# Emotion Analysis in a Pepper Robot

Sebastian Bångerius[1], Simon Tsehaie Russom[2], Simon Wijk Stranius[3], Oscar Linnarsson[4], Philip Ngo[5], Johannes Hägerlind[6], Sami S. Yahya[7]

**Abstract**
In a society with increased robotic presence, the project set out to make a humanoid Pepper robot from Softbank Robotics/Aldebaran be able to recognize people and their emotional state from face detection as well as text and tone analysis of speech. This possibly assists the Human-Robot Interaction (HRI) field of research by helping humans communicate with a robot and also making future research on emotional robots possible. The project team accomplished face analysis and recognition as well as text analysis of speech. Tone analysis was implemented but revoked from the final solution because of stability issues. It was discussed and left as future work to build upon this project and the field of HRI research in general.
**Source code**: `https://gitlab.liu.se/tdde19-2021-5/tdde-19-group-5-pepper`

**Video**: `https://youtu.be/LMxPp7W5oS0?t=321`

**Authors**
[1] *Information Technology Student at Linköping University, sebba262@student.liu.se*
[2] *Computer Science Student at Linköping University, simru247@student.liu.se*
[3] *Computer Science Student at Linköping University, simst932@student.liu.se*
[4] *Information Technology Student at Linköping University, oscli329@student.liu.se*
[5] *Media Technology Student at Linköping University, phing272@student.liu.se*
[6] *Computer Science Student at Linköping University, johha451@student.liu.se*
[7] *Computer Science Student at Linköping University, samya461@student.liu.se*

**Keywords**: Emotion Detection — NLP — Computer Vision — Pepper Robot — Robot Operating System ROS

## Contents

# 1. Introduction

In today's society there are various tasks handled by robots, yet many of those robots are physically limited to their respective tasks.

New research at Linköping University has explored whether emotional state affects how humans perceive robots [31]. Thellman, Thunberg, and Ziemke [31] acknowledged that emotions play a significant role in communication in social interaction, this project implements emotion analysis on a physical robot. Thellman and Ziemke [32] further investigated the difficulty in communication between humans and robots because of the perception issue - people having problems knowing what the robot perceives about the world and its environment. As a further response this project aims to provide a base for future work by the emotion analysis implementation in order to assist in the Human-Robot Interaction (HRI) field of research.

In a wider context, robots are employed in a multitude of areas [17, 26, 11] where emotion analysis may be beneficial to facilitate HRI; for instance in the field of customer satisfaction where emotion may be contagious [27], and also key to the robot's long-term survival in a world of humans [4].

With this background, it is interesting to implement the emotion analysis on a robot in the real world. In this project, we implemented and evaluated emotion analysis on a humanoid robot, the *Pepper* model, by Softbank Robotics [1].

## 1.1 Goal

The primary and secondary goals of the project are as follows:

1. Pepper should detect a person in its main camera frame, start talking to them, listen to their response(s) and determine something about their emotions.

2. Pepper should **find** a **new** person, start talking to them, listen to their response(s) and determine something about their emotions.

# 2. Theory

In this chapter, the underlying background, definitions, and related works of the different aspects of the project are presented.

## 2.1 Robot Operating System

The Robot Operating System (ROS) is used extensively in industrial robotics applications to facilitate development of these systems [13] by acting as middle-ware through message passing between processes on a network. Programs in ROS are fundamentally denoted as *nodes* and communicate on *topics* through *messages*, *services*, and/or *actions*.

[1] https://www.softbankrobotics.com/emea/en/pepper

## 2.2 Emotions

Getting a computer to read human thought might be one of the most challenging tasks [22]. An easier task is to classify the emotions of a human. Researchers use different models for representing emotions. Some use continuous dimensions such as valance and intensity, whereas others use discrete labels such as *happy, sad, surprised* [22].

In a study from Ekman [8], researchers that studied emotion quantitatively were asked 'which emotion labels (out of a list of 18) should be considered to have been empirically established' [8, p. 32]. The result is shown in Table 1.

| emotion | percent that agrees |
|---|---|
| anger | 91 % |
| fear | 90 % |
| disgust | 86 % |
| sadness | 80 % |
| happiness | 76 % |
| shame, surprise, embarrassment | 40-50 % |
| guilt, contempt, love, awe | 30-40 % |
| pain, envy | 28 % |
| compassion | 20 % |
| pride | 9 % |
| gratitude | 6 % |

**Table 1.** Emotions and the percentages of relevant researchers that agree

Emotion can for example be communicated via our voice, facial expressions, and gestures [22]. It could also be inferred from body temperature changes, heart rate changes, hormone levels, etc. Emotion can also be derived from the spoken or written word and might be facilitated by an understanding of the context [22]. An example of context might be that a person's favorite football team just lost, which might increase the probability of sadness or anger in comparison with happiness.

In this project, the authors focus on the voice, the spoken word, and facial expressions to determine emotion. There are limitations to just using these 'channels' of emotional communication since adults have learned to regulate their emotions and can either display their emotion spontaneously or in a controlled manner [22].

## 2.3 Natural Language Processing

Natural Language Processing, or NLP for short, is a branch of Artificial Intelligence that deals with allowing computers to understand human languages [7]. This is done by combining pre-existing linguistic rules within the language, and analysis done by machine and deep learning models. The first-ever documented study and test done on NLP was in the 1950s by Alan Turing, a test called the Turing test [5]. This test still holds on until this day as a criterion of intelligence for machines as a comparison to that of humans. See Figure 1 for a view of the different NLP branches.

Different languages have different levels of complexity in both the written and the spoken versions. In this project, we

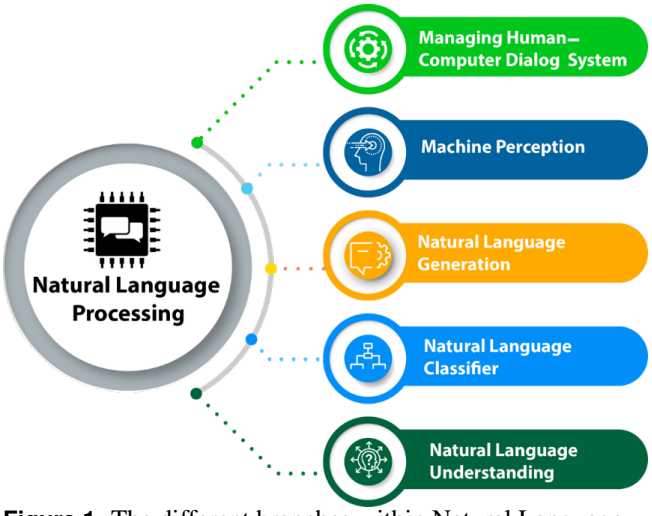



**Figure 1.** The different branches within Natural Language Processing [35].

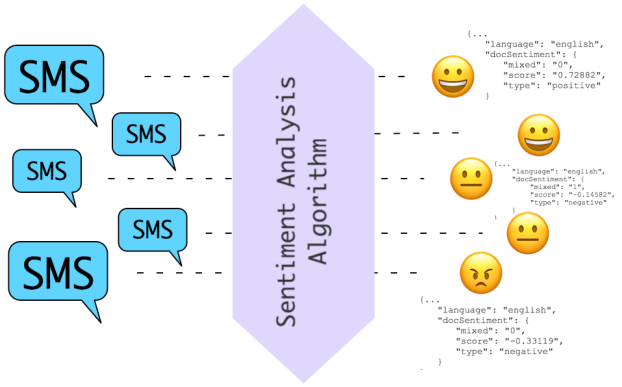



**Figure 2.** A broad summary of what sentiment analysis means [34].

will mainly focus on the English language and the challenges that come with it. All languages generally are ambiguous because words alone do not portray the full context. Other factors such as voice tone, intentions, facial expressions, sentence structure, irregularities such as homonyms, sarcasm, etc. play a huge factor in understanding each other.

#### 2.3.1 Sentiment Analysis

A great place to start with NLP and analyzing language is sentiment analysis. There are different types of sentiment analysis and the most straightforward one mainly focuses on the polarity of the text or speech at hand. Polarity can be measured by seeing whether the text being analyzed is positive, neutral, or negative. Another type of sentiment analysis is detecting the emotion and feeling behind the words being used. In this project, we use two different models where one is a pre-existing model, while the other is built with different NLP methods using a combination of them for prediction. See Figure 2 for a brief view of sentiment analysis.

### 2.4 State Machines and Behavior Trees for Decision Making and Behavior Design

State Machines or Finite State Machines are used to describe behaviors based on a state. These machines are used for decision making, especially for dynamic robotic behavior [9]. These machines can be realized through object-orientated approaches [2]. Another approach to decision-making is the use of behavior trees [33]. Where the core difference is the focus on tasks rather than states. These behavior trees are apt at describing and executing complex behaviors and also to reuse components of lower abstraction levels [10]. Behavior trees have had extensive usage in the field of game AIs [21, 14, 16] as well as within the field of robotics [10, 33, 1].

