EuroVis Workshop on Visual Analytics (2024)
M. El-Assady and H.-J. Schulz (Editors)

Toward a Structured Theoretical Framework for the Evaluation of Generative AI-based Visualizations

L. Podo[1], M. Ishmal[1] and M. Angelini[2,1]

[1]Sapienza University of Rome, Italy
[2]Link Campus University, Rome, Italy

Abstract

The automatic generation of visualizations is an old task that, through the years, has shown more and more interest from the research and practitioner communities. Recently, large language models (LLM) have become an interesting option for supporting generative tasks related to visualization, demonstrating initial promising results. At the same time, several pitfalls, like the multiple ways of instructing an LLM to generate the desired result, the different perspectives leading the generation (code-based, image-based, grammar-based), and the presence of hallucinations even for the visualization generation task, make their usage less affordable than expected. Following similar initiatives for benchmarking LLMs, this paper explores the problem of modeling the evaluation of a generated visualization through an LLM. We propose a theoretical evaluation stack, EvaLLM, that decomposes the evaluation effort in its atomic components, characterizes their nature, and provides an overview of how to implement them. One use case on the Llama2-70-b model shows the benefits of EvaLLM and illustrates interesting results on the current state-of-the-art LLM-generated visualizations. The materials are available at this GitHub repository: `https://github.com/lucapodo/evallm_llama2_70b.git`

CCS Concepts

• ***Human-centered computing*** → ***Visualization design and evaluation methods;***

1 Introduction

In the last year, Large Language Models (LLMs) have become everywhere across various disciplines, demonstrating remarkable efficacy and capabilities in performing different tasks. Noteworthy applications span from finance [WIL*23] to coding, exhibiting tangible impacts in disparate domains and presenting a valuable opportunity for human augmentation. For example, Github reports an enhancement in developer productivity by a 55% increase in writing code, attributed to the introduction of Copilot, an LLM model fine-tuned to generate code. In the visualization field, LLMs exhibit promising capabilities in generating visualization as images and code [WYKN20, TCD*23], using libraries such as D3.js and Matplotlib A significant implication of this capability is that these models may empower non-expert users to generate insightful visualizations without prior data visualization expertise, offering a distinct advantage in creating visualizations through natural language queries [CLM*22, MS23]. Despite the prevalence of models (e.g., GPT-4, LLama) and their continuous improvement, a significant portion of their behavior remains ripe for exploration and further scrutiny. To bridge this knowledge gap, researchers are investigating their capabilities across diverse benchmark datasets [WSM*18, ZCG*23]. While natural language processing, general knowledge, common sense, problem-solving, advanced reasoning, and coding tasks have undergone thorough examination, visualization skills remain an area demanding further exploration due to its preliminary results.

In this paper, we cope with the problem of modeling the evaluation of LLM-generated visualizations and informing specific benchmarks for LLM-based visualizations to foster quantitative multi-faceted evaluation and comparability. We introduce EvaLLM, a conceptual stack to evaluate LLM-generated visualization. It decomposes the evaluation effort in its atomic components, characterizes their nature, and provides an overview of how to implement and interpret their results.

Finally, we present one initial qualitative use case that evaluates Llama2-70b models on 50 representative samples from the NvBench dataset [LTL21a]. The use case analysis shows common errors in visualization generation, from the more structural to the more semantic errors, allowing their identification at specific levels of the EvaLLM stack.

2 Related work

In the literature, the task-aware Visualization Recommendation System (VRS) [PPV24] models predominantly rely on traditional methodologies. For example, in [SBT*16], the authors introduce an interactive visualization tool that employs a hybrid approach integrating Natural Language Processing (NLP) and decision rules. In [SS23], the authors present BOLT, a web-based platform for multi-dashboard authoring using natural language. The authors propose a system based on traditional NLP techniques to map user utterances to prevalent dashboard objectives and generate appropriate visualizations. In contrast, [NSS20] introduces a method for interacting with a dataset based

