Do LLMs Generate Creative and Visually Accessible Data visualisations?

Clarissa Miranda-Pena[1*], Andrew Reeson[2], Cécile Paris[2], Josiah Poon[1], Jonathan K. Kummerfeld[1]
The University of Sydney[1], CSIRO's Data61[2],
amir0532@uni.sydney.edu.au*

Abstract

Data visualisation is a valuable task that combines careful data processing with creative design. Large Language Models (LLMs) are now capable of responding to a data visualisation request in natural language with code that generates accurate data visualisations (e.g., using Matplotlib), but what about human-centered factors, such as the creativity and accessibility of the data visualisations? In this work, we study human perceptions of creativity in the data visualisations generated by LLMs, and propose metrics for accessibility. We generate a range of visualisations using GPT-4 and Claude-2 with controlled variations in prompt and inference parameters, to encourage the generation of different types of data visualisations for the same data. Subsets of these data visualisations are presented to people in a survey with questions that probe human perceptions of different aspects of creativity and accessibility. We find that the models produce visualisations that are novel, but not surprising. Our results also show that our accessibility metrics are consistent with human judgements. In all respects, the LLMs underperform visualisations produced by human-written code. To go beyond the simplest requests, these models need to become aware of human-centered factors, while maintaining accuracy.

1 Introduction

When evaluating AI systems, we typically focus on accuracy. However, generative AI systems, such as language models, are being applied to tasks where other, human-centered, factors are important too. An output can be accurate, but not accessible, e.g., if the colours chosen make a data visualisation hard to read, or there is not enough space between labels, see Figure 1. Similarly, an output can be accurate, but not creative, e.g., if the data visualisation always has a linear scale, when in some cases a log-scale would reveal additional patterns.

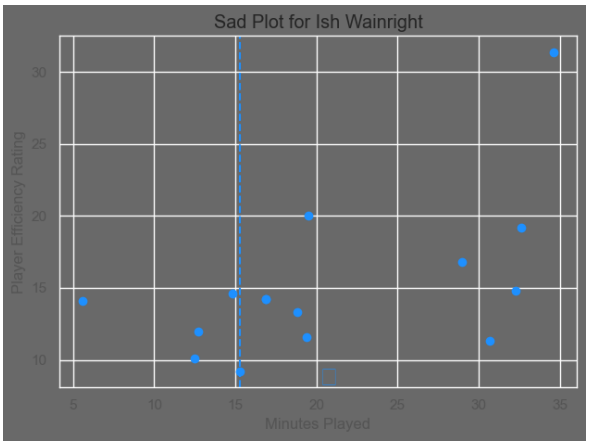


Figure 1: Example of a Claude-2 generated visualisation with a low score in accessibility.

Creativity is present in a range of human activities, from structured goal-oriented tasks like writing code or creating recipes (Noever and Noever, 2023), to more open-ended tasks like writing a story (Kim et al., 2023; Chakrabarty et al., 2023) or painting (Liu and Chilton, 2022). In the context of data visualisation, a creative visualisation presents the data in an unexpected way that more effectively communicates the data to the viewer (Wang, 2023).

Creativity can be defined in terms of value, novelty and surprise (Boden, 2010). Even though LLMs can produce valuable artifacts, achieving novelty and surprise is still a challenge (Franceschelli and Musolesi, 2023). Recent studies concluded that at the individual level, systems are better than some people. However, at the collective level, systems tend to produce homogenous outputs (Anderson et al., 2024; Doshi and Hauser, 2024), which raises concern about the potential impact on creativity when these tools are used by people.

Accessibility is another critical human-centered aspect of various tasks. It is a key part of inclusive design, which aims to make tasks available to everyone (Gilbert, 2019). For data visualisation,

Proceedings of The 22nd Annual Workshop of the Australasian Language Technology Association, pages 12–29

there are many potential pitfalls, such as colours that are hard to distinguish, or text that is difficult to read. Most tasks with LLMs do not have to consider accessibility directly, as the output is text, and accessibility is then the concern of the text-rendering system. Data visualisation is an interesting exception, where accessibility is crucial, and (unlike tasks like image generation), it may be measureable.

This work investigates the creativity and accessibility of LLM generated data visualisations through a human study conducted with 57 people. We show people outputs from two LLMs and examples from the documentation of libraries for data visualisation. The questions probe the notion of creativity in several ways, with absolute judgements (e.g., asking if any data visualisation was surprising) and relative judgements (e.g., selecting the best data visualisation from a small set). We apply several standard approaches to encourage greater LLM creativity, including demonstrative prompts (Issak and Varshney, 2023), e.g., "using your imagination", and variation in configuration hyperparameters, e.g., different temperature values. For accessibility, we define two new metrics, one focused on the spacing of text and the other focused on color choices. Our metrics are automatic, and we use questions in our study to verify their consistency with human perception.

We find that the LLMs can generate data visualisations that are novel, but not surprising. Our visual accessibility metrics are consistent with human perception, indicating that they can be used in future work. Applying the metrics to a large sample, we see that LLM outputs span a far wider range of scores than human created data visualisations do. Do LLMs Generate Creative and Visually Accessible Data visualisations? No, while the data visualisations being produced today are effective for simple tasks, there is scope for improvement in creativity, which must occur without sacrificing accessibility or accuracy.

2 Related work

Metrics for code generation Existing work has evaluated the correctness of programs generated in response to a natural language query. For example, Finegan-Dollak et al. (2018) proposed variations in evaluation of text-to-SQL, and Yin et al. (2018) considered more general programming questions. In both of these cases, there are multiple solutions included in the dataset, but they are only considered in the evaluation for measuring accuracy. In the former case, the results of executing the code are also considered, partly because the authors point out that there are multiple correct solutions. Measuring partial matching of code is similar to measuring partial matches in tasks such as machine translation. Metrics like BLEU and BERTScore have been adapted to code, e.g., in CodeBERTScore (Zhou et al., 2023), which was more accurate than prior metrics on the CoNaLa dataset (Yin et al., 2018). In all of these cases, the focus is on accuracy, rather than the additional human-centred factors we focus on here. Prior work has considered creativity in code (Colton et al., 2018), arguing as we do that they are more than just a task-solving process. However, their focus was on the code itself, whereas we are also interested in the creativity of the output it generates.

Metrics for creativity Looking beyond code, there has been some work considering creativity in the output of generative models. Berns and Colton (2020) considered image generation, arguing that standard loss functions for these models encourage them to produce "more of the same", rather than more unusual out-of-distribution outputs. They point to prior work in computational creativity measurement as a potential avenue for guiding model development. Some have argued that for a system to be creative, it should integrate creativity in the process of self-exploration and self-modification (Cook et al., 2013). We do not subscribe to this view. Instead, we see creativity in output as a property that can be judged by humans, regardless of the process that generated it. In the case of LLMs, inherently creative domains, such as recipes, may seem like a promising space for creativity, but in practise, researchers have needed to set low temperatures in order to achieve consistency between ingredients and instructions (Noever and Noever, 2023). One example of an effort to measure creativity is DeepCreativity (Franceschelli and Musolesi, 2022). The system weights three factors of creativity: value, novelty, and surprise. Valuable is a binary label judged by a trained model. Novelty is the Euclidean distance between a vector representing style and one of typical values. Surprise is the difference between the prior and posterior distribution of a sequential predictive model. This approach is effective at modeling creativity in poetry over time, but the use of a sequential model

means there is a strong assumption of time-based variation, and it is unclear how to generalise their methods to the code setting.

