Emotionally Informed Information Kiosk Robot

Zak Quor

Abstract—This project sought to produce an emotionally informed AI assistant which would incorporate a user's detected emotion with input text and provide appropriate responses based on the two parameters. Research was undertaken to select a reasonable emotional model which ended up being Ekman's model plus the two additional emotions of contempt and neutral. The CNN architecture ConvNextv2_pico was trained on AffectNet data but the method used produced an unreliable model. The LLM ChatGPT 4o-mini was used as the base for the conversational assistant and was configured by an initialisation prompt to become an information kiosk robot. The rules for understanding how to respond to different emotions was also provided in the initialisation prompt. The ChatGPT model behaved quite well to each emotion and gave appropriate responses to the user based on the emotion it was provided with the user's text.

Index Terms—AI Chatbot, Emotion Recognition, Customer Service, Robot Assistant, Machine Learning

1 Introduction

We are Team 2 consisting of Andrew, Jeffrey and Zak. We haven't any particular team logo or special team name other than Team 2; which we are happy to identify as.

2 Planning

The general structure of our project was centred around semester 2 at the University of Auckland. Specifically the course CompSys731 and its taught material.
Weeks 1-6 were spent learning about robots (both commercial and research-orientated), emotional models, machine learning and facial recognition. During weeks 1-6 we planned out this research project by researching different technologies, and selecting a use case for an emotionally informed service robot. We chose an information kiosk in an airport, mall, or train station.
The following is the Gantt chart we used to drive and deadline different areas of our work.

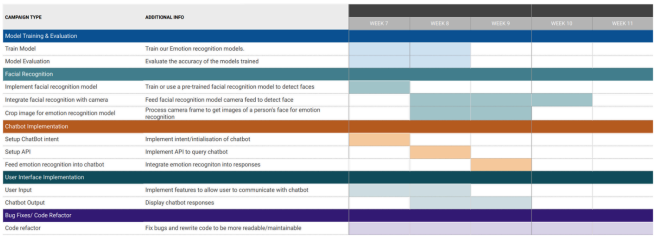

Figure 1. Gantt chart outlining the practical implementation timeline of our research project.
As far as each person's role was concerned, we opted to lean into each other's strengths. We held an initial meeting to review each component of the project (by looking at the Gantt chart which contained the required components) and discussed which part each person would do. Andrew mentioned he had experience with using chat GPT for previous projects, Jeffery was happy to work in any area, and for reasons mentioned previously, I was allocated the training. We meet each week on Fridays for most of the project to review the previous work for the week and plan our work for the next week.
The technology we chose to use was ChatGPT for its useability, LLM conversational adaptability and a team member's prior experience with it. Same justification for Whisper STT which integrates well with ChatGPT as it is developed by the same company.
YOLOv8 was chosen for its wide acceptance as being a reliable and fast facial detection algorithm with easy integration.
Python was chosen to be the programming language because of its ease-of-use, compatibility with the above 2 technologies, and the number of libraries available for it (including machine learning with pytorch and GUI interfaces). It also can be built and run easily which is good for quick prototyping.

3 Design Software Architecture

3.1 Purpose

As alluded to in the planning section, our project's goal is to develop an emotionally informed service robot for a particular use case. Our implemented chosen use case is an information kiosk in Westfield St Lukes Mall located in Auckland New Zealand. While we are not truly connected to the database of Westfield St Lukes, we have prompted GPT to pretend as if it were whilst also collecting as much information as it can from the internet. For demonstration purposes, this 'pretending' is an adequate facsimile of what a fully integrated information kiosk could look like.
We initially prompted ChatGPT with a series of statements defining its behaviour and setting rules for its reactions to user questions or statements. Most of these rules are around how to respond appropriately to each of the emotions defined in our emotional model (described later in this report). The overall nature of the robot behaviour can be summarised as; responding neutrally and quickly to people experiencing aggressive emotions, empathetic and supportive to those feeling sad, and energetic and offering fun and detailed suggestions to those experiencing positive emotions. The target outcome ideally being an information kiosk that is more enjoyable and engaging for people to use. If they are busy or

frustrated, they should be given the information quickly without causing delay or being irritating. If they are sad, it should cheer them up slightly to make their shopping experience better, and if they are happy it should reinforce that and provide an engaging shopping experience.

3.2 Database, Methods and Model

The architecture of our model training algorithm is as follows.

