Received 5 April 2023, accepted 3 May 2023, date of publication 8 May 2023, date of current version 12 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3274199

RESEARCH ARTICLE

Chat2VIS: Generating Data Visualizations via Natural Language Using ChatGPT, Codex and GPT-3 Large Language Models

PAULA MADDIGAN AND TEO SUSNJAK
School of Mathematical and Computational Sciences, Massey University, Auckland 0632, New Zealand
Corresponding author: Teo Susnjak (t.susnjak@massey.ac.nz)

ABSTRACT The field of data visualisation has long aimed to devise solutions for generating visualisations directly from natural language text. Research in Natural Language Interfaces (NLIs) has contributed towards the development of such techniques. However, the implementation of workable NLIs has always been challenging due to the inherent ambiguity of natural language, as well as in consequence of unclear and poorly written user queries which pose problems for existing language models in discerning user intent. Instead of pursuing the usual path of developing new iterations of language models, this study uniquely proposes leveraging the advancements in pre-trained large language models (LLMs) such as ChatGPT and GPT-3 to convert free-form natural language directly into code for appropriate visualisations. This paper presents a novel system, Chat2VIS, which takes advantage of the capabilities of LLMs and demonstrates how, with effective prompt engineering, the complex problem of language understanding can be solved more efficiently, resulting in simpler and more accurate end-to-end solutions than prior approaches. Chat2VIS shows that LLMs together with the proposed prompts offer a reliable approach to rendering visualisations from natural language queries, even when queries are highly misspecified and underspecified. This solution also presents a significant reduction in costs for the development of NLI systems, while attaining greater visualisation inference abilities compared to traditional NLP approaches that use hand-crafted grammar rules and tailored models. This study also presents how LLM prompts can be constructed in a way that preserves data security and privacy while being generalisable to different datasets. This work compares the performance of GPT-3, Codex and ChatGPT across several case studies and contrasts the performances with prior studies.

INDEX TERMS ChatGPT, codex, end-to-end visualisations from natural language, GPT-3, large language models, natural language interfaces, text-to-visualisation.

I. INTRODUCTION

The ability to generate visualisations based on natural language (NL) text has long been a desirable goal in the field of data visualisation. Research into Natural Language Interfaces (NLIs) for visualisation has emerged as the primary field that has recently spearheaded the advancements in this area [1], [2]. These interfaces allow users to generate visualisations in response to NL queries or prompts that are free from programming and technical constructs, thus providing a flexible and intuitive way to interact with data. The ultimate aim is to devise systems enabling users to express queries like *"Show me the sales trend?"* which are correctly understood and depicted by automatically discerning the correct chart type.

The data visualisation paradigm can be difficult to learn for users [3] who must translate their analysis intentions into tool-specific operations which may take the form of point-and-click applications or code in various programming languages. Therefore, NLIs can improve the usability of visualisation tools [4] by making them more convenient and novice-friendly as well as effective, and ultimately, inclusive

The associate editor coordinating the review of this manuscript and approving it for publication was Walter Didimo.

for a broader range of users. As such, approaches like NLIs have the potential to make data and insights more accessible to a wider audience and lower the barrier [3] by allowing users to express their queries and analysis intentions in a form that is most natural to them.

The process of translating NL inputs into visualisations (NL2VIS) involves several non-trivial tasks. Generally, the input query is first parsed and modelled, then the required data attributes are identified, and the low-level analytic tasks expressed within the query are discerned. These low-level tasks, such as filtering, correlation, and trend analysis, must then be translated into code to be executed. Finally, the input query is analysed and matched with the most appropriate visualisation, and then code is invoked to render the data. Each component in the pipeline is error-prone.

The implementation of NL2VIS is a particularly challenging task due to the inherent characteristics of NL, such as ambiguity and the underspecification of requirements in the prompts, as well as unavoidable typographical errors. These characteristics of NL make it difficult to accurately interpret the user's intent and generate appropriate visualisations with the existing technologies and approaches [5]. Despite these challenges, the popularity of NLIs for data visualisation has continued to grow, being driven by the demand for data analytics and the increasing need for flexible and intuitive ways of interacting with data.

The performance of NLIs for visualisations is largely dependent on Natural Language Processing (NLP) models [6] and their robustness in understanding NL. A recent comprehensive survey [6] of the use of NLIs for visualisations noted that while most existing approaches utilise hand-crafted grammar rules that require proficiency with typical NLP toolkits, more complex large language models (LLMs) like GPT-3 which are capable of achieving human-level performance on specific tasks [7] have not yet been implemented or explored for visualisation generation directly from NL. LLMs have revolutionised the field of NL understanding and generation. These models are based on the transformer architecture [8] which has demonstrated remarkable successes in tasks such as sentiment analysis, question-answering, and language generation owing both to the effectiveness of the architecture, and also due to them being trained on vast amounts of data. The data used to train these models typically comes from the internet and can include websites, books, and code repositories. As a result of being trained on such a large amount of data, LLMs have developed a comprehensive understanding of the structure and meaning of language, allowing them to perform tasks in a highly sophisticated manner, but also important to this study, the ability to generate code in response to NL requests. For these reasons, LLMs offer the potential to accurately understand free-form NL input and the capability to convert it into functional code that generates suitable visualisations which correctly pair with the underlying data types.

Contribution: This study is the first of its kind to propose an end-to-end NL2VIS solution which converts free-form conversational language into visualisations via LLMs. The present work leverages the most recent advances in LLM technologies and AI in general, specifically investigating ChatGPT and GPT-3, which are considered state-of-the-art [7]. The advantage of using pre-trained LLMs for this task is that they offer not only accuracy gains in robustly understanding user requests, even when malformed, but they also result in accelerated development turnarounds and a significant decrease in costs. Despite the capabilities of these models to display human-like performance on specific tasks, the integration of LLMs with NLIs for visualisation has not been explored in published literature. Therefore, this study seeks to examine the capability and comparative performances of two types of GPT-3 models and ChatGPT for NL2VIS tasks that also include the automatic selection of chart types, through numerous experiments and examples. The experiments demonstrate the potential of the LLMs to enhance the performance of NL2VIS in terms of accuracy, efficiency, and cost reduction, thus potentially minimising the need to devise new language models for this problem going forward. This work also demonstrates how LLMs can be used in a manner that is data-privacy preserving and security-aware, making the approach generalisable to all types of datasets, irrespective of confidentiality concerns. In the process, this study shows how prompts for LLMs can be engineered to elicit desired outputs. Furthermore, the system developed here has also been made publicly available through an online application for testing and experimentation, with the ability for users to upload their datasets and generate visualisations.[1]

II. RELATED WORK

In recent years, the idea of using NL as a way to create visualisations has gained significant attention within the field of data visualisation [4], [9]. The use of NLIs has also grown in popularity in commercial software as a means of improving the usability of visualisation systems. Various tools, such as IBM Watson Analytics, Microsoft Power BI, Tableau, ThoughtSpot, and Google Spreadsheet, have to varying degrees implemented NLIs that allow users to generate visualisations in response to NL queries or prompts [1], [3], [4], [10], [11] indicating the level of demand for this innovation. These are early commercial iterations of this technology as these systems typically constrain NL interactions to data queries and standard chart types, and do not support more complex or open-ended visualisation tasks [3].

NL modelling techniques, which underpin NL2VIS, can broadly be categorised into traditional symbolic-based NLP approaches which rely on explicit rules and representation of the language structure and the emerging neural machine translation methods which rely on language models developed typically through deep learning [10]. Fig. 1 depicts the

[1]Chat2VIS is currently hosted on Streamlit Cloud and can be accessed via https://chat2vis.streamlit.app/

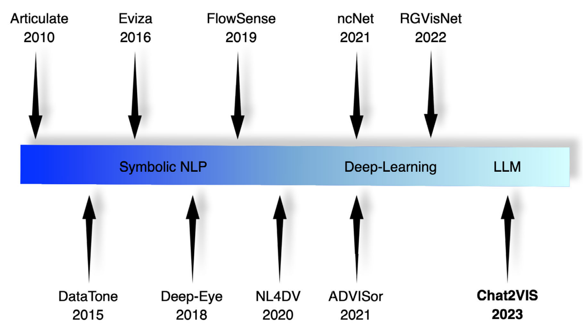


FIGURE 1. **Timeline depicting the recent evolution of NL2VIS systems and the proposed Chat2VIS system.**

recent evolution of NL2VIS systems in published literature,[2] and their trend towards more sophisticated machine learning approaches, where the authors in this study posit that the logical trajectory is pointing in the direction of using the most advanced AI systems like LLMs for language understanding and code generation.