DOI: 10.2312/eurova.20241118

on NLP, focusing on generating a single visualization rather than suggesting a complete dashboard. Moving beyond rule-based approaches, [LTL*21b] proposes ncNet, a seq2seq model that translates Natural Language Queries (NLQs) into a custom visualization grammar, Vega-Zero. While ncNet represents a breakthrough in processing the user inputs as free text, it still faces challenges in handling ambiguous and ill-posed natural queries. Despite the effectiveness of these approaches, they grapple with the challenge of capturing underlying semantics in user utterances, mainly when dealing with ambiguous expressions. More recently, LLM approaches have risen in popularity, overcoming some of the rule-based approaches' limitations while introducing others. In [HC23], the authors propose AI Thread, a chatbot for multi-threaded analytic conversations. Using the chain-of-thoughts reasoning technique [WWS*22], the system leverages GPT-3.5 capabilities to map the user utterance into a visualization using Matplot and Seaborn libraries. A different approach is proposed in [CLM*22]. The authors discuss a method based on few-shot learning [WYKN20] at inference time on the Codex LLM model by OpenAI. The model is fed with natural language-SQL (NL2SQL) pairs examples and the user's natural language query to aid in task understanding. The result is then converted into Vega-Lite specifications using a rule-based approach [CW22]. Similarly, [MS23] presents a comparable study involving Codex, GPT-3, and ChatGPT. Like the previous work, the study lacks a comprehensive discussion of results and relies on a small number of evaluation samples. Finally, [TCD*23] recently introduced ChartGPT, a multi-step pipeline incorporating LLMs into various stages, breaking down the visualization generation problem into logical steps. The authors fine-tune FLAN-t5 [CHL*22] to align the model with the intended task. The evaluation is extensive in the number of tests but is still executed using a custom evaluation scheme. This problem is common even to the other reported papers, leading us to study deeply how a general framework could support LLM-generated visualizations. Focusing on evaluation efforts for generated visualizations, [CZW*23] presents an evaluation study focusing on GPT-3.4 and GPT-4. The study delves into multiple facets, such as data interpretation, visualization design, visual data exploration support, and insight communication. Another in-depth study is conducted in [KMB23], where the authors thoroughly explore the capabilities and limitations of ChatGPT in visualization tasks using a series of questions from the VisGuides forum. The study observes that ChatGPT performs similarly to human responses and, in some cases, even outperforms them. While these works contribute to evaluating LLM usage in existing data visualization pipelines, few have systematically studied the main characteristics needed to evaluate an AI-generated visualization. To fill this gap, this paper proposes a conceptual stack that includes automatic quantitative and human-based evaluation metrics and investigates its application to LLM-based visualizations.

3 The EvaLLM conceptual stack

EvaLLM is a conceptual stack to model the evaluation for LLM-generated visualization, as shown in Figure 1. Drawing inspiration from the ISO/OSI model [Zim80], EvaLLM involves abstract layers to evaluate specific visualization properties and derive a corresponding quality measure. From bottom to top, each layer transitions towards a higher level of abstraction. EvaLLM comprises five primary layers, each characterized by a research question that refers to a different step of the visualization creation process proposed in the literature (e.g., Munzner [Mun14]). Each layer is subdivided into levels, where the focus is directed explicitly toward assessing distinct visualization properties of the same layer. The rationale for this structure is to allow for an initial set of homogeneous categories specialized internally in further detail inside each of them (through levels), supporting a fine-grained evaluation of the generated visualization properties and better characterizing the LLMs' expressive power.

