Expert Systems With Applications 188 (2022) 116073

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Efficient machine learning approach for volunteer eye-blink detection in real-time using webcam

Paulo Augusto de Lima Medeiros [*], Gabriel Vinícius Souza da Silva, Felipe Ricardo dos Santos Fernandes, Ignacio Sánchez-Gendriz, Hertz Wilton Castro Lins, Daniele Montenegro da Silva Barros, Danilo Alves Pinto Nagem, Ricardo Alexsandro de Medeiros Valentim

Laboratory of Technological Innovation in Health (LAIS), Federal University of Rio Grande do Norte (UFRN), Natal, RN, Brazil

ARTICLE INFO

Keywords:
Alternative communication
Locked-in syndrome
Computer Vision
Machine learning
Deep Learning
Eye-blink detection

ABSTRACT

The progressive diminishment of motor capacities due to Amyotrophic Lateral Sclerosis (ALS) causes a severe communication deficit. The development of Alternative Communication software aids ALS patients in overcoming communication issues and the detection of communication signals plays a big role in this task. In this paper, volunteer eye-blinking is proposed as human–computer interaction signal and an intelligent Computer Vision detector was built for handling the captured data in real-time using a generic webcam. The eye-blink detection was treated as an extension of the eye-state classification, and the base pipeline used is delineated as follows: face detection, face alignment, region-of-interest (ROI) extraction, and eye-state classification. Furthermore, this pipeline was complemented with auxiliary models: a rotation compensator, a ROIs evaluator, and a moving average filter. Two new datasets were created: the Youtube Eye-state Classification (YEC) dataset, built from the AVSpeech dataset by extracting face images; and the Autonomus Blink Dataset (ABD), built completely as a result of the present work. The YEC allowed training the eye-classification task; ABD was specifically idealized taking into consideration volunteer eye-blinking detection. The proposed models, a Convolutional Neural Network (CNN) and a Support Vector Machine (SVM), were trained by the YEC dataset and performance evaluation experiments for both models were conducted across different databases: CeW, ZJU, Eyeblink, Talking Face (public datasets) and ABD. The impact of the proposed auxiliary models was evaluated and the CNN and SVM models were compared for the eye-state classification task. Promising results were obtained: 97.44% accuracy for the eye-state classification task on the CeW dataset and 92.63% F1-Score for the eye-blink detection task on the ABD dataset.

1. Introduction

The advancement of technology should allow individuals with severe disabilities to use a computer through human–computer interfaces (Królak & Strumiłło, 2012). One such disability is Locked-in Syndrome (LiS), defined in 1986 as a severe disorder consisting of quadriplegia and anarthria with preserved consciousness (Bauer, Gerstenbrand, & Rumpl, 1979). LiS can be classified in Classic, Incomplete, and Total, with the first two cases conserving upper eyelid movement (Smith & Delargy, 2005). This set of symptoms characterizes, among other diseases, Amyotrophic Lateral Sclerosis (ALS) (Barbalho et al., 2021; Kiernan et al., 2011; van Es et al., 2017) whose patients are potential users of this work. The quality of life of patients affected by ALS is directly related to their communication capacity (Fernandes et al., 2021; Rosa Silva et al., 2020; Rousseau et al., 2015).

Considering that ALS patients are able to blink even in advanced stages of the disease and the interest in enhancing their communication capacity, this work presents a low-cost and real-time eye-blink detector system to be used as an activation and control tool for Alternative Communication Systems (ACS). Our approach requires only a generic webcam and computer, being therefore not only valid, but also practical. The detector will also be made freely available. Thus, any ACS system will be able to integrate our model and benefit from its low-cost and high performance.

* Corresponding author.
E-mail addresses: pauloaugusto99@ufrn.edu.br (P.A.d.L. Medeiros), gabriel.vinicius@lais.huol.ufrn.br (G.V.S.d. Silva), felipe.ricardo@lais.huol.ufrn.br (F.R.d.S. Fernandes), ignacio.gendriz@lais.huol.ufrn.br (I. Sánchez-Gendriz), hertz.lins@lais.huol.ufrn.br (H.W.C. Lins), daniele.barros@lais.huol.ufrn.br (D.M.d.S. Barros), danilo.nagem@lais.huol.ufrn.br (D.A.P. Nagem), ricardo.valentim@lais.huol.ufrn.br (R.A.d.M. Valentim).

https://doi.org/10.1016/j.eswa.2021.116073
Received 12 March 2021; Received in revised form 22 July 2021; Accepted 10 October 2021
Available online 16 October 2021

Eye-blink detection is a well discussed topic and different approaches have been proposed for addressing it. A subset of these approaches relies on capturing and using physiologic signals, such as Electroencephalography (Nguyen, Nguyen, Truong, & Van Vo, 2013), Electrocorticography (Picot, Charbonnier, Caplier, & Vu, 2012) and Electromyography (Yang, Lin, Lin, & Lee, 2013). Although these methods show promising results, these approaches require coupling additional hardware to patients for the signal acquisition step. Given that these devices need to be superficially coupled to patients, they comprise additional cost and may disturb the user.

Another subset of approaches tackles this subject from a computational vision perspective, such as the solution presented in this study. Differently from the aforementioned alternatives, these methods require only a generic webcam for the signal acquisition step, being therefore non-intrusive as well as cheap. Computer vision approaches mainly depend on face detection and eye-state classification. Since the former is an already well-defined and well-studied field in literature, most works tend to contribute on new eye-state classifiers, auxiliary models and pipelines.

With respect to related works, some studies have proposed image flow (Divjak & Bischof, 2008, 2009; Drutarovsky & Fogelton, 2014; Fogelton & Benesova, 2016, 2018; Singh & Singh, 2018) for detecting eyelid movements. While some of these works (Divjak & Bischof, 2008; Drutarovsky & Fogelton, 2014; Fogelton & Benesova, 2016) propose combining the use of Finite State Machines (FSM) to enhance eye-blink detection, the work of Fogelton and Benesova (2018) outperforms this technique by using Recurrent Neural Networks.

Template matching methods have been proposed in Chau and Betke (2005) and Królak and Strumiłło (2012). These works sample patches of the subject's open eyes on a specific phase and keep them as templates. Throughout the video stream, the current eye patches are extracted and a similarity score is computed between these and the templates; low scores indicate closed eyes.

Other methods used the analysis of the variance map of the pixels in the eye region and the distance of the pixel values to a fixed threshold computed at the initialization phase for different color channels . Unlike these works, our base eye-state classifier is able to perform on a single frame without the need of a initialization phase or contextual information.

Other methods analyze the variance map of the pixels in the eye region (Morris, Blenkhorn, & Zaidi, 2002) and the distance of the pixel values to a fixed threshold computed at the initialization phase for different color channels (Panning, Al-Hamadi, & Michaelis, 2011). Unlike these works, our base eye-state classifier is able to perform on a single frame without the need for an initialization phase or contextual information.

Regarding other methods that do not require contextual information, (Danisman, Bilasco, Djeraba, & Ihaddadene, 2010) proposed a function for determining eye openness which uses the horizontal symmetry of open eyes as motivation. In this method, the patch region is extracted, normalized, and split into the upper and lower half. The asymmetry between these halves is measured and high-scores indicate closed eyes.

Narmadha, Mythili, and Nivetha (2014) used the detection of vertical edges in the eye region for indicating eye openness while using the mean shift algorithm. Fathi and Abdali-Mohammadi (2015) computed the distance between eyelids by applying a variance projection derivative function. The blink was then detected by using a FSM.

Weak learners trained by the Adaboost algorithm were used on Pan, Sun, Wu, and Lao (2007) to estimate the eye-closity. This value is then used on a Conditional Random Field for detecting blinks. Ahmad and Borole (2015) developed a different approach, consisting of using a Haar cascade detector to probe for open eyes. If no eyes are found, it is assumed that the eyes are closed.

Other techniques focus on using traditional feature extractors on images, which are then passed as input to a Machine Learning (ML) classifier. The Local Binary Patterns feature extractor and its rotation invariant and uniform versions are used on Wang (2017), followed by a Support Vector Machine (SVM) classifier. His work was able to achieve great validation scores.

Mikhail and e. Kaliouby (2009) used Gabor Jets as a feature extractor and carried out a performance comparison of this extractor with different ML classifiers.

Additionally, the work performed by Song, Tan, Liu, and Chen (2014) explored different combinations of feature descriptors and machine learning classifiers. Their work succeeded in presenting two variants of the Histogram Of Oriented Gradients local shape descriptor to enhance the detection performance of eye-blinks.

The work developed here is mainly inspired by the use of facial landmarks as geometric features, as used in Bacivarov, Ionita, and Corcoran (2008), Maior et al. (2018), Soukupová and Cech (2016) and by the current trend of using Convolutional Neural Networks (CNNs) in classification tasks, which has been explored for eye-state estimation in Anas, Henríquez, and Matuszewski (2017), Li, Chang, and Lyu (2018) and Xiong et al. (2017).

Compared to other computer vision works for eye-blink detection, this paper stands out by creating auxiliary models which allow for greater performance in real use conditions of the system. These models are a rotation compensator, designed for mitigating the muscular impairment of the patient in the face rotation movement, a quality evaluator of the extracted ROIs, and a moving average filter, which allows information from the surrounding frames to integrate the current prediction. The system is also attractive due to its use of recent Deep Learning (DL) methods both in face detection and eye classification sub-tasks. In spite of employing auxiliary and DL models, it was possible to build a robust pipeline, which is able to achieve real-time performance utilizing generic hardware in both the acquisition and processing phases.

Initially, the eye-blink detection task is defined as an extension of the eye-state classification task and our base system tackles each frame independently, sorting each one into open or closed eyes. The system is then able to detect a sequence of closed-eye frames and compute their aggregate duration, yielding a blink with this exact duration. We later improved our base model by adding auxiliary models. Finally, the system outputs the duration of the detected blink which can be used as a communication signal by an ACS.

A robust dataset (Section 2.10.1) was built for training the developed ML models for the eye-state classification task. This dataset was built by extracting Youtube videos and pre-annotating them with the model presented in Li et al. (2018) (simulating a teacher model). Another dataset (Section 2.10.2), simulating the application conditions of the tool, was built and the developed models were tested on it in the eye-blink detection task.

Concerning the structure of this paper, the proposed models are introduced in Section 2. In this section, the main blocks of the system and how they connect to each other are described. In Section 3, experiments for validation and performance evaluation of the models are presented and discussed. Finally, Section 4 contains the conclusion.

The main contributions of this work are:

- Construction of a dataset to evaluate the eye-blink detection task, allowing the discrimination of voluntary blinks, as well as a dataset for the eye-state classification task
- Design and implementation of an eye-state classifier using a well-modeled pipeline based on high-performance models
- Development of a blink detector and implementation of auxiliary models idealized to be used by ALS patients

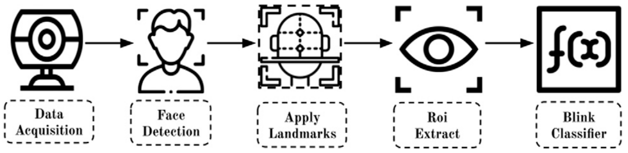


Fig. 1. General flow of the system.

