

# Chat2VIS: Generating Data Visualisations via Natural Language using ChatGPT, Codex and GPT-3 Large Language Models

Paula Maddigan<sup>\*</sup>and Teo Susnjak<sup>†</sup>

School of Mathematical and Computational Sciences  
Massey University  
Auckland, New Zealand

February 7, 2023

## Abstract

The field of data visualisation has long aimed to generate visualisations from natural language text. Research in Natural Language Interfaces (NLIs) has contributed towards the development of a solution which allow users to interact with data using natural language queries. However, the implementation of NLIs is challenging due to the inherent ambiguity of natural language and frequent underspecification of user queries. This study proposes using large language models (LLMs) such as ChatGPT and GPT-3 to convert free-form natural language directly into visualisations. This paper presents a novel system, Chat2VIS, which leverages LLMs and demonstrates how effective prompt engineering can be conducted in order to extract from LLMs the desired code for visualisations. Chat2VIS shows that the use of pre-trained LLMs together with the proposed priming prompts, offers an accurate and reliable approach to generating visualisations from natural language queries, even those that are highly misspecified and underspecified. This approach demonstrates gains in efficiency and cost reduction in the development of these systems, while attaining greater visualisation inference abilities compared to traditional NLP approaches that use hand-crafted grammar rules. The study compares the performance of two types of GPT-3 models and ChatGPT, while demonstrating that the proposed approach is secure, privacy-preserving, and generalisable to different datasets.

### **Keywords—**

ChatGPT; Codex, end-to-end visualisations from natural language, GPT-3, human-computer interaction, large language models, natural language interfaces, text-to-visualisation.

## 1 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 [10, 13]. 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 data visualisation paradigm can be difficult to learn for users [20] 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 [8] by making them more convenient and novice-friendly as well as effective, and ultimately, inclusive 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 [20] by allowing users to express their queries and analysis intentions in a form that is most natural to them [19].

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

---

<sup>\*</sup>P. Maddigan Email: Paula.Maddigan.1@uni.massey.ac.nz

<sup>†</sup>T. Susnjak Email: t.susnjak@massey.ac.nz

---

input query is analyzed and matched with the most appropriate visualisation, and then code is invoked to render the data. Each component is error-prone.

The implementation of NL2VIS is a particularly challenging task due to the inherent characteristics of NL, such as ambiguity and underspecification of requirements in the prompts. These characteristics of NL make it difficult to accurately interpret the user’s intent and generate appropriate visualisations with the existing technologies and approaches [14]. Despite these challenges, the popularity of NLIs for data visualisation has continued to grow, 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 [19] and their robustness in understanding NL. A recent comprehensive survey [19] 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 [1] 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 [17], which has demonstrated remarkable successes in tasks such as sentiment analysis, question-answering, and language generation owing both to the effectiveness of the architecture, but also through 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 importantly, to this study, the ability to generate code in response to NL requests. For these reasons, LLMs offer the potential of understanding free-form NL input and the capability to convert it into code that generates suitable visualisations.

## Contribution

This work 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 [1]. 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 an increase in development efficiency and a reduction in costs of such systems. Despite the capabilities of these models in displaying 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. Our experiments demonstrate the potential of the LLMs to enhance the performance of NL2VIS in terms of accuracy, efficiency, and cost reduction, thus potentially decreasing the need to devise new language models going forward. Our 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, we demonstrate how prompts for LLMs can be engineered to produce desired outputs. Furthermore, the system developed in this study has also been made publicly available through an online application for testing, with the ability for users to upload their datasets and generate visualisations<sup>1</sup>.

## 2 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 [18, 8]. 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 implemented NLIs that allow users to generate visualisations in response to NL queries or prompts [10, 8, 20, 7, 16] indicating the demand for this innovation. These are early 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 [20].

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

---

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

the emerging neural machine translation methods which rely on language models developed through deep-learning [7].

