Slide4N: Creating Presentation Slides from Computational Notebooks with Human-AI Collaboration

Fengjie Wang*
Sichuan University
Chengdu, China
wangfengjie@stu.scu.edu.cn

Xuye Liu*
University of Waterloo
Waterloo, ON, Canada
x827liu@uwaterloo.ca

Oujing Liu
University of Waterloo
Waterloo, ON, Canada
o3liu@uwaterloo.ca

Ali Neshati
University of Waterloo
Waterloo, ON, Canada
aneshati@uwaterloo.ca

Tengfei Ma
IBM Research
Yorktown Heights, NY, USA
tengfei.ma1@ibm.com

Min Zhu†
Sichuan University
Chengdu, China
zhumin@scu.edu.cn

Jian Zhao†
University of Waterloo
Waterloo, ON, Canada
jianzhao@uwaterloo.ca

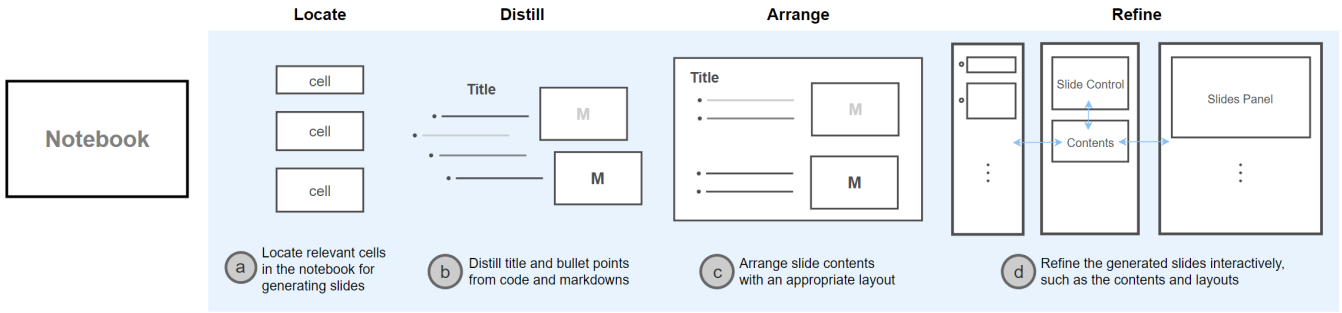

Figure 1: Slide4N is an interactive AI assistant for data scientists to create slides from computational notebooks. It takes user-selected cells as inputs and automatically generates slides as the basis for users to refine later. This is achieved through a human-AI collaborative approach enabled by the back-end and front-end of Slide4N via locating relevant cells for creating a slide (a), distilling information to generate slide title and bullet points (b), arranging generated slide contents with a suitable layout (c), and refining the generated slides with user customization, such as title, bullet points and layouts (d).

ABSTRACT

Data scientists often have to use other presentation tools (e.g., Microsoft PowerPoint) to create slides to communicate their analysis obtained using computational notebooks. Much tedious and repetitive work is needed to transfer the routines of notebooks (e.g., code, plots) to the presentable contents on slides (e.g., bullet points, figures). We propose a human-AI collaborative approach and operationalize it within Slide4N, an interactive AI assistant for data scientists to create slides from computational notebooks. Slide4N leverages advanced natural language processing techniques to distill key information from user-selected notebook cells and then renders them in appropriate slide layouts. The tool also provides intuitive interactions that allow further refinement and customization of the generated slides. We evaluated Slide4N with a two-part user study, where participants appreciated this human-AI collaborative approach compared to fully-manual or fully-automatic methods. The results also indicate the usefulness and effectiveness of Slide4N in slide creation tasks from notebooks.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; • **Computing methodologies** → **Natural language processing**; • **Applied computing** → *Document preparation.*

KEYWORDS

slides generation, computational notebooks, human-AI collaboration, natural language processing, data science.

*These authors contributed equally.
†Correspondence authors.

CHI '23, April 23–28, 2023, Hamburg, Germany

ACM Reference Format:
Fengjie Wang, Xuye Liu, Oujing Liu, Ali Neshati, Tengfei Ma, Min Zhu, and Jian Zhao. 2023. Slide4N: Creating Presentation Slides from Computational Notebooks with Human-AI Collaboration. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3544548.3580753

1 INTRODUCTION

Computational notebooks (e.g., Jupyter Notebook [28], JupyterLab [29]) have been becoming the single most effective means for data exploration and analysis in data science [35, 56]. They combine codes, notes, and outputs (e.g., tables and plots) within a single document, which provides expressive and interactive support for data scientists, such as literate programming, documentation, and insights sharing [33, 38, 51]. In almost every industry, with the growing complexity of the work, multiple collaborators are usually involved in the same analysis [47, 76]. The communication between technical members (e.g., model builders) and non-technical members (e.g., managers, domain experts, stakeholders) is critical and frequent [9, 37, 65]. Despite the sharing capability of notebooks, directly using them to present findings is challenging [7]. For example, a typical notebook tends to be long, poorly formatted, and interspersed with interim notes and findings [20], which is not intended for presentation [56], and can be difficult to understand for other members of a team, especially non-technical ones.

Data scientists are then forced to use other presentation tools (e.g., Google Slides, Microsoft PowerPoint) to create slides for the sake of communication [7]. This causes a phenomenon, we call, *analysis presentation divide*. Much tedious and repetitive work needs to be done to transfer contents between the computational notebooks and the slides [52], since they are fundamentally two different mediums. A few steps [14, 79] are required for data scientists to create a high-quality presentation from their analysis. When creating a slide around a topic in mind, relevant cells first need to be *located* and essential information needs to be *distilled* from the complex, messy, and loosely-connected code [37, 56]. Moreover, all kinds of information such as text, plots, tables, etc. need to be *arranged* in a suitable layout to provide a narrative. Lastly, the slide needs to be *refined* with customization to accommodate different domains and audiences [6, 60]. Data scientists need to frequently switch between the notebook environment and the presentation tool, with a lot of copying and pasting, which is error-prone. What is worse, any updates on the analysis (e.g., a change of resulting plots) would trigger the remaking of relevant slides.

There have been some attempts to address the above challenges. ToonNote [32] allows users to craft a data comic from a notebook by putting visualizations and annotations in a sequence of frames; however, it still requires a lot of manual operations and the results are quite different from a slide deck. Themisto [63] automatically generates short markdowns (i.e., titles) for code snippets in a notebook, but not in a ready-to-use presentation format. Some other researchers have attempted to automatically create slides from text articles [59] and scientific documents [14]. However, these methods cannot be directly applied to notebooks because of the complexity of their format which includes code, text, tables, plots, etc. The closest work to ours may be NB2Slides [79] that uses a template-based method to generate an entire slide deck from a notebook. Nonetheless, it requires very restricted input notebooks that need to have few excess code cells, high-quality documentation, and a complete data science workflow, because of the template-based approach; these requirements are difficult to achieve in data scientists' day-to-day activities.

In this paper, we take a human-AI collaborative approach to augment the aforementioned slide creation workflow from a notebook by data scientists. We design and develop Slide4N (Figure 1), an interactive AI assistant for data scientists to create slides from computational notebooks, which is built within JupyterLab to provide a seamless user experience. To create a slide, a data scientist just *locates* related cells (including code and markdowns), and then Slide4N automatically *distills* key information to generate useful title and bullet points on the slide, using advanced natural language processing (NLP) techniques. Moreover, Slide4N algorithmically *arranges* the contents in groups and further displays them with appropriate layouts. It also provides a set of easy-to-learn interactions to allow for *refining* the generated slide with user customization. High-level human input to the Slide4N automation is further allowed for refinement and customization, such as showing more or fewer details on slides. A data scientist can iteratively select cells and create slides one by one to form a presentation. The slides are fully linked with the computational notebook, allowing for effortless synchronization between the analysis and the presentation. Our human-AI collaborative approach is versatile and dynamic, which does not require an entire analytical workflow completed in the notebook, or many restrictions on the quality of code and documentation. The system automates the tedious parts of slide creation, leverages suitable user inputs, and produces results in a commonly-used presentation style.

We evaluate Slide4N by conducting a two-part study, where in Part 1 participants ($N = 12$) used a provided Kaggle notebook to create presentation slides, and in Part 2 participants ($N = 6$) used their own notebooks to do the same task. We also invited the six participants in Part 2 and two other experts to rate all the created slides on five aspects (e.g., satisfaction, clarity, organization, etc.), and the results demonstrate that Slide4N could help participants generate high-quality slides. Moreover, questionnaire responses and qualitative feedback from the participants indicate the effectiveness and usefulness of the tool. In general, participants appreciated such a human-AI collaborative approach compared to fully-manual or fully-automatic methods. We also distill some design implications from our study to inform the future design of such tools.

In summary, our contributions in this paper include:

- An analytical pipeline that enables interactive and iterative slides creation from computational notebooks powered by recent NLP techniques;
- A human-AI collaborative tool, Slide4N, built within JupyterLab, that assists data scientists in creating presentation slides with a seamless experience; and
- An evaluation of Slide4N that illustrates the value of our approach and inspires the design of similar tools.

2 RELATED WORK

2.1 Presentation Support for Computational Notebooks

Computational notebooks (e.g., Jupyter Notebook [28], JupyterLab [29]) seamlessly weave code, documentation, and outputs (e.g., tables, plots) together, in the form of flexible building blocks [38], which makes it easy for interactive and exploratory programming [51]. With these features, data scientists can rapidly iterate through chunks of code and inspect intermediate results during exploration, which is central to their workflow [33]. Besides, the interleaving code and markdown cells in notebooks can improve the readability of the analysis procedure and support the communication with collaborators or stakeholders [34, 36, 50, 54, 63, 64].

While computational notebooks have become the most effective tool for data scientists, they still have several drawbacks, especially for the purpose of communication [7, 33, 56]. Due to the complexity of analysis, a data science problem requires collaboration among members with various skill levels and backgrounds [46, 47], and thus team communication is frequent [9, 37, 65]. Through an online survey, Zhang et al. [76] confirmed that data science teams are characterized as extremely collaborative with high demands on communication with various stakeholders. Chattopadhyay et al. [7] summarized nine pain points for computational notebooks, where sharing and collaborating is one of them. Data scientists have to use other tools (e.g., Google Slides, Microsoft PowerPoint) to make presentation materials (e.g., slides), causing the analysis presentation divide. This is because creating slides from notebooks is a laborious, repetitive, and error-prone process, and the disconnection between the two different tools takes away most of the benefits of notebooks such as flexibility.

To address this problem, one body of research aims to improve the readability of notebooks by simplifying them. Andrew et al. [20] proposed code gathering techniques to reorganize related code cells and generate more readable, cleaner notebooks. StickyLand [71] allows users to freely organize their code, filling the gap between the linear presentation of code and the non-linear process of exploratory data analysis with notebooks. Adam et al. [55] proposed a technique that folds code chunks with annotations to aid the navigation and comprehension of notebooks. However, these tools still produce notebooks/code and do not generate presentable contents in the form that allows non-technical team members to easily understand the analysis, such as slides.

Another body of research attempts to further close the gap between analysis and presentation by supporting slide generation directly from notebooks. For example, RISE [1], a Jupyter Notebook extension, turns selected notebook cells into slides, which can be exported to PDFs using nbconvert [30], a built-in functionality in Jupyter Notebook. However, different from our approach, RISE only transforms the notebook code and markdowns to the slides as is. It does not analyze the contents or distill key points for non-technical members as Slide4N does, which is required for collaboration in a data science team [56]. ToonNote [32], while not supporting slide generation, allows users to curate data comics to improve comprehension, which is a sequence of frames of plots and annotations extracted from the outputs and markdowns of a notebook. But it still requires many manual operations and configurations for generating a data comic, whereas Slide4N leverages advanced NLP and analytical techniques to automate much of the process.

Recently, NB2Slides [79] applies a template-based approach by retrieving relevant contents from a notebook to fill in the components of the template using methods inspired by NBSearch [40]. While NB2Slides allows for generating a slide deck with one click, it requires the input notebook to have very few excess cells, high-quality documentation, and a complete data science workflow (otherwise blank slides are generated). This is difficult to achieve in practice due to the exploratory nature of data analysis with notebooks; also, following a prescribed workflow to create slides to some extent constrains data scientists from freely expressing their ideas. The system also lacks adequate user interaction support for the generation process. It automatically groups cells in a notebook to generate corresponding slides, which is difficult for users to modify if not satisfied. Our tool, Slide4N, loosens such restrictions on input notebooks using a more dynamic human-AI collaborative approach that allows data scientists to select cells of interest to generate and refine slides one by one. Further, unlike NB2Slides, our tool supports human guidance for the topic, whether to automatically merge relevant cells and the levels of details in slide generation as well as dynamic synchronization of notebook contents and slide components, further closing the analysis presentation divide.

