Mobile-Based Automated Eyelid Blink Detection and Movement Analysis: A Machine Learning Approach

Gustavo Adolpho Bonesso[1], Carlos Marcelo Gurjão de Godoy[1], Tammy Hentona Osaki[2], Midori Hentona Osaki[2], and Regina Célia Coelho[1](✉)

[1] Science and Technology Institute, Federal University of São Paulo, São José Dos Campos, SP, Brazil
rccoelho@unifesp.br

[2] Department of Ophthalmology and Visual Sciences, Paulista School of Medicine, Federal University of São Paulo, São Paulo, SP, Brazil

Abstract. Eye diseases pose a significant global health challenge, affecting millions of people and leading to substantial visual impairment. Analyzing eyelid blinking and movement is crucial for monitoring patients with abnormal eyelid motions. This paper introduces the so-called Bapp (Blink Application), a mobile application designed to evaluate eyelid movement by recording the opening and closing of the eyes over time and detecting eye blinks using a machine-learning approach. The application processes pre-recorded videos and then validates the results against publicly available datasets. The proposed system operates in a unified stage, utilizing the pre-recorded video as input and evaluating the degree of eyelid openness in each frame using predictions from Google ML Kit, a machine learning-based framework integrated into the Flutter platform. The results are stored in a local database and can be exported to Excel for further analysis. Developed on the Flutter platform, the Bapp supports multiple languages, ensuring accessibility to a broad audience. The blink prediction results align with those obtained from other methods applied to the same dataset, demonstrating the app´s effectiveness in objectively monitoring the clinical progression of patients with eyelid diseases.

Keywords: Eyelid Movement · Blink Detection · Mobile Health Application

1 Introduction

Eye diseases are a primary global health concern, impacting millions of people and leading to significant visual impairment. According to the World Health Organization (WHO), at least 2.2 billion people experience vision impairment, with at least 1 billion cases being preventable or treatable.[1]

[1] World Health Organization (2023) Blindness and vision impairment, https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
O. Gervasi et al. (Eds.): ICCSA 2025, LNCS 15648, pp. 140–157, 2025.
https://doi.org/10.1007/978-3-031-97000-9_9

It is essential to analyze eyelid blinking and movement to monitor patients with abnormal eyelid movements. Objectively assessing these movements involves capturing images of the patient's face and processing them using specialized software to recognize and quantify eyelid movements [1–4]. Unfortunately, such approaches often involve complex systems impractical for clinical settings. Besides, they are frequently imprecise, especially for patients with abnormal eyelid movements, and fail to capture other critical parameters, such as amplitude movements [1–4].

Image capture can occur by using external cameras [2, 3] or the built-in cameras of mobile devices [1, 4]. Subsequent analysis can occur using computational tools such as MATLAB to detect and measure eyelid movement [2]. Alternatively, web-based platforms accessed via mobile devices or desktops can also facilitate such studies [5].

This work introduces a mobile application designed to evaluate eyelid movement by recording the opening and closing of the eyes over time. The application leverages a machine learning approach, focusing on the pre-recorded video feature, which validates the app's results against a public dataset [6].[2] The tool provides a robust framework for analyzing the frequency and amplitude of normal and pathological eyelid movements, offering a scalable and accessible solution for clinical and research applications.

2 Related Works

Advancements in technology have significantly enhanced the analysis of eyelid blinking and movement, particularly for patient monitoring in clinical settings. Techniques such as high-speed imaging, deep learning, and smartphone videography have emerged, providing precise and non-invasive methods to assess blink dynamics and their implications for various ocular conditions.

High-speed imaging combined with digital image correlation (DIC) allows for detailed measurement of eyelid motion during blinking. This method captures kinematic data, including blink duration, eyelid displacements, and peak velocities, facilitating a comprehensive analysis of spontaneous and reflex blinks [7].

Intelligent vision measurement systems utilizing deep learning can analyze blinking characteristics, offering insights into the visual function of patients, particularly those with dry eye disease. These systems automatically calculate blink completeness and provide stable, accurate measurements, enhancing clinical assessments [8, 9].

Smartphone videography has proven effective in capturing eyelid movements, allowing for comparative analysis of blink dynamics in patients with conditions like chronic progressive external ophthalmoplegia and facial nerve palsy. This method is cost-effective and accessible, making it suitable for both clinical and research applications [10].

