Multi-Agent Data Visualization and Narrative Generation

Anton Wolter[*†]
Aarhus University

Georgios Vidalakis[*‡]
KTH Royal Institute of Technology

Michael Yu[*§]
KTH Royal Institute of Technology

Ankit Grover[*¶]
KTH Royal Institute of Technology

Vaishali Dhanoa[∥]
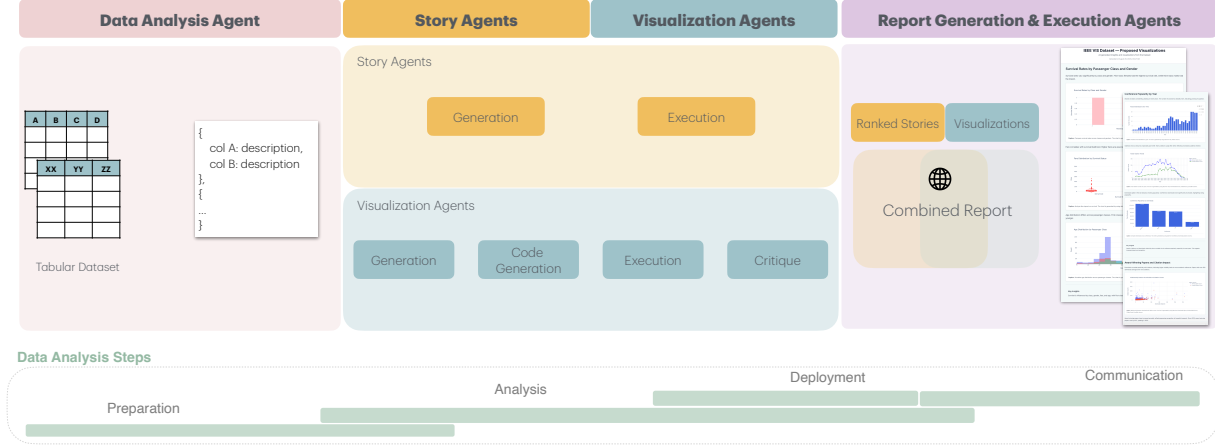Aarhus University
TU Wien

Figure 1: **Overview of agents in our workflow**: Agent roles involved in our system throughout the data analysis workflow, based on Crisan et al. [1]

ABSTRACT

Recent advancements in the field of AI agents have impacted the way we work, enabling greater automation and collaboration between humans and agents. In the data visualization field, multi-agent systems can be useful for employing agents throughout the entire data-to-communication pipeline. We present a lightweight multi-agent system that automates the data analysis workflow, from data exploration to generating coherent visual narratives for insight communication. Our approach combines a hybrid multi-agent architecture with deterministic components, strategically externalizing critical logic from LLMs to improve transparency and reliability. The system delivers granular, modular outputs that enable surgical modifications without full regeneration, supporting sustainable human-AI collaboration. We evaluated our system across 4 diverse datasets, demonstrating strong generalizability, narrative quality, and computational efficiency with minimal dependencies.

Index Terms: Visualization, Agentic Visualization, Agent Based Modelling.

1 INTRODUCTION & BACKGROUND

Data-driven reports are routinely used across a wide range of domains, from large companies to small organizations where they are deployed for a large audience or catered to individual professionals. Creating such reports requires careful planning, in terms of data analysis and thoughtful design to deliver reports that effectively capture the sentiment of the data and presents compelling results in a meaningful way. They must also be continuously updated and maintained as new domains and data emerge. Despite the promises of commercial self-service BI tools that claim to offer a fast path to building data-driven reports, much of the effort still lies in preparing and transforming data, followed by the additional work of designing effective visualizations [12].

With the rapid rise of AI agents, human-AI collaboration is getting reshaped across many domains, from co-creating or autonomously generating software code, creative writing, to even art production. These agents augment and amplify human capabilities [10] by offloading manually intensive tasks from humans and enabling them to focus on critical decision making points. In the field of data visualization, the concept of Agentic Visualization, proposed by Dhanoa et al [2], emphasizes the role of various agents employed the data to communication pipeline.

