Emoji Detection from Facial Data using Kinect HD

Author1
University
Address
email@email.com

Author2
University
Address
email@email.com

ABSTRACT

UPDATED—February 5, 2017. Human emotion detection has been a wide area of research for a very long time. Different ways to detect humans have been researched. Detecting expressions from static images has been done using different classifiers such as Neural Networks or SVMs. Detecting expression from video poses more challenges such as evaluating each frame in real time and figuring out the emotion. Emotion detection has applications in various domains. As humans interact more and more with computers and devices, taking emotions into account will play a vital role in deciding how humans interact with computers. In this work we present a method to detect human facial expressions using the Microsoft Kinect 2. We utilize the FaceHD and FaceBasics data from the kinect to classify expressions and display an emoji accordingly. We present our results in classifying 8 different emojis. We use a gesture based approach to classify the emotions. Each gesture is then mapped to a particular emoji as shown in the results. We calculate threshold values for our conditional statements These threshold values are calculated empirically and we show their performance with our experiments. We also show that just with a few samples from the face data we are able to get good enough features to classify different emojis.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; See http://acm.org/about/class/1998/ for the full list of ACM classifiers. This section is required.

Author Keywords

HCI; Kinect 2; Emoji Detection; emotion detection.

INTRODUCTION

Human emotion detection and classification is a big area of research and definitely has applications in various fields especially Human Computer Interaction. We now have devices such as Google assistant and Amazon echo that are capable of interacting with human beings. However, they don't take into account sentiments or emotions. Emotions play a vital role in communication for humans. Charles Darwin was one of the first scientists to recognize that facial expression is one of the most powerful and immediate means for human beings to communicate their emotions, intentions, and opinions to each other [1]. As we make more and more advances in developing devices that interact with human beings, we would need to come up with ways for efficiently detecting and making sense of facial emotions.

ISBN 123-4567-24-567/08/06.

DOI: http://dx.doi.org/10.475/123_4

Figure 1. Some of the emojis that we detect. Each emoji is mapped to a gesture which is recognized by our program.

Another area of application can be MOOCs. A MOOC is a Massive Open Online Course. MOOCs have gained immense popularity over the past few years due to the advancement and development of online video sharing sites such as YouTube. The classroom of the future could be set up online where students would attend lectures by watching videos at their convenience. The system can then track the emotions and expressions of the students and automatically figure out when the student feels anxious about the material or when the student seems disinterested. Based on this detection that system can improvise to provide a better learning environment for the students.

Facial expressions have also played an important part in some medical diagnosis. The work presented in [5] shows a study of different patients suffering from depression. Emotional analysis plays an important role in the diagnosis of such individuals. Another work presented in [6] has used behavioral studies of facial emotions to study autism spectrum disorders.

In this paper we used the Microsoft Kinect 2 to get facial data and use that data to classify human emotions. We then display the related emoji to that expression. Kinect provides us with two different streams of data: FaceHD and FaceBasics. We use both these streams in this work to classify the emojis from

Figure 2. All the points sampled on using the FaceHD data stream

Figure 3. The FaceBasics data stream

facial data. We provide results based on conditional statements based classifier. The threshold values for the statements were calculated empirically. Our results show the robustness of our method. We also show that it is not important to sample a very large number of points on the face. A small number of points is sufficient to get very good results.

The rest of the paper is organized as follows. We first discuss some of the related work that has been done in this area. After that we describe our interface design and the algorithm design. We then provide results from our experimental analysis. Finally we conclude with some future work and future directions for this work.

RELATED WORK

In this section we will talk about the related work that has been done in this area. Our work is related mostly to gesture recognition, emotion recognition. We also discuss related work from field such as feature recognition from videos. We consider the detection of skin pixels from videos. Skin detection has been extensively studied in the literature and used for various purposes. Once the skin is detected, the positions of various parts of the body can be detected and then the gestures performed can be used for classification.