#### 2.4.1 Ensemble Modeling

Ensemble modeling is a process in which two or more weighted models are used for prediction [12]. The models can either be of different types trained on the same dataset, or they can be the same or similar models trained on different datasets. As long as the models involved are independent, ensemble modeling is expected to reduce the generalization error, that is, the error involved when predicting on previously unseen data. This is because the models can complement each other; where one model underperforms, the other can perform better. For example, one emotion classification model might have a bias towards categorizing happiness as neutral (which might happen because of some bias in the dataset), this bias error can be reduced if it is combined with another model which does not have the bias.

### 2.5 Tone analysis

We, humans, understand emotions not only through the consideration of the message being conveyed but also the tone in which the message is being conveyed. Any advanced emotionally intelligent subject should be able to differentiate between the raw message and the way it is being said as the two may sometimes contradict. For example, a simple “Hello” can communicate different emotions depending on the tone that was used to say it. Therefore using only text analysis can be limiting because a neutral-looking text such as “Hello”, could have been said in a surprised, happy or fearful manner.

To study and classify different spoken sounds (speech) into emotions, it is necessary to investigate their prosodic features. Prosody is the pattern or rhythm of a speech. Speech is full of prosodic features because of grammatical features, language rhythms, and emotions. Such prosodic features are expressed in speech data as pitch levels, intonations, energy intensity modulation, formants, duration, and rhythms [23]. See Figure 3 for a view on speech prosody.

Therefore, extracting these prosodic features and training a model to classify them into different emotions allows us to undergo tone analysis. There are many prosodic features however, there is no consensus on which are the most relevant features to detect emotions [23]. Once the prosodic

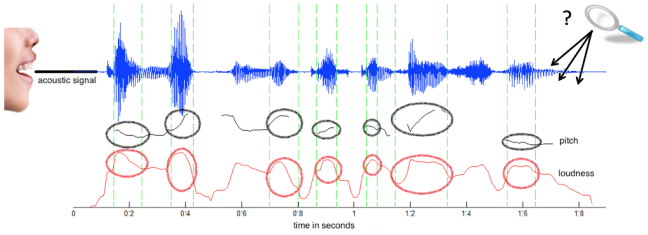



**Figure 3.** Speech prosody [23].

features are extracted it is possible to train a model which maps a combination of different prosodic features to different emotions.

### 2.6 Computer Vision

Computer vision is a field in computer science that aims to give computers the ability to see by mimicking parts of human vision [15]. One way to represent an image to a computer is by using a 2D array for every image pixel value. The value represents intensity and ranges between 0-255 for each of the RGB-channel. Having only one channel results in a gray-scaled image, which can be seen in Figure 4.

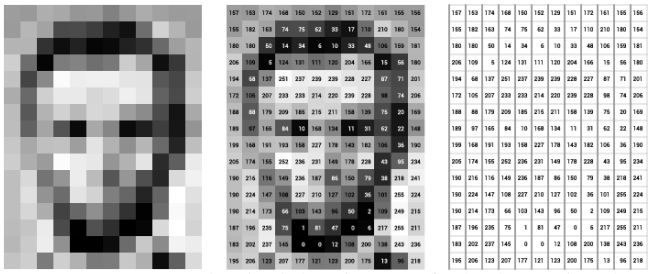


**Figure 4.** Gray-scale pixel data image of Abraham Lincoln [19]

In short, computer vision is all about pattern recognition. One common way to train a computer to understand the visual data given is to feed it a large number of labeled images and use different algorithm techniques to allow finding patterns related to each specific label [15]. With the help of advances in deep learning and neural networks, the field has been able to surpass human ability in some tasks related to detecting and labeling of objects [15].

#### 2.6.1 Emotion Detection with Computer Vision

As stated above, computer vision is all about pattern recognition. How is this used for emotion analysis? In brief, we do the following steps [3]:

1. Acquire the image frame from a camera feed (IP, CCTV, USB camera).
2. Preprocessing of the image (cropping, resizing, rotating, color correction).
3. Extract the important features with a CNN model
4. Perform emotion classification

The data is generally taken from pre-labeled data sets with static images or short 2D video sequences. There are also a few containing 3D images. Since most 2D databases only contain images of frontal faces, training only on them leads to poor performance when trying to predict different head poses [3]. Which emotions the model can predict depends entirely on the classes the data set contains. Most common emotion or sentiment databases contains the following classes [15]:

- Anger
- Disgust
- Fear
- Happiness
- Sadness
- Surprise
- Neutral

### 2.7 ROS

The ROS framework used in the project was ROS2 Foxy. A development computer running ROS2 was used to run the state machine, NLP, and Vision nodes. The development computer communicated with the Pepper robot using drivers that translated the robot API to relevant data such as image and microphone data. The design is shown in Figure 5. To save time in the project, the project used existing code for the Pepper drivers[2] and general ROS2 Docker[3] project setup [4].

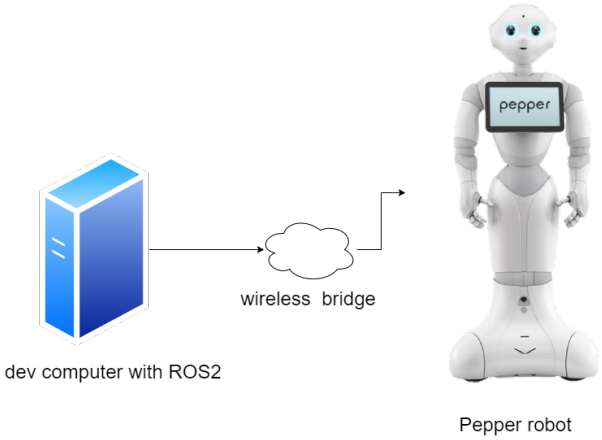



**Figure 5.** The usage of ROS

## 3. Method

Now that we have the knowledge of the tools we need to reach our goals, in this chapter, the methods needed to be implemented will be explained.

### 3.1 Natural Language Processing

The NLP part consists of three nodes communicating with each other. Two *Text-2-Emotion* (T2E) nodes, each using

[2] https://github.com/simwijs/naoqi_driver
[3] https://www.docker.com/
[4] https://gitlab.liu.se/liuhomewreckers/liu-home-wreckers

its own model, take the spoken words as input and output a normalized vector representing the likelihood of the speaker feeling each emotion. The output from these two nodes are then taken as input by a node called *NLP-Emotion-Core* which performs Ensemble Modeling (see Chapter 2.4.1) to improve performance. This fusion is simply a matter of selecting the emotion that has scored the highest value in any of the two input nodes meaning that if the two nodes predict different dominant emotions *NLP-Emotion-Core* will select the emotion with the highest associated value.

The model used by the first T2E node is a pre-trained model packaged in an easy to install and use pypi package called *text2emotion* [30]. The model in this package performs pre-processesing of the input text and filters out words that ”express emotions or feelings” [30]. From these words an emotion vector is created by summing weighted emotion values associated with the relevant words and outputs a value between 0 and 1 for each emotion. This model did not predict the emotion ”neutral” which was added by us using the following formula:

$$neutral = 1 - \frac{< sum - emotion - values >}{5},$$

where *sum-emotion-values* is the sum of all emotion values and the reason for the division with five is because the model predicted 5 different emotions. As this does not create a normalized vector we have added a normalization step at the end.

The second model used to analyze and detect emotion is trained on data retrieved from Twitter. The dataset consists of over 11000 entries. These entries vary from single words, questions, statements, single to multiple sentences joint by punctuation. Since the dataset is from Twitter, it has a lot of punctuation either used in constructing the text or used to make emojis. This in addition to having usernames is why the data needs to be pre-processed and filtered.

When dealing with multiple agents one has two options, either one can build an agent which takes in both your initial models as input or you can handle the output by making it go through a set of conditions or an algorithm. This algorithm returns the right output after weighing each model in the project. In our case, due to time constraints, we went with the latter approach.

An important difference between the outputs each model has is that the first model analyzes the emotion ”surprise” while the second model does not. *Model 2* outputs one of the 4 basic emotions: *joy, fear, anger, sadness,* in addition to *neutral*.

After encoding these emotions, embedding the tokenized data, the model is built as shown in Figure 6. The input was limited to 500 (words) per text/speech. This was a matter of trial and error while checking where the model convergences best before overwhelming it with too many words at once.

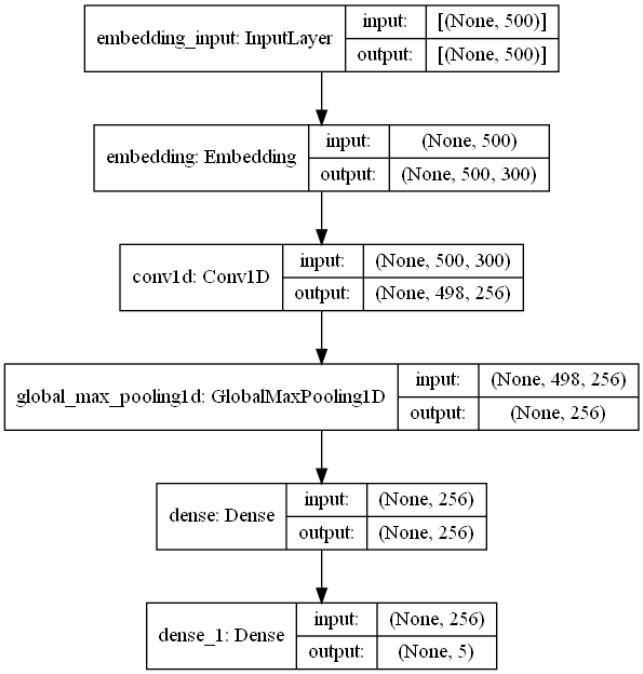



**Figure 6.** The different layers that make up *Model 2*

Since the purpose of this project is not to experiment or explain the different components of the agents used, but instead how we used all these different components to achieve the aim of the project; in this section, we will not go through the details of each layer in *Model 2*. Use Figure 6 to understand how this model functions when being used in the rest of this section to achieve the results in Chapter 4.

