Using EEG Data to Detect Eye Movement

Aishwaroopa Narayanan

1. ABSTRACT

In this paper, we show that it is possible to use EEG data to detect eye movement using machine learning. By recognizing eye movement through EEG results, our goal is to help individuals with disabilities better control object movement and perform daily activities independently. This is especially important as many disabled individuals rely on assistance from others for their daily needs, which can be burdensome for the person providing help. To achieve these objectives, we trained different machine learning models using a data set of eye-state classification from Kaggle. We analyzed the results to assess the accuracy of a KNN (K Nearest Neighbors) model. With the model achieving an accuracy of 95.23% in detecting eye movement in patients. These findings suggest that the model could be effectively utilized in the future, with further research to assist individuals with disabilities. Overall, our research suggests that it is possible to recognize eye movement through EEG results reliably. Further research in this area could lead to the development of more effective and personalized interventions for individuals with poor hand-eye coordination.

2. INTRODUCTION

Recognizing eye movement with electroencephalography (EEG) results can be useful to discover learning patterns while doing certain activities[10]. Hand-eye coordination is needed for daily life, and this research can help people with disabilities control things in their standard of living. An electroencephalogram is a test that measures and records the electrical activity of the brain using electrodes placed on the scalp[5].

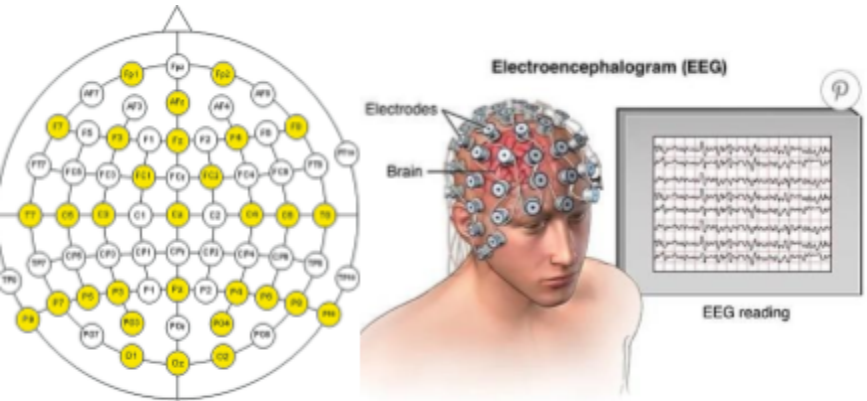

Figure 1 shows the different types of electrodes and their positions, universally[9], and how an EEG works[5].

We used EEG data collected from test subjects, and the data was labeled as either indicating eye movement or no eye movement[4]. We trained a machine learning model on this labeled data to predict whether or not a person was moving their eyes during the tasks. This machine learning model is a classification model that makes it easier to detect and classify eye movements in test subjects. Such a machine learning model has the potential to improve the independence of people with disabilities by allowing them to control external devices or perform actions through the movement of their eyes, potentially in combination with a brain-computer interface (BCI). In the future, a system like that could potentially allow individuals to carry out tasks and control their environment without the need for physical assistance from others[2]. Machine learning provides us with a reliable solution since it allows us to efficiently analyze and classify this data. We changed the hyperparameters and preprocessed the model to confirm that it was still accurate.

3. BACKGROUND

A blog post called, *How to Detect Eye Movement using Neuroscience and Machine Learning - Experimen*t by Mikaela Pisani, discusses a machine learning experiment to detect eye movement using a low-cost EEG device with few electrodes. The device used in the study is the Heart and Brain SpikerBox from Backyard Brains, and it measures the electrooculographic signal (EOG) produced by eye movements and records the data using electrodes placed on either side of the eye and behind the ear. The recorded data is then used to train a machine learning model to classify between left and right eye movements. It is mentioned in the article that previous studies have used more sophisticated devices with more electrodes to accurately detect eye movements and classify them with machine learning models. The study recorded 57 recordings of about three seconds on one subject and used 80% of the data for training the model and the remaining 20% for testing. Overall, this study suggests that it is feasible to use a low-cost device with few electrodes to detect and classify eye movement using machine learning [7].