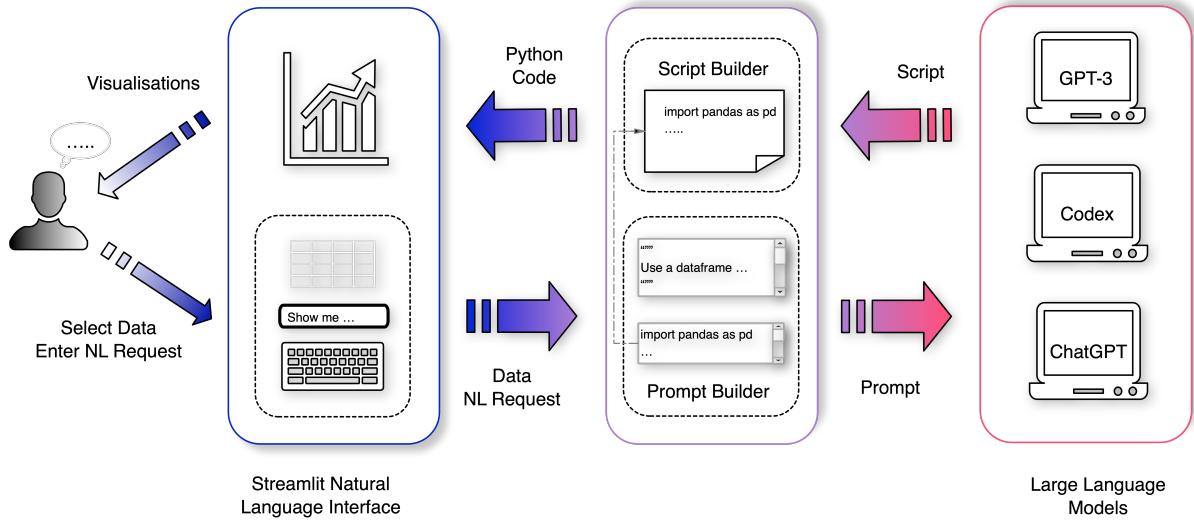


Figure 1: Chat2VIS Architecture

### 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 [5]. 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 [18]. 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 [10].

The majority of the earlier systems have been developed using these approaches [16], including Articulate [15], DataTone [3], Eviza [12], and Deep-Eye [11], 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 [10] and FlowSense [21] using symbolic NLP approaches have relied on semantic parsers such as NLTK [6], NER, and Stanford CoreNLP [9], 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.

### 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 work. These approaches aim to achieve greater robustness, flexibility and adaptability compared to traditional methods [18].

One notable system developed along these lines is ADVISor [4]. ADVISor initially transforms the NL query in table headers of dataset and as well as queries into vectors that represent semantic meaning using the large language transformer-based BERT [2] model. These converted vectors are then used as inputs to determine the necessary data that includes appropriate attributes and filter operations, as well as the type of aggregation if necessary. The visualisation type is then chosen based on a predefined rule-map.

ncNet [8] is a novel approach which also uses transformer-based models and visualisation-aware optimisations. ncNet is a machine learning model that is trained using a dataset called nvBench, 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 quantitative evaluation and user study and has shown promising accuracy in the nvBench benchmark. Recently, this system has been extended to include speech-to-visualisation capabilities [16].

Meanwhile, the RGVISNet [14] system decouples the NL2VIS process into two subtasks which consist of a hybrid retrieval and a generation framework named RGVISNet. The first part of the system retrieves the most relevant visualisation query from a large-scale visualisation codebase which serves as a candidate prototype which is refined in the next step by a GNN-based deep-learning model.

## Summary of Literature and Research Aims

The current trend in NL2VIS 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. The current paper aims to address this gap and examine the potential to simplify the NL2VIS pipeline, making it more robust for free-form NL and complex visualisation tasks, thus improving the accuracy and usability of these systems.

### Research questions

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

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

## 3 Methodology

In this study, we explored the ability of three OpenAI LLMs to generate Python scripts for visualising data based on NL queries. 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<sup>2</sup>.

The Davinci model family consists of billions of parameters, 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 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. 1 depicts the overview of the developed Chat2VIS system. A user entering a NL query via a Streamlit NLI, which is an open-source Python framework for web-based dashboards. The query is combined with a primer 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.

### 3.1 Natural Language Interface

The interface for the Chat2VIS software artefact used in this study is depicted in Fig. 2. The interface enables users to select a dataset and enter free-form text describing their 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 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.

