2023

On the Pursuit of Developer Happiness: Webcam-Based Eye Tracking and Affect Recognition in the IDE

Tamsin Rogers
Colby College

On the Pursuit of Developer Happiness:

Webcam-Based Eye Tracking and Affect Recognition in the IDE

An Honors Thesis

Presented to

The Faculty of the Department of Computer Science

Colby College

In partial fulfillment of the requirements for the

Degree of Bachelor of Arts

By

Tamsin S. Rogers

Honors Thesis Advisor: Naser Al Madi

Waterville, Maine

May 14, 2023

ABSTRACT

Recent research highlights the viability of webcam-based eye tracking as a low-cost alternative to dedicated remote eye trackers. Simultaneously, research shows the importance of understanding emotions of software developers, where it was found that emotions have significant effects on productivity, code quality, and team dynamics. In this paper, we present our work towards an integrated eye-tracking and affect recognition tool for use during software development. This combined approach could enhance our understanding of software development by combining information about the code developers are looking at, along with the emotions they experience. The presented tool utilizes an unmodified webcam to capture video of software developers while interacting with code. The tool passes each frame (Figure 4) to two modules, an eye tracking module that estimates where the developer is looking on the screen, and an affect recognition module that infers developer emotion from their facial expressions. The proposed work has implications to researchers, educators, and practitioners, and we discuss some potential use cases in this paper.

ACKNOWLEDGEMENTS

This project would not have been possible without my incredible advisor and professor, Naser Al Madi. I have learned so much from Professor Al Madi over the course of our three years doing eye-tracking research together, and I am so grateful to have had the opportunity to work with someone who is both motivated to investigate new topics in computer science and also willing to share that experience with students. Thank you for being the person to initially propose this idea for an honors thesis and for working through each phase of it with me as we constantly reimagined new ways to reach our goal. I'm excited to further our work on this topic in the next few weeks!

I must also thank my wonderful roommate who tolerated my testing of the emotion recognition module of this project on her for an untold number of hours. Thank you for making countless silly faces for me whenever I asked and for allowing me to publish and present them just about everywhere this project has touched.

Thank you also to the many friends and colleagues who have entertained my ramblings about this topic, supported me with their presence at various presentations this year, and kept me company as I attempted to calibrate this tool over and over again. I'm so excited to share my findings with you!

TABLE OF CONTENTS

INTRODUCTION

Previous research has demonstrated the usability of computer webcams to track eye movement, but the field lacks studies specific to the use of these eye movement patterns in improving the software development process as a whole. The accessibility and remote nature of webcam eye trackers promise a low-cost alternative to some of the higher-level models used to collect data about eye movement tendencies. In this study, we focus on analyzing patterns derived from empirical trials in webcam-based eye tracking. Building on GazeTracking, a Python library that returns the exact position of the pupils and gaze direction in real time in the form of a webcam-based eye tracking system, we developed a system that calibrates and validates a user's eye movement patterns against a linear regression gaze prediction model. We then connect these patterns to various affects, or general moods and feelings, of programmers in a range of provoking environments. Our results provide insights on the use of webcam eye trackers and gaze prediction models and motivate future work to improve the emotions and productivity of software developers in the workplace. In this paper, we present our work towards an integrated eye-tracking and affect recognition approach during software development. This combined approach can shed new light on our understanding of software development by combining information about the code developers are looking at with the experienced emotion.

LITERATURE REVIEW

Recent research in eye tracking has demonstrated that an unmodified webcam can provide a comparable performance to a dedicated eye tracker in spatial accuracy (Wisiecka 2022). Previous research has also shown that for certain kinds of tasks, emotions correlate with progress and biometric measures can be used to distinguish between emotions. Müller and Fritz (2015) conducted a lab study in which participants wearing biometric sensors periodically assessed their emotions and progress while completing programming tasks. Results indicated that the wide range of emotions experienced by developers is correlated with their progress on tasks.

Girgadi proposes two different views of emotion: the limited set and the continuous function, the latter of which is often used to refer to the affect one experiences in the realm of software development. The continuous function approach involves the Circumplex Model of Affect, which describes emotions as they relate to valence and arousal. In the context of software, developers report being “stuck” as the biggest reason for negative valence. Such negative valence can be categorized by feelings that come about as the result of fear of failure and difficulty working under time pressure, whereas positive valence often involves positive feelings resulting from successfully facing new challenges and being “in flow” (Gigardi 2020). We then seek to measure emotions and perceived progress in the software development process and determine causes for various affects, in addition to strategies for dealing with them.