### 3.1.1 Evaluating the system

To evaluate how well the three nodes perform, tests were conducted on the *go_emotions* dataset which is a dataset with 70 000 forum posts, namely Reddit posts, with emotion tags [29]. A single post can in this dataset have multiple emotion tags and also be tagged with emotions outside the ones this project tries to detect. To make sure the test gives a clear result only posts with indisputable emotion-tags were used, meaning any post with multiple emotion-tags will be excluded from the test. To better test, our models only data points tagged with an emotion our models can predict will be used in the testing. The emotions are angry, fearful, happy, sad, surprised, and neutral. To quantify the different nodes' performance a confusion matrix is created for each emotion, from which the model's accuracy, precision, recall, and F1-score are calculated.

The *accuracy* is the ratio of how many times a datapoint is classified correctly, meaning the datapoint is tagged with an emotion when it should be tagged, called a true positive, and not labeled when it should not be labeled, called a true negative.

The *precision* is the ratio of how many times the prediction is correct when it assigns a label to a data point.

*Recall* on the other hand is the ratio of how many times a data point that should be given a certain label actually gets

said label assigned to oneself.

Finally, the *F1-score* is a metric that combines the value from precision as well as recall. It is the harmonic mean of the precision and the recall and is calculated in the following way:

$$F_1 = \frac{2}{recall^{-1} \cdot precision^{-1}} = 2 \cdot \frac{precision \cdot recall}{precision + recall}.$$

## 3.2 Vision

The visual system used in this study is based on a few key functionalities:

- Face detection
- Target tracking
- Face recognition
- Emotion detection from an image of a face

### 3.2.1 DeepFace

Due to limiting computational resources and not 're-invent the wheel' the group decided to use existing software to detect faces and to analyze emotional expressions. One such software is the python library *deepface*[24] - not to be confused with *FaceBook DeepFace* developed by Facebook [28]. DeepFace provides an interface to many state-of-the-art models in facial recognition such as Google-FaceNet, Facebook DeepFace, and VGG-Face [24]. In this report, the authors decided to set DeepFace to use VGG-Face.

The emotion-detection model was trained by *Sefik Illkin Serengil*, using VGG-Face [25].

Deepface's *DeepFace*-object has couple of callable methods, see [24]. This project mainly used *represent, verify, find* and *analyze*. Depending on which model is used the exact details might differ. Since the authors used VGG-face, its under-the-hood functionality will be described in more detail.

**VGG-face** is a convolutional neural network [20]. The architecture that performed best in the study consisted of 13 blocks. The input consists of $224 \times 224$ sized images. During training, the faces are not aligned, but during testing, it was shown that a 2D alignment of the faces improved the performance. 10 out of the 13 blocks are convolutional blocks and each of them is followed by a Rectified Linear Unit (ReLU). The last three blocks are fully connected [20].

### 3.2.2 Temporal Stability

When receiving predictions from the DeepFace API, it was noticeable that the predictions could jump between emotions sometimes due to noise or other disturbances. A temporal stability function was added to counteract this effect. The temporal stability can be visualized in Figure 7. It works by combining the most recent measurements, applying weights that depend on the index of the measurement. With a fixed-length queue $Q$ and measurements $m_i \in Q$ where $|Q| \leq 10$, a temporally stable measurement $m_s$ is calculated from a discrete Zipf-like distribution, according to the following formula:

$$m_s = \frac{\sum_{i=0}^{|Q|-1} \frac{m_i}{max(1, i*k)}}{\sum_{i=0}^{|Q|-1} \frac{1}{max(1, i*k)}} \quad (1)$$

The parameter $k \in \mathbb{Z}^+$, adjusts how much emphasis is placed on newer measurements. A larger number k means that older predictions have less effect, which makes the system more susceptible to disturbances. For this project, $k = 3$ because it appeared to be a good trade-off between having stable predictions and not having a slow reaction to change in emotion due to old predictions. We estimate that our changes in facial expression took just a couple of seconds to change the system's verdict, while still keeping predictions on long-lasting expressions stable.

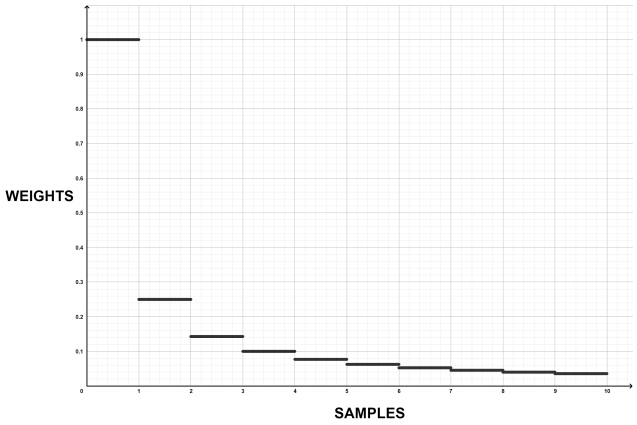



**Figure 7.** Temporal stability where k = 3

### 3.2.3 Tracker

The tracker used in this project uses the Euclidean distance to track objects. The tracker gets the bounding box for every face detected from DeepFace. It then uses the coordinates of every bounding box for each frame and compares it to the next frame. The tracker checks if the distance of a bounding box centroid is less than a certain threshold from one frame to the next. If so, then it is assumed to be the same object. A centroid is a discrete point that lies closest to the geometric center of the bounding box (see Figure 8).

The tracker returns the bounding boxes as well as a corresponding tracker ID. The IDs start from 1 and increment for every new object entering the frame. This process can be visualized in Figure 9.

### 3.2.4 Facial Representation / Recognition

Facial representation is handled by VGG-face through the DeepFace API. From an image (of a face) it returns a 2622-dimensional vector. This vector representation has seemed close enough to linearity, allowing the mean $rep_{avg}$ from measurements $rep_a$ and $rep_b$ of the same person to be a better representation than either of the two individual measurements.

$$rep_{avg} = \frac{rep_a + rep_b}{2}$$

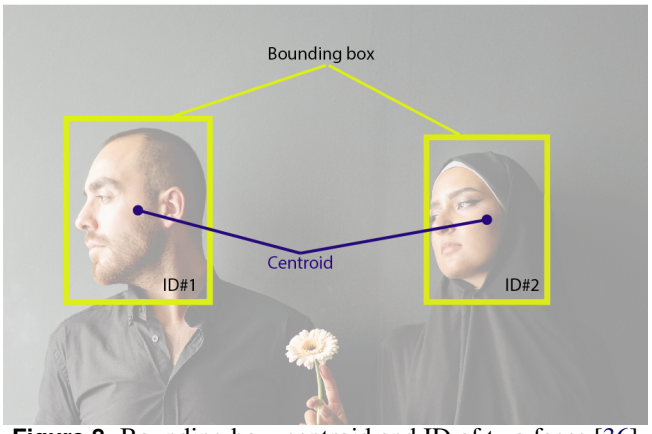



**Figure 8.** Bounding box, centroid and ID of two faces [36]

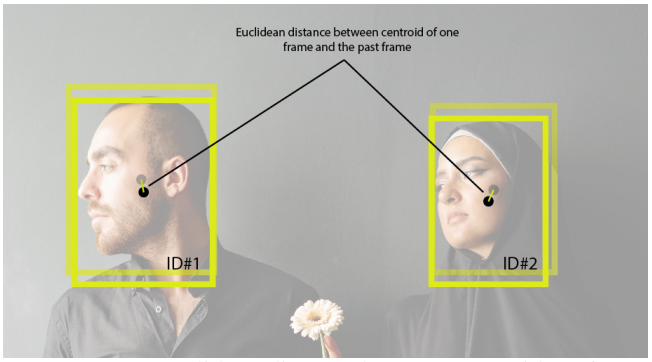



**Figure 9.** Euclidean distance between centroids [36]

We can thus improve the representation of a tracked target as long as it is being tracked. In the project's implementation, such an update is conducted if there are less than 10 measurements on a tracked target, or if a target is rediscovered after being lost by the tracker. Face recognition is done by cosine similarity between two VGG-face representation vectors. Cosine similarity is considered a good distance measurement for face recognition [18].

$$CS(x,y) = \frac{x^T y}{\|x\|\|y\|}$$

If the cosine similarity $CS(rep_a, rep_b)$ is above a threshold $C_{th}$ the arguments are assumed to be representations of the same person.

