FaceTouch: Machine Learning Approach to Detect Face Touching Events

Johan Krause1[1959166], Sebastian Ebi1[2068091], Huidi Zhu1[1984436], and Calvin Antonius Tanama1[1813569]

Karlsruher Institut für Technologie, Kaiserstraße 12, 76133 Karlsruhe, Germany

Abstract. Virus infections can be perpetuated by skin contact especially through the habit of face touching with potentially unhygienic hands. In this paper we present a design and evaluation approach for a face touch detection system using machine learning models and smart glasses. These models were built based on collected data by J!ns Meme Smart glasses. Our experiment includes the collection of balanced natural and training data, data pre-processing for matching data and labels with different sampling rates and subsequent model training based on time series analysis. Using data from 20 respondents, we then conduct the evaluation of these models using several defined metrics. In general, the accuracies of these models are not higher than 80% due to the sensors being noisy and unsuitable to an extent. Further, in the long run the performed online boosting approach is more accurate than that of the utilized long short term memory network, but a generalization of this may prove insignificant due to longer training times being required.

Keywords: Machine Learning, Wearable Computing, Deep Learning, Recurrent Neural Network, Online Boosting, Long Short Term Memory

1 Introduction

In times of COVID-19 global pandemic a generic protocol has been developed to prevent respiratory infections. One prevention stated by these protocols is to avoid the critical face area such as eyes, nose and mouth being touched. Kwok et al. stated that the observed students studying in a library touched their face 23 times per hour on average [1]. However the number of face touches can be a lot higher than in the observed scenario especially in the case of people with face touching habits.

Wearable computing is an emerging research field in computer science and also many prototypes and wearable products are developed and produced in the industry. One of the prominent results are smart glasses. With smart glasses equipped with sensors of different kinds it is even possible to detect physical activity and evaluate some metrics from it.

With the ongoing global pandemic, promising potentials of smart glasses and the effectiveness of machine learning models, following research questions are coming up: Are smart glasses with skin sensors suitable for detecting face touch events especially in the critical facial areas? Which machine learning approaches are the best for handling face touch detection problems and subsequent challenges? How and using which metrics can we measure the quality of these approaches?

In this paper these research questions are answered by providing information on detailed approaches that are undertaken to solve the problem. From data collection including experimental settings and data pre-processing to model training and model evaluation. A long short term memory network (LSTM) and an Online Boosting model are chosen and evaluated, because of their ability to process time series analysis. Furthermore, it's stated towards the conclusion of the paper which improvements could be made regarding the models and the beforehand pre-processing. Along with detailed descriptions of the experiment and model training, fellow researchers can recreate the process and hopefully develop it further.

1.1 J!ns Meme smart glasses

In this project J!ns Meme smart glasses were used to collect raw data from respondents. J!ns Meme smart glasses have a gyro sensor, motion sensors and electrooculography (EOG) sensors. The specifications of the glasses are stated in the table below.

Table 1. Specification of J!ns Meme EOG sensors

Resolution	12bit
Range	-1500 to 1500 μV
Sensitivity	1.37 LSB/μV
Electrode Materials	Stainless Steel (SUS316L)
Sampling Frequency	100 Hz (full), 200 Hz (standard)

J!ns Meme smart glasses were connected via Bluetooth to an Android app called Meme Logger. Meme Logger is a data collecting app developed by the internal J!ns Meme team to test whether the sensors detect impulses and then to map these into visual depictions. The impulse data can also be recorded into a csv file. To use the glasses with a Microsoft Windows environment directly, a separate dongle is required, which was not available to us for this project.

1.2 Anvil

Anvil is a free video annotation tool developed by Michael Kipp from University of Applied Sciences Augsburg [2]. Anvil provides frame-by-frame video annotation with a frame rate of 24 frames per second in the present case. One of the biggest advantages of Anvil is the ability to configure user-defined annotations. In this paper 4 annotations were defined: left eye, right eye, nose and mouth. These denote 4 critical face areas. Furthermore, Anvil also provides export annotation functions to several file types, for example html, csv or txt with different separator characters. This feature is very important for the data pre-processing.