Emotion recognition

The work presented in [14] provides a survey of various 3D and 4D recognition algorithms that have been used to detect different facial expressions. The paper first talks about 3D methods and their challenges. 3D data can be collected and reconstructed from static images. A lot of research has been done to improve the methods of reconstruction of 3D data from static images. One main method of doing this is the 3D Morphable model. The authors provide examples of this method used to construct 3D information from a 2D image. Structured light is also used to capture 3d information from dynamic 2D images. In [11] the authors talk about a method to detect faces under challenging conditions such as different expressions, low resolution of light etc. The 3D sensor of the kinect is used to collect the point cloud information about the face. Since the detection is done using a low resolution sensor, a lot of the data is sparse. The sparse data is filled by the algorithm to produce good results under challenging conditions. The work presented in [7] uses the Kinect to detect human faces and use that for recording MOOC videos. The depth sensor and the kinect is used to recognize the teacher's face and record the video of it. The teachers's head movements are detected using the depth sensor. There are three main parts in this system, a PPT screencast stream, a teacher face video stream, and a whiteboard stream.

Emotion detection from facial expressions has also been extensively studied. The authors in [10] describe a method to classify human emotions from static facial images. The authors here again make use of highly powerful convolutional neural networks to detect the emotions. Convolutional neural networks have proven to generate great results in classifying data thanks to the recent amount of research that has been done in this area. Another work [2] provides a method to incorporate more than just computer vision approach to detect human facial expressions. The main focus on expression detection has been by computer vision researchers. The authors of this paper try to integrate audio and video information to efficiently detect human facial expression. In [4] the authors use a convolutional neural network to classify human emotions. They then use this information to match the emotion with the right emoji. They use a wide variety of dataset to train their neural network. Convolutional neural networks can take a lot of time to train but have shown to produce accurate results given a huge amount of data.

Gesture Recognition

A major part of our work is gesture recognition. Gesture recognition has also been a part of extensive research. Gesture recognition pertains to recognizing meaningful expressions of motion by a human, involving the hands, arms, face, head, and/or body. The paper presented in [12] provides a survey of different gesture recognition methods that have been studied in the literature. The emphasis of this paper was hand based gestures and facial expressions. The work presented in [13] talks about the FERET database that has been made to test different face-recognition algorithms. FERET stands for Face Recognition Technology, and is a large database of facial images. Another device similar to Kinect is the Nintendo wii controller. The work presented in [15] uses the wii controller for gesture recognition. They use the acceleration sensor independent of the gaming console for gesture recognition.

Feature Detection

Other related areas of research include detection of features from videos such as detecting human skin from unconstrained videos. Once skin is detected, the gestures performed can be calculated for various purposes. The work presented in [8] provides a statistical method to detect skin pixels from images. The authors compare a histogram method to a mixture model method.These methods are prone to false positives due to the presence of ambient light and are often not scalable to

Figure 4. Given data from a certain facial expression, can we detect what the underlying emotion conveyed is?

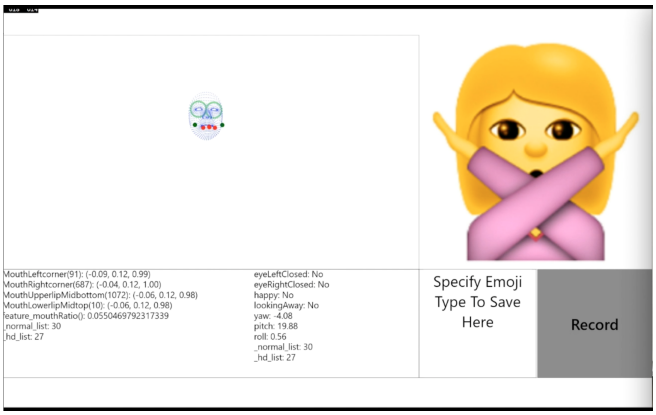

Figure 5. Interface Design. The left side is used to display the different points that we sample from the data. The different points are shown in different colors. The right side of the interface is used to display the emoji. We also have a place where you can click on "Record" and record the corresponding data. There is also space to label the data.

skin colors of different ethnicities. More recent work in this area includes [17]. The work presented here tries to refine the model for each person. A 2D histogram and a mixture model is used to detect the skin pixels. Other robust skin detection methods have been presented in [9] [16]. Adapative skin detection based on a normalized lookup table is presented in [3].

PROBLEM STATEMENT

Given data from Kinect over a period of 20 to 30 frames, generate a gesture-based emoji recognition tool.