Human evaluation for creative tasks We study creativity and calibrate our accessibility metrics by conducting a study in which people judge data visualisations. Human evaluation is often a critical part of evaluating human-centred factors like creativity. He et al. (2023) evaluated open-ended text generation from the WikiText-103 dataset using contextualized embedding metrics such as MAUVE. By comparing automatic and human judgements from two annotators, they identified a range of issues with automatic metrics, emphasising the importance of human evaluation. Chakrabarty et al. (2023) measure creativity using the Consensing Assessment Technique and propose the Torrance Test of Creativity Writing (TTCW). Ten experts rated human and AI stories considering fluency, flexibility, originality, and elaboration in writing. They found that 84% of human stories passed the rubric, while only 9% passed for GPT-4, and 30% for Claude. Like He et al. (2023), they found disagreements between human judgements and automatic metrics. Outside of text, humans have also been used to evaluate a range of other creative tasks. For example, Mechanic Miner (Cook et al., 2013) is a game generation system, which was evaluated by getting over 5,933 people to play generated levels and rate the enjoyment and difficulty of the level. All of this past work supports the idea that human evaluation is critical in creativity judgement.

Accessibility evaluation in images with text Venues such as ASSETS (ACM SIGACCESS Conference on Computers and Accessibility) include extensive work on accessibility in a range of applications. The closest work to our own is on measuring accessibility of websites. In particular, tools have been developed to check if sites meet the Web Content Accessibility Guidelines (WCAG) (Alba et al., 2022; Yang et al., 2021; Hadadi, 2021; NC State University, 2014), or other guidelines, such as Google's material design guidelines (Yang et al., 2021; Google, 2021). Some of these work with UI design mockups and screenshots, while others are focused on html. The WCAG does include recommendations related to non-text content (ie., images), but focusing on the use of tags to provide text alternatives to the image. We are not aware of comparable work on automatic metrics specifically for accessibility of data visualisations.

3 Experiments

This work has three key components: (1) creating examples of LLM generated data visualisations, (2) writing metrics for accessibility, and (3) a human study[1] in which we collect judgements of creativity and accessibility.

3.1 Data

We consider two sources of data visualisations. First, a set created by people, sourced from documentation. Second, a set generated by LLMs, produced by prompting.

3.1.1 Human-written code

We use 83 samples from documentation. These come from matplotlib's quick start guide and seaborn's example gallery (Hunter, 2007; Waskom, 2021). We chose these sources because they show very common use cases of these libraries, often with the default configuration, and are probably widely used with little adjustment. At the same time, they are not highly polished/perfected examples of the ideal way to represent data. In each case, we adapt the code sligtly, just in order to use the same data we provide to the LLMs.

3.1.2 LLM-generated code

For GPT-4 and Claude-2 we made 840 queries as a result of a combination of varying the prompt, the data, and hyperparameters. These variations are described below. Responses with minor syntax errors or missing library imports were manually fixed. In 23 cases, the models refused to generate the code given the prompt. We use a sample of the data visualisations generated for our survey, and all of them when running automatic metrics.

Prompting We explored a range of prompt variations based on prior work on encouraging variation. Our final configuration is "If you were a [persona] write a python program that generates a [style] plot for [audience]", where the persona, style, and audience are varied. We also include the data in the prompt, as described in the next paragraph. Appendix A.1.1 shows the complete list of prompts. The persona is motivated by Salewski et al. (2024), who showed that specifying a persona improved performance on the Massive Multitask Language Understanding (MMLU) dataset. The style and audience variations are motivated by Liu and Chilton

[1] Approved by our university's institutional review board.

(2022), who evaluated text-to-image artwork generation with the prompt "[subject] in the style of [style]" and found that annotators had higher agreement when the subject and style were related.

Data We use four datasets, each with 15 samples. The datasets vary in the composition of the samples, both in terms of the number of fields and their types. These rows are presented in the prompt as a dictionary, preceded by "Given this data: ". Appendix A.1.2 shows the dimensions and types of the datasets used. This design is based on Chat2Vis (Maddigan and Susnjak, 2023), a system for generating data visualisations with LLMs.

Hyperparameters We varied the temperature and top-p value. Table 5 in Appendix A.1.3 shows the variations tried. Both of these can influence the variability in model output, where out-of-sample generations might be more novel and creative. For example, Döderlein et al. (2023) found that temperature and top-p values impact code generation quality as measured on the HumanEval and Leetcode datasets.

3.2 Accessibility metrics

We consider two aspects of accessibility: text color contrast, and text spacing. For each, we define a new metric, inspired by the guidelines in WCAG (Caldwell et al., 2008) and Material Design (Google, 2021). We outline our methods below. In both cases, we first recognize text boxes within the image using pyteserract (Smith, 2007). The equations refered to below can be found in Appendix A.3.

Contrast Higher color contrast makes text and non-text elements easier to differentiate. WCAG's Color Success Criteria 1.4.3 and 1.4.6 recommends (a) a 3:1 color contrast ratio between large text (14pt bold, or greater than 18pt) and the background, and (b) a 4.5:1 ratio for small text. Our method is as follows:

1. Perform color segmentation (Arumugadevi and Seenivasagam, 2015) to separate the foreground text color and the background color, using K-means with $k = 2$. Whenever only one cluster is found, the resulting color is assigned to both the foreground and the background.

2. Calculate the relative luminance between the segmented colors in step 2.1 according to Equation 1 and calculate the contrast ratio as in Equation 2.

3. Evaluate WCAG Success Criteria 1.4.3 and 1.4.6 according to Equation 4.

4. Compute the contrast accessibility metric according to Equation 3. The value for this metric is between zero and one; one means a perfect score for accessibility.

Text spacing To measure how much text should be placed on a visualisation and where it should go? (Hearst, 2023). We used the WCAG's Success Criteria 1.4.12, which aims to improve the reading experience and to ensure content readability and operability. It defines letter spacing as 0.12 times the font size and word spacing as 0.16 times the font size. We explore word spacing evaluation. Our method is as follows:

1. Group inline blocks of words.

2. Calculate the distance between consecutive words according to Equation 5.

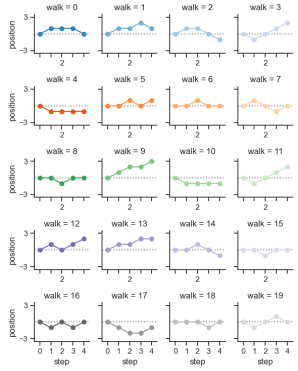
3. Evaluate WCAG Success Criteria 1.4.12 according to Equation 6.

4. Compute the text spacing metric according to Equation 7. The value for this metric is between -inf and inf; scores greater than zero are a reasonable value for text spacing.

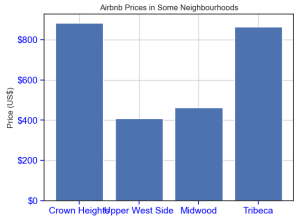
Figure 2 contains examples of how the color contrast score performs in different scenarios, while Figure 3 shows some text spacing data visualisations. Code that renders these visualisations can be found under Appendix A.3.3 in Table 7 and Table 8. Looking at samples, we observe that the contrast accessibility metrics scored the best when all text was black, while detecting text in a lighter color downgrades the score. In the case of the text spacing test, since the distance is relative to the font size, a larger font size tends to achieve higher scores, which is consistent with common advice on making data visualisations.

