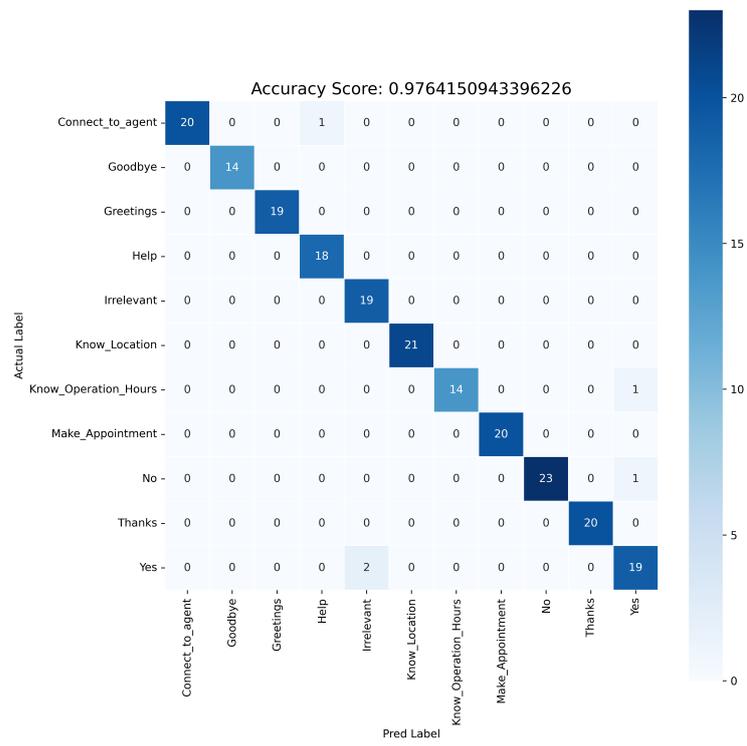
Row-Level Accuracy	
Correct Rows	Accuracy
29/50	0.580
Entity-Level Accuracy	
Entity	Accuracy
sys time	0.821
sys date	0.610
holiday	0.833
no_entities	1.000

It is important to note that in this entity extraction model, a specialized post-processing step is employed to ensure the consistency of the predicted holiday entities. Because of the vast array of holiday values with potential variations in format, such as accents or underscores, and the fact that not all possible holiday entities are explicitly specified in the training dataset, standardizing the output format is crucial for accurate evaluation. This post-processing involves the removal of accents and underscores as well as converting all characters to lowercase. In addition, the cosine similarity function (<https://www.sciencedirect.com/topics/computer-science/cosine-similarity>, Last accessed: 26 March 2024) is used. If the cosine similarity score between the predicted holiday entity and the actual one exceeds a threshold of 0.6, it is considered a match. For instance, variations like “San Valentines Day” and “San Valentines” would be correctly identified as matching entities, even if the format in the model output differs slightly from that in the validation dataset. This comprehensive approach enhances the robustness and accuracy of the entity extraction model for handling diverse holiday entities.

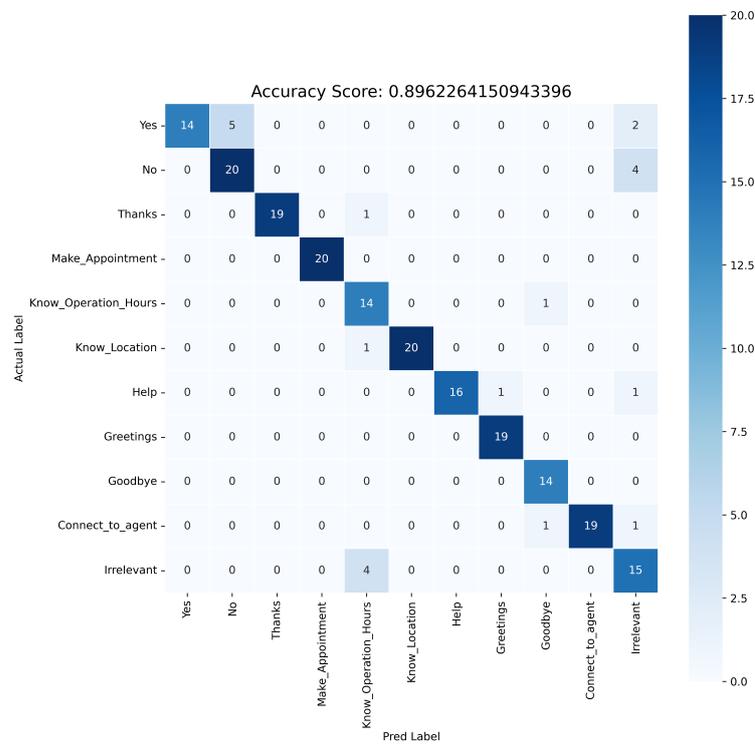
3.2. Fine-Tuned GPT Based System (FT-GPT)