We draw inspiration from several such works [13, 14] to present our lightweight solution that generates web-based data-driven reports with the help of multiple agents[1]. In contrast to existing approaches [5, 3, 6] that use LLMs or other AI agents to create visualizations from data, we employ independent parallel agent processing with aggregation. Our compact approach automates the analysis of datasets and generates comprehensive data-driven reports that seamlessly integrate visualization and narrative explanations, as shown in Figure 1.We evaluate our approach for its generalizability across different datasets.

---

[*]Equal Contribution. Names listed in random order.

[†]e-mail: wol@cs.au.dk

[‡]e-mail: gvid@kth.se

[§]e-mail: mjyu@kth.se

[¶]e-mail: agrover@kth.se

[∥]e-mail: dhanoa@cs.au.dk

---

[1]Our workshop challenge submission report is available on the submission server. The source code is available on GitHub.

Table 1: **Agent roles.** Our lightweight solution consists of several agents with different tasks that collaborate.

Agent	Description	Input → Output
Data Analysis Agent	Analyzes tabular data to create structured metadata (domain-agnostic).	CSV data source → JSON schema with metadata
Story Generation Agent	Creates narrative ideas to support analysis goals.	JSON schema → Story ideas
Story Execution Agent	Ranks narratives and integrates visualizations with text.	Story ideas and visualizations → Curated stories
Visualization Generation Agent	Proposes meaningful visualizations for the dataset.	JSON schema and story context → Vis ideas
Code Generation Agent	Transforms visualization ideas into executable code.	Visualization ideas → Plotly code
Visualization Execution Agent	Executes code to render charts.	Plotly code → Visualizations
Visualization Critique Agent	Evaluates charts against design principles and handles errors.	Generated visualizations → Refined visualizations
Report Generation Agent	Selects and orders content for final output.	Curated stories → Ordered content
Report Execution Agent	Renders the final presentation format.	Selected content → HTML report
Monitoring Agent	Tracks system performance and resource usage.	System events → Performance metrics

2 TECHNICAL CONCEPT

The architecture follows a role-based design pattern, as described by Dhanoa et al. [2] where specialized agents handle distinct aspects of the visualization pipeline, from data preparation to visualization generation along with a narrative for the final report curation. At the same time we strive for a compact and lightweight solution. [8, 11] Therefore solution leverages a custom Python-based node architecture with multiprocessing to orchestrate workflows, enabling automated visual report generation with data-driven narratives from tabular datasets with minimal third party technical dependencies.

Our system adopts a hybrid architecture that strategically externalizes critical logic from LLMs to deterministic components, improving transparency and reliability. This composable design allows specialized modules to handle well-defined tasks while LLMs focus on high-level reasoning and narrative generation. For creating data-driven reports, it uses several agents that either use LLMs or deterministic heuristics to produce relevant output. We describe these agents and their tasks in table 1.

Plotly [4] serves as the primary charting library, supporting interactive visualizations that include line charts, bar plots, scatter plots, and heatmaps. The code generation pipeline creates Python scripts executed with error handling and retry mechanisms to ensure robustness. For structured agentic inputs and outputs, Pydantic [9] models are utilized. Jinja2 [7] templating generates HTML reports that combine narrative explanations with embedded visualizations. The template structure includes modular components for story sections, chart integration, and an optional metrics display using responsive design principles. The system enables monitoring of key metrics, such as story retention rate or LLM API calls with token usage, in order to provide transparency into the system's performance.

The implementation is generalized to analyze and generate reports for any tabular dataset. The dataset specification generation can be utilized for complementing different agent-based data analysis and communication workflows. In our prompts we do not provide few-shot examples, and only rely on dataset-agnostic instructions. Moreover, the prompts of the Visualization and Story agents can be easily adapted to different user groups. Finally, our implemented metrics collector can be used to collect granular feedback and analysis of performance for different datasets. Evaluation across 4 diverse datasets (418-4,111 rows) achieved 75% story retention for most datasets and 6-11 seconds per story runtime, validating generalizability and effectiveness. Each dataset yielded proper reports with relevant visualizations and coherent narratives, demonstrating the system's domain-agnostic capabilities.