3.3 Human Study

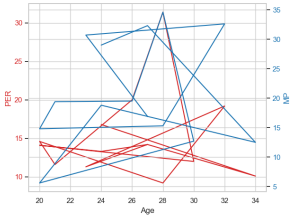
We designed a survey to assess creativity and accessibility. The survey had a few questions about prior knowledge, and six main sections: three on creativity and two on accessibility. We created three versions of the study, which differed in the order



Source: seaborn
Score: 1.0

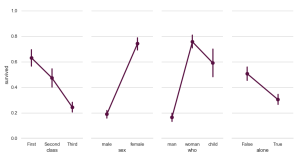


Source: Claude-2
Score: 0.81

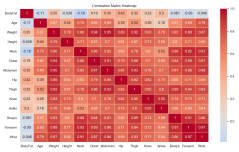


Source: GPT-4
Score: 0.29

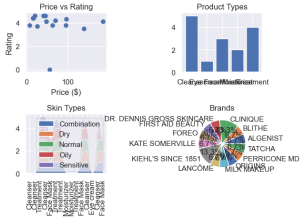
Figure 2: Color contrast examples. From 0 to 1, a perfect color contrast score is given when one.



Source: seaborn
Score: 315.16



Source: GPT-4
Score: -381.36



Source: Claude-2
Score: -2800.56

Figure 3: Text spacing examples. From -inf to inf, the higher the score, the more text space between the words in the image.

of data visualisations. This variation mitigated potential bias due to the order in which participants see the data visualisations. The study design was reviewed and approved by our university's Institutional Review Board (IRB). The complete survey template is included in supplementary material.

Creativity This task presents nine different visualisations to the participants. To answer the research question *Are LLMs creative according to the definition of surprise and novelty?*, the participants had to select and rate a data visualisation according to surprise and novelty. The nine visualisations come from the same prompt and data fragment to OpenAI's API *"Using your imagination write a Python program that generates a plot"*; different model parameters were set in each call. The values for temperature were 0.4, 0.6, 0.8, 1.0, 1.1, 1.2 and top p 0.4, 0.6, 0.8.

Personalization coherence The participants were shown nine data visualisations from each LLM (GPT-4 and Claude-2). This time, the prompt used was *"[Persona] write a python program that generates a [style] plot for [audience]"*. This task evaluates an association between the persona-audience and the style. For example, the prompt "If you were a school teacher write a python program that generates a complex plot for children", may lead to poor results as "children" and "complex" might be contradictory. The first part of the task kept the persona fixed and varied the style, e.g., two queries with a data scientist, one of which has complex and the other has simple. The second part of the task kept the style fixed and varied the persona, e.g., a school teacher and a digital designer both with a simple style. Participants were asked to select which data visualisations they liked the most and least within each set. Appendix A.2.1 contains this task's complete list of prompts.

Rationality To evaluate rationality and the accuracy of the visualisation towards the data. We included two open ended questions to the participants. The participants are asked to summarize what elements in a data visualisation made it more or less appealing. We finished the survey asking the participants to pick their favorite LLM from the personalization task.

Question	Answer
Did you find some plot that surprised you?	52.6% yes, 47.4% no
Which plot surprised you?	30% plot six, 20% plot four, 50% other
In the scale from 1 to 5, how would you rate the repetitiveness of the plot?	56.14% repetitive or more (> 2)
Is there a plot that looked different from the rest?	78.9% yes, 21.1% no
What was the plot that looked different?	51% plot six, 22% plot nine, 27% other
Which of the plots was your favorite?	43.8% plot six, 55.2% other

Table 1: Analysis of the creativity assessment in the survey.

Text spacing accessibility This task aims to check if our metric for text spacing matches with human perception. The participants were presented with ten data visualisations, six data visualisations were generated by LLMs, and four were human-written from sample documentation. Participants were asked if the text-spacing in the data visualisations was accessible. We sample the LLM data visualisations to cover a wide range of the scores given by our metrics (specifically, from the first and third quartiles, and outliers, if any).

Color contrast accessibility The procedure was the same as in the text spacing task, but we ask about the color contrast between the text and the background.

4 Analysis

We obtained responses from a total of 57 participants. Their experience with visualisations varied significantly 12.3% indicated low experience, 64.9% indicated medium experience, and 22.8% indicated high experience. In terms of tools, all but two had used Excel, and 36.8% had never used Tableau. In terms of programming languages, 42.1% had used just Python, and 34% had used both Python and R. 61.4% also reported using, at least one time, a programming language other than Python and R for visualisations. This range of expertise indicates that our sample is not biased towards people with a specific background in terms of tools.

4.1 Creativity evaluation

Creativity questions Here we are interested in two key questions: *Are LLMs capable of generating self-written code showing notions of creativity?*, and *Are LLMs creative according to the definition of surprise and novelty?* First, we will clarify the difference between surprise and novelty (Xu et al., 2021). Consider entering your kitchen. You expect to see your fridge in a certain location. If it is not there then the kitchen has a novel appearance. Is it surprising? That depends on whether you expected it to be there. If you knew it was being repaired then you would not be surprised, but if it was removed without your knowledge then you would be surprised.

The perception of repetitiveness in the data visualisations contradicts the idea of unanticipated surprise. Table 1 presents some of the questions and its answers in percentage.

One might conclude that participants found at least one plot (plot six) novel but had low consistency regarding surprise. This is also explained by a moderate agreement obtained through the Fleiss-kappa score of 0.23. The complete set of results for this task can be found under Appendix A.2.2.

Personalization coherence For the task generated by GPT-4 compared to Claude-2, there is a strong relationship between the favorite and least favorite data visualisations for GPT-4 since the difference between these two columns deviates from zero. This demonstrates the consistency of the answers given by the participants. This task achieved fair reliability with a Fleiss-kappa agreement score of 0.28.

Table 2 presents the prompts per subset of questions varying in style within and between persona-audience. We conclude that there is no precise alignment between the association of style and the persona-audience. For example, the prompt "If you were a digital designer, write a python program that generates a simple plot for the whole world" scored 14 between personas and 28 within its style when asked, "If you were a digital designer write a python program that generates a complex plot for the whole world" the votes shifted to 2 votes between personas and 19 within its style.

This same evaluation was conducted for Claude-2 outputs. Results for this task can be found in

	Persona-audience	Style	Most	Least	Most - Least
Vary audience	Data scientist-stakeholders	Complex	27	13	14
Vary audience	Digital designer-world	Complex	16	14	2
Vary audience	School teacher-children	Complex	14	30	-16
Vary audience	Data scientist-stakeholders	None	51	1	50
Vary audience	Digital designer-world	None	1	43	-42
Vary audience	School teacher-children	None	5	13	-8
Vary audience	Data scientist-stakeholders	Simple	9	42	-33
Vary audience	Digital designer-world	Simple	21	7	14
Vary audience	School teacher-children	Simple	27	8	19
Vary style	Data scientist-stakeholders	None	39	2	37
Vary style	Data scientist-stakeholders	Complex	16	18	-2
Vary style	Data scientist-stakeholders	Simple	2	37	-35
Vary style	Digital designer-world	Simple	30	2	28
Vary style	Digital designer-world	Complex	26	7	19
Vary style	Digital designer-world	None	1	48	-47
Vary style	School teacher-children	Simple	36	1	35
Vary style	School teacher-children	Complex	8	26	-18
Vary style	School teacher-children	None	13	30	-17

Table 2: GPT-4 prompts relating to persona, style, and audience.