---

<sup>2</sup>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>

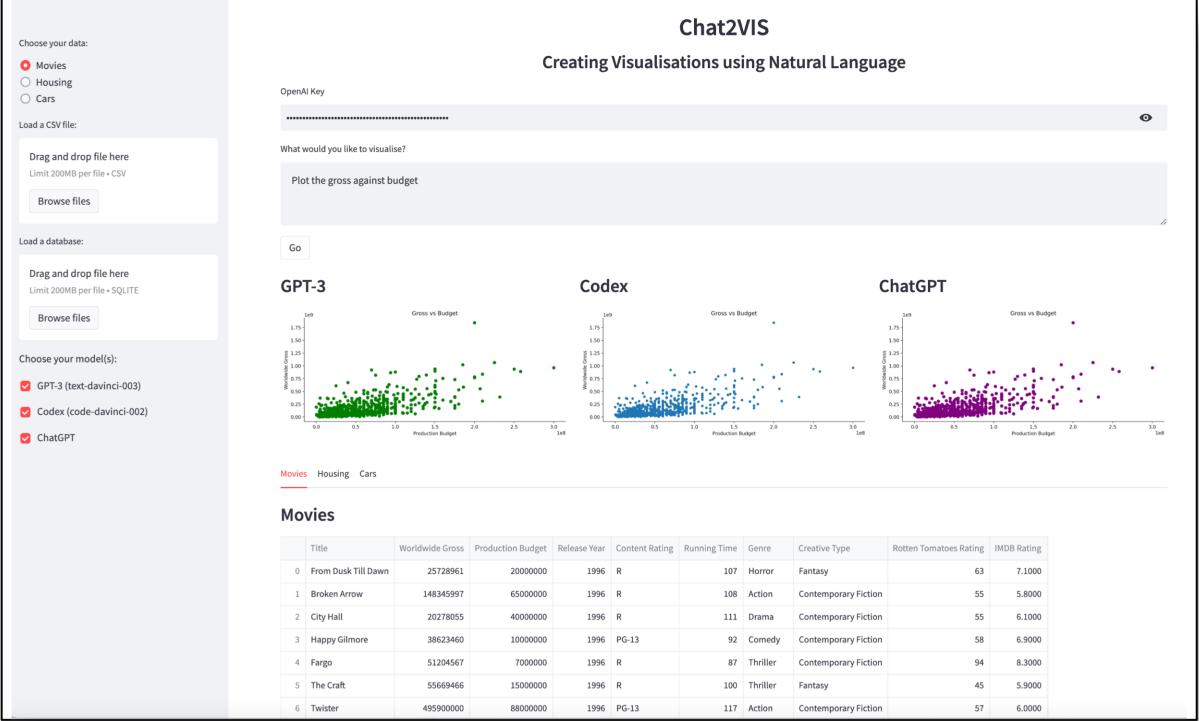


Figure 2: Streamlit Chat2VIS Natural Language Interface

### 3.2 Prompt Engineering

The most effective method for obtaining the desired output from the LLM is to use the "show-and-tell" technique<sup>3</sup> by supplying within the prompt an example together with instructions. The proposed system attempts to clearly generate a prompt primer consisting of two parts: (1) a Description Primer built from a Python docstring and declared using triple double quotes """ at the beginning and the end of the definition, (2) a Code Primer comprising of Python code statements that provide guidance and a starting point for the script.

The structure of the primer is presented in Fig. 3 and is described by way of an example<sup>4</sup> dataset, using the *products* table from the nvBench [7]<sup>5</sup> database *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 Primer and the Code Primer shown in Fig. 3 are depicted in components in order to aid their description. The bolded type highlights substitution values which are variable and dependent on the chosen dataset and thus specific to our example. These provide an overview of the dataframe 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. Each component is described as follows:

1. The Description Primer is initiated with a Python docstring Fig. 3(a)
2. In Fig. 3(b) we explicitly inform the LLM to use a dataframe with the name *df* which enables us to further refer 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 we 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*, we can effortlessly identify and eliminate this code if required before execution.

<sup>3</sup><https://beta.openai.com/docs/guides/completion/prompt-design>

<sup>4</sup>(NL,VIS) pair id=1129@x\_name@DESC in NVBench Benchmark

<sup>5</sup><https://sites.google.com/view/nvbench>

---

### Description Primer

- (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 Primer

- (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 3: Description and Code Primer

- 
3. The Description Primer in Fig. 3(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 primer. This listing could aid the LLM in identifying keywords for certain requests, for instance, prompts relating to keyboards or black jeans.
  4. In Fig. 3(d) we ask the LLM to decide on appropriate naming for the x and y axes, and plot title. In some cases, when working with grouped data, particularly box plots, the LLM adds a plot super-title unnecessarily. To address this, we recommend removing it by using positive instruction setting it to empty.
  5. To encourage correct syntax, we include the Python version as shown in Fig. 3(e).
  6. The Description Primer concludes with an instruction to create a graphing script with the supplied natural language prompt in Fig. 3(f).
  7. For this demonstration, the submitted NL prompt 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. 3(g) closes off the Description Primer
  9. The Code Primer begins with import statements for the required Python packages we would like the LLMs to use in Fig. 3(h).
  10. To foster uniformity in the plot layout, we ask for a single subplot with a fixed figure size in Fig. 3(i) in an attempt to render equivalently sized plots on the interface.
  11. By assigning a copy of the named dataframe in Fig. 3(j) to the variable *df* in the Code Primer, ensures consistency within the script and enables the original dataframe to be retained for further querying.

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

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()

Unspecified parameters remain at their default values. With many variations to Python scripting, we set the model temperature 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. We 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.

