Eye Fixation Pattern Classification using Neural Networks with Bimodal Distribution Removal and Evolutionary Algorithms

Ranjth Raj Kandasamy Senthil Kumar
u7004474@anu.edu.au

Research School of Computer Science, The Australian National University.

Abstract. Visual Perceptual skills are an individual's ability to interpret and understand the information presented through the visual medium. Facial perception and text reading are the two important perceptual skills that we use in day to day life. Face and text have different eye gaze patterns. Eye gazing is used by Psychologists particularly to gain insights and understand complicated cognitive processes. We introduce a Neural Network with the bimodal distribution removal method optimized using Evolutionary Algorithms to classify the eye gaze patterns while observing a face from a text document. This method proves to be more generalized and computationally less expensive for training than a simple Neural Network implementation and Neural Network with Bimodal Distribution Removal. Evolutionary algorithms help the model explore and exploit the solution space, preventing it from getting stuck in local minima. Comparison between the classification errors obtained by our model and the model discussed in the dataset paper shows our model's better performance and generalization.

Keywords: Eye gaze patterns, Eye fixation points, Bimodal distribution removal, Evolutionary algorithms.

1 Introduction

Visual data helps brain to process huge amount of data quickly and efficiently [15]. Understanding how an individual perceives a particular content can be derived from the eye gaze pattern of the individual. Cognition and fixational eye patterns are tightly coupled [4, 8, 9]. Many data points can be obtained while tracking an eye gaze movement. People tend to pause while observing something interesting or informative [12]. These points are recorded and filtered to first five as early stages of observations are plausible compared to rest [14]. These points are referred to as fixations, and hence the same terminology is used throughout this paper. Individuals might have different fixational pattern, but utilizing only the common ones help a machine learning model to localize the target where an individual looks and to predict the order in which they gaze through fixations [2, 5, 7]. Different objects have different eye gaze patterns while observing them. While observing a natural scene, the two important things that captures observer's attention are the face and the text. Face is considered visually captivating proven by evidence from infants [1]. And the man made things like text or cell phone attracts human's gaze [3].

Outliers impact the convergence of the model and results in generalization [11]. We used a modified Bimodal distribution removal (BDR) for detection and removal of outliers. As our dataset is imbalanced, we had to make modifications to BDR to prevent it from entirely removing the data belonging to class with less data. Thus, generalization is maintained.

Even though locally optimal solutions are guaranteed by back propagation, often they don't tend to be satisfactory. In these cases, the model is initiated with different weights and re trained using them, or extended with additional nodes. Extension results in increase of the freedom for the network, and there by the network correctly classifies all the training data. But this resulted model is overfit. Evolutionary algorithms(EA) offers a parallel search, which helps the model to overcome the above issues [16].

The main motivation behind this paper is to classify the fixation patterns between face perception and text reading. The objectives include finding the model which demonstrates the best performance and is computationally less expensive. We discuss three different classification approaches, and compared their performance on few metrices. In the first approach, we straight away train a simple Neural Network using Input data. In the second approach, we extend

the first with BDR techniques to remove outliers while training. And in the final approach, we replaced back propagation used in training weights with EA and used BDR for outlier removal.

2 Method

2.1 Problem statement

Given a fixation pattern which represents an eye gaze movement, we are trying to build a model that differentiates the eye gaze movement for a face perception from a text reading eye gaze movement. We will be describing the dataset used in the experiments, explaining the algorithms and methods used in the three approaches and briefing the metrics used for performance measures with formulas in this section.

2.2 Dataset

The dataset which we are using was collected from an experiment. A total of 10 members volunteered for this experiment. During the experiment, visual data was displayed on a monitor and the participant's eye gaze movements were collected using a Seeing machines eye-tracking system with FaceLAB software (Version 4.5, 2007). The experiment was split into three phases. In first phase, a set of 20 pictures containing faces were displayed. In second phase, 5 text documents were displayed. In the final phase, another set of 20 pictures containing faces were displayed. The sets of images considered at first and final phase don't overlap. There wasn't any break during this experiment.

