Super Rapid Annotator Project at RedHenLab

Teng Chen

March 28, 2024

Summary of the Proposal

The proposal consists of five parts. The first part is background where we can see the basic points of the topic, unknowns and challenges about the topic. The second part shows the goal and objectives I set out for the project. In methods, we can see the challenges I will tackle, the result and the future of the project. The fourth part is timeline and the last part is references. For the part of methods, you can get more details on https://github.com/TengChen11/Ideas-For-Super-Rapid-Annotator.

Background

- Basically, the topic is about developing a system where users can utilize a tool to annotate videos and images. The system mainly has three parts, and I think the tool will be available to users in the form of a web-based application. Therefore, I should develop a gradio app for this tool and the gradio app is presented as a webpage. Users can download the source code and run a command like **python app.py** to start the gradio app. The logic based on all the above processes is as shown in the figure 1:

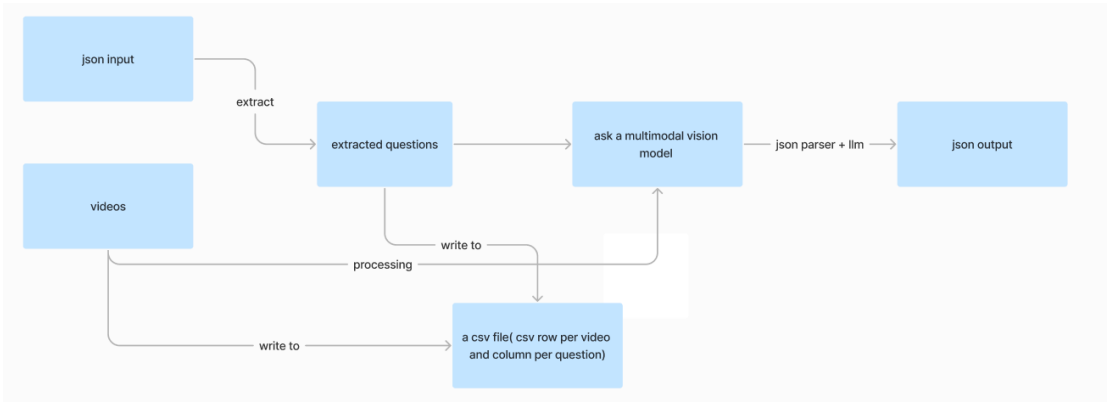


Figure 1: logic

- One of the challenges is how to use the APIs(like the APIs of huggingface, gradio, etc.) properly when developing the system. Another challenge can be how to choose the most suitable model from LLaVA[3], BakLLaVA[2], Video-LLaVA[1], etc.. Moreover, how to extract questions from a json input and how to generate json output after getting the output of a multimodal vision model are also challenges I have to take into account.
- The unknown I want to address is how to speed up model inference. Of course we need to make sure that everything has to work in a local GPU. I also want to know if there is a technique to allow the model to speed up inference while the model is working in a GPU.

Goal and Objectives

The goal 1 of the project is to extract the questions from a json input. Goal 1 will be met by using Kor which can help you extract structured data from text using LLMs. The goal 2 is to make sure that the model is suitable for the task. Goal 2 will be met by using Video-LLaVA, tentatively. The goal 3 is to generate json output. Goal 3 will be met by using strictjson. Moreover, Kor and Strictjson require a LLM.