2. Materials and methods

The proposed system was designed to tackle both eye-state classification and eye-blink detection tasks. The first task is to classify the eye-state of a face image as either open or closed. The inputs in this task do not necessarily present temporal or sequential relation and, for each input, there is one expected output.

On the other hand, eye-blink detection is the task of detecting blinks in a given video i.e. sequence of frames. Since the input of this task is a video, the frames have a sequential relationship. For each input video, the list of occurred blinks is expected as output. A blink is defined as a sequence of consecutive frames with closed eyes. In this way, a blink will be represented by the limits of its correspondent sequence. In this work, we treat the eye-blink detection task as an extension of the eye-state classification task.

First, the workings of the system for the eye-state classification task will be introduced, and then it will be shown how the architecture is extended for eye-blink detection.

The overall architecture of the system (Fig. 1) can be divided into sequential steps or blocks. The first step is to detect the face of the user. Then, landmark points are detected from the current face. These points are then used for face alignment and for extracting coordinates of both eye-patch and eye. Since the eye-patch extraction may fail by retrieving a patch without a real eye, a CNN model is used to classify these extractions into eye images or non-eye images. The last step is eye-state classification and two models were developed for this classification task. The first one is based on a CNN and will classify the correct eye-patches into open or closed eyes, whereas the second one uses an SVM to sort the extracted eye coordinates.

These two models can map, within a low error rate and with low latency, a webcam picture to a class (open/closed eyes), thereby solving the eye-state classification task. Furthermore, the proposed system will also be used for eye-blink detection and hence the temporal dimension is thereby present. Given this new dimension and the high performance of the proposed system, some benefits can be pointed out. These characteristics allow for the development of auxiliary models, such as a sliding-window filter to be applied on the mappings of consecutive frames i.e. eye-state classifier outputs. When such filters are used, instead of thresholding the outputs of the eye-state classifiers right after their computation, the thresholding is only carried out after filtering. These filters act as noise removers and can enhance the classification performance for each frame. A second benefit concerns tackling the problem of missing the detection of rotated faces. A simple yet effective auxiliary model is presented to address this issue as a rotation compensation step.

2.1. Definitions

In this section, concepts and terminologies that were used along the development of the system will be presented.

2.1.1. Blink

For this work, a blink on a video will be defined as a sequence of consecutive frames with closed eyes. Therefore, the frames in which the eyes are not completely shut (frames in which the eyes are closing or opening) will not compose the blink interval. This definition is based on the application of the system and is justified below.

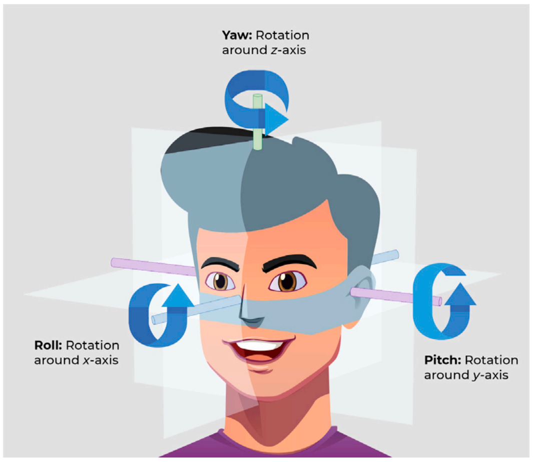

Yaw: Rotation around z-axis
Roll: Rotation around x-axis
Pitch: Rotation around y-axis

Fig. 2. Head rotation movements around the x, y and z axis.

Since a blink will be used as a communication signal and only voluntary blinks (see Section 2.1.2) will be considered, the user must have control over the blink duration. Also, it was agreed among the authors that the user has control only over the time she/he spends with their eyes completely shut, thus not controlling how long it takes for both closing and opening (transition between open-closed eyes) their eyes. Therefore, choosing to define the blink as aforementioned fits the application best. It must also be noticed that, following this definition, unfinished blinks will not be considered as blinks, since no eyes are shut for these events.

2.1.2. Voluntary and involuntary blinks

In the eye-blink detection task, one may classify the detected blinks in one of two classes: voluntary or involuntary. The approaches found for this classification task used only the blink duration as a descriptor (Chau & Betke, 2005; Fathi & Abdali-Mohammadi, 2015; Singh & Singh, 2018). If this duration is greater than a predetermined threshold, the blink is considered to be voluntary. Otherwise, it is an involuntary blink. Therefore, involuntary and voluntary blinks can also be named short and long blinks respectively. In healthy subjects, the average voluntary blink duration was reported to be 572 ± 25 ms (Kwon et al., 2013) and thus lengths of time greater than this value could be used as indicators of a voluntary blink.

The presented system detects an eye-blink and sends its duration to the ACS. The ACS will then be responsible for defining the voluntary-blink threshold. However, it is recommended that a safe margin is granted to the chosen value. This margin is desired because, while missing an expected activation signal requires only the repetition of the blinking action by the user, taking in consideration an undesired signal may trigger an unwanted action of the system.

2.1.3. Head rotation

Regarding the head movements, given the 3D space, there are three different axis for rotation. This work uses the same terminology displayed in Fig. 2: rotation around the x axis is named roll; around the y axis is named pitch; around the z axis is named yaw. These rotations affect the system differently and approaches were developed to tackle some of these impacts.

2.2. Sequential rotation compensation

Recent face detectors are indeed robust (Zhu, Cai, Zhang, Wang, & Xiong, 2020). However, failures still occur when the targeted face is exaggeratedly rolled, since this is not a commonly expected input. To diminish the number of missed detections due to this issue, a rotation

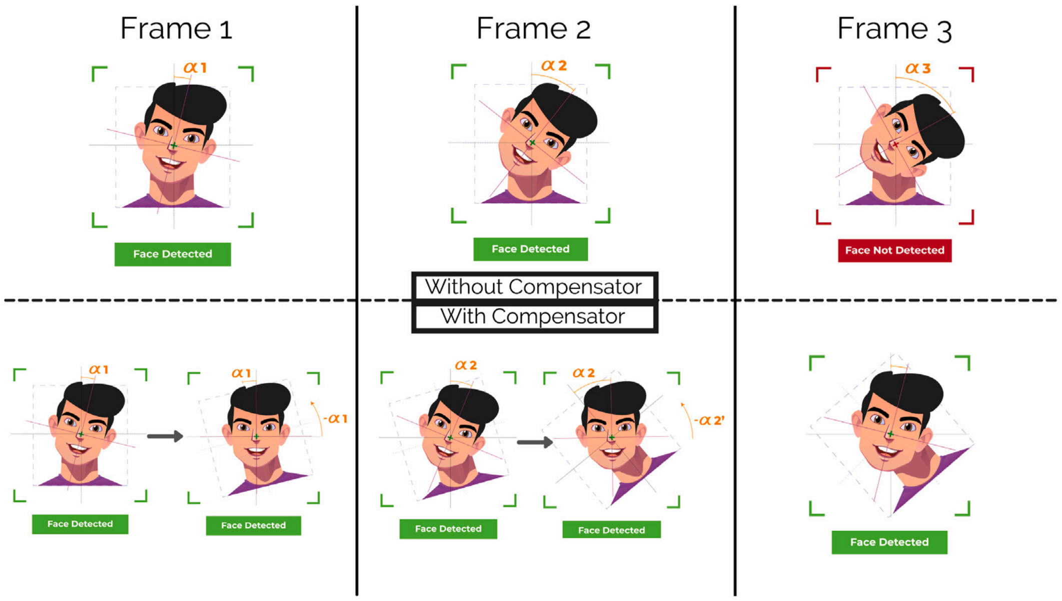


Fig. 3. Rotation compensator.

compensation auxiliary model was designed to work on a sequence of frames. Its process is depicted in Fig. 3.

Given that the ith frame f_i has face rotation angle equal to α_i, one can compensate this rotation angle in f_{i+1} by rotating it by $-\alpha_i$. Hence, this new face rotation angle is $\alpha'_{i+1} = \alpha_{i+1} - \alpha_i$. Given the high frame rate of the system, the angle variation α'_{i+1} is expected to be small. Thus, the chance of such angle causing the reported issue is smaller. The rotation on f_{i+1} occurs around the face center found in f_i. When f_{i+2} is then processed, it will be rotated by $-\alpha_{i+1}$ beforehand, as a rotation compensation step. Such value can be computed using $\alpha'_{i+1} = \alpha_{i+1} - \alpha_i$: α'_{i+1} is computed by the face alignment step and α_i is the last value used by the rotation compensator.

Note that, when α_i and α_{i+1} represent angles of opposite directions, $|\alpha'_{i+1}| = |\alpha_{i+1}| + |\alpha_i|$, which may be harmful to the face detector at higher values. However, given the high frame rate of the system i.e. small interval of time between two frames, if consecutive frames have opposite head rotation angles, these angles have to be low.

This component is expected to reduce the miss rate of the face detector and also contributes to the face alignment, by providing a less rolled face to this step. If the face detection step misses one frame, there will not be a compensation on the next valid frame .i.e no rotation compensation is done.

2.3. Face detection

Face detection is a traditional area of the computer vision research field and is one of the main blocks in the proposed system. In this work, a Single Shot MultiBox Detector (SSD) (Liu et al., 2015) is used for this task. This model is an extension of CNNs.

These architectures were first presented in LeCun, Bottou, Bengio, and Haffner (1998), being later reintroduced in Krizhevsky, Sutskever, and Hinton (2012), and are specialized neural networks that deal with grid-like data (Goodfellow, Bengio, & Courville, 2016). Widely researched and used in the literature, it is the chosen architecture of the majority of state-of-the-art models for computer vision tasks, (Gu et al., 2019; Tan & Le, 2019; Wang, Xu, Liu, Zhu, & Shao, 2019) having already surpassed human-level performance on some tasks (He, Zhang, Ren, & Sun, 2015).

Given an input data $x \in \mathbb{R}^{W \times H \times D}$, where W, H and D represent the image's width, height and depth respectively, CNNs will compute an encoding $y \in \mathbb{R}^{D'}$ for x. These networks are built by a sequence of blocks, composed of convolutional filters K, whose weights are learned during the training phase, followed by non-linear functions. On each of these blocks, the goal is to transform the input by reducing its spatial resolution and expanding its depth. The intermediate representations, outputs of such blocks, are called feature maps. Finally, these networks can be thought of as a non-linear encoding $f : \mathbb{R}^{W \times H \times D} \rightarrow \mathbb{R}^{D'}$ parameterized by K.

One must notice that CNNs cannot detect objects on their own. Different solutions, such as YOLO (Redmon, Divvala, Girshick, & Farhadi, 2016), Faster R-CNN (Ren, He, Girshick, & Sun, 2015), and SSD (Liu et al., 2015) have been proposed in the literature, and, for this work, the latter is employed. This is a single-shot architecture since location and classification of the detections are done on a single forward-pass. The SSD architecture uses a base CNN and specific feature maps computed by this base are used for locating the bounding boxes of the candidate detections. A fixed set of bounding boxes are linked to each cell on these feature maps. Convolutional filters are then applied to these specific feature maps in order to compute the offsets of the boxes and scores for each class. Since these bounding boxes overlap and the same object can be detected by more than one bounding box, a non-maximum suppression is performed to cut off redundant boxes.