3.2.1. Intent Classifier

This model was evaluated on the test subset of the dataset, which comprises 212 of the 1060 total entries. After the fine-tuning phase, the resulting model achieved an accuracy rate of 97.642%. The corresponding confusion matrix is shown in Figure 3a. Additionally, another model was trained using the same dataset, but with label encoding applied (intent classes encoded with numbers), reaching an accuracy of 89.622%. The confusion matrix for this model is shown in Figure 3b.



(a) Full label dataset



(b) Label encoded dataset

Figure 3. Confusion matrices for the Intent classifier (Fine-tuned GPT based system).

3.2.2. Named Entity Recognizer

Again, the row and entity-level accuracies of this model are depicted in Table 5.

Table 5. Named Entity Recognizer model accuracies.

Row-Level Accuracy	
Correct Rows	Accuracy
47/50	0.940
Entity-Level Accuracy	
Entity	Accuracy
sys time	0.964
sys date	0.976
holiday	1.000
no_entities	1.000

Once again, a specialized post-processing step to ensure consistency in predicted holiday entities is applied.

4. Discussion

4.1. Generic GPT Based System (G-GPT)

In this system, it is explored the use of a generic, pre-trained GPT-3.5-turbo model intent classification and entity extraction, employing a substantial prompt to guide the model's responses (Listing 1). The intent classification and entity recognition processes were separately evaluated.

The confusion matrix of intent classification for this generic system reveals a notable predisposition for the model to classify inputs into the 'Irrelevant' category, which is a default classification for instances where no other categories are matched (Figure 2). This tendency can be attributed to the model's lack of specific contextual understanding or domain knowledge, as it operates without the benefit of fine-tuning or training tailored to the task at hand. Despite this limitation, the generic model demonstrates a reasonable accuracy of 83.96%. However, this level of performance, while impressive for a model not specialized through training, may not meet the stringent requirements of applications demanding higher precision, where the cost of misclassification, particularly into a broad 'Irrelevant' category, could be significant.

For the entity extraction component, the challenges become more pronounced, as shown in Table 4. While the model again shows an impressive ability to perform without bespoke training, achieving commendable accuracy, it encounters specific hurdles in accurate entity recognition and formatting.

- **Formatting Specific Entities:** The model's difficulty in converting entities to a specified format, such as transforming '7PM' to '19:00', underscores its reliance on direct text extraction without the capability of applying contextual transformations or understanding the desired output format. This issue can be attributed to the model's fundamental design, which focuses on reflecting the input text as closely as possible in its outputs, rather than interpreting or converting the text based on external conventions or specific task requirements.
- **Dynamic Entities:** The challenge with non-explicit dynamic entities like "tomorrow", "yesterday", or "the next Monday" highlights a gap in the model's contextual understanding. While incorporating the date and time stamp in the prompt can sometimes anchor the model's responses, its inherent lack of real-time awareness and inability to interpret temporal phrases dynamically leads to inconsistencies in detecting such entities, or even not detecting them in some cases.
- **Region-specific Holidays:** The omission of certain holidays, such as Carnival, from the extracted entities may stem from the model's generalized training data, which may not encompass the rich diversity of regional and country-specific holidays. This

limitation reflects the model's broader challenge of adapting to localized contexts without targeted training data or fine-tuning to guide its understanding.

These aspects underscore the necessity of supplementary techniques to compensate for the inherent limitations of using a generic pre-trained model for tasks that demand nuanced understanding and specific contextual knowledge.