There are limited integrated mobile apps for objectively analyzing eyelid movements in the present work approach. The DryEyeRhythm app, which diagnoses dry eye disease, was developed in Japan through a consignment agreement with Juntendo University Graduate School of Medicine and InnoJin Inc., Tokyo [11].

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

The EyeScore mobile application, developed by the Department of Clinical Research at the Westview Eye Institute in San Diego, CA, USA, utilizes blink rates and patterns as early clinical biomarkers for dry eye disease. Although designed for iOS, it is still unavailable for download [12].

Another approach involves hosting the application on a cloud server. The videos are recorded locally with a camera or smartphone and sent to a web server that processes them and reports the eyelid movement results. One project that utilized this technology employed Streamlit to host the application [1].

While all these technological advancements offer significant benefits in monitoring eyelid movement, challenges remain in standardizing these methods across diverse clinical settings and ensuring their widespread adoption. Further research is needed to validate these technologies' effectiveness and reliability in various patient populations.

3 Proposed Method

For clarity, Fig. 1 illustrates the steps and tools used in the proposed method. To validate the mobile application Bapp (Blink Application), we used videos from two reference datasets as input (Eyeblink8 and Talking Face), which have annotations indicating where the blinks occur. These videos were processed by Bapp using the feature for pre-recorded video analysis. This feature analyzes each frame from the video to assess the openness of each eye using Google ML Kit (a machine learning tool) and stores the results in Hive (a local database). The next step in validation is to export the raw analysis results from Bapp to Excel, enabling comparison with the annotations from the reference datasets to check Bapp's efficiency in detecting blinks in the input videos. Bapp addresses the limitations associated with the clinical application of eyelid movement monitoring by providing an automated, precise, and clinically feasible solution. The system performs eyelid movement analysis in a unified, integrated stage.

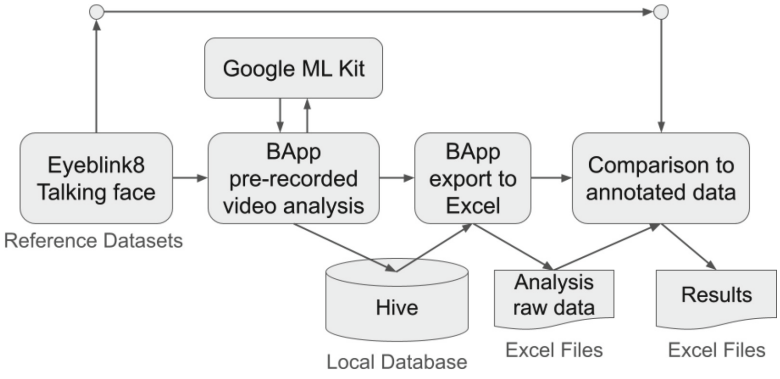


Fig. 1. Diagram with steps and tools used to analyze eyelid movements from dataset reference videos and compare the results from the Bapp mobile application to the annotated data from the datasets

3.1 Eyeblink Detection Dataset

This work used Eyeblink8 [6] and Talking Face[3], two public eyeblink detection datasets. The Eyeblink8 dataset contains over 82,600 frames and 408 blinks. It includes natural facial movements, expressions, and other intensive non-blink motions. The Talking Face dataset comprises 5,000 frames, corresponding to approximately 200 s of recording, capturing a subject engaged in a natural conversation with another person. The video recording uses a static camera to model facial behavior during the interaction. The dataset includes 61 additional blinks, raising the total blinks annotated to 469.

The annotations for each video are provided in two files: the first lists the frame numbers and their acquisition times, while the second contains manually annotated eye states.

Each row in the annotation file contains the following information: Frame ID, Blink ID, NF, LE_FC, LE_NV, RE_FC, RE_NV.

- Frame ID: Frame counter
- Blink ID: A unique blink ID; the eye blink interval definition occurs as a sequence of frames with the same blink ID.
- No frontal face (NF): An eye blink occurs when the subject looks sideways, given that the variable changes from X to N.
- Left eye (LE), right eye (RE), face (F)
- Eye fully closed (FC): If the subject's eyes are closed from 90% to 100%, the flag changes from X to C.
- Eye not visible (NV): While the subject's eye is not visible because of the hand, bad light conditions, hair, or even too fast head movement, this variable changes from X to N.

3.2 Software Tools

The Bapp mobile application was developed using the Flutter development tool.[4] It enables the creation of cross-platform mobile applications for Android and iOS operating systems using the same codebase, thus covering nearly the entire smartphone user base. The application supports three languages—Portuguese, English, and Spanish—by utilizing Flutter's localization library, which ensures a broad and diverse audience.