2.2 Slides and Code Documentation Generation

Human-readable explanations are essential for data scientists to present their findings [75]. For instance, Hou et al. [25] reported a study showing that data science volunteers and domain experts in civic data hackathons sometimes require collaborative tools to help translate data science technical language and other domain languages. In the business domain, Doris et al. [75] also reported that most data scientists need to present their findings to other stakeholders with the help of a human-readable and well-presented slide deck. They prefer to share slides or PDFs instead of code because most of the clients or some analysts in the business team do not have related programming skills [33]. Even though they can use other media like detailed documentation [57] or a model factsheet [23], stakeholders prefer a presentation that allows them to visually convey their findings and reveal domain-specific knowledge [37]. For example, Piorkowski et al. [52] reported that data scientists usually prepare PowerPoint slides to explain the concepts of precision and recall to the stakeholder team, but they were unable to sufficiently map those concepts to the business problem being solved. Therefore, in our work, we focus on helping data science workers draft their work into other domain languages and presenting it in a slide deck.

Several attempts have been made to automatically generate slides from documents, while not from code. Fu et al. [14] introduced a method of generating presentation slides by using a multimodal document with text and figures. They proposed a hierarchical sequence-to-sequence model to explore the internal structure of the documents and slides and consolidate paraphrasing and layout prediction modules for slide generation. Sun et al. [59] presented a query-based document-to-slide method implementing mutual learning based on sentence selection. They proposed an interactive

model in which users enter a short text as the slide title and then it is used to select the appropriate sentences from the paper by using a Dense Vector IR module. Hu et al. [26] proposed the PPsGen system to select the most important sentences from a given publication by using the Support Vector Regression model. However, these works only focus on slide generation for scientific documents, which are fundamentally different from computational notebooks that combine code, markdowns, and execution outputs (e.g., tables, plots) in a single document.

On a related topic, code documentation generation is becoming more necessary in the software and data science areas. One of the most representative neural architectures is the Transformer [62], which is also a framework we use in this work to generate related titles and bullet points on slides. Some emerging work (e.g., CodeTrans [11], PyMT5 [8]) used the T5 framework for the code documentation generation, but they only trained it with limited generation tasks and ignored characteristics from the code including the identifiers or the structure of the code. CuBERT [31], CodeBERT [13], and CodeT5 [69] are three innovative models for code documentation generation. CuBERT implements the BERT framework to get code-specific characteristics and achieves good performance. CodeBERT uses a standard mask language modeling and replaced token detection to learn the code representation. The current state-of-art model, CodeT5 [69], proposes a novel identifier-aware pre-training task allowing the model to recognize the identifiers and recover them when they are masked.

In addition to the generation of user-written comments, Liu et al. [44] proposed HAConvGNN, a hierarchical-attention-based GNN model, to generate high-level documentation for multiple notebook cells, which inspired title generation in our design. Another way to train a title generation model is to use a pre-trained advanced model and fine-tune it based on the collected dataset [45]. There are also some works using transformer-based approaches to generate titles [43, 77]. But they only focus on generating titles for posts from Stack Overflow, which is not suitable for code-related tasks. A recent work, Themisto [67], allows data scientists to interactively generate documentation from code cells using three different approaches: deep-learning based, query based, and user-prompt based. But it only supports technical users while writing the code, not for the goal of presenting their analysis with slides. In our work, Slide4N leverages CodeT5 in its backend analytical pipeline to generate bullet points on slides from user-selected code cells. When generating titles on slides, Slide4N leverages HAConVGNN and a fine-tuned T5-based model to provide title candidates.

2.3 AI-Infused Systems for Data Science

Data science is a labor-intensive field consisting of many investigative tasks such as exploratory data analysis, model development, and hypothesis verification. To improve the efficiency of data scientists, researchers developed many AutoML techniques to automatically process data, train models, perform feature addition, and other data science tasks [15]. In addition, many IT companies (e.g., H2O [39], Azure Machine Learning Studio [4], Google Cloud AutoML [17]) developed some AutoML applications to allow both technical and non-technical users to perform general data science tasks and get insights in a faster and easier way.

There are also some interactive tools that use AutoML to help users do data exploration [42, 61]. Tsiakmaki et al. [61] utilized AutoML to predict students' learning outcomes based on their participation in online learning platforms. Liu et al. [42] incorporated AutoML algorithms to do hyper-parameter searches for various kinds of genomic profiling data. However, most of the AutoML techniques only focus on the main data science tasks such as data processing, feature engineering, model selection, and model building, overlooking the presentation support in the whole workflow, which is our focus in this paper. Further, while fully automatic systems are powerful, there is still much deficiency due to errors and sub-optimal performance of the models. Thus, we leverage a human-AI collaborative approach to automate parts of the slide generation process while allowing users to intervene for achieving better results overall.

Moreover, some interactive AI-Infused tools have been created with friendly integration with the computational notebook setting. Wrex [10] is a Jupyter Notebook extension that supports data transformations with a programming-by-example method. B2 [74] treats data queries as a shared representation between the code and interactive visualizations allowing users to easily move from code to visualization and vice-versa. BURRITO [18] automatically infers a researcher's computational activities by capturing and displaying code outputs with notes. ATENA [3] is a system that takes an input dataset and automatically generates a compelling exploration session, displayed in an EDA notebook. EDAssistant [41] develops an extension to support exploratory data analysis by recommending useful APIs and searching code in the Jupyter Notebook setting. NBSearch [40] provides a semantic code query to help data scientists find relevant cells in a large notebook corpus. This idea inspires our design of Slide4N on auto-merging relevant cells into groups to offer a more concise slide generation and also render the slide layout according to the cell groups. In this work, we aim to employ a human-AI collaborative approach for slide creation, instead of a fully automated system. This way, it provides much flexibility by combining the power of machine automation and human creativity.

3 SLIDE4N DESIGN

In this section, we first describe the design goals that guide the development of Slide4N, then provide an overview of the system, and finally demonstrate the usage of Slide4N with a simple scenario.

3.1 Design Goals

We distill the following design goals to inform our development of Slide4N. These design goals are mainly extracted from the literature and the common challenges of data scientists in making presentation slides based on their notebooks.

G1: Link the notebook and presentation slides. Computational notebooks tend to be messy, poorly formatted, and less documented, which are hard for other members of a team to understand or even one's future self [20, 63]. To facilitate communication, data scientists are forced to transfer contents from the notebooks to the slides using other presentation tools (e.g., Microsoft PowerPoint) [7, 52]. But the exploratory and analytical data science work often requires rapid iterations, which results in frequent updates of presentation slides based on code in notebooks. Thus, the system

should interactively link notebook cells and slides to maintain the association and provenance between them [21, 64], which can also provide technical context to data scientists when making slides.

G2: Help locate relevant cells in the notebook for creating slides. While the code in traditional software is linear, the code cells in notebooks are relatively independent; they can be executed in arbitrary order and executed multiple times. This results in loosely connected codes in notebooks [20, 58, 72], so the cells before or after one cell may not be necessarily connected. On the contrary, when presenting the analysis in the notebook, the contents need to be linear, forming a narrative to allow a better comprehension. Each individual slide is usually based on a set of tightly connected cells, and thus there is a gap between the cells in the notebook and the information needed to create a slide. Therefore, the system should provide an easy way for users to locate these cells, so that they can gain sufficient information to create a slide.

G3: Assist with distilling key information from code and markdowns. Readers seldom have an interest in reading code when trying to understand the notebook at a high level, even for the closest collaborators [55]. Thus, a presentation slide typically does not contain any raw code; instead, it should display contents that are more human-readable, self-explanatory, and concise, such as in titles, bullet points, and pictures/tables. However, it is usually time-consuming to manually digest and summarize the key information from a lengthy, messy, and poorly-documented notebook. Thus, the system should enhance this information extraction and summarization process with automation.

G4: Arrange slide contents in a logical and visually pleasing manner. Unlike the notebook code and markdown cells that are loosely connected, the contents (e.g., headlines, points, and pictures) on a slide are often tightly interrelated [2, 70]. Properly organizing them in groups and displaying them with an appropriate layout can help structure the delivered information, create a narrative flow, and aid understanding [70]. Even with well-curated information, manual organization and layout of the slide contents are tedious and can lead to inconsistent slide styles that raise the cognitive burden on readers. Most presentation tools (e.g., Microsoft PowerPoint) provide several layout templates to mitigate this issue. However, data scientists still need to manually place the contents to certain layout components. Therefore, the system should take the relationship of the presenting information into account and automatically arrange the contents logically and elegantly.

G5: Support necessary human intervention for slide generation. Automating the slide generation is convenient, but the quality of the generated slides (outputs) cannot be guaranteed due to known and unknown limitations of machine learning models [5, 80]. Also, for fully-automated systems, there usually exists little control for users during the entire generation process, which could decrease the overall efficiency when the outputs do not meet the user's expectation [14, 59, 79]. A better solution is to place humans in the loop, enable human-AI collaboration, and balance the trade-off between automation and human agency [22, 48]. Therefore, the system should provide intuitive user interactions to allow for the right amount of control over the slide generation model, such as selecting the input, tweaking high-level model parameters, and adjusting generated slides.

3.2 System Overview

Based on the aforementioned design goals, we developed Slide4N, an interactive and intelligent system that supports data scientists in creating slides from computational notebooks with human-AI collaboration. An overview of the system architecture is shown in Figure 1, which includes two parts: an NLP-enhanced back-end and a front-end user interface, working together to fill in the gap between analysis and presentation needs in data science. Details of each component will be introduced in the following sections. Here, we provide a high-level description of the system.

In particular, based on the user-selected cell(s), the back-end computes cell similarity to help data scientists locate relevant cells (G2). It then analyzes the contents of the selected cells and extracts key information to generate meaningful slide titles, bullet points, and other contents such as plots (G3). This can provide a good "first cut" for data scientists to describe the work in selected cells. The back-end further organizes these contents based on cell relevance and suggests an appropriate layout for rendering the slide (G4).

With the support of back-end, the front-end user interface is implemented as a JupyterLab extension to provide a seamless experience for data scientists to create slides from computational notebooks. As shown in Figure 2, Slide4N can be used together with the main JupyterLab notebook window side-by-side, which consists of four interactively-linked components: (a) a *Notebook Overview* that visually summarizes the code and markdown cells in a notebook and their relationships (G1), (b) a *Control Panel* that allows users to provide high-level guidance to the slide generation (G5), (c) a *Navigation View* that displays the outline of the working slide deck, and (d) a *Slides Panel* that supports interactive customization and refinement of the generated slides (G5).

3.3 Usage Scenario

Here, we use a simple scenario to demonstrate an interactive iterative workflow of creating slides from a computational notebook with Slide4N. Suppose that Crystal is a data scientist who works in a real-estate startup. She uses JupyterLab to explore and analyze a dataset of housing prices in Toronto, Canada. She has finished an initial analysis and drawn some conclusions, now she needs to present her results to other stakeholders consisting of a mixture of technical and non-technical audiences such as her fellow data scientists, program managers, and business people. Moreover, she needs to produce a deck of slides in a relatively short amount of time as required by her manager.

Crystal thus opens her notebook in JupyterLab and clicks the "Slide4N" button on the toolbar to launch Slide4N. She then puts the two windows side by side to start to create slides based on her notebook (Figure 2A). First, she wants to create a slide to introduce the dataset used in the analysis, which is beneficial for people who are not familiar with her project. She clicks the "AI" button on *Control Panel* (Figure 2C) that creates an empty slide on *Slides Panel* (Figure 2E) with some necessary tips on it: select relevant cells, configure the parameters, and refine the generated slide. Following the tips, she goes to the *Notebook Overview* (Figure 2B) to browse the entire notebook. When hovering over the rectangles, which represent code and markdown cells, a tooltip is shown to display the basic information about the corresponding cell (Figure 2B1).