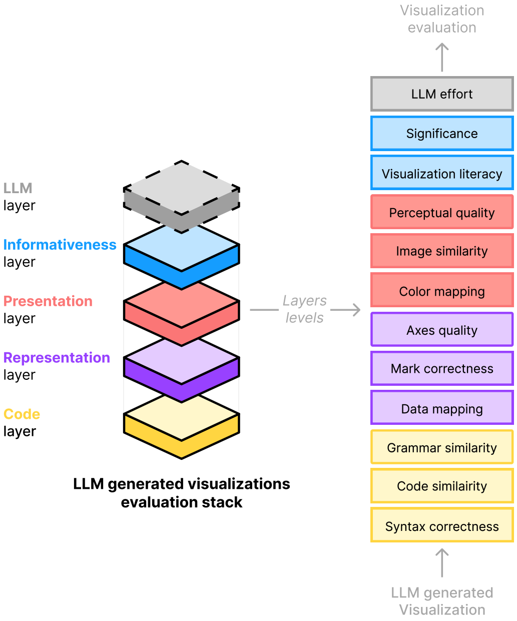

Figure 1: *The EvaLLM conceptual stack.*

Code layer: It lies at the base of the EvaLLM stack, playing a role in evaluating the fundamental structural properties of a visualization. The primary question that the Code layer seeks to address is *RQ.1: "Is the visualization consistent within the code environment it is created with?"*. This inquiry delves into the coherence of the visualization within the structure of the related environment (e.g., a generic programming language, a specialized one, or a grammar), checking whether the generated code aligns with the expected structure or evaluating which differences may be present. These differences are mostly syntactical, but they can already present interesting cases that strongly affect the final result quality (i.e., different syntactic structures that give birth to similar visual results or slightly different codes that produce very different results). In other words, the Code layer focuses on verifying the syntactical correctness and integrity of the visualization's underlying code, laying the groundwork for subsequent layers to delve into more nuanced aspects of evaluation.

Representation layer: Immediately above the Code layer, its focus shifts to the core properties of the visualization related to the representation rules defined by the literature [Spe01, Mun09]. We distilled from the literature three prominent aspects: the data mapping rules, the choice of the appropriate visual encodings (i.e., marks), and the representation properties of eventual axes or reference visual elements. The evaluation question behind this layer is *RQ.2 Are the data representation rules correctly generated in the visualization?*. We notice that already at this layer, it is not granted that some of these aspects are directly captured in the user query (i.e., the query may or may not include a direct reference to these aspects). At the same time, they are an integral part of a correct and accurate visualization. For this reason, at this layer, a more fine-grained and quantitative evaluation is needed concerning a coarse one based only, for example, on evaluating the correctness of the visual encoding. This evaluation should focus on quantitative distances between the generated and expected content.

Presentation layer: This layer is the third layer in the stack, and it assesses the data presentation quality of the visualization from the perceptual standpoint [War19]. This layer aims to model the design choices for perceptual aspects made by the LLM on the generated visualization to answer the question *RQ.3 Is the visualization correctly presented, comprehensible by a human user, and not giving a perceptual error or illusions hindering the human interpretation of its content?*. For example, it evaluates aspects such as the quality and appropriateness of the color mappings, the distinguishability of visual elements, or the perceptual comprehensibility of the visualized information.

Informativeness layer: This layer is responsible for measuring the more intrinsic quality of the visualization in terms of its insightfulness and adherence to best practices in visualization literacy, answering the question *RQ.4 Is the visualization insightful, and how well it supports the user in answering its information needs?*. This involves an evaluation that examines the capability of the visualization to convey meaningful insights concerning the user query and to be aligned with best practices of visualization literacy to ease the visualization understanding by the user as much as possible.

LLM layer: The LLM layer contributes twofold at the end of the stack. Its first goal is to evaluate the strategy for generating the specific visualization. Possible choices relate to single prompting the LLM, applying prompt engineering [WFH*23], or more sophisticated strategies like Chain-of-thought [WWS*22]. Evaluate the configuration elements to discern between different models (i.e., plain foundation model, zero-shot, fine-tuning, or a trained-from-scratch new model). In this way, it is possible to evaluate the effort for the generation process on top of the quality of the generated visualization. The second goal is evaluating the visualization by assessing its significance and adherence to best practices in the visualization literature, but this time, trying to measure the insightfulness of a visualization leveraging the LLM knowledge, answering the questions *RQ.5 What is the cost of the generated visualization?* and *RQ.6 Is the visualization insightful, based on the LLM knowledge?*.

4 Implementing layers: the EvaLLM levels

While the EvaLLM layers provide the overall structure of the fine-grained evaluation supported by the stack, they are still too coarse to be implemented. For this reason, EvaLLM presents a set of levels for each layer that better supports the implementation of the evaluation process and specifies how to interpret the evaluation results for each of them. The implementation of each level depends on the properties to measure and the used metrics. While the stack currently encompasses carefully designed key levels, ongoing advancements in the field warrant continuous evolution. Thus, further contributions can enrich the stack with additional levels, aligning with the progressive landscape of the visualization literature.

🟨 **Syntax correctness level** *What.* The syntax correctness level is designed to verify whether or not an LLM-generated visualization is consistent within the syntax structure it is created with, and it is executable in the related environment to render the visualization. *Why.* This level is pivotal in the stack to start evaluating a visualization. If the visualization code or the generated image has some structural errors and cannot be rendered, all the other levels are disabled in the evaluation process, and it stops here. *How.* Considering a grammar-based visualization, e.g., VegaLite, the syntax correctness level verifies that the generated visualization specification respects the grammar rules and, subsequently, can be executed. An additional check concerns the capability to render a visualization (e.g., the grammar specification could be correct for the data part but not presenting a rendering part).