2 Related Works

In recent study, Bulling. et al aimed at recognising mobile activity such as reading and writing in indoor settings. Using EOG sensors and SVM classifiers, the results yielded average precision values of 76.1% and average recall values of 70.5% [3]. In related works, Ishimaru. et al extended the research to a commercial grade smart glasses and to more mobile activity, for example eating and talking to other persons. This work also incorporated SVM models and yielded an average correct classification rate of 71%.

Inspired by application of deep learning models in the field of computer vision [4], this research is dedicated to evaluating deep learning approaches to the detection of face touch events. Face touches have different properties from other activity presented by previous works. Namely, they do not involve eye movement and can also triggered by natural reflexes of human body. In addition, the robustness of EOG sensors is also evaluated for impulse detection.

3 Evaluation Metrics

The present use case of face touch detection is a typical data classification problem where evaluation metrics are applied in the training phase as well as in the test phase.
In the training phase the evaluation metrics are used to improve the classification algorithm. On the other hand, in the testing stage, the evaluation metrics are used to evaluate the effectiveness of the used classification on unknown data. [7]

3.1 Confusion Matrix

In the case of a binary classification problem, as seen later in this application, the results of the predictions of the classifier during the training of the model can be displayed using a *Confusion Matrix.* The rows represent the current labels, while the columns show the predicted labels. In this confusion matrix, *TP* (True Positive) and *TN* (True Negative) denote the number of positive and negative data points that were correctly classified. In contrast, *FP* (False Positive) and *FN* (False Negative)denote the number of predicted data points that were incorrectly classified. From *Fig.1*, commonly used evaluation metrics for evaluating the performance of the model can be calculated with different focuses. [7]

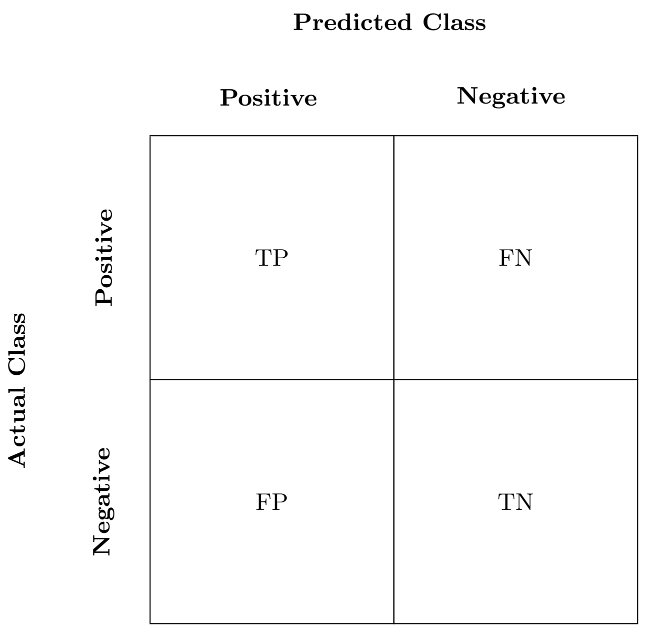

Fig. 1. Confusion Matrix for Binary Classification

3.2 Accuracy

The most commonly used evaluation metric is the accuracy metric. The accuracy measures the ratio of correct predictions over the total number of data points evaluated. It is used to determine the quality of predictions based on percentage of correct predictions over total instances. The range of values is between 0 and 1, with the higher the accuracy score, the better.

$$acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

The advantage of accuracy is that this metric can be calculated easily and with less complexity. It is easy for humans to understand and easy to apply. In addition, this evaluation metric is equally applicable to multi class problems.
However, these advantages are only available if the classes are balanced, i.e. if the data set contains approximately the same number of observations of class 0 and class 1. If one of the classes is under- or over-represented, then the Accuracy is not a robust metric. [7]

3.3 Precision

Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class. The range of values is between 0 and 1, with the higher the precision score, the better.[7]

$$p = \frac{TP}{TP + FP} \tag{2}$$

3.4 Recall

Recall is used to measure the fraction of positive patterns that are correctly classified. The range of values is between 0 and 1, with the higher the recall score, the better. [7]

$$r = \frac{TP}{TP + TP} \tag{3}$$

3.5 F1-Score

This metric represents the harmonic mean between recall and precision values. The range of values is between 0 and 1, with the higher the recall score, the better.[7]

$$F1 = \frac{2 * p * r}{p + r} \tag{4}$$

4 Data Collection

The data for this project were collected from 20 respondents. There are no age, gender and body metrics restriction to ensure robustness and reduce bias. All the action recorded are in the sitting position in indoor setting.

