Integrating Open AI into An R Shiny App: A Tool to Generate Pictures from Stories

Yunlei Tang, Danhe Tang, Prof. Matthew Lanham
Purdue University, Daniels School of Business
tang632@purdue.edu, tang631@purdue.edu, lanhamm@purdue.edu

ABSTRACT

We develop and test a 'Story Picture Generator' application using Open Artificial Intelligence (AI). The tool provides the user a R Shiny Graphical User Interface (GUI) that easily interfaces with OpenAI, providing text comprehension and drawing capabilities. By leveraging OpenAI's capabilities, the application generates relevant images based on the context and themes of the user's story. The motivation for this research is that there exist diverse audiences who seek custom images to support their stories and presentations. Presenters are often tasked with trying to source something "close" to what they are seeking by searching the internet, which is often not exactly what they want, and can be time-consuming. By using this picture generator, users can visualize their text content to amplify the message, aid in comprehension, and make the content more memorable. The Open AI Story Picture Generator aims to bridge this gap, offering an automated solution to provide visual augmentations tailored to the narrative's context. The application utilizes an R Shiny interactive interface where users can input their textual content to generate a corresponding custom image. By submitting the content, the app processes the text, extracting keywords and themes by connecting and using OpenAI's Large Language Models (LLMs). These are then sent to drawing OpenAI to return appropriate images. The Shiny app then displays these images alongside the story. The app is built to be intuitive, requiring no technical expertise from the user, but offers options for image customization and refinement as necessary.

INTRODUCTION

What is Shiny

Shiny is a package that makes it easy to build interactive desktop and web applications straight from R & Python. R Shiny harnesses the computational, statistical, and graphical power of R and channels it into interactive, web-based dashboards. It's a transformative tool that has reshaped the way data-driven decisions are made in many fields.

- **Shiny Dashboarding**: Shiny allows users to build interactive dashboards. Widgets like sliders, buttons, and dropdown menu users to manipulate data in real-time and see results immediately. Outputs on the Shiny dashboard, like plots or tables, are automatically updated when user inputs change. There are two key components of the R shiny:
  - **UI:** The user interface (UI), where analysts specify what they want their app to look like. This typically includes widgets like sliders, buttons, and plots.
  - **Server**: The server function contains the instructions that your computer needs to build the app. This is where developers tell the app what to do when a user moves a slider or clicks a button. It's basically the logic behind the app.

```
# Define the UI
ui <- fluidPage(
    # UI elements go here
)
```

```
# Define the server logic
server <- function(input, output, session) {
    # server code goes here
}
```

What is OpenAI

Founded in 2015, OpenAI artificial intelligence (https://openai.com/) research and development company focuses on researching and developing AI-based tools. OpenAI is known for its research in various areas of AI, including deep learning, reinforcement learning, and robotics. The organization has been at the forefront of developing large language models (LLMs), the GPT series like GPT-3, and GPT-4 which we are familiar, are the results of the development of LLMs, and other high-profile AI models like DALL·E. People are taking advantage of these AI-based tools to realize personal goals by inputting simple words or sourcing the Open AI to create personalized AI tools.

How can these be used together?

Given Shiny's robust interactive features and OpenAI's LLMs' precise text comprehension, we aim to combine R Shiny with OpenAI to develop innovative and useful tools. In this project, by combining Shiny's interactive capabilities with DALL·E's powerful image generation (driven by OpenAI), we can create a tool that transforms textual narratives into visual images, making storytelling more engaging and accessible. This is useful because the user does not need to know or use code or have other technical knowledge to use it for their work.

RESEARCH QUESTIONS

RQ1: How accurately and effectively does the AI interpret and translate textual content into relevant and contextually appropriate images?

RQ2: What are the limitations and challenges in ensuring the generated images align closely with the themes and messages of stories?

LITERATURE REVIEW

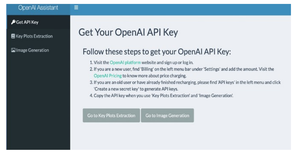
Our study focuses on the use of GPT 4.0 to capture story keywords and DALL·E 3 to generate customized images. We found some studies invested in these AI technologies in isolation or considered others.

Author, Year	Article	GPT 4.0	DALLE-3	MidJourney	DALLE-2	Craiyon
Oppenlaender, J., 2022	The creativity of text-to-image generation	No	No	**Yes**	**Yes**	No
Parsons, 2022	The DALLE-2 Prompt Book	No	No	No	**Yes**	No
Reviriego, P., & Merino-Gómez, E., 2022	Text to image generation: leaving no language behind	No	**Yes**	**Yes**	No	**Yes**
Zeqiang, L. et. al, 2023	Mini-DALLE3: Interactive Text to Image by Prompting Large Language Models	No	**Yes**	No	No	No
Our Study		**Yes**	**Yes**	No	No	No

Table 1: Literature Review Comparison of Technologies Used to Generate Custom Images

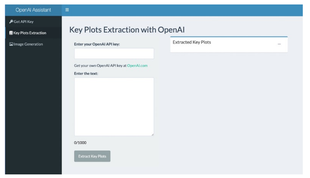
SHINY APP

Our app includes three pages.