A. SYMBOLIC NLP APPROACHES

Symbolic approaches to NLP can involve heuristic, rule-based and probabilistic grammar-based approaches for parsing and understanding NL. Heuristic algorithms use pre-defined rules or heuristics to find approximate solutions for complex problems like NL and tend to be less accurate than other methods [13]. Meanwhile, rule-based approaches rely on predefined rules created by experts for understanding NL. They are more accurate and reliable but, also less flexible in handling complex queries [9]. Probabilistic grammar-based approaches use formal grammar rules and probability distributions over the possible parses of a given input. These approaches are considered more accurate as well as flexible than the previous approaches but require more computational resources. Each of these approaches presents complexities and can be time-consuming and challenging to resolve especially for developers without prior experience [1].

The majority of the earlier systems have been developed using these approaches [11], including Articulate [14], DataTone [15], Eviza [16], and Deep-Eye [17], with each using distinct methodologies for mapping NL queries to visualisations. Certain systems also allow for user interaction to manage ambiguities. Meanwhile, recent state-of-the-art studies like NL4DV [1] and FlowSense [18] using symbolic NLP approaches have relied on semantic parsers such as NLTK [19], NER, and Stanford CoreNLP [20], to automatically add layers of valuable semantic information, such as parts-of-speech (PoS) tagging and named entity recognition to the NL input which improves accuracy. NL4DV [1], an open-source Python toolkit, accepts NL queries for a given dataset and outputs a JSON object with keys for dataset attributes, analytical tasks, and Vega-Lite visualisation specifications. This framework enables those who lack NLP expertise to harness NL2VIS concepts by creating NLIs or rendering the output in their own systems.

[2]YoloPandas [12] , a Python module leveraging LLMs to execute commands with natural language that also includes visualisations was released as this manuscript was being prepared for publication.

B. DEEP-LEARNING MODEL APPROACHES

A more promising route has been highlighted by recent advancements in NL2VIS systems which have shifted the focus towards neural end-to-end models that leverage deep learning. These approaches combine the processes of language understanding, reasoning and chart generation in a single system and are closer to the proposed system in this study. These approaches aim to achieve greater robustness, flexibility and adaptability compared to traditional methods [9].

One notable system developed along these lines is ADVISor [21]. ADVISor comprises multiple customised deep learning modules for determining the necessary visualisation data and appropriate attributes, as well as the filter and aggregation operations. These models are underpinned by the large language transformer-based BERT [22] model which performs the vectorisation of user input and the data attribute names for subsequent steps. Meanwhile, the chart type is chosen based on a predefined rule map.

ncNet [4] is a novel approach which also uses transformer-based models and visualisation-aware optimisations. ncNet is a machine learning model that is trained using the nvBench [23] dataset, which maps natural language queries to visualisations. The system accepts the NL query and an optional chart template as an additional input to constrain the possible visualisations being outputted. The approach has been evaluated through a quantitative evaluation as well as a user study, and has shown promising accuracy in the nvBench benchmark. Recently, this system has been extended to include speech-to-visualisation capabilities [11].

Meanwhile, a system called RGVisNet [5] was developed that decouples the NL2VIS process into two subtasks which consist of a hybrid retrieval and a generation framework. The first part of the system retrieves the most relevant visualisation query from a large-scale visualisation codebase which serves as a candidate prototype that is refined in the next step by a GNN-based deep-learning model.

Summary of Literature and Research Aims: The trends in NL2VIS research highlight that the focus is moving towards using transformer-based deep learning models and end-to-end solutions, and away from the complex task of engineering symbolic-based language models. Recent advancements, such as the use of pre-trained language models like BERT and domain-specific models like ncNet, have shown promising results in various benchmarks. However, there is a gap in the literature exploring the recent state-of-the-art pre-trained LLMs which are significantly larger and more sophisticated. This work aims to address this gap

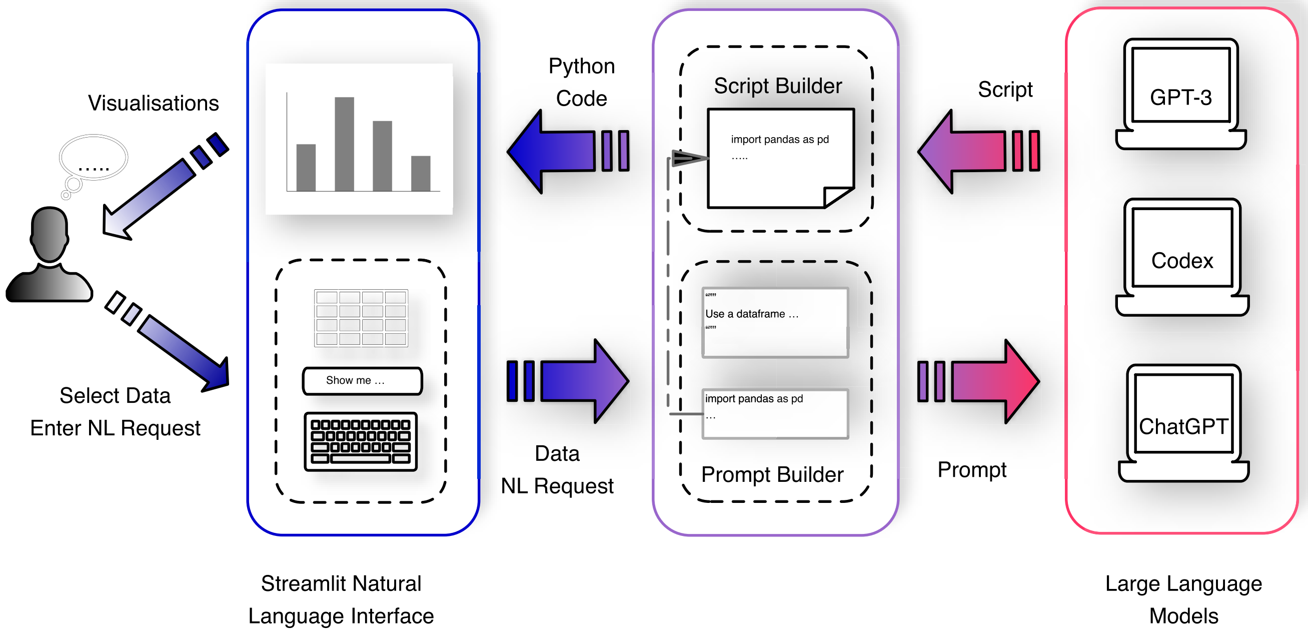


FIGURE 2. **The Chat2VIS architecture for converting free-form query text into visualisations using the large language models GPT-3, Codex, and ChatGPT.**

and examines the potential to simplify the NL2VIS pipeline, while making it more robust for free-form NL and complex visualisation tasks. Additionally, while advanced systems like ADVISor [21] and ncNet [4] rely on various explicit mechanisms for determining which types of visualisations are to be rendered, the proposed system addresses this gap by investigating the ability to delegate the chart selection decision-making to the AI component.

C. RESEARCH QUESTIONS

In light of the existing literature, this study poses the following research questions:

- (RQ1) Do current LLMs support accurate end-to-end generation of visualisations from NL?
- (RQ2) How can LLMs be effectively leveraged to elicit the generation of correct and appropriately rendered charts?
- (RQ3) Which LLMs tend to perform more robustly to NL prompts? How do they perform against other state-of-the-art approaches?
- (RQ4) What are the limitations of using LLMs for NL2VIS as well as the future research directions?

III. METHODOLOGY

This study explored the ability of three OpenAI LLMs to generate Python scripts for visualising data based on NL queries without explicit direction as to which types of graphs to generate. The models chosen for this investigation include the most advanced model family, Davinci, specifically the GPT-3 model *"text-davinci-003"* and the Codex model *"code-davinci-002"*, as well as the most recent addition, ChatGPT.[3]

The Davinci model family consists of billions of parameters,[4] and is widely considered to be the most capable of all available models in its ability to follow instructions. This model family is based on GPT-3, with the Codex model receiving additional training data from a massive quantity of GitHub repository code. This makes the Codex model particularly well-suited for translating NL into code, being proficient in over a dozen programming languages, with Python being the target language in this study.

Fig. 2 depicts the overview of the developed Chat2VIS system. A user enters a NL query via a Streamlit NLI app which is an open-source Python framework for web-based dashboards. The query is combined with a prompt script which engineers a suitable prompt for a selected dataset. The prompt is forwarded to selected LLMs, which return a Python script that is subsequently rendered within the Streamlit NLI.

A. NATURAL LANGUAGE INTERFACE

The interface for the Chat2VIS software artefact used in this study is depicted in Fig. 3. The interface enables users to select a dataset and enter free-form text describing their data visualisation intent. The side toolbar provides the functionality to import additional CSV files and SQLite databases, with options to choose the desired LLMs to render the

[3]OpenAI documentation refers to "text-davinci-003", "code-davinci-002" and ChatGPT models as belonging to the GPT-3.5 series https://platform.openai.com/docs/model-index-for-researchers During the final preparation of the manuscript, GPT-4 has been released.