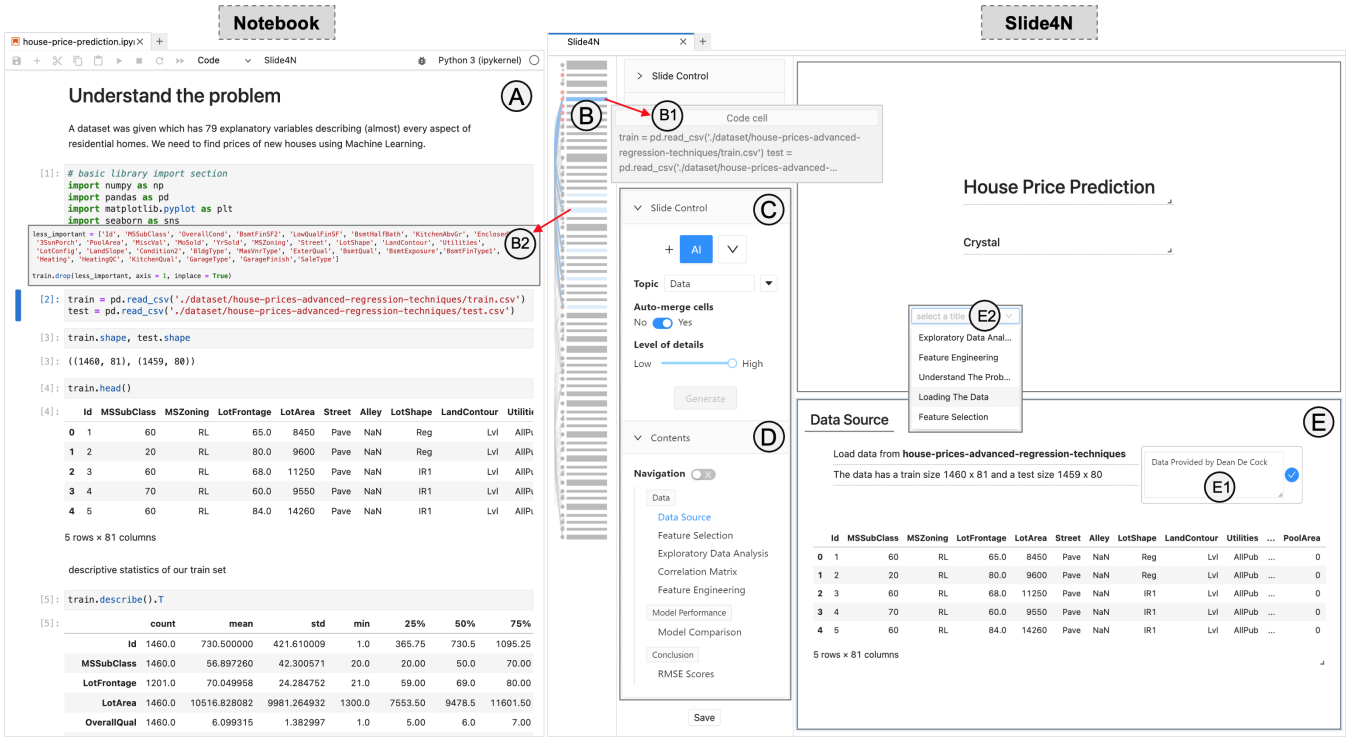

Figure 2: Slide4N is an interactive AI assistant for data scientists to create slides from computational notebooks, which can be used side-by-side with JupyterLab (A). The front-end user interface of Slide4N consists of four interactively linked components: a *Notebook Overview* (B) that visually summarizes all the cells in a notebook and their relationships, a *Control Panel* (C) that allows users to provide high-level guidance to the slide generation, a *Navigation View* (D) that displays the outline of the working slide deck, and a *Slides Panel* (E) that supports interactive customization and refinement of the generated slides.

She clicks a cell that loads data which she thinks is quite relevant to the slide in her mind, and then a set of curves appear on *Notebook Overview* to indicate the relevant cells to the selected one based on shared variables. These relevant cells are also color-coded with shades of blue to indicate the extent of relevance. Meanwhile, Slide4N automatically scrolls down the notebook window to the actual cell, so that Crystal can view the detailed code. She then finds some cells nearby that are related to the topic of the slide to create, and selects them.

After, Crystal moves to the *Control Panel* (Figure 2C) to configure the automatic slide generation at a high level. The selected cells are all about data, so she sets the "Topic" to Data. Considering that many cells are selected and they are messy, she also activates "Auto-merge cells" which can help her merge relevant cells. Finally, she sets the "Level of details" to the highest because she wants a more detailed version of the slide contents, which would be easier for others to understand. After this simple configuration, she clicks "Generate" and within seconds, a slide with generated title, bullet points, and plots is rendered on *Slides Panel* (Figure 2E). These contents are also nicely laid out.

However, Crystal thinks the generated title doesn't fit her style, so she rephrases it. She is also not satisfied with the first generated bullet point, so she slightly changes it. In addition, she adds another bullet point (Figure 2E1) to describe the source of the data, which is not exhibited in any cell of her notebook, but now she thinks it is nice to cite the source for a presentation. She further rearranges the slide contents based on the originally suggested layout by Slide4N. Crystal is now satisfied with the slide and moves on to creating other slides in a similar manner. The system is versatile enough to analyze the relationships among the selected cells to extract key information and suggest different slide layouts based on it. With the assistance of Slide4N, Crystal quickly creates a slide deck for her presentation and experts it for the meeting.

4 SLIDE4N

The scenario above is made possible in Slide4N through a series of operations (Figure 1), including locate, distill, arrange, and refine, which are detailed below.

4.1 Locate: Finding Relevant Cells

The Slide4N back-end maintains a cell relevance graph which is built upon the most up-to-date cell information captured by the front-end (G2). From each code cell, we extract structural and content level granularity. The usage of the abstract syntax tree provides levels of context to the inherent nature of programming languages. Our algorithm characterizes the semantic coherence between cells by aggregating the percentage of overlapping attributes and identifiers between code cells. For pairwise relevance calculation, we define

the relevance score as:

$$S = 1 - (1 - \frac{x}{a}) \cdot (1 - \frac{x}{b}) \quad (1)$$

where a denotes the number of variables in the source code, b denotes the number of variables in the destination code cells, and x denotes the shared number of variables between source and destination code cells. The score ranges from 0 to 1 where its maximum is achieved when variables from either of the code cell are a subset of the other one, and the minimum is achieved when variables from both cells are strictly exclusive to each other. Following the formula, the back-end composes a weighted, undirected, and complete graph of the notebook. The data is sent to the front-end to render in the *Notebook Overview* that indicates all the cell relevance. For example, in Figure 2B, rectangles with shades of blue indicate relevant cells. Figure 2B1 is selected by the user (the source code), B2 is a relevant cell (the destination code), and they share the "train" variable.

The *Notebook Overview* (Figure 2B) concisely displays the cells of the notebook and their states, linking the notebook and Slide4N (G1, G2), allowing a user to select cells used for later slide generation and displaying the cells associated with the slide. Each cell of the notebook is visualized as a rectangle with equal width. The more content the cell contains the higher the rectangle is, and the rectangles are placed from top to bottom to align with the order of the cells in the notebook. The dot on the left of each rectangle reflects the state of a cell with colors, including default (gray), focused (blue), and selected (pink); and a red border indicates the cell being updated. For markdown cells, there is a # below the dot, so that users can quickly distinguish different types of cells.

Several basic interactions are provided to facilitate the navigation of the notebook. When the user hovers over a cell, a tooltip pops up and gives a quick overview of the cell content, including the cell type, the first three rows of code, and the output plots or tables (if any). Also, the related cells sharing variables and functions are connected by blue arcs. When the user clicks on a cell, all the related cells are highlighted in blue (if any). A darker shade means greater relevance, and the current cell is always the darkest. At the same time, the notebook window jumps to the corresponding actual cell.

4.2 Distill: Generating Contents from Cells

Based on the selected cells, the back-end of Slide4N generates two types of content for each slide: titles and a set of bullet points(G3).

Title Generation. In total, four or five slide title candidates are recommended to the users using various methods, because we want to provide users with different detail levels of title recommendations and ensure the usability of the recommended titles. For example, Figure 2E2 shows the title candidates in a list, where the first two, third, and last two candidates are from the first, second and last methods, respectively. The fourth candidate is usually more accurate so it's selected by default.

First, the Slide4N back-end uses five topics (tags)—Introduction, Data, Model, Model Performance, and Conclusion—proposed in NB2Slide to generate two candidates, because these topics reflect the main stages of a data science analysis workflow [79] that can be the seeds of suitable summary titles. But compared to NB2Slide, we provide different titles for each topic after conducting a think-aloud formative design session with three data engineers, as shown in Table 1. Based on the user-specified topic in *Control Panel* (Figure 2C "Topic"), two candidates are selected by using the SequenceMatcher [1] from the Difflib library to find the text correlation between the title and the selected cells.

The third candidate title is generated from markdown cells. If there are markdown cells selected by the user and they contain hashtags (indicating user-documented titles), the system extracts this hash-tagged sentence and processes it for generating the slide title candidate. However, if no markdown cell is selected, the title is not generated.

The last two candidate titles are generated by two neural-network models. We employ two neural-network models, each generating a candidate title that more specifically summarizes the selected code. One model is HAConvGNN [44, 66] and the other is the T5-Base model [53] fine-tuned by our own collected notebook database. Even though high-level documentation might be too detailed sometimes for the title, users still need this kind of title for an in-depth presentation. Inspired by Mastropaolo et al. [45] who demonstrated a good performance after fine-tuning a T5-based model in code-related tasks, we also fine-tuned a T5-base model to generate other candidate titles. The reason for choosing two models is that we find sometimes the recommendation of one model is not satisfactory, and the two models can play complementary roles.

Bullet Points Generation. Besides the slide title, the Slide4N back-end generates one bullet point for each selected cell by using the codeT5 [69] model, which is the state-of-the-art code documentation generation model. We directly use the model for bullet point generation without fine-tuning since its performance is already good enough(G3).

However, in some cases, the generated bullet points may be too detailed or too high-level based on the number of selected code cells. Slide4N provides two user interactions with *Control Panel* (Figure 2C) that allow users to customize the complexity of the generated bullet points (G5). The first is an "Auto-merge cells" switch, and when turned on, the selected code cells are grouped based on their relevance scores (see Section 4.3 for details) and concatenated before being sent to the model. The second is a "Level of details" slider (Figure 2C) that controls the detail level of the generated bullet points and offers three different levels. According to the user's choice, the corresponding hyper-parameters of the model are adjusted to generate different bullet points. The above two interactions are also useful for different presentation scenarios. For example, if the main audience includes non-technical stakeholders who need a more concise description, the "Level of details" can be set to the lowest and the "Auto-merge cells" can be turned on; and when most of the audiences are experienced engineers with a deep understanding of related technologies, the "Level of details" can be set to the highest and the "Auto-merge cells" can be off.

4.3 Arrange: Organizing and Laying out Contents

In addition to the generated slide title and bullet points, the system captures the outputs (e.g., plots and tables, if any) of the selected code cells and renders these contents on a slide (G4). The slide title is obvious, however, other contents need a meaningful layout. To

[1] https://docs.python.org/3/library/difflib.html

Table 1: The association between distinct topics (tags) and slide titles that Slide4N uses to generate title candidates which are further selected using SequenceMatcher.

Topic (tags)	Candidate slide titles
Introduction	Purpose, Workflow
Data	Data Loading, Data Preparation, Exploratory Data Analysis, Data Preprocessing, Feature Engineering
Model	Model Building, Train-Test Splitting, Model Training, Model Parameter Tuning, Model Validation
Model Performance	Performance, Model Interpretation
Conclusion	Limitations & Future Work, Suggestions

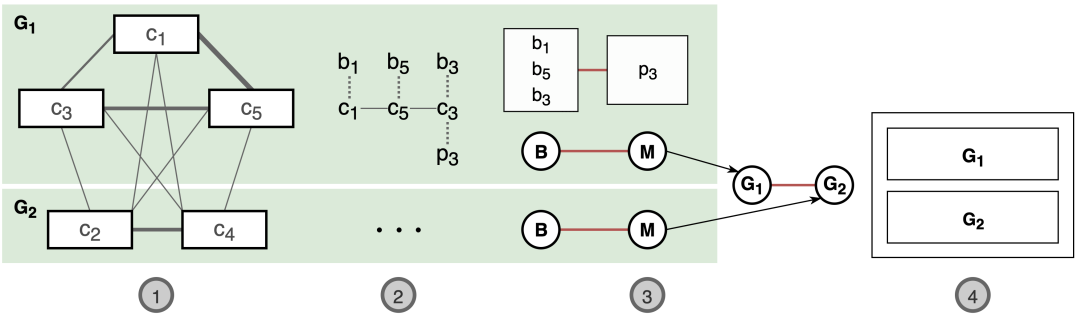

Figure 3: The four-step process to arrange contents on a slide (e.g., bullet points, plots, and tables): (1) separate user-selected cells into groups based on the relevance scores; (2) arrange bullet points and cell outputs based on the logical order of cells; (3) align text and plots by linking them (red lines) and construct the backbones within and between groups; (4) map the two-layer backbones to the layout design patterns (see Figure 4 for details).

do so, the Slide4N back-end goes through the following four-step process, inspired by Zhao et al.'s [78] data comic layout generation method based on "narrative backbones" (i.e., a tree structure logically connecting the layout elements). Essentially, we construct a two-layer narrative backbone for arranging various slide contents in our case (Figure 3).