3 DISCUSSION

Our role-based multi-agent approach demonstrates promising capabilities for automated visualization generation, but some limitations warrant discussion.

Strengths: The multi-agent structure enables specialized processing at each workflow stage, improving output quality through focused expertise. The orchestration layer ensures reliable state management and supports effective error recovery. The modular architecture facilitates component-wise optimization and testing. Notably, our system is lightweight: it relies on only one LLM dependency while being otherwise completely self-contained using standard popular dependencies from the Python ecosystem. This design choice makes it comparatively easy to port into other infrastructures without other changes.

Technical Limitations: The system relies heavily on LLM consistency for code generation, with occasional syntax errors requiring retry mechanisms. Missing semantic descriptions of the dataset columns can significantly affect the generated story ideas and the visualization interpretation.

Future Directions: While agents have demonstrated their capabilities in generating an automatic data-driven report, we believe that human-in-the-loop is required to validate and verify each intermediate step and ensure the validity of the results, especially in case of complex domain-specific datasets. Additionally, the report could benefit from interactive capabilities to enable reader-driven discovery through exploration much beyond the AI's narrative. Further exploration and research is also required to manage the resources effectively. To this end, we have incorporated a quantitative metrics for evaluation the failure rate of visualization generation, but it could be enhanced. We also believe that human intervention and feedback can significantly lower the error rates.

REFERENCES

[1] A. Crisan, B. Fiore-Gartland, and M. Tory. Passing the data baton : A retrospective analysis on data science work and workers. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1860–1870, 2021. doi: 10.1109/TVCG.2020.3030340

[2] V. Dhanoa, A. Wolter, G. Molina León, H.-J. Schulz, and N. Elmqvist. Agentic visualization: Extracting agent-based design patterns from visualization systems, 2025.

[3] V. Dibia. Lida: A tool for automatic generation of grammar-agnostic visualizations and infographics using large language models, 2023.

[4] P. T. Inc. Collaborative data science, 2015. Version used in this work: 5.3.x, Accessed: 2025-08-30.

[5] P. Maddigan and T. Susnjak. Chat2vis: Generating data visualisations via natural language using chatgpt, codex and gpt-3 large language models, 2023.

[6] A. Narechania, A. Srinivasan, and J. Stasko. Nl4dv: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):369–379, Feb. 2021. doi: 10.1109/TVCG.2020 .3030378

[7] Pallets. Jinja documentation (3.1.x), 2025. Version used in this work: 3.1.x, Accessed: 2025-08-30.

[8] D. L. Parnas. Designing software for ease of extension and contraction. *IEEE Transactions on Software Engineering*, SE-5(2):128–138, 1979. doi: 10.1109/TSE.1979.234169

[9] Pydantic. Pydantic: Data validation using python type annotations, 2025. Version used in this work: 2.11.x, Accessed: 2025-08-30.

[10] B. Shneiderman. *Human-centered AI*. Oxford University Press, 2022.

[11] W. P. Stevens, G. J. Myers, and L. L. Constantine. Structured design. *IBM Systems Journal*, 13(2):115–139, 1974. doi: 10.1147/sj. 132.0115

[12] C. Walchshofer, V. Dhanoa, M. Streit, and M. Meyer. Transitioning to a commercial dashboarding system: Socio-technical observations and opportunities. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):381–391, 2023.

[13] C. Wang, J. Thompson, and B. Lee. Data formulator: Ai-powered concept-driven visualization authoring. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1128–1138, 2024. doi: 10. 1109/TVCG.2023.3326585

[14] Y. Zhao, J. Wang, L. Xiang, X. Zhang, Z. Guo, C. Turkay, Y. Zhang, and S. Chen. Lightva: Lightweight visual analytics with llm agent-based task planning and execution. *IEEE Transactions on Visualization and Computer Graphics*, 2024.