In the context of using generic GPT models, it is essential to take into account potential risks and biases from the pre-training data. Since these models are trained on vast, diverse datasets from third parties, managing specific biases is challenging. The data, which often includes broad internet content, can reflect societal biases, affecting the fairness and inclusiveness of responses. Although this study mainly examines technical performance, it is important to take into account that biases in the pre-trained data, specially in future deployments.

4.2. Fine-Tuned GPT Based System (FT-GPT)

In this system, GPT-3.5-turbo is used as the base model to develop specialized models for intent classification and entity extraction. The intent classification model was fine-tuned in two dataset configurations to explore the impact of label representation on performance:

1. **Label-Encoded Version:** The first variant utilized a dataset in which each intent class is represented by a numerical label. This approach was hypothesized to economize on token usage within each query, potentially leading to cost and time efficiency of the model.
2. **Full Label Strings Version:** The second variant employed at complete textual representation of each class label, embracing the inherent natural language characteristics of the dataset.

The comparative analysis, illustrated in Figure 3a,b reveals a marked disparity in performance between the two fine-tuned models. The full-label string model achieved a significantly higher accuracy of 97.64%, compared to 89.62% for the label-encoded counterpart. This divergence underscores the fine-tuning process' capacity to harness the semantic meaning embedded in full textual labels. By operating with the complete linguistic context of each label, the fine-tuned model can more effectively align its pre-trained knowledge with the specific nuances of the intent classification task. This synergy between the model's inherent language understanding and the explicit semantic cues provided by the full labels likely contributes to the superior performance observed, highlighting the importance of leveraging natural language representations within the fine-tuning process for nuanced tasks, such as intent classification.

The entity extractor model presents impressive accuracy (see Table 5). In particular, its performance on holiday entities, further exemplifies the strengths of fine-tuning. The model demonstrates exceptional accuracy, correctly identifying every instance of holiday entities, including those not explicitly covered in the training data. This success is attributed to the model's ability to generalize and recognize new entities based on learned patterns and contextual clues.

Furthermore, the model's proficiency in discerning date and time entities is particularly noteworthy. By incorporating a date and time stamp as a reference within the training data, the model leverages this contextual anchor to enhance its understanding and handling of temporal expressions (solving the issues related to handle calendar-dependant entities that provide these kinds of GPT approaches [26]). This approach enables the model to adapt to the complexities of calendar-dependent phrases such as 'the next Monday' or 'last Tuesday'. The inclusion of a diverse range of training examples encompassing various temporal expressions is critical. Through exposure to these examples, the model learns to apply its underlying natural language understanding capabilities with specific date and time references. This allows the model to recognize, accurately interpret, and resolve complex temporal entities relative to the reference timestamp. As a result, the fine-tuned model demonstrates remarkable performance in extracting and accurately identifying date and time entities, even in complex scenarios that demand deep understanding of

the calendar and temporal dynamics. This capability underscores the model's advanced comprehension and application of contextual information, thereby setting a high standard for entity extraction performance.

The FT-GPT system presents the risk of introducing new biases or perpetuating existing ones from its generic model base. While fine-tuning allows customization to specific contexts, it can also have human biases from the training data creation. Though this data was carefully compiled and reviewed by domain experts to reduce some risks, the possibility of inadvertent biases, either from the original base model or newly introduced data, persists. The presence of both inherited and introduced biases emphasizes the need for a comprehensive approach to bias detection and mitigation in future developments, since this is out of the scope of this study.

4.3. Comparative Analysis between Both Systems

The comparative analysis between the generic and fine-tuned GPT-based systems for intent classification and entity extraction reveals distinct advantages and challenges associated with each approach.

4.3.1. Handling Dynamic Entities

- **G-GPT.** This system utilizes a very detailed and large prompt to provide context and reference for dynamic entities, such as dates and times. However, its ability to accurately interpret and extract calendar-dependent entities is limited by the lack of specific training. The model relies heavily on the prompt's context to approximate the current date and time but struggles with complex temporal expressions and specified output formats, leading to inconsistent performance.
- **FT-GPT.** The fine-tuned system shows a marked improvement in handling dynamic, calendar-dependent entities. By including date and time stamps as contextual references in the training data, along with diverse examples of temporal expressions, the fine-tuned models develop a nuanced understanding of how to interpret and extract such entities accurately, demonstrating impressive performance even with complex expressions.