When a face is detected by the tracker, the face gets a current tracker ID that follows it until it leaves the frame or until the movement becomes to hasty resulting in the of the distance of the centroids becoming larger than a threshold.

Whenever a target is tracked with a new ID, a VGG-face representation is made of the target. If the representation does not match any known person, a new global ID is created for the new target and any new measurements are published attached to this global ID. If the calculated measurement matches a known person, the tracker ID is linked to the global ID of that person and following measurements are published attached to the global ID of that known person. An outline of the tracked ID to global ID loop algorithm is presented in Algorithm 1. For simplicity the averaging of existing representations has been omitted in this pseudo code.

**Algorithm 1** Tracker global id resolver

**with** $dict < global\_id, face\_representation > known\_faces$
**with** $dict < tracked\_id, global\_id > track\_map$
**with** $list < tuple < image, tracked\_id >> tracked\_targets$
**with** $func < image \rightarrow representation > rep$
$0 \leq C_{th} \leq 1$ ▷ 0.7 in our implementation
**returns** $global\_id$

**for** $< image, t\_id > \in tracked\_targets$ **do**
  **if** $t\_id \in track\_map$ **then**
    **return** $track\_map[t\_id]$
  **else**
    $t\_rep \leftarrow rep(image)$
    **for** $< p\_id, p\_rep > \in known\_faces$ **do**
      **if** $cosine\_similarity(t\_rep, p\_rep) \geq C_{th}$ **then**
        **return** $p\_id$
      **end if**
    **end for**
    $known\_faces \xleftarrow{+} < target\_id, tracked\_rep >$
    **return** $target\_id$
  **end if**
**end for**

### 3.2.5 Evaluating the system

To test visual emotion prediction this project used the Pepper robot. Two project team members were instructed to act out each of the detectable emotions in front of its camera. Since our initial intention with this data set was to test the complete system, each test is around half a minute long recording of the target reading a text written to correspond to one of the base emotions, trying to act out as much facial and tonal expression as possible. None of the targets were trained actors.

The measured emotion vectors were then logged together with a timestamp. In the result section plots of this output are available for each of the targets. Since one of the ideas for future improvement of the system was to weigh each measurement with a learned prior per target, we also present modified output based on the posterior class distribution of the target measurements.

## 3.3 State Machine

The lightweight object-oriented Python framework *transitions* [5] was used to implement the state machine in Figure 10. This

[5] https://github.com/pytransitions/transitions

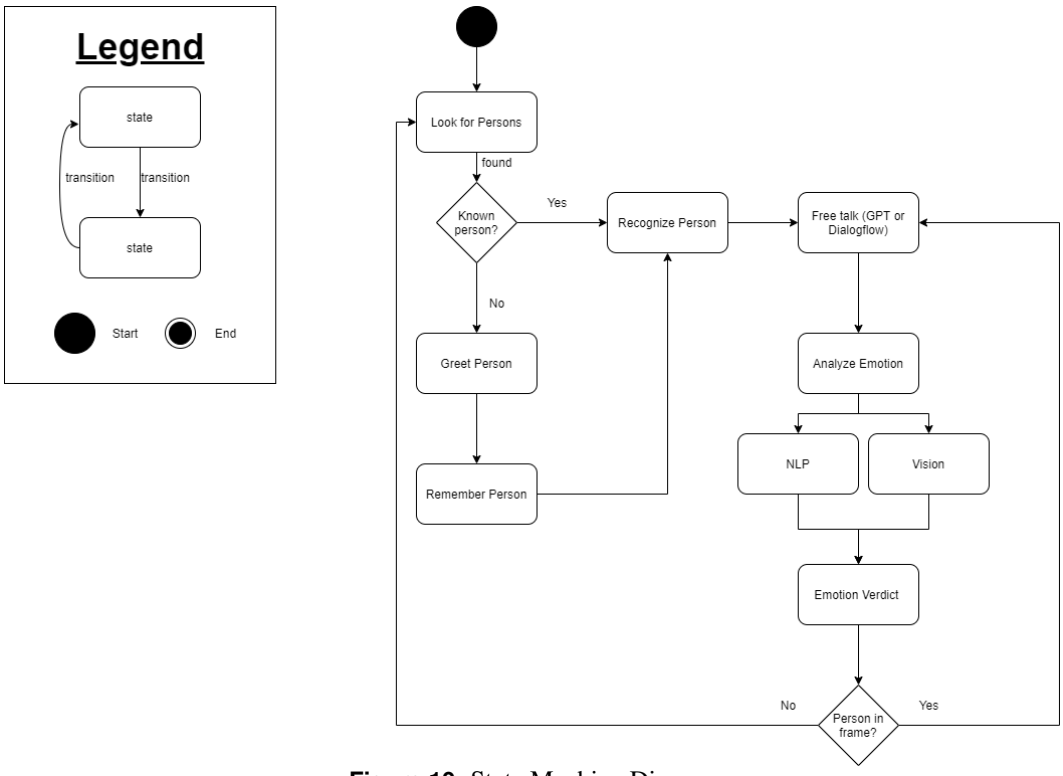



**Figure 10.** State Machine Diagram

framework was used for its seemingly easy and fast implementation of a small state machine for the project. It is able to handle states, triggers, and conditions for triggers to and between states.

## 4. Result

In this chapter, the results of the project are presented. It is divided into the respective areas.

### 4.1 General Framework

The framework upon which the project lies is code written in both C++ and Python in the ROS (specifically ROS2 Foxy) framework. The project added nodes in the existing Pepper project pertaining to the state machine, NLP, and Vision nodes.

### 4.2 NLP

After filtering the *go_emotions* dataset the trimmed version contained 28 189 data points. The number of data points tagged with each emotion in the filtered dataset is shown in Figure 11. There are significantly more data points tagged with ”neutral” compared to the other emotions, roughly six times more than the second most common emotion. The least common emotion in the dataset is ”fear” which occurs 953 times.

The results from testing the three nodes on the *go_emotions* dataset is shown in Figure 12 and includes the accuracy, precision, recall and F1-score for the three NLP nodes, namely the two *Text-2-Emotion* (T2E) nodes and *NLP-Emotion-Core* node.

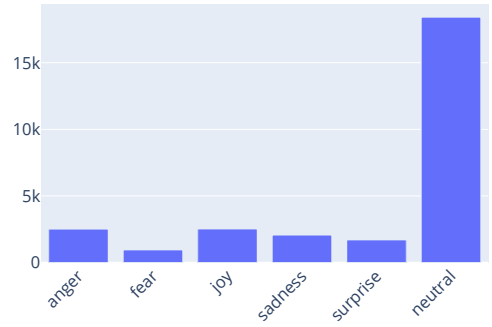
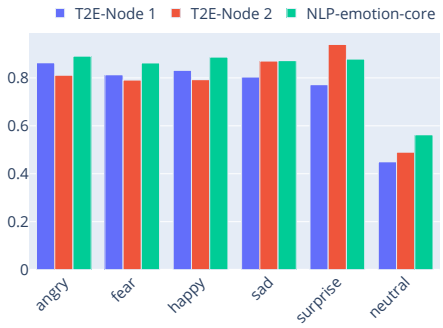
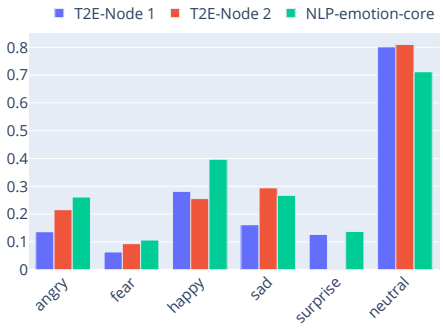



**Figure 11.** Distribution of datapoints tagged emotion in the *go_emotions* dataset

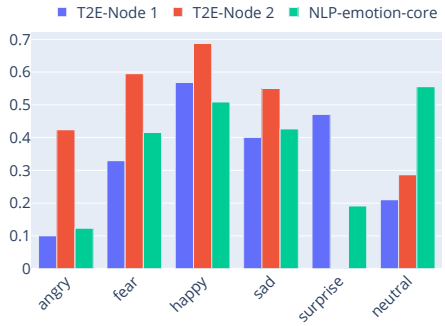
Figure 12 shows us that the combining of the two models, which is displayed by the green bar on each graph, has increased the accuracy in comparison to each separate model. There is an exception to be made when it comes to the emotion *surprise*. As mentioned earlier in this report, our second model (displayed as red in the graphs), does not contribute with predictions for the emotion *surprise*. This is why the bars representing this model are missing for the rest of the metrics. In contrast to *Accuracy*, which is pretty straightforward, the rest of the metrics show us a more peculiar pattern since they analyze each label differently. This will be illustrated in another way and further discussed in chapter 5.