INTERFACE DESIGN

Figure 5 shows the design of our interface. The left side is used to display the different points that we sample from the data. The different points are shown in different colors. The right side of the interface is used to display the emoji. We also have a place where you can click on "Record" and record the corresponding data. There is also space to label the data. This way we provide an interface to collect labelled or unlabelled data easily. We were not able to display the video along with the data points. The reason for this was that when we display the video, the frame rate drops rapidly and the program

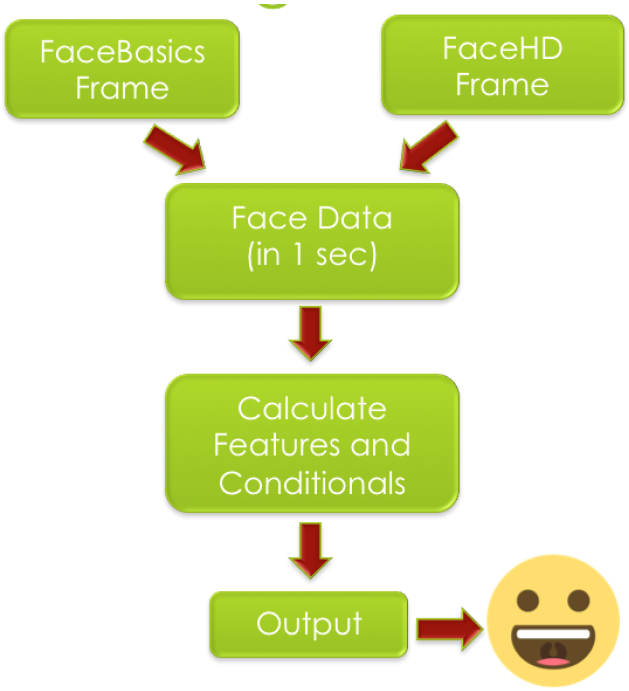

Figure 6. Basic outline of the algorithm. Data is taken in from two different data streams: FaceHD and FaceBasics. 30 frames are captured in a second and averaged out to take into account missing or null frames.

becomes very slow. We were limited by the hardware here as we used a Surface Pro 3 for our program. To use a Kinect with PC the basic requirements are 4Gb RAM and a 3.1GHz processor. While the RAM in our Surface Pro 3 was 4Gb, the processor speed was only about 2GHz. So as you can see, the Surface Pro 3 barely matches the required specifications for using a Kinect 2.

SYSTEM DESIGN AND ALGORITHM

In this section we will describe the system and the algorithm we used. Figure 6 shows the basic outline of the system. It gathers data from data sources and then calculates features and conditions. As you can see, we have two different data sources as input: FaceHD and FaceBasics. One second of data is captured, which is about 30 frames. From these 30 frames, features and conditions will be calculated And the system will generate an emoji based on these features. The details are as follows:

Data Sources

Kinect provide 2 data streams for face tracking: FaceHD and FaceBasics. FaceHD stream provides more than 1000 facial points in the 3D space per frame. These points represent different parts of the surface of the face. Every poins is a triple of X, Y, and Z value. The origin is located at the camera's optical center (sensor), Z axis is pointing towards a user, Y axis is pointing up (as shown in Figure 7). The measurement units are meters for translation and degrees for rotation angles. These points seems to be very accurate. Unfortunately, it's not the case, and it lacks some very important information. So that's the reason why we need to use data from FaceBasics.

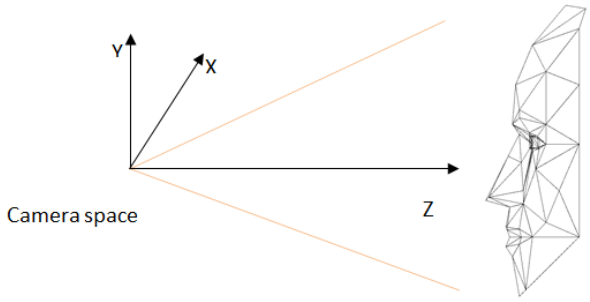

Figure 7. The coordinate system of Kinect.

On the other hand, FaceBasics stream provides us with some basic face features, such as whether an eye is closed or not. This proved to be extremely helpful in our case because the FaceHD data do not capture the change of everything on the face. The points around the eyes are available but the eyelids are not captured. This makes it hard to figure out when the eyes are closed or open as there is not change in the points around the eyes in either case. Each feature value might be one of the 4 different ones provided by Kinect. The details are in Table 1.