First the respondents were required to wear the J!ns Meme smart glasses. The sensors then activated through connection to Meme Logger. To avoid unintended turbulence, the respondents were needed to stay still for 5 seconds before the recording started.

After the respondents were still, the recording were started. There are two recordings that were taken, the impulse recording from smart glasses and video recording for data labeling. Because of inconsistency in the starting point of recording from smart glasses and video, the respondents were prompted to touch their nose 3 times rapidly in the beginning of the recording and at the end of recording. This served as flag for data pre-processing, especially when the annotations are applied to the impulse data. These flag was also removed in the data pre-processing to avoid redundant.

4.1 Natural Data

Natural data is data collected from the respondents in natural settings meaning that the respondents were free to do activity such as learning for exam and doing paperwork while recorded as long as the activity did not violate the general condition e.g. indoor and in sitting position.

The main purpose of this type of data is to provide test data for our machine learning model in real world situation where no constraints were given.

This data collecting lasted one hour.

4.2 Training Data

Training data is data collected from the respondents in order to train the model. The most important aspects that this type of data should achieve were quantity, balance and variation of the impulse.

As stated, there were 4 critical face area, which represented by 4 labels: left eye, right eye, nose and mouth. For each data collecting the respondents are prompted to touch each critical face area 10 times, thus resulting 40 touches per run. The sequence of the instruction was randomized to ensure that the model did not learn specific pattern of sequence. Furthermore the duration of an instruction was also randomized between 7 until 13 seconds.

Algorithm 1 Protocol Prompter

```
1: procedure PROMPT(zone, $n$)    ▷ zone: list of critical zone in string, n: number of repetition
2:     total ← zone
3:     for $i \leftarrow 2, n$ do
4:         total ← $append(total, zone)$
5:     end for
6:     $rand(total)$                ▷ shuffle entries in the list total
7:     for $i \leftarrow 1, total.length$ do
8:         time ← $rand(7, 13)$
9:         for $j \leftarrow 1, time$ do
10:            $print(j, total[i])$
11:            $timer(1000)$        ▷ 1 second in machine clock
12:        end for
13:    end for
14: end procedure
```

Based on *Algorithm 1*, a prompter program was developed with graphical user interface. In the graphical user interface a suitable image and remaining time were shown for each critical face area. In order to smoothing the transition between two instruction, a contrast warning image was shown for the last three seconds of each instruction. The whole run lasted at most 10 minutes.

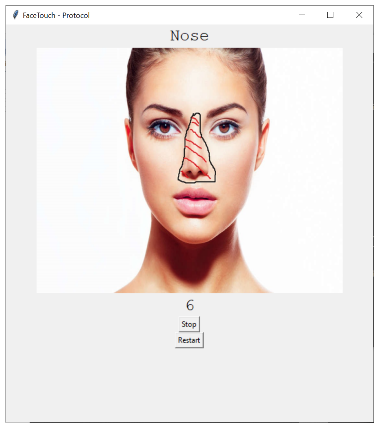

Fig. 2. GUI of the protocol

5 Data Pre-Processing

As a result of the previous step, the data collection, one obtains two separate CSV-files, one with the recordings of the J!ns Meme Glasses and one with the exported labels of Anvil. These files have a different size due to the fact that the recordings of the glasses and the labels were recorded in a different frame rate. The J!ns Meme glasses use a frequency of 50 Hertz, while Anvil only supports a frame rate of 24 frames per second. As a result, the record of the recordings of the glasses is slightly more than twice as long as the record with the labels from Anvil. Furthermore, the recordings are not started at the same moment because two different devices have to be used for this, which is why the peaks in the data and the labels are not synchronized.

