Universiteit Leiden

Opleiding Informatica

Video rating and sorting with a genuine approach

Name:	Terry Zabel
Date:	27/07/2015
1st supervisor:	F. Verbeek
2nd supervisor:	M. Lamers

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

1 Introduction

In this section we will give a short introduction, followed by the problem statement that comes with a research question. Also we will discuss some related work about the research topic and give an overview of what will be discussed in this thesis.

1.1 Problem Statement

In the past few years the technology has rapidly evolved. Together with this increased performance also the software could be and is developed to a much higher level. In this research these improvements will be used to discover more about human computer interactive systems, in particular a system that is based on facial recognition to see if you can get a more genuine interaction than the current human computer interactive systems. The current popular human computer interactive systems for watching videos and voting for them only use buttons to vote. Examples of those popular platforms are Vine and Youtube, see Figure 1 for their voting systems. Since these platforms are used by millions of people we thought of a way to satisfy these users even more and in more various and inventive ways. In the first place we wanted to design a commercial application that maybe could compete with the current platforms by letting people vote through their facial expression. As a way to attract people to our platform instead of the current platforms we wanted to create a application that would rate videos only in a positive and easy way and thought smiling to vote for a video would be the best way. This means that the videos we want to show in our application are meant to be funny videos and the rating of how many smiles will show how funny the video actually is. To accomplish this we have made a design and a prototype that only works locally in the phone. This project has been elaborated in a paper [1]. After doing this project that lead to a somewhat minimalistic prototype we wanted to actually use it for something and came up with a research. In this research we want to find out if there will be a difference in amount of votes for a video if people vote using the current clicking system compared to the automatic and more genuine way of voting by smiling. So we want to find out if our new way of voting actually will produce other ratings for videos than the current way. We want to do this using the following research question: *How does a rating system based on smile recognition function compared to a "standard" voting system?*

1.2 Related work

1.2.1 Smile Recognition

As there are many academic researches on facial expression recognition, there are not much researches specific about smile detection according to [2]. Especially when used on a mobile phone in an application other than the camera of the phone there is no research on how it could be implemented well enough to make it usable and usefull. However, research on facial expression recognition is also interesting with regard to our research project as rating can be approached a lot of different ways other than smiling. Taking this into account we think our project is a start of showing affection for something through a electronic device in a different way. A step in the same direction is already made with the idea of an "affect button", designed for users to give affective feedback by hovering over a button [3]. While the users hovers over the button the facial expression shown on the button changes according to the coordinates. When the user agrees with the facial expression shown on the button he clicks the button and gives affective feedback this way. The disadvantage of this method is that it still takes a lot of effort for the user to give feedback, while we try to rate with as low effort as possible by the user to get a most genuine as possible response.

1.2.2 Video rating systems

Besides the standard way of upvoting by clicking a button there are other ways of rating videos also shown in Figure 1. Vine for example also keeps track of the amount of times a video is watched in amount of loops, the right symbol in the figure. Youtube also provides the possibility to downvote to rate the video. Another way to rate videos could be by giving an amount of stars out of a maximal amount to give a wider view on how good the video is. The problem with all these methods is that they all require an action of the user which has influence on the initial genuine response of the user.

1.3 Overview

To answer the research question we have used the prototype we made in the previous project [1]. This prototype was by far not sufficient to use in the research as it could only be used locally on one mobile phone. To actually make it sufficient for the research we have expanded the prototype, deleted features where necessary and added a backend to make it possible to use the application online on every compatible mobile phone. Then we spread the application online to collect results and at last we have come to some conclusions according to the results. So in short we will discuss the following in this thesis:

- Material and methods including: API's/libraries used for smile recognition, database, server.
- Design of the application and the experiment.
- Implementation of the design.
- Results of the tested implementation.
- Conclusions and discussion about the results.

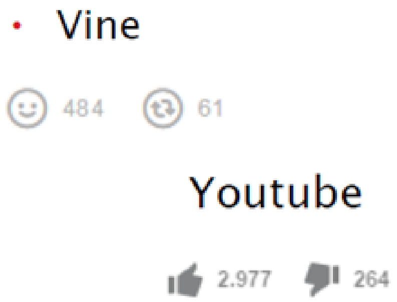


Figure 1: Vote button Vine (left) and vote button youtube (right)

2 Material and methods

In this section we will elaborate on the API's and libraries that we used, the concept of smile recognition and the backend that we used to set up the experiment.

2.1 API's and libraries

For the application and its backend we have used a few API's and libraries which will be elaborated in the following subsections.

2.1.1 Snapdragon SDK for android

After researching various smile detection algorithms, we found that the one used in Snapdragon SDK for Android was the best choice, [4]. Snapdragon is a suite of system-on-chip (SoC) semiconductor products designed and marketed by Qualcomm for mobile devices. Snapdragon semiconductors are embedded in most Android devices. With an easy to implement Java library for facial recognition available in the Snapdragon SDK it appeared to provide the quickest and most precise degree of smile detection. The algorithm first recognizes the face(s) and then rates a smile for every face that it could find on the camera in real-time. Furthermore, this SoC is common in many popular android smartphones such as the Nexus 4, Samsung Galaxy S4, LG optimus G, HTC one etc. Since the algorithm uses the SoC the choice resulted in a surprisingly quick detection, while not putting much strain on system resources, due to the simple fact that the algorithm runs at the hardware level. Other options that would not utilize this level of hardware acceleration may have resulted in a less fluid interface. We found for example that our rather powerful laptops already had to assign quite a bit of processing power to a Java based smile detection algorithm that relied on software only. This made us somewhat worrisome about the limited processing power of mobile devices, especially considering the fact that these have to play a high resolution video at the same time. Also with regard to our targeted user group, this fast and fluid experience was extremely important. As with all massive mobile media, speed is of the essence. Waiting for a video to buffer, lag or an unresponsive device has a devastating effect on the users utility, even if it is only for a few hundred milliseconds.