Clean architecture is the chosen architectural framework for guiding the development of Bapp mobile applications. It offers advantages such as improved maintainability, reduced complexity, streamlined codebase modifications, and the development of robust, easily upgradable, and verifiable applications. Those features enhance software quality and adaptability to evolving requirements and technological advancements [13].[5]

Concerning the analysis of the eyelid opening states, the framework relies on using Google ML Kit, an application programming interface (API) for computer vision developed by Google.[6] We used a plugin to integrate Google ML Kit into the Flutter platform.

[3] Talking Face Video, https://personalpages.manchester.ac.uk/staff/timothy.f.cootes/data/talking_face/talking_face.html.

[4] Flutter Flutter Developer Tools, https://flutter.dev.

[5] Martin R (2012) The Clean Architecture, https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html.

[6] Google ML Kit | Google for Developers, https://developers.google.com/ml-kit/guides.

Optimized for mobile devices, the Google ML Kit performs all processing locally on the device. The specific Google ML Kit module used in this task is Face Detection. It enabled the detection of eye openness probability, eye contours, and other facial features to recognize the state of eyelid opening.

The Google ML Kit receives an image as input and performs face detection. Even though the framework can identify multiple faces within a single image, the Bapp mobile application considers only the first detected face, as the app analyzes the user individually. After detecting a face, the framework proceeds to classify the status of the eyes. This classification provides a probability value indicating the degree of openness for each eye.

The selected local database is Hive, a NoSQL solution tailored to the local storage needs of Flutter applications.[7] Hive integrates seamlessly with Flutter and offers a robust data storage and retrieval API. The Hive database allows information to be stored using JSON format.

3.3 Eye Blink Detection Algorithm

The interpretation of raw data from Google ML Kit detects the frames where a blink occurred. The first approach to detecting eye blinks involved determining a blink when the probability of eye openness drops below 0.5, where "0.0" represents "totally closed eye" and "1.0" refers to "totally opened eye". Figure 2 illustrates a problem with this method: When the probability bounces around probability 0.5 - falling below 0.5, rising above 0.5, and then falling again below 0.5 - the algorithm counts two blinks, which is incorrect, as only one blink or no blink would have happened.



Fig. 2. Illustration of the problem of eye openness probability chart with two frames below probability 0.5.

Using a single threshold to indicate a blink is not a practical solution. To solve this problem, considering two distinct thresholds - one to signal the start of the blink and another to mark its end – achieves better outcomes. Implementing a third threshold further differentiates between partial and complete blinks. Figure 3 illustrates these thresholds.

[7] Isar Database Hive, https://github.com/isar/hive.

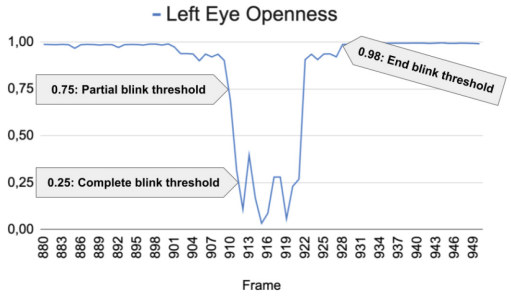


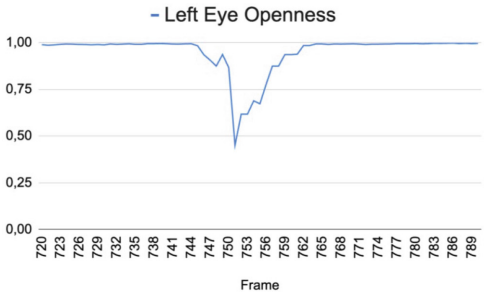
Fig. 3. Illustration of probability charts with thresholds indicating the beginning and ending of partial or complete blinks.

The pseudocode for eye blink detection is in Fig. 4.