In order to achieve lower latency, depthwise separable convolutions (Howard et al., 2017) are used in this work. This kind of convolutional blocks provide an enhancement on the time performance and parameter count of deep models at the cost of a slightly higher error rate (Howard et al., 2017).

Moreover, Contrast Limited Adaptive Histogram Equalization (CLAHE) (Zuiderveld, 1994) is applied to each frame as a preprocessing step, before face detection. Since these are RGB images, the color space is first converted to YCbCr and then this equalization technique is employed at the luma component. Finally, the color space is reconverted to the RGB color space.

When no faces are found in the eye-state classification task, the system is unable to produce an output. The behavior of this component when no faces are detected in the eye-blink detection task is described in Section 2.9.

2.4. Face alignment

Face alignment concerns transforming a face image in order to fit it into a geometric facial model. Following other works (Li et al., 2018;

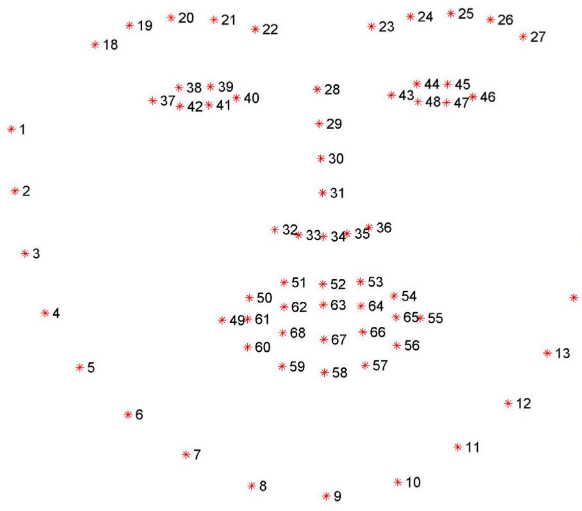


Fig. 4. Facial landmarks used at the IBUG dataset (Sagonas, Antonakos, Tzimiropoulos, Zafeiriou, & Pantic, 2016).

Wang, 2017), the method used in the presented system also relies on extracting frontal facial landmarks, features that are also used on other tasks such as face recognition (Soltanpour, Boufama, & Wu, 2017), gender recognition (Bekios-Calfa, Buenaposada, & Baumela, 2014) and recognition of emotion from facial expression (Martinez & Du, 2012).

Facial landmarks are strategic points carefully chosen to resemble the face structure. Hence, their location can be useful for face alignment and, as also used in this system, for extraction of coordinates and patch of the eye region.

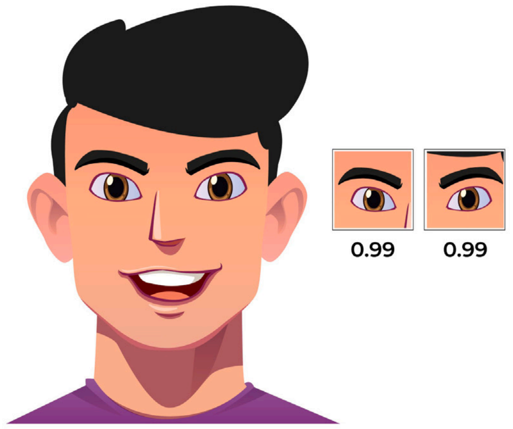
Different sets of facial landmarks can be found in the literature, each of them defined by a specific dataset (Belhumeur, Jacobs, Kriegman, & Kumar, 2013; Le, Brandt, Lin, Bourdev, & Huang, 2012; Sagonas et al., 2016). Also, the same dataset can provide different sets of points, each of them having a specific purpose. In this work, the annotations of the IBUG dataset (Sagonas et al., 2016) were used (see Fig. 4).

For landmarks extraction, the model presented by Kazemi and Sullivan (2014) was used. It consists of a cascade of regression trees, learned with the gradient boosting algorithm. For each regressor at the cascade, its input is a function of the shape estimation from the past regressor, which generates better features for the current tree. Also, in order to reduce the search space complexity, the shape estimates are assumed to be linear and kept into a linear subspace. By combining these points into their work, a new, low-error and ultra-real time shape predictor was conceived.

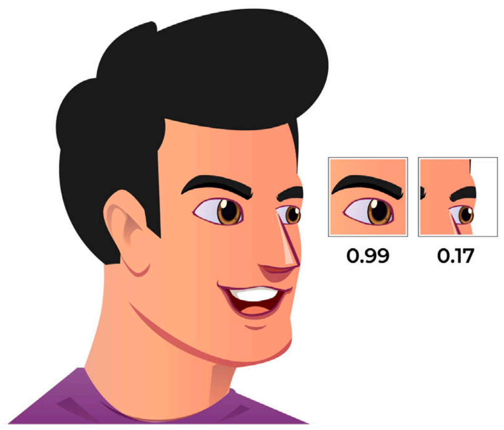
In order to align the faces to their correct positions, the Umeyama algorithm (Umeyama, 1991) was used. This method presents a solution to the problem of finding the similarity transformation parameters that give the least mean squared error between two sets of points of the same size. In this work, one set are the extracted landmarks of the face frame and the other is the alignment goal. The latter was retrieved by computing the average of point patterns extracted from the YEC dataset presented at Section 2.10. Additionally, the CLAHE technique was applied to each frame before extracting the landmarks.

2.5. Eye patches and coordinates extraction

The extracted landmarks will also be used for eye coordinates and eye-patch extraction. The used eye-patch size is 40×40. As depicted in Fig. 4, the left and right eye are represented by $C_{left} = \{P_i \mid 43 \le i \le 48\}$ and $C_{right} = \{P_i \mid 37 \le i \le 40\}$ respectively. These are the coordinates to be extracted. This subset was also used by Li et al. (2018) for eye-patch extraction.



(a) Example of high evaluation



(b) Example of low evaluation

Fig. 5. Performance of the eye evaluator on different situations.

In this work, the eye-patches can be extracted from two different axis-parallel rectangles for each eye: R^{min}_{left}, R^{max}_{left}, R^{min}_{right} and R^{max}_{right}. For the left eye, R^{max}_{left} is the smallest rectangle containing $C_{left} \cup \{P_{18}, P_{22}, P_{29}\}$, while R^{min}_{left} uses only the points on the corners of the eye: P_{37}, P_{42}. For the latter, the smallest rectangle containing these points is computed and then its height is leveled to its width, while maintaining the original middle point. For the right eye, the symmetrical shapes are extracted: R^{max}_{right} uses $C_{right} \cup \{P_{23}, P_{27}, P_{29}\}$, whereas R^{min}_{right} uses P_{43}, P_{46}. These rectangles were designed aiming R^{min}_{left} to be inside of R^{max}_{left} and R^{min}_{right} inside of R^{max}_{right}.

Both R^{max}_{right} and R^{max}_{left} present a greater area and thus a safety margin as eye-patches. Therefore, these forms are more interesting for extraction in real-life scenarios. A more appealing usage for R^{min}_{right} and R^{min}_{left} is shown in Section 2.7.

An auxiliary model based on a CNN was designed to evaluate the quality (whether an eye is indeed present) of the extracted eye-patches. This evaluation is critical since the facial landmarks are designed for frontal faces and thus unsatisfying eye-patches may be extracted from non-frontal faces (resulting from rolling, yawing or pitching). These likely incorrectly extracted eye-patches containing no eyes may cause unexpected behavior, compromising the system's performance. Fig. 5 depicts the usage of the evaluation model.

As presented, CNNs are used as image encoders. When followed by a sequence of fully-connected layers, these networks can be used for image classification and achieve state-of-the-art results (Tan & Le, 2019). The presented model follows the architecture depicted in Fig. 6. Its small size is crucial for keeping the system latency low. The expected

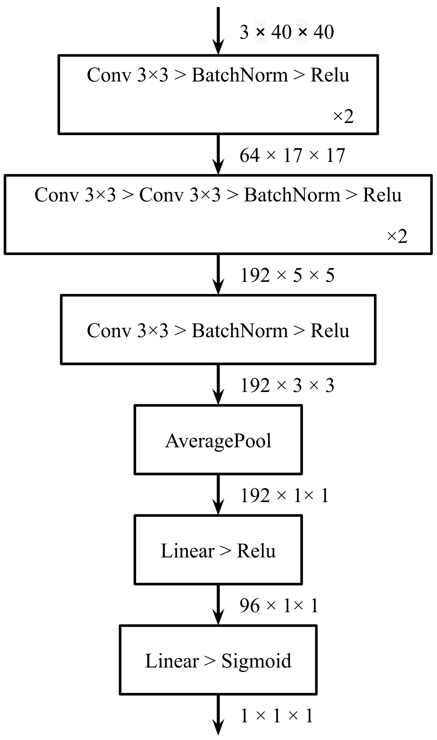


Fig. 6. Architecture for the CNNs used in this work. Blocks containing convolutional (Conv.) and dense (Linear) layers, activation functions (Relu), pooling (AveragePool) and/or normalization (BatchNorm), are represented as boxes. Tensors flowing between these blocks have their three dimensions described following a depthwidthheight format. Batch dimension is omitted.

input are the extracted patches in the RGB color channel (a 3 × 40 × 40 tensor) and the model outputs the probability of such patches being adequate eye-patches (a scalar for each patch).

A simple algorithm (Algorithm 1) was designed for this auxiliary model. It expects the extracted eye patches from the current frame as input and filters them according to two fixed probability thresholds, $upper\_thres$ (upper limit) and $lower\_thres$ (lower limit), $upper\_thres > lower\_thres$. Note that the output of the algorithm is affected by the task it is running. Since eye-blink detection is a video task, the system can opt to return None and therefore skip the classification step for the current frame. The eye-state classification task has only one image as input, hence no skipping is possible and at least one patch must be returned.

The 1 algorithm expects the extracted patches as input and determines the best and worst patches according to their evaluations (lines 1–7). Later, an analysis of how the scores assigned to each patch lay within the intervals induced by the thresholds is carried out (lines 8–19).

- If the best score is not greater than $lower\_thres$, the worst score will also be lower, due to transitivity (lines 8–12). In this case, if the current task is eye-state classification, there must be an output. Then, the best patch is returned. However, if the task is eye-blink detection, given the temporal property and high frame rate of the system, the invalid frame can be easily skipped and the possible ongoing blink disregarded. Thus, None is returned.
- If the best score is within the interval (*lower_thres*, *upper_thres*), only the best patch will be returned (lines 13–14).
- If only the best score is greater than *upper_thres*, only this score will be returned. However, if the worst score is also greater than *upper_thres*, both patches will be returned (lines 15–19).