Result	Description
Yes	very certain that the property is true
No	very certain that the property is false
Maybe	pretty sure that the property is true
Unknown	don't have enough information

Table 1. The feature values from FaceBasics.

Data Processing

As mentioned before, We used both the FaceHD and FaceBasics frames provided by Kinect. All data within 1 second will be accumulated, and then be used for the following stage. We used one queue for each type of data, so there are 2 queue internally: one for FaceHD data and one for FaceBasics data. Whenever a new frame arrive, we extract the data we want and put it on the back of its queue, and then remove the old frames whose arrival time is 1 second earlier than the latest one.

Ideally, the frame rate is 30 from Kinect, so the length of each queues is around 30. However, sometimes the frame rate will drop due to insufficent computation resources, which is a little problematic because it's hard to apply classifiers if the amount of data will vary. Our system right now uses only features, so it's fine. But in order to extend our system to incorporate more advanced classifier, we also considered keeping more frames (e.g. all frames in 2 seconds) and then using only the latest fixed number of frames (e.g. the latest 30 frames). And we might consider paucing the system if the frame-rate is too low.

Features

We calculated features from the data we gathered. There are 2 kinds of features in our system: static features and dynamic features. Static features are calculated frame by frame, which means there are about 30 values for a given static feature whenever we do calculation. For example, for each frame from FaceHD, we calculated the ratio of the distance between the upper lip and the lower lip to the width of the mouth. And there will be about 30 of it, with different timestamps, in the data.

On the other hand, dynamic features are calculated from all frames at once. So there will be only 1 value at a given time. For example, the maximum movement of head. For each FaceHD frame arrived, we gather both static and dynamic features in current database, and then use these features to decide which emoji to display.

For most of the static features, we apply an additional dynamic counting feature on them. It's because the data from Kinect is not stable, and only when the number of a static feature reach a given threshold do we treat it as active. Some features are discussed below:

Eye Closed or Open:
With the help of FaceBasics data we are able to get "Eye-Closed" feature. This has a value of Yes, No, MayBe and Unknown. It's a static feature, so we apply the dynamic counting feature. Only when the total number of "Yes" exceed 60% of total number of frame will the system signal an "Eye Open" dynamic feature. Many other static features from FaceBasics apply the same kind of method, but with different threshold and features values.

Angles of movement of head:
The angle of movement of the head is a dynamic feature. We had three different gestures that used this feature. The yes emoji was mapped to the vertical movement of the head. "No" emoji was mapped to the horizontal movement of the head. The confused emoji was also mapped to the movement in the horizontal plane but with different angles.

Mouth Ratio:
Using the FaceHD data we sampled some points around the mouth as shown in figure 8. This ratio was used to calculate and figure out when the user was smiling, or when the user had their mouth wide open etc. The threshold values for these were empirically determined.

Conditional Statements

In the classification, the pure dynamic features (e.g. nodding) take precedence over the actived static features. So the corresponding emojis such as "shaking head" or "nodding" will be displayed first. And for static features, we active them with the following sequence. Our algorithm is based on conditional statements as shown below. We show that even with basic intuitive statements, it is very easy to classify different emojis. A sophisticated classification algorithm would then be able to generalize and scale our solution to incorporate more emojis.

1. Tired: We active "tired" emoji when both eyes are closed and mouth is opened.

2. Happy: The happy emoji use the static feature "happy" from Face Basics. It's a rather stable feature, we use 0.5 as active threshold.

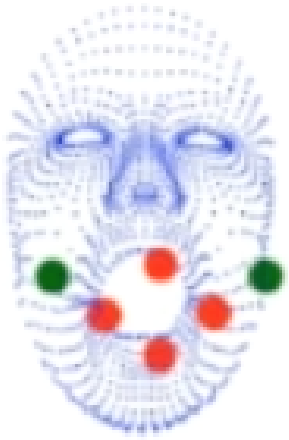

Figure 8. The points used for mouth ratio

3. Wink: It will be triggered if one eye is closed and one eye is open.
4. Shock: It use mouth ratio and will be activated when mouth is opened long enough.
5. Annoyed: It will be displayed when both eyes are closed and the person is not happy.

EXPERIMENTAL SETUP

For the experiments we ran tests on a single user. The user in this case was Chia-Cheng (Jeremy) Tso. We spent some time in figuring the accurate distance from the Kinect and the correct lighting conditions. Surprisingly, the Kinect was very robust against different lighting conditions and backgrounds. This was extremely helpful since we did not have to spend much time in setting up the experiments.