First, based on the calculated relevance scores for selected cells (see Section 4.1), Slide4N composes a complete weighted graph (Figure 3-1), where each node in the graph represents a cell and each edge represents the relationship between the cells. The relevance score between two cells is defined as the edge weight. After constructing the graph of selected cells, we apply hierarchical agglomerative clustering (HAC) [19] to separate the cells into groups (subgraphs). For example, in Figure 3-1, there are two groups, where G_1 includes cells c_1, c_3, c_5, and G_2 includes cells c_2, c_4.

Then, for each subgraph, we compute a maximum spanning tree to find the subsets of edges that connect all the cells with the biggest total relevance score. This tree represents the structure of the most relevant connections for the selected cells. Based on the execution order and the relevance score of the code cells, we conduct an internal walk-through from the root node of the tree to a leaf node; this order is likely to follow the logical order in the narration. The execution order is used to select the root node and determine the order of multiple cells with the same relevance score. For example, in Figure 3-2, c_1 is the root node and the tree is from left to right. We use this walk-through to arrange bullet points generated by different cells, and cell outputs (e.g., plots and tables, if any) in a linear form, respectively. As shown in the upper part of Figure 3-3, the bullet points (b_1, b_3, b_5) in the group G_1 are ordered (from top to bottom) and placed together (denoted by B), and the output of c_3, a picture p_3 (denoted by M) is also included.

Next, we form an intra-group backbone by linking M and B elements from the same cell (e.g., the red lines in Figure 3-3). This is because the generated bullet points and cell outputs are naturally bonded to the cell (indicated by the dashed lines in Figure 3-2). Similarly, we build a weighted complete graph using the root cell from each group to linearly order different groups, thus forming an intergroup backbone as illustrated on the right of Figure 3-3. Based on the two-layer (intra-group and inter-group) backbones, we generate the final layout (Figure 3-4). For a slide that has four or fewer elements (i.e., nodes in the backbone), there are seven different structures to reflect the logical association between them (Figure 4). This covers most of the cases after grouping the cells, however, there are occasions when a slide contains more than four elements. We then place the elements linearly from left to right, for example, bullet points on the left and plots/tables on the right.

Last, the system converts the narrative backbones to the layout design patterns, which is inspired by Bach et al.'s data comic layouts [2]. We use a rule-based approach to map the two-layer narrative backbone to the layout patterns in the presentation space, as detailed in Figure 4. One narrative backbone might be reasonably matched with multiple layouts. Currently, however, there are no agreed rules or principles for automated matching. Thus, based on Wang et al.'s survey for layouts in infographics [68], we adopt frequently-used layouts such as Tiled (40.4%) and Parallel (28.6%). We also consider the space efficiency of the layouts, such as balancing the size of points and plots/tables and increasing the data-ink ratio. For example, in Figure 7b, the left is the user-selected notebook cells, and the top right is the generated slide with contents properly organized. The selected cells first sort the prediction models, and then visualize their RMSE scores. Accordingly, the generated bullet points and associated cell outputs are separated into two

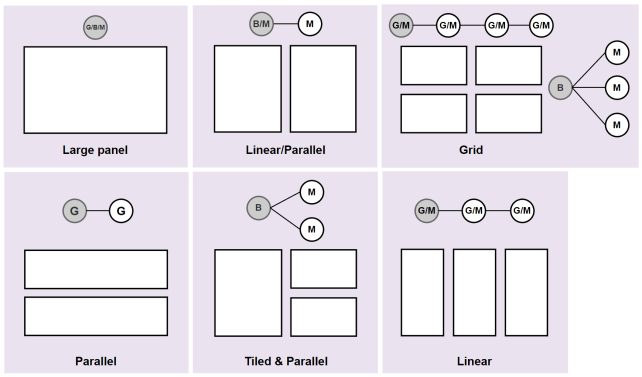

Figure 4: The mapping between slide narrative backbones (in a tree structure) to Bach et al.'s layout patterns [2]. G denotes a cell group, B denotes all the bulleted points in a group, M denotes a cell output such as a plot or table, and the gray circle denotes the root node of the tree.

groups. "Parallel" and "Linear/Parallel" are used for the inter- and intra-group backbones, respectively. White gaps are also used to distinguish different groups.

4.4 Refine: Customizing Generated Slides

From the generated slide, which can be a "first cut," the user can make further refinements directly on the *Slides Panel* (G5). It renders the slide contents and layout configuration that are obtained from the back-end.

Overall, each slide in the *Slides Panel* (Figure 2E) is divided into title area (top) and content area (below), which uses a grid-based approach to place bullet points, plots, tables, etc., all of which can be individually dragged and dropped to adjust the layout. The design of *Slides Panel* draws on commercial presentation tools like Microsoft PowerPoint and provides a set of accessible interactions for users to refine and customize the generated slides. For example, the user can add new bullet points, select generated bullet points, and modify them using markdown-like grammar. The user can also insert, resize, and remove plots or tables, as well as order, delete, and restore slides, etc. In cases where the user wants to create a completely customized slide manually, similar to traditional presentation tools, Slide4N supports this flexibility by offering a set of slide templates with five different layouts (i.e., title slide, title with one-column content, title with two-column content, title only, and blank). The user can just click the "down arrow" on the *Control Panel* (Figure 2C) to access them.

To support easy customization, Slide4N also offers interactive linking between the slides and their associated cells (G1, G5), which is shown in Figure 1d (blue bidirectional arrows). After selecting a slide in *Navigation View* (Figure 2D) or *Slides Panel* (Figure 2E), *Notebook Overview* (Figure 2B) highlights the associated cells (pink dots) and *Control Panel* (Figure 2C) is also updated to show the current configuration for the selected slide. This interactive linking not only allows users to easily trace the source codes to adjust potentially inaccurate code summaries, but also provides them with the flexibility to update the associated slide after adjusting the codes. In addition, an outline of all the slides is displayed on the *Navigation View* (Figure 2D) using the slide topics and titles. This can be used for navigating the slide deck, similar to most slide-making tools. There is also a navigation switch to control the display of navigation information on each slide, including a navigation bar and a page number. For example, on the right of Figure 7c, the navigation information is created by using the topics of the generated slides and the number of slides in each topic, and is placed at the bottom of the slides.

5 USER STUDY

To evaluate our approach, we conducted a user study in which participants were asked to use Slide4N to create slides from a notebook. Our goals for this study are: (S1) assess the usability of Slide4N in supporting presentation slides creation from computational notebooks, (S2) understand users' behaviors when creating presentation slides from computational notebooks with Slide4N, and (S3) collect users' attitudes and feedback on Slide4N's support for creating presentation slides from computing notebooks. We did not conduct a comparative study since there are no appropriate, widely-available baselines. Currently, commercial presentation software such as Microsoft PowerPoint and Google Slides is commonly used by data scientists; however, these tools require much tedious and laborious work as mentioned earlier, which results in an unfair comparison in terms of our study tasks and goals. The closest system to ours is NB2Slides [79] which still has many differences as discussed in Section 2.1. It uses a template-based approach to generate slides fully automatically with little human intervention and has a set of very strict requirements on the input notebooks; whereas, in our study, we focus on investigating the human-AI collaborative aspects in slide creation without much constraints on inputs. Further, it is impossible to replicate their approach without the data and trained model. Thus, we decided to qualitatively assess users' experience of Slide4N in comparison to their current practices.

5.1 Study Design and Data

Our study contained two parts, where in Part 1, participants used a notebook provided by us to create slides with Slide4N, and in Part 2, they brought their own notebooks. In a more controlled setting, Part 1 enabled us to evaluate Slide4N with exhaustive types of cells and topics in the data science workflow, because a user-created notebook may miss certain aspects. In a more realistic setting, Part 2 allowed us to assess the flexibility and generalizability of Slide4N. This way, we could investigate the capabilities of Slide4N from multiple perspectives more thoroughly. The entire study was conducted remotely via video conferencing software. Thus, we deployed Slide4N's back-end and front-end on Google Cloud to ensure the participants could access it via their computers.

Specifically for Part 1, our experimental notebook consisted of 32 code cells and included a complete data science workflow. The notebook combined two notebooks from the Kaggle competitions House Prices Prediction[2] with the goal of building models to predict the price of each house. We chose this competition because it is one of the most popular competitions on Kaggle and many data science courses use this competition as a tutorial. We combined two

[2]https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques

notebooks to form a data science workflow with five stages (i.e., Introduction, Data, Model, Model Performance, and Conclusion). We also made some minor edits to make the notebook more concise and readable for participants, including deleting some redundant and duplicate cells. Note that these changes were not intended to improve the quality of the generated slide, but to guarantee the user study would not be too long.

5.2 Participants

For Part 1 of our study, we recruited 12 participants (10 males, 2 females; aged M=23.5, SD = 3.47) through social networks and word of mouth. The participants were all professional data scientists or software engineers with previous experience conducting and presenting data science projects, as well as familiar with Python and Jupyter Notebook. On a 1-7 Likert Scale, the medians of participants' self-reported familiarity with Jupyter Notebook, Python, and Machine Learning algorithms were 6, 5.5, and 5.5, respectively (1 = "no experience" and 7 = "expert"). The information collected from the participants also showed that on average, our participants make 5 data science-related presentations per week, as part of their work routine, indicating that our participants were familiar with creating presentations to present their data science projects.

In Part 2 of the study, through social networks, six new participants were recruited (3 males, 3 females; aged M=24.1, SD = 2.9). Among them, four had master's degrees and the rest had doctoral degrees in fields related to data science, such as computer science and statistics. There were five participants who identified themselves as experts in machine learning and one participant as a specialist in data visualization, indicating that all of the participants had sufficient professional data science and coding background. With the same Likert Scale as above, the medians of participants' self-reported familiarity with Jupyter Notebook, Python, and Machine Learning algorithms were 5, 6, and 5, respectively. Also, our participants had sufficient experience in generating data science-related presentations by creating 4.8 data science-related presentations per week, on average.

5.3 Procedure and Task

The procedure and task for Parts 1 and 2 of the study were the same. After participants used their own computers to join our online study session and gave consent, we shared the URL for accessing Slide4N. During the study, participants first completed a brief training session to get familiar with Slide4N by watching a short video and using a sample notebook to explore the system and generate some slides. Then, based on a given experimental notebook (Part 1) or one of their own notebooks (Part 2), participants were asked to create a slide deck to present what the notebook does by imagining their audience as technical and non-technical members of their teams. We instructed participants that the slide deck should consist of at least five slides based on their preferences and cover as many stages of the workflow in the notebook as possible. They were given about 30 minutes to create the slide deck but we did not put a hard cut-off. Participants were asked to use the AI option of Slide4N to generate the slides for sections "Data", "Model", and "Model Performance" if corresponding content was included in the notebook, and use the manual template option for sections "Introduction" and "Conclusion", which is usually not included in the notebook. In this way, they could have both automatic and manual slide creation experiences with Slide4N. They were free to refine the slides, such as editing the title and bullet points, and adjusting the layouts.

After, participants completed three post-study questionnaires about their impression of Slide4N, followed by a semi-structured interview. The three questionnaires requested them to rate their created slides, the back-end of Slide4N, and its user interface, respectively (see Figure 5 for questions). The interview included topics on their experiences with Slide4N, practices with existing tools, qualitative comparison between Slide4N and tools they used before, as well as general opinions towards human-AI collaboration. Detailed questions in our study can be viewed in the supplementary materials. In total, the study session lasted about an hour and participants were remunerated by $15.

6 RESULTS

Overall, 17 out of 18 participants successfully completed the task within 30 minutes, and one participant took 35 minutes due to intentionally trying out all of the system's features. In the following, we report our user study results including participants' questionnaire responses, quality assessment of the generated slides, and qualitative feedback to the system.