The data records from the J!ns Meme smart glasses contain the data from the EOG sensors, the gyro sensor and the accelerometers as well as the timestamps. Since only the EOG sensors provide relevant data for the detection of face touch events, only their data is used for pre-processing and training of the models. The Anvil label file consists of several columns, with a separate column for each specific touch zone and one for the timestamps. Only the relevant columns are used here, namely those for the nose zone, mouth zone, right eye zone and left eye zone as well as the timestamps.
Data pre-processing is crucial in order to obtain meaningful and valid results. Before applying any classification algorithm, the data need to be cleaned and formatted into an appropriate format.

In the first step in pre-processing, the first peak, triggered by the synchronization gesture, must be found in the data of the J!ns Meme recordings, which is usually clearly visible in the plots, as you can see in *Fig. 2.* This has to be done manually for each recording, because not every peak in the data is triggered by a face touch event, so the timestamp of the first event cannot be found automatically.

Next, the label data will be searched for touch events and a list of labels will be created with ’0’ for no touch, ’1’ for nose, ’2’ for mouth, ’3’ for right eye and ’4’ for left eye. This is then appended to the original label data frame. All labels are cut off before the first touch label.

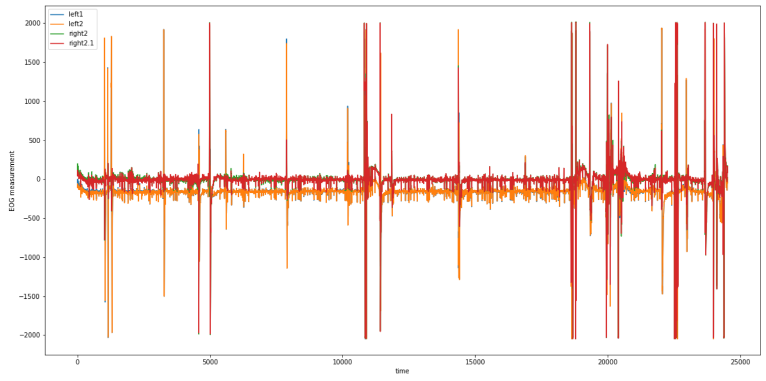

Fig. 3. Plot of EOG sensor data

Afterwards, the timestamp found in the first step from the recording of the glasses for the beginning of the first nose touch of the synchronization gesture is set as the first timestamp of the label data and all subsequent timestamps are recalculated. This is necessary to be able to merge the labels with the sensor data.

To merge the recordings with the labels, a comparison of the time stamps of the two data sets is necessary. A distinction must be made between the two cases. If the time stamp of the recordings is smaller than that of the recordings, the system skips to the next time stamp. However, if the time stamp of the recordings is greater than that of the label, the system checks whether the time stamp of the recordings is smaller than the next greater time stamp of the label data. Then the label is transferred from the label data frame to the recording data frame. This makes it possible to scale the labels to the length of the J!ns Meme record. *Algorithm 2* shows in detail the algorithm for merging the J!ns Meme recordings with the label data.

Algorithm 2 Join Data Label with Recordings

```
Inputs:
    labels := table with labels ($T \times 2$)
    recordings := table with sensor data ($N \times m$)
1: $i = 0$
2: // Adding a column with zeros to recordings table
3: recordings.addZeroColumn('label')
4: for $t \in \{0, 1, \ldots, T-1\}$ do
5:     $tstp_{label}$ = labels[t].timestamp
6:     while i < N do
7:         $tstp_{recording}$ = recordings[i].timestamp
8:         if $tstp_{recording} < tstp_{label}$ then
9:             $i = i + 1$
10:        else if $tstp_{recording} \geq tstp_{label}$ then
11:            if $tstp_{recording} < labels[t+1].timestamp$ then
12:                // Adding label to the recording
13:                recordings[i].label = labels[t].label
14:                $i = i + 1$
15:            else
16:                // do nothing
17:                break
18:            end if
19:        end if
20:    end while
21: end for
22: return recordings
```