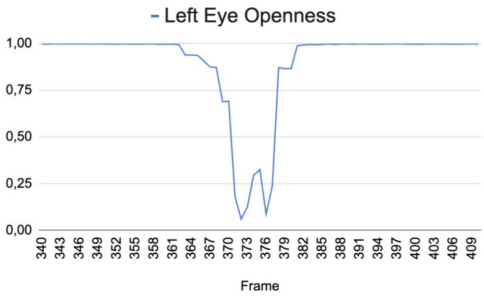
```
BEGIN

    DEFINE partial_blink_detected AS FALSE
    DEFINE full_blink_detected AS FALSE

    IF eye_open_probability < 0.75 THEN
        IDENTIFY partial_blink
        SET partial_blink_detected TO TRUE
    END IF

    IF eye_open_probability < 0.25 THEN
        IDENTIFY full_blink
        SET full_blink_detected TO TRUE
    END IF

    IF eye_open_probability > 0.98 AND
      (partial_blink_detected OR full_blink_detected) THEN
        END blink
        SET partial_blink_detected TO FALSE
        SET full_blink_detected TO FALSE
    END IF

END
```

Fig. 4. Pseudocode for blink detection algorithm

Figure 5(a) illustrates an instance of a partial blink, where the eye openness probability decreases below 0.75 but remains above the 0.25 threshold, which characterizes a complete blink. The blink is classified as complete upon the eye openness probability returning to 0.98. Figure 5(b) demonstrates a complete blink when the eye openness probability drops below 0.25.



(a)



(b)

Fig. 5. Illustration of probability charts of partial blink (a) and complete blink (b)

3.4 Pre-recorded Video Processing

The first step involves dividing the video into frames for individual analysis. For each frame, the Google ML Kit detects faces. When face detection occurs, it calculates the likelihood of the eyes being open, which ranges from 0 (closed) to 1 (open). The local database registers the data for later analysis of the start and end of the blinks. Figure 6 illustrates the sequence of processing the pre-recorded video.

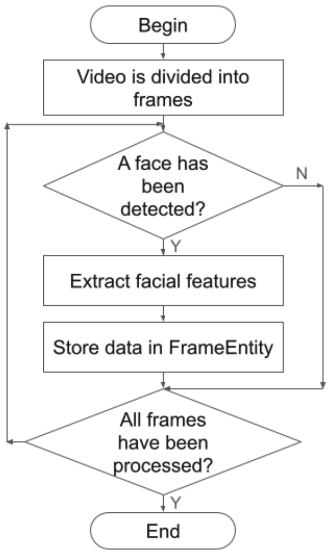


Fig. 6. Pre-recorded video processing sequence

For each analysis, the Bapp stores date and time information, duration, and an array of details from each frame within an AnalysisEntity. Consequently, the FrameEntity operates as a sub-entity of the AnalysisEntity, with every frame documenting the frame number, date, time, and the detected probability of eye-opening. Figure 7 shows an example with data from the AnalysisEntity and FrameEntity.

AnalysisEntity

```
{
  "analysisDateTime": "2024-04-23T11:21:30.0",
  "duration": 5.0,
  "frameList": [
    {
      "frameNumber": 0,
      "frameDateTime": "2024-04-23T11:21:30.0",
      "rightEyeOpenness": 0.9956856369972229,
      "leftEyeOpenness": 0.998312771320343
    },
    ...
    {
      "frameNumber": 124,
      "frameDateTime": "2024-04-23T11:21:34.96",
      "rightEyeOpenness": 0.898965235782322,
      "leftEyeOpenness": 0.80787288723898928
    }
  ]
}
```

FrameEntity

FrameEntity

Fig. 7. Analysis example with data from the AnalysisEntity and FrameEntity

3.5 Excel Export

The export to Excel feature saves an analysis's raw data in an Excel sheet. This functionality is valuable for debugging and provides flexibility in viewing data. It also allows sorting, filtering, and performing operations on the raw data to understand the analysis's results better.

The Excel export feature is crucial for comparing the dataset's video analysis results with the annotated blink information to calculate metrics such as RMSE and precision.

As shown in Fig. 8, this feature is accessed via the Export to Excel button on the Analysis Detail screen.

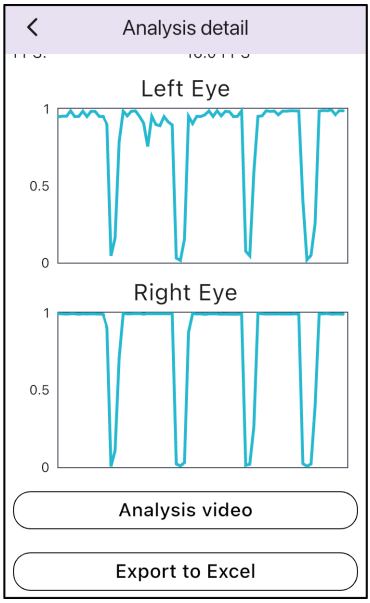


Fig. 8. A screenshot of the Bapp mobile application illustrating the analysis detail screen, with the "Export to Excel" button at the bottom.