4.3.2. Operational Efficiency and Customization

- **G-GPT.** This system offers a "plug and play" convenience, eliminating the need for a training process and dataset creation. However, needing large, detailed prompts to achieve decent results increases token usage and leads to higher costs. Moreover, the system's inability to guarantee correct performance in all cases, particularly with specified output formats, highlights its limitations despite its notable overall accuracy.
- **FT-GPT.** Although creating a dataset for fine-tuning can be tedious, the process does not require a vast dataset, owing to the fine-tuning nature of the task. The fine-tuned system offers significant advantages in terms of accuracy, reduced latency, and cost savings by not requiring large, specific prompts. It also allows for the specification of output formats, ensuring more reliable and customizable results. The system's flexibility extends to detecting less-known entities, such as regional holidays, by learning from provided examples, making it less constrained by the original pre-training dataset limitations.

4.3.3. Adaptability and Versatility

- **G-GPT.** This system boasts easier adaptability to new situations or languages, and offers a versatile solution that can be quickly implemented without the need for additional training or datasets.
- **FT-GPT.** While also adaptable, the fine-tuned system requires the creation of a new (but small) dataset for fine-tuning to new contexts or languages, which might introduce some overhead but allows for more tailored and effective solutions.

4.3.4. Resource Implications and Maintainability

- **G-GPT.** It uses generic pre-trained models that do not require local fine-tuning or extensive computational resources for deployment. These models are continuously updated and maintained by their developers (e.g., OpenAI), thus alleviating the need for local expertise and computational overhead.
- **FT-GPT.** While more resource-intensive due to the fine-tuning process, it relies on the computational power and servers of OpenAI, rather than local resources. Although initial training and updates require significant computational efforts, these processes are managed remotely on OpenAI’s servers. Maintaining and expanding the FT-GPT involves human expertise to periodically update and retrain the model with new data, although the frequency of updates depends on the model and its application context. The fine-tuning of already fine-tuned models is made more efficient by incorporating new examples and retraining only on these, avoiding the exponentially growing training demands typical of building large models from scratch.

Finally, it is also important to consider that the subject matter associated with the data (a store, in this case) also could influence the conclusions obtained. Although it is true that both systems are generalizable, if the subject had been different (for example, cooking), other issues would have arisen, but of the same nature (for example, with units of measurement and calculation of quantities). This underlines a key limitation of this study. While providing detailed insights within the retail context domain, the question arises as to how these findings might translate to different conversational contexts with other challenges and requirements. Thus, while the methodology framework shows potential across general conversational agent applications, more experiments are needed to extrapolate these results to sectors or contexts beyond retail, where different operational dynamics might affect the performance and applicability of the proposed solutions.

Even so, to address potential disparities and biases, extensive efforts were made to ensure the diversity and quality of the datasets used in training the models. The datasets were carefully curated to represent a variety of user interactions by including data from multiple sources within the retail domain, ensuring a broad spectrum of customer intents and entity types. Additionally, to mitigate biases that could affect intent classification and entity recognition, several strategies were employed, including the use of balanced datasets (created and reviewed by domain experts), to ensure no single category or demographic disproportionately influenced the model’s learning.

Table 6 summarizes the comparison between the G-GPT and FT-GPT systems. The choice between these systems depends on the specific requirements and constraints of the conversational AI application in development, the balance between ease of use and deployment, and the need for high precision and customization.

Table 6. Comparative Analysis: Generic vs. Fine-Tuned GPT-Based Systems.

Feature	Generic GPT-Based System	Fine-Tuned GPT-Based System
Handling Dynamic, Calendar-Dependent Entities	Utilizes date and time stamps in context as references, but struggles with complex calendar-dependent expressions due to lack of fine-tuned contextual understanding.	Excellent handles complex calendar-dependent expressions by leveraging date and time stamps in context, enhanced by training examples specific to these scenarios.
Ease of Deployment	High—Operates as a plug-and-play system without the need for a training process or dataset creation.	Lower—Requires the creation of a training dataset for fine-tuning, though not as extensive as training from scratch.
Prompt Specificity and Token Cost	Requires specific, often large prompts to achieve decent results, leading to higher token costs. Not guaranteed to always perform correctly, especially with format-specified entities.	Requires less specific prompts due to learning from training examples, resulting in lower token costs and guaranteed output formats.