6.1 Questionnaire Responses

We designed three post-study questionnaires to evaluate the created final slides, Slide4N's back-end, and its user interface, revealing insights into our study goals S1-3. Figure 5 shows the distributions of ratings as well as the means and IQRs for all the participants in study Parts 1 and 2. The results from the two parts are very similar; please see our supplementary materials for additional figures.

Q1-4 regard participants' impressions with the final slides generated by Slide4N. Overall, the participants were satisfied with the final slides (Q1), with a median rating of 5 (IQR = 1). They all thought the contents (titles and bullet points) on slides were easy to follow (Q2; MD=6, IQR = 0) and properly arranged (Q3; MD=6, IQR = 0.5). The rating for the aesthetics of the final slides (Q4) was relatively low (MD=5, IQR = 1), but still positive. Two participants (P3 and P5) gave negative ratings. With our follow-up interviews, we found out this was due to personal preference in creating slides, with some people preferring a concise style and others preferring to add more stuff (e.g., bold, underline, and colors).

Q5-12 show participants' impression of the usefulness of the automatic slide generation (i.e., the AI) of Slide4N. Overall, participants were satisfied with the experience of making slides using the AI-Assisted tool (Q5; MD=6, IQR = 1.5), except that P5 and P18 rated 3. By observing P5's other ratings, we found this was actually because of the layout adjustment and bullet points editing, and he also suggested Slide4N to support some keyboard shortcuts (e.g., copy and paste, redo and undo) and more fine-grained layout adjustments. P18 wanted to have more bullet points for cells with many lines of code. More specifically, the participants thought they could find relevant cells (Q6; MD=6, IQR = 1.5), the model is easy to configure (Q7; MD=6, IQR = 0.5), the generated bullet points (Q10; MD=4.5, IQR = 1) and titles (Q11; MD=6, IQR = 2) roughly matched what

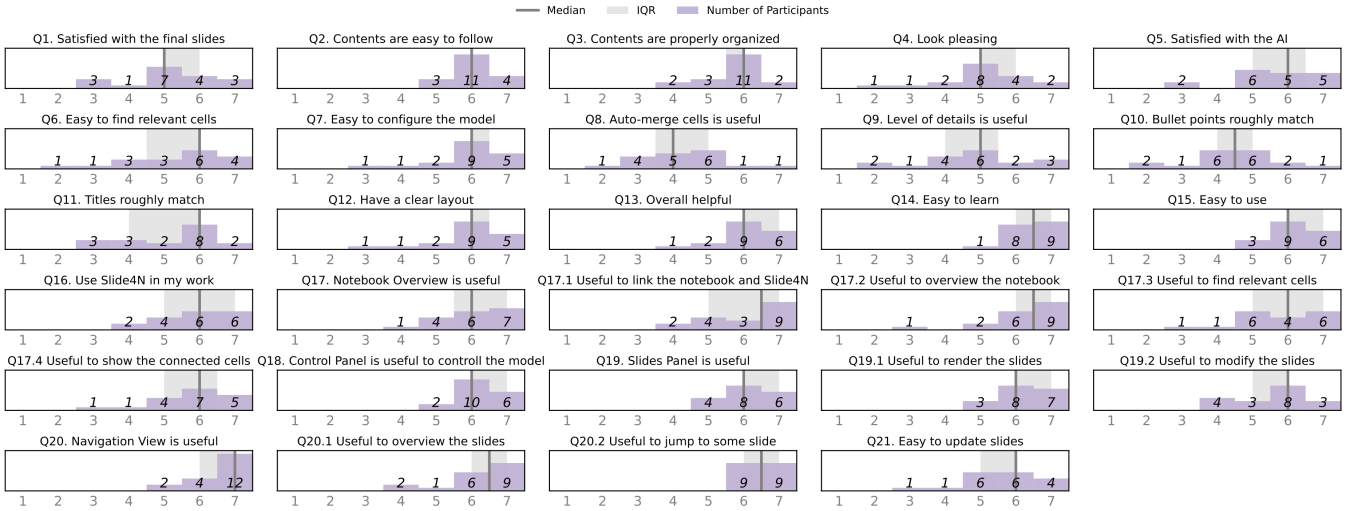

Figure 5: Participants' ratings on the post-study questionnaires including the created slides (Q1-4), the back-end (Q5-12), and the user interface (Q13-21), on a 7-point Likert Scale (1 = "strongly disagree" and 7 = "strongly agree").

they wanted to put on the slides, and the generated slides had clear layouts (Q12; MD=6, IQR = 0.5). However, P3 and P11 found it hard to find relevant cells (Q6), citing that they didn't fully understand the visual encodings in *Notebook Overview* (Figure 2B). In addition, Slide4N's features, "Auto-merge" (Q8) and "Level of details" (Q9), were rated relatively lower; however, the medians remained positive. By analyzing the recorded videos, we found most participants used the default settings, indicating they may not be aware of the impact of these two configurations.

Q13-21 reflect participants' impression of the usefulness of the Slide4N user interface. Overall, Slide4N was perceived as very helpful (Q13), easy to learn (Q14), easy to use (Q15), all having a median rating of 6 or more. Although several participants rated low on some aspects of Slide4N, they all wanted to use Slide4N in their future work (Q16) and rated it fairly high (six rated it as perfect). P5, while giving more negative ratings on other questions, still rated 6 for Q16; he further commented, *"Slide4N effectively links data analysis work with the presentation slides, it can save me a lot of time in creating slides, although Slide4N should be improved to support an easier way to customize slides."* Participants also felt that the four panels in Slide4N (Figure 2) were very helpful (Q17-20), all with a median of 6 or more. But P5 thought that *Notebook Overview* (Figure 2B) was cluttered when there were many arcs, and thus rated 3 for Q17.2. P16, who rated 3 for Q17.3, thought that *Notebook Overview* was not easy to select cells because the notebook he used was very long and the rectangles were sometimes too small (Figure 7a). Almost everyone thought that *Control Panel* (Figure 2C) was useful for controlling the slide generation model (Q18). *Slides Panel* (Figure 2E) was rated relatively low, probably because the user interactions provided by Slide4N for editing the bullet points and adjusting the layout could not be on par with professional software such as Microsoft PowerPoint (Q19.2). All participants thought that *Navigation View* (Figure 2D) was useful for slide overview (Q20.1) and navigation (Q20.2). Updating slides was also perceived easily (Q21), except that P3 rated 3. Combined

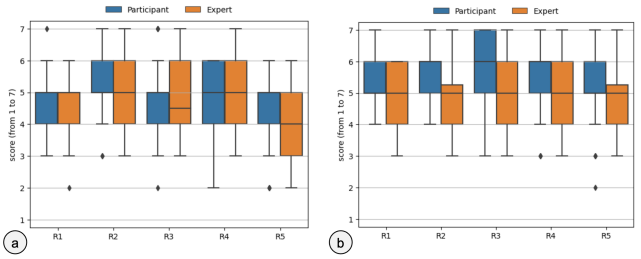

Figure 6: Audience ratings on five aspects (R1-5) of the slides created by Slide4N: (a) boxplots of ratings for the 12 decks of slides created using the experimental notebook in study Part 1, and (b) boxplots of ratings for the 6 decks of slides created using participants' own notebooks. The five aspects include overall satisfaction (R1), clarity of structure (R2), ease of understanding the content (R3), proper organization of the content (R4), and aesthetics (R5), on a 7-point Likert Scale (1 = "strongly disagree" and 7 = "strongly agree").

with the interview, we found that this was caused by Slide4N hiding the image he pasted onto the slide after the update.

6.2 Slides Quality Assessment

While Q1-4 in our questionnaire indicate participants' self-impression of the created slides, these do not reflect how the slides are received by the audience. To assess the quality of the slides created with Slide4N more comprehensively, we invited the six participants in Part 2 to rate all the slides in the whole study (except their own slides); thus each of them rated 17 decks of slides. We did not invite the participants in Part 1 for the slides rating because they all created slides from the same experimental notebook, which might generate bias. Additionally, we invited two experts, who are university professors with years of experience in preparing, delivering, and judging presentations, to rate all the 18 decks of slides created in Part 1 and Part 2.

The results of their ratings are shown in Figure 6, on five aspects including overall satisfaction (R1), clarity of structure (R2), ease of understanding the content (R3), proper organization of the content (R4), and aesthetics (R5). For the slides created in Part 1 (Figure 6a), most of the median ratings were 5 or above for both the participants and experts. The experts rated the ease of understanding slightly lower (MD=4.5). Another exception was the aesthetics which received the lowest rating from the experts (MD=4); however, in this paper, we did not focus on the aesthetic capabilities for Slide4N. This could be easily enhanced by employing themes and templates like those in commercial presentation tools. For the slides created in Part 2 (Figure 6b), the overall median ratings were higher, where all aspects were 5 or above. This might be due to the fact that participants were more familiar with their notebooks and the contents were richer, which influenced the quality of the created slides. In summary, the ratings from the six participants and two experts indicated the overall effectiveness of Slide4N and its stable performance in aiding slide creation from notebooks.

6.3 Qualitative Feedback

Our interview mainly focused on collecting participants' feedback on all aspects of Slide4N, practices with current tools, and human-AI collaboration, to assess S1-3 qualitatively. In general, participants appreciated our human-AI collaborative approach to creating presentation slides compared to the fully-manual or fully-automatic methods. In the following, we report our results on the challenges of current presentation software and their feedback on Slide4N.

6.3.1 Challenges of Current Presentation Software . All participants said that Microsoft PowerPoint was the tool they mostly used to report on their data analysis work. The process of creating slides with PowerPoint can be roughly divided into three stages: (a) write an outline to sort out the content to be reported or find a template that has been used before, (b) distill key content from the code and transfer it to the slides, and (c) style the slides to make them more presentable. They agreed that using PowerPoint to report data analysis work had the following problems. First, it was difficult to find relevant cells from a long and messy notebook, as P1 said: *"As the notebooks got longer, I couldn't remember the interconnections between the cells and had trouble finding them."* Second, distilling presentable content from the notebook was time-consuming. *"After the analysis, I usually spend a lot of time thinking about the content of the presentation."* -P9. Third, organizing, laying out and styling the content on the slides was also tedious. Fourth, when the code was modified, the corresponding slides needed to be located and updated manually, a process that was error-prone and laborious, as P14 said: *"it's harder to change slides than to make them."* Besides, most participants were willing to clean up the notebook before making the slides, such as deleting excess codes and adjusting the cell order, but almost no one is willing to add documentation. *"Documentation helps me understand the notebook, but it's tedious and time-consuming to do so."* -P1.

The result confirms our hypotheses on the deficiency of current practice in presenting data science work conducted with computational notebooks. It also validates our design goals and the development of Slide4N. Further, it extends our vision of closing the analysis presentation divide.

6.3.2 Feedback on Slide4N . The feedback from all participants on Slide4N was highly positive. In general, participants thought Slide4N's user interface is clear and good-looking, and they were more willing to use our tool than commercial tools such as Microsoft PowerPoint, especially when their time is limited. In the following, we group their feedback based on our design goals in Section 3.1.

G1: Link the notebook and presentation slides. When asked about the advantages of Slide4N, almost all participants mentioned it was very convenient to link notebook cells and slides. Most of them felt this helped structure and organize their presentation slides. On one hand, this allowed them to quickly adjust and update the corresponding slides after changing the code. *"I can quickly go back to the source code I referenced when I created the slides [for adjustments]."* -P1. *"Slide4N relieves me from the burden of remembering the association between slides and notebook cells, which is needed for updating."* -P4. *"This saves me the time to copy and paste the cell output image."* -P12. Also, participants suggested some improvements to Slide4N. *"I would like to link bullet points to cells, which would make customization easier."* -P8. *"Currently it supports synchronization of slide content after code changes, but I would like to keep my layout while updating."* -P6. On the other hand, they believed it made communication within the team easier, as P1 and P16 mentioned *"Slides are easier to accept and understand, while Slide4N allows me to quickly link to my codes when I discuss technical details with developers."* In addition, participants thought the link helped them organize their notebooks, as P17 said: *"I don't have to organize my notebooks linearly to facilitate the creation of the slides, while I believe Notebook Overview can also assist me with this."* Furthermore, three participants mentioned that building Slide4N into JupyterLab made the link even tighter. *"I can do data analysis and slides in one window."* -P16. *"The process [to create slides] is simpler and smoother, and it saves me the time switching between PowerPoint and the notebook."* -P18.

