Live Mood

Intelligent Interactive Systems final project

Ting-Hsuan Lien
dept. Information Technology
Uppsala Universitet
Uppsala, Sweden
thlien@ntnu.edu.tw

Tzu-Hsiang Kao
dept. Information Technology
Uppsala Universitet
Uppsala, Sweden
tzkao@ntnu.edu.tw

Cheng-Han Shen
dept. Information Technology
Uppsala Universitet
Uppsala, Sweden
alan900324@gmail.com

***Abstract*—Abstract—In this project, we developed an intelligent interactive system featuring Furhat as a bartender. The system utilizes Large Language Models (LLMs) to generate distinct personalities for three virtual bartenders, each with unique characteristics and stories. This approach allowed us to create a more immersive and engaging user experience. To complement the personality generation, we implemented a Convolutional Neural Network (CNN) for real-time emotion detection, trained on the FER2013 dataset to classify user emotions with improved accuracy.**

The system integrates voice input, emotion recognition, and personality-driven interactions. Users can converse with the bartenders, experiencing dynamic dialogues shaped by emotional and contextual inputs. We addressed challenges such as transcription inaccuracies and formatting issues by optimizing prompts and refining system architecture. Our work demonstrates the potential of combining LLMs and emotion detection for creating responsive and personalized human-computer interactions. This system offers application of conversational AI in a virtual bartending scenario.

I. Introduction

In our project, we chose a bartender as Furhat's character rather than a mindfulness coach. This decision was based on our goal, outlined in the specialization section, to leverage Large Language Models (LLMs) to enable Furhat to embody at least two distinct and stable personalities. We believed that a bartender character could showcase more pronounced differences between personalities compared to a mindfulness coach.

For creating distinct personalities using an LLM, we decided to implement three different personas instead of just two. With only two personas, Furhat's gender (male and female) paired with varied appearances and voices would naturally create significant differences. To explore further diversity, we added a third female persona, aiming to test whether interactions with two female bartenders could still feel distinct.

In addition to using an LLM for personality generation, we implemented a CNN-based emotion detection model. By training this model with a large dataset, we aimed to achieve higher accuracy compared to using Scikit-learn-based models.

A. Contributions

- **Ting-Hsuan Lien**: Responsible for the main programming tasks.
- **Cheng-Han Shen**: Focused on training the CNN for emotion detection.
- **Tzu-Hsiang Kao**: Designed and created the prompts for the LLM.

II. Methodology

A. Overall system design

The overall system design is showed in Fig. 1.

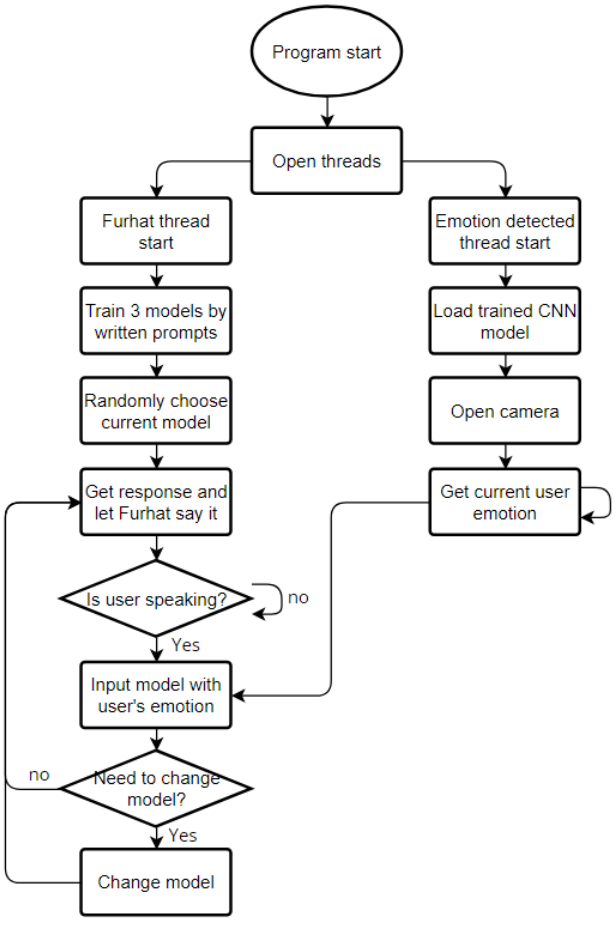

Fig. 1. Overall System Design

B. User Perception sub-system

In the User Perception subsystem, users can interact with Furhat through voice input, and their emotions are detected via the camera to provide a superior interactive experience. For emotion detection, we use OpenCV (cv2) to control the camera while annotating the user's current emotion on the screen. Furthermore, a Convolutional Neural Network (CNN) is employed as the emotion prediction model, with the detailed operation of the CNN elucidated in the subsequent section.

C. Interaction sub-system

In the Interaction subsystem, we receive two inputs: the user's emotional state from the User Perception subsystem and the transcribed text of the user's speech, obtained directly using Furhat's built-in functions.

Using these two inputs as a foundation, we employ a Large Language Model (LLM) to generate both the content and gestures that Furhat needs to use when responding to the user. Additionally, users can choose to interact with different virtual bartenders, each having a distinct personality and background. We offer three unique personas for conversation, all powered by the same LLM, ensuring consistent training while delivering diverse interactions.