Table 6. Cont.

Feature	Generic GPT-Based System	Fine-Tuned GPT-Based System
Accuracy and Performance	Notable but not exceptional. Tends to default to broad categories like 'Irrelevant' for unclear cases.	Impressively accurate, showing significant improvements in entity extraction and intent classification.
Flexibility and Adaptability	More easily adapted to different scenarios or languages, but might require larger or more specific prompts for each new context.	Offers high flexibility in detecting nuanced or less-known entities by including them in the training examples. Adapting to new languages or scenarios may require creating a new, albeit small, dataset for fine-tuning.
Resource Implications and Maintainability	Utilizes pre-trained models with no need for ongoing local tuning, reducing the need for local computational resources and expertise. Updated and maintained by OpenAI.	More resource-intensive, leveraging OpenAI's computational power for fine-tuning. Requires human expertise for periodic updates, but maintainability is facilitated by retraining only with new examples.
Ideal Use Case	Suited for applications where ease of deployment and versatility are prioritized over precision.	Best for scenarios where high accuracy, specific output formats, and efficiency in token usage are critical, despite the initial investment in dataset creation and model fine-tuning.

5. Conclusions

In summary, this study critically analyzed the use of Generic and Fine-tuned GPT models within conversational agent systems, revealing the distinct advantages and limitations inherent to each approach. The Generic GPT-based system (G-GPT) demonstrates broad applicability and ease of implementation, making it an attractive option for applications requiring quick deployment across diverse domains without extensive customization. However, it falls short in scenarios demanding high precision and contextual specificity.

On the other hand, the Fine-tuned GPT-based system (FT-GPT) shows superior performance in terms of accuracy, efficiency, and adaptability to specialized domains. Investment in dataset preparation and model fine-tuning pays off with enhanced conversational agents capable of nuanced understanding and interaction.

The decision to employ a G-GPT or FT-GPT system depends on the specific requirements of the conversational application, including the desired balance between deployment speed, accuracy, and the need for customization. This choice is fundamentally influenced by the specific needs and trade-offs required for the application in question. For instances where high precision, customization, and efficiency are paramount, the fine-tuned approach is more suitable, albeit at the cost of the initial setup and dataset creation. Conversely, the generic GPT system offers ease of deployment and broad versatility, and is ideal for applications where immediate implementation and flexibility across various contexts or languages are paramount. Each approach has its merits, and decisions should be guided by the desired balance between training investment and operational efficiency, precision, and adaptability in the conversational AI application being developed.

While the current study provides a foundational methodology for employing Large Language Models in a specific conversational agent domain, this suggests a pathway for future research. Expanding this research to encompass various industries and contexts would not only validate and possibly enhance the generalizability of the proposed solutions but also tailor the conversational agents more effectively to meet diverse user expectations and needs. Future work will aim to adapt this framework to different domains, thereby broadening its applicability and impact.

It is also considered doing an analysis in terms of latency and response times of both proposed systems. In addition, a comparison with other local LLMs (different from GPT) and third-party systems will be conducted to further explore and validate the findings.

Furthermore, the research shown in this study and the importance of conversational systems, emphasize their applicability at the forefront of social robotics, especially in initiatives such as SHARA. SHARA is a social robot in development (derived from work [5]) that stands out for its ability to engage in self-initiated conversations and perform proactive actions with users. Implementation of the techniques discussed in this study within SHARA's conversational system would not only enrich user-robot interaction and allow for maximum flexibility in its conversational design, but also demonstrate the transformative potential of these approaches for creating broader, humanized communication experiences with robots. Thus, it is considered the integration of comprehensive user experience assessments into this research. While this study has focused on comparing the performance of two approaches using Large Language Models for intent and entity recognition processes specific to CDPs, future research will aim to explore user satisfaction and interaction quality more thoroughly. These evaluations are crucial as it is aimed to integrate these models within the SHARA platform [5].