2.1.2 Node.js server

As previously mentioned the application we made works completely online with a server and database in the backend. In the first place we had to chose the API for the server which had to accomplish communication between the database and android application. Although there are possibilities to directly communicate to a database with Android, we did not want to expose the data of our database by putting the login details in the application itself. Because Android uses Java as the main programming language, Android applications are easy to reverse-engineer and it would be easy to extract the login details of our database. So to accomplish a more secure way to communicate with the database we chose the relatively new node.js API. This API is using javascript as the main language which means in our case we did not have to translate JSON-objects sent from our Android application to other models. Furthermore, the API uses a wide variaty of modules which you can chose from and install freely on your server. In our case, because we were familiar with MySQL we chose a MySQL module to make it possible to communicate directly with our MySQL database. Then we had to find a way to send messages, containing JSON-objects, between the node.js server and the Android application.

We searched the internet for possible ways and the only thing that kept popping up in every search for communication between node.js and android was using the websocket.io module available for node.js. This meant we could send and receive messages in the form of websockets to the Android application with our server and at the same time directly communicate with the MySQL database to get the data needed to handle the requests from the application. The main reason to use node.js over other API's to communicate is that it is much faster than the other methods when handling a large amount of requests at the same time, Figure 2. As our server needs to handle a lot of requests, described later in the implentation section, it was by far the best way to accomplish a fast and reliable communication.

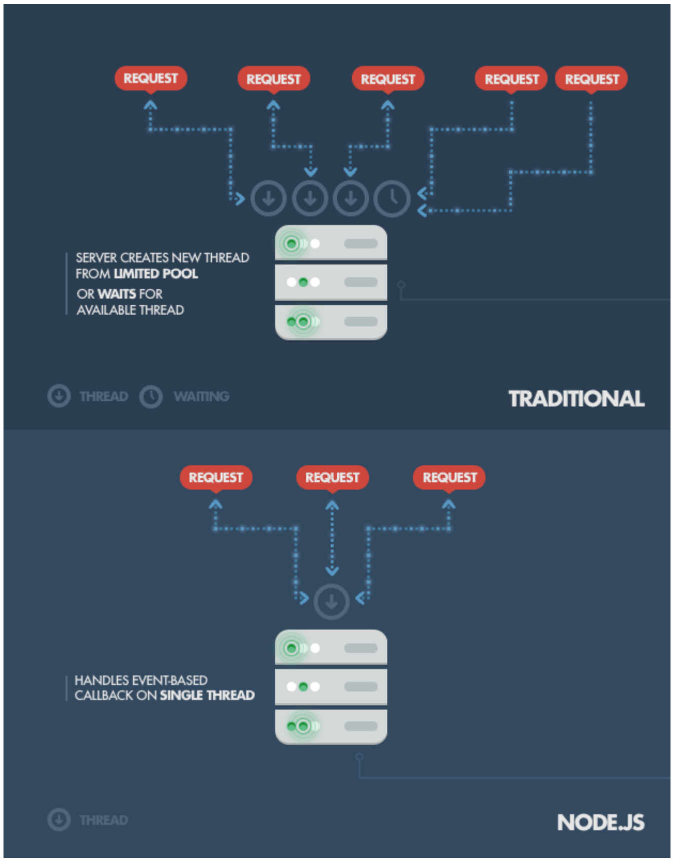


Figure 2: Node.js server compared to traditional server

2.1.3 Gottox library

Now we could handle the requests of our application we needed to be able to send and receive websockets, containing JSON-objects, with the Android application itself. To make this more clear and easy to do than the standard way to send websockets with the Android API, we used the Gottox library, [5]. This library makes it possible to easily implement the connection with the server and keep this connection alive to be able to send and receive messages, in the form of websockets, directly at the moment it is needed.

2.1.4 MySQL database

To store our data we used a MySQL database. This was the best option since we only needed a simple storage to keep track of the users, which videos they already have watched, provide the metadata for the videos and store the results. The most important SQL queries we used are to select one random video that the user did not watch yet and storing the data after watching a video. These queries are the following:

```
SELECT * FROM videos WHERE video_id NOT IN (SELECT video_id FROM
      watched WHERE user_name = userName) ORDER BY rand() LIMIT
    0,1
INSERT INTO watched SET user_name = userName, video_id =
    video_id, voted = vote, laughed = laughed
```

2.2 Smile recognition

While this research initially was meant to create a more genuine approach for rating videos and discovering what the differences are in amount of votes using different methods, it also can be looked at in a different way. Since this project was set up as a research project the focus emphasis was to get an answer on the research question formulated in the introduction. On the other hand it can be looked at as a new and innovating interaction technique, which should take minimal effort and is more convenient for the user. In this section we will elaborate on the current use of smile recognition algorithms and the ways we think it can be used in already existing platforms and applications.

2.2.1 Current use of smile recognition algorithms

The oldest and still most common use of smile detection algorithms is in digital photography, this is also the only purpose smile detection is actually used for at the moment. Many modern cameras and smartphones boast various setting using this clever technique. The camera detects all faces in the picture, checks whether they are all smiling and closes the shutter when they do. The use of the smile detection algorithm in user interfaces (UI) is not completely new. For example, Deniz, Castrilion, Lorenzo, Anton & Bueno [6], designed an application that automatically added emoticons to an instant messaging client when the user smiled at the webcam, to enhance written communication. However, we couldnt find any previous occurence of UI integration on mobile platforms. We feel that the possible UI applications of this technology are therefore highly underexplored.

2.2.2 Possibilities for smile recognition algorithms

Since the use of smile recognition is not very diversified yet we think there is plenty of space in other areas than just photography to make good use of smile recognition. For example the application we made can maybe be looked at as a expansion of youtube where users not only can see how good the video is, but also how funny. For youtube videos this maybe is less appropriate because there is the possibility to upload very long videos and there is a much higher chance a user laughs about something else while watching the video. Vine at the other hand is a platform where users upload videos of exactly 6 seconds. Also the videos are categorized on this platform from which "comedy" is one of the categories. In this case the amount of smiles can be used as a fast and more genuine rating of how funny the video is. This can be expanded to other emotions, but as most people watch videos for their amusement we suggest to limit the expressions to only the positive expressions.