🟨 **Code similarity level** *What.* The Code Similarity level processes the visualizations from the previous level, treating both the generated and the ground truth as code snippets and evaluating their similarity. *Why.* This step is needed as, most of the time, the same or similar visualizations can be generated through quite different code constructs. On the contrary, small code changes may produce big changes in the final visualization. LLMs are tested at this level for their capability to construct a visualization code similar to what a human user would do or to evaluate the differences and eventually the reasons behind them (e.g., better generalizability, better usage of coding practices, more efficient code). *How.* Haq and Caballero [HC21] propose an extensive review of more than 70 binary code similarity approaches that can be leveraged for this level depending on the chosen code environment. Another example is discussed in [LPX*18].

🟨 **Grammar similarity level** *What.* This level is tasked with measuring the similarity of the generated visualization's grammar structure compared to the ground truth structure. This level focuses mainly on the structure and less on the exact matching (e.g., identifier names). *Why.* The primary goal is to assess how effectively the model translates user requests into a correct grammar structure. This structure should align with the expected structure within the chosen grammar. Then, the assessment involves a comparison with the ground truth to check for efficiency in representation. Additionally, it aims to highlight structural differences in how a Language Model (LLM) represents the user query in the given grammar, comparing the results with the human ground truth or with other LLMs. *How.* A conceivable implementation would focus on comparing only the keywords of the generated grammar with their ground truth counterparts, omitting consideration of the values assigned to the keywords, which are reserved for higher levels. A practical implementation is represented by Playwright [Mic20].

🟪 **Data mapping level** *What.* This level measures how well the generated visualization encodes the right data from the dataset compared to the ground truth. It assesses whether the columns chosen from the dataset align with the ground truth and if the model adeptly maps them according to their correct types, for example, ordinal or temporal. *Why.* The main challenge is represented by ambiguous queries that could lead the model to select incorrect columns from the dataset, encode them in the wrong types on the axes, or hallucinate and select non-existing columns. *How.* A possible metric is the data axes accuracy proposed by Podo et al. [PPV24]. A similar approach to the previous, taken from the text-to-SQL field, is the Query match accuracy [XLS17].

🟪 **Mark correctness level** *What.* This level assesses the similarity between the visualization mark of the generated output and the corresponding ground truth and its usage of the chosen mark(s). *Why.* When a model produces a visualization, errors extend beyond simple mislabeling of the mark type, such as mistaking a bar for a line. The model may also introduce errors in how the mark is used in the overall structure of the visualization, creating what we labeled "visual hallucinations". An example is illustrated in Figure 2-g, where the model correctly recognizes the bar mark as the one to use. Still, it fabricates an uncommon and not usual representation of a bar chart. *How.* A potential implementation could adopt a hybrid methodology: automated checks could verify the accuracy of the mark while identifying hallucinations might necessitate human scrutiny.

🟪 **Axes quality level** *What.* The Axes Quality Level is structured to assess the quality of the axes' properties in a visualization. *Why.* The efficacy of a visualization hinges on its ability to clearly convey information to the users, with the axes playing a pivotal role. Optimal selection of these properties, such as axes orientation, scale, and ticks selection, is essential for delivering meaningful insights to users. *How.* One plausible implementation strategy entails an evaluation approach that compares the axes of the generated visualization with its ground truth. Alternatively, a set of rules defined by domain experts could be employed to assess the quality of the axes.

🟥 **Color mapping level** *What.* This level examines the efficacy of color usage in encoding data attributes to convey data features. *Why.* The selection of a color palette depends on the nature of the data being represented. Evaluating how the model employs colors based on the data type and the chosen mark is pivotal for effective visualization and correct user interpretation. *How.* For example, Szafir et al. [Sza17] provide a set of perceptual data results from crowd-sourced studies that could be used to create probabilistic models to provide support and evaluate the color properties of the visualizations. Another implementation by Liu et al. [LH18] provides methods to generate color recommendations.