The detailed workings of the LLM will be explained in the following section.

D. Specialisation

1) Convolutional Neural Network (CNN): Due to the low accuracy of emotion recognition using py-feat, which could not meet our requirements, we decided to train our own model for emotion classification. We chose the FER2013 dataset [1] to train and test a CNN model. The dataset contains multiple grayscale facial expression images labeled into seven emotion categories: anger, disgust, fear, happiness, sadness, surprise, and neutral. A total of 35,860 images were used for training. The dataset was split into training and testing sets, with the training set further divided into training and validation subsets at a 50/50 ratio. All images were resized to 48x48 pixels and normalized to the [0, 1] range.

Our model employed a Convolutional Neural Network (CNN) architecture, consisting of three convolutional layers with 32, 64, and 128 filters, respectively. Each layer was followed by MaxPooling and Dropout layers to extract features and prevent overfitting. Finally, the model included a fully connected layer with 128 neurons and a Softmax activation function for the output layer. The optimizer used was Adam with a learning rate of 0.001, and the loss function was categorical cross-entropy. Accuracy was used as the evaluation metric. During training, the batch size was set to 32, and the total number of epochs was 50. The steps per epoch were calculated based on the number of samples in the training and validation sets.

After evaluation on the test set, the model achieved a test accuracy of 54.14%.

2) Prompt Engineering: We use Gemini 1.5 as our tool to train model. We designed three different personality characters based on three different MBTI personalities (ENTP, ESFJ, INFP). We wrote different prompt for each personality, each of them has its own name, its own interests, its own reactions to things based on what we trained them.

We use format input and format output for easy use in our coding section. The format input includes "user_emotion", "user_text", "called_by". "user_emotion" is the current user's emotion in text. "user_text is the current user's text. "called_by" is who is calling this personality, since we had designed that user can call other personality to talk to, so the personality might be called by his colleague instead of the user, so it should respond differently.

The format output contains "furhat_emotion", "furhat_text", and "personality". "furhat_emotion" is what emotion futhat should express now, because we don't want to deal with the response text and try to get a fine emotion, since it's an NLP problem. Instead of using NLP to process the text to get the appropriate expressions, we thought it would be more accurate to let Gemini generate them for us directly. "furhat_text" is the response text that furhat will say. "personality" is the next personality the user will talk to. Since we have the "people switch" feature, it will be easier in our code section if we let LLM tell us who the user will talk to next.

We also wrote some important rules to make sure that the mods don't get caught up, such as not mentioning the Gemini, not mentioning how they were trained.

III. General Discussion

A. Overall Pipeline

Originally, we didn't want to use CNNs for image recognition because we thought that the prompt project was already a lot of work, even more than applying CNNs. However, in the first two reviews, it seems to be recognized that more can be done than just applying prompt engineering. So before the third review, we implemented and applied it, and it was approved.

When writing the program, we put a lot of effort into the articulation because we had three personalities to switch between. In order to make the user really feel like they are talking to different people, we also did face and voice switching.

B. Challenges

1) Transcribe: The transcribe that Furhat's built-in functions do is often not accurate. This poses a great difficulty for us, as names are often unrecognizable. We find out that the name with no similar words are more easily to recognize, such as Tom, Jenny and Sophia. So these three names are our bartender's final names. The way we decided on these names was through constant testing by prefixing the name with "Can I talk to ...", "I want to talk to ..." and see which name has a better recognition rate.

2) Format Output: We expected the format output to be a JSON file, and it often is. But because it's an LLM model, there's still a small chance that formatting errors will occur and stop our program. We've tried to avoid this by emphasizing the importance of formatting in the prompt.

3) Accuracy: While the model demonstrated some capability in emotion classification, there is still room for improvement. Factors affecting accuracy may include subtle differences in facial expressions in the FER2013 dataset, as well as variations in lighting and occlusions. The relatively simple model architecture and high Dropout rates might have also limited the model's ability to learn features effectively.

C. Usage of AI Tool

1) Microsoft Copilot: We use Microsoft Copilot to help us write the main program faster.

2) DeepL Write: We used DeepL Write to help us write this report, ensuring that our wording was more precise and coherent.

3) ChatGPT: In the CNN part, we use ChatGPT to help us write the code as we train our emotion detect model.

4) Gemini: We used Gemini as the main LLM model, and three different personalities of bartenders were formed after the prompt was dropped into Gemini.

D. Ethical Issues

Because we use so many AI tools, we have no way of ensuring that their training process and the training materials are ethical. It is difficult for us to ensure that LLM's models do not make discriminatory, offensive statements. These are potential ethical issues.

IV. Conclusion

For this final project, we implemented a project that allows furhat to chat with people. Our scene was set in a bar with three different bartenders to chat with. Each bartender had their own personality and story. We used CNN to recognize the user's facial expressions and provide them as input to a trained model. Our model was trained using Gemini 1.5 and was prompt engineered. Since there are three bartenders with different personalities, we wrote three prompts in total. In the end, we made a bartender that can chat normally, has facial expressions, and can switch to different people.

References

[1] *Kaggle - FER-2013*. URL: https://www.kaggle.com/datasets/msambare/fer2013.

Appendix