# Creating Ollie the Happiness Agent – Designing for Happiness in Education

L. van Breda
0995866
l.v.breda@student.tue.nl

T. T. van Bussel
0990679
t.t.v.bussel@student.tue.nl

C. Liu
1508849
c.liu2@student.tue.nl

A. van Luijk
0956693
a.v.luijk@student.tue.nl

## Abstract

Mental health remains a challenging area in education, partly due to stress. Smiling more may improve the mood of students in an educational context. This project is an exploration of how an agent may play a role in improving students' mental health by motivating them to smile more. Using an emotion detection model and explainable artificial intelligence, an interface was created that can mirror the user's facial expressions. The result is a tangible agent that can be placed at the user's desk and can provide the user with insight into their current facial expression. Finally, while research with regards to the user experience and further development would be needed in order to create an impact, this project gives insights concerning how artificial intelligence may play a role in improving the student's happiness.

## Keywords

Embodying intelligence; happiness; agent; students; education; smiling; emotion recognition; facial expression

## 1. Introduction

### 1.1 The problem

In education, a challenging area is the mental health of students. School and education-related issues are reported as being an important contributor to stress (Smith & Brooks, 2015). Additionally, a study by Evans et al. (2018) shows that graduate students are more than six times as likely to experience depression and anxiety as compared to the general population. Besides that, 39% of the graduates scored in the moderate to severe depression range, compared to 6% of the general population that was measured previously (Evans et al., 2018). This means there may be a need for creating awareness about the students' mental health in order to start improving it.

There are some indications that facial expressions may have an impact on the overall happiness of humans. An example is the James-Lange theory of emotion, which proposes that bodily and physiological changes cause certain emotions (Coleman & Snarey 2011). In other words: *"We don't laugh because we're happy – we're happy because we laugh."* (James, 1950). In a study by Kleinke et al. (1998), it was found that participants experienced a more positive mood when they had more positive facial expressions. It was also found that when they had negative facial expressions, their positive mood decreased.

For this reason, we decided to focus on increasing students' happiness in an educational context. We developed a tangible agent which can mirror the students' current facial expression, in order to stimulate them to smile. To achieve this, we used a machine learning algorithm for recognizing facial expressions. The aim is to provide students with insights into what their current facial expression or emotion is while at the same time stimulating them to smile every now and then and therefore increase their happiness.

### 1.2 Related work

#### *Types of research and appliances*

Addressing mental health in an educational context in itself is not a new concept. Previous research on the mental health of students has shown the potential benefits of actively addressing the students' well-being, whether that is through mindfulness (Bernay et al., 2016) or through a digital appliance (Tsujita & Rekimoto, 2011). One study takes a step into self-monitoring the emotional well-being by developing a mobile phone application (Rickard et al., 2016).

Including technology to increase awareness and stimulating mental health has also been researched by Miner, Milstein, and Hancock (2017). They discussed the conversational agent *"Gabby"*, a software program that uses conversational artificial intelligence to interact with users in order to help patients with chronic pain and depression.

#### *Role technology and artificial intelligence can play*

Artificial Intelligence is playing an increasingly important role in mental health care, Luxton (2016) notes that this presents many opportunities and benefits. A similar note was made by Lovejoy (2018) in stating that *"successful integration of AI into healthcare could dramatically improve quality of care"*. An example of this is a study that implemented a stress-detection system using artificial intelligence, with the aim to improve a consistent diagnosis and decisions (Madhuri, Mohan, Kaavya, 2013).

Asides from the integration of artificial intelligence in mental health care in general, it may also specifically play a role in the interaction with the user and stimulating them to smile more. A study done on the effects of a virtual agent's smile, found that, when the agent was smiling, the participants themselves also smiled longer (Krämer et al., 2012).

#### *Current experience with digital appliances*

An example of the implementation of artificial intelligence for improving a person's mental health can be seen in a study that created the "HappinessCounter" (Tsujita & Rekimoto, 2011). In this study, a system named the HappinessCounter was created with the aim to improve the mood of users who live alone. The HappinessCounter used visual smile recognition and provided the users with feedback by showing a sad or smiling icon in the corner of a mirror. They found that this system indeed had a positive effect on the user's mood, as well as on their family's mood.

Tsujita & Rekimoto (2011) provide very interesting insights into how such a system would be used which may help in creating a system that fits within the educational context. What could be of added value is to provide users with more accurate feedback on their facial expressions rather than just ‘happy’ or ‘sad’. This may provide the user with more insight into what their facial expression is as well as it gives users insight into what the system actually measures.