Author Contributions: Conceptualization, L.V. and D.C.-P.; methodology, L.V., D.C.-P., C.C.D. and A.S.-M.; software, L.V.; validation, L.V., D.C.-P. and A.S.-M.; investigation, L.V. and R.H.; resources, L.V., C.C.D. and A.S.-M.; data curation, L.V., A.S.-M. and G.C.; writing—original draft preparation, L.V.; writing—review and editing, L.V., D.C.-P., C.C.D., A.S.-M., G.C. and R.H.; supervision, R.H.; project administration, R.H.; funding acquisition, R.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by JUNTA DE COMUNIDADES DE CASTILLA-LA MANCHA grant number SBPLY/21/180501/000160 (SHARA3); by AGENCIA ESTATAL DE INVESTIGACIÓN grant number TED2021-130296A-I00 (TAICare as part of the Proyectos Estratégicos Orientados a la Transición Ecológica y a la Transición Digital 2021); by MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES grant number PDC2022-133457-I00 (SSITH project, under Proyectos Prueba de Concepto) and FPU22/00839 (predoctoral contract); and by UNIVERSITY OF CASTILLA-LA MANCHA grant number 2022-PRED-20651 (predoctoral contract).

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and following the Ethics Committee guidelines of University of Castilla-La Mancha (<https://www.uclm.es/misiones/investigacion/serviciosinvestigacion/portaleticacientifica>). Last accessed: 26 March 2024).

Data Availability Statement: The dataset generated and analyzed for this study can be found in <https://mamilab.eu/datasets/> (Last accessed: 26 March 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CDP	Conversational Agent Development
FT-GPT	Fine-tuned GPT based system
GPT	Generative Pretrained Transformer
G-GPT	Generic GPT based system
LLM	Large Language Model
NLP	Natural Language Processing
NLU	Natural Language Understanding
PLM	Pre-trained Language Model