🟥 **Image similarity level** *What.* This level involves a pixel-level comparison between the generated visualization and the ground truth. *Why.* Conducting a pixel-level comparison abstracts the assessment of the generated visualization from the specific generating environment or evaluates its characteristics by the final generated image, offering a direct and perceptual evaluation of how well the image aligns with the ground truth. *How.* An effective strategy entails applying computer vision techniques to quantify the structural similarity between the images, such as SSIM [RH08] or LPIPS [KHL19].

🟥 **Perceptual quality level** *What.* This level examines the visualization from a perceptual standpoint, focusing on ensuring that all perceptual properties are effectively encoded in the visualization and that it does not break the perceptual rules that are listed in visualization research (e.g., [War19]). *Why.* Leveraging principles of visual perception, such as position along a common scale, length, direction, angle, area, and color hue, can help the visual interpretation from a human user and design more informative graphics. It is important to consider the effectiveness of visual encoding, ensuring that the importance of the attribute matches the salience of the channel used for them and avoid perceptual pitfalls like watchdog effects or usage of wrong visual channels. *How.* The control could be based on a stop list of perceptual pitfalls, eventually organized by visual marks, to be tested against the generated visualization to check for their presence automatically. On the contrary, assessing a human assessor is crucial at this level, as it can help identify more high-level problems faster.

🟦 **Visualization literacy level** *What.* This evaluation level seeks to determine whether the model adhered to best practices in visualization literacy while generating the visualization. *Why.* After the model produces a visualization that excels in other previous evaluation criteria, it does not automatically imply optimal configuration. The visualization could be requested to comply with best practices tailored to fit the task or particular analysis on top of the visual representation and presentation choices. *How.* In this direction, different works have been proposed in the literature, such as the work by Boy et al. [BRBF14] that proposes a series of visualization best practices for line charts, bar charts, and scatterplots.

Another possible approach is discussed by Lee et al. [LKK16] that introduces a visualization literacy assessment test (Vlat) exploitable for assessing the compliance of LLM-based visualizations.

🟦 **Significance level** *What.* The significance layer represents the layer in the EvaLLM to measure the insightfulness [BO23] of a visualization. *Why.* The significance is essential for analyzing complex data, identifying patterns, and extracting valuable insights. By simplifying complex information and presenting it visually, decision-makers can make informed and effective decisions quickly and accurately. *How.* This level should be implemented by involving a human reviewer using a dedicated platform. Performing this task automatically is complex because the insightfulness lacks a mathematical formulation that could make the evaluation automatic. Recently, some work emerged in the Visual Analytics literature trying to mathematically formulate the relation between task support and insights generation, such as Suh et al. [SMWC23].

■ **LLM effort** *What.* This level focuses on assessing the effort of generating a visualization considering computational and methodological factors. Looking at the former, we refer to computational costs, the inference time (real-time versus quasi-real-time), and the models' size. Looking at the latter, we list plain prompting (e.g., a single prompt representing the full user query), variants of prompt engineering, Chain-of-thoughts [WWS*22] or by chaining results of multiple models or the same model multiple times. *Why.* Generating a visualization using an LLM is not only a matter of visualization quality but also of the strategy's performance, costs, and explanation [LRBB*23]. For instance, even if using the same model with two different learning strategies could still generate the same expected visualization, the computational costs could be extremely different. *How?* This level could be developed as a scoring function considering all the factors described to return a normalized effort score.

5 Use case

The chosen scenario involves evaluating the capabilities of Llama2-70b[†] to generate VegaLite visualizations from a given dataset and a user query. The dataset is a reduced version of NVBench dataset [LTL21a] that remains representative of its characteristics.

To generate responses, the model is queried with a user utterance mapped into a predefined prompt template based on the one proposed by Alpaca [TGZ*23]. The quality of the generated visualization is evaluated against ground truth.

The evaluation employs an automatic analysis for the code and representation layers, complemented by a human-based evaluation focusing on the presentation and informativeness layers.