## 2. Methods and materials

Before diving into the design of the agent, we elaborate on the theories and learning algorithms used in the design.

### 2.1 The approach

In developing an agent that increases students’ happiness in an educational context, we decided to approach this from the James-Lange theory of emotion (Coleman & Snarey 2011). This theory proposes that physiological changes can cause certain emotions. In other words: smiling can make you happy. This theory was applied here to increase students' happiness by challenging them to physically smile, which in turn should lift their mood.

In order to develop an agent worth smiling with, some factors need to be taken into consideration. Firstly, the agent must have some sort of interactivity responding to the facial expression of the user. In this project, this is achieved by using a webcam and pre-trained learning algorithms for facial detection and emotion classification.

Secondly, it needs to elicit a smile from the user. To achieve this, we decided to give the agent cute features. The expressions displayed by the agent are loosely based on the Personality characteristics described in Govers’ thesis (2004). One can compare the eliciting of a smile by the agent to how one would smile at a sad baby to cheer them up.

The agent was made to be tangible and sit on a user’s desk, next to their computer setup. This choice was made to provide the user with a distraction from their computer screen, and because in our opinion having a soft and physical agent leads to more empathy for it than a digital representation.

### 2.2 Learning algorithm

In the present project, two learning algorithms were used in order to assess the current facial expression of the user. As mentioned in the previous section, both of these algorithms were pre-trained by others and applied in this project, in order to avoid reinventing the wheel.

The first learning algorithm is used by the agent for facial detection. The second one is an emotion detection algorithm that uses the detected face from the first one to assess the facial expression of the user.

For facial recognition, we used a Haar Cascade algorithm. Haar feature-based cascade classifiers are excellent for object detection and were first proposed by Paul Viola and Michael Jones (2001). The first step in this algorithm is to collect the Haar features of an image (in our case containing a face). A Haar feature sums up pixel intensities in an image by considering adjacent rectangular regions and calculates the difference between the sums. Examples of Haar features are shown in figure 1.

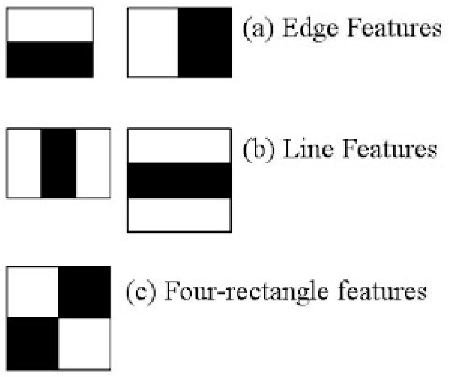

*Figure 1: Haar features. Retrieved from https://docs.opencv.org/*

Using these Haar features, a cascade classifier detects if they contain a face or not. Cascade classifiers are excellent at rejecting negative samples (not containing a face) as fast as possible. This is important since even a 24x24 pixel sample results in 160.000 features. The Haar Cascade algorithm for detecting faces is built into the OpenCV (2020) library used in this project.

Once a face is recognized, a 48x48 pixel grayscale image of the face is sent into the emotion detection algorithm. This learning algorithm was developed and written up by Karan Sethi (2020). This learning algorithm makes use of a Convolutional Neural Network (CNN), a neural network most commonly applied to analyzing visuals. CNNs are multi-layer perceptrons, meaning that all neurons in one layer are connected to all neurons in the next. This specific CNN consists of 7 layers, which are described in-depth in the article by Sethi (2020). From the 48x48 pixel input, the network outputs the detected facial expression of the user: angry, happy, neutral, sad, surprised. The CNN was trained using the Facial Emotion Recognition 2013 (FER-2013) dataset, containing 35.685 examples of grayscale images of faces categorized into one of seven emotions. The FER-2013 dataset was created by Pierre Luc Carrier and Aaron Courville for the ICML 2013 workshop on facial expression recognition (Goodfellow et al., 2013).

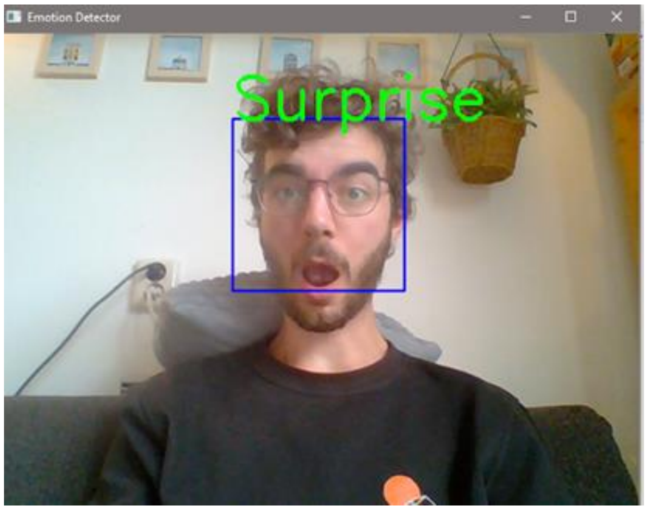

*Figure 2: Emotion detection using the CNN*

Using the detected facial expression, the agent can decide on what to do. This is described in the next section.