Algorithm 1: Filter Frames

```
   Input: left_patch, right_patch
   Output: valid patches for classification
1  best_patch ← left_patch
2  worst_patch ← right_patch
3  punct_best ← eval(best_patch)
4  punct_worst ← eval(worst_patch)
5  if punct_best < punct_worst then
6      swap(best_patch, worst_patch)
7      swap(punct_best, punct_worst)
8  if punct_best ≤ lower_thres then
9      if current_task = eye-state classification then
10         return best_patch
11     else if current_task = blink detection then
12         return None
13 else if punct_best ≤ upper_thres then
14     return best_patch
15 else
16     if punct_worst > upper_thres then
17         return best_patch, worst_patch
18     else
19         return best_patch
```

2.6. Eye-state classification

Two different classifiers were used in this work to estimate the eye-state from the previously evaluated and possibly filtered eye-patches. The first model is a CNN which expects the figure of an eye as input and outputs the probability of it being a closed eye. The second model is an SVM that takes as input the X coordinates of an eye and also outputs the probability of it being a closed eye. Note that, on each frame, two eye patches/coordinates will be classified, unless the step presented at Section 2.5 disqualifies one of these. When two patches/coordinates are used as input, the average of their outputs will be used to define the output of the frame i.e. its final probability.

The SVM algorithm was originally introduced in 1963 by Vladimir N. Vapnik. Since then, it underwent updates that allowed it to classify non-linear data. SVM aims to find the hyperplane that best discriminate the classes in a N-dimensional space (N being the number of features). Therefore, this algorithm seeks to maximize the distance between the hyperplane and the points.

The Support Vectors are points of data that are closest to the hyperplane, being used to maximize the margin between classes and hyperplane. For non-linear classification, SVM uses the kernel trick, a method that increases the dimension of the data to classify and separates classes in a new space.

In order to build an SVM model for eye-state classification, 6 landmarks are used for each eye, resulting in 12 features in the $\mathbb{R}^2$ for each eye. The data scale was modified to range from 0 to 1 using the minimum and maximum values for each variable, this transformation is calculated as:

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

Where x represents a single feature vector. Since the classifier's outputs represent the probability of the frame containing closed eyes, the correspondent class still needs to be assigned if the task is eye-state classification. For this, a fixed value T_{image} is used for thresholding. When the task is eye-blink detection, the thresholding may done only after the process described in Section 2.8.

2.7. Data augmentation

In order to build robust CNNs, data augmentation was key during the training phase of these models. This is a technique that has been widely used due to its positive impacts (Shorten & Khoshgoftaar, 2019). Since this system is designed to be widely used, different equipment and environment conditions are expected during the data acquisition phase i.e. frame capture. Hence, brightness, saturation, and contrast were randomly modified for each training image. Also, various exposure settings for frame capturing are expected, something that is highly related to image noise. Therefore, different noise was randomly introduced to simulate these variations.

Frame proportion variation was also used in order to prevent biasing the classifier towards the same resolution. This augmentation is done by extracting R^{min}_{left} and R^{min}_{right} and randomly expanding these shapes independently on all four directions without traversing R^{max}_{left} and R^{max}_{right}.

2.8. Sliding-window filtering

Sliding window filters were used as auxiliary models to suppress the noise from the still non-threshold outputs of the classifiers, benefiting from the temporal relation between consecutive frames. Expecting the noise process to have high-frequency and zero mean (Rangayyan, 2015), average filters are thus a feasible solution. These are low-pass filters i.e. they preserve low-frequency signals and attenuate high-frequency ones. Also, the mean of the sampled noise will tend to zero as the size of the window increases.

For this work, two weighted versions of the moving average filter were used: Gaussian and rectangular. Additionally, the median filter was also implemented due to its well-known effectiveness in noise removal (Huang, 1981). The filters were applied to the outputs of the classifiers for each frame.

2.9. Eye-blink detection

The eye-blink detection model considered the outputs of each frame, already processed by the proposed filters (Section 2.8). Each of these outputs is a value between $[0;1]$, which will be binarized by a threshold T_{video}, implying that the respective frame presents an open (0) or closed (1) eye.

Following the aforementioned definitions (Section 2.1), a blink is any sequence of consecutive frames whose outputs are 1. Since only voluntary blinks are used as a communication signal, a method for measuring the blink duration needs to be presented.

Blink duration can be measured in two ways. If a fixed throughput (frames per second) is assumed, the number of closed eye frames will indicate the duration of the ongoing blink. Hence, a simple counter can be used in this case. However, on real-life applications, the system will be exposed to performance variations, making the fixed throughput assumption unfeasible. Therefore, timestamps are a more precise manner of measuring a blink duration. Marking the start and end of the blink and then calculating the difference is a simpler and more precise way to measure blink duration. When a blink is then detected, the system can finally send its duration to the ACS.

The complete system for eye-blink detection is depicted in Fig. 7.

2.9.1. Face-detection misses

Face-detection misses are handled differently from the eye-blinks since frames can be skipped or the head positioning of the previous frame can be reused. The 2 algorithm describes the process of face detection and the handling of missed detections, which is sensible to the usage of the rotation compensator and eye-patch evaluator.

This algorithm relies on the eye-patch evaluator in the sense that, when active (*use_eye_eval* is set to *true*), this auxiliary model may repeat the previous detected face position (*last_face_pos*) and then it will be responsible for deciding if the frame should be skipped (in the case of not extracting valid eye-patches) as explained in Section 2.5.

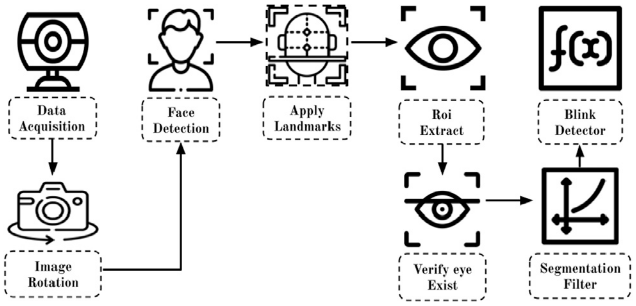


Fig. 7. General Flow of the system.

Algorithm 2: Face-detect miss
```
Input: last_face_pos
unrot_frame ← capture_frame()
frame ← unrot_frame
if use_rc then
    frame ← compensate(unrot_frame)
curr_face_pos ← detect_face(frame)
if curr_face_pos == None then
    if use_rc and not rc.is_reseted() then
        unrot_face_pos ← detect_face(unrot_frame)
        if unrotated_face_pos == None then
            if use_eye_eval then
                return last_face_pos
            else
                rc.reset()
                return None
        else
            rc.reset()
            return unrot_face_pos
    else
        if use_eye_eval then
            return last_face_pos
        else
            return None
```

Also, if the rotation compensator auxiliary model is active (*use_rc* is set to *true*), the missed detection is followed by the re-execution of the face detector on the unrotated frame (*unrot_frame*). If a face is found, it will be used in the following steps and the rotation compensator is reset (*rc.reset()*). Else, if the eye-patch evaluator is inactive, the frame is skipped, which also happens when the rotation compensator is not used. Note that the face detector processes the unrotated frame only when necessary, thus not harming the latencies of the models.

2.9.2. Usage conditions

A generic webcam and a computer are the minimum requirements to use the provided detector. The camera must be positioned in front of the user's face so that the whole face is present in the extracted frame. ALS patients spent a considerable part of their day sitting (Güell et al., 2013; Karam, Paganoni, Joyce, Carter, & Bedlack, 2016; Soriani & Desnuelle, 2017), so the camera might be positioned on a table or in a support attached to their chairs. The only actual requirement is that a frontal face is detected. In Section 3.2, the system will be evaluated across different datasets, under different resolutions, illumination, and user conditions.

2.10. Proposed datasets

Different datasets have already been proposed for evaluating the eye-blink detection task (Drutarovsky & Fogelton, 2014; Fogelton & Benesova, 2016; Pan et al., 2007). However, most of them do not match the purposes of this work regarding annotation issues and voluntary blink duration. Therefore, in order to develop and evaluate the designed models in this work, two datasets were built: a training dataset and a testing one.

The training dataset, named Youtube Eye-state Classification (YEC), was built by extracting frames from Youtube videos, while the testing dataset, named Autonomus Blink Detection (ABD), consists of video recordings of volunteers — with the due agreement of the parts. The YEC dataset contains face images and is annotated with the eye-states of these faces. It will thus be used for training the eye-state classification task (which may be extended to the eye-blink detection task). The ABD dataset contains videos in which each frame has the eye-state annotations and will thus be used for evaluating the eye-blink detection task. Blinks were annotated in this dataset following our definitions presented in Section 2.1.

2.10.1. YEC dataset

The YEC dataset was built by extracting faces from the videos presented at the AVSpeech dataset (Ephrat et al., 2018) and annotating the state of the eyes of these faces. The AVSpeech dataset consists of a collection of links for Youtube videos and in each of the videos there is a segment in which a single visible person speaks.

For each video segment of interest, a pre-annotation step was carried out in which, for each frame, face detection and alignment were executed, eye-patches were extracted and the eye-states are annotated using the model presented in Li et al. (2018). Later, a final annotation step was performed in which wrong pre-annotations were manually corrected. Even though the AVSpeech dataset itself provides the face position annotations, the UltraLight[1] face detector was used.

In order to build a balanced dataset, it was sought to extract the same number of frames for each class (open/closed eyes) from each video. Also, since diversity and unbiasing are targeted, no more than 20 frames of each video were used, which increases variability.

Therefore, different faces with different resolutions were extracted to form this training dataset. In total, 15 052 closed-eye faces and 28 998 open-eye faces were extracted, totaling 44 050 face images.

2.10.2. ABD dataset

The experimental protocol was approved by the Research Ethics Committee of the Federal University of Rio Grande do Norte, Natal, Brazil, through letter; CAAE-No. 25687819.3.0000.5537, and in accordance with the Helsinki Accords (modified 2004). The ABD was built by recording healthy volunteers blinking, with a laptop integrated webcam. The subjects were seated to a distance of approximately 70 cm in front of the laptop, looking towards the camera, simulating real use conditions. Videos were captured on a closed room with controlled light.

The recording resolution was of 640 × 480 pixels at 30 frames per second. Each video is 40 s long and the volunteers were instructed to blink voluntarily exactly two times. Such blinks occurred at equiprobable random instants between 10 and 35 s of the recordings, with each voluntary blink lasting 2 s. There were no restrictions concerning involuntary blinks. The blinks were annotated following our definition (Section 2.1.1) i.e. only frames with completely shut eyes composed blinks. During all this process, the participants were oriented to avoid excess movement.

In total, 10 persons took part in the experiment. All the subjects recorded videos without wearing glasses and 7 with glasses. Concerning the characteristics of the subjects, 7 were males and 3 were females. Also, 5 were white-skinned and 5 brown-skinned and the average age was 23.1.

In order to annotate the recorded videos, a simple pipeline was developed. First, a pre-annotation step was executed for the YEC dataset (using the same eye-state classification model). Then, for each video, manual annotation was performed. A script consisting of three steps was developed: (I) validation of the pre-annotation on each frame; (II) correction of the wrong annotated frames; (III) analysis of possible inconsistent sequences, in which a small number of frames are dissonant of their surroundings.

The whole dataset consists of 17 videos. Altogether, there are 16921 images with open eyes and 3054 images with closed eyes, totaling 19975.

3. Experiments and results

To evaluate the performance of the presented models on the proposed datasets, different evaluation experiments were conducted.

Since the eye-blink detection task was approached as an extension of the eye-state classification task, the need for evaluating our system in the latter task arises (Section 3.1). The performance of the base model and the impact caused by adding the eye-patch evaluation model described in Section 2.5 to the pipeline, namely Eye Evaluator (EE), were assessed in both the ABD and Closed Eyes in the Wild (CeW) image datasets (Song et al., 2014).