3 Design

The most important thing in researching this kind of human computer interaction (HCI) is that the user does not have to do any more effort than normal as it will alter the results and have a good reference to compare with. With this in our mind we designed a application followed by an experiment which will be described in the following subsections.

3.1 Application Design

Because we did not want the user to have to do any more effort than needed we initially wanted to make the application as much like a commercial application as possible, [1]. This way the user would be already familiar with the kind of application which means the user interface would not be a interfering factor for the results. When we started working on the application and were thinking about how to set up the experiment this did not seem entirely convenient. As said, in a research like this where you want to compare results, you need a good reference to compare your new results with. We thought of making a reference application, but this would mean when the user uses our application he already knows what the application is about and maybe control his smile according to what he normally would vote for. Also the user would have to watch the same video twice. One time using our new application and one time using the reference application. In many cases the video is not as funny the second time as the first time, which would give distorted results. So we decided to not use a reference application, but make the application look like a standard video application without giving anything away about the smile recognition that was going on. To not give away anything about the smile recognition we removed the preview and left out the response system and toplists rated by smiles. Removing all this also meant we could make our video player a lot bigger which is quite an improvement if you are watching a video on a phone. There is no possibility to watch the video fullscreen as the front camera we use for the smile recognition would not be in the direction of the face. We also removed the possibility to log out, Figures 3 and 4. As we want the user to watch the videos only once, we do not want them to be able to log out and watch the videos again with another username. Another option would be to give the user a specified username and password so they could use it only once, but to conveniently do this without spending too much time of the user the names and passwords should be generated. This means a user can just generate another new username and password and do it twice anyway. So we just kept the user logged in without the possibility to log out, so when they are finished they will be stuck at the end screen. The only way to generate results more than once is to completely remove the application and install it again, so you have the option to pick a username again. As it is not entirely possible to have a convenient application nd no possibility to do the research twice we were fine with the effort and knowledge it would take this way to disrupt the data. So in the end we have an application which should be easy to use and collects all the data we need.

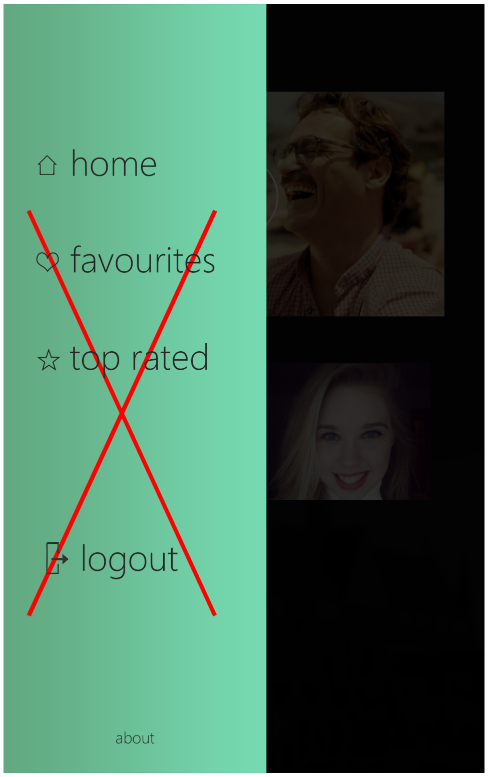


Figure 3: Favourites, top rated and logout removed.

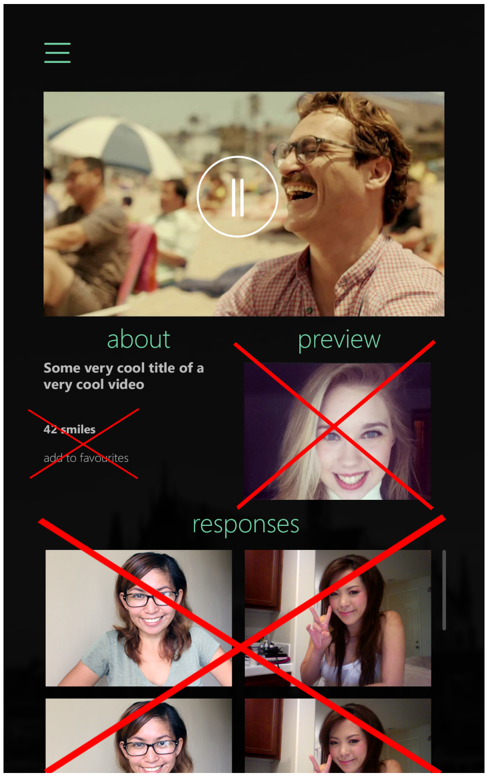


Figure 4: Preview, votes, favourites and response system removed.

3.2 Experiment Design

To make sure as much people as possible would participate in the research we tried to make participating a small effort. To accomplish this we wanted to set up a small list of "funny" videos that the user would watch. To collect these videos and to make sure they were specified as "funny" videos we went to the preselected comedy section of Vine to make sure our own sense of humor would not alter the outcome of the research. While watching the videos the front camera of their phone will be used to secretly capture whether they smiled watching the video or not, including a few seconds after the video. This extra time is needed since in a lot of funny movies the punch line is at the end of the video and the smile about it will often come just after the video ends. After this time has expired and for each video a window will pop up asking if they want to vote for the video they just saw or not. At the moment they pick their option the data about that one video will be send to the server and the next video will be loaded instantly to make it a fluid experience. So for the user it is just watching the video, clicking a button to vote or not vote, watching the next video etc. As we did not want to give away to the users what the experiment was about, we did not state anywhere that there was smile recognition involved until the point where the user has watched all videos and therefore finished their participation in our research. The only thing we had to tell our users in advance that could give away the point was, was that the front camera was used and they had to make sure it was pointed in the direction of their face. This was needed to make the smile recognition precision as accurate as possible. Apart from the minimal effort we want our experiment to take for the user, we also wanted to spread the application in some way that participating could be done from anywhere and at anytime. The only constraint we had to make was the possibility of a steady internet connection as the videos are streamed from our server and the data needs to be sent to the database for our research. To accomplish this we made a website where the application can be downloaded as a .apk-file (Android package) and information on how to install this file on your mobile device, [7]. This way it is theoratically possible for everyone with internet connection and to download the application and be part of the research as long as they have an android device with the SoC described in the Snapdragon SDK section implemented. We did not want to let our users figure out themselves if they had the right specifications before using the application, so we check if the specifications are sufficient as soon as the user opens the application. If the specifications are not sufficient the application will handle this with a corresponding screen without telling them what the research was about in case they want to try it on another mobile device which may have sufficient specifications. So in the end the user only needed to download our .apk, install the file on their mobile device, pick a username, watch the videos and for every video choose to vote or not.