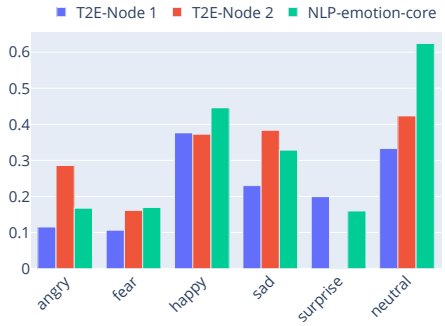

**(a)** Accuracy




**(b)** Precision




**(c)** Recall




**(d)** F1-score

**Figure 12.** NLP-nodes performance on the go_emotions dataset

### 4.3 Vision

The detailed results for the vision assessment are presented in the visual emotion test results Appendix A.1. Each row of plots represents one of the base measurable emotions, enacted by a test person. The first column contains plots of weighted measurements based on the test person's full emotional distribution, and the second column has the raw measurements. Labels of the dominant emotion of each measurement are included as well. For each of the tests, the average confidence in the intended emotion (over the duration of the whole test) is shown below the plot. There is one plot collection for test person 1 (A.1) and one for test person 2 (A.1). Another interesting measurement is the dominant emotion reported at each interval. With the intended emotion in each test assumed to be the displayed one, it is possible to calculate the precision, recall, and $F_1$-measure of the visual dominant emotion predictor for each detectable emotion. These $F_1$-scores for each emotion, with and without prior, are displayed in Figure 13.

In total the system managed to record a total of 199 prediction points across the 7 emotion tests by the 2 targets. These 199 predictions are considered by the project team to be enough to draw some general conclusions about system performance.

| | F1, raw | F1, with prior | Prior F1 change |
|---|---|---|---|
| Anger | 0.00% | 12.70% | +12.70% |
| Disgust | 69.77% | 48.15% | -21.62% |
| Fear | 18.87% | 23.08% | +4.21% |
| Happy | 47.37% | 60.87% | +13.50% |
| Neutral | 49.28% | 48.39% | -0.89% |
| Sad | 27.42% | 30.95% | +3.53% |
| Surprise | 0.00% | 32.43% | +32.43% |
| **Total** | **31.66%** | **35.68%** | **+4.02%** |

**Figure 13.** F1-score of raw and prior-weighted visual predictions

## 5. Discussion

In this chapter, the discussion of the report follows.

### 5.1 NLP

Text-2-Emotion node 2 (T2E2) was trained with data obtained from Twitter, where each document is in a lower case labeled with an emotion. Unfortunately, the dataset used to train Text-2-Emotion node 1 (T2E1) is unknown. Which adds uncertainty to the results from this node. As a group, we can assume that T2E1 has not been trained by data collected from social media the same way T2E2 was.

### 5.1.1 Accuracy

To re-iterate what this metric measures, this is the most intuitive one of them all. Accuracy is simply the ratio of the correctly labeled subjects compared to the total number of data points. Figure 14 displays the results we obtained as a radar graph. The green area, which represents the node combining both our models, covers almost the rest of the node. This tells us that merging the efforts of both agents optimized the accuracy with which our Pepper robot predicts the emotion of the subject.

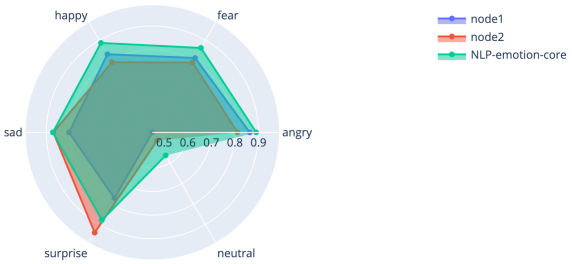



**Figure 14.** A radar graph illustrating the *accuracy* of each node on the *go_emotions* dataset

Now, let us quickly take a deeper look into the emotion *surprise* and why our second model has higher accuracy compared to the rest of the nodes. Although that model was never trained on this emotion, the way the metric 'accuracy' is calculated gives it an almost perfect score. Accuracy is calculated by summing the true positives with the true negatives and dividing it with all the data points: true negatives, true positives, false negatives, and false positives. So, in the case of *T2E-Node 2* by definition of what a true negative is, this node always predicts negative when a document is not labeled with the emotion *surprise*. Which gives it a perfect value on true negatives. This results in a 'useless' perfect score of 1, or 100% inaccuracy. Therefore this small jump in the overall pattern we see in Figure 14 can be ignored.

### 5.1.2 Precision

On the other hand, here we see a missing bar for the node *T2E-Node 2*. This makes sense because precision only takes into consideration the data points resulting in true positives divided by the sum of those points and the false positives. Hence, our second agent has no contribution on how precise our robot is going to be when it comes to detecting how surprised our subjects are.

An interesting change of pattern seen in figure 12 (b), is that the combined efforts of both our models worsened the precision of analyzing text which is *neutral*. We are not sure as to why this happens, but it could be amusing to further experiment with this area in the future.

### 5.1.3 Recall

The metric *recall* measures the sensitivity of our predictions on each emotion. It does that by calculating how many data points (texts) have been predicted correctly by our nodes in comparison to only the positively labeled data points in the *go_emotions* dataset. At this point, we start seeing a staggering difference between our two models. *T2E-Node 2* is highly sensitive to the majority of the emotions we are seeking while having a value of zero with the emotion *surprise* since it predicted no text as 'surprise'.

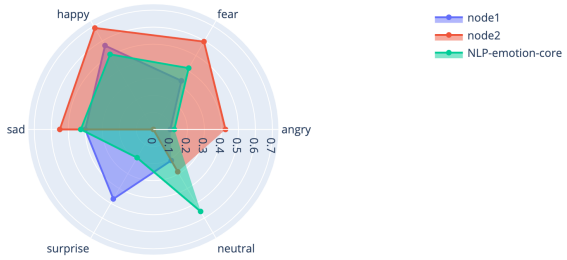



**Figure 15.** A radar graph illustrating the *recall* of each node on the *go_emotions* dataset

This metric is probably the least intuitive and the closest we can get to an absolute measurement when looking at each emotion. What figure 15 further illustrates is that our group could specifically work on *T2E-Node 1* by finding the dataset it was trained on, adjusting its hyperparameters, etc. to make the first model more sensitive to these emotions.

### 5.1.4 F1-score

As mentioned before, *F1-score* takes both precision and recall into consideration. We want our nodes to score a high F1-score to indicate to us that there is a good balance between the precision and the recall. The radar in figure 16 shows us that combining both our models gives us an overall better balance between those two metrics, except for the emotions *sad* and *angry*. This is understandable since the difference between the other two nodes is noticeably bigger with those emotions.

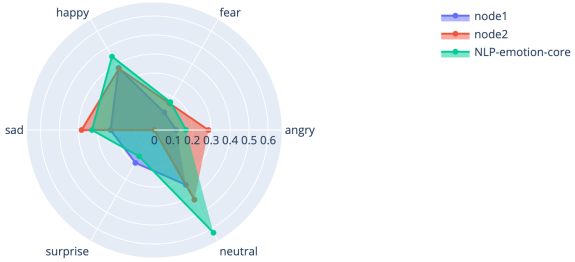



**Figure 16.** A radar graph illustrating the *F1-radar* of each node on the *go_emotions* dataset

Nevertheless, regarding the text analysis part of our pepper robot, we see that combining both our models was a good idea. Running our test with a dataset that is alien to both agents and displaying the results in the fashion shown above has given us clear areas for improvement.

Moreover, it has shown us the strengths and weaknesses of our implementation. We can for example, either fully depend on *T2E-Node 2* when predicting emotions such as 'fear' and

'anger', while depending heavily on *T2E-Node 1* with the emotion 'surprise'.

#### 5.1.5 Sensor fusion

The input from the two *T2E-Nodes* was merged based on the highest value from any of the two nodes. By merging the results in a different way, for example by taking their mean, the results might improve. The potential issue with taking the maximum value ruled by any of the nodes is that the merged result could potentially give overall worse performance as it could select the worst-performing nodes verdict if said node is very certain.

### 5.2 Vision

As seen in Figure 13, the F1 score is around 32% for the raw data, but increases to around 36% when a prior is used. Without the prior, the system never predicts Anger or Surprise for any of the targets, but with the prior these classes become active. Looking at the results for test person 2 (A.1) we saw a great improvement with the added prior since most of the measurements of this target are biased towards sadness. For test person 1 (A.1) we can see that the introduced prior helps prediction of some emotions (Anger, Surprise), at the cost of reducing the confidence in others (Fear, Disgust, Neutral). In a real system, a prior like this would be learned continuously throughout the tests though, so the performance improvement might not be realizable in a real scenario. To properly train such a bias it would also be required to have a good estimation of the likelihood of each emotion. In our case the tests of each emotion were of similar length, so assuming an even distribution over the emotions was reasonable. In a real-world scenario, this is probably not the case.

Apart from the quantitative emotion prediction results, we observed good performance by the tracker and recognizer systems. This is promising for any future work on scaling the system up to be used by Pepper in a more complex task involving longer scenarios with multiple (moving) targets.

Being able to make predictions on several targets at once was not something we planned from the start, since the intended interaction with the robot was one person at a time. However, having a measurement history on other in-frame targets readily available can be beneficial in case any of these secondary targets take over the conversation with Pepper.