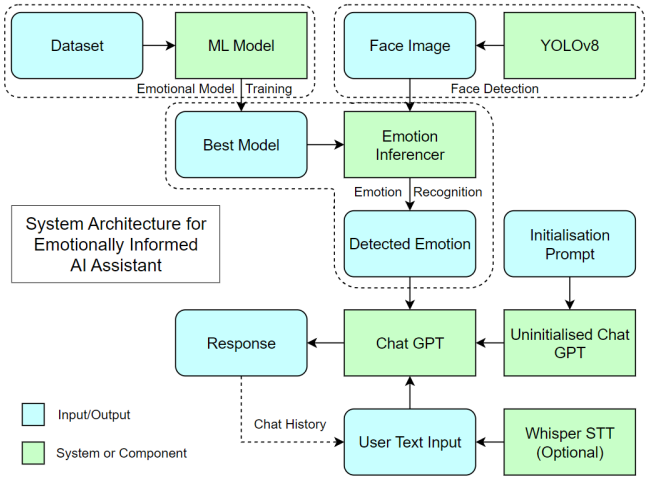

Figure 2. Full system architecture of the AI chatbot. N.B. GUI component omitted.

Our architecture is essentially broken down into 4 sections. They are as follows:

1) Model Training
2) Facial Detection
3) Emotional Recognition
4) ChatGPT

Please note: The Graphical User Interface (GUI) has been omitted from the figure as it has no significant effect on the functionality of the system but it is included with the code and adds to increase ease of use for the user/tester.

The model training section describes the train_extended.py code in our repository. When this file is run it uses the parameters defined in the __init__ section of the class Trainer, and trains the provided CNN architecture on the provided dataset. It outputs each epoch of the model to the defined run_name checkpoints folders. The short checkpoints files hold the following information in figure 3.

```
# Save short-style checkpoint (state_dict)
checkpoint = {
    'epoch': epoch,
    'model_state_dict': self.model.state_dict(),
    'optimizer_state_dict': self.optimizer.state_dict(),
    'accuracy': accuracy,
    'f1_score': f1,
    'precision': precision,
    'recall': recall
}
```

Figure 3. Object structure of a short-style checkpoint.

If the accuracy of the current model is better than the last, it is also saved in a folder in a separate directory (outside of Model_Training) which is the Facial_Detection directory. These 'best' models can be used in the emotional inferencer.

The provided image set/dataset to train our model on was the AffectNet dataset, which we were instructed on how to use in a laboratory tutorial and was specifically mentioned in the project overview. The data had an approximately 95/5 split between train and test images. We used this AffectNet data in the training of a model imported using the TIMM library of CNN models. The particular model used was convnextv2_pico.fcmae_ft_in1k. This model is a ConvNext2 architecture downsized to run and train more efficiently (denoted by pico), and has the option to be read pre trained on using FCMAE and the ImageNet1k dataset.

From the research stage of our project, we determined that the best model for our use case is the Ekman's Emotional model + neutral and contempt. The reasoning is because customers are likely to feel the emotions defined in Ekman's model + neutral and contempt in a public shopping setting more likely than others models which describe arousal and other unrelated emotions. The one exception to this is disgust which we feel is somewhat unlikely to be felt by a user in our use case. However, the AffectNet dataset is very conveniently the exact dataset containing the emotions for the model previously described. Therefore we were comfortable using it to begin with.

For AffectNet in general, the dataset ended up appearing to us to be one of the least consistent datasets for labelling. We felt that a lot of the images had been mislabelled. Due to this we tried several approaches:

1) Adding approximately 10 images of our own faces into the AffectNet dataset for each emotion/label with an approximately 70/30 split into test and train.
2) Similarly, add our images to the ImageNet dataset and train on those images.
3) Trying different split percentages up to 90/10.
4) Use a different model architecture, notably VGG16, also using the changes mentioned above.

With the architecture, we also experimented with other TIMM available models but they did not appear to provide any more accuracy on either dataset. This will be expanded upon in Section 5. Results. I coded the train_extended.py file to have numerous additional automated functions in addition to training such as splitting the file save locations, appending the metrics data, and extracting the convolution matrices. The training file has been designed for it to be relatively easy to change the dataset, model architecture, hyper parameters, save locations, optimiser and scheduler. This has been useful for trying the variety of different training methods described earlier in order to improve the emotional recognition accuracy (despite the unsatisfactory results of the different training methods). At this stage, our final submission is a ConvNext2 pico model trained on the AffectNet dataset. This model is set in the facial_detection.py file, clearly marked as emotion_model_path.

The YOLOv8 integration was very easy as it can quickly and reliably detect a person's face. This component is widely known for being accessible and robust. It should be noted that the emotional recognition feature of YOLOv8 was not used as it was explicitly mentioned in the project outline as one of the limitations.

