โครงการ CO-OP SHOWCASE : ก้าวแรกสู่อาชีพ ประจำปี 2567

I-BIT CWIE DAY 2024

ผลงานสหกิจศึกษาดีเด่น
ประเภทโครงงาน/ผลงาน
วิทยาศาสตร์และเทคโนโลยี

CODE

An Interactive Graph Website for Generative GraphLearner

Phichai Thumsema[a], Asst. Prof. Dr. Silanee Nuchitprasitchai[b], Asst. Prof. Dr. Pudsadee Boonrawd[c]

[a]FernUniversität Campus Hagen, Universitätsstraße 11, 58097 Hagen, Germany
[b]King Mongkut's University of Technology North Bangkok, 1518 Pracharad Road, Wong Sawang, Bang Sue, Bangkok 10800, Thailand

Abstract

The GraphLearner website is focused on creating a seamless website that can both calculate the prediction result and display the graph for the user to observe to gain more insight. The program has functionality that lets users choose which range of data they want to focus on by using the range slider. The graph itself is also intractable, so if the users do not prefer to use a range slider, they can also interact with the graph directly. For the framework, we utilize Flask for the backend and HTML, CSS, and JavaScript for the frontend. We also implement this program to be able to convert the numeric data into the form of text and convert the text into numeral data too by using the external program that we created.

Introduction

Nowadays, the complexity of generative AI is increasing, especially in AI systems that use deep learner models, which require computational resources and specialized knowledge to maintain and keep the system operating. Recognizing these challenges, we developed the GraphLearner web-based program to offer a more user-friendly and resource-efficient alternative. Our motivation starts with the desire to make AI training, predicting, and more user-friendly by providing graph figures on the website.

To enhance the user experience, we provided interactive data visualization and manipulation on the website. Focus on providing users with the option to choose which area of the graph they want to focus on, enabling users to gain deeper insights into the AI's learning process and predictions. As well as the purpose to develop the GraphLearner algorithm, which aims to approximate higherorder Markov chains while maintaining the computational efficiency of a first-order process.

Methodology

1. Data Collection • For collecting data we can take the input data directly from the user or import the existing file to train the GraphLearner as showcased in Figure 2. We also have a separate program with the purpose of converting the text file into numeral data that can be used to train the program. The common method we use to get required data is to take the information from Wikipedia convert the text in there into the numeral data and import it into the program. The data will be displayed at the bottom of the webpage as history data displayed in Figure 2

2. Making Prediction • After the user have input their root and number of branches and successfully submit, the program will use the value to search within each learner and return the result. However if there is no available data to make prediction the program will return error value on the webpage to inform the user that there is no suffice data. By using the root value that we got, our program will take this value and display it in the graph provided in the webpage as displayed in the Figure 1.

3. Choosing the Node • In the case of there's available data, the backend program will return that predicted value to the frontend for the user to choose which value they want. After the user has successfully selected the value, that value will be combined with the root along with previous data to create a single sequence that can be used as mentioned in the section below. This value will be displayed in the graph as demonstrated in Figure 1. As for the node selecting section in Figure 2.

4. Reinforcement Learning • The combined value displayed in the "Train Data Learner3" section in Figure 2 which is the combination of root value and selected node can be sent back to train the graph. As for the combined sequence that belongs to Learner number 3, when the user chooses to take that data to perform reinforcement learning, the data will be sent to only Learner number 3.

5. Translation • Utilizing our add-on mapping program we can convert the numeric data into the text, for example, if the value is [26, 52, 18, 20, 15, 4, 17, 2, 0, 17] when the user run the program this sequence will be converted into " A supercar ". The user also has a choice to convert the text file into numeric data that can be used as root values for prediction in our webpage. In this case, if the text is "Cat" when we run the program we will receive [28, 0, 19] in return.