### 5.3 State Machine

The state machine was sufficiently competent to be used in the project when only a specific scenario with 1 person was tested. From the experience of the project team members, this state machine implementation is limited by the framework used. In order to aptly be used in a wider context - for instance with more people and additionally in another scenario, it was not diligently designed for - a stronger framework is suggested. There exists at least one suitable framework that is able to handle the ROS ecosystem as well as more complex situations than the framework used. One of these frameworks is *Py Trees* [6]. In comparison with the transitions framework that was used in this project; Py Trees boasts superior usage in the field of robotics and is based on behavior trees to facilitate more complex behaviors. Such complex behaviors include priority handling or 'reactivity' that is built-in to the framework. One of the main reasons for switching the framework to a behavior tree model is to avoid scalability issues when the combinations of traditional state machine nodes increase (which in turn increases the complexity of the system). This combinatorial problem can be expected when this project would be implemented in the wider context of the Pepper robot, where it is capable of more actions than simply analyzing emotion; such as navigating at the same time.

### 5.4 Combined Neural Network

Another point of interest would be to train a neural network on the output from all of our modules from NLP and Vision to assist in connecting the parallel detections to one. It was difficult to connect NLP emotion output with Vision emotion output since these were not temporally connected. Additionally, the nodes output information in different rates, making it harder to realise how to connect the verdicts. A neural network could learn how to combine the areas like in sensor fusion to provide a general emotion verdict that possibly could be stronger than the areas by themselves.

## 6. Conclusion

In conclusion, the project has implemented emotion analysis on a physical Pepper robot as well as provided a base for future work. The robot and its automatic emotion verdict can be used in a wider context such as future research or integrated with other software toward a more general AI and robot. In terms of the primary and secondary goals of the project, both were fulfilled to the point discussed earlier. The robot is able to find new people, recognize them, and determine emotion based on facial expression and transcribed speech.

### 6.1 Future Work

Our model currently uses vision and text analysis to make use of facial expressions and spoken messages respectively to determine the emotion of the speaker. However when we humans detect emotions during a conversation we observe and make use of facial expressions, the message being conveyed, and also the tone of the spoken voice. Therefore, as part of future work, tone analysis of the voice of the speaker can also be utilized to get the full information to determine emotions and thereby enhance the emotional intelligence of the robot.

We already have a working prototype of a tone analysis model which takes in voice (audio data) and classifies it as different emotions. To achieve this we used the model from Derek Hung [6] in conjunction with PyAudio. PyAudio takes raw audio data from the microphone, encodes it in a suitable way, and saves it as a ".wav" file.

[6] https://py-trees.readthedocs.io/en/devel/

The speech analysis model then takes in the .wav file and converts it into a spectrogram. Then from the spectogram, it extracts different features and uses them to classify the voice into emotions. According to the authors of the model, the model can achieve more than 70% accuracy.

The model was tested on a PC and showed some promising results. Therefore, integrating it into the ROS environment of the robot and weighing its prediction in emotion core could potentially boost the accuracy. A possible challenge in the integration process is the correct encoding of the raw audio data stream before the model could use it. The model expects audio data sampled at 16kHz and encoded in int16. In our prototype, we could use PyAudio which takes in the raw audio data directly from the microphone to achieve the correct sampling and encoding. However, the robot sends streams of raw audio data via its microphone ROS node which cannot be fed directly to PyAudio. Therefore, sampling and encoding the audio data right is one of the problems that will need to be solved to achieve proper integration of the tone analysis module.

#### 6.1.1 Behavior Trees

In future work, behavior trees can be explored in a wider context to handle more complex scenarios and behaviors than in this project. By for instance implementing interruptions to the state machine flow when analyzing emotion. In reality, there may be instances where humans refuse analysis, interrupt or divert the robot, or other distractions such as other people intervening. All these behaviors are well suited for a behavior tree decision-making model that can handle interruptions and closed-loop decision-making more efficiently than a standard (lightweight) state-machine implementation.

### 6.2 Acknowledgements

Thanks to our supervisor Cyrille Berger. Thanks to the RoboCup @Home Team LiU@Homewreckers for participating in a demo and the robotics association FIA for providing boilerplate code for the project.

## References

[1] Rahib H Abiyev, Nurullah Akkaya, and Ersin Aytac. "Control of soccer robots using behaviour trees". In: *2013 9th Asian Control Conference (ASCC)*. IEEE. 2013, pp. 1–6.

[2] Paul Adamczyk. "The anthology of the finite state machine design patterns". In: *The 10th Conference on Pattern Languages of Programs*. 2003.

[3] Gaudenz Boesch. *AI Emotion and Sentiment Analysis With Computer Vision in 2022*. https://viso.ai/deep-learning/visual-emotion-ai-recognition/.

[4] Cynthia Breazeal. "Emotion and sociable humanoid robots". In: *International Journal of Human-Computer Studies* 59.1 (2003). Applications of Affective Computing in Human-Computer Interaction, pp. 119–155. ISSN: 1071-5819. DOI: https://doi.org/10.1016/S1071-5819(03)00018-1. URL: https://www.sciencedirect.com/science/article/pii/S1071581903000181.

[5] Prince Canuma. *The brief history of NLP*. URL: https://medium.datadriveninvestor.com/the-brief-history-of-nlp-c90f331b6ad7. (accessed: 25.11.2021).

[6] Mitesh Puthran Derek Hung. *Speech-Emotion-Analyzer*. 2017. URL: https://github.com/MITESHPUTHRANNEU/Speech-Emotion-Analyzer.

[7] IBM Cloud Education. *What is Natural Language Processing?* URL: https://www.ibm.com/cloud/learn/natural-language-processing. (accessed: 25.11.2021).

[8] Paul Ekman. "What scientists who study emotion agree about". In: *Perspectives on psychological science* 11.1 (2016), pp. 31–34.

[9] Michalis Foukarakis et al. "Combining Finite State Machine and Decision-Making Tools for Adaptable Robot Behavior". In: *Universal Access in Human-Computer Interaction. Aging and Assistive Environments*. Ed. by Constantine Stephanidis and Margherita Antona. Cham: Springer International Publishing, 2014, pp. 625–635. ISBN: 978-3-319-07446-7.

[10] B. Iske and U. Ruckert. "A methodology for behaviour design of autonomous systems". In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*. Vol. 1. 2001, 539–544 vol.1. DOI: 10.1109/IROS.2001.973412.

[11] M Shamim Kaiser et al. "Healthcare robots to combat COVID-19". In: *COVID-19: Prediction, Decision-Making, and Its Impacts*. Springer, 2021, pp. 83–97.

[12] Vijay Kotu and Bala Deshpande. "Chapter 2 - Data Mining Process". In: *Predictive Analytics and Data Mining*. Ed. by Vijay Kotu and Bala Deshpande. Boston: Morgan Kaufmann, 2015, pp. 17–36. ISBN: 978-0-12-801460-8. DOI: https://doi.org/10.1016/B978-0-12-801460-8.00002-1. URL: https://www.sciencedirect.com/science/article/pii/B9780128014608000021.

[13] Anis Koubâa et al. *Robot Operating System (ROS)*. Vol. 1. Springer, 2017.

[14] Chong-U Lim. "An AI Player for DEFCON: an evolutionary approach using behavior trees". In: *Imperial College, London* (2009).

[15] Ilija Mihajlovic. *Everything You Ever Wanted To Know About Computer Vision*. https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e. Accessed: 2021-11-29.

[16] Ian Millington and John Funge. *Artificial intelligence for games*. CRC Press, 2018.

[17] Yoshihiko Murakawa et al. "Verification of the effectiveness of robots for sales promotion in commercial facilities". In: *2011 IEEE/SICE International Symposium on System Integration (SII)*. 2011, pp. 299–305. DOI: 10.1109/SII.2011.6147464.

[18] Hieu V. Nguyen and Li Bai. "Cosine Similarity Metric Learning for Face Verification". In: *Computer Vision – ACCV 2010*. Ed. by Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 709–720. ISBN: 978-3-642-19309-5.

[19] Nguyen Dang Hoang Nhu. *Pixel data diagram. At left, our image of Lincoln; at center, the pixels labeled with numbers from 0–255, representing their brightness; and at right, these numbers by themselves.* [Online; Accessed 29, 2021]. URL: https://miro.medium.com/max/945/0*CI5wgSszZnpHu5Ip.png.

[20] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. "Deep face recognition". In: (2015).

[21] Derrick Pemberton et al. "AI or Nay-I? Making moral complexity more accessible". In: *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts*. 2019, pp. 281–286.

[22] Rosalind W Picard. *Affective computing*. MIT press, 2000.

[23] Fabien Ringeval. "Ancrages et modèles dynamiques de la prosodie : application à la reconnaissance des émotions actées et spontanées. (Speech anchor and dynamic models of prosody : application to acted and spontaneous emotion recognition)". In: 2011.

[24] Sefik Ilkin Serengil. *deepface*. URL: https://github.com/serengil/deepface (visited on 12/04/2021).