Figure 9 shows the exported Excel sheet containing information about each frame's overall analysis and eye status. This information allows the user to work more flexibly with the raw data and possibly gain more insights.

	A	B	C	D	E
1	Patient	Gustavo Bonesso			
2	Analysis type	AnalysisType.realTime			
3	*** Pre/Post Inte	PrePostIntervention.pre			
4	*** Days Pre/Pos	15			
5	*** Observations				
6	Duration	4.4 seconds			
7	Number of Blinks	L: 4	R: 4		
8	Blink Frequency:	L: 54.1 BPM	R: 54.1 BPM		
9	FPS:	16.0 FPS			
10					
11	***Frame	***Right Eye Ope	***Left Eye Oper	***Right Eye EAI	***Left Eye EAR
12	0	0,99	0,95	0,25	0,26
13	1	0,99	0,95	0,26	0,28
14	2	0,99	0,95	0,25	0,28
15	3	0,99	0,99	0,28	0,28
16	4	0,99	0,95	0,28	0,29
17	5	0,99	0,95	0,26	0,28

Fig. 9. Excel sheet displaying raw data from the analysis

3.6 Validation

The app processes videos from the EyeBlink8 and Talking Face public datasets to validate its efficiency in detecting blinks. The eye blink is considered detected if there is any intersection between the detected blink and the annotation. The intersection interval with the ground truth is counted just one as a True Positive. The results are exported to Excel spreadsheets and compared using the root mean square error (RMSE). The results were normalized using the StandardScaler function and evaluated using the evaluation metric 'mean_squared_error', both from Scikit-Learn.[8]

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y - \hat{y})^2} \tag{1}$$

where ŷ is a vector with the predicted values, and y is the ground truth value for each value analyzed. The result is the root of the difference between the predicted and the ground truth values divided by the number of values (value represented by n).

To compare the results to other papers using the same datasets, we used the following equations:

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

$$Recall(TPrate) = \frac{TP}{TP + FN} \tag{3}$$

$$FPrate = \frac{FP}{N} \tag{4}$$

[8] scikit-learn - Machine Learning in Python, https://scikit-learn.org/

$$MA = \frac{TP + TN}{P + N} \tag{5}$$

where TP is True Positive (the eye blinks are annotated in the reference dataset and were predicted by the app), FP is False Positive (when the app predicts the eye blink, but it is not found in the annotated dataset), FN is False Negative (when an eye blink is annotated in the dataset but not predicted by the app). N is the number of frames with open eyes. MA is the mean accuracy. TN is the True Negative, calculated by subtracting the TP, FP, and FN frames from the total number of frames of the videos. P is calculated by subtracting the total number of frames from N.

4 Results

The analysis starts after selecting a video from the smartphone's gallery. Figure 10 shows the Bapp mobile application interface for processing pre-recorded videos.

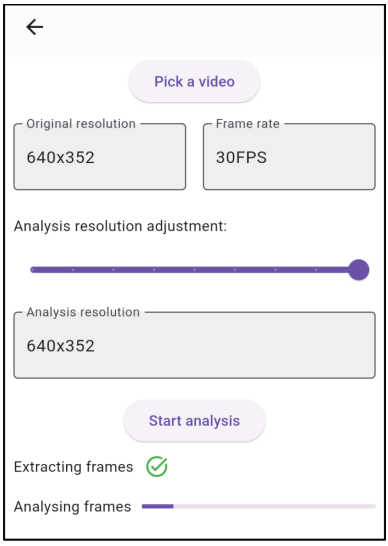


Fig. 10. Screenshot of the Bapp mobile application interface for processing pre-recorded videos.

After completing the analysis, Bapp displays the analysis detail screen shown in Fig. 11. This screen presents the analysis data, including the analysis date, duration, number of blinks detected, calculated blink frequency per minute, and a graphical representation for enhanced visual clarity.

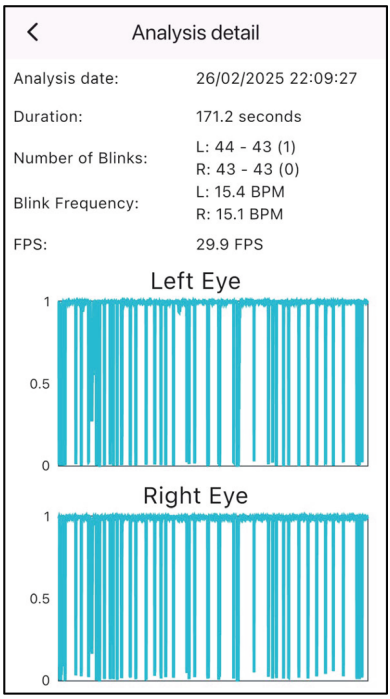


Fig. 11. Screenshot of the Bapp mobile application displaying the detailed analysis information.