Therefore, so I have to deploy a LLM to this app for Kor and Strictjson. The goal 4 will be met by using the python package named transformers. The goal 5 is to develop a gradio app for the tool. Goal 5 will be met by understanding how the following APIs of gradio are used:

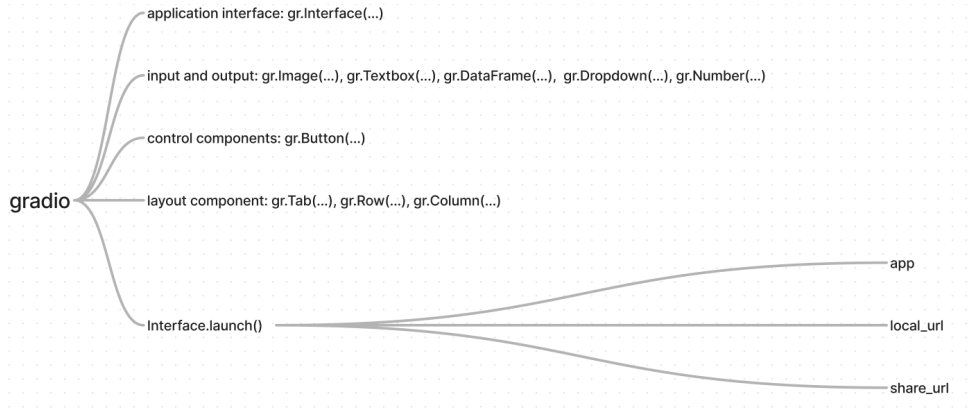


Figure 2: gradio APIs

Methods

- The challenges I will tackle: How to extract the questions from a json input? How to choose the most suitable model? How to generate json output? How to build a gradio application for the system? How to speed up model inference? Everything has to work without API calls to external service, and Kor and Strictjson require a LLM, so I need to deploy a suitable model to this app for Kor and Strictjson, which is also a challenge.
- To extract the questions from a json input, my peference is to use Kor, the process of it is as shown in the figure 3.

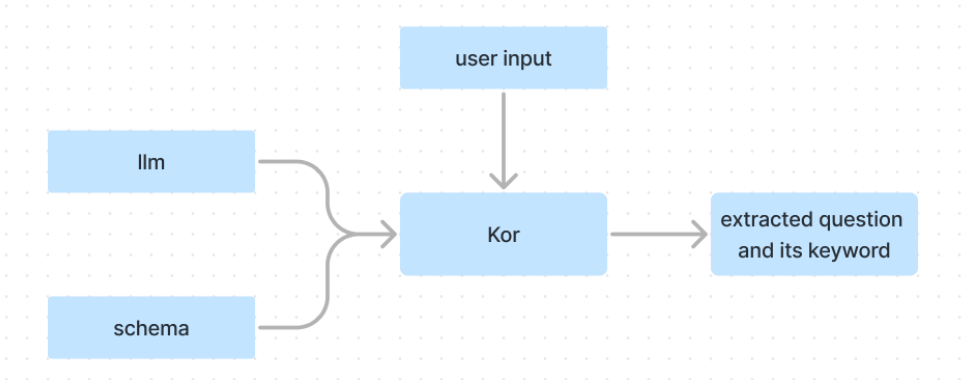


Figure 3: Kor usage

To generate json output, try to use strictjson. The basic usage of it is as shown in the figure 4.

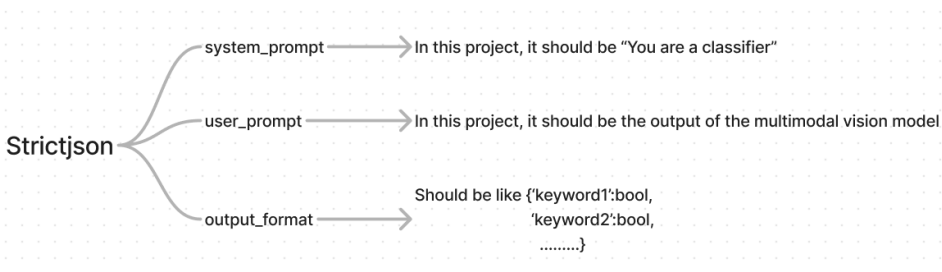


Figure 4: Strictjson usage

To develop a graido app, utilize the APIs shown in Figure 2 properly and the reason for using them is that they are fast and easy setup. The overall process of the system is shown in Figure 5:

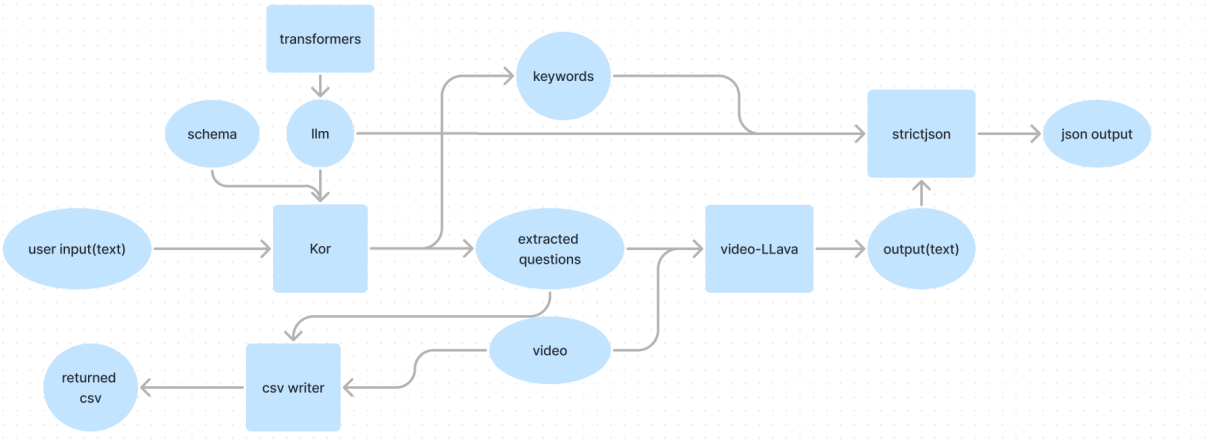


Figure 5: overall process

- No additional resources tentatively.
- A .zip file that contains the source code of the gradio app. When the use is complete, the app will return a csv file and it will be like(q means question):

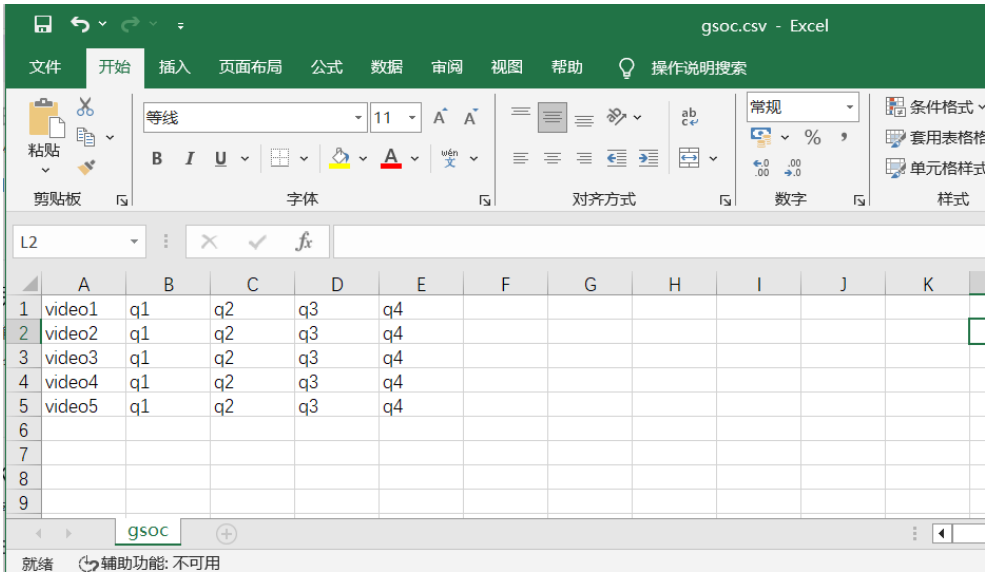


Figure 6: returned csv

And a tentative gradio app is shown in figure 7. As we can see, users can upload videos they want to annotate and send questions to the model. Then, the right side of the app can show the questions extracted from user input, show the json output and the csv file. Both the json output and the csv file can be downloaded. So far, we have only implemented the interface of the application, and the functions corresponding to each button have not been implemented. For more information, take a look to the link: https://github.com/TengChen11/Ideas-For-Super-Rapid-Annotator/blob/main/app.py.