Results Our analysis revealed interesting insights: out of the 50 samples, 34 visualizations were successfully generated without any errors in adherence to the VegaLite schema. The performance of the LLM was evaluated across three dimensions for the representation layer: mark type accuracy, x-axis field accuracy, and y-axis field accuracy. Notably, there was a slight dip in mark type accuracy, with the LLM correctly identifying the mark type in 29 out of 34 visualizations. Results highlighted the following insights:

[†] https://ai.meta.com/llama/

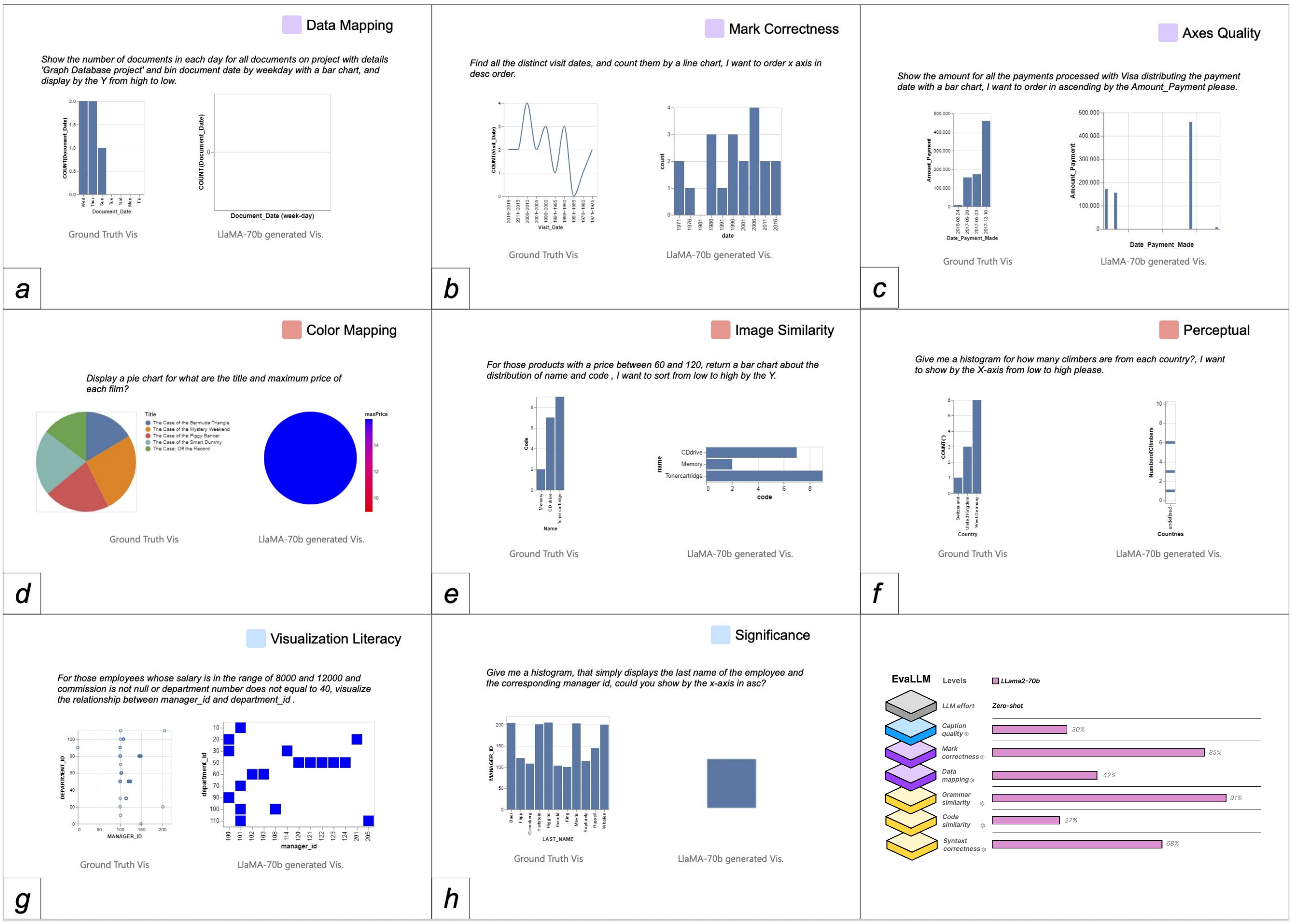

Figure 2: *Examples of wrong generation by LlaMA2-70b split by EvaLLM levels. For non-visual levels, the example is not reported.*