The video was recorded separately using and iPhone 6 camera. We were not able to switch on the video input from the kinect. As explained earlier, this was because switching on the video drops the frame rate rapidly and leads to slowing down of the program. After the video was recorded we stitched the two videos together to make the demo. The demo can be found at this link: https://www.youtube.com/watch?v=yg0jN_frlSM

RESULTS

In this section we will show the results from our system and discuss the implications. Figure 9 to Figure 15 show all the different emojis classified by our program. We were able to detect and classify 7 different emojis with over 95% accuracy. There were cases where our program was confused between emojis for example consider the emoji for tired and the emoji for surprised. The only difference is that eyes are closed in case of tired. Sometimes our program would not detect

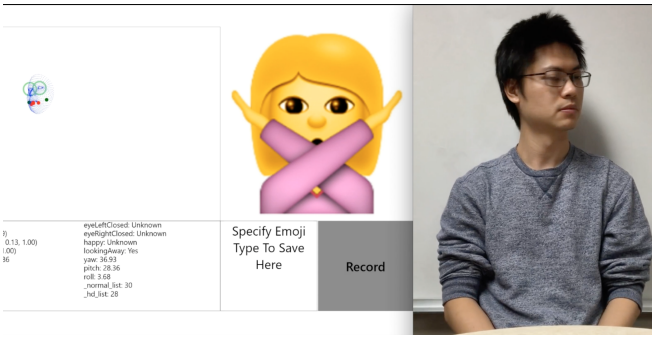

Figure 9. Detection of the "No" emoji. A head shake in the horizontal direction displays this emoji.

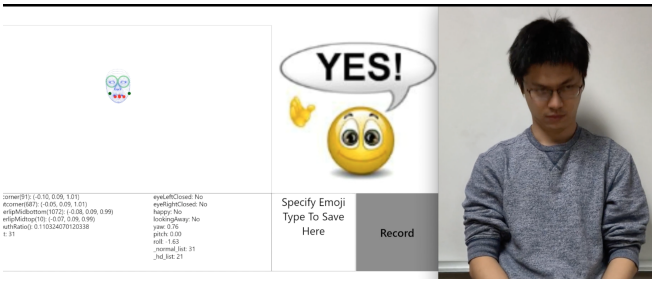

Figure 10. Detection of the "Yes" emoji. A head shake in the vertical direction displays this emoji.

whether the eyes were closed or not and that would lead to it classifying these emojis in a wrong way. However, we found that this problem was easy to solve if the user was at a sufficient distance from the Kinect. The distance must be sufficient enough to sample the points near the eyes accurately.

The other emojis were easy to classify and had enough mutual exclusion to guarantee 100% detection results. Our results show that it is not necessary to capture all the points from the face to do the classification. For capturing the information about the shape of the face we sampled 4 points. We can see that it was enough for us to use just these points to do the classification.

Emoji	Classification Accuracy
Wink	100%
No	100%
Yes	100%
Tired	95%
Confused	95%
Happy	100%
Annoyed	100%

Table 2. Classification accuracy for different emojis.

DATA COLLECTION AND DEEP LEARNING

As shown in the interface design part we have space to record the data. The data can then be stored in any desired format. We planned to use the data to train a deep learning neural network. For writing our neural network, we used the Keras library in Python. The Keras library provides easy way to write a

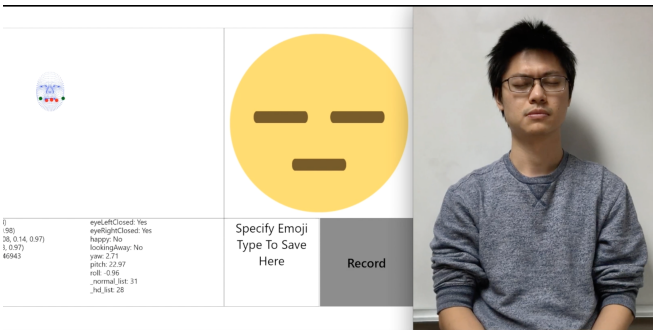

Figure 11. Detection of the “Annoyed" emoji. Both eyes closed and a straight face produces this emoji.