As for the experiments with the eye-blink detection task, some considerations were raised concerning the difference between eye-blink annotations across the different used datasets. It was expected that such differences impact the results.

First, similarly to the eye-state classification task, the base model's performance and the impact caused by adding the EE and the Rotation Compensator (RC) auxiliary models to the pipeline were evaluated. For each possible combination, the moving filter configuration that maximizes the evaluation score was found. Next, a similar experiment was conducted but considering only voluntary blinks. In addition, a study for analyzing the impact of different filter configurations on the system's performance was conducted. Finally, a latency analysis of the final model and a comparison to related works were performed.

As stated in Sections 2.3 and 2.4 , the proposed system is built with a face detector and a landmarks extractor. The *UltraLight* model will implement the face detection step. This is an ultra real-time state-of-the-art model. For the landmarks extraction step, *Dlib* (King, 2009) was used.

The CNN and the SVM eye-state classifiers, presented at Section 2.6, were named Eye Classifier CNN (ECC) and Eye Classifier SVM (ECS), respectively. The RC, presented at Section 2.2, and the window filters (Section 2.8) were analyzed regarding the eye-blink detection task. Combinations of such models are described by concatenating the model names with the '+' symbol.

The implementation of the architectures vary depending on the model. The ECS model was implemented using Scikit Learning (Pedregosa et al., 2011) with $C = 1$. The kernel used was `rbf` with $\gamma = \}scale'$. Both ECC and EE were implemented and trained using *pytorch* library (Paszke et al., 2017). The batch-size was 32 and the Adadelta optimizer (Zeiler, 2012) was used. The models were trained for 180 epochs each. ECS, ECC and EE models were trained on the YEC dataset (Section 2.10). This dataset is not used for testing, hence no biasing can affect the results.

Given that the YEC dataset is constituted by face images and that the EE model expects an eye-patch candidate as input, the training phase of this model had to be conducted differently. Therefore, on each training iteration of this model, it was randomly and equiprobably chosen whether an eye or a negative sample (non-eye patch) should be used as input. Such negatives samples were extracted from the upper half of the image without intersecting with the eyes region, these being

[1] https://github.com/Linzaer/Ultra-Light-Fast-Generic-Face-Detector-1MB

retrieved by the landmarks extractor and the positive samples were the ones cropped out from the eyes region. The trained EE model achieves .996 AUC score in a subset of the YEC dataset, not used for training. Concerning the thresholds for valid eye-patch filtering, was set to .

Given that the YEC dataset is constituted by face images and that the EE model expects an eye-patch candidate as input, the training phase of this model had to be conducted differently. So, on each training iteration of this model, it was randomly and equiprobably selected whether an eye or a negative sample (without an eye-patch) should be used as input. Such negative samples were extracted from the upper half of the image without intersecting with the eyes region, these being computed by the landmarks extractor (Section 2.5). The positive samples were the ones cropped out from the eyes region. The trained EE model achieved a 0.996 Area Under the ROC Curve score (AUC) in a subset of the YEC dataset that was not used for training. $Thres_A$ and $Thres_B$, the thresholds for valid eye-patch filtering (Section 2.5), were set to 0.54 and 0.2 respectively.

3.1. Eye-state classification experiment

In this experiment, all possible combinations of ECC and ECS models with the EE auxiliary model were analyzed for accuracy, precision, recall and AUC in the image classification task i.e eye-state classification. Two datasets were used: CeW (Song et al., 2014) and ABD$_{images}$, an adaptation of the ABD dataset.

The *CeW dataset* is a challenging dataset for eye-state classification in the wild, having the unconstrained real-world as scenario. Both lighting and angle variations as well as occlusion are present. It is a balanced dataset containing faces of subjects with either open or closed eyes. There are at total 2423 images.

ABD$_{images}$ dataset dataset is an image dataset built from the original ABD dataset. For the ith video on the ABD dataset, it was considered that it has c_i closed eye images and o_i open eye images. Since $c_i < o_i$ and a balanced dataset was sought after, c_i closed eye images and c_i open eye images are extracted without replacement. The final dataset is thus balanced having 3054 samples for each class. Compared to the CeW dataset, this is a much less challenging dataset.

The metrics that were used for this evaluation experiment depend on the possible outcomes of this binary classification task, which are:

- True Positive (TP): Closed eyes classified as closed eyes
- False Positive (FP): Open eyes classified as closed eyes
- True Negative (TN): Open eyes classified as open eyes
- False Negative (FN): Closed eyes classified as open eyes

3.1.1. Results

Accuracy, precision, recall and F1-Score were plotted as a function of the threshold value in Fig. 8, varying within the range of $(0, 1)$ for the eye-state classifiers. Results for ECC+EE and ECS+EE models are plotted for both the CEW and ABD$_{images}$ datasets.

It can be noticed that the curves for the ECC+EE model are more stable to the threshold variation and present higher scores than the ones plotted for the ECS+EE model. This greater stability is related to the ability of the binary classifier to separate positive samples from negative samples. Furthermore, when comparing the curves in Fig. 8(a) to the curves in Fig. 8(b), it is possible to observe that both models achieve better results in the ABD$_{images}$ dataset, which confirms that eye-state classification in this dataset is a less challenging task. Also, since this dataset, as stated in Section 2.10.2, approximates the expected usage conditions of the model, these results indicate the system's performance under those conditions.

The Area Under the ROC Curve (AUC) and best accuracy values obtained are shown in Table 1. While all variations present great scores, the ECC model showed better values than the ECS models, displaying its best results when combined with the EE model. For the CeW dataset,

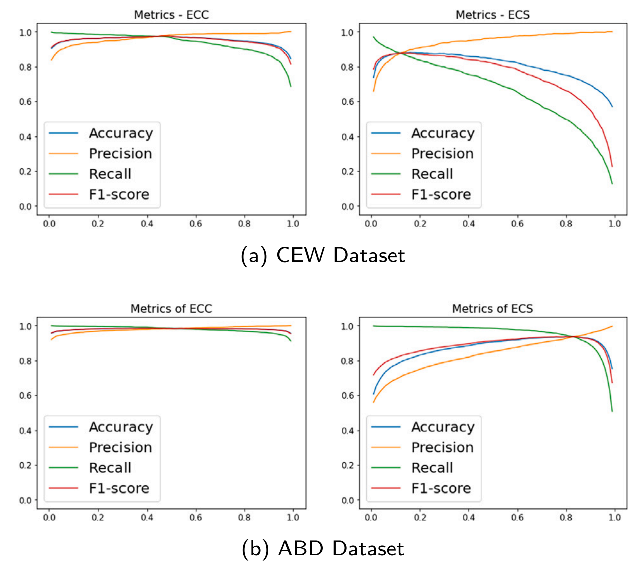


Fig. 8. Eye-state classification task scores as a function of threshold variation.

Table 1
AUC and best accuracy (BA) scores for ECC and ECS models in combination with the EE model for eye-state classification.

Models	CeW		ABD$_{images}$	
	BA	AUC	BA	AUC
ECS	0.9050	0.9571	0.9484	0.9571
ECS+EE	0.9063	0.9569	0.9484	0.9862
ECC	0.9744	0.9966	0.9805	0.9966
ECC+EE	0.9744	0.9967	0.9805	0.9982

the ECC+EE model was able to diminish the error rate in 72.7% in comparison to the ECS+EE model.

The ECC and ECS models differ both at the feature extraction level and on the used architectures. Whereas the latter depends highly on the performance of the landmarks extractor, the former benefits from an error tolerance that is introduced when the eye patches are extracted using the proposed margin (Section 2.5). Concerning the different architectures, CNNs are known for their superiority in processing images. Also, the ECC model benefits from the data augmentation process presented in Section 2.7. Such a technique allowed for the development of a model that is less susceptible to variance, being able to achieve great results even on the CeW dataset. Thus, the higher performance of the ECC model over the ECS was already expected.

3.2. Video experiments

These experiments focused on testing the efficiency of the system in the eye-blink detection task by evaluating precision, recall, and F1-score.

Four datasets were used: the original ABD dataset, *Eyeblink8* (Fogelton & Benesova, 2016), *ZJU* (Pan et al., 2007) and *Talking Face*.[2]

The *Eyeblink8* dataset consists of recordings of 4 different Caucasian subjects, each appearing on 2 videos. The videos were captured in 630×480 resolution and contain 70992 frames in total.

The *ZJU* dataset contains 80 short length videos, captured in 320×240 resolution at 30 FPS. 20 subjects recorded 4 videos each, varying camera position and glasses usage. The videos have, on average, 136 frames.

[2] https://personalpages.manchester.ac.uk/staff/timothy.f.cootes/data/talking_face/talking_face.html

Talking Face dataset consists of a single video of the face of a Caucasian male talking in front of a camera. The video is recorded at 25 FPS in 720 × 576 resolution, with a duration of 200 s for a total of 5000 frames.

In all of these datasets, light, background and camera conditions are stable. Subjects are mostly still, presenting few head movements, none of them exaggerated. Also, there is no distinction between voluntary and involuntary blinks. The blink annotations used for these datasets are the ones provided in Fogelton and Benesova (2016).

3.2.1. *Computing the metrics*

Precision and recall can be computed within a certain threshold by analyzing the correctly detected blinks, which can be found by computing the Intersection over Union (IoU) value between detected and ground-truth blinks (Fogelton & Benesova, 2016). The IoU function, well known for its use on object detection tasks in images (Everingham, Gool, Williams, Winn, & Zisserman, 2010; Lin et al., 2014), is computed as the ratio of two areas: the intersection between a detection and an annotation over the area of the union between these elements. However, while images lay on two dimensions, blink intervals lay on only one dimension. Hence, the areas must be replaced by lengths.

To compute the IoU value, let us denote a an annotated blink and d a detected blink. Also, let us define $b, e : \texttt{blink} \rightarrow \mathbb{N}$ for retrieving their beginning and ending positions, respectively. The lengths I and U of the intersection and union between a and d can be computed as follows:

$$I_{a,d} = max(min(e(a), e(d)) - max(b(a), b(d)), 0)$$

$$U_{a,d} = (e(a) - b(a)) + (e(d) - b(d)) - I_{a,d}$$

Now, a quadratic algorithm is able to match detected and annotated blinks using this score. For the detected blink d, find $\texttt{match}_d = \arg\max_a \texttt{IoU}(a, d)$ i.e. the annotated blink which maximizes the IoU scores for this blink.

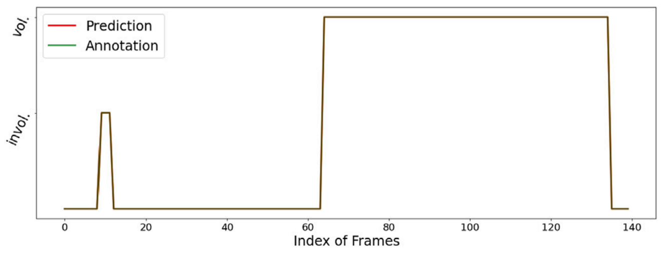
If, $\texttt{IoU}(\texttt{match}_d, d) > 0.2$, d was thus correctly detected i.e. it is a True Positive. Else, it is a False Positive. Finally, all annotated blinks that were not associated with a TP are then False Negatives. In other words, TP are blinks which occur in both annotation and detection output and are matched, FP are blinks that are only detected but not matched with the annotation and FN are annotated blinks which were not matched with any detections. Precision, recall and F1-score can thus be computed using the found TP, FP and FN values.

