PuddingBot: Creating data journalistic reports with AI agents

Dominikus Baur*
Munich University of Applied Sciences HM

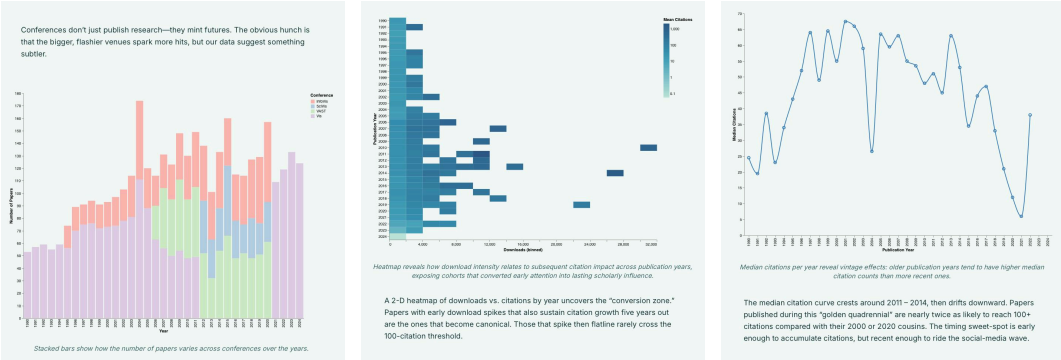

Figure 1: Sections from a *PuddingBot* output. The article structure is supported with narratively and aesthetically fitting charts.

Abstract

Data journalism teams produce engaging and entertaining dives into complex datasets. Their outputs are far above simple data reports and produce rich analyses - guided tours through the data with interactive charts - that work even for lay audiences. Given the right setup, LLM-based AI agents can produce similar outputs: *PuddingBot* is a first foray into automated data journalism for turning a given dataset into a web-based report. These automated reports follow a curiosity-driven narrative while highlighting aspects of the data as interactive, aesthetically pleasing data visualizations. In this paper, we explain the agent's design, its inner workings and implementation and discuss challenges as well as future extensions.

1 Introduction

Progress in web-based data visualization technology[2] as well as improving data literacy and new approaches to teaching journalism led to the rise of data journalism in the early 2010s. Impressive examples by publications like the New York Times (e.g., [1]) or The Pudding[2] provide easy-to-understand and entertaining deep dives into various topics and use interactive charts to make their points.

Creating such data journalism pieces, however, requires effort and includes in-depth data analysis, writing a gripping article as well as designing and developing web-based charts. Such diverse tasks require highly skilled teams of specialists. Automation would certainly be a welcome help in deadline-driven newsrooms.

In this paper, we present *PuddingBot* (see Figure 1 and example output[3]), an LLM-based agent system that autonomously creates aesthetically-pleasing data journalism articles, exploring and narrating arbitrary datasets. We view *PuddingBot* as an exploratory tool for personal data exploration as well as an idea generator for data journalists.

*e-mail: dominikus.baur@hm.edu

[1] https://www.nytimes.com/interactive/2015/03/19/upshot/3d-yield-curve-economic-growth.html

[2] https://pudding.cool

[3] https://www.visagent.org/api/output/d9ff0f3d-97ff-4cbd-adc5-89196d3f68df

2 Related work

Automated generation of data visualizations has been an active research topic long before the recent rise of generative AI. Prominent examples are the "Show Me" feature in Tableau[6] or the Voyager system[15]. These approaches rely on user-initiated queries to generate charts on the fly. Systems such as DATATALES[13] use generative AI to analyze charts and produce textual representations. In addition, there is also a growing body of work exploring human-machine collaboration for data storytelling (cf. [5]).

The recently introduced *canvas* feature[4] in ChatGPT lets users write and run code - including visualizations - but is geared towards standalone charts instead of full written reports.

The value of narrative visualizations have been extensively discussed in the literature since Segel and Heer's 2010 paper[11], yet full-on autonomous report generation is still scarce and restricted to specialized domains (e.g., [8]) or requires human goal-setting and intervention[12].

Work on explicitly generating data journalism is missing from the academic context. The Pudding team themselves experimented with Anthropic's Claude to create a data-driven visual story yet were not convinced in the end (their conclusion: "Do we feel replaceable? In short, not right now" [9]).