According to [13], there are supposed to be 450 eye fixation patterns. But there are only 323 patterns in the dataset provided, i.e., 169 for training and 154 for testing. So we assume that the rest were removed from the dataset either because they were overlapping with other patterns or they were noisy. As mentioned earlier in Introduction, we consider five fixation points. Each row in the dataset contains five columns, and the first four contains the absolute difference of vertical axis values between two consecutive fixation points. The final column contains the class label. ***Fig. 1*** visualizes how imbalanced the dataset is.

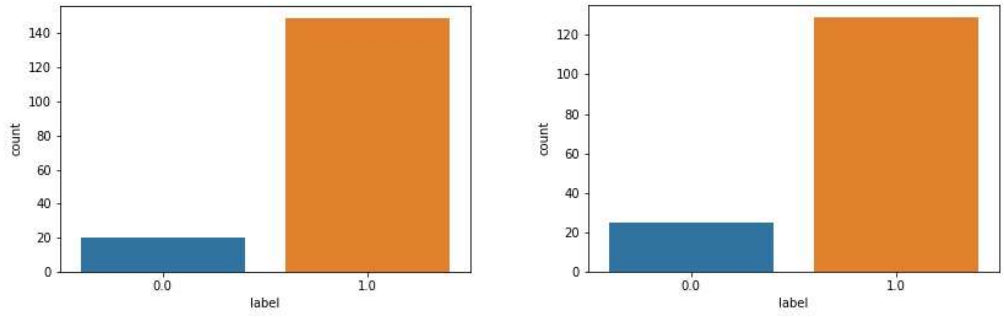

Fig. 1*:* From the Training data distribution(left plot) and Testing data distribution(right plot), we can observe how imbalanced the dataset is. Label 0.0 belongs to text reading fixation pattern and 1.0 belongs to face perception fixation pattern.

2.3 Simple Neural Network

A Neural Network with many hidden units is considered complex. Training such complex networks results in overfitting. We considered these two simple rule of thumbs before starting the experiment – Number of hidden units should be between the number of the number of input units and output units [19], and Number of hidden units shouldn't be more than twice the number of input units [18]. In order to find the right number of hidden units and hidden layer,

we further divided the training data into training and validation data. 10% of training data was considered for validation. We experimented with a single hidden layer containing hidden units of size 5, 4 ,3 and 2. As expected, layer with hidden units 4 and 5 achieves more accuracy on the training data and more validation loss compared to the layer with 3 units. These claims can be inferred from ***Table 1***.

Table 1. Experimental results obtained with varying hyper parameters

# of layers	Single			Double	
# of hidden units	3	4	5	3, 2	4,3
Training Accuracy (%)	99.1	100	100	99.2	99.3
Validation Accuracy(%)	95.0	88.4	88.5	96.4	96.1
Validation Loss	0.156	0.245	0.278	0.146	0.192

We extended our experiment to multiple layers as a second phase. We compared the results between two architectures – layer 1 with 3 and layer 2 with 2, and layer 1 with 4 and layer 2 with 3. After 2000 epochs, both produced same results. First architecture achieves 99% accuracy in training, 96% in validation data and validation loss of 0.15 at 750th iteration. This architecture produced better results in less amount of time. So, this is the optimal architecture we will be using in further experiments and discussions.

As we are trying do classification, the number of hidden units gradually decrease as we move through the sequential network. The hidden layers have relu as their activation function, while the output layer has sigmoid activation function. Loss is calculated using binary cross entropy. This loss function outputs a score that summarizes the average difference between the ground truth and predicted probability distributions for predicting class 1. The score is minimized using Adam's optimization approach and experiments were conducted modifying the learning rate. The imbalanced training data pushes the weights towards local minimum most of the times.