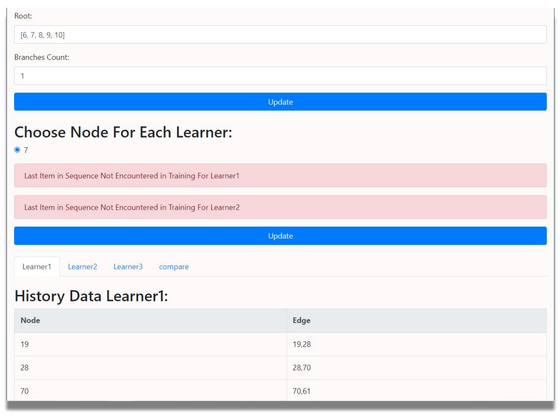

Figure 1

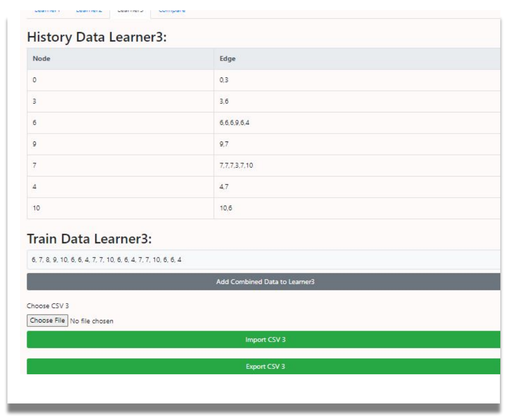

Figure 2

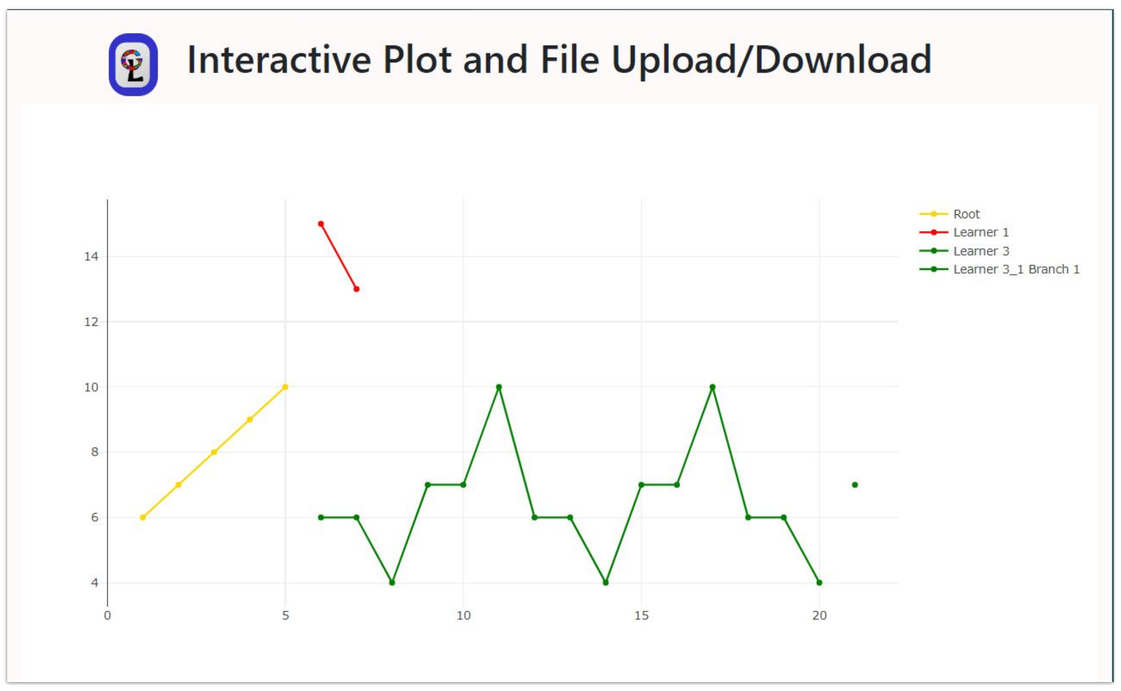

Figure 3: Graph representing the chosen value along with its predicted value

Results

The generated answers is displayed on the provided graph. There are also three graph learner in the web application, as a test case to see what will happened if there are multiple learner that use the same Python library and memories and only one of them is provided with the data. The result is that only one learner that got the data can answer a question similar to how only person in a group which study in medical field be able to answer a question regarding pills while the others can not.