## 3. Design

## 3.1 Design of the interaction

To create more awareness about the mental health of students and to increase happiness in an educational context, we designed an agent called Ollie. Ollie, named after the screen used to express his facial expression, is a tangible object that can be placed next to the user's computer. The main purpose of Ollie is to remind the user to smile every once in a while. Next to that, by mirroring the user's facial expression, Ollie can help gain better insight in one's facial expressions and emotions. As mentioned, Kleinke et al. (1998), found that participants experienced a more positive mood when they had more positive facial expressions. It is expected that when the user receives a reminder to smile, he or she might experience a more positive mood.

### *The prototype*

A tangible prototype was created, which can be seen in figure 3. The prototype is supposed to be placed in the user's working space in direct eyesight. The prototype has a face that consists of three OLED screens, one for each eye and one for the mouth. By making different facial expressions, Ollie can express different emotions. A camera is implemented on top of Ollie to be able to capture the user's face and detect the user's facial expression. The core of the prototype is formed by a Raspberry Pi model 3B.

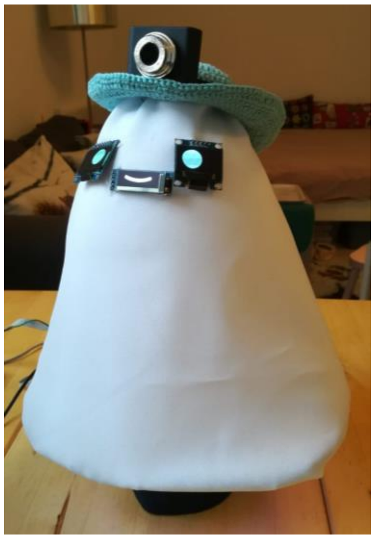

*Figure 3: The tangible prototype*

The two main requirements for the looks of Ollie were that it should be cute and an abstract shape. It is expected that users would feel more likely to smile at Ollie to cheer it up, if it was looking cute. To achieve this, Product Personality was taken into account. Following the theory from Govers' thesis (2004), the shape of the prototype and facial expressions are generally round and soft. It was decided to go for an abstract shape to emphasize the roundness and softness and to make Ollie blend into the study environment.

### *Facial expressions*

The facial expressions form an important part of the prototype, since Ollie's face forms the main form of communication with the user. Through the facial expressions, Ollie can express emotions. This is applicable when the user has not smiled for a certain amount of time. At that moment, Ollie starts mirroring the user to draw the user's attention.

It was decided that Ollie would blink every few seconds, to give it a more life-like feeling and to give an indication that Ollie is still functioning. Figure 4 shows the different facial expressions that Ollie can express.

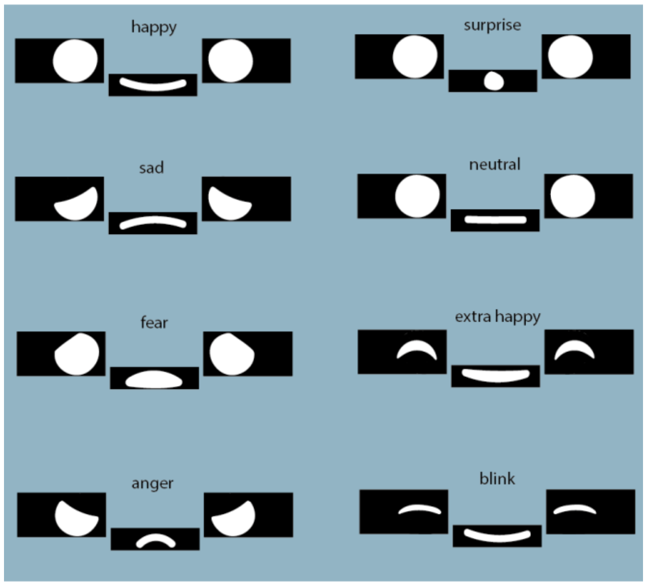

*Figure 4: The different facial expressions and emotions that Ollie can express*

### *Social battery*

To avoid that the user needs to interact with Ollie all the time, a "social battery" was implemented, which can be seen in figure 5. When the social battery is full, LED lights at the bottom of Ollie light up and Ollie smiles. Over time, the social battery slowly drains, this is indicated by fading of the LED lights. When the social battery is drained, the LED lights are completely faded and Ollie starts mirroring the user's facial expression in order to grab the user's attention. The user can charge the battery by smiling at Ollie for approximately ten seconds. Ollie indicates that the battery is fully charged by showing an 'extra happy' facial expression (figure 4).