### 3.3 Script Refinement and Rendering

On the return of the script from the API for each model, the Code Primer 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 expand their NL prompt, including stating the type of chart, plot colours, labels etc.

### 3.4 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 selecting appropriate charting elements is demonstrated over six example case studies. These cover five datasets. Four of these case studies are reproduced from examples in existing literature, comparing our visualisations with those from prior studies. In the remaining two

case studies, we test the capabilities of the LLMs and the priming 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 reproducible from the online web app. However the non-deterministic nature of the LLMs leads to variability in plot generation even when an identical prompt is resubmitted.

## 4 Results

### 4.1 Case Study 1: Department Store Dataset

The first example is based on the *products* table used above in the description of the prompt engineering, originating 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 classed by nvBench as an *easy* visualisation. Fig. 4 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 3 LLMs provide more informative *x* and *y* axis labelling and titles than their nvBench counterpart.

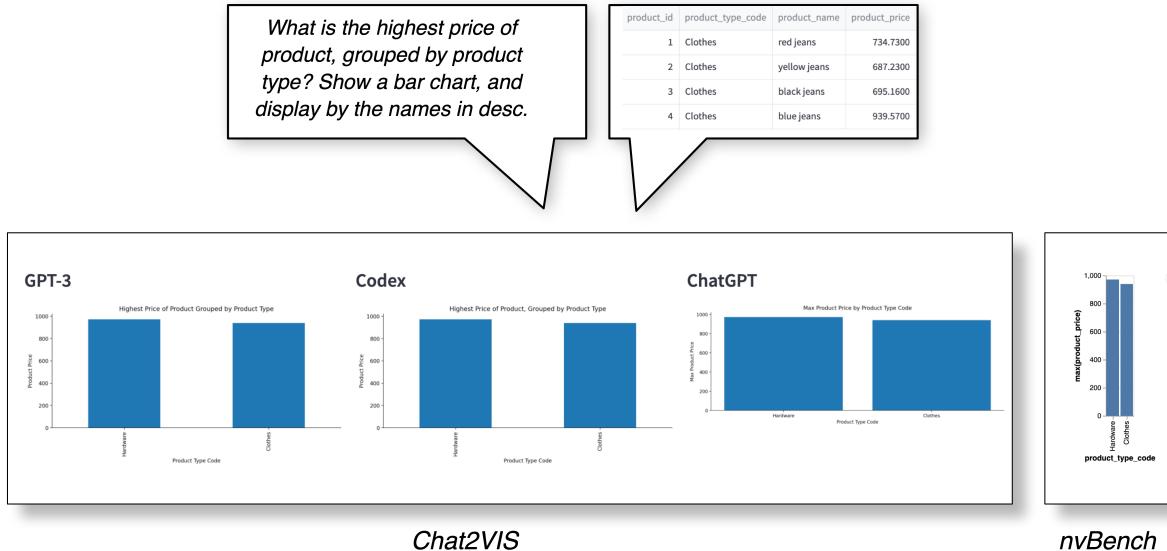


Figure 4: Case Study 1: Department Store Dataset

### 4.2 Case Study 2: Colleges Dataset

The *colleges* dataset was used by Narechania et al. [10] to demonstrate their Vega-Lite editor for rendering queries visualised by NL4DV<sup>6</sup>. Their output is compared with our Chat2VIS results. The dataset holds information on students and staff at U.S. public and private colleges. 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.

Fig. 5 shows GPT-3 produces a scatter plot similar to those produced by NL4DV Narechania et al. [10], plotting median debt against earnings for all data points by college type. Codex interpreted the query by plotting the mean value of the debt and earnings columns as a bar chart for each college type. ChatGPT renders a box plot for each of debt and earnings figures for public and private colleges thus focusing on depicting the distributions. The LLMs successfully make the distinction between public and private colleges using the *control* column, not immediately recognisable as a college type by naming standards. By supplying categorical values in the prompt, we inform the model of which column holds the public/private indicator to correctly categorise the data.

<sup>6</sup><https://nl4dv.github.io/nl4dv/documentation.html>

Additional analysis by the models is necessary as the NL prompt mentions debt and earnings of colleges, with all models accurately interpreting this through the use of the *Median Debt* and *Median Earnings* columns.