References

1. Motger, Q.; Franch, X.; Marco, J. Software-Based Dialogue Systems: Survey, Taxonomy, and Challenges. *ACM Comput. Surv.* **2022**, *55*, 1–42. [CrossRef]
2. Safi, Z.; Abd-Alrazaq, A.; Khalifa, M.; Househ, M. Technical Aspects of Developing Chatbots for Medical Applications: Scoping Review. *J. Med. Internet Res.* **2020**, *22*, e19127. [CrossRef] [PubMed]
3. Ralston, K.; Chen, Y.; Isah, H.; Zulkernine, F. A voice interactive multilingual student support system using IBM watson. In Proceedings of the 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA), Boca Raton, FL, USA, 16–19 December 2019; pp. 1924–1929. [CrossRef]
4. Villa, L.; Hervás, R.; Dobrescu, C.C.; Cruz-Sandoval, D.; Favela, J. Incorporating Affective Proactive Behavior to a Social Companion Robot for Community Dwelling Older Adults. In *Proceedings of the HCI International 2022—Late Breaking Posters*; Stephanidis, C., Antona, M., Ntoa, S., Salvendy, G., Eds.; Springer: Cham, Switzerland, 2022; pp. 568–575.
5. Villa, L.; Hervás, R.; Cruz-Sandoval, D.; Favela, J. Design and Evaluation of Proactive Behavior in Conversational Assistants: Approach with the Eva Companion Robot. In Proceedings of the International Conference on Ubiquitous Computing & Ambient Intelligence (UCAI 2022), Córdoba, Spain, 29 November–2 December 2022; Bravo, J., Ochoa, S., Favela, J., Eds.; Springer: Cham, Switzerland, 2023; pp. 234–245.
6. MacEclo, P.; Pereira, C.; Mota, P.; Silva, D.; Frade, A.; Madeira, R.N. Conversational agent in mhealth to empower people managing Parkinson’s disease. *Procedia Comput. Sci.* **2019**, *160*, 402–408. [CrossRef]
7. Husak, V.; Lozynska, O.; Karpov, I.; Peleshchak, I.; Chyrun, S.; Vysotskyi, A. Information system for recommendation list formation of clothes style image selection according to user’s needs based on NLP and chatbots. *COLINS* **2020**, *2604*, 788–818.
8. Samuel, I.; Ogunkeye, F.A.; Olajube, A.; Awelewa, A. Development of a Voice Chatbot for Payment Using Amazon Lex Service with Eyowo as the Payment Platform. In Proceedings of the 2020 International Conference on Decision Aid Sciences and Application (DASA), Sakheer, Bahrain, 8–9 November 2020; pp. 104–108. [CrossRef]
9. Bocklisch, T.; Faulker, J.; Pawlowski, N.; Nichol, A. Rasa: Open Source Language Understanding and Dialogue Management. *arXiv* **2017**, arXiv:1712.05181.
10. Malamas, N.; Papangelou, K.; Symeonidis, A.L. Upon Improving the Performance of Localized Healthcare Virtual Assistants. *Healthcare* **2022**, *10*, 99. [CrossRef] [PubMed]
11. Gupta, J.; Singh, V.; Kumar, I. Florence- A Health Care Chatbot. In Proceedings of the 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 19–20 March 2021; pp. 504–508. [CrossRef]
12. Lins, L.F.; Melo, G.; Oliveira, T.; Alencar, P.; Cowan, D. PACAs: *Process-Aware Conversational Agents*; Lecture Notes in Business Information Processing; Springer: Cham, Switzerland, 2022; Volume 436, pp. 312–318. [CrossRef]
13. Astorga, M.; Cruz-Sandoval, D.; Favela, J. A Social Robot to Assist in Addressing Disruptive Eating Behaviors by People with Dementia. *Robotics* **2023**, *12*, 29. [CrossRef]
14. Daniel, G.; Cabot, J.; Deruelle, L.; Derras, M. Xatkit: A Multimodal Low-Code Chatbot Development Framework. *IEEE Access* **2020**, *8*, 15332–15346. [CrossRef]
15. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z. A Survey of Large Language Models. *arXiv* **2023**, arXiv:2303.18223.
16. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018. Available online: <https://api.semanticscholar.org/CorpusID:49313245> (accessed on 26 March 2024).
17. Abdullah, M.; Madain, A.; Jararweh, Y. ChatGPT: Fundamentals, Applications and Social Impacts. In Proceedings of the 2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS), Milan, Italy, 29 November–1 December 2022; pp. 1–8. [CrossRef]
18. Ray, P. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet Things Cyber-Phys. Syst.* **2023**, *3*, 121–154. [CrossRef]
19. Li, M.; Wang, R.; Zhou, X.; Zhu, Z.; Wen, Y.; Tan, R. ChatTwin: Toward Automated Digital Twin Generation for Data Center via Large Language Models. In Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, Istanbul, Turkey, 15–16 November 2023; pp. 208–211. [CrossRef]
20. Mir, Y.O.V.; Pupo, I.P.; Piñero Pérez, P.Y.; Acuña, L.A.; Pérez, R.B. Ecosystem for Construction of Hybrid Conversational Systems (BRasa). *Stud. Comput. Intell.* **2024**, *1134*, 213–239. [CrossRef]
21. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
22. Tamagnone, N.; Fekih, S.; Contla, X.; Orozco, N.; Rekabsaz, N. Leveraging Domain Knowledge for Inclusive and Bias-aware Humanitarian Response Entry Classification. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, 19–25 August 2023; pp. 6219–6227. [CrossRef]
23. Sidogi, T.; Mbuva, R.; Marwala, T. Stock Price Prediction Using Sentiment Analysis. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 46–51. [CrossRef]
24. Liu, C.; Sun, K.; Zhou, Q.; Duan, Y.; Shu, J.; Kan, H.; Gu, Z.; Hu, J. CPML-ChatGLM: Parameter-efficient fine-tuning ChatGLM with Chinese patent medicine instructions. *Sci. Rep.* **2024**, *14*, 6403. [CrossRef] [PubMed]

25. ISO 8601: Date and Time Format. Available online: <https://www.iso.org/iso-8601-date-and-time-format.html> (accessed on 26 March 2024).
26. Villa, L.; Carneros-Prado, D.; Sánchez-Miguel, A.; Dobrescu, C.C.; Hervás, R. Conversational Agent Development Through Large Language Models: Approach with GPT. In Proceedings of the 15th International Conference on Ubiquitous Computing & Ambient Intelligence (UCAmI 2023), Riviera Maya, Mexico, 28–30 November 2023; Bravo, J., Urzáiz, G., Eds.; Springer: Cham, Switzerland, 2023; pp. 286–297.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.