[4]The reported number of parameters is 175B.

visualisations. An OpenAI Access Key is required to access the models and must be entered prior to querying. An input box is provided for entering the NL free-format text. Visualisations are presented for each selected model, with the actual dataset also shown to the users.

B. PROMPT ENGINEERING

The most effective method for obtaining the desired output from the LLM is to use the "show-and-tell" technique[5] by supplying within the prompt an example together with instructions. The proposed system generates an LLM prompt consisting of two parts: (1) a Description Prompt built from a Python docstring and declared using triple double quotes """ at the beginning and the end of the definition, (2) a Code Prompt comprising of Python code statements that provides guidance and a starting point for the script.

The structure of the prompt is presented in Fig. 4 and is described by way of an example dataset, using the *products* table from the nvBench database[6] *department_store*, pre-loaded into dataframe *df_products*. For context, the example dataset lists the prices for a selection of clothing products such as coloured jeans and tops, together with hardware products like monitors and keyboards. There are four columns, *product_id*, *product_type_code*, *product_name* and *product_price* which have data types *int64*, *object*, *object* and *float64* respectively.

The Description Prompt and the Code Prompt shown in Fig. 4 are depicted in components in order to aid their explanation. The bolded type highlights substitution values which are variable and dependent on the chosen dataset and are thus specific to this illustration. These provide an overview of the DataFrame (a tabular data object) to the LLM, listing column names, their data types and categorical values, which assists the LLM in understanding the context, while maintaining data privacy by withholding the actual raw values from being sent to the LLMs. Each component of the proposed prompt is described as follows:

1) The Description Prompt is initiated with a Python docstring Fig. 4(a)
2) In Fig. 4(b) the LLM is explicitly informed to use a DataFrame with the name *df* which enables a reference to be made to this DataFrame by a specific name, thereby avoiding any confusion that might arise if the LLM were to assign a different name to the DataFrame. Even though it was opted to utilise a pre-loaded CSV file, there may be instances when the LLM (typically ChatGPT) includes code for loading the file. By anticipating the file name as data_file.csv, one can easily identify and eliminate this code if required before execution.
3) The Description Prompt in Fig. 4(c) consists of one entry for each column indicating its data type. If a column with an object data type has less than 20 distinct values, it is deemed as a categorical type, and its values are enumerated in the prompt. This listing could aid the LLM in identifying keywords for certain requests, for instance, prompts relating to keyboards or black jeans in this example.
4) In Fig. 4(d) the LLM is asked to decide on appropriate naming for the x and y axes, and the plot title. In some cases, when working with grouped data, particularly box plots, the LLMs add a plot super-title unnecessarily. To address this, it is recommended to remove it by using positive instructions and explicitly setting it to empty.
5) To encourage correct syntax, the Python version is included as shown in Fig. 4(e).
6) The Description Prompt concludes with an instruction to create a plotting script with the supplied user query in Fig. 4(f).
7) For this demonstration, the submitted NL query from nvBench is: *What is the highest price of product, grouped by product type? Show a bar chart, and display by the names in desc.*
8) The Python docstring in Fig. 4(g) closes off the Description Prompt
9) The Code Prompt begins with import statements for the required Python packages that is deemed helpful for the LLMs to use in Fig. 4(h).
10) To foster uniformity in the plot layout, the prompt asks for a single subplot with a fixed figure size in Fig. 4(i) in an attempt to render equivalently sized plots on the interface.
11) By assigning a copy of the named DataFrame in Fig. 4(j) to the variable *df* in the Code Prompt, consistency is ensured within the script and enables the original DataFrame to be retained for further querying.

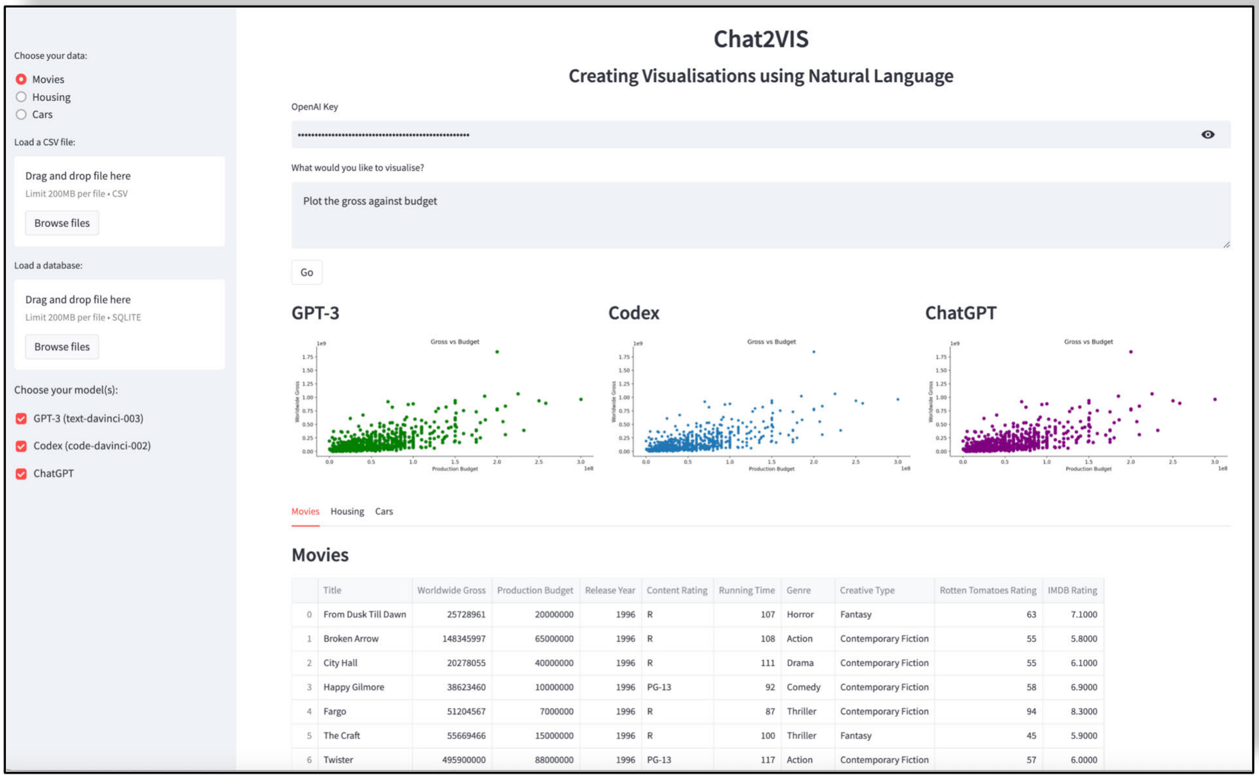
[5]https://beta.openai.com/docs/guides/completion/prompt-design
[6]https://github.com/TsinghuaDatabaseGroup/nvBench/blob/main/nvBench_VegaLite/VIS_6.html

TABLE 1. API parameters.

Parameter	Setting
engine	text-davinci-003 or code-davinci-002
prompt	<constructed NL prompt string>
temperature	0
max_tokens	500
stop	plt.show()

Once formulated, the two prompt elements are amalgamated, with the resulting string submitted to the LLMs via the text completion endpoint API with settings shown in Table. 1.

Unspecified parameters remain at their default values. With many variations to Python scripting, the model temperature is set to 0 to help ensure the LLM is less creative and more consistent in its code generation. To encourage the model to avoid any unnecessarily long script responses, a token limit of 500 is enforced for responses. The authors believe this limit is sufficient for generating a Python script within the context of this study. A stopping point is specified to avoid returning multiple script examples and code alternatives.



Choose your data: [✓ Movies] Housing Cars

Choose your model(s): [✓ GPT-3 (text-davinci-003)] [✓ Codex (code-davinci-002)] [✓ ChatGPT]

What would you like to visualise? Plot the gross against budget

Movies

	Title	Worldwide Gross	Production Budget	Release Year	Content Rating	Running Time	Genre	Creative Type	Rotten Tomatoes Rating	IMDB Rating
0	From Dusk Till Dawn	25728961	20000000	1996	R	107	Horror	Fantasy	63	7.1000
1	Broken Arrow	148345997	65000000	1996	R	108	Action	Contemporary Fiction	55	5.8000
2	City Hall	20278055	40000000	1996	R	111	Drama	Contemporary Fiction	55	6.1000
3	Happy Gilmore	38623460	10000000	1996	PG-13	92	Comedy	Contemporary Fiction	58	6.9000
4	Fargo	51204567	7000000	1996	R	87	Thriller	Contemporary Fiction	94	8.3000
5	The Craft	55669466	15000000	1996	R	100	Thriller	Fantasy	45	5.9000
6	Twister	495900000	88000000	1996	PG-13	117	Action	Contemporary Fiction	57	6.0000

FIGURE 3. The Streamlit Chat2VIS Natural Language Interface enabling dataset visualisations from free-form text queries.