Graziotin takes a people-first approach to achieving efficiency in the industry, focusing on the idea that a better mood leads to better work. Simply put, “happy software developers perform better than unhappy ones” (Graziotin 2014). This research explores the role of emotional affect on debugging performance, and suggests that these two factors are correlated.

Graziotin suggests that happier programmers will be more analytical, but acknowledges that such a hypothesis requires additional evidence. From two correlation studies, it was determined both that there existed a positive correlation between affects of developers (in terms of valence and arousal) associated to a programming task and their self-assessed productivity, and that “happy developers are indeed better problem solvers [than unhappy ones] in terms of their analytical abilities” (Graziotin 2014). Graziotin’s work also emphasized the need for studying human factors in software engineering with a multidisciplinary viewpoint.

RESEARCH PROBLEM & MOTIVATION

Software development requires both creativity and problem solving skills - personal characteristics which are both strongly tied to one's affect, or emotional experience as a whole. As our emotional awareness of self and others develops, it stands that affect can be both caused by the workplace and have an impact on the workplace. There then lies valuable information in the emotional state of software developers, which can be analyzed in efforts to improve collaborative software development processes altogether.

Understanding emotions in software development has been linked to important factors such as developer productivity, code quality, and team dynamics (Imran 2022, Fucci 2021, Graziotin 2018). Researchers have explored various methods and artifacts to understand developer emotions, including sentiment analysis of code review comments (Ahmed 2017), version control commit messages such as GitHub (Guzman 2014), and technical Q&A sites such as Stack Overflow (Novielli 2018, Calefato 2018).

One of the areas explored to understand developer emotions through biometric measures is correlating emotions to eye tracking patterns using a dedicated research eye tracker (Müller 2015). Where positive and negative emotions were classified based on eye movement over code. More recently, eye tracking research has demonstrated that an unmodified webcam can provide a comparable performance to a dedicated eye tracker in spatial accuracy (Wisiecka 2022).

This paper presents an approach to utilize an unmodified webcam for eye-tracking and affect recognition in software engineering tasks. This approach allows us to know the emotion experienced by the developer and where in the code they were looking when they experienced that emotion. While eye tracking through a webcam might be marginally less accurate than a dedicated eye tracker, it offers unique advantages in pervasiveness, no added cost, and ease of integration with other measures such as affect recognition.

METHODOLOGY

The presented tool, which is illustrated in detail in Figure 3, utilizes an unmodified webcam to capture software developers while interacting with code. The tool passes each frame to two modules, an eye tracking module that estimates where the developer is looking on the screen, and an affect recognition module that infers developer emotion from their facial expressions.

This study primarily involves the development of a gaze prediction model that will accurately visualize the intended gaze point of a user. The gaze tracking component of this tool is based on the GazeTracking[1] repository. Using OpenCV, we enhanced this tool and constructed a calibration model that takes in the position of the user's right pupil on the screen over the course of 10 calibration points on the screen. We found the most accurate calibration to occur at 10 calibration points, with accuracy trailing off at 15 points or higher (Figure 2). The model then validates the recorded pupil coordinates against the actual coordinates of the calibration points and predicts the user's gaze point by feeding it into a linear regression model. These experimental and actual coordinates are then compared and an average euclidean distance is calculated between the two such that the accuracy of the gaze prediction model can be determined (Figure 1). The program writes these coordinate and accuracy values to a personalized text file for each user.

The second dimension of this study involves an emotion detection neural network. This network is a MobileNet model (Nan et al. 2022) trained on 350 images of 7 different common emotions: angry, disgust, fear, happy, neutral, sad, and surprise. We prepared our data using an image data generator and fit the model to 10 steps per epoch, for a total of 20 epochs, with 8

[1] https://github.com/antoinelame/GazeTracking

validation steps between data points. The resulting model takes in an image and matches it to one of the 7 emotion classes based on the match with the greatest similarity value. The model was tested for accuracy with predefined pictures of each emotion, presented in Figure 6. The output of the model is presented in Figure 5. While the aforementioned gaze prediction model is running, a "most-likely" emotion is also being detected at each frame point, which is written to the user's personalized text file along with the predicted fixation coordinates (Figure 7).