3.2.2. *Blink range analysis*

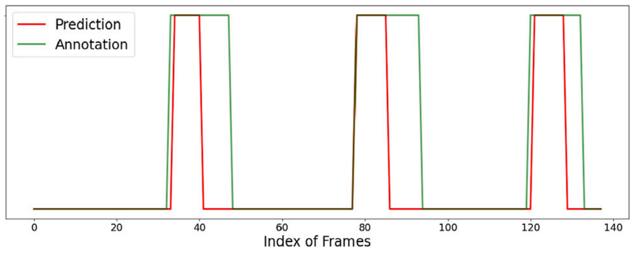
Before heading to the main results, one must note that the datasets used in this experiment have different origins. Therefore, the annotation rules and blink definitions are not necessarily the same. It was noticed that, while blinks on the ABD dataset are in accordance with our definition (Section 2.10), the ranges of blinks in other datasets tend to start at the onset of eyes shutting and finish only when they are completely open. Thus, the blinks on the ABD dataset have, by definition, a narrower range than those annotated by Fogelton and Benesova (2016).

Given that our models also follow our blink definition, it is expected for them to better fit the blink range in the ABD dataset. A comparison was drawn between the ECC+EE+RC performance on parts of both the ABD (Fig. 9(a)) and ZJU (Fig. 9(b)) datasets, as shown in Fig. 9. These figures were built by extracting video segments from the datasets and displaying prediction and annotation values throughout these videos. From ZJU, a whole video was used and, in order to keep the same scale, a video segment with the same frame count was extracted from ABD. For the ABD dataset the classification as voluntary or involuntary is shown on the y-axis; the ZJU dataset does not have such annotations.

The contrast between the blink range adopted in ZJU and ABD annotations is visible when examining Fig. 9, which highlights the alignment difference between annotation and prediction across datasets. The reader must remember that the presented models (including ECC+EE+RC) were trained on the YEC dataset, an image dataset, thus showing no bias for the ABD dataset.



(a) ABD Dataset



(b) ZJU Dataset

Fig. 9. Blink range analysis.

3.2.3. *Evaluation on different datasets*

This experiment evaluates the performance of all proposed models, exploring all possible filter configurations (as well as non filter configurations), on all video datasets. It was conducted in order to explore the best possible scores that can be reached with the model variations. Since all these combinations rely on a binary classifier, thresholds within the range of $[0.01; 0.99]$ were evaluated and only the best F1-Score achieved from these are shown and used as comparison parameter.

Since more than one filter configuration may result in the same maximum score, only the simplest one is shown. For this purpose, the first criterion for simplicity concerns the window function (the absence of filters is simpler than rectangular, which is simpler than Gaussian, which is simpler than median). And the second criterion concerns the window size: the smaller, the simpler. Results are shown in Table 2.

Initially, it can be noticed that the window sizes which maximize the F1-Score are small (at most 7) and, also, simpler window functions require a greater size for achieving the maximum scores. In addition, independently of the auxiliary models (EE, RC) used, the same filter configuration will maximize the score of the models on a given dataset. Thus, the filter configuration depends more on the base model (ECC, ECS) and dataset in question than on the analyzed auxiliary models combination.

Furthermore, as shown in Section 3.1, the ECC model achieved better results than the ECS. Since the eye-blink detection task in only an extension of the eye-state classification task, the ECC and ECS models are used in the same manner. Thus, the reasons presented in Section 3.1 (different feature extraction pipeline and architectures) for the superiority of the ECC model over the ECS can also be applied here. The error reductions across datasets obtained by comparing the ECC model to the ECS ranged between 59.68% (Talking Face dataset) and 79.19% (ABD dataset), achieving great results across all four datasets. Given that the different datasets present various camera positions, luminance, resolution and user ethnicity, the authors believe that the ECC model and its variations present a robust solution for the eye-blink detection task.

Furthermore, ECC and ECS models variations behaved similarly across datasets. For both base models, while the best results were achieved on the Talking Face dataset, the worst results occurred with the Eyeblink dataset, and, for both models, the results in ZJU and ABD

Table 2
Results for ECS and ECC combined with auxiliary models for the eye-blink detection task on all datasets.

Models	Eyeblink			ZJU			Talking Face			ABD		
	F1-Score	$W_{function}$	W_{size}	*F1-Score*	$W_{function}$	W_{size}	*F1-Score*	$W_{function}$	W_{size}	*F1-Score*	$W_{function}$	W_{size}
ECS	0.5582	Median	3	0.6940	Rect.	7	0.8640	Rect.	5	0.6429	Median	3
ECS+EE	0.5570	Median	3	0.6896	Rect.	7	0.8640	Rect.	5	0.6429	Median	3
ECS+RC	0.5507	Median	3	0.6905	Rect.	7	0.8760	Rect.	5	0.6460	Median	3
ECS+EE+RC	0.5532	Median	3	0.6905	Rect.	7	0.8760	Rect.	5	0.6460	Median	3
ECC	0.8656	Rect.	7	0.9219	Rect.	5	0.9500	Rect.	5	0.9256	–	–
ECC+EE	0.8624	Rect.	7	0.9237	Rect.	5	0.9500	Rect.	5	0.9256	–	–
ECC+RC	0.8697	Rect.	7	0.9140	Rect.	5	0.9500	Rect.	5	0.9263	Gaussian	3
ECC+EE+RC	0.8641	Rect.	7	0.9140	Rect.	5	0.9500	Rect.	5	0.9241	Gaussian	3

Table 3
Results for the ECS and auxiliary models on the voluntary eye-blink detection task.

Models	ABD_{vol}		
	F1-Score	$W_{function}$	W_{size}
ECS	0.9032	–	–
ECS+EE	0.9032	–	–
ECS+RC	0.8955	Rect.	17
ECS+EE+RC	0.8955	Rect.	17

datasets were similar. These results mirror the difficulty of the datasets. While the Talking Face is an easy dataset for both models, ZJU and ABD present medium difficulty and the most challenging dataset was Eyeblink.

The EE and RC auxiliary models, however, were not specifically tested in this experiment. As stated in Section 2, these models are useful when the subject presents head rolling, eye occlusion or other challenging eye-extraction characteristics. These are not present in the datasets used in this experiment. However, the introduction of the auxiliary models in the ECC and ECS models does not significantly hinder their performances and may even increase them for some settings.

Therefore, since it is believed that both the EE and RC models produce more adaptable variations and due to the great scores achieved by the ECC model, the ECC+EE+RC variant is considered the main contribution of this work for the eye-blink detection task.

3.2.4. Voluntary blinks

Since only voluntary blinks are considered a communication signal, an experiment was conducted to evaluate the performance of the models in the voluntary eye-blink detection task. Since the ABD dataset contains both voluntary and involuntary blinks (according to the definitions in Section 2.1), it was chosen for this experiment. All blinks shorter than 55 frames were disregarded both from annotations and from the outputs of the models, constituting the ABD_{vol} dataset. As per Section 3.2.3, the maximum achieved F1-Score was described. Results for the ECS model and its variations are shown in Table 3. The ECC model achieved 1.0 F1-Score on all four variations, for almost every filter configuration, and also when no filters were used.

Comparing the score of involuntary and voluntary blinks, it can be directly concluded that the task concerning only voluntary blinks is easier.

Since the system will use only voluntary blinks as a communication signal, the models built on ECC appear suitable for the proposed goal. Also, as already said in Section 3.2.3, the EE and RC auxiliary models produce model variations that are more adaptable. Therefore, the ECC+EE+RC model is considered the final model for the voluntary eye-blink detection task.

3.2.5. Analyzing filter parameters

Both experiments in Sections 3.2.3 and 3.2.4 explore all possible filter configurations (Section 2.8). However, one experiment about the behavior of these different filter configurations on the final model (ECC+EE+RC) must be analyzed. The results of this experiment can determine the suitable filter configurations to be used along with this model in production.

For this experiment, all odd window sizes at the interval $[3; 31]$ were tested and different thresholds within the range of $[0.01; 0.99]$ were analyzed. Also, rectangular, Gaussian and median functions of moving filters were considered. This experiment analyzes the results of the models on the ABD dataset, due to its simulation of the intended usage conditions, and on the Eyeblink dataset, due to its challenging properties.

A general approach was followed and took in consideration both involuntary and voluntary blinks in this experiment. Restricting ABD to only voluntary blinks was disregarded given that different scenarios may use different thresholds for voluntary blinks.

The stability through window size variation and the best score obtained with the configurations of the filters are the criteria taken into account for this analysis. Also, the results obtained with the ABD dataset are the ones considered for making the final decision, since this dataset best represents the proposed usage conditions.

Figs. 10(a) and 10(b) depict the experiment. These figures have 3 columns containing each a graph for one of the proposed metrics. For each of these graphs, 3 curves were plotted, showing the impact of the window size on the respective score for a window function. The diameter of each point in these curves is proportional to the threshold which maximizes the correspondent metric.

The configurations can be initially analyzed using the chosen criteria. The median and Gaussian functions show stability through different window sizes. Probably due to the fact that, while the median function is able to ignore outliers, the Gaussian function will give more weight to the outputs that originate from frames closer to the current, which attributes to these functions low sensibility to the increase in window size. The rectangular function, however, presents high instability of the F1-Score, stimulated mostly by the decrease of recall when higher window sizes are tested.

Regarding the best scores, all three functions present comparable results. Since the Gaussian and median functions present higher stabilities, their F1-Scores are almost constant. The rectangular, nevertheless, presents a bell-shaped curve for the Eyeblink dataset, whose maximum is the highest score for this dataset. On the ABD dataset, however, the curve for the rectangular functions present a maximum score similar to those achieved by the other functions.

Given that the ABD dataset is our focus and the instability of the rectangular function, only the Gaussian and median functions were considered as candidates. The scores achieved by these functions are extremely similar on the ABD dataset. Therefore, the values obtained on the Eyeblink dataset were the ones considered and the Gaussian function is then elected the most suitable.

Due to the high stability of the Gaussian curve, any window size presents valuable scores. However, smaller windows require a smaller buffer and fewer computations, which makes them better candidates. Intuitively and arbitrarily, the window size of 7 was chosen.