C. SCRIPT REFINEMENT AND RENDERING

On the return of the script from the API for each model, the Code Prompt is inserted at the start and the Python code may be edited to eliminate unnecessary instructions. It is rendered on the interface for each LLM. To further enhance the visualisations, users can refine their NL query, including requesting alternative chart types, plot colours, labels etc.

D. Chat2VIS EVALUATION

The capabilities of the proposed Chat2VIS system to render visualisations based on NL input via LLMs, and their unique decision-making skills in autonomously selecting appropriate charting elements are demonstrated over six case studies covering five datasets. Four of these case studies are reproduced from examples in existing literature, comparing the resulting visualisations here with those from prior studies using nvBench SQLite databases[7] and the NL4DV Python package.[8] One case study is taken from NL4DV [1], one from ADVISor [21] in which the authors also use NL4DV in their evaluation, and the final two using databases from nvBench [23].

In the remaining two case studies, this study tests the capabilities of the LLMs and the prompt scripts to handle misspecification in the form of typographical errors as well as acute underspecification, where the ability of the LLMs to exhibit reasoning and assumptions in the context of the priming text is explored.

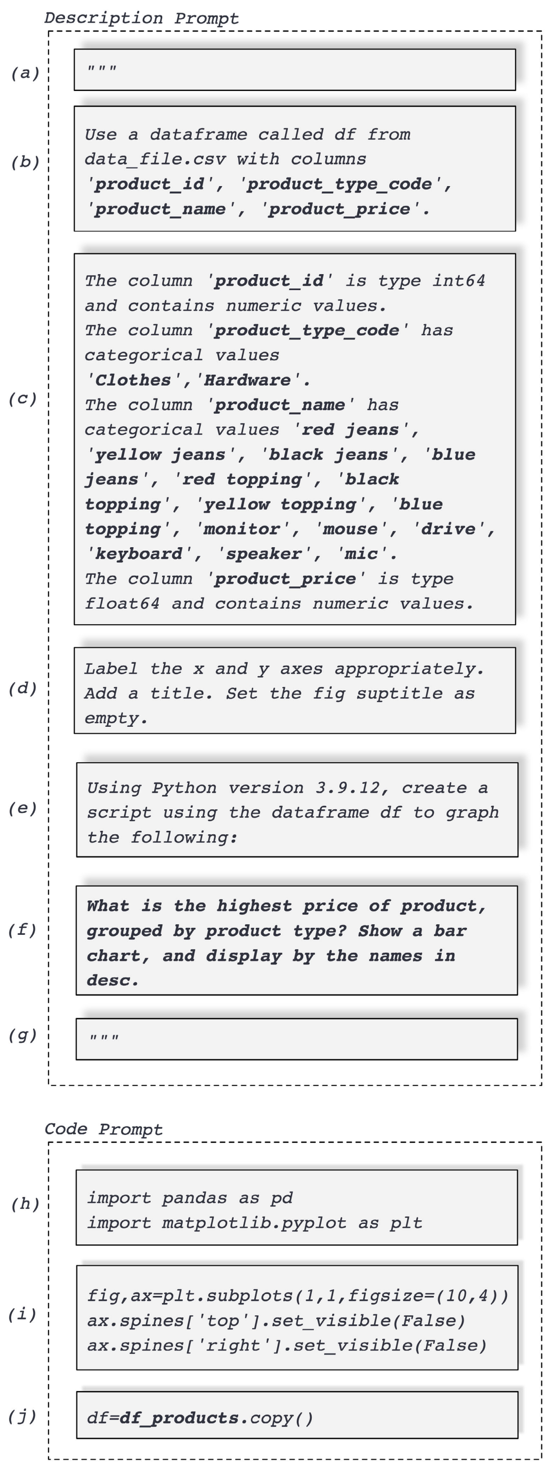
The results are inspected and evaluated visually for correctness and suitability. All case study examples are also reproducible from the online web app. The non-deterministic nature of the LLMs does occasionally lead to variability in plot generation even when an identical prompt is resubmitted.

[7]https://sites.google.com/view/nvbench
[8]https://nl4dv.github.io/nl4dv/documentation.html

IV. RESULTS

A. CASE STUDY 1: DEPARTMENT STORE DATASET

The first example is based on the *products* table illustrated above in the description of the prompt engineering which originated from the nvBench *department_store* database. The test query is as follows: *"What is the highest price of product, grouped by product type? Show a bar chart, and display by the names in desc."*. The query is categorised by nvBench as an *easy* visualisation for NLI systems. Fig. 5 shows results generated by Chat2VIS alongside the correct nvBench visualisation. GPT-3 and Codex produce identical results, with ChatGPT rotating labels for ease of reading and arguably providing a slightly more comprehensive title. All three LLMs provide more informative *x* and *y* axis labelling and titles than the ground truth example from nvBench.


Description Prompt

(a)
```
"""
```
(b)
```
Use a dataframe called df from
data_file.csv with columns
'product_id', 'product_type_code',
'product_name', 'product_price'.
```
(c)
```
The column 'product_id' is type int64
and contains numeric values.
The column 'product_type_code' has
categorical values
'Clothes','Hardware'.
The column 'product_name' has
categorical values 'red jeans',
'yellow jeans', 'black jeans', 'blue
jeans', 'red topping', 'black
topping', 'yellow topping', 'blue
topping', 'monitor', 'mouse', 'drive',
'keyboard', 'speaker', 'mic'.
The column 'product_price' is type
float64 and contains numeric values.
```
(d)
```
Label the x and y axes appropriately.
Add a title. Set the fig suptitle as
empty.
```
(e)
```
Using Python version 3.9.12, create a
script using the dataframe df to graph
the following:
```
(f)
```
What is the highest price of product,
grouped by product type? Show a bar
chart, and display by the names in
desc.
```
(g)
```
"""
```
Code Prompt

(h)
```
import pandas as pd
import matplotlib.pyplot as plt
```
(i)
```
fig,ax=plt.subplots(1,1,figsize=(10,4))
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
```
(j)
```
df=df_products.copy()
```

FIGURE 4. Description and Code Prompts built from combining the free-form text query with the selected dataset to be submitted to the LLMs.

B. CASE STUDY 2: COLLEGES DATASET

This dataset holds information on students and staff at U.S. public and private colleges. The *colleges* dataset[9] was used by [1] to demonstrate the Vega-Lite editor for rendering queries visualised by NL4DV. The output from their toolset rendered via Streamlit is compared with Chat2VIS results. The broadness of the submitted query used in their study *"Show debt and earnings for Public and Private colleges."* allows flexibility for different interpretations of the request.

The figure (Fig. 6) demonstrates that GPT-3 generated a scatter plot, similar to that produced by NL4DV, to represent the relationship between median debt and median earnings for all colleges by type. Codex interpreted the query by creating a bar chart to show the mean value of debt and earnings for each college type. ChatGPT focused on the distributions of debt and earnings by creating a box-and-whisker plot for public and private colleges. The models were able to differentiate between public and private colleges using the "Control" column, which was not easily recognisable as a college type. However, by providing categorical values in the prompt, the models were able to accurately categorise the data based on the "public/private" indicator in the column. In addition, the models were required to perform further reasoning in order to arrive at the decision to extract data from the "Median Debt" and "Median Earnings" columns, having only been queried to show debt and earnings, thus illustrating the capability of AI inference potentials.