Table 6 in Appendix A.2.3. Here, we can notice that contradictory prompts such as "If you were a school teacher write a python program that generates a complex plot for children" were highly rated with a total of 30 votes, when comparing style. Also, the Fleiss-kappa agreement score for Claude-2 was about 0.16, indicating poor reliability and a less clear pattern among responses.

Rationality Participants described data visualisations as appealing when they had the following characteristics: good readability, simplicity, and accurate visualisation trends. On the other hand, having too busy information, bright colors, and either a lack of or obstructed labels are among the worst characteristics in the data visualisations. When asking participants to choose a preferred LLM for this task, Claude-2 obtained 25 votes, while GPT-4 got 17 votes, and 15 participants could not decide. However, these variations do not indicate a consistent trend. The Fleiss-kappa score is -0.01, indicating no agreement (McHugh, 2012).

4.1.1 Token evaluation

We also considered evaluating the code itself in terms of creativity. Specifically, do these models generate creative implementations of data visualisations? To test this, we considered the token distribution in the 1,657 outputs from the language models. Appendix A.4 presents three selected prompts per experiment; either persona, style and audience, and their token distributions. Overall, these experiments showed no significant changes in the token space. We can learn from this null result though. First, it indicates that the model might ignore the use of impersonation in coding assistants. Second, it shows that the generated code is drawn from a consistent distribution and that any creativity observed in the outputs is the result of small variations in the code rather than major changes in implementation.

4.2 Accessibility evaluation

First, we will consider the human evaluation of accessibility, to determine whether our new metrics are consistent with human perception.

Text spacing accessibility The data visualisations for this task were: four human-written, three generated by GPT-4, and three by Claude-2. One visualisation of each source was considered non-accessible by the participants. This task obtained a score of agreement using Fleiss-kappa of 0.47, this value suggests a fair reliability (Fleiss, 2003). After gathering the results from the survey, we ranked the data visualisations by how many people said they were accessible. We compare this ranking

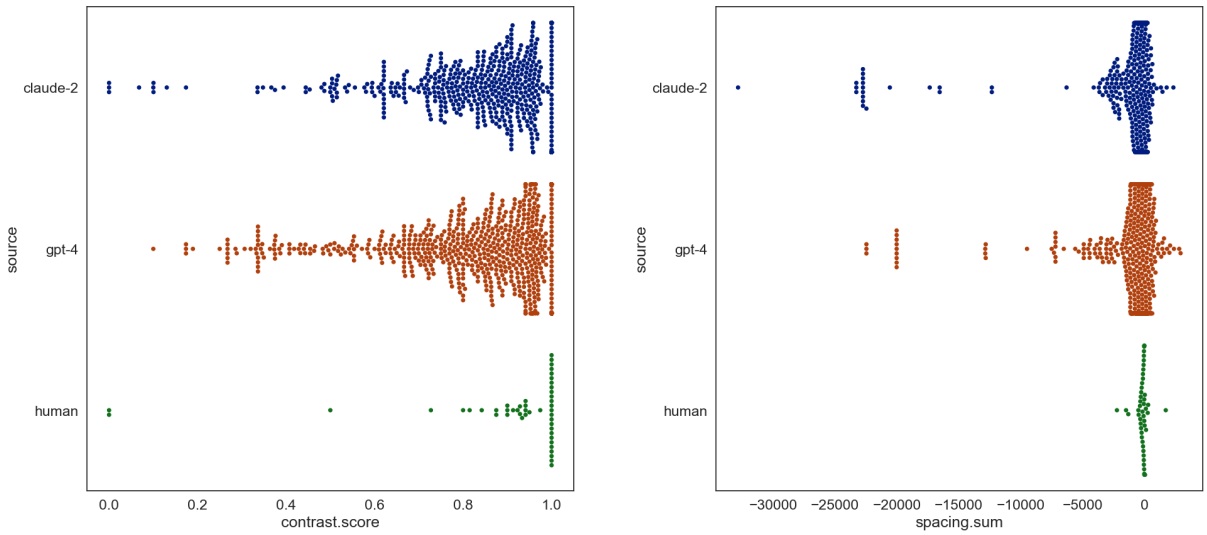


Figure 4: Accessibility scores. The contrast score is on the left, and the spacing score is on the right. These are swarm plots, where each point is placed at approximately the right x-value, with movement so all are shown. This makes the distribution visible in a more nuanced way than a box plot or violin plot.

with the ranking produced by our automatic metric and measure Spearman's coefficient between the human and automatic rankings. We find a correlation of 0.73 with a p-value of 0.02, indicating high agreement that is statistically significant.

Color contrast accessibility We apply the same analysis to this question. Inter-rater reliability is lower, with a Fleiss-kappa of 0.11. However, Spearman's coefficient for comparing human and machine rankings was 0.73 with a p-value of 0.02[2]. These results are likely due to the fact that issues with color contrast had a narrow separation among participants.

Interestingly, some of the samples that came from human-written documentation performed poorly. This suggests that there is value in these metrics for human-written code as well, to inform the creation of more accessible data visualisations.

4.2.1 Metric-based evaluation

Now we turn to automatic metrics, which we apply to the full set of data visualisations we generated. Figure 4 shows the scores from the Section 3.2 methodology in a swarm plot. The scores are grouped by the source that generated the code. In both metrics, the human code achieves scores that are almost all positive and far more consistent than the LLMs.

We performed the Levene test to validate these assumptions to compare the variances among our non-normal distributed samples (Gastwirth et al., 2009). Even after accounting for different sample sizes, GPT-4 and Claude-2 showed higher variances than human-written code. Also, when setting a threshold of 0.8, as a value of good contrast, 74% of the LLMs outputs were on this set, while 95% of the human output surpassed the threshold. Regarding spacing, 34% of the LLMs showed a positive spacing, while 59% of the human samples were greater or equal than zero.

4.3 Limitations

Creativity When conducting a study with people it is not possible to consider every variation of interest. It is possible that these models do exhibit creativity, but that it was not reflected in the data visualisations sampled for use in our survey.

Accessibility We did not prompt the LLM to generate visualisations considering accessibility. However, from our findings in the survey's section about personalization coherence, we see that LLM's responses do not relate to the prompt.

This paper has set a baseline to quantify accessibility metrics. These metrics can be used further to fine-tune models whose output renders interfaces combining images and text, such as visualisations. Similarly, exploring these metrics as a reward after code execution with reinforcement learning is an exciting direction. However, the accuracy of the proposed metrics highly depends on the reliability of the text detection model. Improving the object recognition model could produce better results. It

[2]These results were reviewed and we can confirm that both metrics correlate at the same level, to human rankings.

would also be beneficial to extend its capacity to differentiate elements of the data visualisations, such as bars or markers, that could provide more informative and explainable summaries on accessibility.

5 Conclusions

Do LLMs generate creative and visually accessible data visualisations? Regarding creativity, the code itself is not particularly creative, and the outputs are sometimes novel, but not surprising. For accessibility, generated data visualisations are typically effective, but can span a wide range of effectiveness. Overall, this work shows that data visualisation remains a challenging space for LLMs to generate creative outputs. That is not a major issue for generating simple data visualisations, but more work is needed to be able to handle more personalized or complex requests.

Ethics statement

This study involved human participants. The protocol was reviewed and approved by our university's institutional review board before the experiment was conducted. Participants could drop out at any time with no penalty. There was minimal risk to participants and a small ($6.50 USD) gift card as a reward for participation. Participants agreed that their responses be shared and analyzed once anonymized and de-identified to protect their privacy. The findings of the study do not have significant ethical implications.