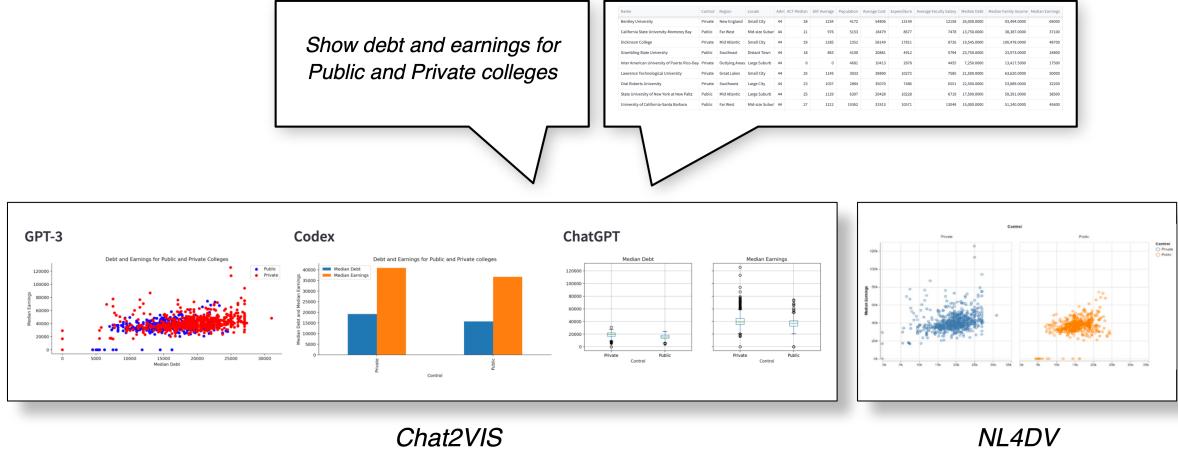


Figure 5: Case Study 2: Colleges Dataset

### 4.3 Case Study 3: Energy Production Dataset

The next set of visualisations are depicted on the Energy Production dataset described in the ADVISor [4] 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.

The test NL prompt used in the ADVISor [4] study is "*What is the trend of oil production since 2004?*". The results from Chat2VIS are shown in Fig. 6 together with the output from NL4DV<sup>7</sup> and can be compared with that from ADVISor [4]. 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 [4]. The plot within the ADVISor [4] study 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.

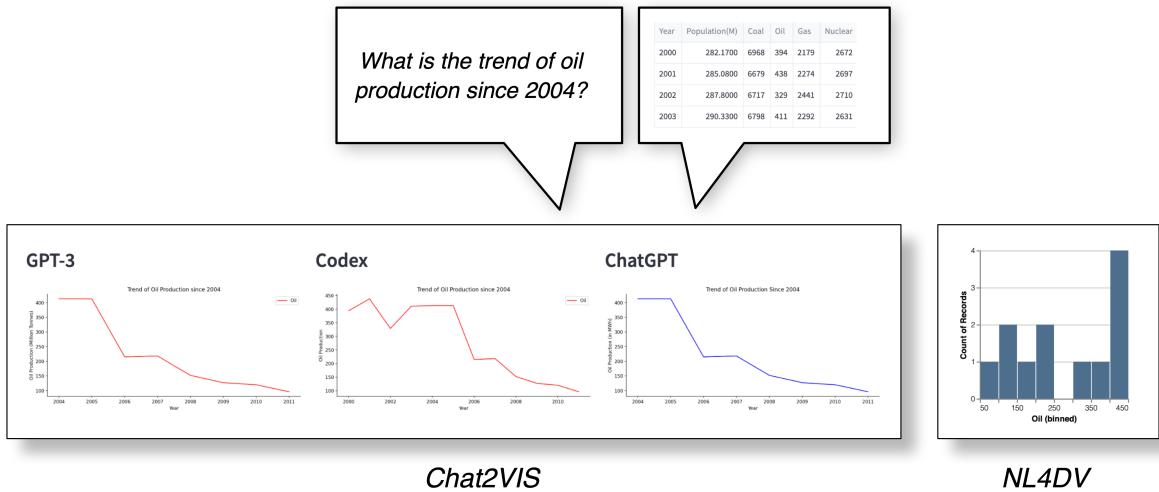


Figure 6: Case Study 3: Energy Production Dataset

<sup>7</sup><https://nl4dv.github.io/nl4dv/documentation.html>

## 4.4 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 the *customers\_and\_products\_contacts* database, which the nvBench [7]<sup>8</sup> example<sup>9</sup> 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. 7 shows all three models generate similar visualisations to the ground truth example specified in nvBench. 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 informative axis labels than their nvBench counterpart. The example demonstrates the high capability levels of Chat2VIS to handle challenging NL queries.