C. CASE STUDY 3: ENERGY PRODUCTION DATASET

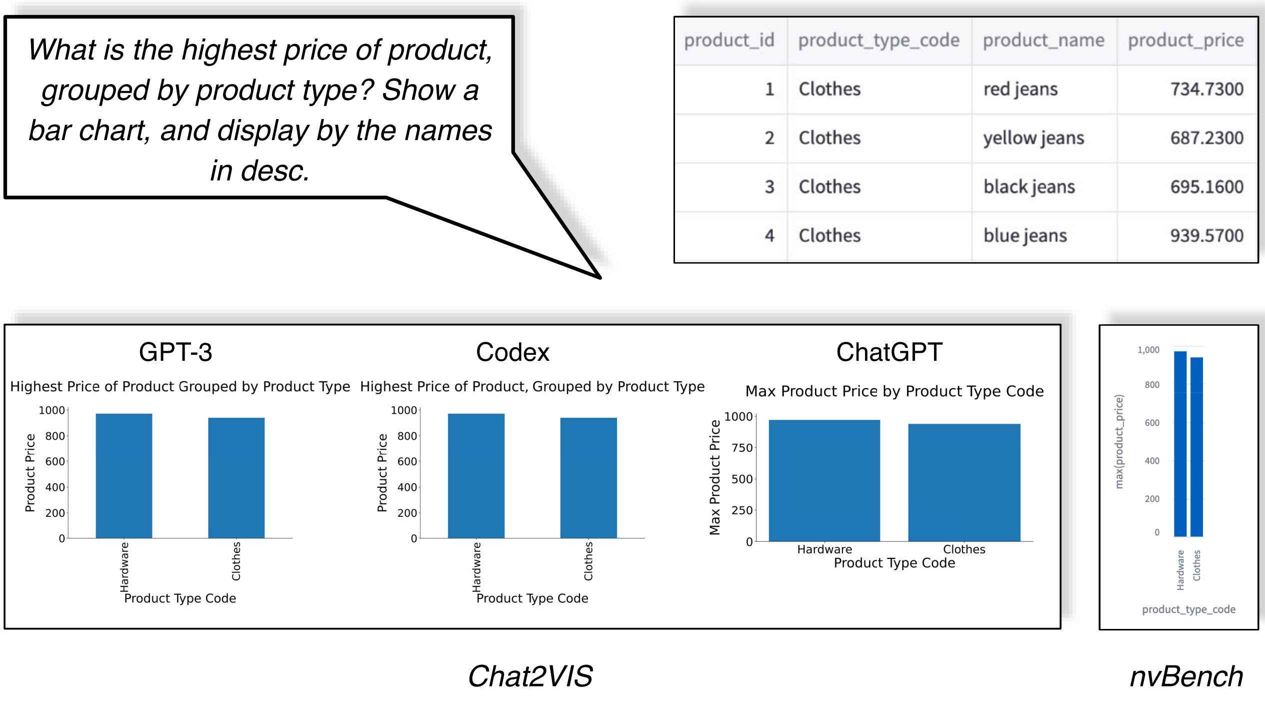
The next set of visualisations is depicted on the Energy Production Dataset described in the ADVISor study and compared to those of both ADVISor and NL4DV outputs. The dataset details coal, oil, gas, and nuclear energy production in megawatt-hours per person per year from 2000 to 2011 for an unspecified country, including population statistics.

Here, the test query used is from the ADVISor study, *"What is the trend of oil production since 2004?"*. The results from Chat2VIS are shown in Fig. 7 together with the output from NL4DV rendered via Streamlit. The results can be compared with the ADVISor plot presented in their study. All three LLMs select a line plot as the most suitable style of plot for this query, with GPT-3 and ChatGPT correctly showing data from 2004 onward. Codex, however, neglects to incorporate this detail into its code generation and has depicted data from 2000 onwards. NL4DV has produced an incorrect visualisation due to its semantic parsing limitations which lacks flexibility, as mentioned in [21]. The ADVISor plot also selected the correct chart type, but like Codex, it was unable to filter the data to include 2004 onwards. It did, however, highlight the data points from 2004 onwards drawing attention to the oil trend and the selected data range.

D. CASE STUDY 4: CUSTOMERS AND PRODUCTS CONTACTS DATASET

The following example provides a demonstration of a more complex NL2VIS task from the *products* table in

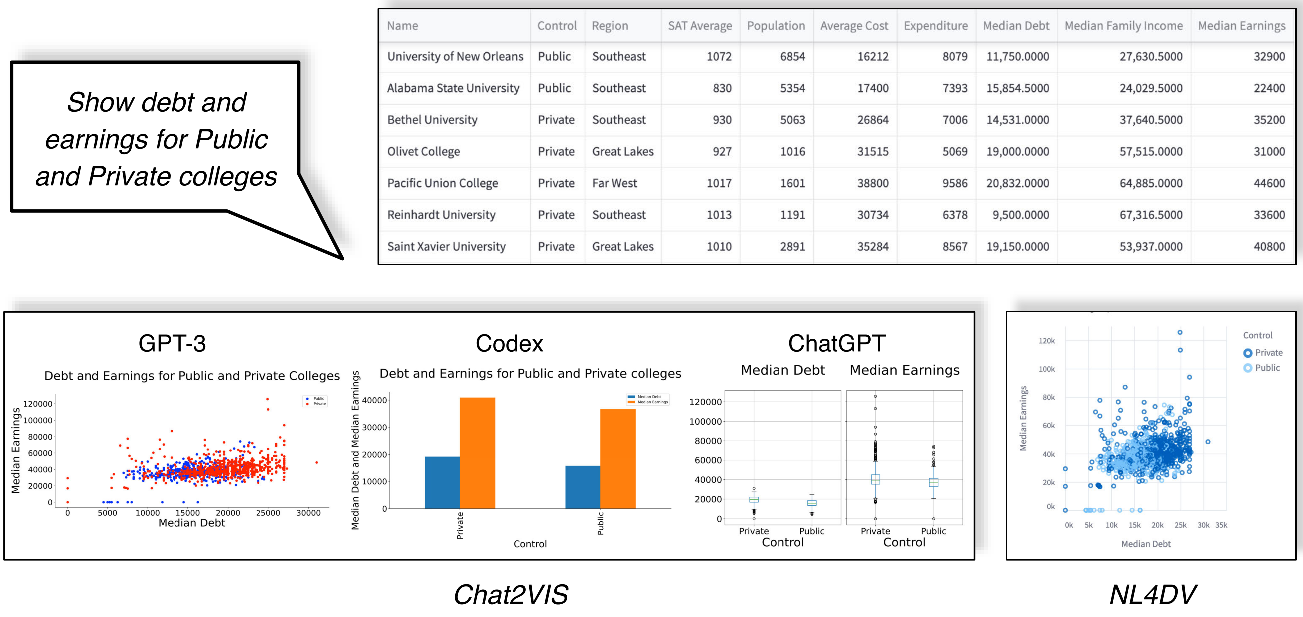
[9]https://github.com/nl4dv/nl4dv/blob/master/examples/assets/data/colleges.csv



What is the highest price of product, grouped by product type? Show a bar chart, and display by the names in desc.

product_id	product_type_code	product_name	product_price
1	Clothes	red jeans	734.7300
2	Clothes	yellow jeans	687.2300
3	Clothes	black jeans	695.1600
4	Clothes	blue jeans	939.5700

FIGURE 5. **Case Study 1: Chat2VIS results compared with nvBench using the department store dataset with query *"What is the highest price of product, grouped by product type? Show a bar chart, and display by the names in desc."***



Show debt and earnings for Public and Private colleges

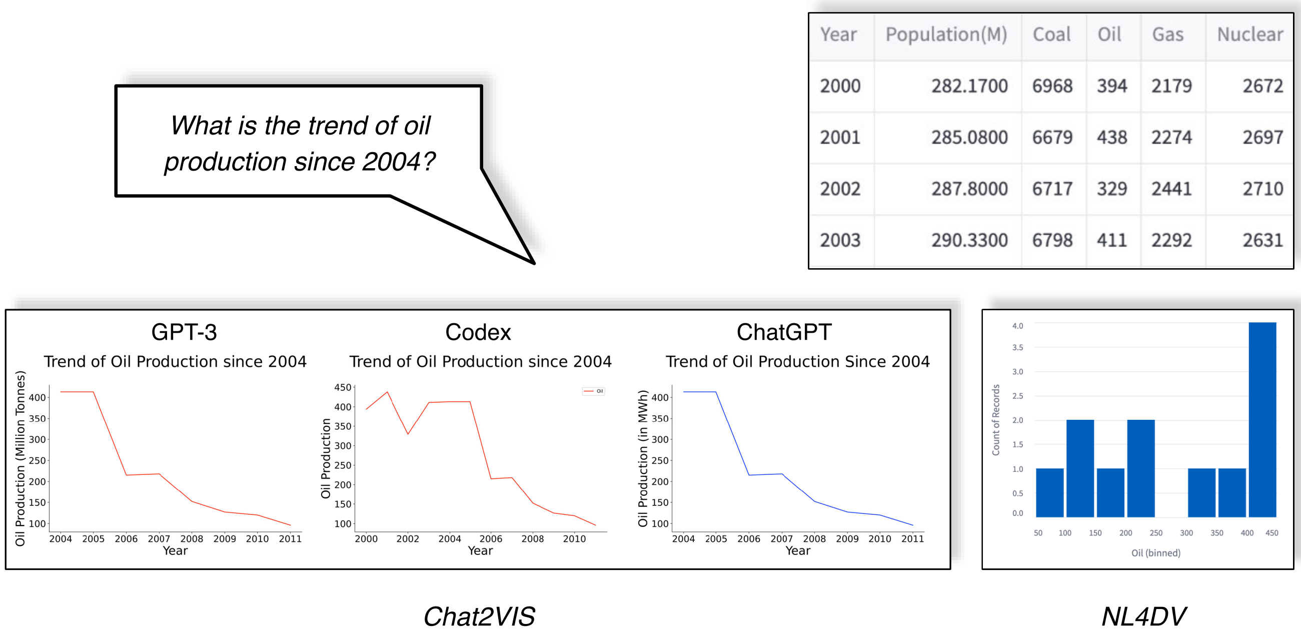
Name	Control	Region	SAT Average	Population	Average Cost	Expenditure	Median Debt	Median Family Income	Median Earnings
University of New Orleans	Public	Southeast	1072	6854	16212	8079	11,750.0000	27,630.5000	32900
Alabama State University	Public	Southeast	830	5354	17400	7393	15,854.5000	24,029.5000	22400
Bethel University	Private	Southeast	930	5063	26864	7006	14,531.0000	37,640.5000	35200
Olivet College	Private	Great Lakes	927	1016	31515	5069	19,000.0000	57,515.0000	31000
Pacific Union College	Private	Far West	1017	1601	38800	9586	20,832.0000	64,885.0000	44600
Reinhardt University	Private	Southeast	1013	1191	30734	6378	9,500.0000	67,316.5000	33600
Saint Xavier University	Private	Great Lakes	1010	2891	35284	8567	19,150.0000	53,937.0000	40800

FIGURE 6. **Case Study 2: Chat2VIS results compared with NL4DV using the Colleges Dataset with query *"Show debt and earnings for Public and Private colleges."***

the *customers_and_products_contacts* database,[10] which the nvBench example benchmark classifies as *Extra Hard*. The query is *"Show the number of products with price higher than 1000 or lower than 500 for each product name in a bar chart, and could you rank y-axis in descending order?"*.

