Benjamin Houldridge

Current/Emerging Trends in Computer Science

CS 370 11999-M01

Module One Assignment

Several years ago, I was interested in learning how to use 3D modeling software. There was a poster in one of my labs with a cutaway look at a simplified nuclear reactor and I thought it would be fun to turn that into a small 3D mockup. The one barrier was turning the poster into a 3D printable file. I downloaded some modeling software and went through a couple tutorials. After failing to find a tutorial that would give me some of the shapes I needed and realizing how difficult modeling was to learn, progress slowed. Competing priorities pushed the project down my to-do list and eventually it fell off. With a little help or extra time, I would likely have learned how to do the 3D modeling. Many people would like to pick up skills with steep learning curves but the amount of time and effort to get off the ground can make these pursuits less feasible. For this essay I want to focus specifically on learning how to use applications with complex user interfaces. Video editing, audio recording, or my attempt to learn 3D computer modeling all fall into this category.

The best way for AI to help people learn complex user interfaces is with an integrated chat prompt that can generate tutorial style responses. So, let’s imagine a user is new to Blender, a comprehensive 3D modeling tool. They want to learn how to take a long cylinder and add a 90-degree bend in it. Instead of searching through the blender documentation or online videos, the user could ask the AI. “How do I put a bend in this cylinder?” The AI could respond with a text response walking through what values need to be changed or where to left click on the display to reveal a needed menu. Maybe the solution doesn’t work the way the user was intending and so can follow up with. “Can I bend it toward me instead of to the right?” Again the AI responds with instructions tailored to the user’s input.

This kind of application specific AI would be complex but ultimately save the time of users and improve the experience of using the software. To build such a system we need the AI to have several capabilities. It needs to be able to interpret user input, understand the application, and produce a response. A pretrained large language model can provide language comprehension and response capabilities. It’s becoming increasingly common to modify these base LLM models, and this is exactly what we will be doing here. If we just insert a pretrained LLM it will likely respond in ways that are not helpful to the user. We want the AI to respond to the state of the application and give highly factual information. To do this we will train with the intent to modify the capabilities of the base LLM. We will need a large dataset of sequential, and preferably labeled, application states, combined with user inputs, existing tutorials, and documentation. Training will preferably return a model that gives clear concise instructions on how to use the interface and responds to the specific states of the application. To run this training will require large cloud computing services and will need to be retrained occasionally to improve the model and update it as things change. The model will likely need to run in the cloud, so the user side application will need to support a connection to the servers that run the model. Each request will be sent with the user’s input, past conversation, and application state. Each reply will include a text response. This system will allow users to quickly learn exactly what they need, greatly shortening the time to get what they want done and improving their experience.

If we wanted to expand the capabilities of this AI, we could start with adding pictures to the responses. Screenshots showing where to click or what settings might look like could be helpful in providing clear instructions to the user. The next large step is to have the AI perform actions in the user interface by directly making API calls. This would look like the user asking the AI to change an object’s color and the AI response making API calls to the application to effectuate the color change. This could revolutionize how users interact with the application. Instead of learning commands and options users would have a conversation with the AI to get the outcome they desire.

The ethical concerns that come from this system stem in the gathering of the training data. The easiest and cheapest way to acquire a huge amount of data regarding how the application is run, is to let that data be generated by existing users. Saving sequences of application states from actual use would be great for training but can easily be an invasion of privacy. If we make the scraping of such data from users something they need to opt into, we may gain trust but miss out of valuable training data. Not making the request and simply scraping data could easily lead to loss of trust, leaking of intellectual property, or legal action.

An AI chat assistant built into applications that have complex user interfaces could greatly help users. It can speed up learning, help productivity, and potentially revolutionize how users interact with these applications.

**References**

*How to Train LLM on Your Own Data in 8 Easy Steps*. (2024, November 19). Airbyte.com; Airbyte. https://airbyte.com/data-engineering-resources/how-to-train-llm-with-your-own-data

‌