The first version of this tool, the "Still-Frame Pupil Position Recognition" model, incorporated only a still-frame recognition of pupil position, in which the position of the user's pupil on the screen was recorded from a single shot. The second version of the tool, the "Real-Time Pupil Recognition" model, translated the pupil position on the screen to the user's real time pupil position in a similar manner. Version three, the "Continuous, Real-Time gaze Prediciton" model, utilized ten calibration and ten validation points to construct a gaze prediction model based on a linear regression that compared the recorded real-time pupil position to the actual points on the screen. The fourth version of the tool, the "Continuous Gaze Predicition + Emotion Recognition" model, was a continuous gaze prediction and emotion recognition that operated in real time with improved estimation accuracy. Gaze prediction accuracy was improved from a euclidean distance (pixel distance between actual and estimated gaze points) of about 800 an average of 83.67.

The final version of the tool will have a further improved accuracy measurement of predicted gaze. This will be achieved by replacing the current linear regression model with a neural network trained on series of 10 samples of calibration points. Essentially, 10 recordings of the user's gaze will be recorded at the time of each frame screen capture, as opposed to the 1 recording in the current verison. Although the current model can tell us the region of the screen the user is looking at, it is not accurate enough to tell us the token or line number - hence the need for a more precise tool. Once this is achieved, the pupil movement pattern data will be evaluated in a lab setting against the emotional affect data with the intent to observe a potential relation between the two.

RESULTS & FIGURES

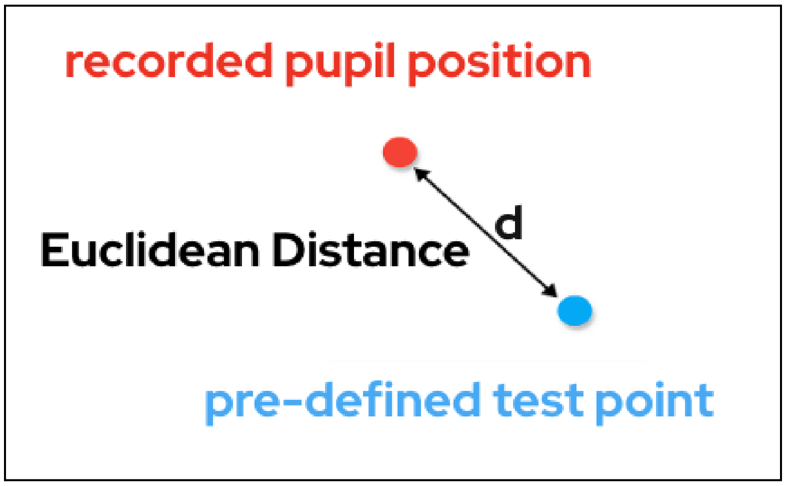


Figure 1: Gaze estimation calculated by euclidean distance between actual and experimental model calibration points.