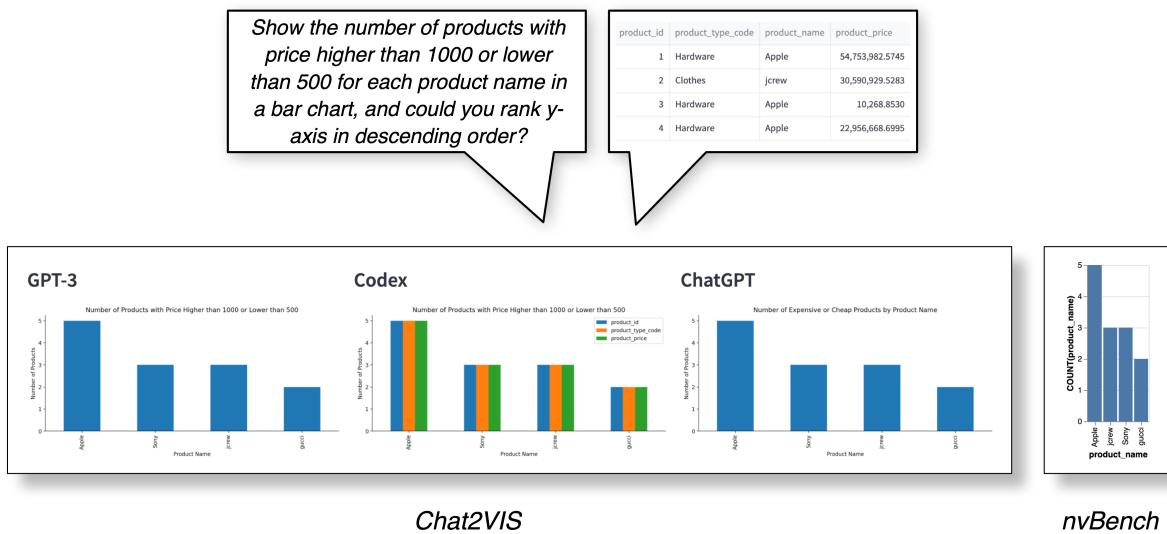


Figure 7: Case Study 4: Customers and Products Contacts Dataset

## 4.5 Case Study 5: Misspecified Prompts

The following example illustrates the robustness of Chat2VIS to input errors that take the form of typographical mistakes. This dataset used here is from IMDb<sup>10</sup> 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.8. 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.

## 4.6 Case Study 6: Underspecified and Ambiguous Prompts

The IMDb 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. 9 demonstrates the results. Remarkably, the figures demonstrate

<sup>8</sup><https://sites.google.com/view/nvbench>

<sup>9</sup>(NL,VIS) pair id=1009@y\_name@DESC in NVBench Benchmark

<sup>10</sup><https://www.imdb.com/interfaces/>

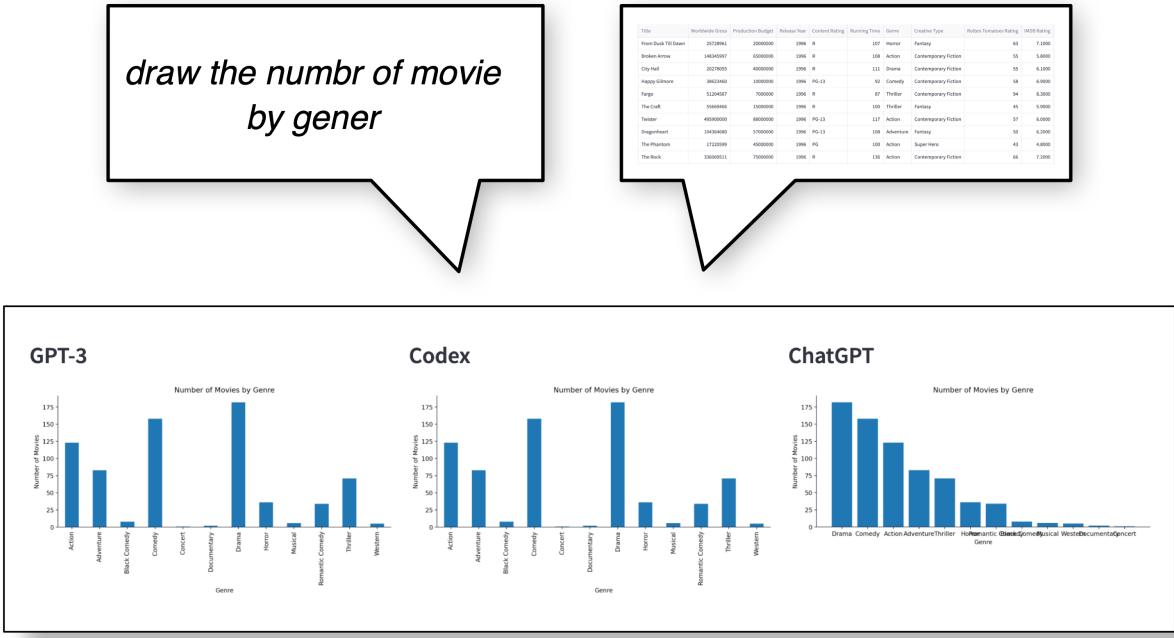


Figure 8: Case Study 5: Misspelled Prompts

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 IMBD 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.