Another article called “Combining EEG and Eye Tracking: Identification, Characterization, and Correction of Eye Movement Artifacts in Electroencephalographic Data” discusses the various sources of eye movement artifacts in EEG recording and the difficulties in correcting them[8]. It talks about previous research on the different types of eye movement artifacts, like chorioretinal dipole changes and eyelid artifacts, and their relation to different types of eye and eyelid movements. The authors also compare the performance of two correction methods, linear regression and independent component analysis (ICA), and propose an algorithm using ICA and eye tracker information to automatically identify and correct eye movement artifact components in EEG data. The authors show the effectiveness of the tiered approach using simultaneous EEG and eye movement recordings from a guided eye movement paradigm. Overall, the article discusses the challenges of eye movement artifacts in EEG recording and presents various approaches for addressing these artifacts.

4. Dataset

The data used in this paper was sourced from the online data platform Kaggle[4], and it is an eye-state classification data set. It has 15 attributes in total and separate train and test data. We only used the train data since that is the one with the eye detection labeled. Eye state was obtained by a video recording that was done while collecting the EEG data, which used an EEG device with 14 electrodes,[4]. We do not know what type of activities the participants were doing when the data was taken since it was not specified. The data is numerical data, and the number of samples is 10,486. There are fifteen features in the data. We split the data by 80 % for training and 20% for testing datasets. The dataset features are eye detection and the different electrodesEye detection is also measured through electrodes[4].

5. Methodology/Models

We used logistic regression, random forest regressor, and K-Nearest neighbors classifier (KNN), three popular machine learning algorithms for classification, and compared their results. Logistic regression describes the relationship between a dependent binary variable and one or more independent variables.Random forest regressor uses multiple decision trees to make predictions and improve accuracy, while also helping to prevent overfitting[3]. Lastly, KNN can be used to classify data or make predictions based on the values of nearby points[6]. We performed our model learning procedure by splitting the dataset into train and test data. Afterward, we implemented the different models from sklearn[1]. Finally, we evaluated the performance of our models on the test portion of the data using various metrics such as accuracy, recall, ROC curve, f1, and precision. The results are shown in figure 4.

6. Results and Discussion

As described in section 5, we used several types of algorithms (Figure 4) to find the best one and found out that KNeighborsClassifier (KNN) worked out the best. The outcome was close to the 93rd percentile. The accuracy score was 95.23 %, precision was 94.87 %, and recall was 94.25%. A confusion matrix summarizes the performance of a classification model. This model had the most true positive and true negative (See Figures 5-7).

Performance Of the Different Algorithms

Algorithms	Accuracy(%)	Recall(%)	Precision(%)
LogisticRegression	63.73	49.67	60.66
RandomForestRegressor	91.61	87.74	92.78
KNeighborsClassifier	95.23	94.25	94.87

Figure 4 shows the different model we tried and the performance of them

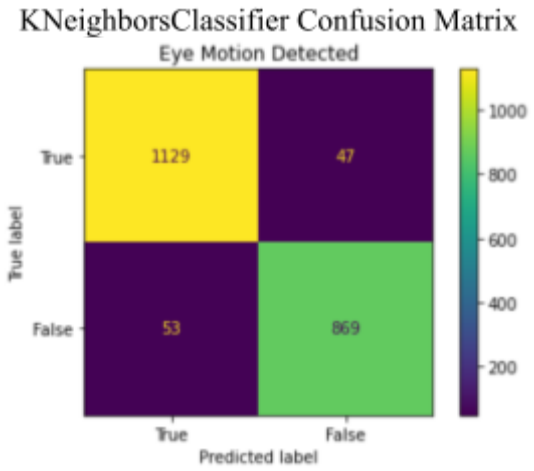

Figure 5 shows that the model predicted 100 false predictions

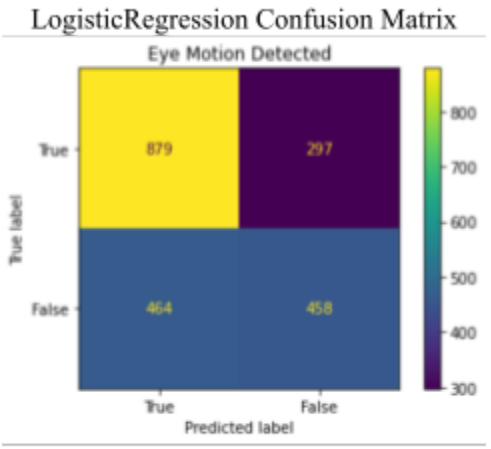

Figure 6 shows that the model predicted 761 false predictions

Figure 7 shows that the model predicted 176 false predictions

No matter how much we changed the hyperparameter K, the result was always above 94%. The result was constantly above 95% percent when we changed the algorithm of KNN. Thus, we concluded hyperparameter does not affect our model at all (Figure 8). Our method performed well with the raw data as it did after the data cleaning procedure. Not having enough information about the data, and what the participants were doing when the data was collected, could cause errors since we do not know if the participants were performing strenuous activities or activities that require minimal movement.