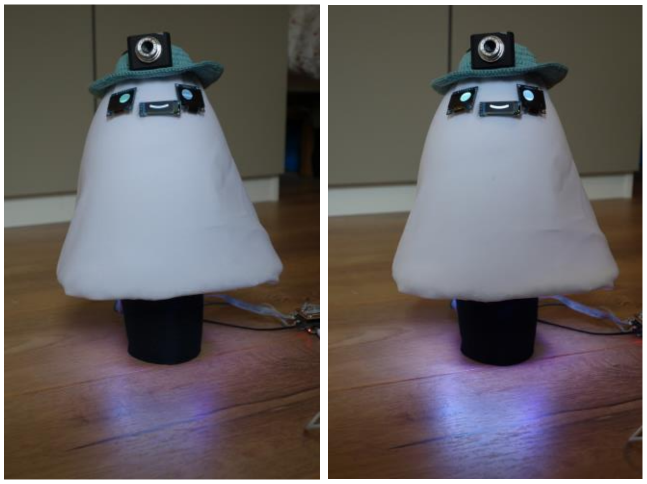

*Figure 5: Social battery of Ollie*

### *Explainable AI*

The main function of Ollie is to recognize the user's emotion, to be able to determine whether or not the user is smiling. In order to know if Ollie is functioning correctly, it has to communicate what it detects. By mirroring the user's emotion on the screens, Ollie directly shows what emotion it has detected. This way, there can be no confusion on what Ollie is currently observing.

### *Interaction scenario*

The interaction between the user and Ollie can be seen in figure 6. Starting from the top left of the diagram, the social battery is full. Ollie looks extremely happy to indicate this. After a few seconds, Ollie's emotion changes to happy. The social battery slowly drains over time. While this happens, there is no interaction between Ollie and the user. Ollie looks happy and the user can do his/her own thing. When the social battery is drained, Ollie starts mirroring the user's emotion. For example, when the user looks angry, Ollie looks angry. This way, Ollie is asking the user for attention. When the user looks happy at Ollie, Ollie looks happy and the battery recharges. This is indicated by the LEDs that slowly light up. When the user has smiled at Ollie for ten seconds consecutively, the battery is fully recharged. Ollie looks extremely happy for a few seconds and the cycle starts over again.

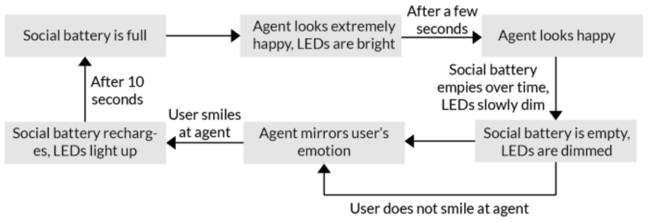
Social battery is full
Agent looks extremely happy, LEDs are bright
After a few seconds
Agent looks happy
Social battery empies over time, LEDs slowly dim
Social battery is empty, LEDs are dimmed
Agent mirrors user's emotion
User does not smile at agent
User smiles at agent
Social battery recharg-es, LEDs light up
After 10 seconds

*Figure 6: The interaction between the user and Ollie*

## 3.2 Intelligent behavior and embodiment

### *Sensory information*

To be able to determine the user's emotion, the user's facial expression is captured through a webcam. In the current prototype the webcam is placed on top of Ollie and pointed towards the user's face. When Ollie starts mirroring the user, the webcam continuously collects data about the user's facial expression, providing a constant input of data for the algorithm. The image from the webcam is first processed by the previously described Haar Cascade algorithm for facial detection. Next the detected face is converted to a grayscale image of 48x48 pixels and fed into the Convolutional Neural Network for emotion detection.

### *Learning algorithm*

The main goal of Ollie was to determine the user's facial expression. To achieve this, two pre-trained learning algorithms were used. The Haar Cascade algorithm by Paul Viola and Michael Jones (2001) was used for facial recognition. An algorithm by Karan Sethi (2020) is used for emotion detection. These algorithms are described in more depth in section 2.2.