4 Implementation

As you can see in the design there were quite a few changes made from the original idea during the process. Most of this changes were necessary changes that came up when implementing the original design in the application and while asking people their thoughts about the idea of our application to respond to their needs and dislikes. The process of implementing our ideas in the application and the experiment will be described in the following subsections. An overview of the communication network between application, node.js server and database is shown in Figure 5.

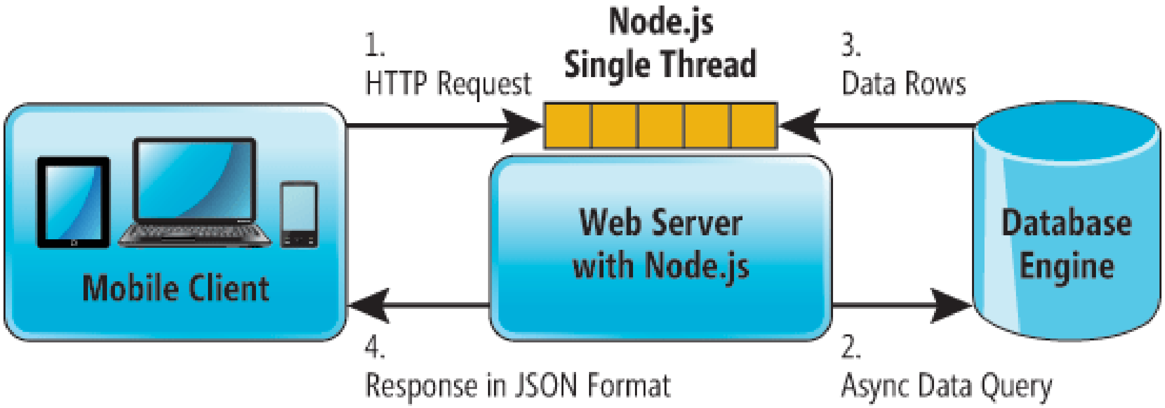
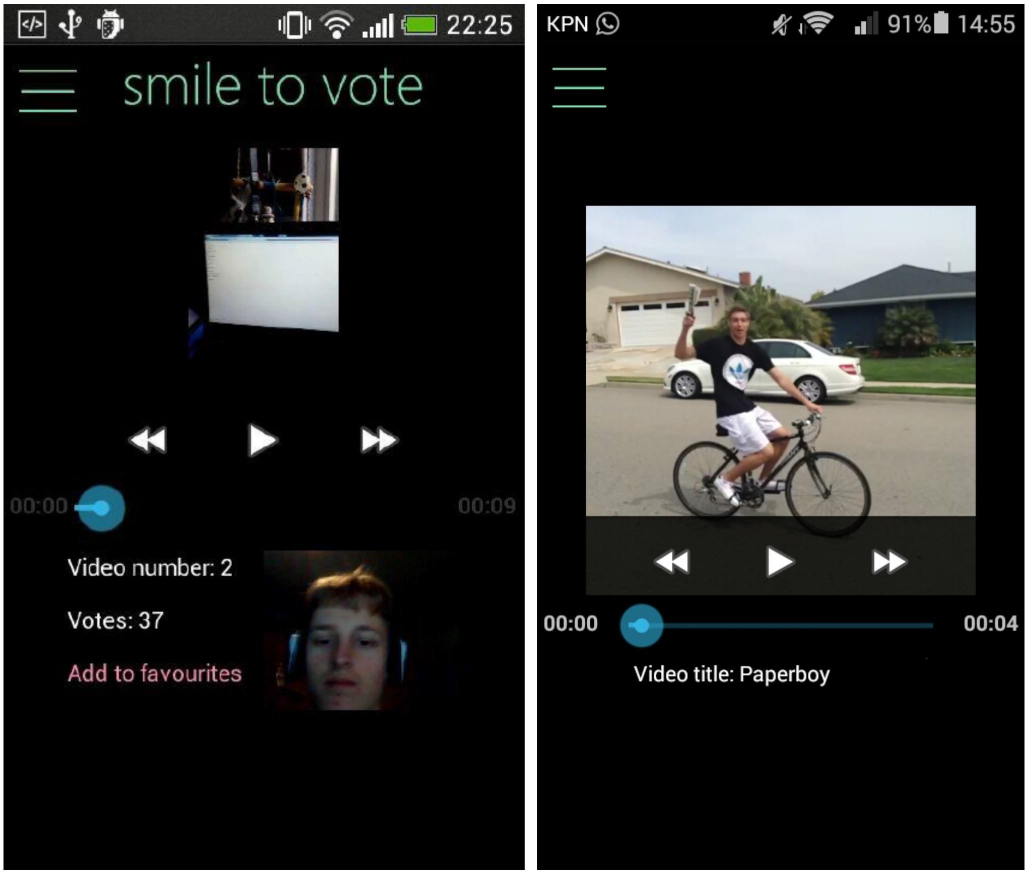


Figure 5: Overview of the communication network.