Hyperparameter Tuning Results

Hyperparameter	Value 1	Result 1 (%)	Value 2	Result 2 (%)
K	5	95	7	96
Algorithm	'ball_tree'	95.09	'kd_tree'	95.14

Figure 8 shows the results of the hyperparameter tuning, there is not much difference

7. Conclusion

In this research paper, the goal was to address the issue of recognizing eye movement with EEG results with the underlying question of how poor hand-eye coordination in individuals with disabilities can impact their ability to perform daily tasks. We used a dataset of eye movement detection from Kaggle and employed machine learning algorithms, including logistic regression, random forest regressor, linear regression, and K-Nearest neighbors classifier (KNN) to analyze the data. We found that the KNN model was the most effective, achieving an accuracy of 95.23% in classifying eye movement in participants. Further research in this area could lead to the development of more personalized interventions for individuals with poor hand-eye coordination using brain-computer interface (BCI), which will allow the individual to perform their desired action using brain activity. It could also lead to the development of assisting individuals with disabilities in controlling their daily activities and improving their quality of life with further research.

8. Acknowledgments

I would like to thank my mentor Tomer Arnon for his encouragement, support, and guidance on this research project. I would also like to thank Inspirit AI for providing me with this research paper opportunity and a great mentor.

9. References

[1] "1. Supervised Learning." *Scikit*, https://scikit-learn.org/stable/supervised_learning.html.

[2] Abiyev, Rahib H., et al. "Brain-Computer Interface for Control of Wheelchair Using Fuzzy Neural Networks." *BioMed Research International*, Hindawi, 29 Sept. 2016, https://www.hindawi.com/journals/bmri/2016/9359868/.

[3] Donges, Niklas. "Random Forest Classifier: A Complete Guide to How It Works in Machine Learning." *Built In*, 28 Sept. 2022, https://builtin.com/data-science/random-forest-algorithm.

[4] Dutta, Gaurav. "Neuroheadstate Eye-State Classification." *Kaggle*, 6 July 2022, https://www.kaggle.com/datasets/gauravduttakiit/neuroheadstate-eyestate-classification.

[5] "EEG and Brainwaves." *BRIGHT BRAIN CENTRE - LONDON'S EEG, NEUROFEEDBACK AND BRAIN STIMULATION CENTRE*, 23 May 2021, https://www.brightbraincentre.co.uk/electroencephalogram-eeg-brainwaves/.

[6] Joby, Amal. "What Is K-Nearest Neighbor? an ML Algorithm to Classify Data." *Learn Hub*, 19 July 2021, https://learn.g2.com/k-nearest-neighbor.

[7] Pisani, Mikaela. "How to Detect Eye Movement Using Neuroscience and Machine Learning - Experiment." *Rootstrap*, 2 Sept. 2022, https://www.rootstrap.com/blog/how-to-detect-eye-movement-using-neuroscience-and-machine-learning-experiment/.

[8] Plöchl, Michael, et al. "Combining EEG and Eye Tracking: Identification, Characterization, and Correction of Eye Movement Artifacts in Electroencephalographic Data." *Frontiers*, Frontiers, 9 Sept. 2012, https://www.frontiersin.org/articles/10.3389/fnhum.2012.00278/full.

[9] Sanz-Aznar, Javier, et al. "Neural Responses to Shot Changes by Cut in Cinematographic Editing: An EEG (ERD/ERS) Study." *PLOS ONE*, Public Library of Science, 14 Oct. 2021, https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0258485.

[10] Zhang, Bingxue, et al. "Design and Implementation of an EEG-Based Learning-Style Recognition Mechanism." *Brain Sciences*, U.S. National Library of Medicine, 11 May 2021, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8150355/#:~:text=In%20addition%2C%20we%20verified%20the,for%20effective%20learning%2Dstyle%20recognition.