Acknowledgments

This work is partially funded by the Australian Research Council through a Discovery Early Career Researcher Award and the Collaborative Intelligence Future Science Platform (FSP) of the Commonwealth Scientific and Industrial Research Organisation (CSIRO). We also acknowledge the reviewers providing feedback and advice on our submission.

References

NC State University. 2014. *Color Contrast Analyzer*. IT Accessibility Office NC State University, North Carolina, USA.

Bryan Alba, Maria Fernanda Granda, and Otto Parra. 2022. Ui-test: A model-based framework for visual ui testing– qualitative and quantitative evaluation. In *Evaluation of Novel Approaches to Software Engineering*, pages 328–355, Cham. Springer International Publishing.

Barrett R Anderson, Jash Hemant Shah, and Max Kreminski. 2024. Evaluating creativity support tools via homogenization analysis. In *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI EA '24, New York, NY, USA. Association for Computing Machinery.

S. Arumugadevi and V. Seenivasagam. 2015. Comparison of clustering methods for segmenting color images. *Indian Journal of Science and Technology*, 8:670.

Abid Ali Awan. 2021. Cosmetics datasets.

Arian Azmoudeh. 2022. Airbnb open data.

Sebastian Berns and Simon Colton. 2020. Bridging generative deep learning and computational creativity. In *International Conference on Innovative Computing and Cloud Computing*.

Margaret A Boden. 2010. *Creativity and Art : Three Roads to Surprise*. Oxford University Press, Incorporated, Oxford, UNITED KINGDOM.

Benjamin Caldwell, Michael Cooper, Loretta Guarino Reid, and Gregg C. Vanderheiden. 2008. Web content accessibility guidelines (wcag) 2.0.

Tuhin Chakrabarty, Philippe Laban, Divyansh Agarwal, Smaranda Muresan, and Chien-Sheng Wu. 2023. Art or artifice? large language models and the false promise of creativity. In *Proceedings of the 2024 ACM CHI Conference on Human Factors in Computing Systems*, Hawaii' 24:, New York, NY, USA. Association for Computing Machinery.

Simon Colton, Edward J. Powley, and Michael Cook. 2018. Investigating and automating the creative act of software engineering. In *Proceedings of the Ninth International Conference on Computational Creativity, ICCC 2018*, pages 224–231. Association for Computational Creativity (ACC).

Michael Cook, Simon Colton, and J. Gow. 2013. Nobody's a critic: On the evaluation of creative code generators - a case study in video game design. In *International Conference on Innovative Computing and Cloud Computing*.

Jean-Baptiste Döderlein, Mathieu Acher, Djamel Eddine Khelladi, and Benoit Combemale. 2023. Piloting copilot and codex: Hot temperature, cold prompts, or black magic? *arXiv preprint arXiv:2210.14699*.

Anil R. Doshi and Oliver P. Hauser. 2024. Generative ai enhances individual creativity but reduces the collective diversity of novel content. *Science Advances*, 10(28):eadn5290.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Myunghee Cho Paik Fleiss, Bruce Levin. 2003. *The Measurement of Interrater Agreement*, chapter 18. John Wiley & Sons, Ltd.

Giorgio Franceschelli and Mirco Musolesi. 2022. Deepcreativity: measuring creativity with deep learning techniques. *Intelligenza Artificiale*, 16(2):151–163.

Giorgio Franceschelli and Mirco Musolesi. 2023. On the creativity of large language models. *arXiv preprint arXiv:2304.00008*.

Joseph L Gastwirth, Yulia R Gel, and Weiwen Miao. 2009. The impact of levene's test of equality of variances on statistical theory and practice. *Statistical Science*, 24(3):343–360.

Regine M Gilbert. 2019. *Inclusive Design for a Digital World: Designing with Accessibility in Mind*, 1st ed edition. Apress L. P, Berkeley, CA.

Google. 2021. *Material Design 3*. Google User Experience Research, Mountain View, USA.

Samine Hadadi. 2021. Adee: Bringing accessibility right inside design tools. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '21, New York, NY, USA. Association for Computing Machinery.

Tianxing He, Jingyu Zhang, Tianle Wang, Sachin Kumar, Kyunghyun Cho, James Glass, and Yulia Tsvetkov. 2023. On the blind spots of model-based evaluation metrics for text generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12067–12097, Toronto, Canada. Association for Computational Linguistics.

Marti A. Hearst. 2023. Show it or tell it? text, visualization, and their combination. *Commun. ACM*, 66(10):68–75.

J. D. Hunter. 2007. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Alayt Issak and Lav R. Varshney. 2023. Prompt programming for the visual domain. In *The First Tiny Papers Track at ICLR 2023, Tiny Papers @ ICLR 2023, Kigali, Rwanda, May 5, 2023*. OpenReview.net.

Roger W. Johnson. 2023. Body fat extended dataset.

Jeongyeon Kim, Sangho Suh, Lydia B Chilton, and Haijun Xia. 2023. Metaphorian: Leveraging large language models to support extended metaphor creation for science writing. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference*, DIS '23, page 115–135, New York, NY, USA. Association for Computing Machinery.

Vivian Liu and Lydia B Chilton. 2022. Design guidelines for prompt engineering text-to-image generative models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–23.

Paula Maddigan and Teo Susnjak. 2023. Chat2vis: Fine-tuning data visualisations using multilingual natural language text and pre-trained large language models. *arXiv preprint arXiv:2303.14292*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochem. Med. (Zagreb)*, 22(3):276–282.

David Noever and Samantha Elizabeth Miller Noever. 2023. The multimodal and modular ai chef: Complex recipe generation from imagery. *arXiv preprint arXiv:2304.02016*.

Leonard Salewski, Stephan Alaniz, Isabel Rio-Torto, Eric Schulz, and Zeynep Akata. 2024. In-context impersonation reveals large language models' strengths and biases. *Advances in Neural Information Processing Systems*, 36.

R. Smith. 2007. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633.

A. Wang. 2023. The art of data visualization: Communicating insights with impact. *Journal of Research in International Business and Management*, 10(4):01–02.

Michael L. Waskom. 2021. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021.

Jamie Welsh. 2023. Nba per game and advanced stats (2022-23 season).

He A Xu, Alireza Modirshanechi, Marco P Lehmann, Wulfram Gerstner, and Michael H Herzog. 2021. Novelty is not surprise: Human exploratory and adaptive behavior in sequential decision-making. *PLoS Comput Biol*, 17(6):e1009070.

Bo Yang, Zhenchang Xing, Xin Xia, Chunyang Chen, Deheng Ye, and Shanping Li. 2021. Don't do that! hunting down visual design smells in complex uis against design guidelines. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 761–772.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. In *Proceedings of the 15th International Conference on Mining Software Repositories*, pages 476–486.

Shuyan Zhou, Uri Alon, Sumit Agarwal, and Graham Neubig. 2023. Codebertscore: Evaluating code generation with pretrained models of code. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13921–13937.

A Appendix

A.1 Data

A.1.1 Prompts

The next is a list of the 35 prompts that generated the visualisations, Table 3 shows the categories of each variation.