4.1 Application

In this project we started with an application that already had a few things ready, described in [1]. In short the prototype embodied an android application that could play videos from the SD-card while using the front camera to check if someone was smiling or not and if someone smiled a picture was taken and saved on the SD-card as well, Figure 6. So at this point we already had the Snapdragon SDK algorithm implemented in our application for the smile recognition. At first we wanted to expand this prototype to a prototype with the same functionality but working online rather than with the local SD-card. To accomplish this we first installed a server to store the database and videos we are using in our experiment. Next we made an overview of the database we would need to store all the data to run the application and remember which videos are already watched by which user. We put this overview in an ER diagram of our database, Figure 7. After that we needed a way to communicate between the application and the database in a safe way. For that purpose we chose a node.js server which is further explained in the according section in API's and libraries. At completion of this communication we upgraded the prototype to an application that had almost everything we needed as it could stream online videos, recognize a smile, keep track of all users and store the results. At this point we decided to omit the redundant features that we described in the application design section which resulted in the final application. A screenshot of the user interface and how the communication network used was implemented for our purposes are shown in Figure 6 and Figure 8. Finally we finished our setup by making a website to distribute the application to potential users [7].



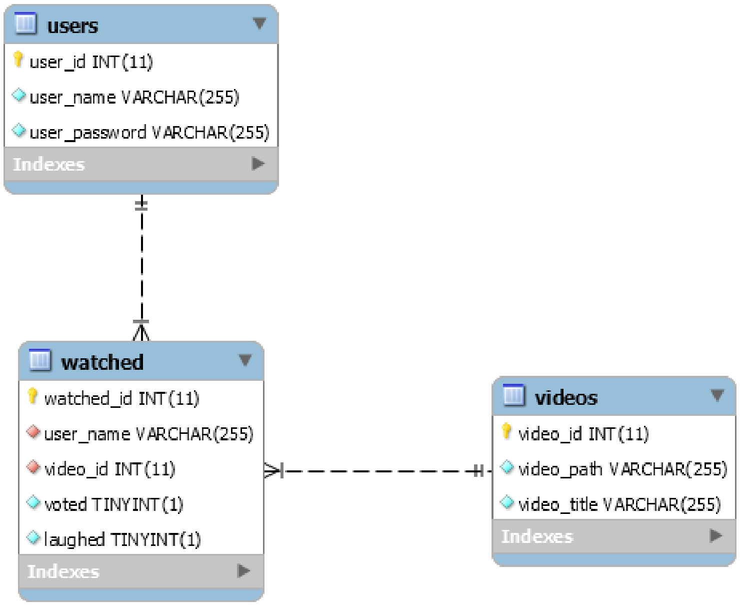
Video number: 2

Votes: 37

Add to favourites

Video title: Paperboy

Figure 6: Old user interface (left) and user interface used in this research (right).


users

- user_id INT(11)
- user_name VARCHAR(255)
- user_password VARCHAR(255)

Indexes

watched

- watched_id INT(11)
- user_name VARCHAR(255)
- video_id INT(11)
- voted TINYINT(1)
- laughed TINYINT(1)

Indexes

videos

- video_id INT(11)
- video_path VARCHAR(255)
- video_title VARCHAR(255)

Indexes

Figure 7: ER Diagram.

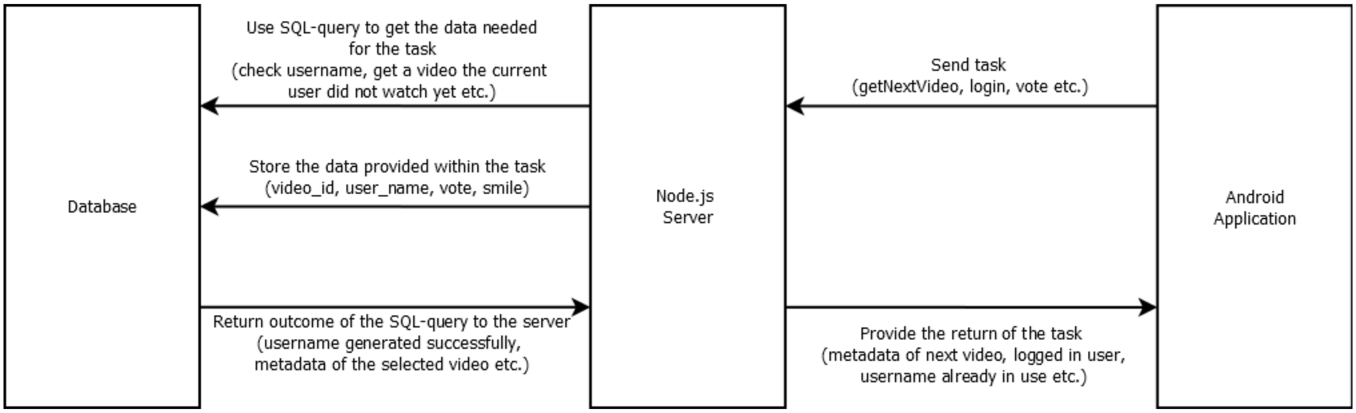

Database
Node.js Server
Android Application
Use SQL-query to get the data needed for the task (check username, get a video the current user did not watch yet etc.)
Store the data provided within the task (video_id, user_name, vote, smile)
Return outcome of the SQL-query to the server (username generated successfully, metadata of the selected video etc.)
Send task (getNextVideo, login, vote etc.)
Provide the return of the task (metadata of next video, logged in user, username already in use etc.)

Figure 8: The implementation of our communication network.

4.2 Experiment

After finishing the setup and desgning the experiment it was time to actually implement the experiment. At first we tested if everything worked correctly with only a couple of videos on two different mobile phones, one with sufficient specifications and one without sufficient specifications. When everything seemed to work correctly we expanded the experiment to our 14 videos and tested if everything still worked on both phones for a last time. When everything worked correctly we distributed our website over Facebook and Skype and kept track of the results while they were collected by accessing our database once in a while. To make sure the participants used the application as we designed it we made clear directions and an installation instruction page on our website. Moreover, after finishing the experiment and explaining what it was about, we asked them to not discuss it with others in case the others wanted to participate as well.

5 Results

As discussed we tried to reach as many potential users as possible and collected data for about two months. In the end 15 people tried to participate, 13 with sufficient specifications and 2 who could not participate because of incompatible hardware, described in the materials and methods section. Although this does not seem much, probably because it was during the summer, we gathered data from almost 200 watched videos because every user watched 14 videos. In this section we will review and discuss our final application and the data we gathered in our experiment.