- **Inability of Incorporation of Data Values**: While the automated performance evaluation revealed a major dip in the model's performance, a following human evaluation uncovered a few more inabilities of the model. Such is the inability of the model to correctly understand and incorporate the data values in the visualization based on user query. In some cases, the model was observed generating no data values at all, as shown in Figure 2-[a,h]. A few other generated visualizations were found to have data values ignored but had data linked to a separate data file, which was quite strange behavior.
- **Largely Structured Prompts Ignored**: Investigating why the model was generating visualizations with no data values plotted at all unearthed some findings on the cause for the missing generation: in particular, in a few of the cases where the prompt structure was relatively simple but with a lot of data, the model sent back as response the prompt and sometimes just blank strings without any text or warning of some kind.
- **Low Visualization Significance**: In several visualizations generated by the model, there was little to no significance to the visualization in terms of the user query and data. Such examples can be seen, to different degrees, in Figure 2 [a,c,f,g,h].
- **Incorrect or missing Sorting**: In cases where the user query explicitly identified the nature of the sorting, the model was unable to understand it, resulting in unordered or incorrectly ordered data values. One instance of this can be seen in Figure 2-e.

6 Conlusion

This paper investigated the elements to consider when evaluating an LLM-based generated visualization in a comprehensive and fine-grained way. Those elements were condensed and structured formally into the proposed EvaLLM stack, the first proposal targeted at LLMs. Based on Llama2-70b, one use case shows the benefits and results obtained using the proposed evaluation stack and platform. We report as current limitations of this work that we plan to overcome in current working activities are the limited number of experiments (we plan to run the experiments on the full nvBench), coverage of visualization techniques, and deeper characterization of LLMs errors.

Proceedings published by Eurographics - The European Association for Computer Graphics.

References

[BO23] Battle L., Ottley A.: What do we mean when we say "insight"? a formal synthesis of existing theory. *IEEE Transactions on Visualization and Computer Graphics* (2023), 1–14. doi:10.1109/TVCG.2023.3326698. 4

[BRBF14] Boy J., Rensink R. A., Bertini E., Fekete J.-D.: A principled way of assessing visualization literacy. *IEEE transactions on visualization and computer graphics 20*, 12 (2014), 1963–1972. 4

[CHL*22] Chung H. W., Hou L., Longpre S., Zoph B., Tay Y., Fedus W., Li Y., Wang X., Dehghani M., Brahma S., et al.: Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416* (2022). 2

[CLM*22] Chen Y., Li R., Mac A., Xie T., Yu T., Wu E.: Nl2interface: Interactive visualization interface generation from natural language queries. *arXiv preprint arXiv:2209.08834* (2022). 1, 2

[CW22] Chen Y., Wu E.: Pi2: End-to-end interactive visualization interface generation from queries. In *Proceedings of the 2022 International Conference on Management of Data* (2022), pp. 1711–1725. 2

[CZW*23] Chen Z., Zhang C., Wang Q., Troidl J., Warchol S., Beyer J., Gehlenborg N., Pfister H.: Beyond generating code: Evaluating gpt on a data visualization course. *arXiv preprint arXiv:2306.02914* (2023). 2

[HC21] Haq I. U., Caballero J.: A survey of binary code similarity. *ACM Computing Surveys (CSUR) 54*, 3 (2021), 1–38. 3

[HC23] Hong M.-H., Crisan A.: Conversational ai threads for visualizing multidimensional datasets. *arXiv preprint arXiv:2311.05590* (2023). 2

[KHL19] Kettunen M., Härkönen E., Lehtinen J.: E-lpips: robust perceptual image similarity via random transformation ensembles. *arXiv preprint arXiv:1906.03973* (2019). 4

[KMB23] Kim N. W., Myers G., Bach B.: How good is chatgpt in giving advice on your visualization design? *arXiv preprint arXiv:2310.09617* (2023). 2

[LH18] Liu Y., Heer J.: Somewhere over the rainbow: An empirical assessment of quantitative colormaps. In *Proceedings of the 2018 CHI conference on human factors in computing systems* (2018), pp. 1–12. 4

[LKK16] Lee S., Kim S.-H., Kwon B. C.: Vlat: Development of a visualization literacy assessment test. *IEEE transactions on visualization and computer graphics 23*, 1 (2016), 551–560. 4

[LPX*18] Liu W., Peng X., Xing Z., Li J., Xie B., Zhao W.: Supporting exploratory code search with differencing and visualization. In *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)* (2018), IEEE, pp. 300–310. 3