### *Results*

The outcome of the CNN algorithm does not provide us with any prediction certainty, but merely outputs one of the 6 possible emotions, or "None" if no face was detected by the Haar Cascade. For this application, this does not matter that much. We evaluated the performance of the algorithm to be sufficient, and the results of the emotion detection are directly displayed using the face of Ollie. Thus the user can evaluate whether Ollie detected their expression correctly, and adjust their smile in case it did not.

## 3.3 Testing and analysis

The functionality of the prototype was tested by one of the team members. The prototype was placed next to her computer for a day. This test was performed to test the functionality during a full day. During this test, a few things were noticed:

- At the start, the blinking was a bit distracting for her, because something is moving. This got a bit less distracting after some time, but it was still noticeable
- Sometimes she did not notice that the battery was drained
- When smiling she sometimes had a tendency to look at the camera instead of at the face
- It reminded her a little bit of when she had a pet in her student room (also partly because it blinked and distracted her a bit because her pet also would move around before going back to sleep)
- She saw the mirroring a bit as the time to take a break as it was every hour
- In the beginning, the lights didn't really give her an indication of the battery level. but towards the end of the day she did get a better idea of whether it was almost drained or not

These results confirmed some of our expectations and also showed some points of improvement for the project. Further discussion of these results can be found in section 5.

## 4. Conclusions

The goal of the project was to create an implementation of explainable artificial intelligence in an educational context. We decided to design for mental health in education by creating an agent, named Ollie, which makes the user smile more during study sessions. Ollie is a tangible buddy that can be placed next to the user's computer. It has a soft, abstract and cute look to encourage empathetic behavior from the user towards Ollie. Intelligent behavior is implemented through the recognition of the user's emotion, for which the Haar Cascade algorithm (2001) and the algorithm by Karan Sethi (2020) were used. Ollie's actions are made explainable by showing the detected emotion on Ollie's face as it mirrors the user's facial expression. The test that was performed showed insights into the functionality of Ollie, which resulted in some points of improvement. Future steps should include a user test since this could provide more insights into the effect of Ollie on happiness in an educational context.

This project formed an exploration of using artificial intelligence and machine learning to encourage people to smile more. By mirroring the user's facial expression, the user could gain more insight into their emotion, providing them with the opportunity to reflect on their current emotion. The combination of these two effects could form a promising design opportunity for XAI in an educational context. While future research is needed, his project shows that explainable artificial intelligence and machine learning could form a solution for more happiness in education.

## 5. Discussion

In this project, we designed a happiness agent that aims to improve the moods of students in an educational environment. While it could motivate users to smile more frequently, there are still challenges that remain.

### *User testing*

The effect of how much the agent could enhance the happiness of people in the educational context requires more rigorous experimentation. Due to the time limitation, after building the first prototype of the agent we could only conduct a one-day test with a group member. We were concerned about the bias of having team members as the participants. However, regarding keeping safety within the COVID-19 situation, we decided to make this decision. This issue could be improved in a future study by experimenting in relevant contexts, such as the working environment in the school or at home. In a future study, it may be needed to test with a diverse sample of participants who are in education. We should also consider conducting a qualitative interview afterward, in order to gain more in-depth insights into the user experience.

### *Design of the agent*

Even though a theoretical background was used while designing the looks of Ollie, we suggest that the outlook of the agent, as well as the interaction of facial expression, and the performance of the social battery could be tested with more different possibilities. For instance, the researcher could test out different types of shapes for the eyes or the time-interval that the eyes blink, in order to find out the best combination for the users.

### *Machine learning*

In future research, an option would be to train a more precise algorithm that caters more personally to the user's needs. We would also advise optimizing the software since the software used for this project requires quite a lot of processing power. Therefore, the reaction of the agent running on Raspberry Pi is slower than on a PC. We did not explore if optimizations could help out increasing recognition speed, or if the Raspberry Pi is just not powerful enough. In addition, a possible better webcam resolution could help out with more accurate emotion recognition, as we noticed during testing that higher-quality cameras lead to better predictions.

### *Future steps in designing Ollie*

At this stage, we do not know yet how users perceive Ollie and if they fully understand how the social battery works. One of the main future steps includes testing, which would also allow us to gain more insight into if, and how, the used algorithms and electronics would need to be adjusted.