3 PuddingBot

PuddingBot is an LLM-based agent that mimics a data journalist's approach to writing a visual report. It works with arbitrary datasets, performs statistical analyses to arrive at insights and derives a narrative from them: it starts with a question about or a surprising observation on the data, presents this curiosity-driven narrative hook in the opening section and produces several sections containing text and accompanying interactive visualizations answering the original question while also providing an overview of the dataset. The complete workflow is autonomous and does not require any human intervention. While we used the VIS publications dataset [3] during development, the bot itself is dataset agnostic.

4 Implementation

The bot has been written in Python and generates HTML-based outputs with charts implemented in Vega Lite[10]. It uses a combina-

[4] https://openai.com/index/introducing-canvas/

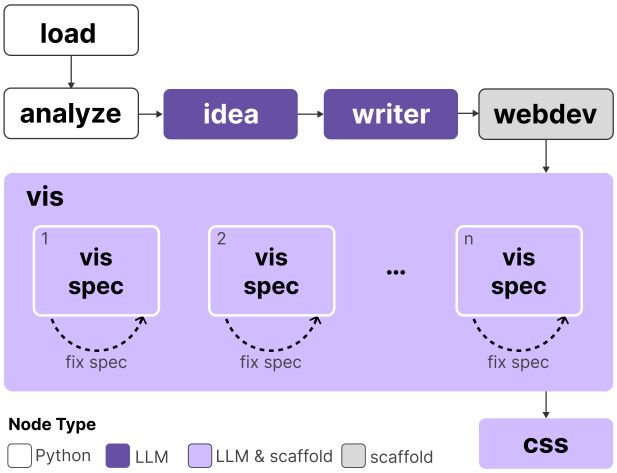

Figure 2: PuddingBot's node flow combines LLM queries with Python functions and libraries as well as HTML scaffolding.

tion of Python-internal functions and libraries, LLM queries, and HTML / JS / CSS scaffolding.

We're using the Pocket Flow framework[5], a minimal, 100-line framework for writing agents in Python, that provides a simple shared store as well as node execution and branching.

PuddingBot's overall workflow is generally linear, with analysis, followed by writing and implementation (see Figure 2):

The *load* and *analyze* nodes load the dataset from a given CSV-file and extract statistical key figures using `pandas`[7].

Idea creates a high-level, keypoint-based narrative around a specific insight or question and *writer* fleshes this out into a full Markdown[6] article. *Webdev* turns the Markdown code into an HTML skeleton using Python's built-in `markdown` library.

At this point, the HTML contains placeholders for visualizations, CSS- and Vega Lite styles. The *vis* node iterates linearly through all required visualizations and generates a Vega Lite specification for each of them. These specs are evaluated using Altair[14] and potential errors are either fixed or given up on after three tries. Working Vega Lite specs are integrated into the HTML skeleton with some scaffolding code for actually initializing the charts.

Finally, a prompt in the *css* node generates suitable CSS rules for a light-hearted, aesthetically pleasing color scheme, font styles and layouts which are then integrated into the HTML. Additionally, a second prompt turns this CSS style into a set of visual Vega Lite properties (`background, titleColor, gridColor, ...`) that are integrated with every visualization spec for consistency.

The result is stored as a self-contained HTML-file.

5 Challenges

One core insight during our development was that the best results came from a combination of native Python tools, HTML-scaffolding, and the inherently nondeterministic LLM outputs. Despite the leniency of web-based technologies towards syntax errors and irregularities, the internal parts of the agent still expect inputs and outputs in specific formats which some LLM-results might break. Therefore, our prompts usually ask for output in less error-prone machine-readable formats: *idea* formulates the general structure as YAML, *writer* outputs Markdown and the *viz spec* subtasks output Vega Lite specifications as JSON while the actual embedding in a `<script>` tag, loading of the required libraries etc. is all part of the HTML scaffold.

Similarly, simple regular expression-based error correction is applied to each LLM-result (for removing superfluous ```` ```javascript ```` or ```` ```css ```` tags for example) and Vega Lite specifications are checked with Altair[14], feeding potential errors back into the *fix spec* LLM subtasks. Yet, not all valid specs are sensible specs, so inappropriate choices such as year scales ranging from 0 to 2024 or formatting year labels with a thousand separator ("2,020") can appear (cf. Li et al.[4] for more examples).

6 Reflection

6.1 Keeping the agent data-agnostic

Our initial version contained a web search node for learning more about the dataset, but that would ramp up the non-determinism of the process even further. So now we're relying on the LLM's built-in understanding of the world (e.g., how academia and publishing works) to select interesting insights and create a coherent narrative.