G2: Help locate relevant cells in the notebook for creating slides. For locating relevant cells, P1 appreciated *Notebook Overview* for doing so, *"I do like Notebook Overview, and I can quickly find relevant cells in my messy notebook."* However, P13 and P15 suggested that when computing cell relevance, it is insufficient to only consider variables with the same name, but also the data they are associated with. For the design of *Notebook Overview*, P17 appreciated it very much: *"I can intuitively see the selected cells, their positions, and some details of the cells, which helps in the overall understanding of the notebook."* Meanwhile, participants offered some suggestions. When selecting cells, P8 suggested supporting a one-click selection of relevant cells, which was envisioned at the beginning of our design. However, it might lead to unclear associated cells for slides, less predictable generated content, and users might spend much time adjusting the generated content. P2 and P8 suggested reflecting the hierarchy of the notebook more obviously (e.g., using dotted lines to separate different parts), *"It will allow me to quickly understand the structure of the notebook, which facilitates the organization of the report."* -P8. P7 and P16 (who have light color vision deficiency) recommended using larger dots or brighter colors to mark cell states, *"because it would be more obvious when there are too many cells."* -P7.

G3: Assist with distilling key information from code and markdowns. All participants thought having AI to assist them to

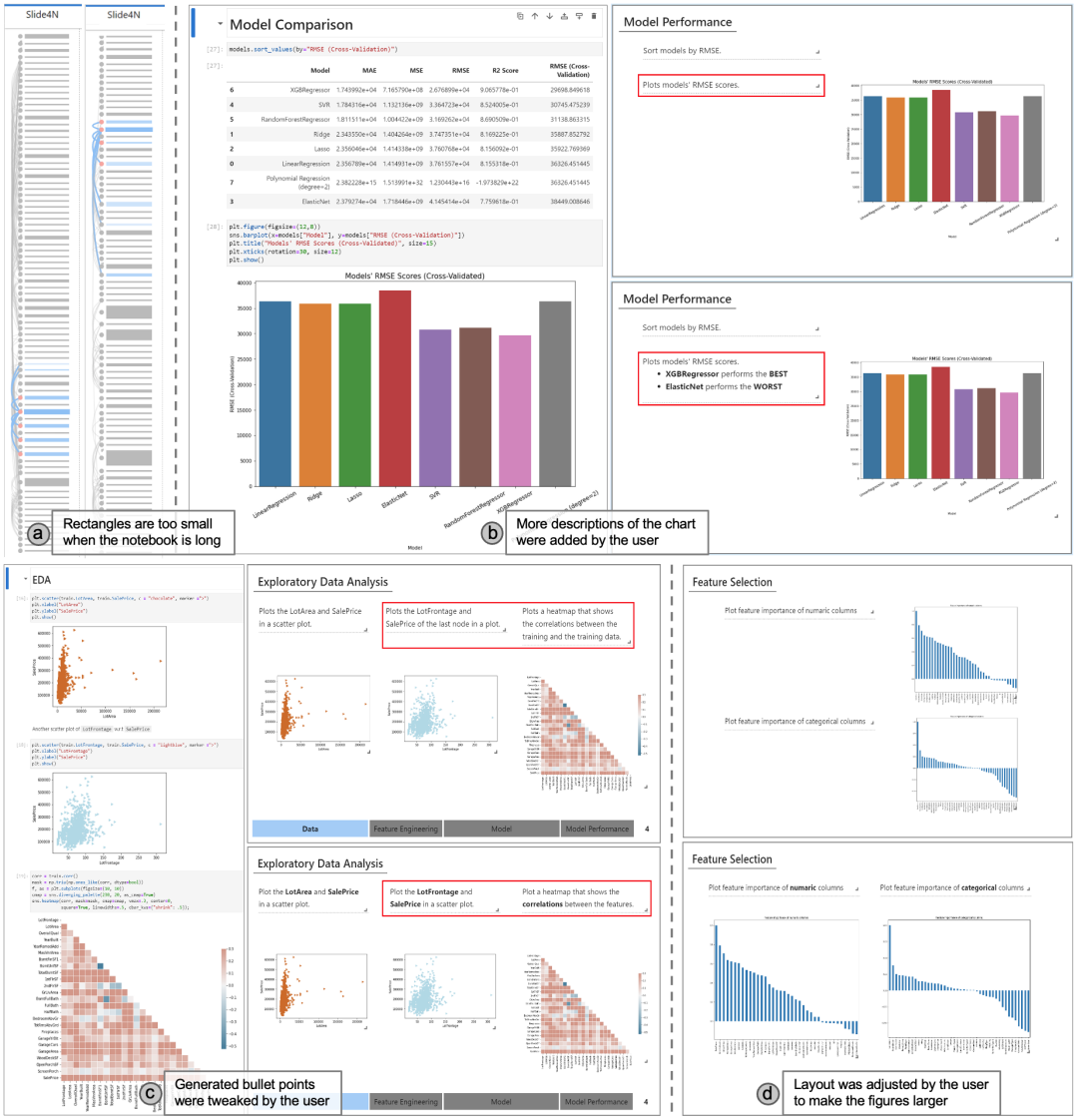

Figure 7: (a) When the notebook is too long, the rectangles in *Notebook Overview* can be small and difficult to select. (b)(c) Slide4N sometimes generates less satisfying bullet points and the participants needed to improve the text (indicated by the red boxes) on their slides (left: the original notebook; top-right: AI-generated bullet points; bottom-right: edited bullet points). (d) Slide4N sometimes produces undesired layouts and the participant wanted to make the figures larger.

distill titles and bullet points from the code is very convenient. As shown in Figure 7c, P9 and P17 mentioned *"AI gives me a good 'first-cut.' I just need to tweak the generated content a bit to my preference and it can be used for the report."* We also found that participants had different requirements for the generated bullet points. Most of them felt that the distilled points roughly matched what they want to put on slides. Two expected the distilled points to answer not only what was done, but also how it was done. Regarding the topic, most of them thought the options we provided were sufficient for reporting. While P1 suggested that the "Data" can be subdivided, *"because I spend most of the time doing data processing, and there is a lot to tell about data."* The interfaces of Slide4N allow users to provide high-level guidance for slides generation: "Auto-merge cells" and "Level of details". We found that most participants explored these options first but later adopted the default settings, which generated more detailed content for the slides. But P15 preferred the settings to generate more concise content. In addition, four participants thought that some options for configuring the model could be added to guide the model. P7, P13, and P17 jointly suggested allowing users to set their preference for generating content from code or documentation. P18 thought that more bullet points should be generated for cells with many lines of code to provide more details, so he suggested providing a configuration that automatically breaks down long cells. Regarding the time delay of the generation, almost all participants could accept it.

G4: Arrange slide contents in a logical and visually pleasing manner. All participants felt that the organization of the content on the slides matched the logical correlation between them. All praised the initial layout provided by Slide4N. *"I like the generated layouts, I don't need to spend much time thinking about how to organize the content on the slides and keep the style consistent between slides."* -P12. *"The generated layouts save me a lot of time, and a simple tweak to them can satisfy my layout needs."* -P11. We also found that participants' layout preferences varied. Most preferred a consistent layout style, while P14 wanted more diversity of the generated slide layouts, *"because it draws the audiences' attention to the presentation."* In addition, P14, P19, and P20 specifically mentioned that the alignment of elements on the slides was important. Three participants also made suggestions for the layout. P5 and P13 preferred a more fine-grained grid *"because I can be more flexible in adjusting the layout."* P7 wanted bigger images (Figure 7d) because *"when I place images on slides, I tend to care more about the images than the texts."* He also suggested enabling users to specify the layout of the generated slides. In addition, three participants appreciated the navigation provided by Slide4N. *"Navigation allows audiences to quickly know the current progress of the report, while the page number is convenient for Q&A."* -P10. *"This is what I want to put on the slides, but PowerPoint does not support one-click generation, and adding it manually is laborious and time-consuming."* -P7, which was also agreed by P20.

G5: Support necessary human intervention for slide generation. Slide4N provides necessary human intervention for users to customize the generated slides, which can be roughly divided into three stages: select the inputs (cells) to the slides generation model, configure the parameters of the model, and adjust the content generated by the model. All participants thought such human-AI collaboration was acceptable and met their needs for AI. *"Slide4N meets all my requirements [for AI]. It is very comfortable to use."* -P9. *"The human intervention provided by Slide4N is necessary and it helps me with the tedious and basic things, so I can spare some time to focus on how to tell the story effectively."* -P10. Besides, participants preferred a human-AI collaborative approach compared to a fully-manual or a fully-automatic method to create slides. As we said in Section 6.3.1, all participants thought that the fully-manual approach was tedious and time-consuming. For the fully-automatic approach, all held a negative attitude. As P9 said: *"I don't trust the black-box model, and I want to intervene in the generation process."* P16 and P18, who agreed with P9, said: *"The generated contents can't be predictable, and it's hard to adjust when the slides are not what I want, like the structure and [slide] contents."* P1 also said: *"I don't believe today's AI can automatically generate high-quality slides. Realistically, I don't want to have such an AI either, because I don't want to lose my job."* Moreover, we believe today's AI can't go beyond the notebook, so some slides (e.g., background and limitations) can't be provided by AI, and human involvement is needed. When asked what they thought about creating slides one by one, all gave positive feedback, as P6 said: *"I can have more control of the slide creation process, the content generated by the model is more predictable."* P11 said: *"It's easy for me to incorporate my own ideas into the slides, which is very important for a presentation."*

7 DISCUSSION

7.1 Design Implications

From our user study, we consolidate a few design implications that could shed light on the future design of similar tools.

7.1.1 More Adaptive Slides Generation. Slide4N allows users to manage the level of detail and complexity of the slides. However, depending on the scenario in which the slides will be used, data scientists may have different requirements for their slides. The designers of these tools should therefore take a user-specific approach in terms of providing different types of content and different layouts to generate the slides. For example, participants suggested that it is an advantage for Slide4N to generate complex slides that contain in-depth descriptions with more complex visualizations from the notebooks for those who need to present in-depth reports or research proposals, as well as creating slides with simple figures and less complex contents for those unfamiliar with the notebook, especially non-technical individuals or students. We recommend that future AI-assisted slide creation tools should generate presentations adapted for various audiences by collecting slides in different usage scenarios and fine-tuning the content generation model. Participants also suggested the generated bullet points can cover how the code got the result instead of just describing what the code does. For example, inline code comments can be used to recommend more alternatives for the generation.

7.1.2 More Obvious Data Provenance. Creating slides often requires some materials (e.g., text, images, code, etc.) which are *data provenance* for slides; in this paper, it mainly refers to the notebook cells. Participants found it very helpful to bind the slides to the cells when asked about data provenance. Such bindings can quickly locate the original cells, which 1) simplifies the modification and updating of slides, and 2) provides more detailed information for presentations to accommodate different scenarios, such as using original code to facilitate technical communication, and using interactive visualizations for more vivid upward reports. Additionally, bindings can assist the user in locating incorrect annotations in the notebook, which can be adjusted to make it easier to comprehend. In this regard, we can argue that slide creation tools could integrate the binding feature between slides and the original materials used for slides (e.g., code, text, and images) to enable the users with a more flexible and interactive slide creation experience.

7.1.3 Better Human-AI Collaboration. When creating slides with Slide4N, the AI can assist data analysts in three aspects: 1) selecting cells as input to the model, 2) configuring the model at a high level, and 3) adjusting and customizing the generated content. By repeating the above process, users can quickly create the slides they want one by one. Such Human-AI collaboration not only simplifies the process of creating slides and enables users to participate more actively but also ensures that AI-generated content matches users' expectations and is customized according to their needs. Combined with the results of the user study, we believe that there is room for improvement in all three aspects.

For input, Slide4N currently supports selection at the cell level, future extensions can be made to support selecting at the sub-cell level, which can leverage the location of inline comments and the

functional scope of code. With *Notebook Overview* (Figure 2B), we can apply brush interaction to facilitate this feature. It would be beneficial to allow the user to control the threshold of relevance, such as the relevance score (see Section 4.1) and the number of shared variables, which not only allows users to filter for highly relevant cells but also reduces the clutter caused by many arcs when there are many relevant cells.