1. If you were a designer write a python program that generates a plot for family.
2. If you were a designer write a python program that generates a plot for stakeholders.
3. If you were a designer write a python program that generates a 3D plot
4. If you were a designer write a python program that generates a complex plot.
5. If you were a doctor write a python program that generates a plot for family.
6. If you were a doctor write a python program that generates a complex plot.
7. If you were a marketing team write a Python program that generates a complex plot.
8. If you were a marketing team write a python program that generates a plot for stakeholders.
9. If you were a school teacher write a Python program that generates a 2D plot.
10. If you were a school teacher write a python program that generates a plot for primary school children.
11. If you were feeling angry how would you write a python program that generates a complex plot.
12. If you were feeling happy how would you write a python program that generates a complex plot.
13. If you were feeling sad how would you write a python program that generates a complex plot
14. Using your imagination write a Python program that generates a plot for the whole world.
15. Using your imagination write a Python program that generates a informative plot.
16. Using your imagination write a python program that generates a plot for family.
17. Using your imagination write a python program that generates a plot for stakeholders.
18. Using your imagination write a python program that generates a angry plot
19. Using your imagination write a python program that generates a communicative plot.
20. Using your imagination write a python program that generates a happy plot.
21. Using your imagination write a python program that generates a sad plot
22. Write a Python program that generates a complex plot for primary school children.
23. Write a python program that generates a angry plot for colleagues.
24. Write a python program that generates a angry plot for stakeholders.
25. Write a python program that generates a communicative plot for family.
26. Write a python program that generates a communicative plot for friends.
27. Write a python program that generates a complex plot for colleagues.
28. Write a python program that generates a complex plot for family.
29. Write a python program that generates a complex plot for friends.
30. Write a python program that generates a complex plot for stakeholders.
31. Write a python program that generates a complex plot for the whole world.
32. Write a python program that generates a happy plot for primary school children.
33. Write a python program that generates a happy plot for stakeholders.

34. Write a python program that generates a happy plot for the whole world.

35. Write a python program that generates a sad plot for primary school children.

A.1.2 Datasets

Table 4 contains the name of the dataset, a descriptive type of the types of attributes in the dataset, the citation reference, the column size, and the column names with the data type of each sample dataset for the experiments.

A.1.3 Parameters

Table 5 describes the models' parameters used for constructing the LLM's generated dataset.

A.2 Creativity evaluation

A.2.1 Survey: Communication

1. If you were a data scientist write a python program that generates a plot for stakeholders.

2. If you were a data scientist write a python program that generates a simple plot for stakeholders.

3. If you were a data scientist write a python program that generates a complex plot for stakeholders.

4. If you were a school teacher write a python program that generates a plot for children.

5. If you were a school teacher write a python program that generates a simple plot for children.

6. If you were a school teacher write a python program that generates a complex plot for children.

7. If you were a digital designer write a python program that generates a plot for the whole world.

8. If you were a digital designer write a python program that generates a simple plot for the whole world.

9. If you were a digital designer write a python program that generates a complex plot for the whole world.

A.2.2 Analysis: Creativity

Figure 5 contains the statistics of each question presented in the survey for the creativity section.

A.2.3 Analysis: Communication

Table 6 shows the participants' votes for prompts in task E (Claude-2).

A.3 Visual accessibility

A.3.1 Color contrast

The equation 1 of the **Relative Luminance** is:

$$L = 0.2126 * R + 0.7152 * G + 0.0722 * B \tag{1}$$

The equation 1 to calculate the relative brightness of any point in the *sRGB* color space.

The equation 2 of the **Contrast Ratio** is:

$$ContrastRatio = \frac{max(color_x, color_y)+0.05}{min(color_x, color_y)+0.05} \tag{2}$$

The equation 2 to calculate the contrast ratio between the luminance of two colors x and y.

The equation 3 of the **Contrast Accessibility** is:

$$ContrastScore = 1 - \frac{UnsuccessfulCriteria}{NumTexts} \tag{3}$$

The equation 3 is calculated as the ratio of identified texts that do not qualify in the Success Criteria 1.4.3 and 1.4.6.

The equation 4 of the **Unsuccess Contrast Criteria** is:

$$\begin{cases} 1 \text{ if } (FontSize > 14) \wedge (ContrastRatio > 4.5) \\ 1 \text{ if } (ContrastRatio > 3) \\ 0 \text{ else} \end{cases} \tag{4}$$

The equation 4 is unsuccessful when text with a font size of 14pt or smaller has a contrast ratio with the background less than 4.5, and for larger text, the contrast ratio is less than 3. Font size equals the height obtained from the text through pyteserract OCR.

A.3.2 Text spacing

The equation 5 of the **Distance between words** is:

$$distance(w_i, w_{i+1}) = left_{i+1} - (left_i + width_i) \tag{5}$$

The equation 5 is calculated between two consecutive words w_i and w_{i+1}, in an inline group of blocks of words. This is defined as the subtraction of the right corner of the first word from the left corner of the second word. The right corner is equivalent to the left corner of the word in the direction of its width.

Persona	Style	Audience
Designer	2D	colleagues
Doctor	3D	family
Marketing team	angry	friends
School teacher	communicative	primary school children
Feeling angry	complex	stakeholders
Feeling sad	happy	the whole world
Feeling happy	informative	
Using your imagination	sad	

Table 3: Selected categories for prompt engineering.

Dataset name	**NBA players**	**Cosmetics**	**Body composition**	**Airbnb**
Type	**Mixed**	**Categorical only**	**Numerical only**	**Mixed**
Author	Welsh, 2023	Awan, 2021	Johnson, 2023	Azmoudeh, 2022
# of columns	**11**	**9**	**14**	**24**
Column name (datatype)	Player Name (str)	Label (str)	Age (int64)	id (int64)
	Position (str)	Brand (str)	BodyFat (float64)	NAME (str)
	Team (str)	Name (str)	Weight (float64)	host id (int64)
	Age (int64)	Combination (int64)	Height (float64)	host name (str)
	GP (int64)	Dry (int64)	Neck (float64)	neighbourhood group (str)
	AST (float64)	Normal (int64)	Chest (float64)	neighbourhood (str)
	TRB (float64)	Oily (int64)	Abdomen (float64)	country (str)
	TS% (float64)	Sensitive (int64)	Hip (float64)	country code (str)
	WS/48 (float64)	Rank (float64)	Thigh (float64)	cancellation_policy (str)
	PER (float64)		Knee (float64)	room type (str)
	MP (float64)		Ankle (float64)	host_identity_verified (str)
			Biceps (float64)	instant_bookable (str)
			Forearm (float64)	review rate number (float64)
			Wrist (float64)	Construction year (float64)
				minimum nights (float64)
				number of reviews (float64)
				reviews per month (float64)
				calculated host listings count (float64)
				availability 365 (float64)
				last review (time)
				lat (float64)
				long (float64)
				price (float64)
				service fee (float64)

Table 4: Datasets used to build the prompts.

LLM	Temperature	Top p
GPT-4	0.8, 0.9, 1.0, 1.1, 1.2	0.8, 0.9, 1.0
Claude-2	0.8, 0.9, 1.0	0.8, 0.9, 1.0

Table 5: LLM's parameters for experiments.

The equation 6 of the **Unsuccess Spacing Criteria** is:

$$\begin{cases} 1 \text{ if } distance(w_i, w_{i+1}) > \\ SpacingCriteria * FontSize \\ 0 \text{ else} \end{cases} \quad (6)$$

The equation 6 is unsuccessful when overlapping text occurs between the proportion of the font size and the spacing criteria. For word spacing, the spacing criteria is a constant equal to 0.16.

The equation 7 of the **Text Spacing Accessibility** is:

$$SpacingScore = \sum_j^n \sum_i^m distance(w_i, w_{i+1}) \\ -0.16 * FontSize(w_i) \quad (7)$$

The equation 7 is based on the Success Criteria 1.4.12. For *n* inline blocks, calculate the distance between the consecutive pairs of the *m* words of the block, and subtract the spacing criteria concerning the jth word font size. The more negative the score means more text overlaps.