[25] Sefik Ilkin Serengil. *Pre-Trained Models for DeepFace*. URL: https://github.com/serengil/deepface_models (visited on 12/04/2021).

[26] Chao Shi et al. "How would store managers employ social robots?" In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2016, pp. 519–520. DOI: 10.1109/HRI.2016.7451835.

[27] Ruth Maria Stock. "Emotion transfer from frontline social robots to human customers during service encounters: Testing an artificial emotional contagion modell". In: (2016).

[28] Yaniv Taigman et al. "Deepface: Closing the gap to human-level performance in face verification". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1701–1708.

[29] GoEmotions Team. *GoEmotions - README*. URL: https://github.com/google-research/google-research/tree/master/goemotions (visited on 12/03/2021).

[30] text2emotion Team. *text2emotion 0.0.5 - Project description*. URL: https://pypi.org/project/text2emotion/ (visited on 12/03/2021).

[31] Sam Thellman, Sofia Thunberg, and Tom Ziemke. "Does Emotional State Affect How People Perceive Robots?" In: *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*. HRI '21 Companion. Boulder, CO, USA: Association for Computing Machinery, 2021, pp. 113–115. ISBN: 9781450382908. DOI: 10.1145/3434074.3447140. URL: https://doi.org/10.1145/3434074.3447140.

[32] Sam Thellman and Tom Ziemke. "The Perceptual Belief Problem: Why Explainability Is a Tough Challenge in Social Robotics". In: *J. Hum.-Robot Interact.* 10.3 (July 2021). DOI: 10.1145/3461781. URL: https://doi.org/10.1145/3461781.

[33] Jonathan Tremblay. "Understanding and evaluating behaviour trees". In: *McGill University, Modelling, Simulation and Design Lab, Tech. Rep* (2012).

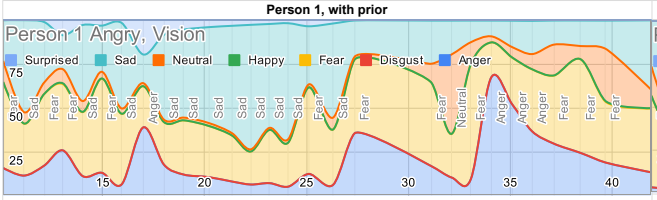
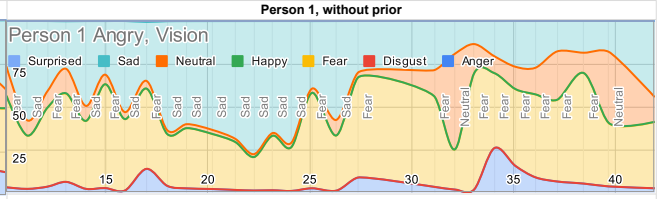
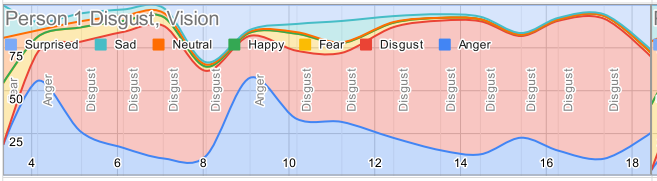
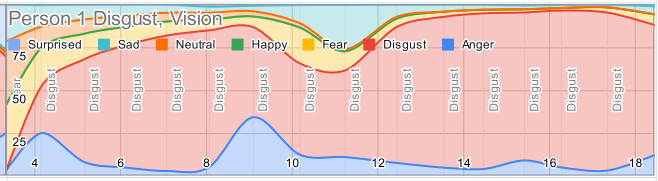
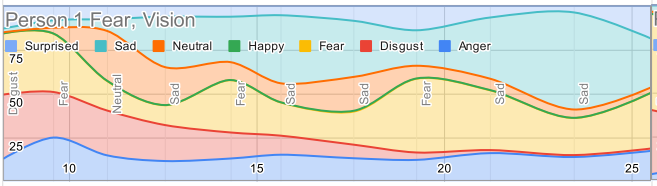
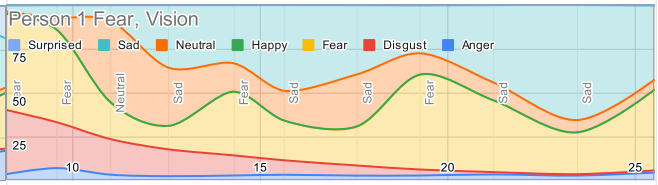
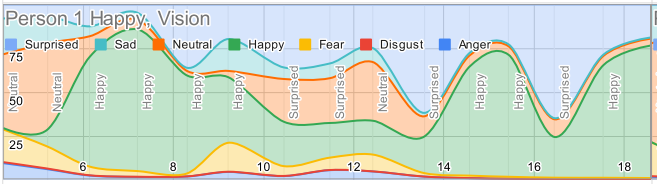
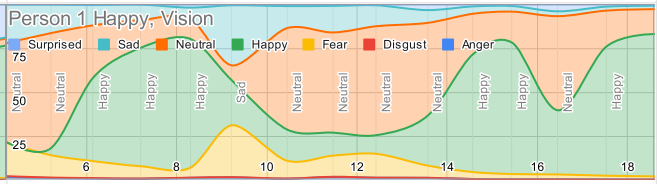
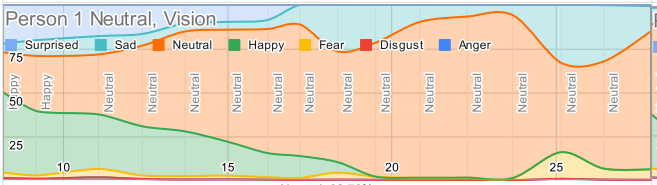
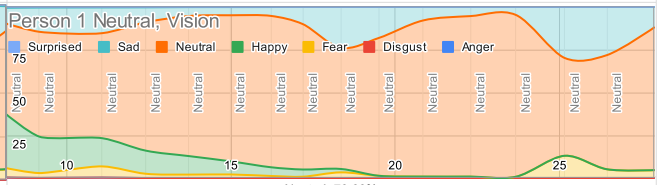
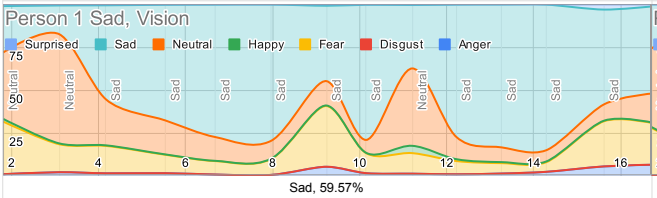
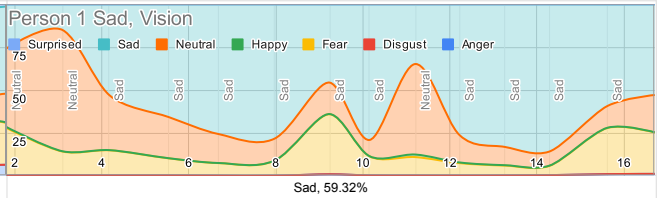
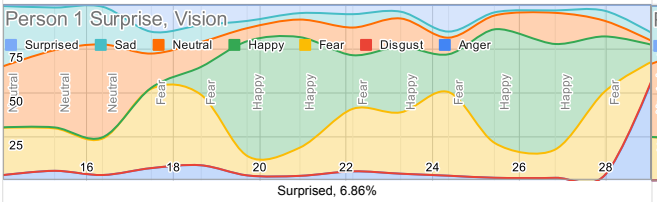
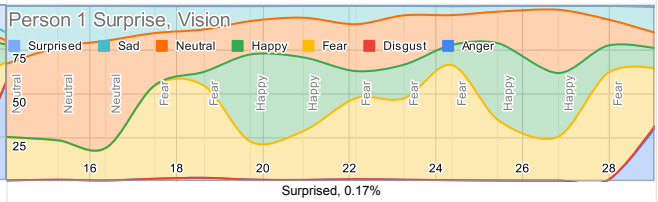
[34] Twilio. *Sentiment Analysis*. Accessed: 2021-12-09. URL: https://www.twilio.com/docs/glossary/what-is-sentiment-analysis.