[LRBB*23] La Rosa B., Blasilli G., Bourqui R., Auber D., Santucci G., Capobianco R., Bertini E., Giot R., Angelini M.: State of the art of visual analytics for explainable deep learning. *Computer Graphics Forum 42*, 1 (2023), 319–355. doi:https://doi.org/10.1111/cgf.14733. 4

[LTL21a] Luo Y., Tang J., Li G.: nvbench: A large-scale synthesized dataset for cross-domain natural language to visualization task. *arXiv preprint arXiv:2112.12926* (2021). 1, 4

[LTL*21b] Luo Y., Tang N., Li G., Tang J., Chai C., Qin X.: Natural language to visualization by neural machine translation. *IEEE Transactions on Visualization and Computer Graphics 28*, 1 (2021), 217–226. 2

[Mic20] Microsoft: Playwright, 2020. URL: https://playwright.dev/. 3

[MS23] Maddigan P., Susnjak T.: Chat2vis: Generating data visualisations via natural language using chatgpt, codex and gpt-3 large language models. *IEEE Access* (2023). 1, 2

[Mun09] Munzner T.: A nested model for visualization design and validation. *IEEE transactions on visualization and computer graphics 15*, 6 (2009), 921–928. 2

[Mun14] Munzner T.: *Visualization analysis and design*. CRC press, 2014. 2

[NSS20] Narechania A., Srinivasan A., Stasko J.: Nl4dv: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (2020), 369–379. 1

[PPV24] Podo L., Prenkaj B., Velardi P.: Agnostic visual recommendation systems: Open challenges and future directions. *IEEE Transactions on Visualization and Computer Graphics* (2024). 1, 3

[RH08] Rouse D. M., Hemami S. S.: Understanding and simplifying the structural similarity metric. In *2008 15th IEEE international conference on image processing* (2008), IEEE, pp. 1188–1191. 4

[SBT*16] Setlur V., Battersby S. E., Tory M., Gossweiler R., Chang A. X.: Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th annual symposium on user interface software and technology* (2016), pp. 365–377. 1

[SMWC23] Suh A., Mosca A., Wu E., Chang R.: A grammar of hypotheses for visualization, data, and analysis, 2023. arXiv:2204.14267. 4

[Spe01] Spence R.: *Information visualization*, vol. 1. Springer, 2001. 2

[SS23] Srinivasan A., Setlur V.: Bolt: A natural language interface for dashboard authoring. 1

[Sza17] Szafir D. A.: Modeling color difference for visualization design. *IEEE transactions on visualization and computer graphics 24*, 1 (2017), 392–401. 4

[TCD*23] Tian Y., Cui W., Deng D., Yi X., Yang Y., Zhang H., Wu Y.: Chartgpt: Leveraging llms to generate charts from abstract natural language. *arXiv preprint arXiv:2311.01920* (2023). 1, 2

[TGZ*23] Taori R., Gulrajani I., Zhang T., Dubois Y., Li X., Guestrin C., Liang P., Hashimoto T. B.: Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html 3*, 6 (2023), 7. 4

[War19] Ware C.: *Information visualization: perception for design*. Morgan Kaufmann, 2019. 3, 4

[WFH*23] White J., Fu Q., Hays S., Sandborn M., Olea C., Gilbert H., Elnashar A., Spencer-Smith J., Schmidt D. C.: A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382* (2023). 3

[WIL*23] Wu S., Irsoy O., Lu S., Dabravolski V., Dredze M., Gehrmann S., Kambadur P., Rosenberg D., Mann G.: Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564* (2023). 1

[WSM*18] Wang A., Singh A., Michael J., Hill F., Levy O., Bowman S. R.: Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2018). 1

[WWS*22] Wei J., Wang X., Schuurmans D., Bosma M., Xia F., Chi E., Le Q. V., Zhou D., et al.: Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems 35* (2022), 24824–24837. 2, 3, 4

[WYKN20] Wang Y., Yao Q., Kwok J. T., Ni L. M.: Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur) 53*, 3 (2020), 1–34. 1, 2

[XLS17] Xu X., Liu C., Song D.: Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436* (2017). 3

[ZCG*23] Zhong W., Cui R., Guo Y., Liang Y., Lu S., Wang Y., Saied A., Chen W., Duan N.: Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364* (2023). 1

[Zim80] Zimmermann H.: Osi reference model-the iso model of architecture for open systems interconnection. *IEEE Transactions on communications 28*, 4 (1980), 425–432. 2