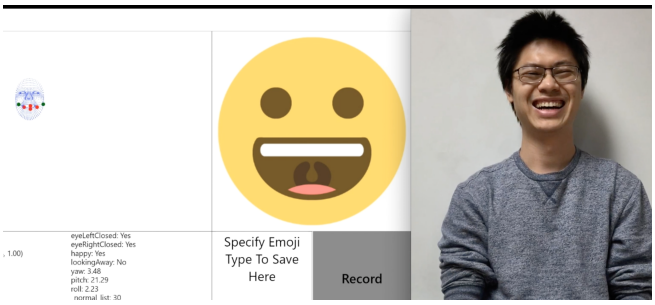

Figure 12. Detection of the “Happy" emoji.

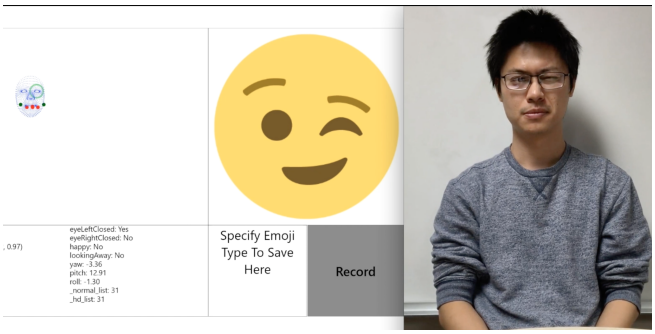

Figure 13. Detection of the “Wink" emoji.

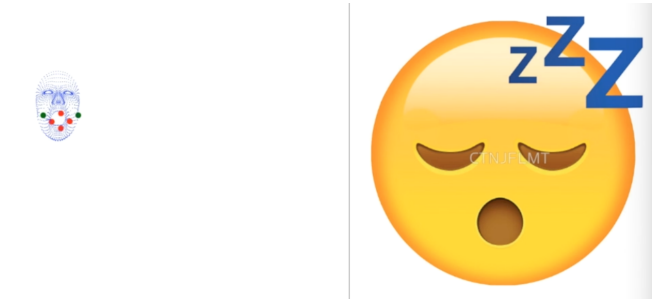

Figure 14. Detection of the “Surprised" emoji.

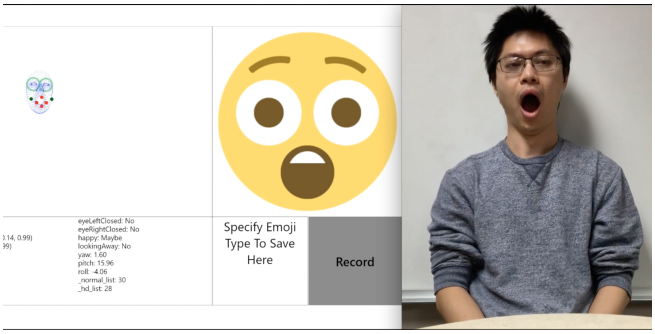

Figure 15. Detection of the “Surprised" emoji

Neural Network and then train it with the given data. It can be customized to use Tensorflow or Theano in the background. We were able to write the code for the network and train it on some data that we found online.

However, due to the lack of time, we were not able to collect enough data to train the network. It was hard to figure out if the data recorded was correct and it takes a long time to actually collect all the data. Neural Networks require a large amount of data to train and produce good results. We did not have enough time to collect the data and this is something we would like to do for future work. We would also need more time to make sure that our data being collected is accurate enough and not prone to errors. This task may be harder than it looks because we would need some kind of metric to classify if the collected data is accurate enough to be used by a classifier. The naive way of doing this would be to collect a lot of data and measure the accuracy of a well known classifier on it.

FUTURE WORK

We want to be able to collect large amounts of data from the kinect and train our deep learning network. We already have the mechanism of collecting the data so now all we need is to have different users and collect the data from their facial expressions. We would also like to try the performance of different classifiers on the given data. This would also allow us to scale our system to incorporate more emojis.

Another area of future work is to have an unsupervised learning approach to detect the emoji. Unsupervised learning would eliminate the need to classify the emojis in the data and would lead to better scalability in terms of the number of emojis classified.