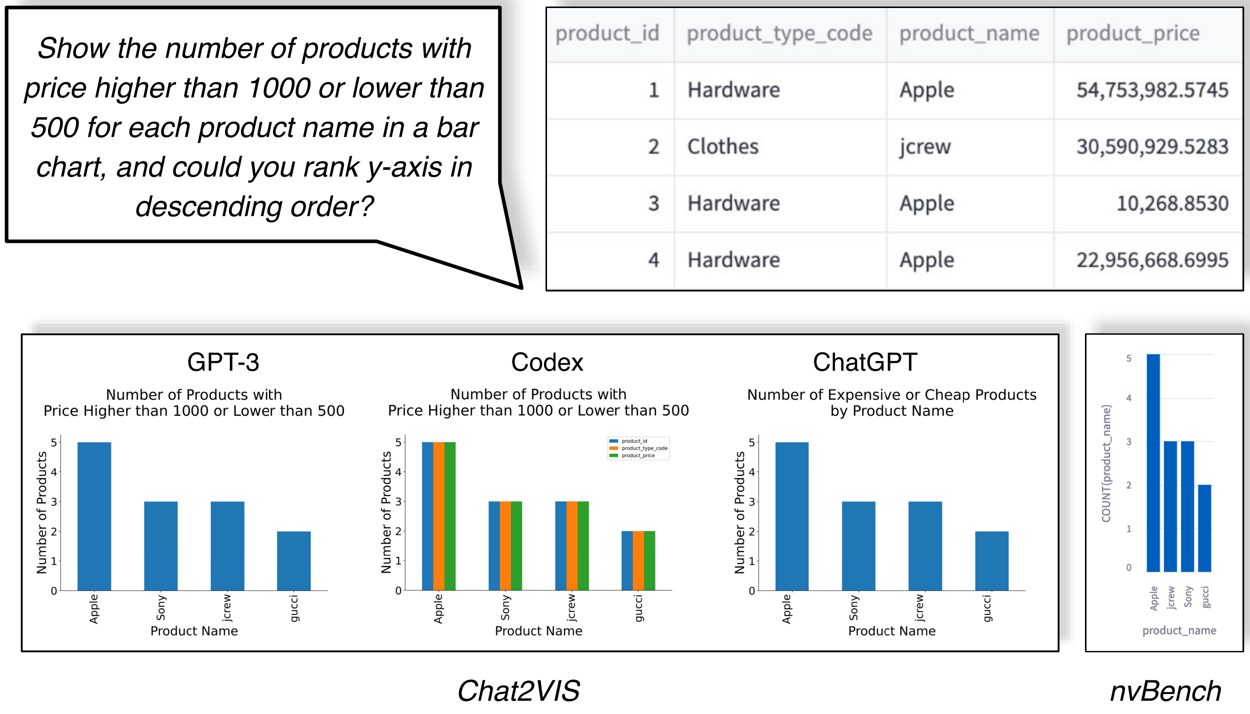
Fig. 8 shows all three models generate similar visualisations to the ground truth example specified in nvBench.

[10]https://github.com/TsinghuaDatabaseGroup/nvBench/blob/main/nvBench_VegaLite/VIS_299.html



Year	Population(M)	Coal	Oil	Gas	Nuclear
2000	282.1700	6968	394	2179	2672
2001	285.0800	6679	438	2274	2697
2002	287.8000	6717	329	2441	2710
2003	290.3300	6798	411	2292	2631

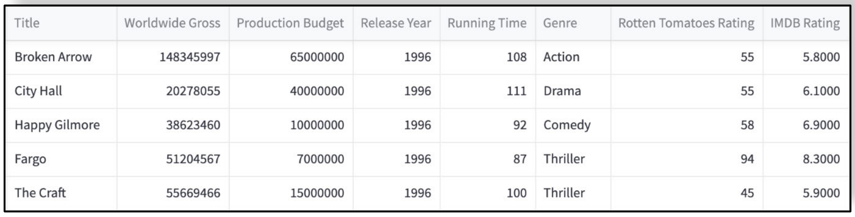
FIGURE 7. **Case Study 3: Chat2VIS results compared with NL4DV using the Energy Production Dataset with query *"What is the trend of oil production since 2004?"***



product_id	product_type_code	product_name	product_price
1	Hardware	Apple	54,753,982.5745
2	Clothes	jcrew	30,590,929.5283
3	Hardware	Apple	10,268.8530
4	Hardware	Apple	22,956,668.6995

FIGURE 8. **Case Study 4: Chat2VIS results compared with nvBench using the Customers and Products Contacts Dataset with query *"Show the number of products with price higher than 1000 or lower than 500 for each product name in a bar chart, and could you rank y-axis in descending order?"***

ChatGPT provides a slightly more informative title, conveying that products with a price higher than 1000 are "expensive" and those lower than 500 are "cheap". Despite Sony and & jcrew products swapped in comparison to nvBench, both have a value of 3 and are plotted accurately. The visualisation from Codex, while correct, is sub-optimal, requiring some further improvement in its coding structure to eliminate the multiple bar plotting. All three plots show more *draw the numbr of movie by gener*



Title	Worldwide Gross	Production Budget	Release Year	Running Time	Genre	Rotten Tomatoes Rating	IMDB Rating
Broken Arrow	148345997	65000000	1996	108	Action	55	5.8000
City Hall	20278055	40000000	1996	111	Drama	55	6.1000
Happy Gilmore	38623460	10000000	1996	92	Comedy	58	6.9000
Fargo	51204567	7000000	1996	87	Thriller	94	8.3000
The Craft	55669466	15000000	1996	100	Thriller	45	5.9000

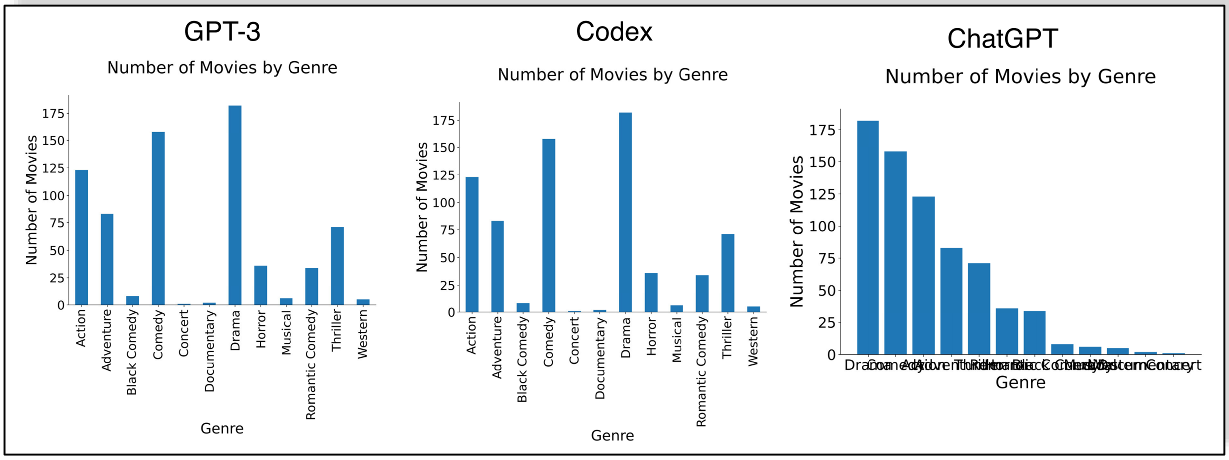


FIGURE 9. **Case Study 5: Chat2VIS plots depicting the output from the misspelt prompt *"draw the numbr of movie by gener"*.**

informative axis labels than their nvBench counterpart. The example demonstrates the high capability levels of Chat2VIS to handle challenging NL queries.