Regarding model configuration, other effective high-level controls can be provided to the user, suggested by our participants from the user study. For instance, users may adjust the model's preference to accommodate different presentation scenarios by taking into account analysts' programming habits (e.g., rich or sparse annotations) and the intent of the generated content (e.g., inform, convince, tell a story). However, when users have multiple configurations, they may be overwhelmed and do not know what to do. To avoid unnecessary time overhead, it is essential to clarify to the users the possible impact of the configuration (e.g., provide metrics to measure the impact or in-situ instructions) or weighted merging of multiple related configurations into an integrated one.

As for customization, the AI should learn user behaviors for adjusting slides, including wording style, bolded text (often for content that needs to be emphasized), preferred layout methods, etc., and provide suggestions on how to refine future slides similarly. Furthermore, the AI should be able to extract templates from user-created slides and add them to the user's personal template library, allowing for customized layout manipulation. Additionally, given that visualizations are common parts of such presentations, and that analysts may find it bothersome to do so (e.g., which toolkit to use, what APIs are available, and how to use them), we believe it would be useful to recommend and add relevant visualizations to the slides, based on the code. The recommendation can be supported by several existing tools built in the Jupyter Notebook (the classic notebook interface compared to JupyterLab) [12, 74].

7.1.4 Less Bias and Risks. The story we tell, in this case, data science work, is influenced by our personal background and life experience. Therefore, we may not realize that we are limiting ourselves to certain aspects and ignoring others that may be meaningful (we call this personal *bias*), and the impact is often detrimental. The results of our user study show that Slide4N can reduce users' personal bias when making slides. Our analysis determined that two participants (P16 and P17) were inclined to choose cells containing plots to make slides, thus ignoring other cells (e.g., long code cells). This implies that users may be reluctant to digest long code cells, while our AI does not differentiate between cells of different lengths in digesting code. To resolve this issue, *Notebook Overview* may be enhanced to highlight used cells, which will facilitate the user's discovery of unused cells. In addition, people tend to tell stories from certain fixed perspectives, which may lack appeal to audiences, especially in similar scenarios, but AI can provide some new perspectives that will guide users to create slides with more comprehensive and diverse content.

Although the AI component of our system is designed to assist data analysts in creating slides, there are some potential *risks* with this process. First, our analysis of participants' feedback indicates that privacy is one of the main concerns of our participants. Three participants suggested that Slide4N should run locally when sensitive data is involved so as to prevent the leakage of data. While our system relies on online tools for implementation, further improvements may be considered to enhance user privacy, by encrypting their data as well as executing Slide4N on the user's local computer. Second, users may over-rely on the automation solution provided by Slide4N, leading to their inexperience with slides and thus affecting the quality of presentations. Considering our system is designed to create slides with human-AI collaboration and requires user interaction, we believe that the negative impact is relatively insignificant. In addition, the user study revealed that participants integrated their own elements (e.g., insights and styles) into AI-generated content, which led to a greater familiarity with the slides. But future study needs to further explore the potential over-reliance on AI.

7.2 Limitations and Future Work

There still exist several limitations in our tool and user study. We outline them below and provide directions for future work.

7.2.1 Slides Generation Model. Slide4N only generates bullet points based on each code cell, not including markdowns or code outputs. Markdowns are often considered as a summary of a section, which can provide a supplementary explanation for code. Code outputs can be used to explain the code purpose and method in detail. The system can be improved by conducting an empirical study to investigate the relationship among code cells, code outputs, and markdown cells and figure out how to make good use of such information. Moreover, code outputs in Jupyter Notebooks can be complicated, often including charts and tables, which are leveraged by our method to generate visually appealing slides. However, vision-impaired people would need to rely on alternative text or the help of a screen reader [49]. To make the generated slides more accessible, inspired by the work on chart and table summarization [16, 24, 27, 73], future work may explore automatic methods that generate alternative text for the charts and tables on the slides, which will broaden the audience and use-cases for Slide4N. As shown in Figure 7b, this is an example where the participant enriched the bullet points based on the chart. Also, even though most participants were satisfied with the slide generation, several participants were still not very satisfied with the slide content. Future work can be done to build a specific dataset for the slide creation task and train the model in a large repository.

7.2.2 User Interface. The front-end of Slide4N can be enhanced. When creating slides, we usually need to consider the structure (often refers to the outline), content, layout, and styling. Slide4N can help users distill the key bullet points and titles from notebook cells, render these contents with appropriate layouts and support further adjustments. Thus, we believe Slide4N can effectively address the needs for content and layout. However, Slide4N provides limited support for structure (e.g., selection and creation of topic, which is similar to the paper section). Five participants indicated that they would reuse the structure of previous slides in similar presentations, which is also pointed out by [52]. As a result, Slide4N should enable users to extract structures from existing slides to reuse them again. As for styling, Slide4N currently only supports markdown-based

editing, which we believe is often insufficient for formal situations (as five participants stated in the user study, while they liked the preset styles provided by Slide4N). Future improvements may include support for more styling features, such as colors, bolding, font size, etc. In addition, providing in-situ guidance for the users and supporting export to other formats (e.g., .pptx) will improve Slide4N and address some user needs.

7.2.3 Evaluation. Our user study also has limitations. First, in this paper, we mainly focused on designing and developing the system, and we did not choose a baseline to compare in our study as explained in Section 5. However, a thorough empirical investigation comparing different families of approaches would help gain a broader understanding of the advantages and disadvantages of these different approaches. These approaches include deep coupled human-AI collaboration (e.g., Slide4N), fully-automatic generation (e.g., NB2Slide [79]), and fully-manual operation (e.g., Microsoft PowerPoint). Second, we evaluated Slide4N using both a provided experimental notebook and participants' own notebooks in study Part 1 and Part 2, respectively. This indeed provides insights into the usage of Slide4N in both controlled and natural settings. However, future work is needed to carry out a long-term deployment study with more participants to better assess the usefulness of the system in the wild without any constraints. Third, Our current study design is in a controlled environment with limited notebooks and task procedures, but slides creation tasks in the wild can be diverse and flexible. Most of the participants in our user study were data engineers, but non-technical users such as managers also need similar support in creating slides for their work. Future studies need to be conducted with different groups of participants to examine the effectiveness of our system.

8 CONCLUSION

This paper presents Slide4N, an intelligent and interactive tool built within JupyterLab to support data scientists in creating slides using computational notebooks. By leveraging advanced natural language processing techniques, Slide4N distills titles and bullet points from user-selected notebook cells and arranges them algorithmically into groups, and displays them with appropriate layouts. A tool such as this automates many of the tedious tasks associated with the creation of slides, leverages appropriate user inputs, and produces results in a format that is widely used for presentations. A user study was conducted to evaluate Slide4N. The results indicated that it was both useful and effective in supporting slide creation tasks from notebooks as well as encouraging human-AI collaboration. Our study also contains some design implications that will inform future tool development.

ACKNOWLEDGMENTS

We thank all our participants for their time and valuable input. The work was completed during Fengjie Wang's summer internship at the University of Waterloo. This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) through the Discovery Grant, the Waterloo AI Institute, and the National Natural Science Foundation of China (NSFC) with grant No. 62172289.

REFERENCES

[1] Damian Avila. 2019. RISE. https://github.com/damianavila/RISE
[2] Benjamin Bach, Zezhong Wang, Matteo Farinella, Dave Murray-Rust, and Nathalie Henry Riche. 2018. Design patterns for data comics. In *Proceedings of the 2018 chi conference on human factors in computing systems*. Association for Computing Machinery, New York, NY, USA, 1–12.
[3] Ori Bar El, Tova Milo, and Amit Somech. 2020. Automatically generating data exploration sessions using deep reinforcement learning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 1527–1537.
[4] Jeff Barnes. 2015. Azure machine learning. In *Microsoft Azure Essentials*. Microsoft, Washington, DC, USA.
[5] Matthias Blohm, Glorianna Jagfeld, Ekta Sood, Xiang Yu, and Ngoc Thang Vu. 2018. Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension. *arXiv preprint arXiv:1808.08744* abs/1808.08744 (2018).
[6] Matthew Brehmer and Robert Kosara. 2021. From jam session to recital: Synchronous communication and collaboration around data in organizations. *arXiv preprint arXiv:2107.09042* abs/2107.09042 (2021).
[7] Souti Chattopadhyay, Ishita Prasad, Austin Z Henley, Anita Sarma, and Titus Barik. 2020. What's wrong with computational notebooks? Pain points, needs, and design opportunities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–12.
[8] Colin Clement, Dawn Drain, Jonathan Timcheck, Alexey Svyatkovskiy, and Neel Sundaresan. 2020. PyMT5: multi-mode translation of natural language and Python code with transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 9052–9065.
[9] David Donoho. 2017. 50 years of data science. *Journal of Computational and Graphical Statistics* 26, 4 (2017), 745–766.
[10] Ian Drosos, Titus Barik, Philip J Guo, Robert DeLine, and Sumit Gulwani. 2020. Wrex: A unified programming-by-example interaction for synthesizing readable code for data scientists. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. Association for Computing Machinery, New York, NY, USA, 1–12.
[11] Ahmed Elnaggar, Wei Ding, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Silvia Severini, Florian Matthes, and Burkhard Rost. 2021. CodeTrans: Towards Cracking the Language of Silicon's Code Through Self-Supervised Deep Learning and High Performance Computing. *arXiv preprint arXiv:2104.02443* abs/2104.02443 (2021).
[12] Will Epperson, Doris Jung-Lin Lee, Leijie Wang, Kunal Agarwal, Aditya G Parameswaran, Dominik Moritz, and Adam Perer. 2022. Leveraging Analysis History for Improved In Situ Visualization Recommendation. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, USA, 145–155.
[13] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. CodeBERT: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155* abs/2002.08155 (2020).
[14] Tsu-Jui Fu, William Yang Wang, Daniel McDuff, and Yale Song. 2022. Doc2PPT: Automatic presentation slides generation from scientific documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. Association for the Advancement of Artificial Intelligence, California, CA, USA, 634–642.
[15] Sainyam Galhotra, Udayan Khurana, Oktie Hassanzadeh, Kavitha Srinivas, Horst Samulowitz, and Miao Qi. 2019. Automated Feature Enhancement for Predictive Modeling using External Knowledge. In *2019 International Conference on Data Mining Workshops (ICDMW)*. IEEE, New York, NY, USA, 1094–1097. https://doi.org/10.1109/ICDMW.2019.00161
[16] Heng Gong, Xiaocheng Feng, Bing Qin, and Ting Liu. 2019. Table-to-Text Generation with Effective Hierarchical Encoder on Three Dimensions (Row, Column and Time). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, Stroudsburg, PA, USA, 3141–3150.
[17] Google. 2021. Cloud AutoML Custom Machine Learning Models. https://cloud.google.com/automl
[18] Philip J. Guo and Margo I. Seltzer. 2012. BURRITO: Wrapping Your Lab Notebook in Computational Infrastructure. In *4th Workshop on the Theory and Practice of Provenance, TaPP'12, Boston, MA, USA, June 14-15, 2012*. USENIX Association, Boston, MA, USA.
[19] Jiawei Han, Jian Pei, and Hanghang Tong. 2022. *Data mining: concepts and techniques*. Morgan kaufmann, San Francisco, CA.
[20] Andrew Head, Fred Hohman, Titus Barik, Steven Mark Drucker, and Robert DeLine. 2019. Managing Messes in Computational Notebooks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*. ACM, Glasgow, Scotland, UK, 270.

[21] Andrew Head, Jason Jiang, James Smith, Marti A. Hearst, and Björn Hartmann. 2020. Composing Flexibly-Organized Step-by-Step Tutorials from Linked Source Code, Snippets, and Outputs. In *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*. ACM, New York, USA, 1–12.

[22] Jeffrey Heer. 2019. Agency plus automation: Designing artificial intelligence into interactive systems. *Proceedings of the National Academy of Sciences* 116, 6 (2019), 1844–1850.

[23] Michael Hind, Sameep Mehta, Aleksandra Mojsilovic, Ravi Nair, Karthikeyan Natesan Ramamurthy, Alexandra Olteanu, and Kush R. Varshney. 2018. Increasing Trust in AI Services through Supplier's Declarations of Conformity. *CoRR* abs/1808.07261 (2018).

[24] Weixiang Hong, Kaixiang Ji, Jiajia Liu, Jian Wang, Jingdong Chen, and Wei Chu. 2021. GilBERT: Generative Vision-Language Pre-Training for Image-Text Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1379–1388. https://doi.org/10.1145/3404835.3462838

[25] Youyang Hou and Dakuo Wang. 2017. Hacking with NPOs: Collaborative Analytics and Broker Roles in Civic Data Hackathons. *Proceedings of the ACM on Human-Computer Interaction* 1 (12 2017), 1–16. https://doi.org/10.1145/3134688

[26] Yue Hu and Xiaojun Wan. 2014. PPSGen: Learning-based presentation slides generation for academic papers. *IEEE transactions on knowledge and data engineering* 27, 4 (2014), 1085–1097.