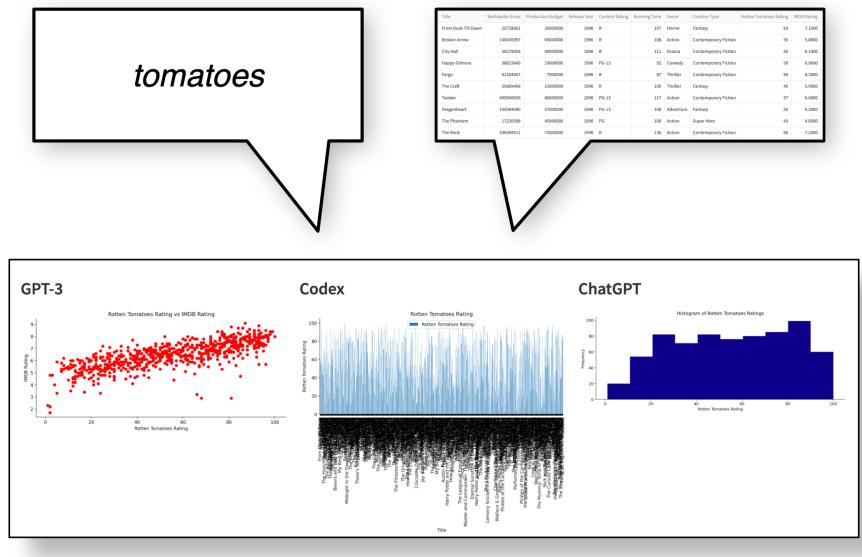


Figure 9: Case Study 6: Underspecified and Ambiguous Prompts

---

## 5 Discussion

The experimental results from the 6 case studies in this work confirm that LLMs can effectively support the end-to-end generation of visualisations from NL when supported by additional prompts, and are therefore, a viable solution for the NL2VIS problem (RQ1). The proposed method exhibits a number of advantages over existing NL2VIS systems which rely on symbolic NLP and deep learning approaches. The primary ones are efficiency, cost savings and accuracy. LLMs offer a pre-trained and simplified solution to the most difficult problem of understanding NL, while also encapsulating the code generation component as well. As such, this approach removes the need to create hand-crafted symbolic NLP solutions, thus saving resources. The abilities of current LLMs raise questions about the necessity to further explore symbolic NLP approaches for providing solutions to NL2VIS as a whole. Meanwhile, the proposed method demonstrated comparable if not indications of superior performances with existing methods, while showing robustness to misspecified and underspecified NL requests. The accuracy of LLMs will only continue to improve with time and therefore they represent the most viable solution for developing NL2VIS system 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 Primer definitions that indicate to the LLMs attributes of the underlying data as well as a coding guide. The results confirm that LLMs can be effectively primed with the proposed prompts to elicit the generation of appropriately rendered charts when accompanying the NL requests.

In terms of performance (RQ3), the preliminary results indicate that the capability of the three LLMs tends not to show large deviations. This is likely due to the fact that the 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 centring around aesthetic features of graphs and variable results. These challenges are addressed individually below.

### 5.1 Remaining challenges

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

**Setting the Plot Background Colour:** Changing the background colour of a plot using natural language prompts is a more challenging task, and results can depend on plot type and hardware/software theme settings. The natural language instructions are often misinterpreted, leading to unintended outcomes. For this reason we exclude these settings in the primer.

**Varying Understanding of Horizontal Grid Lines:** A further challenge with converting natural language into visualisations is the varying understanding of the three models regarding the rendering of grid lines in a plot. Experimentation with both natural language prompts and code prompts has not produced a consistent result. As a result, it is necessary to set the grid line preferences within the natural language prompt, and results may depend on the model being used.

**Specifying Colouring of Lines and Plots:** Suggesting plotting colours for lines and bars proved difficult as different style plots required small variations in function parameters to correctly set the element colours. This inconsistency, with sometimes colour arguments supplied to functions that are not recognised, caused execution issues on rendering the plot. Hence it is left to the users to specify more specifically what elements within the plot they require colour changes to as part of their prompt.

**Variability in Plot Generation:** The repetition of the same prompt to the 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. The non-deterministic nature especially of ChatGPT, is difficult to address at this stage since parameters that regulate the stochastic processes in its reasoning are not yet available.

**Refining the Prompt for Best Results:** The generic and verbose nature of the natural language prompt caters to all three models, but experimentation has shown that the ideal prompt for each model can vary slightly. To achieve the best results, the prompt may need to be refined when targeting a specific model for visualisation. Fine-tuning the model may also be necessary in some cases.