In a one-day test with a group member, it was found that the blinking was seen as a light distraction, which could be something to improve on as it might hinder productivity. On the other hand, the student's experience did confirm our expectation that the blinking would give the agent a more life-like feeling. As one goal of the look of the agent is empathy from the user for the agent, the life-like feeling can play an important part. A future iteration could include how the blinking feature can be implemented without it being distracting for the user, while at the same time preserving the life-like feeling. This could be done by, for example, making the time in between blinking personal for each user. By ignoring Ollie, it may learn what time is best to start asking for attention so as not to disrupt the workflow of the student too much.

Further iterations in the process may also include reflecting on how the facial expressions are perceived by the users. While the current design is based on the literature by Govers (2004), reflection on the perceived personality is at the moment not implemented, but this too could be helpful in remaining the feeling of Ollie gaining some sense of personhood and having a personality.

## 6. References

[1] Bernay, R., Graham, E., Devcich, D. A., Rix, G., & Rubie-Davies, C. M. (2016). Pause, breathe, smile: a mixed-methods study of student well-being following participation in an eight-week, locally developed mindfulness program in three New Zealand schools. *Advances in School Mental Health Promotion*, 9(2), 90–106. https://doi.org/10.1080/1754730X.2016.1154474

[2] Evans, T., Bira, L., Gastelum, J., Weiss, L., & Vanderford, N. (2018). Evidence for a mental health crisis in graduate education. *Nature Biotechnology*, 36(3), 282-284. doi: 10.1038/nbt.4089

[3] Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., ... & Zhou, Y. (2013, November). Challenges in representation learning: A report on three machine learning contests. In *International conference on neural information processing* (pp. 117-124). Springer, Berlin, Heidelberg.

[4] James, W. *The Principles of Psychology* (Vol. 2). Dover Publications, New York, USA, (1950).

[5] Kleinke, C.L., Peterson, T.R., and Rutledge, T.R. Effects of self-generated facial expressions on mood. Journal of Personality and Social Psychology, 74 (1998), 272-279.

[6] Krämer, N., Kopp, S., Becker-Asano, C., & Sommer, N. (2013). Smile and the world will smile with you - The effects of a virtual agent's smile on users' evaluation and behavior. *International Journal of Human Computer Studies*, 71(3), 335–349. https://doi.org/10.1016/j.ijhcs.2012.09.006

[7] Lovejoy, C. A., Buch, V., & Maruthappu, M. (2019). Technology and mental health: The role of artificial intelligence. *European Psychiatry*, 55, 1–3. https://doi.org/10.1016/j.eurpsy.2018.08.004

[8] Luxton, D. D. (2016). An Introduction to Artificial Intelligence in Behavioral and Mental Health Care. In *Artificial Intelligence in Behavioral and Mental Health Care* (pp. 1–26). https://doi.org/10.1016/B978-0-12-420248-1.00001-5

[9] Madhuri, V. J., Mohan, M. R., & Kaavya, R. (2013). Stress management using artificial intelligence. *Proceedings - 2013 3rd International Conference on Advances in Computing and Communications, ICACC 2013*, 54–57. https://doi.org/10.1109/ICACC.2013.97

[10] Miner, A. S., Milstein, A., & Hancock, J. T. (2017). Talking to machines about personal mental health problems. *JAMA - Journal of the American Medical Association*, 318(13), 1217–1218. https://doi.org/10.1001/jama.2017.14151

[11] OpenCV. (2020). Retrieved October, 2020, from https://opencv.org/

[12] Rickard, N., Arjmand, H.-A., Bakker, D., & Seabrook, E. (2016). Development of a Mobile Phone App to Support Self-Monitoring of Emotional Well-Being: A Mental Health Digital Innovation. *JMIR Mental Health*, 3(4), e49. https://doi.org/10.2196/mental.6202

[13] Sethi, K. (2020, June 24). Emotion Detection Using OpenCV and Keras. Retrieved October 26, 2020, from https://medium.com/swlh/emotion-detection-using-opencv-and-keras-771260bbd7f7

[14] Smith, E. & Brooks, Z. Graduate Student Mental Health (University of Arizona, 2015).