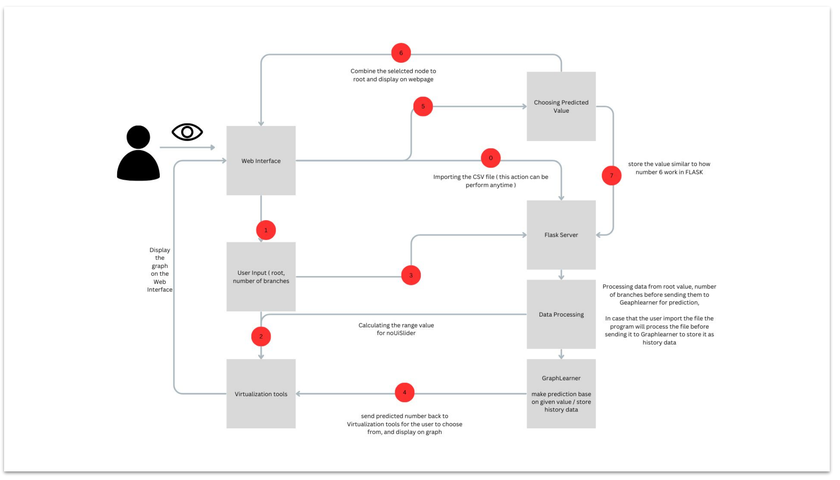

Figure 4 : Solfware Architecture of how graphlearner play it role within the backend program

Conclusion

In conclusion, our GraphLearner web-based program, which focuses on accessibility and a user-friendly experience, is created by using Flask as a backend, and for the frontend, we utilize HTML, CSS, and Javascript. In order to ensure that the website will be user-centered, we provide a range slider and interactive graph for the users to manipulate the parameters of the graph displayed on the webpage. During the development, we noticed some limitations of our program. The HTML did not support a range slider with two handles. We have to use the external library to overcome this limitation, and due to the functionality that we have, including all of them on one single webpage has become a challenge. The ability to handle large datasets efficiently, which aimed to capture the complexities of neocortical processes. Along with the aim to advancing the field of artificial general intelligence, this will be done in our future work.

Acknowledgements

This project is mainly focus on memory sufficiency and reduce the complexity of nowadays AI technology. I would like to show my gratitude to Prof. Dr.-Ing. habil. Dr. h.c. Herwig Unger, my benefactor who gave me the opportunity to have the cooperative program in Germany. The project is develop under his supervision along with Timothy Harrison another supervisor, thanks for the feedback provided by both of them the project Ui/Ux design can be develop to the acceptable point.

With their support we are be able to create the program that can predict the value and create the actual sentence regrading the knowledge that has been taught to the GraphLearner as well as creating the virtualize graph displaying the necessaries value. I would like to show my gratitude to those who supported me during the cooperative period regrading their feedback, the knowledge they provided me and the excellent working environment.

References

1. • Flux Copy. Learning TensorFlow and PyTorch in 2024: Build Powerful AI Models. [Internet]. Available from: https://www.linkedin.com/pulse/learning-tensorflow-pytorch-2024-buildpowerful-ai-models-flux-copy-eaede. Accessed: 2024-07-01.

2. • John Doe. Scikit-Learn, TensorFlow, PyTorch, Keras: But Where to Begin?. [Internet]. Available from: https://towardsdatascience.com/scikit-learn-tensorflow-pytorch-keras-but-where-tobegin-9b499e2547d0?gi=62656319d17b. Accessed: 2024-07-01.