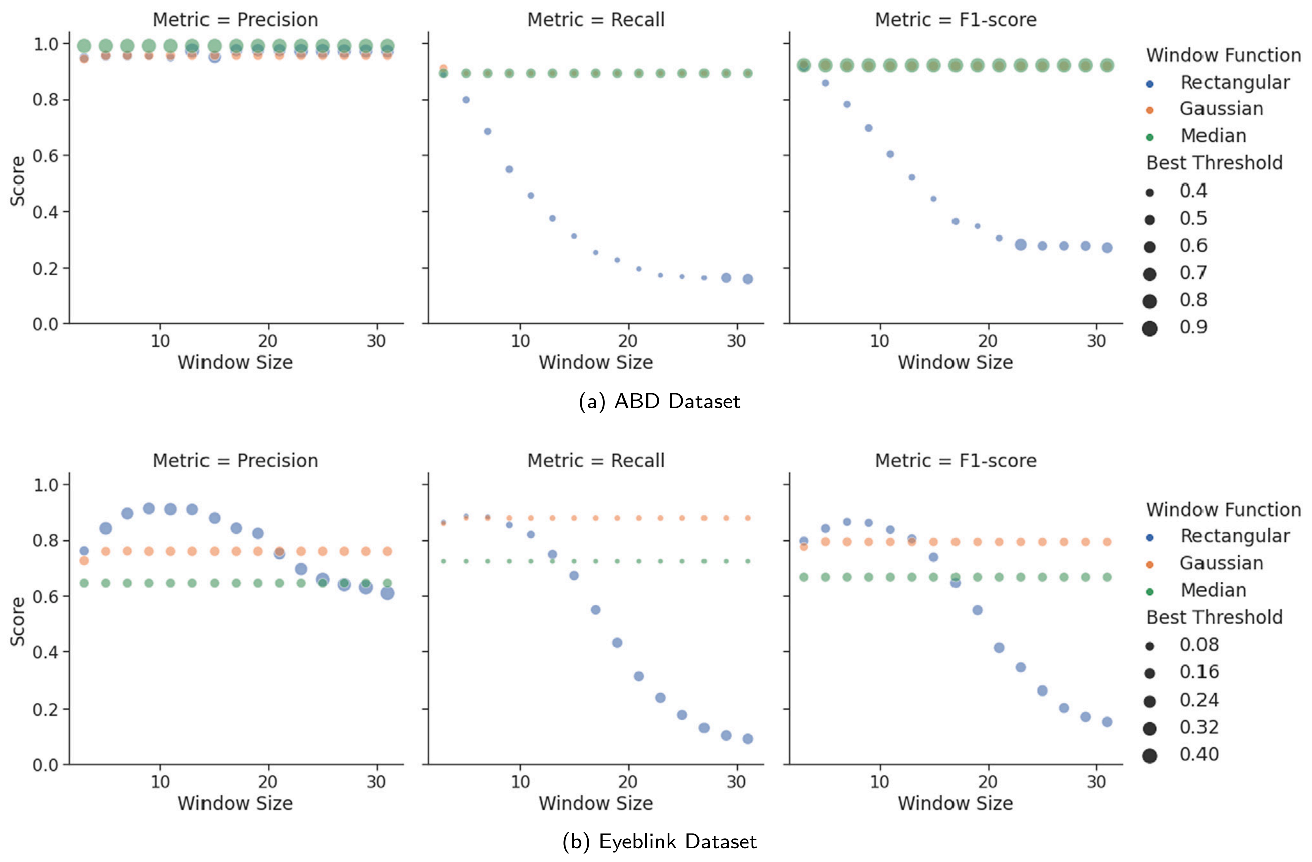


Fig. 10. Filter analysis.

Table 4
Mean and standard deviation (std.) values of latencies for each step and for the whole model. Some steps are presented along with their acronyms.

Step	Mean (ms)	Std. (ms)
Data acquisition	6.8	1.5
Rotation Compensator (RC)	1.4	0.2
Face detector	14.6	1.1
Face Aligner	1.6	0.1
ROI Extraction	1.6	0.2
Eye Eval (EE)	4.1	0.3
Classifier (ECC)	3.1	0.3
Filter	<0.01	<0.01
Whole	33.5	1.7

3.2.6. *Real-time performance*

A time performance experiment was also conducted in order to evaluate the latency of the models. The ECC+EE+RC model was used and a generic webcam live-captured the frames. The experiment took 3 min and, for each processed frame, the latencies of the whole model and of each individual step were computed. The steps listed here are the ones from Section 2. The average and standard deviation of the values collected are shown in Table 4. The experiment was conduced on an Ubuntu 20 machine with an i5-7200U processor and 8 GB of RAM.

At first, it can be noticed that the system is both stable and real-time, given the low standard deviation values and that its throughput lies between 28 and 31 frames per second (33.5 ± 1.7 ms). Secondly, the most costly steps were the data acquisition and Face Detection steps, requiring respectively 20,3% and 43.6% of the total time. While the former is dependent on the operational system, drivers, and hardware specifications and is currently out of scope for optimization, the latter is still an open task in the literature and can be optimized.

Furthermore, the proposed auxiliary models in this work (RC and sliding-window filters) do not compromise the real-time property of the system. Altogether, these models add up to only 16.4% of the total processing time while delivering a more robust system, therefore being a feasible upgrade to the base model.

Also, since both the EE and ECC steps are using the same CNN architecture (as described in Section 2.5), similar time costs can be found for both steps. ECC, however, may be less costly than EE, since some eye-patch candidates may be disregarded (and thus not used as input for the ECC model) due to their low scores as eye-patches.

3.2.7. *Comparison to other works*

Different works have tested their models on the same datasets. Nevertheless, as already exposed (Section 3.2.2), these works may present different annotation patterns, definitions or even evaluation methodologies — with such differences being mainly motivated by having different objectives. These differences inhibit an exact comparison between works. With that in mind, Table 5 displays results from different works that might still prove useful to the scope of this paper.

As examples of these differences, Fogelton and Benesova (2016, 2018) focused on the monitoring of the dry eye syndrome, hence taking in consideration incomplete blinks, and not focusing on voluntary blinks. Concerning evaluation methodologies, different techniques from the IoU method have been used, such as a peak-based method (Al-gawwam & Benaissa, 2018) or the ones described in Soukupová and Cech (2016) or Drutarovsky and Fogelton (2014). Moreover, some works failed to describe their evaluation procedure, ground truth, and other aspects, as pointed by Fogelton and Benesova (2018).

The experiments conducted here, as previously mentioned, used the annotations provided in Fogelton and Benesova (2016) and also the IoU evaluation methodology with the same threshold as that in Fogelton and Benesova (2016).

Finally, while some works report only the eye-blink detection results (Fogelton & Benesova, 2016, 2018), the experiments here conducted involve face detection and alignment, eye-patch extraction, and

Table 5
Comparison of best F1-Scores between different works under different experimental setups and criteria.

Work	Eyeblink8	ZJU	Talking Face
Anas et al. (2017)	–	0.937	1
Soukupová and Cech (2016)	0.952	0.952	0.948
Radlak and Smolka (2013)	–	0.992	–
Fogelton and Benesova (2018)	0.913	0.976	0.971
Our work	0.8697	0.9237	0.95

the validation of the proposed auxiliary models. Consequently, the system presented in this study is end-to-end and is evaluated as such.

4. Final considerations

A robust, real-time, and low-cost system for eye-blink detection was presented. This system can be integrated with an Auxiliary Communication System, emitting the detected blinks and their durations as communication signals. The eye-blink detection system was built containing a robust pipeline, well-designed auxiliary models (Eye-Evaluator, Rotation Compensator), and a moving filter.

The ABD video dataset used for the eye-blink detection task (following the criteria described in the study) and the YEC image dataset for the eye-state classification task were built and will be published in order to contribute to the academic community.

Furthermore, experiments showing the real-time and low error properties of the system were conducted. The results show that the proposed extensions positively contribute to the system.

We thus conclude that our system meets the proposed requirements within its application context. By associating high quality, low-cost and free availability, we expect our proposal to be an attractive plugin for ACSs that use blinks as activation signal. An array of benefits result from improving the communication abilities of ALS patients:

- Improving quality of life of both patients and caregivers
- Maintenance of cognitive skills of the patients
- Contribution with the independence of patients in advanced ALS stages

With regard to future work, some points that can be considered are:

- Different alternatives for eye-patch extraction, avoiding the usage of landmarks
- Enhancement of the RC model
- Experiments with ALS patients
- Implementation of concurrency techniques focusing on latency optimization

CRediT authorship contribution statement

Paulo Augusto de Lima Medeiros: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft. **Gabriel Vinícius Souza da Silva:** Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft. **Felipe Ricardo dos Santos Fernandes:** Conceptualization, Methodology, Investigation, Writing – review & editing. **Ignacio Sánchez-Gendriz:** Conceptualization, Methodology, Investigation, Writing – review & editing. **Hertz Wilton Castro Lins:** Conceptualization, Methodology, Writing – review & editing, Investigation. **Daniele Montenegro da Silva Barros:** Conceptualization, Methodology, Writing – review & editing, Investigation, Supervision. **Danilo Alves Pinto Nagem:** Conceptualization, Methodology, Writing – review & editing, Investigation, Supervision. **Ricardo Alexsandro de Medeiros Valentim:** Conceptualization, Methodology, Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We kindly thank the Laboratory of Technological Innovation in Health (LAIS) of the Federal University of Rio Grande do Norte (UFRN) and Ministry of Health Brazil for supporting the research.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2021.116073.

References

Ahmad, R., & Borole, J. (2015). Drowsy driver identification using eye blink detection. *IJISET - International Journal of Computer Science and Information Technologies*, *6*, 270–274.

Al-gawwam, S., & Benaissa, M. (2018). Robust eye blink detection based on eye landmarks and savitzky–golay filtering. *Information*, *9*, http://dx.doi.org/10.3390/info9040093, URL https://www.mdpi.com/2078-2489/9/4/93.

Anas, E. R., Henríquez, P., & Matuszewski, B. (2017). Online eye status detection in the wild with convolutional neural networks. In *VISIGRAPP*.

Bacivarov, I., Ionita, M., & Corcoran, P. (2008). Statistical models of appearance for eye tracking and eye-blink detection and measurement. *IEEE Transactions on Consumer Electronics*, *54*, 1312–1320.

Barbalho, I., Valentim, R., Júnior, M. D., Barros, D., Júnior, H. P., Fernandes, F., et al. (2021). National registry for amyotrophic lateral sclerosis: a systematic review for structuring population registries of motor neuron diseases. *BMC Neurology*, *21*, 269. http://dx.doi.org/10.1186/s12883-021-02298-2.

Bauer, G., Gerstenbrand, F., & Rumpl, E. (1979). Varieties of the locked-in syndrome. *Journal of Neurology*, *221*, 77–91. http://dx.doi.org/10.1007/BF00313105.

Bekios-Calfa, J., Buenaposada, J. M., & Baumela, L. (2014). Robust gender recognition by exploiting facial attributes dependencies. *Pattern Recognition Letters*, *36*, 228–234.

Belhumeur, P. N., Jacobs, D. W., Kriegman, D. J., & Kumar, N. (2013). Localizing parts of faces using a consensus of exemplars. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*, 2930–2940.

Chau, M., & Betke, M. (2005). *Real time eye tracking and blink detection with USB cameras: Technical report*, Boston, MA 02215, USA: Boston University Computer Science.

Danisman, T., Bilasco, I. M., Djeraba, C., & Ihaddadene, N. (2010). Drowsy driver detection system using eye blink patterns. In *2010 international conference on machine and web intelligence* (pp. 230–233).

Divjak, M., & Bischof, H. (2008). Real-time video-based eye blink analysis for detection of low blink-rate during computer use. In *First international workshop on tracking humans for the evaluation of their motion in image sequences* (pp. 99–107).