E. CASE STUDY 5: MISSPECIFIED PROMPTS

The following example illustrates the robustness of Chat2VIS to input errors that take the form of typographical mistakes. The movies[11] dataset is used here and contains information on movies released between 1996 and 2010, including details such as financials, ratings, and classifications. The ideal query in this example is *"Plot the number of movies by genre"* however, the system is prompted with *"draw the numbr of movie by gener"*.

The results are shown in Fig.9. The correct results across all three LLMs highlight the ability of the LLMs to interpret language even in the presence of multiple typographical errors and misspecification, thus emphasising their robustness. However, from the point of view of clarity of insights, the figure generated by ChatGPT is superior to those of the other models since it has decided to render the results as a rank-ordered bar graph in a descending order, while GPT-3 and Codex presented movies alphabetically.

[11]https://github.com/nl4dv/nl4dv/blob/master/examples/assets/data/movies-w-year.csv

F. CASE STUDY 6: UNDERSPECIFIED AND AMBIGUOUS PROMPTS

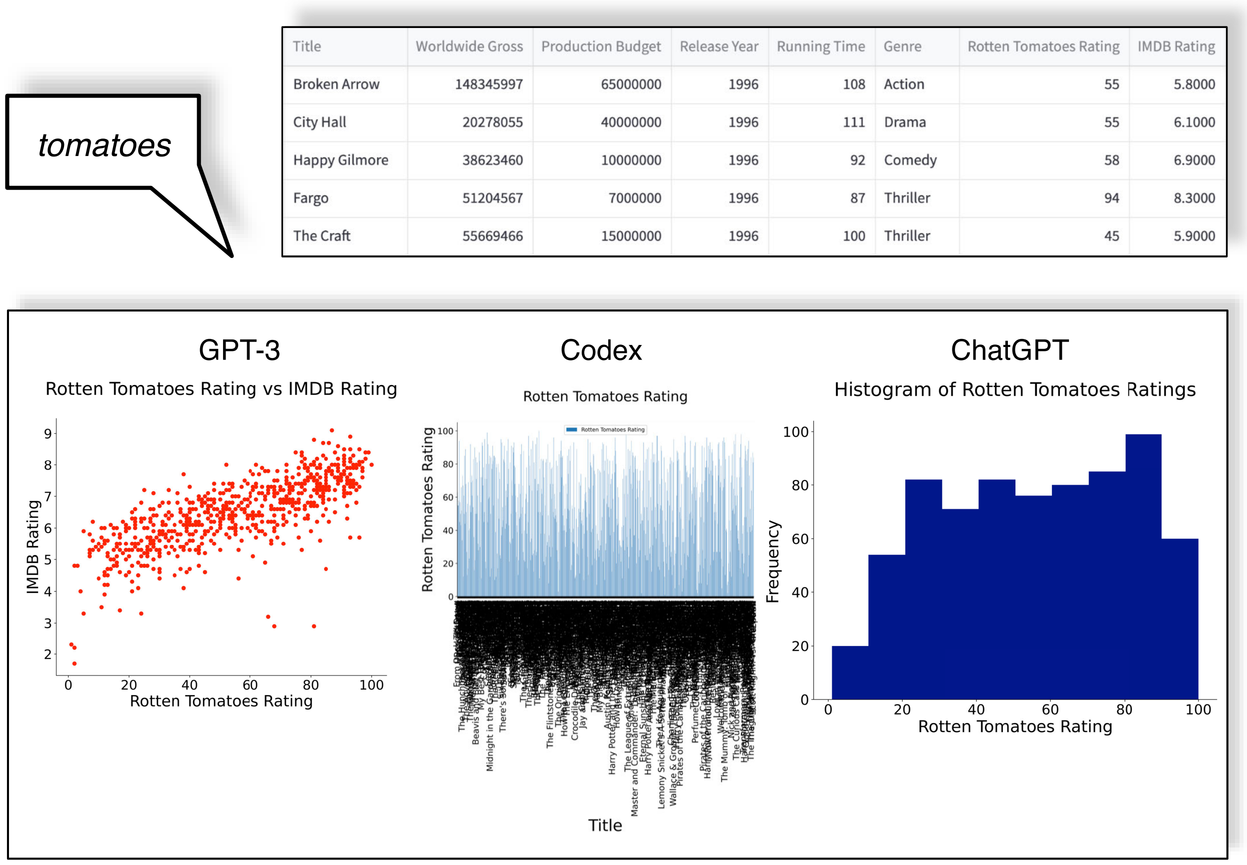
The movies dataset is again used in this example in order to demonstrate the inference capabilities of the LLMs together with the underlying priming prompts developed for Chat2VIS, to creatively make decisions based on extremely underspecified or ambiguous queries.

The test query used here is: *"tomatoes"*, which has an association with an existing column in the dataset called *Rotten Tomatoes Rating*. Fig. 10 demonstrates the results. Remarkably, the figures demonstrate that each LLM was able to make inferences and produce a figure that connected the results with the *Rotten Tomatoes Rating* despite a lack of direction.

GPT-3 plots the rating against the IMDb Rating column. It is uncertain whether this column is selected due to it being a rating column or simply because it is the next column in the dataset. Codex plots the rating for every title, producing an aesthetically unusable visualisation due to overcrowding, while ChatGPT produces a meaningful distribution plot of the ratings.

V. DISCUSSION

The experimental results from the six case studies in this work confirm that LLMs can effectively support the end-to-end



Title	Worldwide Gross	Production Budget	Release Year	Running Time	Genre	Rotten Tomatoes Rating	IMDB Rating
Broken Arrow	148345997	65000000	1996	108	Action	55	5.8000
City Hall	20278055	40000000	1996	111	Drama	55	6.1000
Happy Gilmore	38623460	10000000	1996	92	Comedy	58	6.9000
Fargo	51204567	7000000	1996	87	Thriller	94	8.3000
The Craft	55669466	15000000	1996	100	Thriller	45	5.9000

FIGURE 10. **Case Study 6: Chat2VIS plots depicting the output from the underspecified and ambiguous prompt *"tomatoes"*.**

generation of visualisations from NL when supported by well-engineered prompts, and are therefore, an ideal solution for the NL2VIS problem (RQ1). The proposed method exhibits a number of advantages over existing NL2VIS systems which rely on developing tailored symbolic NLP and deep learning approaches for the NL semantic analysis component. The primary one is efficiency since LLMs provide an end-to-end solution from language understanding to code generation. This results in reduced development expenses, but also, in higher accuracies. As such, LLMs offer a pre-trained and simplified solution to the most difficult problem of understanding NL, while also providing advanced capabilities for automated chart selection as well as the encapsulation of the code generation. The abilities of current LLMs raise questions about the necessity to continue to further explore symbolic NLP approaches for providing solutions to NL2VIS as a whole. Meanwhile, the proposed method demonstrated superior performances over existing methods, while showing robustness to misspecified and highly underspecified NL queries. Since the accuracy of LLMs will only continue to improve with time, they therefore represent the most viable solution for developing NL2VIS systems going forward.

To leverage this technology (RQ2), this study demonstrated the importance of prompt engineering in providing the LLMs with clear and concise NL requests. The study demonstrated how effective prompts can be engineered using Description and Code Prompt definitions that inform LLMs on the underlying data attributes as well as a coding guide. The results confirm that LLMs can be effectively primed with the proposed prompts, and it is the prompt engineering that helps elicit the correct chart selection and the appropriately rendered charts when accompanying the NL requests. The demonstration of the autonomous selection of correct plotting types by LLMs goes beyond the capabilities of prior studies.

In terms of performance (RQ3), the preliminary results indicate that the capability between ChatGPT, GPT-3 and Codex LLMs tend not to show large deviations. Arguably, some enhanced performance was exhibited by ChatGPT. The largely comparable performances can likely be attributable to the fact that the three LLMs were trained on similar datasets.

While the results demonstrate the potential of LLMs for NL2VIS, there are still some challenges to this technology (RQ4), mostly minor, and centering around aesthetic features of graphs and variations in visualisation results. These challenges are addressed individually below.

A. REMAINING CHALLENGES

While the proposed framework has performed impressively well and provided a viable solution to the problem of converting free-form NL directly into visualisation, some minor challenges still remain.

1) SETTING THE PLOT BACKGROUND COLOUR

Providing instruction within the engineered prompt to consistently and successfully generate code to change the background colour of a plot proved unsuccessful. With multiple factors at play, ranging from the type of plot generated and the Python container it is rendered within, through to software theme settings, no consistent approach was unearthed.