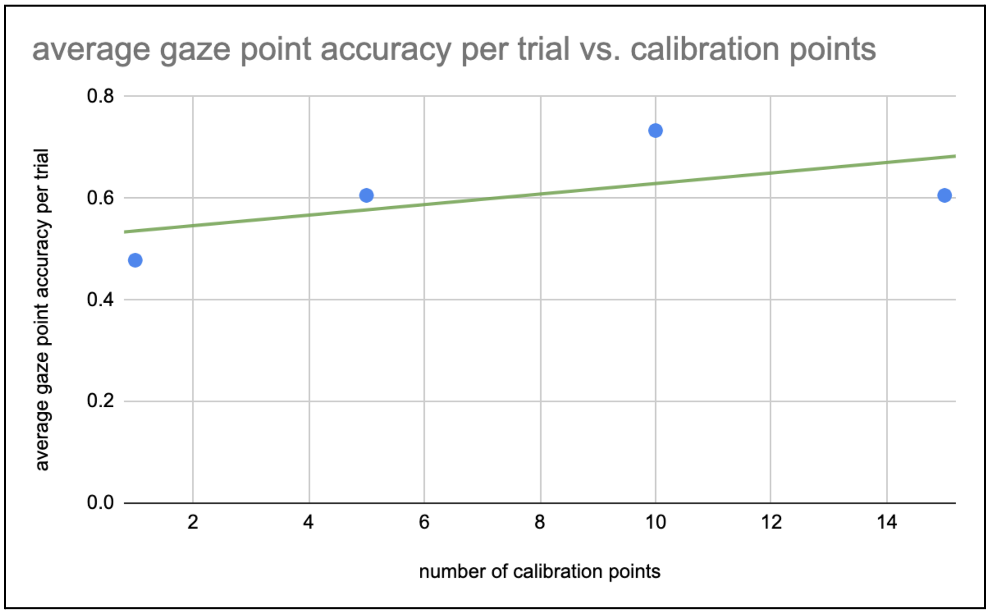


Figure 2: Best gaze point estimation accuracy consistently recorded at about 10 calibration points per trial.

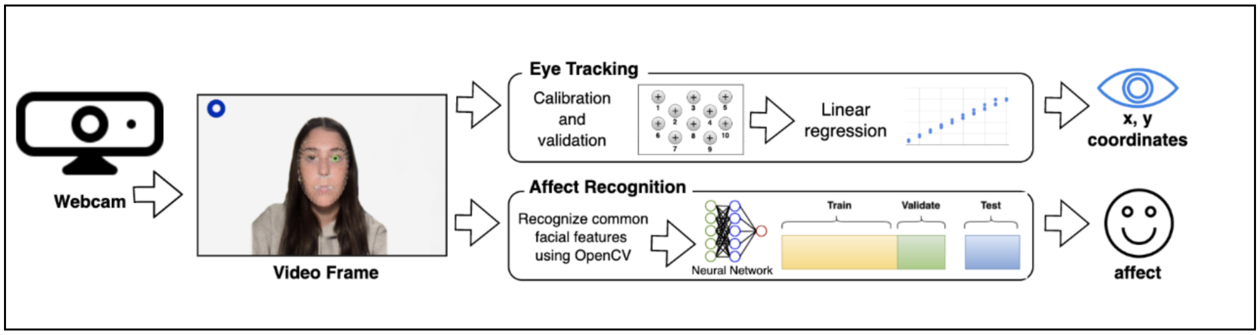


Figure 3: Gaze estimation based on linear regression of recorded pupil position and emotion recognition based on facial feature data

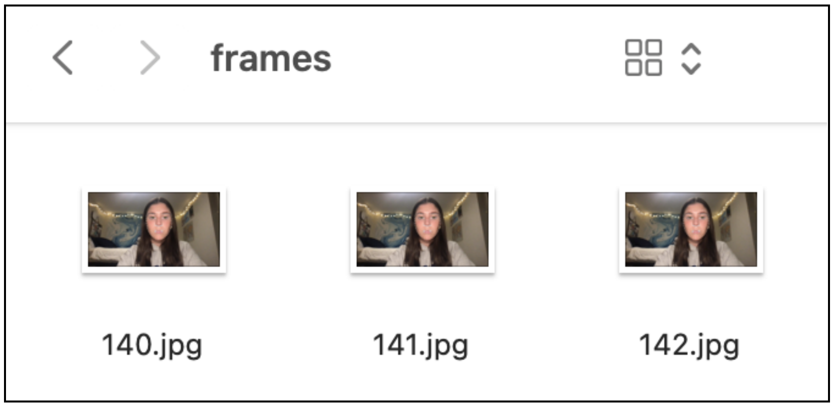
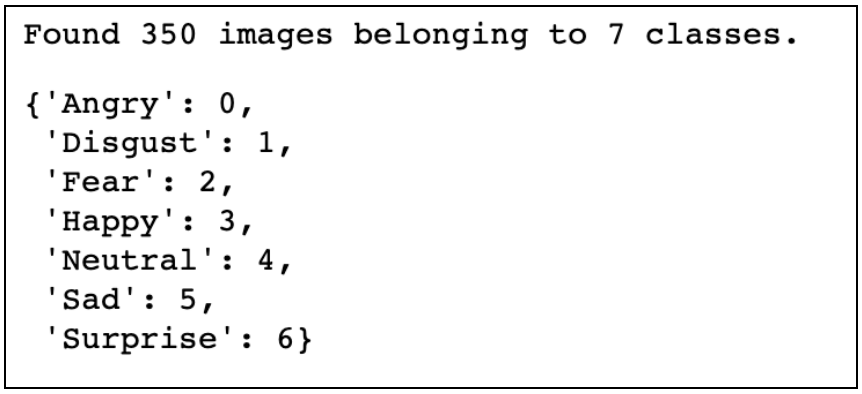