Fig. 12. Frames sequence for the first blink in the Talking Face video

Figure 12 shows a frame sequence from one of the videos used to validate the first blink annotated in the dataset.

Table 1 shows the annotated reference for the first blink from the Talking Face video. It determines the beginning of the blink in the frame 168 and the end of the blink in the frame 176.

Table 1. Annotated frames for the first blink in the Talking Face video

Frame_ID	Blink_ID	NFF	LE_FC	LE_NV	RE_FC	RE_NV
166	-1	X	X	X	X	X
167	-1	X	X	X	X	X
168	1	X	X	X	X	X
169	1	X	X	X	X	X
170	1	X	C	X	C	X
171	1	X	C	X	C	X
172	1	X	X	X	X	X
173	1	X	X	X	X	X
174	1	X	X	X	X	X
175	1	X	X	X	X	X
176	1	X	X	X	X	X
177	-1	X	X	X	X	X

Figure 13 shows the chart of openness probability for both eyes predicted by Google ML Kit for the first blink in the Talking Face video.

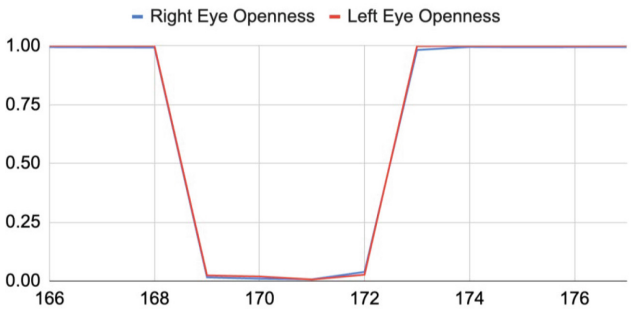


Fig. 13. Eye openness probability for both eyes predicted by Google ML Kit for the first blink in Talking Face video

The app's results for the number of eyeblinks predicted and annotated blinks from the datasets are shown in Table 2.

Table 2. The number of eye blinks predicted by the app compared to the annotated eye blinks

Video	Number of Annotated Eyeblinks	Number of Eyeblinks Predicted by the App
EyeBlink8 #1	38	37
EyeBlink8 #2	88	88
EyeBlink8 #3	65	61
EyeBlink8 #4	31	30
EyeBlink8 #5	30	30
EyeBlink8 #6	41	41
EyeBlink8 #7	72	72
EyeBlink8 #8	43	43
Talking Face	61	61
Total	469	463

The RMSE calculated for the dataset normalized comparison with the app prediction results was 0.0654. Considering the mean RMSE results, the model effectively predicted the blinks in the reference datasets.

Table 3 compares the results from our app to those of other papers using different methods against the same datasets.

Table 3. Comparison of the results from our app to those of other papers using different methods against the same datasets

	Dataset	Precision	Recall	FP rate	Mean accuracy
Divjak & Bischof [14]	Talking	-	95.0%	19.0%	88.0%
Lee et al. [15]	Talking	83.3%	91.2%	-	-
Drutarovsky & Fogelton [6]	Talking	92.2%	96.7%	0.7%	99.0%
Our app	Talking	**95.3%**	**100.0%**	**0.5%**	**99.6%**
Drutarovsky & Fogelton [6]	EyeBlink8	79.0%	85.3%	0.7%	99.5%
Our app	EyeBlink8	**84.3%**	**98.5%**	**0.8%**	**99.2%**

The Precision metric of the EyeBlink8 dataset is lower than that of the Talking Face dataset due to a higher number of false positives (FPs). Occasionally, the app predicts partial blinks without corresponding annotated blinks in the dataset. Figure 14 illustrates a sequence of frames the app predicts as a partial blink, which are not annotated in the reference dataset, as the subject is looking down.



Fig. 14. A sequence of frames showing a False Positive case where the app predicts a partial blink

Figure 15 displays the eye openness chart for a false positive case, where the app predicts a partial blink. The openness of both the left and right eyes falls below 0.75, indicating the onset of a partial blink. The blink concludes when the openness rises to 0.98.