[27] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, Breckenridge, Colorado, USA, 4904–4916.

[28] Project Jupyter. 2015. *Project Jupyter: Computational Narratives as the Engine of Collaborative Data Science*. the Helmsley Trust, the Gordon and Betty Moore Foundation and the Alfred P. Sloan Foundation. https://blog.jupyter.org/project-jupyter-computational-narratives-as-the-engine-ofcollaborative-data-science-2b5fb94c3c58

[29] Project Jupyter. 2018. *JupyterLab: the next generation of the Jupyter Notebook*. Project Jupyter. https://blog.jupyter.org/jupyterlab-thenext-generation-of-the-jupyter-notebook-5c949dabea3

[30] Project Jupyter. 2021. *nbconvert*. Project Jupyter. https://github.com/jupyter/nbconvert

[31] Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020. Learning and Evaluating Contextual Embedding of Source Code. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, Breckenridge, Colorado, USA, 5110–5121.

[32] DaYe Kang, Tony Ho, Nicolai Marquardt, Bilge Mutlu, and Andrea Bianchi. 2021. ToonNote: Improving communication in computational notebooks using interactive data comics. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, United States, 1–14.

[33] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E John, and Brad A Myers. 2018. The Story in the Notebook: Exploratory Data Science Using a Literate Programming Tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, United States, 1–11.

[34] Miryung Kim, Thomas Zimmermann, Robert DeLine, and Andrew Begel. 2016. The emerging role of data scientists on software development teams. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. IEEE, ACM, New York, NY, USA, 96–107.

[35] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian E Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica B Hamrick, Jason Grout, Sylvain Corlay, et al. 2016. *Jupyter Notebooks-a publishing format for reproducible computational workflows*. Vol. 2016. IOS Press, Virginia, USA.

[36] Laura Koesten, Emilia Kacprzak, Jeni Tennison, and Elena Simperl. 2019. Collaborative practices with structured data: Do tools support what users need?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–14.

[37] Sean Kross and Philip Guo. 2021. Orienting, Framing, Bridging, Magic, and Counseling: How Data Scientists Navigate the Outer Loop of Client Collaborations in Industry and Academia. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW2 (2021), 1–28.

[38] Sean Kross and Philip J Guo. 2019. Practitioners teaching data science in industry and academia: Expectations, workflows, and challenges. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. ACM, New York, USA, 1–14.

[39] Erin LeDell and Sebastien Poirier. 2020. H2O AutoML: Scalable Automatic Machine Learning. In *Proceedings of the AutoML Workshop at ICML*, Vol. 2020. automl.org, Freiburg and Hannover, Germany.

[40] Xingjun Li, Yuanxin Wang, Hong Wang, Yang Wang, and Jian Zhao. 2021. NBSearch: Semantic Search and Visual Exploration of Computational Notebooks. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, Vol. abs/2102.01275. ACM, New York, USA, 1–14.

[41] Xingjun Li, Yizhi Zhang, Justin Leung, Chengnian Sun, and Jian Zhao. 2021. EDAssistant: Supporting Exploratory Data Analysis in Computational Notebooks with In-Situ Code Search and Recommendation. https://doi.org/10.48550/ARXIV.2112.07858

[42] Denghui Liu, Chi Xu, Wenjun He, Zhimeng Xu, Wenqi Fu, Lei Zhang, Jie Yang, Zhihao Wang, Bing Liu, Guangdun Peng, et al. 2021. AutoGenome: an autoML tool for genomic research. *Artificial Intelligence in the Life Sciences* 1 (2021), 100017.

[43] Ke Liu, Guang Yang, Xiang Chen, and Chi Yu. 2022. SOTitle: A Transformer-based Post Title Generation Approach for Stack Overflow. *CoRR* abs/2202.09789 (2022).

[44] Xuye Liu, Dakuo Wang, April Yi Wang, Yufang Hou, and Lingfei Wu. 2021. HA-ConvGNN: Hierarchical Attention Based Convolutional Graph Neural Network for Code Documentation Generation in Jupyter Notebooks. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*. Association for Computational Linguistics, Pennsylvania, USA, 4473–4485.

[45] Antonio Mastropaolo, Simone Scalabrino, Nathan Cooper, David Nader Palacio, Denys Poshyvanyk, Rocco Oliveto, and Gabriele Bavota. 2021. Studying the usage of text-to-text transfer transformer to support code-related tasks. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, IEEE, New York, NY, USA, 336–347.

[46] Andreas Mathisen, Tom Horak, Clemens Nylandsted Klokmose, Kaj Grønbæk, and Niklas Elmqvist. 2019. InsideInsights: Integrating Data-Driven Reporting in Collaborative Visual Analytics. *Comput. Graph. Forum* 38, 3 (2019), 649–661.

[47] Michael Muller, Ingrid Lange, Dakuo Wang, David Piorkowski, Jason Tsay, Q Vera Liao, Casey Dugan, and Thomas Erickson. 2019. How data science workers work with data: Discovery, capture, curation, design, creation. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. ACM, New York, USA, 1–15.

[48] Andrew Myers. 2022. *In Human-Centered AI, the Boundaries Between UX and Software Roles Are Evolving*. Stanford University. https://hai.stanford.edu/news/human-centered-ai-boundaries-between-ux-and-software-roles-are-evolving

[49] Microsoft Office. 2021. Make Your PowerPoint Presentations Accessible to People with Disabilities. https://support.microsoft.com/en-us/office/make-your-powerpoint-presentations-accessible-to-people-with-disabilities-6f7772b2-2f33-4bd2-8ca7-dae3b2b3ef25

[50] Samir Passi and Steven J Jackson. 2018. Trust in data science: Collaboration, translation, and accountability in corporate data science projects. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–28.

[51] Jeffrey M Perkel. 2018. Why Jupyter is data scientists' computational notebook of choice. *Nature* 563, 7732 (2018), 145–147.

[52] David Piorkowski, Soya Park, April Yi Wang, Dakuo Wang, Michael Muller, and Felix Portnoy. 2021. How AI Developers Overcome Communication Challenges in a Multidisciplinary Team: A Case Study. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (2021), 1–25.

[53] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. https://doi.org/10.48550/ARXIV.1910.10683

[54] Bernadette M Randles, Irene V Pasquetto, Milena S Golshan, and Christine L Borgman. 2017. Using the Jupyter notebook as a tool for open science: An empirical study. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. IEEE, IEEE Computer Society, Washington, DC, USA, 1–2.

[55] Adam Rule, Ian Drosos, Aurélien Tabard, and James D Hollan. 2018. Aiding collaborative reuse of computational notebooks with annotated cell folding. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–12.

[56] Adam Rule, Aurélien Tabard, and James D Hollan. 2018. Exploration and explanation in computational notebooks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–12.

[57] Cliff Click SriSatish Ambati. 2021. *H2O*. H2O.ai. https://h2o.ai

[58] Krishna Subramanian, Johannes Maas, and Jan Borchers. 2020. Tractus: Understanding and supporting source code experimentation in hypothesis-driven data science. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–12.

[59] Edward Sun, Yufang Hou, Dakuo Wang, Yunfeng Zhang, and Nancy Xin Ru Wang. 2021. D2S: Document-to-Slide Generation Via Query-Based Text Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*. Association for Computational Linguistics, Pennsylvania, USA, 1405–1418.

[60] Harini Suresh, Steven R Gomez, Kevin K Nam, and Arvind Satyanarayan. 2021. Beyond expertise and roles: A framework to characterize the stakeholders of interpretable machine learning and their needs. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–16.

[61] Maria Tsiakmaki, Georgios Kostopoulos, Sotiris Kotsiantis, and Omiros Ragos. 2019. Implementing AutoML in educational data mining for prediction tasks. *Applied Sciences* 10, 1 (2019), 90.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[63] April Yi Wang, Dakuo Wang, Jaimie Drozdal, Michael Muller, Soya Park, Justin D Weisz, Xuye Liu, Lingfei Wu, and Casey Dugan. 2022. Documentation Matters: Human-Centered AI System to Assist Data Science Code Documentation in Computational Notebooks. *ACM Transactions on Computer-Human Interaction* 29, 2 (2022), 1–33.

[64] April Yi Wang, Zihan Wu, Christopher Brooks, and Steve Oney. 2020. Callisto: Capturing the" Why" by Connecting Conversations with Computational Narratives. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–13.

[65] Dakuo Wang, Q Vera Liao, Yunfeng Zhang, Udayan Khurana, Horst Samulowitz, Soya Park, Michael Muller, and Lisa Amini. 2021. How much automation does a data scientist want? *arXiv preprint arXiv:2101.03970* abs/2101.03970 (2021).

[66] Dakuo Wang, Lingfei Wu, Xuye Liu, Yi Wang, Chuang Gan, Jing Xu, Xue Ying Zhang, and Jun Wang. 2022. Learning-based automated machine learning code annotation with graph neural network. US Patent App. 17/088,018.

[67] Dakuo Wang, Lingfei Wu, Yi Wang, Xuye Liu, Chuang Gan, Si Er Han, Bei Chen, and Ji Hui Yang. 2022. Learning-based automation machine learning code annotation in computational notebooks. US Patent App. 17/069,402.

[68] Yun Wang, Zhida Sun, Haidong Zhang, Weiwei Cui, Ke Xu, Xiaojuan Ma, and Dongmei Zhang. 2019. DataShot: Automatic generation of fact sheets from tabular data. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 895–905.

[69] Yue Wang, Weishi Wang, Shafiq Joty, and Steven CH Hoi. 2021. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. *arXiv preprint arXiv:2109.00859* abs/2109.00859 (2021).

[70] Zezhong Wang, Shunming Wang, Matteo Farinella, Dave Murray-Rust, Nathalie Henry Riche, and Benjamin Bach. 2019. Comparing effectiveness and engagement of data comics and infographics. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–12.

[71] Zijie J Wang, Katie Dai, and W Keith Edwards. 2022. StickyLand: Breaking the Linear Presentation of Computational Notebooks. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. ACM, New York, USA, 1–7.

[72] John Wenskovitch, Jian Zhao, Scott Carter, Matthew Cooper, and Chris North. 2019. Albireo: An interactive tool for visually summarizing computational notebook structure. In *2019 IEEE visualization in data science (VDS)*. IEEE, IEEE, New York, NY, USA, 1–10.

[73] Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-Table: A New Way of Information Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, Pennsylvania, USA, 2518–2533.

[74] Yifan Wu, Joseph M Hellerstein, and Arvind Satyanarayan. 2020. B2: Bridging code and interactive visualization in computational notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, USA. https://doi.org/10.1145/3379337.3415851

[75] Doris Xin, Eva Yiwei Wu, Doris Jung-Lin Lee, Niloufar Salehi, and Aditya Parameswaran. 2021. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows, In CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021. *CoRR* abs/2101.04834, 83:1–83:16.

[76] Amy X Zhang, Michael Muller, and Dakuo Wang. 2020. How do data science workers collaborate? roles, workflows, and tools. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (2020), 1–23.

[77] Ting Zhang, Ivana Clairine Irsan, Ferdian Thung, DongGyun Han, David Lo, and Lingxiao Jiang. 2022. iTiger: An automatic issue title generation tool. *arXiv preprint arXiv:2206.10811* abs/2206.10811 (2022), 1637–1641.

[78] Jian Zhao, Shenyu Xu, Senthil Chandrasegaran, Chris Bryan, Fan Du, Aditi Mishra, Xin Qian, Yiran Li, and Kwan-Liu Ma. 2021. ChartStory: Automated partitioning, layout, and captioning of charts into comic-style narratives. *arXiv preprint arXiv:2103.03996* 29, 2 (2021), 1384–1399.

[79] Chengbo Zheng, Dakuo Wang, April Yi Wang, and Xiaojuan Ma. 2022. Telling Stories from Computational Notebooks: AI-Assisted Presentation Slides Creation for Presenting Data Science Work. In *CHI Conference on Human Factors in Computing Systems*. ACM, New York, USA, 1–20.

[80] Ingrid Zukerman and Diane Litman. 2001. Natural language processing and user modeling: Synergies and limitations. *User modeling and user-adapted interaction* 11, 1 (2001), 129–158.