Inserting the best model into the facial_detection.py was

also quite straightforward and streamlined the process of detecting the face and getting the emotion. This whole process is set up to run as a thread and operates concurrently with the GUI and ChatGPT.
We evaluated the two options for LLM AI chatbot and determined ChatGPT (GPT) to be better for our application than DialogFlow, primarily due to its ability to handle divergent conversations, easier role initialisation, and simpler python integration. We've found that it can respond reasonably well to the request we have made and the emotions that we passed in.

4 Results

4.1 Model Performance

The first thing done was use the provided Dataset and model, which we were able to train to an accuracy of 81.47% over 59 epochs. The problem with this is that we suspected that it was being over fitted. We inspected the dataset and saw that the training and test split was very high, about a 95/5 split. We also noticed that there were a few training images in the test data. Anecdotally, we found that testing it on our faces appeared to give us stable emotions on about 3 out of the 8 emotions, and we had to pull quite exaggerated or specific faces for the motions to change to that which we desired. We extracted a confusion from this model which is shown in Figure 4.

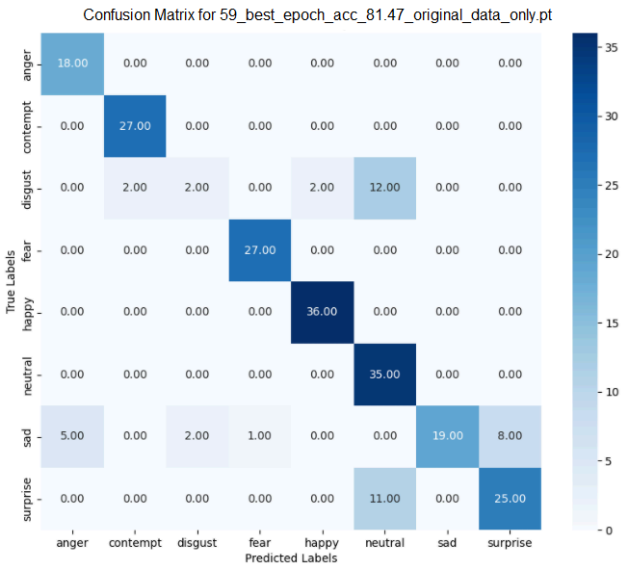

Figure 4. Confusion Matrix for Epoch 59 using the original AffectNet data.
It can be seen that sadness, surprise and disgust are the most commonly mislabelled emotions, but also that the confidence scores for the other emotions are quite low, especially anger. This aligns with our testing of the webcam on our own faces.
We extracted other relevant model metrics, such as f1 score, which is displayed in the following figure.

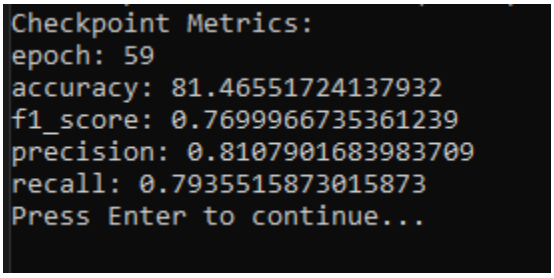

Figure 5. Machine Learning Metrics for Epoch 59 using the original AffectNet data.
These results can be considered quite good if taken at face value, but while the model's performance in our program was occasionally correct, it was not sufficient enough to consistently detect even the most exaggerated facial expressions of each emotion. This was in spite of our best efforts. We attempted to remedy this by adding about 50 images of our own faces in for each emotion that performed poorly in the original and training it from scratch to around the same epoch level. This lowered the maximum accuracy to 72.69 % and it still experienced the same inconsistency as the previous model. At the time of writing this report, we had lost the exact dataset relating to this model but recreated the test set to the best of our ability and extracted the following confusion matrix.