Figure 4: Frames taken per second and sent to the dual-purpose model



```
Found 350 images belonging to 7 classes.

{'Angry': 0,
 'Disgust': 1,
 'Fear': 2,
 'Happy': 3,
 'Neutral': 4,
 'Sad': 5,
 'Surprise': 6}
```

Figure 5: MobileNet training output, trained on 273 images of 7 different common emotions

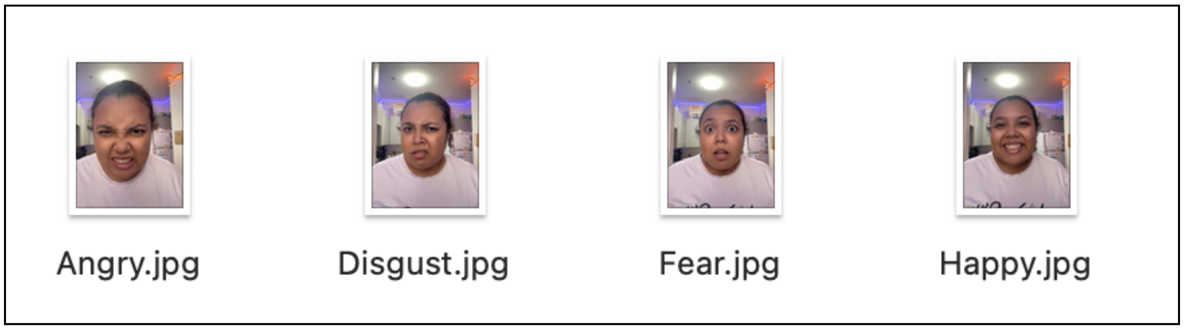
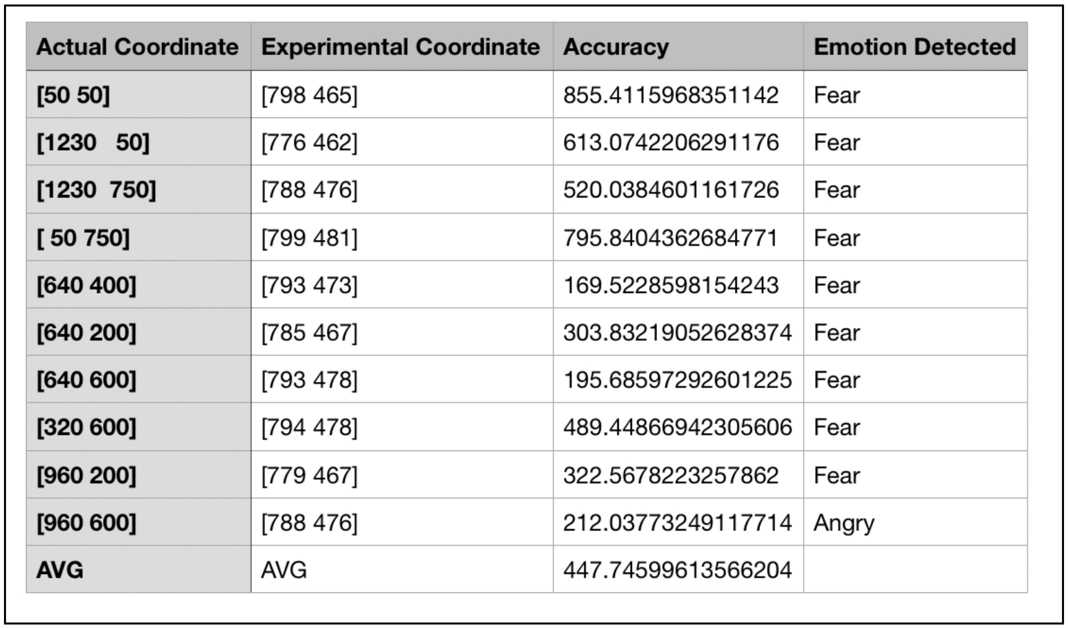