[15] Snarey, J., & Coleman, A. E. (2011). Encyclopedia of Child Behavior and Development. In *Encyclopedia of Child Behavior and Development* (pp. 844–845). https://doi.org/10.1007/978-0-387-79061-9

[16] Tsujita, H., & Rekimoto, J. (2011). HappinessCounter: Enhancing Positive Mood and Communication with Smile-Encouraging Digital Appliances. *Computer Software*, 29(4), 305–315. https://doi.org/10.11309/jssst.29.4_305

[17] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. doi:10.1109/cvpr.2001.990517

## 7. Appendix

In the following pages, the embodied software for the agent can be found.

```python
# --- Filename: Main_Program.py
# --- Required files: Emotion_Detect.py, Variables.py

from collections import Counter
import Variables
import time
from time import sleep
from random import seed
from random import randint

import sched
import Adafruit_GPIO.SPI as SPI
import Adafruit_SSD1306
import board
import neopixel

from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
from PIL import ImageOps

# ---- Initialize NeoPixel
pixel_pin = board.D21
num_pixels = 30
ORDER = neopixel.GRB
pixels = neopixel.NeoPixel(
    pixel_pin, num_pixels, brightness=0.2, auto_write=False, pixel_order=ORDER
)
pixels.fill((255, 0, 0))
pixels.show()

# ---- Initialize Screens
RST = 19
DC = 26
SPI_PORT = 0
SPI_DEVICE = 0
font = ImageFont.load_default()

rightEye = Adafruit_SSD1306.SSD1306_128_64(rst=RST, i2c_bus=4)
leftEye = Adafruit_SSD1306.SSD1306_128_64(rst=RST, i2c_bus=1)
mouth = Adafruit_SSD1306.SSD1306_128_32(rst=RST, i2c_bus=3)

mouth.begin()
mouth.clear()
mouthImage = Image.open('images/happyMouth.png').convert('1')
mouth.image(mouthImage)
mouth.display()

rightEye.begin()
rightEye.clear()
rightEye.display()

leftEye.begin()
leftEye.clear()
leftEye.display()

# ---- Import Emotion Detection
from Emotion_Detect import CurrentEmotion
print("Ready")

# ---- Setup social battery and past emotions
blinkInterval = 6
capacity = 60
chargingTime = 5
memory = 5
battery = Variables.socialBattery(capacity, chargingTime)
pastEmotions = Variables.emotions(memory)

# ---- Functions
def batteryEmpty():
    chargeTime = battery.chargingTime

    if pastEmotions.prevalent() == 'Happy': # If the prevalent emotion is happy
        sinceLastChange = round(time.time()) - pastEmotions.timeLastChange

        if sinceLastChange <= chargeTime: # And the time since last change is smaller than the time needed to recharge the battery
            battery.recharging = 1
            print("Time to recharge: ", chargeTime - sinceLastChange)

            brightness = reMap(sinceLastChange, chargeTime, 0, 100, 0)
            ledRing(brightness)

            displayEmotion('Happy')

        else: # If it is larger then we are done charging
            print("Recharged!!")
            battery.reset()
            displayEmotion('ExtraHappy')
            sleep(3)
    else:
        ledRing(0)
        battery.recharging = 0
        emotionToDisplay = pastEmotions.prevalent()
        displayEmotion(emotionToDisplay)
        print("Started mirroring: ", emotionToDisplay)

def blink():
    displayEmotion('Blink')
    sleep(1)

def displayEmotion(emotion):
    #eyeImage = Image.open('Eye.png').resize((rightEye.width, rightEye.height), Image.ANTIALIAS).convert('1')

    imageFolder = 'images/'

    if emotion == 'Happy':
        eyeImage = 'happyEye.png'
        mouthImage = 'happyMouth.png'

    elif emotion == 'Sad':
        eyeImage = 'sadEye.png'
        mouthImage = 'sadMouth.png'

    elif emotion == 'Fear':
        eyeImage = 'fearEye.png'
        mouthImage = 'fearMouth.png'

    elif emotion == 'Angry':
        eyeImage = 'angerEye.png'
        mouthImage = 'angerMouth.png'

    elif emotion == 'Surprise':
        eyeImage = 'surpriseEye.png'
        mouthImage = 'surpriseMouth.png'

    elif emotion == 'ExtraHappy':
        eyeImage = 'extraHappyEye.png'
        mouthImage = 'extraHappyMouth.png'

    elif emotion == 'Blink':
        eyeImage = 'blinkEye.png'
        mouthImage = 'blinkMouth.png'

    elif emotion == 'Neutral':
        eyeImage = 'neutralEye.png'
        mouthImage = 'neutralMouth.png'

    elif emotion == None:
        eyeImage = 'noneEye.png'
        mouthImage = 'noneMouth.png'

    else:
        eyeImage = 'neutralEye.png'
        mouthImage = 'neutralMouth.png'
        print("Called displayEmotion with unknown: ", emotion)


    #eyeImage = Image.open(imageFolder + eyeImage).resize((rightEye.width, rightEye.height), Image.ANTIALIAS).convert('1')
    #mouthImage = Image.open(imageFolder + mouthImage).resize((mouth.width, mouth.height), Image.ANTIALIAS).convert('1')

    leftEyeImage = Image.open(imageFolder + eyeImage).convert('1')
    rightEyeImage = ImageOps.mirror(leftEyeImage)