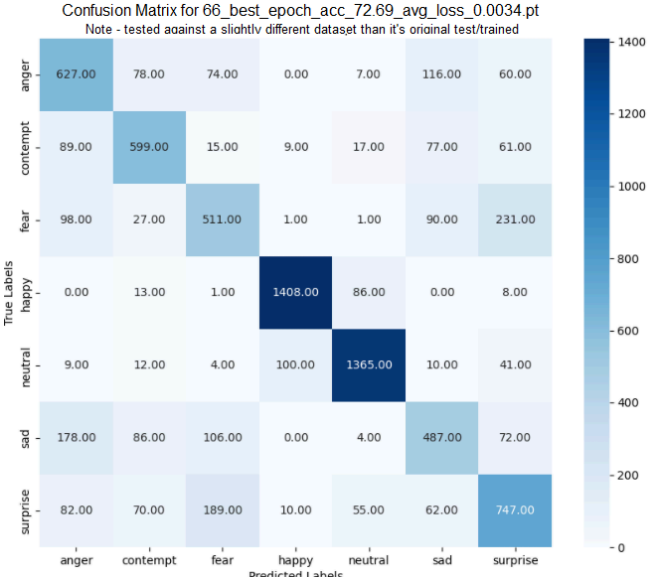

Figure 6. Confusion Matrix for Epoch 66 using the original AffectNet data + our own faces.
You may notice that the scale of the level of confusion is different to the original model but this is due to an increase in images in the test set. Please observe the relative values of this matrix or the colour gradient of the heatmap.
This confusion matrix vs the original one shows that the addition of our faces to selected emotions reduced the intensity of some of the mislabelling, for example the original had a high surprise mislabelled as neutral. However, it did not in turn explicitly increase the frequency of correct labelling, instead it spread the difference over a wider variety of emotions. I believe this made the model more likely to detect an incorrect emotion but have less confidence in the score of the incorrect emotion. Upon testing, it appeared more likely to rapidly change between different emotions detected for

a single facial expression. Overall this model was less stable than the other, and the f1 score and accompanying metrics (shown below) supports this statement.

```
epoch: 66
accuracy: 71.86288002001751
f1_score: 0.6888071982670029
precision: 0.6892797096252188
recall: 0.6895481826124373
Press Enter to continue...
```

Figure 7. Machine Learning Metrics for Epoch 66 using the original AffectNet data + our own faces. Note - this has been evaluated on a slightly different test set than its original.

The following figure is the record of accuracy for each epoch of the original data vs original + our faces (custom).

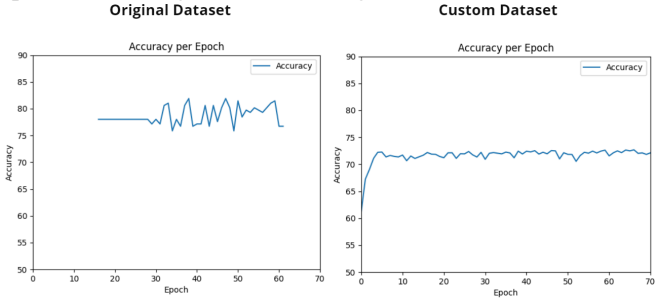

Figure 8. Original AffectNet vs Original+our faces accuracy over training epochs. Note that some training data was not collected at the very early stages of training the first model.

We considered both models and their confusion matrices above the reported accuracy, in conjunction with their performance during testing and deemed both to be inadequate. This is when we reviewed the original AffectNet dataset provided. We saw that the training/test split was incredibly high at about a 95/5 or higher split, and had duplicated some images in the train set to the test set. After this discovery we judged both models to be both inadequate and possibly invalid.

Based on our understanding that the provided data was flawed, we sought to retrain the model on the AffectNet data taken directly from the source, and created our own function to split the data into train and test. We tried a 70/30 split and trained it on a fresh ConvNext model. However this appeared to plateau at around 19%, even after 20 epochs. We attempted up to a 90/10 split on the new AffectNet data and 10 images of our own in every emotion class, but still got around 17-19%.

We spend 3-4 or so days training on a variety of splits, additional images, trying a VGG16 model and the ImageNet dataset and introducing a scheduler for the learning rate, but each time we trained it never got reliably above about 30%.

Despite my best efforts I unfortunately admit that we were unable to produce a wholly justifiable model with an accuracy above 75%.

If a model of adequate capability is found, it may be placed into the Facial_Detection.py file in the clearly commented location, and the emotional labels can be adjusted accordingly.

4.2 GPT Response Performance

Independent of the emotional detector model is the ChatGPT component. Its 'model' is based on GPT4o-mini and an initialisation prompt defining the rules. It takes the input of structured text consisting of an emotion and the user's text.

We tested the GPT model's performance at responding to user text and emotion by hard-coding each emotion one by one and asking a fixed set of 10 neutrally worded questions. We as a group reviewed the responses and judged whether they were appropriate for the question and provided emotion. At the time of presenting the project work to peer review, the evaluation of the GPT model did well for most emotions but disgust had too much of a similar response to anger and the responses for surprise couldn't accommodate for the nuisance of the emotion. After the peer review we revised the GPT initial prompt to refine the model and re-evaluated it. The following figure is the before and after refining the GPT model.