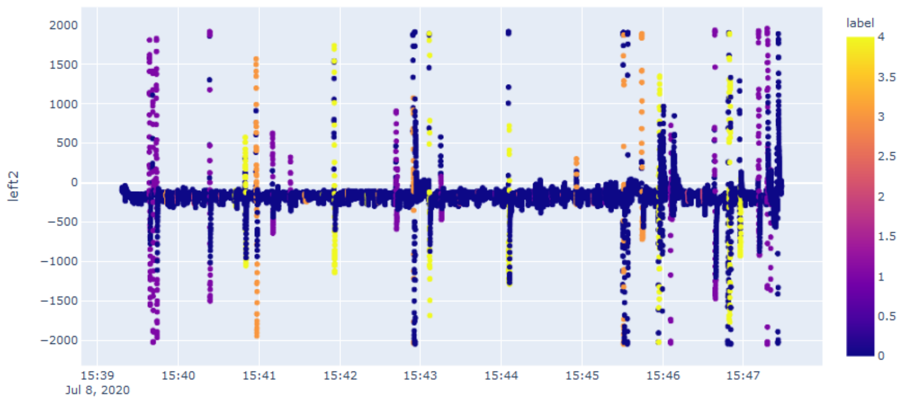

Fig. 4. Plot of merged data set for EOG sensor 'left2' with labels

In the last step of data pre-processing, the synchronization gestures at the beginning and end of the recording are cut off. These would influence the result of our classifier, as it searches for regularities in the data. Since these three nose-grasping gestures are unnatural, they are not intended for the training of our models.

Since the synchronization gesture was labeled with three individual nose touches each, these can be cut off automatically. The label column of the data record is counted up once from the front and once from the back after the label '1' for a nose touch and an index for each area with this label. If the index reaches the value 3, the index for the current line in the data frame is returned. Then all lines up to this index are cut off.

These data pre-processing steps must be performed for each individual recording data set.

6 Models

In this section, we briefly introduce the two models that are applied in this work. Namely, the LSTM and Online-boosting models.

6.1 Long-Short-Term-Memory Network

Following data collection and extensive pre-processing steps, the initial system to be constructed and trained in order to classify face touch events based on the glasses' sensor data was chosen to be one of a modern approach. In accordance with current research trends, a type of Long-Short-Term Memory (LSTM) network was selected. As a type of recurrent neural network architecture, LSTMs are thought to be suitable for sensor data tasks since the input sequence plays a key role in their functioning. This means the time series structure of the recorded sensor data would be kept as valuable information, as the input data stream is auto-correlated to its previous values.

Specifically, recurrent neural networks (RNN) are a type of artificial neural network that suggests the abstracted simulation of neural activity in the human brain. RNNs incorporate the so-called back-propagation to pass on output information from neural network layers on to previous layers, hence enabling the model to learn from preceding time steps. In addition, the LSTMs are a sub-type of the RNN class as they furthermore contain cell states in the layer's nodes. These cell states serve a memory function for the LSTM and are regulated through an input, output and forget gate for each node. In particular, the gates control which information is kept in the cell state and therefore manage the memory of the corresponding node. This model design is thought to be suitable for learning dependencies and trends in sequential data sets.

One major requirement for the application of LSTM models to the present task is that the input data dimensionality is required to be of equal shape for each sample. The samples that were fed to the model during training needed to have a fixed number of data points and extraction of sample slices from the original time series was necessary for each of the four sensor data streams. As the smart glasses recorded on a 50 Hz frequency, this resulted in a data point every 0.02 seconds. In order to capture face touch events and both a significant amount of previous as well as following information, the size of a slice was set to 200 time steps. Accordingly, one time step included 200 sequential data points covering a total time frame of 4 seconds. The duration of face touching activities was beforehand analyzed visually using plotted and labeled sensor data, resulting in the choice of the mentioned time frame. Slices of this duration were extracted from all recorded training data with a 95% overlap each. A sliding window with an overlap to the previous window can be thought of as a visualization of this process.

Regarding the input space of the LSTM model, roughly 260000 time steps were gathered from all the recorded, manipulated short data samples. After extracting the partitions of fixed length with overlap from these samples, around 25000 slices with 200 time steps and four features each. Every sensor in the smart glasses served a slightly different data stream, producing a 4-D input for one time step. For training, a split of 0.95 was utilized, resulting in around 24000 training samples and 1000 samples for testing. This may seem a small test set at first, however the afterwards evaluation was also carried out based on the longer recordings in natural behavior, i.e. the clips of around 60 minutes length with non-artificial face touch events. Therefore the training set was characterized by 24000 x 200 x 4 dimensions.