5.1 Application

Everyone who used the application was able to finish the experiment by themselves completely. Only a few users were doubtfull about which username to choose, but that did not affect the results of the experiment. So the application seems to have worked fine to gather the results in the experiment. The only problem seemed to be the effort it would take to participate. Although we tried to reach a lot of people who were capable of participating, going to the website and downloading the application before the actual experiment even started seemed to be too big of a threshold. But even with this few participants we have shown that the application worked as it should and gathered some insight in the comparison between voting by clicking and smiling when watching a "funny" video.

5.2 Experiment

In this subsection we will show the results of our experiment and point out the most outstanding differences. For every video that was watched we gathered two booleans, vote and smile. In the first global comparison, Figure 9, we can see there is quite a difference between the votes and the smiles. In most cases this means there were more votes than smiles for the video, but in a few cases it was the other way around. Then we classify the data in two groups, the people who used their own phone in private to do the experiment compared to the people who used our phone to do the experiment knowing we were somewhere around, Figure 11 and Figure 10. As more people used our phone to do the experiment there seems to be a difference, but when we look at the relation between amount of votes compared to amount of smiles there is no clear difference between these groups seperated compared to all users together. In the end the difference that we notice between votes and smiles can exist because people vote but not smile, but this could be compensated by people who smile but not vote. In this experiment a person who votes but not smiles negates a person who smiles but not votes and the other way around. To get some insight in the actual differences between voting and smiling we compared the four possibilities: no vote and no smile, vote but no smile, smile but no vote, vote and smile, Figure 12 and Table 1. From this we can see that the differences we saw in the global overview were caused by a large amount of votes without a smile. Although a small amount of smiles without vote negated that and also caused the few cases where more smiles than votes exist, the difference is substantial. Figure 12 and Table 1 also show that the votes and smiles do not match in 40% of the entries.

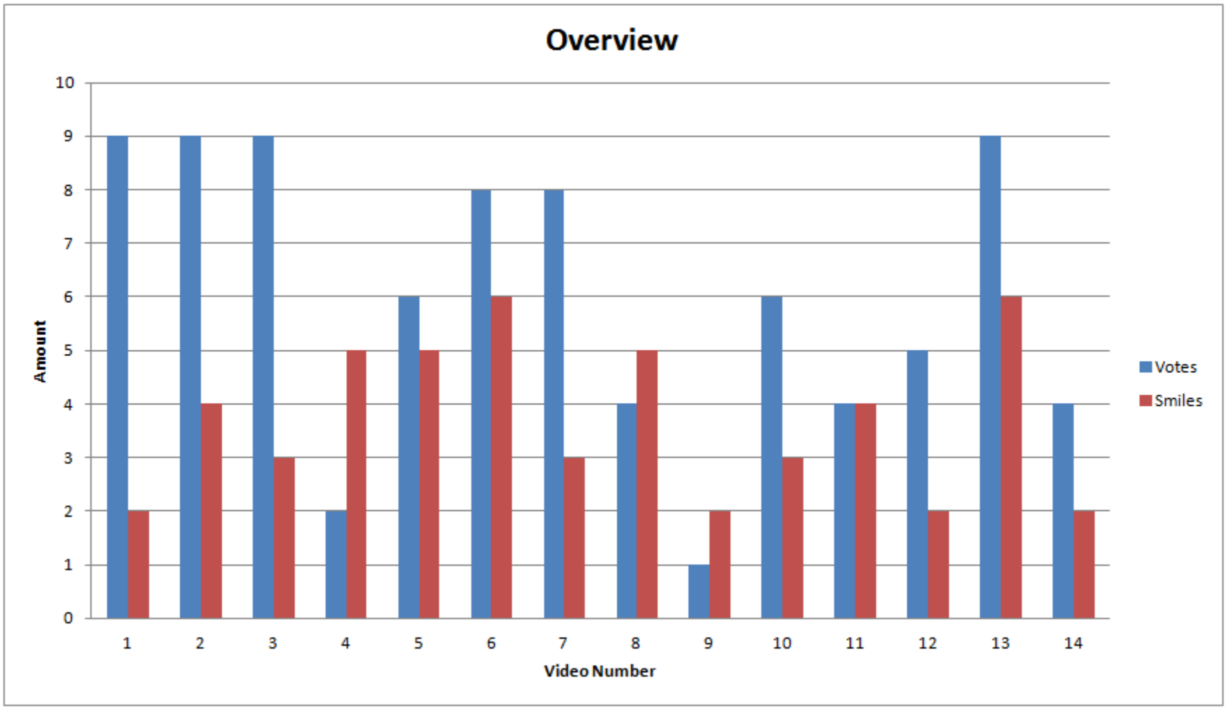


Figure 9: Overview.