6.2 Graphic and web design

A central aspect of data journalistic pieces is their look-and-feel. Simply relying on Vega Lite's default theme and a white background was too basic to recreate the best online visual essays. Therefore, both prompts and scaffolding are infused with best practices for design both from journalism itself (cf. The Pudding on their approach to visual storytelling[7]) as well as our own rich experience in creating data journalism online (e.g., [1] or [16]).

6.3 Context windows and sequential specialization

Aside from the improved robustness, the combination of Python-internal functions with LLMs was also required by the restricted context window of the LLMs. Feeding a large CSV-file into the LLM would quickly overwhelm said window and prevent reliable insights.

Also, the overall approach of a sequence of specialized nodes that are set up using role-based prompting (e.g., *"You are a data visualization developer..."* for the *vis* node) led to more reliable and interesting results than simply having a single LLM prompt producing a website based on the `pandas` analysis.

6.4 Model choice

Experiments with different (OpenAI) models showed large differences regarding quality both of writing and chart/style creation. Reasoning models such *o4-mini* or *o3* would usually fare better in creating working output than generalized models such as *GPT-4o*. Our work coincided with the release of *GPT-5*, with increased quality in writing and types of charts generated. Unfortunately, its reasoning version tends towards extremely long runtimes with frequent time-outs, so we're using *o3* instead.

7 Conclusion and Future Work

PuddingBot is an autonomous LLM-based agent that produces visual reports inspired by data journalism. During development we saw several avenues for improvements: the current `pandas` data analysis is the bottleneck for all following steps, so extending this would give the LLMs more insights to work with. Similarly, filtering out outliers would prevent the models from sometimes coming up with harebrained narratives that an actual data journalist would never arrive at (such as the one around the single paper with a length of 3608 pages). Additionally, a human-guided version of this with interaction with and input from a data journalist could lead to more interesting results.

[5] https://github.com/the-pocket/PocketFlow

[6] https://daringfireball.net/projects/markdown/

[7] https://pudding.cool/process/how-to-make-dope-shit-part-2/

REFERENCES

[1] D. Baur and A. Thudt. Full of themselves: An analysis of title drops in movies, July 2024. https://www.titledrops.net. 2

[2] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. 1

[3] P. Isenberg, F. Heimerl, S. Koch, T. Isenberg, P. Xu, C. Stolper, M. Sedlmair, J. Chen, T. Möller, and J. Stasko. vispubdata.org: A Metadata Collection about IEEE Visualization (VIS) Publications. *IEEE Transactions on Visualization and Computer Graphics*, 23, 2017. doi: 10.1109/TVCG.2016.2615308 1

[4] G. Li, X. Wang, G. Aodeng, S. Zheng, Y. Zhang, C. Ou, S. Wang, and C. H. Liu. Visualization generation with large language models: An evaluation. *arXiv preprint arXiv:2401.11255*, 2024. 2

[5] H. Li, Y. Wang, and H. Qu. Where are we so far? Understanding data storytelling tools from the perspective of Human-AI collaboration. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pp. 1–19, 2024. 1

[6] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007. 1

[7] W. McKinney et al. pandas: a foundational Python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011. 2

[8] H. Pu, X. Yang, J. Li, and R. Guo. Autorepo: A general framework for multimodal LLM-based automated construction reporting. *Expert Systems with Applications*, 255:124601, 2024. 1

[9] R. Samora and M. Pera-McGhee. Can an AI make a data-driven, visual story?, July 2024. https://pudding.cool/2024/07/ai/. 1

[10] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, 2016. 1

[11] E. Segel and J. Heer. Narrative visualization: Telling stories with data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1139–1148, 2010. 1

[12] D. Shi, X. Xu, F. Sun, Y. Shi, and N. Cao. Calliope: Automatic visual data story generation from a spreadsheet. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):453–463, 2020. 1

[13] N. Sultanum and A. Srinivasan. DATATALES: Investigating the use of Large Language Models for Authoring Data-Driven Articles. In *2023 IEEE Visualization and Visual Analytics (VIS)*, pp. 231–235, 2023. doi: 10.1109/VIS54172.2023.00055 1

[14] J. VanderPlas, B. Granger, J. Heer, D. Moritz, K. Wongsuphasawat, A. Satyanarayan, E. Lees, I. Timofeev, B. Welsh, and S. Sievert. Altair: interactive statistical visualizations for Python. *Journal of open source software*, 3(32):1057, 2018. 2

[15] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):649–658, 2015. 1

[16] World Bank Group. Atlas of Sustainable Development Goals 2023, June 2023. https://datatopics.worldbank.org/sdgatlas/. 2