CHALLENGES AND LESSONS LEARNED

We had initially thought that using the FaceHD data we would be easily able to detect various kinds of emotions. However we soon figured out that this is not always the case. More data does not always means better results. We also faced a lot of challenges in the beginning. It took us almost a week just to get everything set up. At first we did not have a windows laptop to start working with Kinect. When we acquired a windows laptop we realized that we would need a Kinect adapter to connect to the laptop. We had to spend some time acquiring the adapter. The lesson learned here was that whenever you

work with hardware, you need to take into account the time it may require to set it up and get everything working.

CONCLUSION

In this paper we have presented a method to detect emotions using data from facial expressions. We used the Kinect 2 from Microsoft to collect the data. We were successfully able to classify seven different emojis with very high accuracy. We have shown our results from the experiments to validate our claims. We showed that it is not necessary to sample all the points from the face to make a good classifier. A few correct points and some conditional statements are sufficient to generate a basic emoji classifier. We also provided an interface to ease the collection of data to train a classifier in the future. The data can be stored in different formats and can then be fed to a classifier for training purposes. We also talk about the deep learning classifier that we developed. For future work we would like to use our data with the classifier.

REFERENCES

1. Marian Stewart Bartlett, Gwen Littlewort, Ian Fasel, and Javier R Movellan. 2003. Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction.. In *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on*, Vol. 5. IEEE, 53–53.
2. Liyanage C De Silva, Tsutomu Miyasato, and Ryohei Nakatsu. 1997. Facial emotion recognition using multi-modal information. In *Information, Communications and Signal Processing, 1997. ICICS., Proceedings of 1997 International Conference on*, Vol. 1. IEEE, 397–401.
3. Fernando Roberti de Siqueira, William Robson Schwartz, and Helio Pedrini. 2013. Adaptive detection of human skin in color images. (2013).
4. Dan Duncan, Gautam Shine, and Chris English. Facial Emotion Recognition in Real Time. (????).
5. Ruben C Gur, Roland J Erwin, Raquel E Gur, Alexander S Zwil, Carolyn Heimberg, and Helena C Kraemer. 1992. Facial emotion discrimination: II. Behavioral findings in depression. *Psychiatry research* 42, 3 (1992), 241–251.
6. Madeline B Harms, Alex Martin, and Gregory L Wallace. 2010. Facial emotion recognition in autism spectrum disorders: a review of behavioral and neuroimaging studies. *Neuropsychology review* 20, 3 (2010), 290–322.
7. Yuran Hu and Zeyu Chen. 2015. Kinect-Based Face Recognition and Its Application in MOOCs Production. In *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2015 7th International Conference on*, Vol. 2. IEEE, 298–301.
8. Michael J Jones and James M Rehg. 2002. Statistical color models with application to skin detection. *International Journal of Computer Vision* 46, 1 (2002), 81–96.
9. Michal Kawulok, Jolanta Kawulok, and Jakub Nalepa. 2014. Spatial-based skin detection using discriminative skin-presence features. *Pattern Recognition Letters* 41 (2014), 3–13.
10. Gil Levi and Tal Hassner. 2015. Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. ACM, 503–510.
11. Billy YL Li, Ajmal S Mian, Wanquan Liu, and Aneesh Krishna. 2013. Using kinect for face recognition under varying poses, expressions, illumination and disguise. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*. IEEE, 186–192.
12. Sushmita Mitra and Tinku Acharya. 2007. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37, 3 (2007), 311–324.
13. P Jonathon Phillips, Harry Wechsler, Jeffery Huang, and Patrick J Rauss. 1998. The FERET database and evaluation procedure for face-recognition algorithms. *Image and vision computing* 16, 5 (1998), 295–306.
14. Georgia Sandbach, Stefanos Zafeiriou, Maja Pantic, and Lijun Yin. 2012. Static and dynamic 3D facial expression recognition: A comprehensive survey. *Image and Vision Computing* 30, 10 (2012), 683–697.
15. Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. 2008. Gesture recognition with a Wii controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*. ACM, 11–14.
16. Mohammad Shoyaib, Mohammad Abdullah-Al-Wadud, and Oksam Chae. 2012. A skin detection approach based on the Dempster–Shafer theory of evidence. *International Journal of Approximate Reasoning* 53, 4 (2012), 636–659.
17. Wei Ren Tan, Chee Seng Chan, Pratheepan Yogarajah, and Joan Condell. 2012. A fusion approach for efficient human skin detection. *IEEE Transactions on Industrial Informatics* 8, 1 (2012), 138–147.