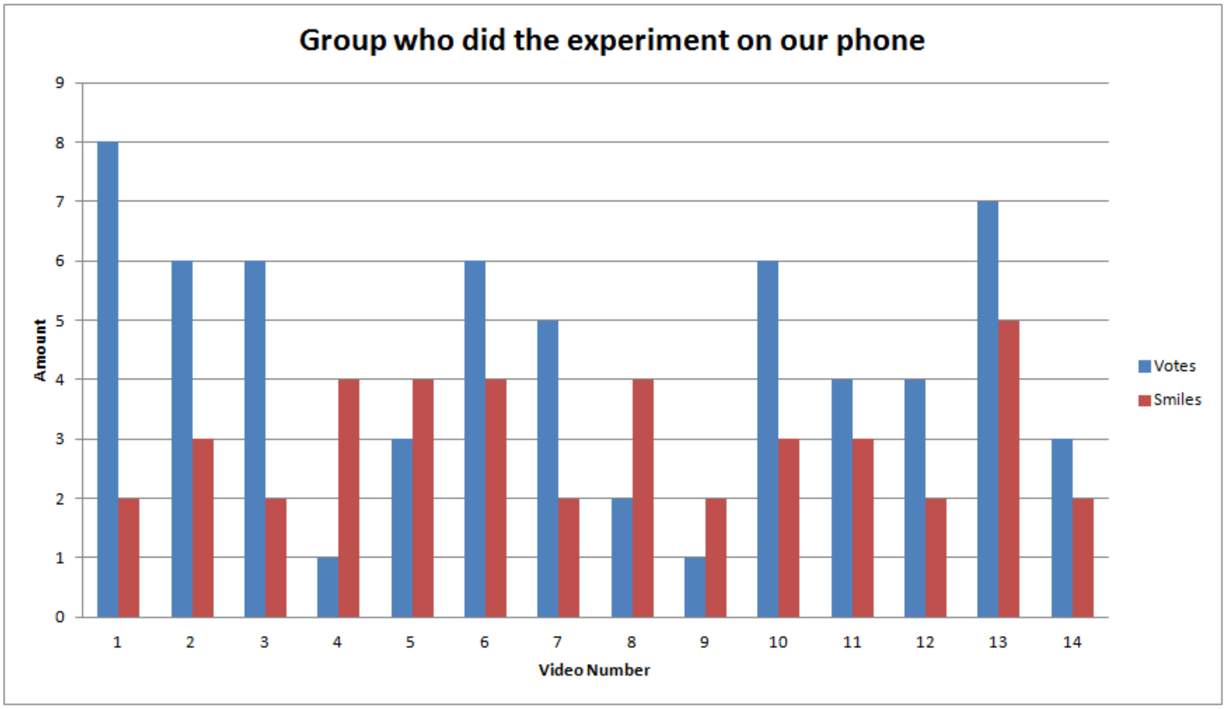


Figure 10: Results on our phone.

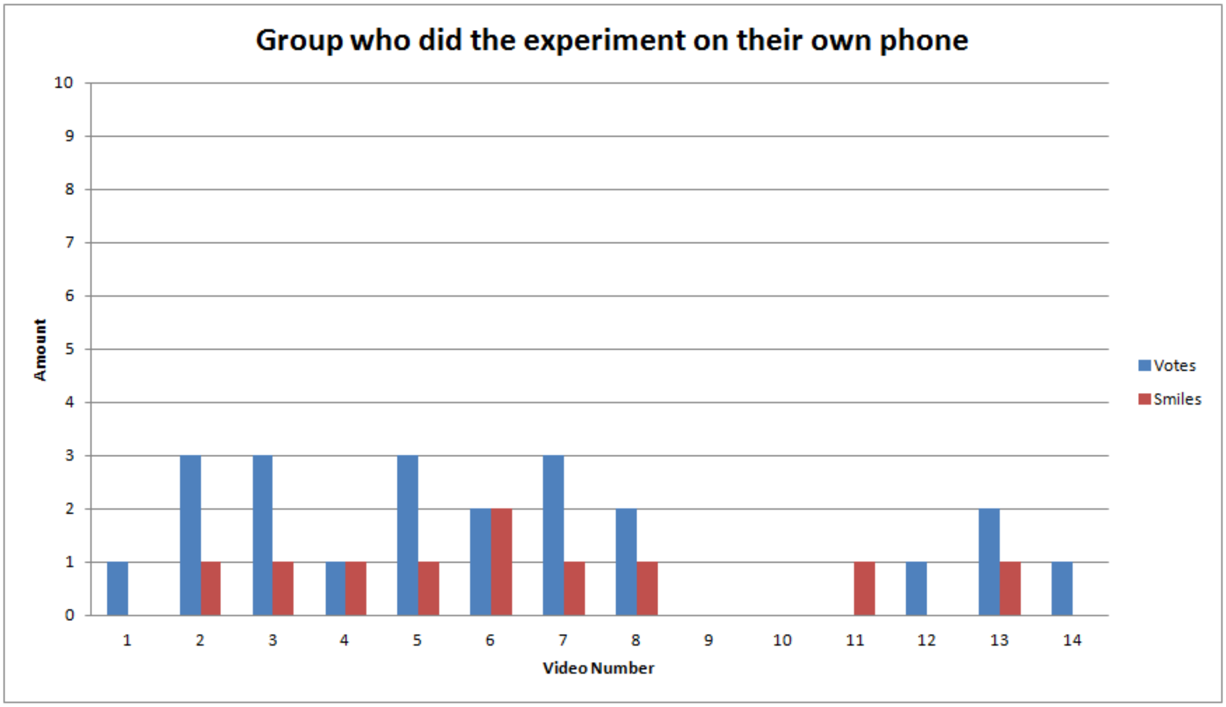


Figure 11: Results on own phone.