2) DISPLAY OF PLOT GRID LINES

The incorporation of grid lines into a plot can enhance its aesthetics and aid in its interpretation. However, generating precise code for the successful rendering of grid lines is often determined by the choice of plot type. As this is not specified within the engineered prompt, it is difficult to inform the LLM of the correct methods and function parameters to render the grid lines successfully. With thorough experimentation, the LLMs produced varying levels of success when the request for horizontal grid lines was included in the engineered prompt. Therefore, it is more favourable to request the adjustment of styling elements such as grid lines during the refinement of the NL query, and will be most successful when performed in conjunction with explicitly stating the type of plot to be rendered.

3) SPECIFYING COLOURING OF PLOT LINES AND ELEMENTS

Analogously to the problems encountered with rendering grid lines, specifying refinements to other plotting elements such as line colour proved challenging within the engineered prompt. With the dependence of some function and parameter values on plot type selection, LLMs on occasion attempted to invoke unsuitable functions or assign values to unknown parameters while endeavouring to style plot elements as requested. As with grid line refinement, styling of plot lines will be most successful when combined with plot type and included as an extension to the user query.

4) VARIABILITY IN PLOT GENERATION

Identical prompts to the same language model can result in significant variability in the type of plot generated and its features. This can make it difficult to consistently generate the desired visualisations. With their non-deterministic nature, especially of ChatGPT, is not possible to address this adequately at this stage since parameters that regulate the stochastic processes in its reasoning are not yet publicly available.

5) REFINING THE PROMPT FOR BEST RESULTS

The generic and verbose nature of the engineered prompt caters to all three LLM models, but experimentation has shown that the ideal prompt for each model can be varied slightly to achieve the best results. Once a specific LLM is selected, the prompt may be optimised and refined for generating the best visualisations.

Study Limitations and Future Work: The current study included a limited number of case studies consisting of a selection of NL queries and example visualisations. Ideally, a comprehensive evaluation of a system like Chat2VIS would include end-users and a qualitative assessment of the system's usefulness based on their feedback. While the evaluation of NLIs in data visualisation is a complex task in the context of end-user experience, it is the intention of the authors to expand this study and conduct this type of evaluation in the subsequent work.

Future work will explore the incorporation of the nvBench benchmark dataset into the refinement of the Chat2VIS capabilities and make use of the dataset for a more comprehensive quantitative analysis of its capabilities across a wider set of queries, thus enabling a more robust comparison against results from prior studies. Additionally, a valuable future research direction is the study of the effects of perturbations in the LLM prompts for this domain, quantifying the sensitivity in the changes of the quality of the outputs. Comparing the capabilities of a wider range of LLMs, including YoloPandas, to solve the NL2VIS problem is also a worthwhile undertaking; however, methods and benchmark datasets that support the automation of comparisons need to be further developed in this area in order to facilitate progress.

VI. CONCLUSION

The ability to generate visualisations based on natural language has been a long-standing goal in the field of data visualisation. The development of Natural Language Interfaces has paved the way for advancements in this area making data visualisation more accessible to a broader range of users by allowing them to express their queries and analysis intentions in natural language. However, the process of accurately and reliably translating natural language inputs into visualisations (NL2VIS) has been a challenging problem to solve due to the difficulty in understanding natural language.

This study proposed a novel end-to-end solution for converting free-form natural language into visualisations using state-of-the-art Large Language Models (LLMs). This study explored ChatGPT and its predecessors like GPT-3 and Codex for their ability to solve the task of understanding the queries and both auto-generating code while using their internal inference abilities for selecting the appropriate visualisation types. The proposed system, Chat2VIS, has demonstrated that the use of pre-trained LLMs together with well-engineered prompts, provides an efficient, reliable and accurate solution for the problem of NL2VIS. Chart-type selection is automatic, and the LLMs are able to understand vague user queries as well as those that are malformed. Moreover, the approach is also data-privacy preserving and security-aware, making it generalisable to all types of datasets.

The present study highlights the viability of LLMs to further the capabilities of existing NLIs for visualisation, providing a simpler pathway towards robust solutions which do not involve the task of defining grammars and customised domain-specific language models for language understanding. The results of this study provide valuable insights for researchers and practitioners in the field of data visualisation and NLIs, and offer simpler and more accurate solutions for making data and insights more accessible to a wider audience.

REFERENCES

[1] A. Narechania, A. Srinivasan, and J. Stasko, "NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 369–379, Feb. 2021.

[2] L. Shen, E. Shen, Y. Luo, X. Yang, X. Hu, X. Zhang, Z. Tai, and J. Wang, "Towards natural language interfaces for data visualization: A survey," 2021, *arXiv:2109.03506*.

[3] Y. Wang, Z. Hou, L. Shen, T. Wu, J. Wang, H. Huang, H. Zhang, and D. Zhang, "Towards natural language-based visualization authoring," *IEEE Trans. Vis. Comput. Graphics*, vol. 29, no. 1, pp. 1222–1232, Jan. 2022.

[4] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin, "Natural language to visualization by neural machine translation," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 1, pp. 217–226, Jan. 2022.

[5] Y. Song, X. Zhao, R. C.-W. Wong, and D. Jiang, "RGVisNet: A hybrid retrieval-generation neural framework towards automatic data visualization generation," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 1646–1655.

[6] Q. Wang, Z. Chen, Y. Wang, and H. Qu, "A survey on ML4VIS: Applying machine learning advances to data visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 12, pp. 5134–5153, Dec. 2022.

[7] T. B. Brown et al., "Language models are few-shot learners," 2020, *arXiv:2005.14165*.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[9] H. Voigt, M. Meuschke, K. Lawonn, and S. Zarrieß, "Challenges in designing natural language interfaces for complex visual models," in *Proc. 1st Workshop Bridging Hum.–Comput. Interact. Natural Lang. Process.*, 2021, pp. 66–73.

[10] Y. Luo, N. Tang, G. Li, C. Chai, W. Li, and X. Qin, "Synthesizing natural language to visualization (NL2VIS) benchmarks from NL2SQL benchmarks," in *Proc. Int. Conf. Manage. Data*, China, Jun. 2021, pp. 1235–1247.

[11] J. Tang, Y. Luo, M. Ouzzani, G. Li, and H. Chen, "Sevi: Speech-to-visualization through neural machine translation," in *Proc. Int. Conf. Manage. Data*, Jun. 2022, pp. 2353–2356.

[12] YoloPandas Developers. (2023). *YoloPandas. Python Package Index (PyPI).* [Online]. Available: https://pypi.org/project/yolopandas/

[13] G. Liu, X. Li, J. Wang, M. Sun, and P. Li, "Extracting knowledge from web text with Monte Carlo tree search," in *Proc. Web Conf.*, Apr. 2020, pp. 2585–2591.

[14] Y. Sun, J. Leigh, A. Johnson, and S. Lee, "Articulate: A semi-automated model for translating natural language queries into meaningful visualizations," in *Proc. 10th Int. Symp. Smart Graph. Smart Graph.*, Banff, AB, Canada: Springer, Jun. 2010, pp. 184–195.

[15] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios, "DataTone: Managing ambiguity in natural language interfaces for data visualization," in *Proc. 28th Annu. ACM Symp. User Interface Softw. Technol.*, Nov. 2015, pp. 489–500.

[16] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang, "Eviza: A natural language interface for visual analysis," in *Proc. 29th Annu. Symp. User Interface Softw. Technol.*, Oct. 2016, pp. 365–377.

[17] X. Qin, Y. Luo, N. Tang, and G. Li, "DeepEye: Visualizing your data by keyword search," in *Proc. EDBT*, 2018, pp. 441–444.

[18] B. Yu and C. T. Silva, "FlowSense: A natural language interface for visual data exploration within a dataflow system," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 1–11, Jan. 2020.

[19] E. Loper and S. Bird, "NLTK: The natural language toolkit," 2002, *arXiv:cs/0205028*.

[20] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics, Syst. Demonstrations*, 2014, pp. 55–60.

[21] C. Liu, Y. Han, R. Jiang, and X. Yuan, "ADVISor: Automatic visualization answer for natural-language question on tabular data," in *Proc. IEEE 14th Pacific Vis. Symp. (PacificVis)*, Apr. 2021, pp. 11–20.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2018, pp. 4171–4186.

[23] Y. Luo, J. Tang, and G. Li, "NvBench: A large-scale synthesized dataset for cross-domain natural language to visualization task," 2021, *arXiv:2112.12926*.

PAULA MADDIGAN received the B.Sc. degree (Hons.) in statistics and operations research from the Victoria University of Wellington Te Herenga Waka, New Zealand, and the Master of Analytics degree in business from Massey University, Auckland, New Zealand, in 2022. She is currently pursuing the Ph.D. degree in computer science. She has extensive industry experience as a Software Engineer and more recently held Research Assistant positions within the university.

TEO SUSNJAK received the Ph.D. degree in computer science with a focus on machine learning. He is currently the Subject Lead of Data Science and the Coordinator for the Master of Analytics degree with Massey University. He has broad industry experience, as a Machine Learning Analyst, and he continues to apply his expertise to solving real-world problems using machine learning approaches.