---

## Study limitations and future work

The current study included only 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. 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 conduct this in the subsequent work.

Future work will also explore the incorporation of the nvBench benchmark dataset into the refinement of the Chat2VIS capabilities and use the dataset for a more comprehensive quantitative analysis of its capabilities.

## 6 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 NL inputs into visualisations (NL2VIS) has been a challenging problem to solve due to the inherent ambiguity in NL and frequent underspecification by users.

This study proposed an end-to-end solution for converting free-form natural language into visualisations using state-of-the-art Large Language Models (LLMs), specifically ChatGPT and its predecessors like GPT-3 which has been previously unexplored. The proposed system, Chat2VIS, has demonstrated that the use of pre-trained LLMs together with well-engineered prompts, for this task offers not only accuracy gains in understanding user requests but also an increase in development efficiency and a reduction in costs.

The study examined the capability and comparative performance of two types of GPT-3 models and ChatGPT for NL2VIS tasks which also included the automatic selection of chart types from the user queries. The study demonstrated the potential of LLMs to provide a viable solution for NL2VIS, that exceeds the previous state-of-the-art solutions in terms of capability, flexibility and complexity in implementation. 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 languages 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 a promising solution for making data and insights more accessible to a wider audience.

## References

- [1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, A. Holtzman, J. Feng, J. Gao, X. Liu, J. Stokes, L. Zettlemoyer, D. Amodei, and C. Hesse. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.
- [2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 4171–4186, 2018.
- [3] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th annual acm symposium on user interface software & technology*, pages 489–500, 2015.
- [4] C. Liu, Y. Han, R. Jiang, and X. Yuan. Advisor: Automatic visualization answer for natural-language question on tabular data. In *2021 IEEE 14th Pacific Visualization Symposium (PacificVis)*, pages 11–20. IEEE, 2021.
- [5] G. Liu, X. Li, J. Wang, M. Sun, and P. Li. Extracting knowledge from web text with monte carlo tree search. In *Proceedings of The Web Conference 2020*, pages 2585–2591, 2020.
- [6] E. Loper and S. Bird. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

- 
- [7] Y. Luo, N. Tang, G. Li, C. Chai, W. Li, and X. Qin. Synthesizing natural language to visualization (nl2vis) benchmarks from nl2sql benchmarks. In *Proceedings of the 2021 International Conference on Management of Data, SIGMOD Conference 2021, June 20–25, 2021, Virtual Event, China*. ACM, 2021.
  - [8] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin. Natural language to visualization by neural machine translation. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):217–226, 2021.
  - [9] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.
  - [10] A. Narechania, A. Srinivasan, and J. Stasko. Nl4dv: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):369–379, 2020.
  - [11] X. Qin, Y. Luo, N. Tang, and G. Li. Deepeye: Visualizing your data by keyword search. In *EDBT*, pages 441–444, 2018.
  - [12] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang. Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th annual symposium on user interface software and technology*, pages 365–377, 2016.
  - [13] 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. *arXiv preprint arXiv:2109.03506*, 2021.
  - [14] Y. Song, X. Zhao, R. C.-W. Wong, and D. Jiang. Rgvisnet: A hybrid retrieval-generation neural framework towards automatic data visualization generation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1646–1655, 2022.
  - [15] Y. Sun, J. Leigh, A. Johnson, and S. Lee. Articulate: A semi-automated model for translating natural language queries into meaningful visualizations. In *Smart Graphics: 10th International Symposium on Smart Graphics, Banff, Canada, June 24-26, 2010 Proceedings 10*, pages 184–195. Springer, 2010.
  - [16] J. Tang, Y. Luo, M. Ouzzani, G. Li, and H. Chen. Sevi: Speech-to-visualization through neural machine translation. In *Proceedings of the 2022 International Conference on Management of Data*, pages 2353–2356, 2022.
  - [17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.
  - [18] H. Voigt, M. Meuschke, K. Lawonn, and S. Zarrieß. Challenges in designing natural language interfaces for complex visual models. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, pages 66–73, 2021.
  - [19] Q. Wang, Z. Chen, Y. Wang, and H. Qu. A survey on ml4vis: Applying machinelearning advances to data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2021.
  - [20] Y. Wang, Z. Hou, L. Shen, T. Wu, J. Wang, H. Huang, H. Zhang, and D. Zhang. Towards natural language-based visualization authoring. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1222–1232, 2022.
  - [21] B. Yu and C. T. Silva. Flowsense: A natural language interface for visual data exploration within a dataflow system. *IEEE transactions on visualization and computer graphics*, 26(1):1–11, 2019.