Get API Key Page:
On this page, users are given instructions to get their own API key for further use of Open AI.

Key Plots Extraction Page:
This page is driven by GPT-4. Users first enter the API key on this page from the previous step. Then, put the text they intend to convert to an image in the "*Enter the Text*" box. After clicking the "*Extract Key Plots*" button, we can get the extracted key plot based on the text we enter.



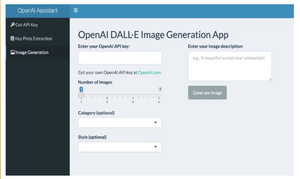
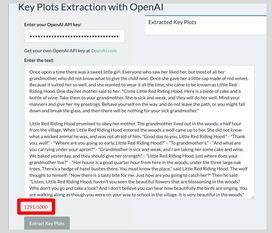


Image Generation Page:
This page is driven by DALL·E 3. The user can put any description in the text box, optionally select the style of the picture, and then click "*Generate Picture*" to get the desired picture. If the picture is not satisfying, the user can generate it repeatedly by adding more descriptions in the text box and clicking the button. Every time the user clicks the button, AI will generate a new picture and show it in the R shiny dashboard.

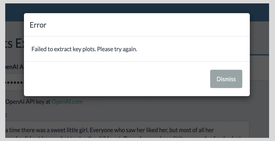
EVALUATION AND RESULT

To examine how accurately the application can convert textual stories into corresponding images. We design an experiment that is divided into four parts, aiming to find the most effective method to generate images that match the story content.

We take the classic fairy tale Little Red Riding Hood as an example, and the entire story is got from the American literature website (https://americanliterature.com/childrens-stories/little-red-riding-hood). There are 1,442 words in the text, filled with dialog, description, and mental activity, including a number of longer paragraphs.

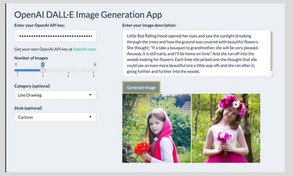


The first part of the experiment is extracting keywords using "*Key Plot Extraction*", which extracts key plots based on the user's input via GPT-4 and then outputs the key plots of the story back to the user interface. The experiment failed because the input text contained too many words and reached the GPT-4 limit (GPT-4 provides up to 8000 contextual tokens). In this case, we added the word count to remind the user not to enter too much content.

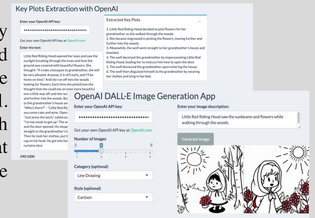


The second part of the experiment involved importing the complete story text directly into the image description box to generate images. The experiment failed for the same reason as the first experiment, which is the word limit.

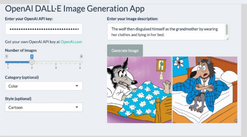
In the third experiment, we entered a paragraph from the story. The experiment was not very successful. Although DALL-E can handle relatively short texts, the images generated were far from what was expected and did not accurately reflect the content and style of the pictures (neither the line drawings nor the cartoons we selected from the sidebar options).



In the fourth experiment, we first extracted key episodes from selected passages and then used any of the extracted key episodes as image descriptions. This approach was successful. After accurately extracting key episodes with GPT-4, DALL-E is able to generate images that better match the story content based on more specific text prompts.



After these four experiments, we conclude that the best user experience is to first use the *Key Plot Extraction* to extract key plots from the whole story, and then use one of the extracted plots to generate an image, which gives satisfactory results.



CONCLUSIONS

In conclusion, our integration of R Shiny with OpenAI for our "Story Picture Generator" project underscores the vast potential at the crossroads of interactive platforms and AI models. The preliminary results show a transformative approach to visual storytelling. The AI's ability to interpret and translate textual content into images showed varying degrees of accuracy and effectiveness, largely dependent on the method of text processing and the complexity of the story content. Several key limitations and challenges were DALL-E's limitations in processing long text segments, and also inherent unpredictability in how AI understands and visualizes certain concepts, especially abstract or symbolic ones. Future experiments could delve into harnessing user preferences for image aesthetics, or introducing a feedback loop for continuous model optimization, and so on.

https://github.com/Yunlei02/Generate-Pictures-Tool