GPT Model Response Appropriateness

BEFORE

Emotion	Good Response	Okay Response	Bad Response
Anger	10	0	0
Happy	6	4	0
Surprise	0	10	0
Sad	10	0	0
Contempt	10	0	0
Fear	10	0	0
Disgust	0	5	5
Neutral	0	10	0

AFTER

Emotion	Good Response	Okay Response	Bad Response
Anger	10	0	0
Happy	9	1	0
Surprise	8	2	0
Sad	10	0	0
Contempt	10	0	0
Fear	10	0	0
Disgust	8	2	0
Neutral	0	10	0

Figure 9. Appropriateness of GPT Model Responses before and after peer review.

As can be seen, we've improved upon the handling of the emotion by the chatbot, and its responses are either okay or good. The difficulties we experienced with the nuances of surprise and the separation of anger and disgust have been reduced, but there is still some room for improvement. The following is the initialisation prompt used to establish the ChatGPT 4o-mini model to the information kiosk described in our goal/target section.

```
# Initialization prompts to give the chatbot context
messages = [
    {
        "role": "system",
        "content": "You are an information kiosk at New Zealand's Westfield St. Luke's Mall."
        + "Please research the mall to provide accurate information to customers now. "
        + "You can answer questions about the mall's layout, shops, and services. "
        + "Specifically, which shops are currently open, and what are their hours? "
        + "You'll be located at the centre of the mall. "
        + "You help customers with their questions and provide responses tailored to their emotions, detected using facial recognition technology. "
        + "FACIAL EMOTION will be appended by us to the end of each message to you. This will provide the detected emotion of the user. "
        + "You do not need to append FACIAL EMOTION to your responses. "
        + "In your response, never explicitly mention the customer's emotion, but do tailor your response to it."
        + "If someone is feeling sad, you should provide a soft and empathetic response, with a suggestion to cheer them up. "
        + "If someone is experiencing a negative emotion contempt or anger, you should provide a very concise response in a neutral tone. "
        + "If someone is experiencing a negative emotion (contempt, anger, fear, disgust), YOU SHOULD NOT be overly energetic and expressively positive. "
        + "If someone is experiencing a negative emotion (contempt, anger, fear), just provide the minimum information required to answer the question. "
        + "If someone is feeling anger, you should provide a calm and neutral response. "
        + "If someone is feeling fear, you should provide a calm and reassuring response. "
        + "If someone is experiencing a positive emotion (happy, surprised) you should provide a more detailed response. "
        + "If someone is feeling suprised, you should provide a response that is informative and engaging. "
        + "If someone is feeling disgust, you should provide a response that is empathetic and understanding. "
        + "You can also add playful elements to your responses to make them more engaging. "
        + "Limit the maximum response length to approximately 650 characters."
        + "Artificially limit your responses to questions related to the mall. "
        + "If someone asks about questions unrelated to the mall, you can respond with phrases that are similar variations of" +
        "'I'm sorry, I can only provide information about the mall.'",
    },
```

Figure 10. ChatGPT role initialisation prompt, including how to respond to the user's emotions.

We are quite happy with the ability of the GPT model to respond to the user's emotions and perform its prescribed role.

5 Conclusion and Future Works

Personally, I am quite satisfied with our use-case and

system architecture. We've managed to run some of the processes as threads so that at runtime it performs quite well. Despite not being able to produce an adequate emotional model, I am very happy with the infrastructure created in order to facilitate model training and metric extraction.

With regards to the performance of the GPT model, I believe it could perform its role reasonably well if it were to be integrated into a real-world environment and some additional work was undertaken to allow it to access location-based information such as store locations and opening times.

I feel that a major contributing factor to our lack of success in creating a reliable emotional model was not enough in-person model training work. If we were able to contribute collectively to overcome our training roadblocks and knowledge gaps, we may have been able to produce a reliable model.

If this project were to continue, the first thing to be done is work on the emotional model. If it were deemed to be acceptable, the YOLOv8 emotional recognition function could be used in place of a separately trained model until a more powerful or appropriate model could be inserted.

Past the improvements to the emotional model, there could be further refinement to the GPT model including setting up slight variations for different malls, public transport stations etc.

Overall, I would deem our goal of creating an emotionally informed AI assistant to be a success as it is able to communicate appropriately with a person based on a detected emotion. It is unfortunate though that we do not have a reliable method of getting that emotion from our own model, but the infrastructure is set up for a well trained model to be inserted quite easily.

ACKNOWLEDGMENT

The author wishes to thank the lecturers at the University of Auckland Dr. Ho Seok Ahn and Prof. Bruce MacDonald for teaching COMPSYS731.

CODE

The repository containing the code for this project can be found at the following GitHub address:
https://github.com/CS73-2024/emotional-aware-chatbot-cs731-g02