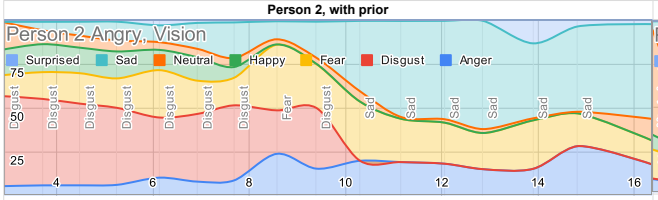
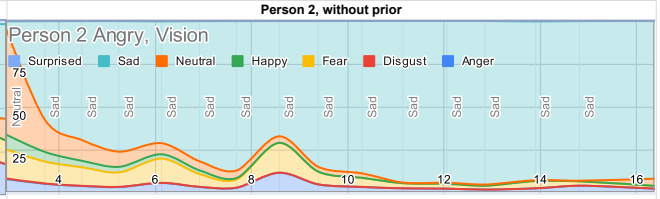
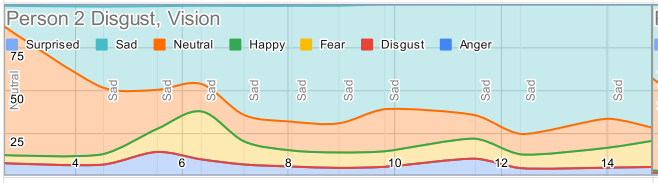
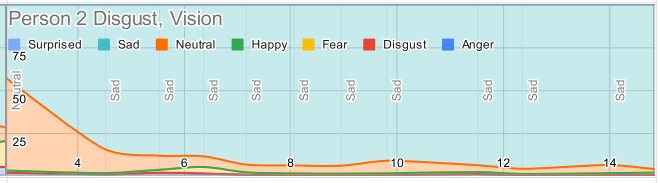
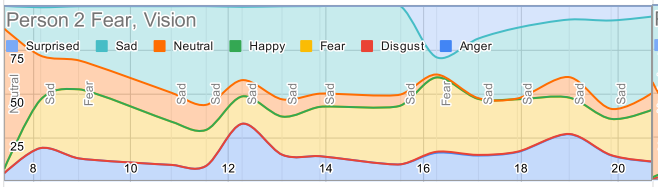
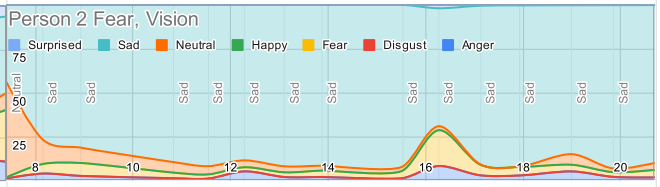
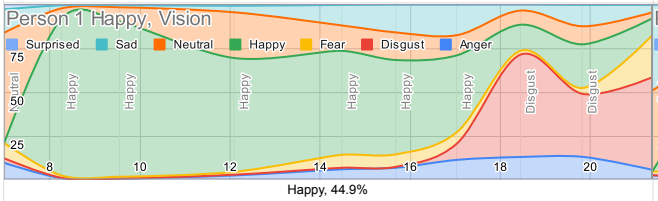
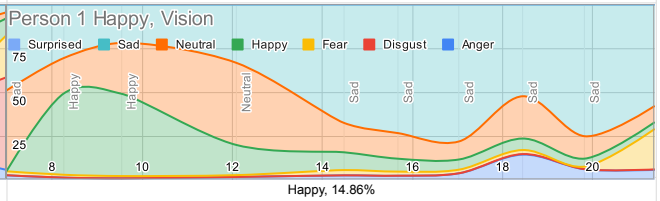
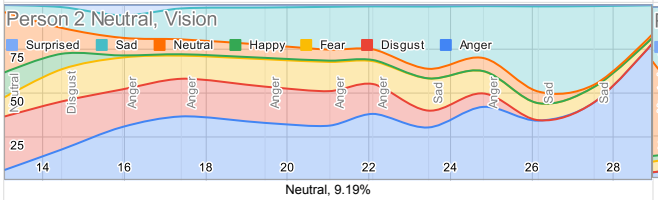
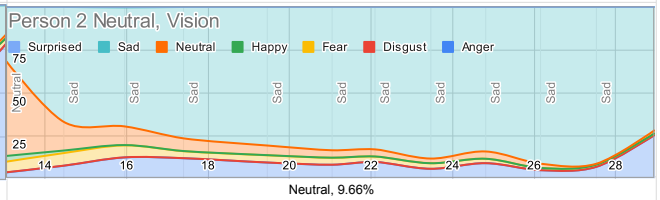
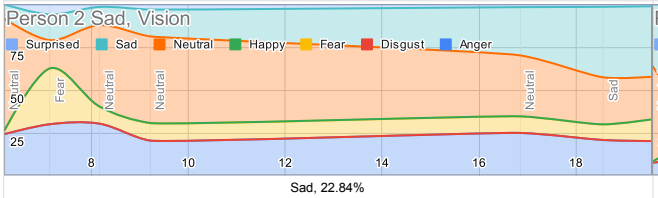
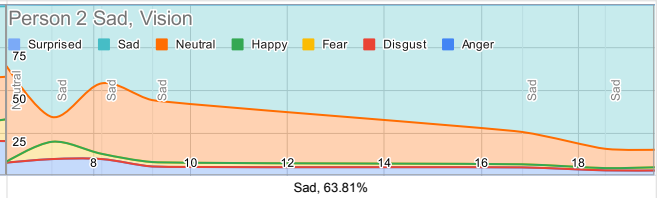
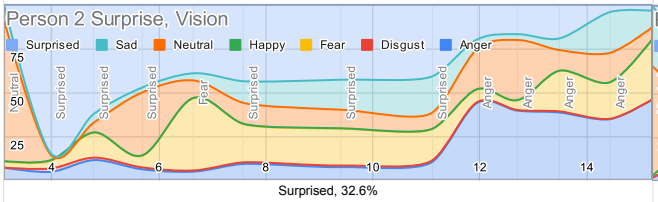
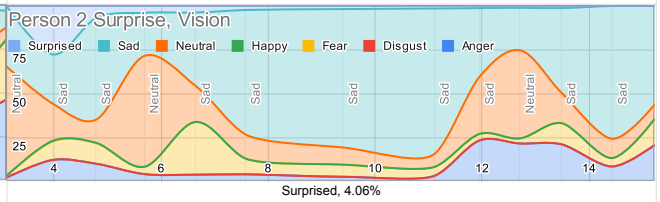
[35] Xoriant. *Natural Language Processing: The next disruptive technology under AI–Part I*. Accessed: 2021-12-09. URL: https://www.xoriant.com/blog/machine-learning/natural-language-processing-the-next-disruptive-technology-under-ai-part-i.html.

[36] Yan Krukov. *Couple Wearing Black Outfit Looking on the Side*. [Online; Accessed 29, 2021], Image has been edited by Philip Ngo in Adobe Illustrator. URL: https://www.pexels.com/photo/couple-wearing-black-outfit-looking-on-the-side-8911470/.

# 1. Appendix

## A.1 Visual Emotion Test Results

| Person 1, with prior | Person 1, without prior |
|---|---|
|  |  |
| Anger, 19.16% | Anger, 4.8% |
|  |  |
| Disgust, 52.92% | Disgust, 67.32% |
|  |  |
| Fear, 26.63% | Fear, 35.22% |
|  |  |
| Happy, 41.78% | Happy, 41.43% |
|  |  |
| Neutral, 62.76% | Neutral, 76.66% |
|  |  |
| Sad, 59.57% | Sad, 59.32% |
|  |  |
| Surprised, 6.86% | Surprised, 0.17% |

| Person 2, with prior | Person 2, without prior |
|---|---|
|  |  |
| Anger, 13.85% | Anger, 4.12% |
|  |  |
| Disgust, 0% | Disgust, 0% |
|  |  |
| Fear, 25.88% | Fear, 4.93% |
|  |  |
| Happy, 44.9% | Happy, 14.86% |
|  |  |
| Neutral, 9.19% | Neutral, 9.66% |
|  |  |
| Sad, 22.84% | Sad, 63.81% |
|  |  |
| Surprised, 32.6% | Surprised, 4.06% |

## A.2 Statement of Contribution

| Team member | Main contributions |
|---|---|
| sebba262 | Produced most of the code in the Vision node (partly with phing272 and johha451), with an exception for the euclidean tracker. Generated test-set and results for the Vision system (with simst932), and analyzed it. Presented the Visual system. Wrote much of method and results for Vision in this report. |
| simru247 | Wrote skeleton ROS nodes for the team and integrated NLP models into ROS. Built a prototype for tone analysis by putting together a tone analysis model and deepSpeech model for speech to text. Contributed to this report in NLP theory and future work. |
| simst932 | Provider of general ROS framework, state machine implementation and related report sections, introduction to project and project administration (meetings, hand-in, how-to-run documentation). Wrote some general discussion and future work related to state machines and neural networks for a single emotion output. Worked a little bit on Vision with sebba262 in terms of producing tests and results. |
| oscli329 | Build text-2-emotion node 1 and integrated with ROS with help from simru247. Implemented NLP-emotion-core together with samya461. Built tool for testing the different NLP-nodes and visualize the results. Contributed to this report as well, mainly method, results and discussion regarding the NLP parts. Helped simru247 trying to get the microphone encoding to work, necessary for the Tone-2-Emotion to work. Unfortunately we did not manage to solve the encoding issue within our time budget. |
| phing272 | Built the euclidean tracker as well as worked together with sebba262 and johha451 to write the general vision logic. Contributed to document writing and powerpoint design. |
| johha451 | Worked with the vision part of emotion detection. Focus a bit on finding articles and code. Contributed to this report as well. |
| samya461 | Built *Model 2* in NLP and integrated with ROS with help from simru247. Implementation of *NLP-emotion-core* together with oscli329. Contributed in document writing: NLP theory, method, result, discussion and revisioning of the whole document. |

**Table 2.** A list of contributions from each group member in *Pepper Robot: Emotion detection* project