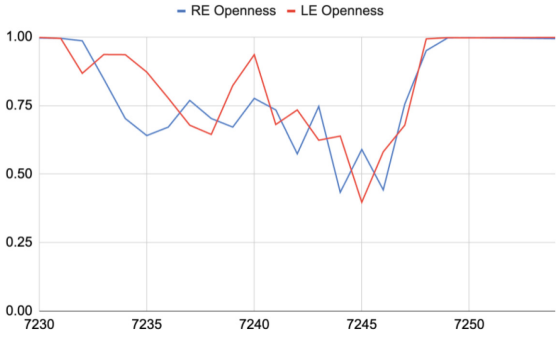


Fig. 15. Eye openness chart for a case of False Positive

5 Discussion

This work presents the results of Bapp, a mobile application with machine-learning capabilities that analyzes eyelid movements in pre-recorded videos. It achieved 95.3% precision in blink detection using the Talking Face reference dataset and 84.3% using the EyeBlink8 dataset. The precision level achieved by Bapp in the Talking Face dataset is higher than the 83.3% attained by Lee et al. [15] and the 92.2% achieved by Drutarovsky and Fogelton [6]. Additionally, the precision level achieved by Bapp in the blink analysis of the EyeBlink8 dataset surpasses the 79% obtained by Drutarovsky and Fogelton [6].

Bapp's recall also exceeds that of previous works. Bapp recalled 100% of the Talking Face dataset and 98.5% of the EyeBlink8 dataset. Divjak & Bischof [14] attained a recall of 95.0% for the Talking Face dataset, while Lee et al. [15] achieved 91.2%, and

Drutarovsky and Fogelton [6] reported 96.7%. For the Eye-Blink8 dataset, Drutarovsky and Fogelton [6] reported a recall of 85.3%.

These levels of precision and recall allow Bapp to analyze eyelid movements as an alternative to manually counting blinks, primarily facilitating the monitoring of abnormal eyelid movements.

The lack of annotated datasets specifically focused on eye diseases highlights a significant gap that must be addressed. Although the existing datasets are valuable for validation, developing one that aligns more closely with the Bapp context and objectives is essential to ensure a more robust evaluation framework. For future research, I recommend utilizing datasets designed explicitly for ocular disease detection and annotated by experts in the field. That could involve creating a new dataset encompassing various ocular conditions, including blepharospasm, hemifacial spasm, ptosis, and dry eye syndrome. Collecting data under diverse lighting conditions, camera angles, and subject demographics will enhance the model's robustness.

A recent search in the electronic databases revealed only two mobile applications with eyelid movement analysis capabilities: EyeScore [12] and DryEyeRhythm [11]. We cannot find validations of these mobile applications against blink reference datasets, such as the one presented in this work, and these validations are essential for comparing the software capabilities.

Bapp can perform the analysis in a single step using only a smartphone and does not require a computer, unlike the approaches by Hu et al. [9] and Osaki et al. [2], which utilize a computer with custom software written in MATLAB for analysis. Additionally, the method by Silkiss et al. [3] involves custom software developed by a third party, Visage Technologies, to extract facial features alongside another software written in MATLAB for calculating eyelid aperture. Hedayati Amlashi et al. [10] utilized images captured at a high frame rate using a smartphone and employed Kinovea, software designed for tracking markers in human motion analysis. Silva et al. [1] developed a different approach in Python and used the Streamlit library to process the captured video through a cloud server instead of a local computer. In contrast, Bapp processes the video locally, eliminating the need to transfer data to the cloud or maintain a server. The Bapp method streamlines the operational process and allows eyelid movement analysis to reach a broader audience.

This study has limitations. One limitation is that the time required for analysis depends on several factors, including the resolution of the acquired video (the higher the resolution, the more processing time is needed) and the device used for analysis. Devices with faster processors conduct the analysis more quickly but are generally more expensive than those with slower processors, which take longer to perform the same analysis. Another significant limitation affecting the analysis's quality is the illumination conditions and camera angles from which the images were captured. Optimal results can be achieved with good lighting and the subject facing directly toward the camera in a frontal position.

6 Conclusion

The Bapp (Blink Application) represents a significant advancement in the objective monitoring of eyelid movements and blink patterns, particularly for patients with eyelid-related diseases.

By leveraging machine learning through Google ML Kit and the Flutter platform, the application provides a unified, accessible, and multilingual solution for analyzing eyelid openness and blink detection.