    mouthImage = Image.open(imageFolder + mouthImage).convert('1')

    leftEye.image(leftEyeImage)
    rightEye.image(rightEyeImage)
    mouth.image(mouthImage)

    leftEye.display()
    rightEye.display()
    mouth.display()

def ledRing(brightness):

    brightness = int(reMap(brightness, 100, 0, 255, 10))

    pixels.fill((brightness, brightness, brightness))
    pixels.show()

def reMap(value, maxInput, minInput, maxOutput, minOutput):

    value = maxInput if value > maxInput else value
    value = minInput if value < minInput else value

    inputSpan = maxInput - minInput
    outputSpan = maxOutput - minOutput

    scaledThrust = float(value - minInput) / float(inputSpan)

    return minOutput + (scaledThrust * outputSpan)


while True:
    label = CurrentEmotion()
    pastEmotions.append(label)

    #ledRing(255)

    if not battery.currentLevel() > 0:
        batteryEmpty()
    else:
        blinkValue = randint(0,20)

        if blinkValue == 1:
            displayEmotion('Blink')
        else:
            displayEmotion('Happy')

        print("Battery level: ", battery.currentLevel())
        ledRing(battery.currentLevel())
```

```python
# --- Filename: Variables.py

from collections import deque
import time

startTime = time.time()
pastEmotions = deque(["Hello"])

def TimeElapsed():
    elapsedTime = round(time.time() - startTime)
    return elapsedTime

# Classes
class socialBattery:

    # Variables you can set:
    #   capacity: The capacity of the battery in seconds (how long it takes to run out
    #   chargingTime: how long it takes to recharge the battery

    def __init__(self, capacity, chargingTime, initialized=0, level=0, recharging = 0):
        self.capacity = capacity
        self.initialized = round(time.time())
        self.chargingTime = chargingTime
        self.level = level
        self.recharging = 0

    # Function to calculate the battery level in percentage
    def currentLevel(self):
        elapsed = round(time.time()) - self.initialized
        end = self.capacity
        level = elapsed/end
        level = 100-round(level*100)

        if not level >= 0:
            level = 0

        self.level = level
        return level

    # Function to reset the battery when it was recharged
    def reset(self):
        self.initialized = round(time.time())
        self.recharging = 0

class emotions:

    # Variables you can set:
    #   maxlen: how far back the 'emotion memory' goes. If you increase this, it decreases noise in the output but increases response time

    def __init__(self, maxlen, timeLastChange=0, prevalentEmotion=0):
        self.list = [range(maxlen)]
        self.maxlen = maxlen
        self.timeLastChange = round(time.time())
        self.prevalentEmotion = prevalentEmotion

    # Function to add the most recently detected emotion to the list and remove the oldest one
    def append(self, item):

        if len(self.list) < self.maxlen:
            self.list.append(item)
        else:
            self.list.append(item)
            self.list.pop(0)

        previous = self.prevalentEmotion
        if not previous == self.prevalent():
            self.timeLastChange = round(time.time())


    # Function that returns the most prevalent emotion over the past 'maxlen'
    detections
    def prevalent(self):
        self.prevalentEmotion = max(set(self.list), key = self.list.count)
        return self.prevalentEmotion

```

```python
# --- Filename: Emotion_Detect.py
# --- Adapted from Karan Sethi https://github.com/karansjc1/emotion-detection
# --- Required files: EmotionDetectionModel.h5, haarcascade_frontalface_default.xml

from keras.models import load_model
from keras.preprocessing.image import img_to_array
from keras.preprocessing import image
import cv2
import numpy as np
import csv
import Variables
from collections import deque

face_classifier=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
classifier = load_model('EmotionDetectionModel.h5')

class_labels=['Angry','Happy','Neutral','Sad','Surprise']

cap=cv2.VideoCapture(1)

def CurrentEmotion():
    ret,frame=cap.read()
    labels=[]
    gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces=face_classifier.detectMultiScale(gray,1.3,5)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray=gray[y:y+h,x:x+w]
        roi_gray=cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)

        if np.sum([roi_gray])!=0:
            roi=roi_gray.astype('float')/255.0
            roi=img_to_array(roi)
            roi=np.expand_dims(roi,axis=0)

            preds=classifier.predict(roi)[0]
            label=class_labels[preds.argmax()]

            return label

        else:

            return("no face found")
```