A.3.3 Code rendered

Table 7 presents the code that renders Figure 2, Table 8 shows code for Figure 3.

A.4 Token evaluation

A.4.1 Persona

Figure 9 presents the comparison in terms of normalized unique tokens bins of frequencies for three different persona prompts: "If you were feeling

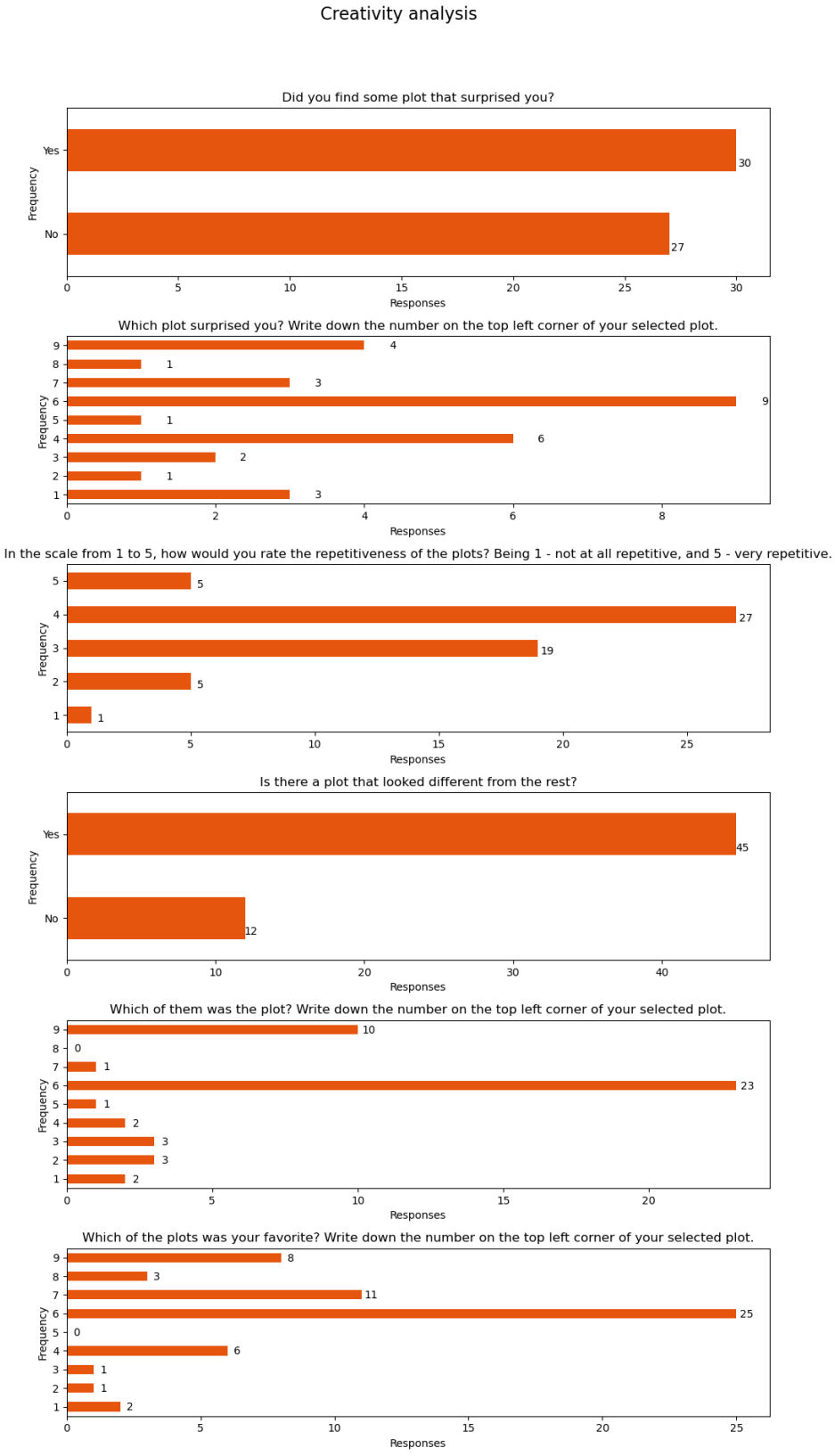


Figure 5: Results from survey's task on creativity.

sad", "Using your imagination" and "If you were a marketing team".

A.4.2 Style

Figure 10 presents the comparison in terms of normalized unique tokens bins of frequencies for three

	Persona-audience	Style	Most	Least	Most - Least
Vary audience	Data scientist-stakeholders	Complex	13	17	-4
Vary audience	Digital designer-world	Complex	25	21	4
Vary audience	School teacher-children	Complex	19	19	0
Vary audience	Data scientist-stakeholders	None	20	8	12
Vary audience	Digital designer-world	None	32	2	30
Vary audience	School teacher-children	None	5	47	-42
Vary audience	Data scientist-stakeholders	Simple	36	7	29
Vary audience	Digital designer-world	Simple	11	17	-6
Vary audience	School teacher-children	Simple	10	33	-23
Vary style	Data scientist-stakeholders	None	11	23	-12
Vary style	Data scientist-stakeholders	Complex	14	22	-8
Vary style	Data scientist-stakeholders	Simple	32	12	20
Vary style	Digital designer-world	Simple	9	27	-18
Vary style	Digital designer-world	Complex	24	21	3
Vary style	Digital designer-world	None	24	9	15
Vary style	School teacher-children	Simple	19	7	12
Vary style	School teacher-children	Complex	34	4	30
Vary style	School teacher-children	None	4	46	-42

Table 6: Claude-2 prompts relating to persona, style, and audience.

different style prompts: "complex plot", "2D plot" and "happy plot".

A.4.3 Audience

Figure 11 presents the comparison in terms of normalized unique tokens bins of frequencies for three different audience prompts: "for the whole word", "for primary school children" and "for stakeholders".

```
Source: seaborn
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_theme(style="ticks")


rs = np.random.RandomState(4)
pos = rs.randint(-1, 2, (20, 5)).cumsum(axis=1)
pos -= pos[:, 0, np.newaxis]
step = np.tile(range(5), 20)
walk = np.repeat(range(20), 5)
df = pd.DataFrame(np.c_[pos.flat, step, walk],
                  columns=["position", "step", "walk"])


grid = sns.FacetGrid(df, col="walk", hue="walk", palette="tab20c",
                     col_wrap=4, height=1.5)


grid.refline(y=0, linestyle=":")


grid.map(plt.plot, "step", "position", marker="o")


grid.set(xticks=np.arange(5), yticks=[-3, 3],
         xlim=(-.5, 4.5), ylim=(-3.5, 3.5))


grid.fig.tight_layout(w_pad=1)
```

```
Source: Claude-2
import matplotlib.pyplot as plt

# Choose some data to plot
neighbourhoods = ['Crown␣Heights', 'Upper␣West␣Side', 'Midwood', 'Tribeca']
prices = [882, 407, 463, 864]

# Create the bar plot
fig, ax = plt.subplots()
ax.bar(neighbourhoods, prices)

# Add labels and title
ax.set_ylabel('Price␣(US$)')
ax.set_title('Airbnb␣Prices␣in␣Some␣Neighbourhoods')

# Make plot more readable for kids
ax.grid(True)
ax.set_facecolor('white')
ax.tick_params(colors='blue', labelsize=14)

# Add the dollar sign
import matplotlib.ticker as mtick
formatter = mtick.StrMethodFormatter('${x:,.0f}')
ax.yaxis.set_major_formatter(formatter)

fig.tight_layout()
```

```
Source: GPT-4
import numpy as np
import matplotlib.pyplot as plt

data = {...}

x = data['Age']
y_1 = data['PER']
y_2 = data['MP']

fig, ax1 = plt.subplots()

color = 'tab:red'
ax1.set_xlabel('Age')
ax1.set_ylabel('PER', color=color)
ax1.plot(x, y_1, color=color)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()

color = 'tab:blue'
ax2.set_ylabel('MP', color=color)
ax2.plot(x, y_2, color=color)
ax2.tick_params(axis='y', labelcolor=color)

fig.tight_layout()
```