The output of the model was designed in such a way that the system predicted the probabilities for the binary case: face touch = true or face touch = false. We chose to focus on the binary case only and to not consider different types of face touches in varying facial zones, as this was thought to reduce the complexity of the model and its tendencies to overfit by learning too specific touch activities. Regarding the research question and the challenge posed to recognize facial touches in general, this approach therefore seems legit. One could think of classifying different types of touches using the labeled data in future research works, as suggested in the concluding chapters.

The neural network consisted of several concatenated layers that were implemented using the Keras framework in Python 3. Following the input layer, the LSTM layer was placed with 100 units and an input defined in the shape of 200 x 4. Additionally, the return of the sequences was set to true to preserve the chronology of the time steps. We then connected each of nodes in the LSTM layer with each of the nodes in the subsequent dense layer with 50 nodes. This dense layer was activated by the rectified linear unit activation function ("relu"). The relu function has several beneficial characteristics, one of which is the relatively efficient computation. Next, a dropout layer with the dropout probability set to 0.5 was added. The utilization of dropout layers is common practice to effectively reduce overfitting [5]. The last layer was responsible for producing the output probabilities for the two possible events: "occurrence of face touch" or "no occurrence of face touch". As these two options are mutually exclusive, the softmax activation function is suitable for such cases as all its outputs always sum up to 1.0.

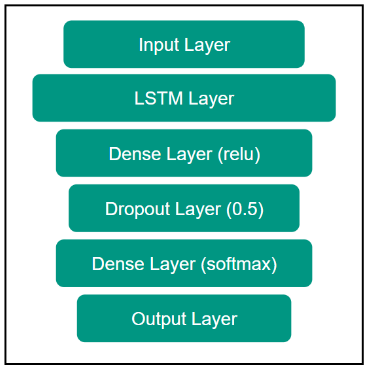

Fig. 5. Structure of the LSTM Model

When compiling the sequential layers to a model, the adaptive moment estimation optimizer ("Adam") with a learning rate of 0.0001 was applied. Further, the loss function implemented was that of the categorical crossentropy for minimizing the model's losses during training. Overall, the system incorporated roughly 47000 trainable parameters which were to be trained in the following steps.

Once training was finished, evaluation of the system was carried out. On the previously extracted test set the accuracy was computed, with a score of 0.929. While this can be considered a high score, additional investigation into the predicted labels quickly revealed that the model optimizes this score by prediction the more frequent event. That is, the "no face touch" event. Combined with the imbalanced data set at hand, this produced highly accurate results, however this was not the case for the more relevant event. Therefore the recall and F1 score were calculated to gain more insights into the models potential prediction bias.

For the face touch event, the recall score was 0.21 and the F1 score was 0.2. For the no face touch event, the scores were significantly higher. More specifically, recall was computed to be 0.94 and the F1 score 0.95. This can be seen as a set of imbalanced results due to the training on imbalanced data. Further, a visual evaluation of the model's performance on data recorded naturally, i.e. the longer data sets, the model did classify some of the EOG peaks as face touch events. However, a remarkable amount of data points were classified to be face touches that did not involve any peak in the streamed data. Perhaps the model learned to recognize face touches in a regular manner with similar time frames in between the face touches from the artificially manipulated training data sets. Due to this and in order to achieve more reliable and less biased results, the application of another approach to the classification problem was then chosen.

6.2 Online Boosting

As mentioned before, an unignorable challenge we are facing is the imbalanced data streams. The majority class is non-touch event, whereas the touch event is the one of greater interest. Because a misidentifying of a negative event in our case is not that harmful comparing to miss a positive one. As the Conventional machine learning algorithms optimize the objective by minimizing the overall error rate, which implicit a equally costs on all misclassifications, it is usually difficult to learn from such imbalanced class problems. Thus, by handling this problem in the context of learning from data streams, the online ensemble learning algorithms with embedded cost-sensitive framework are particularly effective[6].