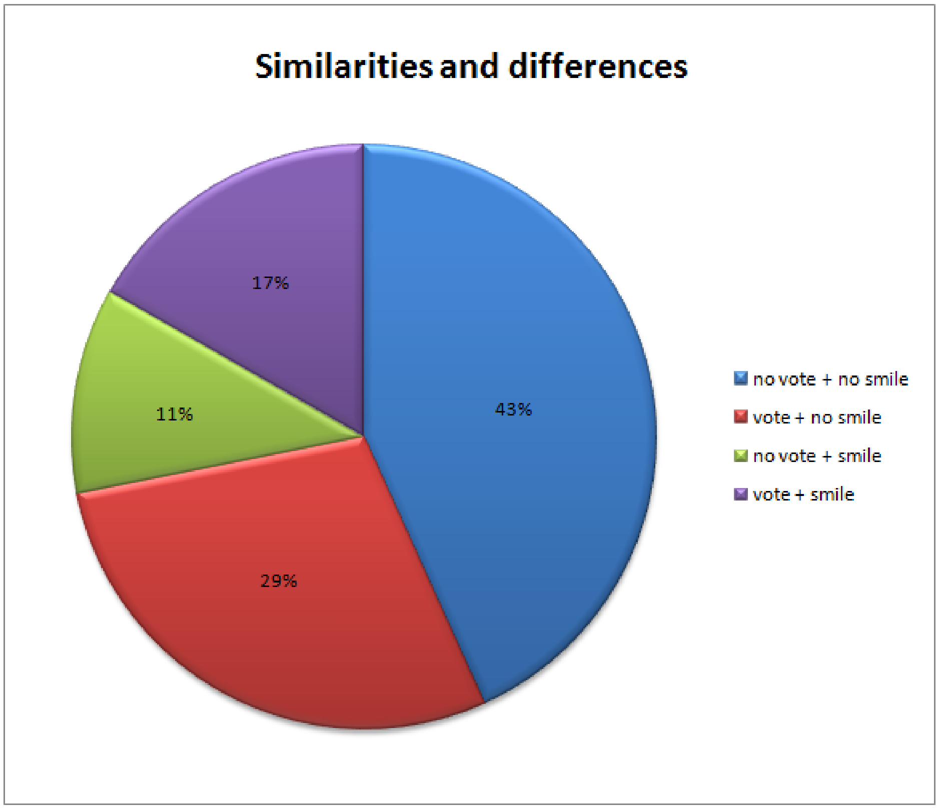


Figure 12: Comparison.

	Smile	**No Smile**
Vote	31	53
No vote	21	80

Table 1: Confusion Matrix.

6 Conclusions and discussion

In this section we will elaborate on what conclusions can be made looking at our results and discuss what could have been done to improve the experiment.

6.1 Conclusions

As described and shown in the results there is a substantial difference in the votes and the smiles. Because the difference is this big, it definitely rules out the error rate in the smile recognition and the possibility of missclicking a vote. The most obvious reason for explaining a vote but no smile is that the user thinks the video is funny, but it is not funny enough that it makes him smile. Another possible reason could be that the user already has watched the video before and still thinks the video is funny, however he does not smile any more this time. A reason for smiling but not voting is a lot harder to distinguish. Since the body language shows the joy while watching the video, it seems weird to not vote for the video. However, in the case of funny movies there also exists a genre of very gentle humor. This can make a person smile, but at the same time think it is not funny. In this case the smile recognition will register a smile, but the user will not vote. In general there is also the possibility someone gets distracted or reacts on something in his environment which cannot be distinguished by the facial recognition from the reaction on the video if it happens while watching the video. However, this amount is in the same order of magnitude as the error rate of the smile recognition, so it does not explain the large difference the results show. If we had a larger group of participants that would participate regardless, we could have a survey after the participation to investigate the reasons behind the difference we found, but as it was already hard to find participants and we did not know there would be such a difference beforehand there was no reason to have such a survey in our research project. In the end the results show that when a user votes for a "funny" video it is hard to tell if he smiled while watching. On the other hand it shows that if a user does not vote for a "funny" video it is unlikely that he smiled while watching the video. Wether a user smiles or not while watching a "funny" video does not seem to have a lot of influence on the voting, although it looks like the more genuine answer seems to have a slight advantage. The results also show that our prototype combined with the backend we made worked perfectly together as there were no corrupt results except the small possibility of errors in the smile recognition.

6.2 Discussion

The concept of this research project is in general the new way of rating a video using a physiological response rather than a thoughtfull action of the user. This way we tried to get a more genuine rating, which should correspond more to the actual feelings of the user. As the group of participants was quite small we could not determine the reasoning behind the differences we found, so it would be better to have a larger group of participants and also give them a survey afterwards to give a better idea of the cause of the results. The outcome of this research however can be looked at from a broad perspective within the field of affective computing. As we now know there is a difference between the current way of rating and rating the affective way, it could be interesting for the current rating platforms to expand their ratings with affective ratings which can be different systems according to the content that is rated. So the idea of rating by a physiological response does not have to replace the current way of rating completely, because combined with the other ratings you get just more and more information about what the content actually represents. For example for our video rating it could be interesting to have the current rating system to show how good the video is and an affective rating system to show what the users feel when they watch the video. On the other hand for a more commercial purpose like designing advertisements it could be interesting to see how people respond to the advertisement and what they feel like when they watch it and adjust the advertisement in a way it matches the product the advertisement is for. Ofcourse previously advertisers are also playing on the emotions of people, but it was always designed by asking people how they felt rather than measuring the physiological response. These are just two examples of using affective computing and a physiological response as a rating system, but we can see it working for every rating system that is related to the feelings of people. In the far future the visible physiological response maybe is not even needed and the affection can be measured by brain activity as research shows that specific emotions are related to activity in specific parts of the brain [8].

References

[1] Smile To Vote: designing a smile based rating system, Jeroen van Oorschot and Terry Zabel supervised by David van Es and dr. ir. Fons J. Verbeek.

[2] Yu-Hao Huang and Chiou-Shann Fuh, Face detection and smile detection, (2015). [online] Available at: http://www.csie.ntu.edu.tw/ fuh/personal/FaceDetectionandSmileDetection.pdf [Accessed 26 Jul. 2015].

[3] AffectButton: a method for reliable and valid affective self-report, Joost Broekens and Willem-Paul Brinkman.

[4] Qualcomm Developer Network, (2010). Snapdragon SDK for Android. [online] Available at: https://developer.qualcomm.com/software/snapdragon-sdk-android [Accessed 26 Jul. 2015].

[5] GitHub, (2013). Gottox/socket.io-java-client. [online] Available at: https://github.com/Gottox/socket.io-java-client [Accessed 26 Jul. 2015].

[6] O. Deniz, M. Castrilion, J. Lorenzo, L. Anton and G. Bueno (2008) Smile Detection for User Interfaces. Advances in Visual Computing, Part II, LNCS, 5359 p.p. 602-611.

[7] Liacs.leidenuniv.nl, (2015). Terry Zabel. [online] Available at: http://liacs.leidenuniv.nl/ tzabel [Accessed 27 Jul. 2015].

[8] Subcortical and cortical brain activity during the feeling of self-generated emotions, Antonio R. Damasio, Thomas J. Grabowski, Antoine Bechara, Hanna Damasio, Laura L.B. Ponto, Josef Parvizi & Richard D. Hichwa.