Table 7: Source code for the color contrast example. Left to right in Figure 2 is top to bottom in this table.

```
Source: seaborn
import seaborn as sns
sns.set_theme(style="whitegrid")


titanic = sns.load_dataset("titanic")


g = sns.PairGrid(titanic, y_vars="survived",
                 x_vars=["class", "sex", "who", "alone"],
                 height=5, aspect=.5)


g.map(sns.pointplot, color="xkcd:plum")
g.set(ylim=(0, 1))
sns.despine(fig=g.fig, left=True)
```

```
Source: GPT-4
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Given data
data = {...}

# Create DataFrame
df = pd.DataFrame(data)

# Calculate correlation matrix
corr = df.corr()

# Plot the heatmap
plt.figure(figsize=(14, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', center=0, linewidths=0.5)
plt.title('Correlation Matrix Heatmap')
```

```
Source: Claude-2
import matplotlib.pyplot as plt


# Plot a bar chart showing average price for each product label
avg_prices = df.groupby('Label').Price.mean()
ax = avg_prices.plot.bar(rot=0)

# Add axes and title
ax.set_ylabel('Average Price ($)')
ax.set_title('Average Price by Skincare Product Category')

# Annotate each bar with the exact price
for p in ax.patches:
    x = p.get_x() + p.get_width() / 2
    y = p.get_height()
    ax.annotate('${:.2f}'.format(y), (x, y), ha='center', va='bottom')
```

Table 8: Source code for the text-spacing example. Left to right in Figure 3 is top to bottom in this table.

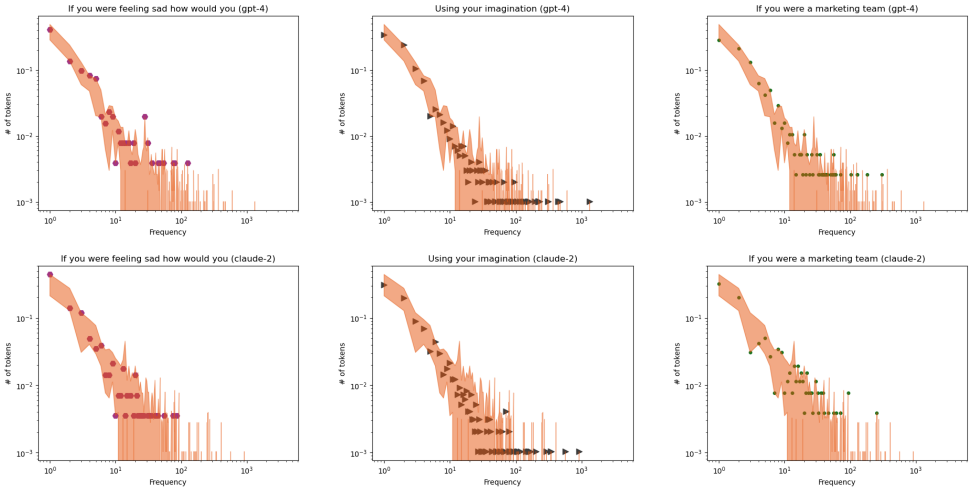


Figure 9: Unique tokens' distribution of three persona prompts grouped by bins. On top output from GPT-4, on the bottom output from Claude-2.

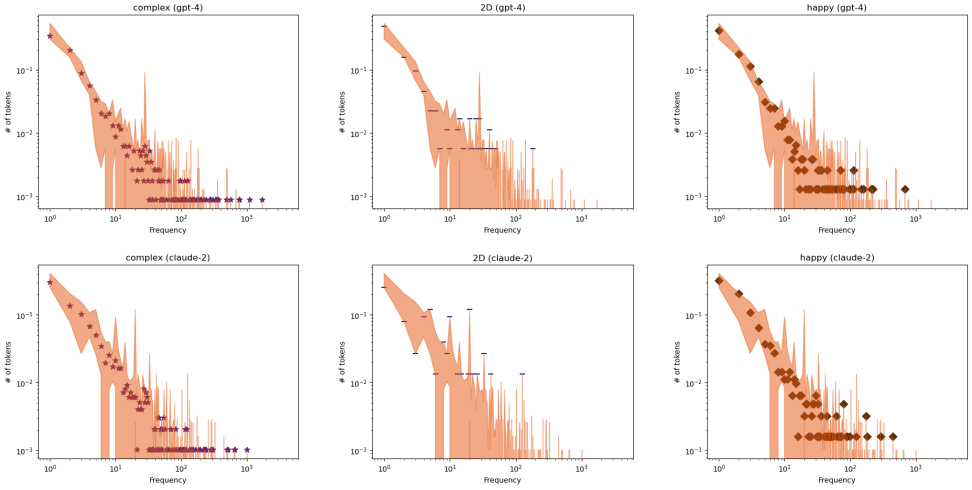


Figure 10: Unique tokens' distribution of three style prompts grouped by bins. On top output from GPT-4, on the bottom output from Claude-2.

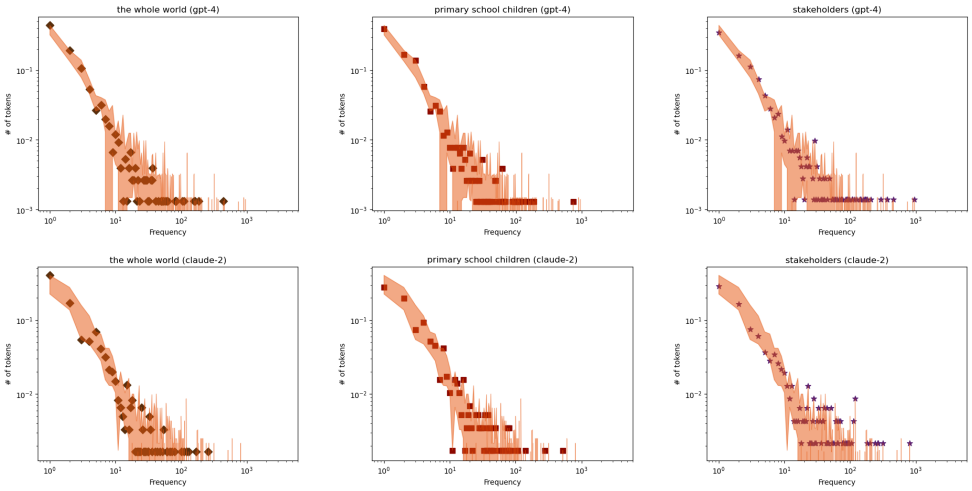


Figure 11: Unique tokens' distribution of three audience prompts grouped by bins. On top output from GPT-4, on the bottom output from Claude-2.