In this work an online-boosting classifier is applied. It is the online version of AdaBoost, which improve the performance by focusing more on difficult instances, namely the incorrectly classified samples during the current iteration. The base estimators of the online-boosting are 10 k-NN classifier. Averaging the outputs of several classifiers helps it to reduce the variance component and/or the bias in the classification error. It simulate the approximation of Poisson distribution by training each arriving samples multiple times. An ADWIN change detector (ADWIN: Adaptive Windowing) is added to better adapt the learning algorithms in case of concept drift. The implementation is based on the scikit-multiflow API.

Regarding the huge computation cost, the 4 EOG signals includes horizontal and vertical both for left and right eyes provide little extra information as they are highly correlated to each other. Before training the model, the dimension is reduced from 4 to 2 by fitting the PCA (Principal Component Analysis) with standardized training data with a sum of explained variance ratio over 99%. The training is executed with a warm start and partially fitted with the stream generator. The performance is evaluated by comparing the predictions of the unseen test data set with its target value set.

As shown in the CM (Confusion Matrix) Figure 6 , the hit rate of touch class increases while the non-touch decreases. Scores including accuracy, precision, recall and F1-score are calculated as well and shown in Table 2 below. Considering the high ambulance of the data set, the weighted scores in Table 3 would be much more suitable. Intuitively, a slice of the test results are shown below in Figure 7 .

Table 2. Table of the test scores

Accuracy	Precision	Recall	F1
0.66	0.19	0.40	0.25

Table 3. Table of the weighted test scores

Precision	Recall	F1
0.76	0.66	0.70

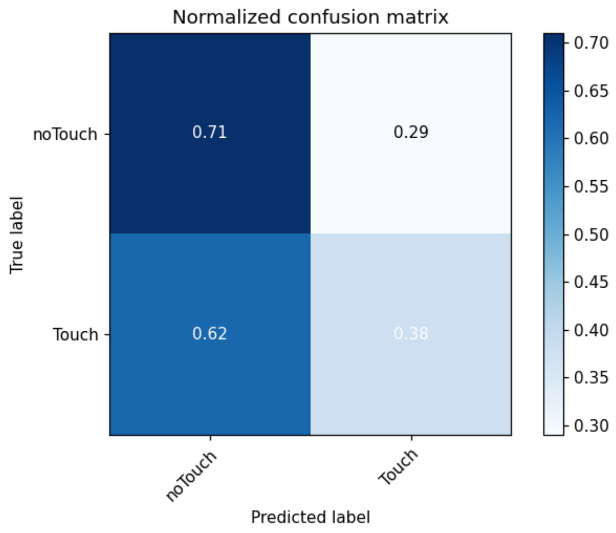

Fig. 6. Confusion Matrix of the test results via Online-Boosting

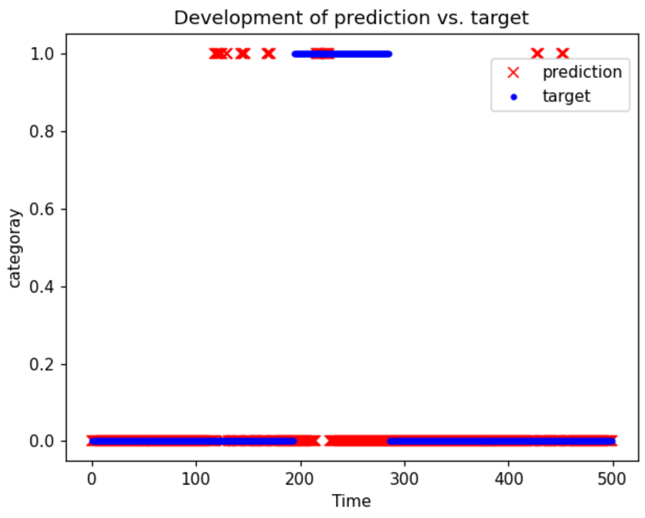

Fig. 7. A slice of predictions of test data via Online-Boosting

7 Conclusion & Outlook

The established models shown preciously utilized different approaches and produce considerably differing results. Obviously, neither the LSTM nor the online-boosting is a perfect model.One main pain-point is the imbalanced data set. Due to the deficiency of time and the limitation of computer power, we could not apply all possible models. Therefore we recommend other relevant techniques or and models here for further works.