Divjak, M., & Bischof, H. (2009). Eye blink based fatigue detection for prevention of computer vision syndrome. In *MVA2009 IAPR conference on machine vision applications, Yokohama, Japan* (pp. 350–353).

Drutarovsky, T., & Fogelton, A. (2014). Eye blink detection using variance of motion vectors. In *European conference on computer vision* (pp. 436–448). Springer.

Ephrat, A., Mosseri, I., Lang, O., Dekel, T., Wilson, K., Hassidim, A., et al. (2018). Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. arXiv preprint arXiv:1804.03619.

Everingham, M., Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vision*, *88*, 303–338. http://dx.doi.org/10.1007/s11263-009-0275-4.

Fathi, A., & Abdali-Mohammadi, F. (2015). Camera-based eye blinks pattern detection for intelligent mouse. *Signal, Image and Video Processing*, *9*, 1907–1916. http://dx.doi.org/10.1007/s11760-014-0680-1.

Fernandes, F., Barbalho, I., Barros, D., Valentim, R., Teixeira, C., Henriques, J., et al. (2021). Biomedical signals and machine learning in amyotrophic lateral sclerosis: a systematic review. *BioMedical Engineering OnLine*, *20*, 61. http://dx.doi.org/10.1186/s12938-021-00896-2.

Fogelton, A., & Benesova, W. (2016). Eye blink detection based on motion vectors analysis. *Computer Vision and Image Understanding*, *148*, 23–33.

Fogelton, A., & Benesova, W. (2018). Eye blink completeness detection. *Computer Vision and Image Understanding*, *176*, 78–85.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press, http://www.deeplearningbook.org.

Gu, Z., Cheng, J., Fu, H., Zhou, K., Hao, H., Zhao, Y., et al. (2019). Ce-net: Context encoder network for 2d medical image segmentation. *IEEE Transactions on Medical Imaging*, *38*, 2281–2292.

Güell, M. R., Antón, A., Rojas-García, R., Puy, C., Pradas, J., et al. (2013). Comprehensive care of amyotrophic lateral sclerosis patients: a care model. *Archivos de BronconeumologíA*, *49*, 529–533, (English ed.).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. CoRR, arXiv:1502.01852.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., et al. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. CoRR, URL http://arxiv.org/abs/1704.04861.

Huang, T. (1981). Two-dimensional digital signal processing II: Transforms and median filters.

Karam, C. Y., Paganoni, S., Joyce, N., Carter, G. T., & Bedlack, R. (2016). Palliative care issues in amyotrophic lateral sclerosis: an evidenced-based review. *American Journal of Hospice and Palliative Medicine®*, *33*, 84–92.

Kazemi, V., & Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1867–1874).

Kiernan, M. C., Vucic, S., Cheah, B. C., Turner, M. R., Eisen, A., Hardiman, O., et al. (2011). Amyotrophic lateral sclerosis. *The Lancet*, *377*, 942–955. http://dx.doi.org/10.1016/S0140-6736(10)61156-7, URL http://www.sciencedirect.com/science/article/pii/S0140673610611567.

King, D. E. (2009). Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, *10*, 1755–1758.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Królak, A., & Strumiłło, P. (2012). Eye-blink detection system for human–computer interaction. *Universal Access in the Information Society*, *11*, 409–419. http://dx.doi.org/10.1007/s10209-011-0256-6.

Kwon, K.-A., Shipley, R., Edirisinghe, M., Ezra, D., Rose, G., Best, S., et al. (2013). High-speed camera characterization of voluntary eye blinking kinematics. *Journal of the Royal Society, Interface / the Royal Society*, *10*, Article 20130227. http://dx.doi.org/10.1098/rsif.2013.0227.

Le, V., Brandt, J., Lin, Z., Bourdev, L., & Huang, T. S. (2012). Interactive facial feature localization. In *European conference on computer vision* (pp. 679–692). Springer.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

Li, Y., Chang, M., & Lyu, S. (2018). In ictu oculi: Exposing AI created fake videos by detecting eye blinking. In *2018 IEEE international workshop on information forensics and security* (pp. 1–7).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft COCO: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), *Computer vision* (pp. 740–755). Cham: Springer International Publishing.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. E., Fu, C., et al. (2015). SSD: single shot MultiBox detector. CoRR, URL http://arxiv.org/abs/1512.02325.

Maior, C. B. S., Moura, M. C., ao M. M. de Santana, J., do Nascimento, L. M., Macedo, J. B., Lins, I. D., et al. (2018). Real-time SVM classification for drowsiness detection using eye aspect ratio. Probabilistic safety assessment and management PSAM 14..

Martinez, A., & Du, S. (2012). A model of the perception of facial expressions of emotion by humans: Research overview and perspectives. *Journal of Machine Learning Research*, *13*, 1589–1608.

Mikhail, M., & e. Kaliouby, R. (2009). Detection of asymmetric eye action units in spontaneous videos. In *2009 16th IEEE international conference on image processing* (pp. 3557–3560).

Morris, T., Blenkhorn, P., & Zaidi, F. (2002). Blink detection for real-time eye tracking. *Journal of Network and Computer Applications*, *25*, 129–143. http://dx.doi.org/10.1006/jnca.2002.0130, URL http://www.sciencedirect.com/science/article/pii/S108480450290130X.

Narmadha, R., Mythili, T., & Nivetha, R. (2014). Real time HCI using eye blink detection. *Internationa Journal of Computer Science and Mobile Computing*, *3*.

Nguyen, T., Nguyen, T. H., Truong, K. Q. D., & Van Vo, T. (2013). A mean threshold algorithm for human eye blinking detection using EEG. In V. V. Toi, N. B. Toan, T. Q. Dang Khoa, & T. H. Lien Phuong (Eds.), *4th international conference on biomedical engineering in Vietnam* (pp. 275–279). Berlin, Heidelberg: Springer Berlin Heidelberg.

Pan, G., Sun, L., Wu, Z., & Lao, S. (2007). Eyeblink-based anti-spoofing in face recognition from a generic webcamera. In *2007 IEEE 11th international conference on computer vision* (pp. 1–8).

Panning, A., Al-Hamadi, A., & Michaelis, B. (2011). A color based approach for eye blink detection in image sequences. In *2011 IEEE international conference on signal and image processing applications* (pp. 40–45).

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., et al. (2017). Automatic differentiation in PyTorch.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Picot, A., Charbonnier, S., Caplier, A., & Vu, N.-S. (2012). Using retina modelling to characterize blinking: comparison between EOG and video analysis. *Machine Vision and Applications*, *23*, 1195–1208. http://dx.doi.org/10.1007/s00138-011-0374-4.

Radlak, K., & Smolka, B. (2013). Blink detection based on the weighted gradient descriptor. In R. Burduk, K. Jackowski, M. Kurzynski, M. Wozniak, & A. Zolnierek (Eds.), *Proceedings of the 8th international conference on computer recognition systems* (pp. 691–700). Heidelberg: Springer International Publishing.

Rangayyan, R. M. (2015). *Biomedical signal analysis*. John Wiley & Sons.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779–788).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).

Rosa Silva, J. P., Júnior], J. B. S., dos Santos, E. L., [de Carvalho], F. O., de França Costa, I. M. P., & de Mendoņca, D. M. F. (2020). Quality of life and functional independence in amyotrophic lateral sclerosis: A systematic review. *Neuroscience & Biobehavioral Reviews*, *111*, 1–11. http://dx.doi.org/10.1016/j.neubiorev.2019.12.032, URL http://www.sciencedirect.com/science/article/pii/S0149763418309448.

Rousseau, M.-C., Baumstarck, K., Alessandrini, M., Blandin, V., Billette de Villemeur, T., & Auquier, P. (2015). Quality of life in patients with locked-in syndrome: Evolution over a 6-year period. *Orphanet Journal of Rare Diseases*, *10*, 88. http://dx.doi.org/10.1186/s13023-015-0304-z.

Sagonas, C., Antonakos, E., Tzimiropoulos, G., Zafeiriou, S., & Pantic, M. (2016). 300 faces in-the-wild challenge: Database and results. *Image and Vision Computing*, *47*, 3–18.

Shorten, C., & Khoshgoftaar, T. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*, 1–48.

Singh, H., & Singh, J. (2018). Real-time eye blink and wink detection for object selection in HCI systems. *Journal on Multimodal User Interfaces*, *12*, 55–65. http://dx.doi.org/10.1007/s12193-018-0261-7.

Smith, E., & Delargy, M. (2005). Locked-in syndrome. *BMJ*, *330*, 406–409. http://dx.doi.org/10.1136/bmj.330.7488.406, URL https://www.bmj.com/content/330/7488/406.

Soltanpour, S., Boufama, B., & Wu, Q. J. (2017). A survey of local feature methods for 3D face recognition. *Pattern Recognition*, *72*, 391–406.

Song, F., Tan, X., Liu, X., & Chen, S. (2014). Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients. *Pattern Recognition*, *47*, 2825–2838. http://dx.doi.org/10.1016/j.patcog.2014.03.024, URL http://www.sciencedirect.com/science/article/pii/S0031320314001228.

Soriani, M.-H., & Desnuelle, C. (2017). Care management in amyotrophic lateral sclerosis. *Revue Neurologique*, *173*, 288–299.

Soukupová, T., & Cech, J. (2016). Eye blink detection using facial landmarks. In *21st computer vision winter workshop, rimske toplice, Slovenia* (pp. 1–8).

Tan, M., & Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. CoRR, URL http://arxiv.org/abs/1905.11946.

Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*, 376–380. http://dx.doi.org/10.1109/34.88573.

van Es, M. A., Hardiman, O., Chio, A., Al-Chalabi, A., Pasterkamp, R. J., Veldink, J. H., et al. (2017). Amyotrophic lateral sclerosis. *The Lancet*, *390*, 2084–2098. http://dx.doi.org/10.1016/S0140-6736(17)31287-4, URL http://www.sciencedirect.com/science/article/pii/S0140673617312874.

Wang, X. (2017). *Eye-blink detection based on SVM: Technical report*, Shanghai Jiao Tong University.

Wang, Z., Xu, J., Liu, L., Zhu, F., & Shao, L. (2019). Ranet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 3978–3987).

Xiong, S., Zhu, S., Ji, Y., Jiang, B., Tian, X., Zheng, X., et al. (2017). Iblink: Smart glasses for facial paralysis patients. In *Proceedings of the 15th annual international conference on mobile systems, applications, and services* (pp. 359–370). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3081333.3081343.

Yang, S.-W., Lin, C.-S., Lin, S.-K., & Lee, C.-H. (2013). Design of virtual keyboard using blink control method for the severely disabled. *Computer Methods and Programs in Biomedicine*, *111*, 410–418. http://dx.doi.org/10.1016/j.cmpb.2013.04.012, URL http://www.sciencedirect.com/science/article/pii/S016926071300134X.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

Zhu, Y., Cai, H., Zhang, S., Wang, C., & Xiong, Y. (2020). Tinaface: Strong but simple baseline for face detection. arXiv preprint arXiv:2011.13183.

Zuiderveld, K. (1994). Contrast limited adaptive histogram equalization. *Graphics Gems*, 474–485.