- I expect the gradio app can support multiple languages and a technique which speeds up the model inference can be deployed to the gradio app in the future.

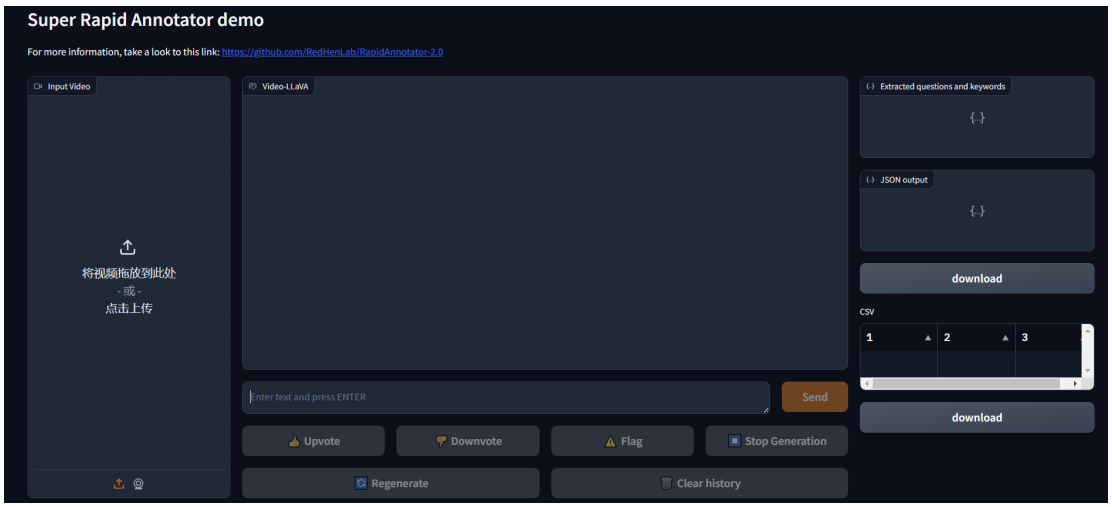
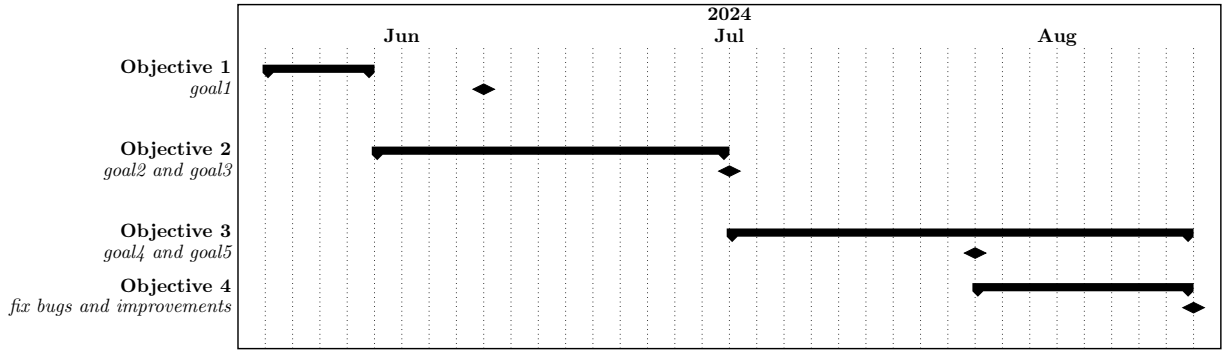


Figure 7: app

Tentative Timeline



References

[1] Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[2] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023.

[3] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916. Curran Associates, Inc., 2023.