As described before, the Jins MeMe smart glasses is equipped with 3-point EOG sensors to detect eye movements. The Blind signal separation (BSS) which is referenced in applications in the filed of biomedical signal processing of Multi-sensors could recover the original component signals from a mixture signal with FastICA. As the the PCA did not take the characteristic features of time series into account, the Singular Spectrum Analysis (SSA) would be more appropriate. It is applied in processing biomedical signals like Electroencephalogram (EEG)) to eliminate high-amplitude artifacts and suppress noise contributions or extract informative components.

Besides the online-boosting we chose in this work, there are several online-ensemble learning models. One special class that must be mention is the cost-sensitive online-boosting and -bagging. Bagging, also known as bootstrap aggregating obtains the diversity via the resampling procedure. The class of an unseen sample is predicted by the majority or weighted vote. The costs of misclassification of cost-sensitive methods are treated differently via biased resampling/reweighting of the data before each iteration. Some popular online cost-sensitive ensemble learning algorithms are Online UnderOverBagging, Online SMOTEBagging, Online AdaC2, Online CSB2 and Online RUSBoost, which are also avaliable in sckit-multiflow.

7.1 Next Steps

This seminar paper provides a solid basis for the classification of face touches with Smart glasses, but also offers room for further work.

The quality of the collected data could be improved by introducing more parameters into the experiment for data collection. For example, the protocol Application could not only be implemented to determine which face zone the subject should reach into, but also to define how this action should be performed. This would allow more natural behaviour to be recorded in the data during the experiment, which could optimise classification by the LSTM Network or by Online Boosting.

Furthermore, an extended data pre-processing could serve to reduce the noise in our training data, which could also improve the results of our classifier. For example, a Fourier transform could be used to remove periodic noise from the sensor data, or a Principal Component Analysis (PCA) could be used to better structure, simplify or illustrate the sensor data by approximating by a smaller number of most meaningful linear combinations.

Due to a missing dongle, which is necessary for connecting the J!ns Meme Smart glasses to Microsoft Windows, it was not possible to connect the classifiers with live streaming data. In the next step it would be important to test and optimize them with live streaming data and then implement them in a face touch detection application. In this environment, the integration of gamification could also be considered. With this approach, the users of this app could be motivated to break the bad habit of touching their face through a reward system. This would reduce the likelihood of contracting infections through this way.

References

1. Kwok, Y. L. A., Gralton, J., McLaws, M. Face touching: a frequent habit that has implications for hand hygiene. https://pubmed.ncbi.nlm.nih.gov/25637115/
2. Kipp, M. (2014) ANVIL: A Universal Video Research Tool. In: J. Durand, U. Gut, G. Kristofferson (Eds.) Handbook of Corpus Phonology, Oxford University Press, Chapter 21, pp. 420-436
3. Bulling, A., Ward, J. A., Gellersen, H., Tröster, G. Eye Movement Analysis for Activity Recognition. In IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 33 , Issue: 4 , April 2011)

4. Daescu, O., Huang, H., Weinzierl, M. Deep Learning Based Face Recognition System with Smart Glasses. PETRA '19: Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments, June 2019, Pages 218–226. https://doi.org/10.1145/3316782.3316795
5. Laixiang Xu, Gang Liu, Bingxu Cao, Peigen Zhang, and Sen Liu. 2019. Infrared Target Recognition Based On Improved Convolution Neural Network. In Proceedings of the 2019 8th International Conference on Networks, Communication and Computing (ICNCC 2019). Association for Computing Machinery, New York, NY, USA, 83–87. DOI:https://doi.org/10.1145/3375998.3376000
6. B. Wang and J. Pineau, "Online Bagging and Boosting for Imbalanced Data Streams," in IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 12, pp. 3353-3366, 1 Dec. 2016, doi: 10.1109/TKDE.2016.2609424.
7. Hossin, Mohammad and M.N, Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations", in International Journal of Data Mining & Knowledge Management Process, vol. 5, pp. 1-11,March 2015, doi: 10.5121/ijdkp.2015.5201