Validation against publicly available datasets confirms the app's reliability and effectiveness, aligning with established methods. These results show Bapp's potential as an alternative to more resource-intensive methods using desktop computers and cloud servers.

Despite some limitations, the Bapp not only meets the need for accessible and accurate eyelid movement analysis but also provides clinicians with a practical and scalable tool to support continuous monitoring of disease progression and treatment effectiveness, thereby improving patient care. That facilitates earlier interventions and more personalized care strategies.

While some limitations remain, such as the availability and diversity of labeled training data, Bapp's success in real-world validation scenarios signals a promising future. Continued development should focus on expanding and diversifying annotated datasets, particularly those covering a wide range of ocular diseases and demographics, to further enhance diagnostic precision and clinical utility.

Overall, Bapp significantly contributes to digital ophthalmology, potentially enhancing patient engagement, remote monitoring, and clinical workflows in vision health care.

Conflict of Interest. The authors declare that they have no conflict of interest.

References

1. da Silva, J.L.S., de Godoy, C.M.G., Osaki, T.H., et al.: Mobile App for Assessing Hemifacial Spasm Treatment Response Using Machine Learning, pp. 197–206 (2024)
2. Osaki, M.H., Osaki, T.H., Garcia, D.M., et al.: An objective tool to measure the effect of botulinum toxin in blepharospasm and hemifacial spasm. Eur. J. Neurol. **27**, 1487–1492 (2020). https://doi.org/10.1111/ene.14258
3. Silkiss, R.Z., Koppinger, J., Truong, T., et al.: Cannabidiol as an adjunct to botulinum toxin in blepharospasm - a randomized pilot study. Transl Vis Sci Technol **12**, 17 (2023). https://doi.org/10.1167/tvst.12.8.17
4. Yabumoto, C., Osaki, M.H., Gameiro, G.R., et al.: Smartphone and custom-made software to assess the effect of botulinum toxin in essential blepharospasm: initial results. Eur. J. Neurol. **30**, 887–891 (2023). https://doi.org/10.1111/ene.15666
5. Da Silva, J.L.S.: Mobile App for Assessing Hemifacial Spasm Treatment Response Using Machine Learning
6. Drutarovsky, T., Fogelton, A.: Eye Blink Detection Using Variance of Motion Vectors, pp. 436–448 (2015)

7. Seamone, A., Shapiro, J.N., Zhao, Z., et al.: Eyelid motion tracking during blinking using high-speed imaging and digital image correlation. J Biomech Eng **147** (2025). https://doi.org/10.1115/1.4067082
8. Ren, Y., Wen, H., Bai, F., et al.: Comparison of deep learning-assisted blinking analysis system and Lipiview interferometer in dry eye patients: a cross-sectional study. Eye and Vision **11** (2024). https://doi.org/10.1186/s40662-024-00373-6
9. Hu, R., Sun, D., Shi, G., Pan, A.: Practical Application of Intelligent Vision Measurement System Based on Deep Learning **48**, 380–384 (2024). https://doi.org/10.12455/j.issn.1671-7104.230652
10. Hedayati Amlashi, N., Rajabi, M.T., Shafie, M., et al.: Videographic analysis of blink dynamics in patients with chronic progressive external ophthalmoplegia, myogenic ptosis, and facial nerve palsy using smartphone camera: a comparative analysis. Health Sci Rep **6** (2023). https://doi.org/10.1002/hsr2.1631
11. Okumura, Y., Inomata, T., Midorikawa-Inomata, A., et al.: DryEyeRhythm: a reliable and valid smartphone application for the diagnosis assistance of dry eye. Ocul. Surf. **25**, 19–25 (2022). https://doi.org/10.1016/j.jtos.2022.04.005
12. Zhang, S., Echegoyen, J.: Design and usability study of a point of care mhealth app for early dry eye screening and detection. J. Clin. Med. **12**, 6479 (2023). https://doi.org/10.3390/jcm12206479
13. Martin, R.: Clean Architecture: A Craftsman's Guide to Software Structure and Design, 1ra ed ed (2017)
14. Divjak, M., Bischof, H.: Eye blink based fatigue detection for prevention of Computer Vision Syndrome
15. Lee, W.O., Lee, E.C., Park, K.R.: Blink detection robust to various facial poses. J. Neurosci. Methods **193**, 356–372 (2010). https://doi.org/10.1016/j.jneumeth.2010.08.034