Figure 6: Images used to test emotion-recognition neural network



Actual Coordinate	Experimental Coordinate	Accuracy	Emotion Detected
[50 50]	[798 465]	855.4115968351142	Fear
[1230 50]	[776 462]	613.0742206291176	Fear
[1230 750]	[788 476]	520.0384601161726	Fear
[50 750]	[799 481]	795.8404362684771	Fear
[640 400]	[793 473]	169.5228598154243	Fear
[640 200]	[785 467]	303.83219052628374	Fear
[640 600]	[793 478]	195.68597292601225	Fear
[320 600]	[794 478]	489.44866942305606	Fear
[960 200]	[779 467]	322.5678223257862	Fear
[960 600]	[788 476]	212.03773249117714	Angry
AVG	AVG	447.74599613566204	

Figure 7: Program output, including actual calibration coordinates, recorded experimental coordinates, euclidean distance between the two, and detected emotion

USE CASES

The proposed work has implications to researchers, educators, and practitioners. Researchers can gain a better understanding of the cognitive processes involved in software development by observing the emotional response of developers when interacting with specific source code elements. In education, an eye movement-affect enabled Integrated Development Environment (Eye-DE) could respond automatically to student frustration by highlighting potential bugs or showing hints or documentation. In the industry, adding developer emotion to software artifacts might help in understanding the relationship between being tired/exhausted and code quality.

CONTRIBUTIONS & FUTURE WORK

In this paper, we have demonstrated the viability of a combined approach for eye tracking and affect recognition through an unmodified webcam. In addition, we discussed potential use cases for this tool in software development with future implications to researchers, educators, and developers.

We aim to increase the accuracy of the eye tracking module and use this tool in a controlled lab experiment in addition to collecting data from volunteer developers in the field. The goal of the lab experiment is to compare the accuracy of this tool to a research-grade eye tracker in software development tasks. The field experiment could help us understand the potential benefits of this tool in collecting eye tracking and affect data remotely (outside the lab) and at scale.

CONCLUSION

Our current results provide insights on the use of webcam eye tracking and its increased accessibility and usability when compared to high-scale models. Our future work includes plans to analyze the eye movement patterns of programmers as they are faced with a variety of affect-provoking stimuli during the software development process. Our hypothesis is that these programmers will experience a variety of emotional effects and feelings in reaction to the stimuli, which will be exhibited by their eye movement behavior. Eye tracking will then in turn provide valuable information about the emotional state of programmers, and can subsequently be used to improve the software development process.

REFERENCES

Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson. 2018. What happens when software developers are (un) happy. Journal of Systems and Software 140 (2018), 32–47.

Daniel Graziotin, et al. “Software Developers, Moods, Emotions, and Performance.” *IEEE Software*, vol. 31, no. 4, 2014, pp. 24–27, https://doi.org/10.1109/ms.2014.94.

Katarzyna Wisiecka, Krzysztof Krejtz, Izabela Krejtz, Damian Sromek, Adam Cellary, Beata Lewandowska, and Andrew Duchowski. 2022. Comparison of Webcam and Remote Eye Tracking. In 2022 Symposium on Eye Tracking Research and Applications. 1–7.

Mia Mohammad Imran, Yashasvi Jain, Preetha Chatterjee, and Kostadin Damevski. 2022. Data Augmentation for Improving Emotion Recognition in Software Engineering Communication. arXiv preprint arXiv:2208.05573 (2022).

Nicole Novielli, Daniela Girardi, and Filippo Lanubile. 2018. A benchmark study on sentiment analysis for software engineering research. In Proceedings of the 15th International Conference on Mining Software Repositories. 364–375.

Sebastian C Müller and Thomas Fritz. 2015. Stuck and frustrated or in flow and happy: Sensing developers’ emotions and progress. In 2015 IEEE/ACM 37th IEEE International

Conference on Software Engineering, Vol. 1. IEEE, 688–699.

Toufique Ahmed, Amiangshu Bosu, Anindya Iqbal, and Shahram Rahimi. 2017. SentiCR: a customized sentiment analysis tool for code review interactions. In 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 106–111.

Yahui Nan, Jianguo Ju, Qingyi Hua, Haoming Zhang, and Bo Wang. 2022. A-MobileNet: An approach of facial expression recognition. Alexandria Engineering Journal 61, 6 (2022), 4435–4444.