2.4 NN with Bimodal Distribution Removal

Bimodal Distribution Removal (BDR) [11] is one of outlier detection and removal techniques. This technique depends on the frequency distribution of the error measured for all training patterns in a single epoch. The error is calculated using Binary Cross Entropy. In the initial stages of training, the error seems to be widespread. After certain iterations, error of many patterns reduced drastically, and there might be few patters with relatively high error. Now we try to separate the training patterns into three groups based on their error. The first group is low error peak, which contains patterns that the network has well trained on, the second group contains patterns which are being learnt by the network and are called slow coaches ,and the final group is high error peak, which contains outliers. From the two peaks in the error distribution it is clear that the network can identify outliers itself. If we consider a network learning labels given patterns, then outliers are patterns that doesn't produce labels which the network expects.

Variance is considered as one of the important statistical measures in the approach, as it helps in identifying the bimodal error distribution – which is difficult and time consuming during training phase. Outlier removal resulted in removal of many text reading fixation patterns. Because the network tends to produce high error values for text reading patterns as it doesn't have enough patterns belonging to that class. So we use this method in our application to remove the outliers present in face perception fixation patterns. Imbalanced dataset tends to overfit the model, we are trying to reduce this effect by removing the outlier patterns belonging to face perception only. By this, we are not losing any data belonging to text reading. Because of this minor modification to the method, outliers exist even after applying this method and we aren't sure when to end the training.

2.4.1 Algorithm

1. Consider the whole training data and begin training.
2. Repeat step1, until variance of errors ν_{ts} reaches below 0.1.
3. Calculate the mean error for the entire training data, $\overline{\delta_{ts}}$.
4. Create a subset with the data having error greater than $\overline{\delta_{ts}}$.
5. Calculate the mean $\overline{\delta_{ss}}$ and standard deviation σ_{ss} of this subset.
6. Remove the patterns belonging to class 1 from the training data, which satisfy the below equation.

$$error \geq \overline{\delta_{ss}} + \alpha\sigma_{ss}, \qquad where\ 0 \leq \alpha \leq 1$$

7. Repeat steps 2-6 every 50 epochs until $\nu_{ts} > 0.01$.
8. End Training.

2.5 Evolutionary Algorithm

The structure of the neural network has 4 input neurons, 3 neurons in first hidden layer, 2 neurons in second hidden layer and 1 output neuron. So a total of 20 weights are needed to be trained. All weight matrices are un-squeezed to form a weight vector of shape (20×1). This weight vector is considered as a chromosome, while each individual weight in the vector is considered as a gene. We will be further presenting the optimal set of parameters and how we concluded. Each result published in this section is an average of 100 runs on that particular set of parameters for the sake of randomness. The accuracy is considered as fitness measure.

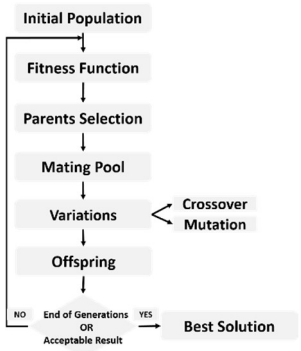

Fig. 2 Flow chart explaining the steps followed in a generic evolutionary algorithm.

EA have six key steps to be implemented which are visualized as a flowchart in **Fig. 2**. The first step is initialization of the population. We populate weight vectors with gene values ranging between -0.1 to 0.1. Population size is fixed to 16, after experimenting with different population sizes. Keeping the population size as small as possible helps us computationally and at the same time maintains best accuracy. The reason for choosing 16 as population size can be inferred from the graph displayed in ***Fig. 3***. In the second step of EA, we find the fitness value for each weight matrix. Then in next step, we select parents based on 4 way tournament selection method. EA's performance is highly influenced by the parent selection component. Tournament selection approach converges in a short amount of time with good success rate [17]. The number of people mating set to constant value during the above experiment is tuned next to find the optimal number of chromosomes for mating. From the ***Fig. 3***, we can clearly see that model with mating pool of size 10 achieves better performance in a short amount of time. The mating pool now produces off springs using two point cross-over approach. All crossover approaches produces significant results [17]. Next step is to perform mutation on the cross over off springs. Mutation rate also has a strong effect on EA's performance. So, we used the optimal parameters obtained from above experiments and varied mutation rate. The ***Table 2*** tabulates results from both test and training data. The optimal mutation rate is 0.1 as we give importance to recall and f1-reason for the reason of imbalanced dataset.

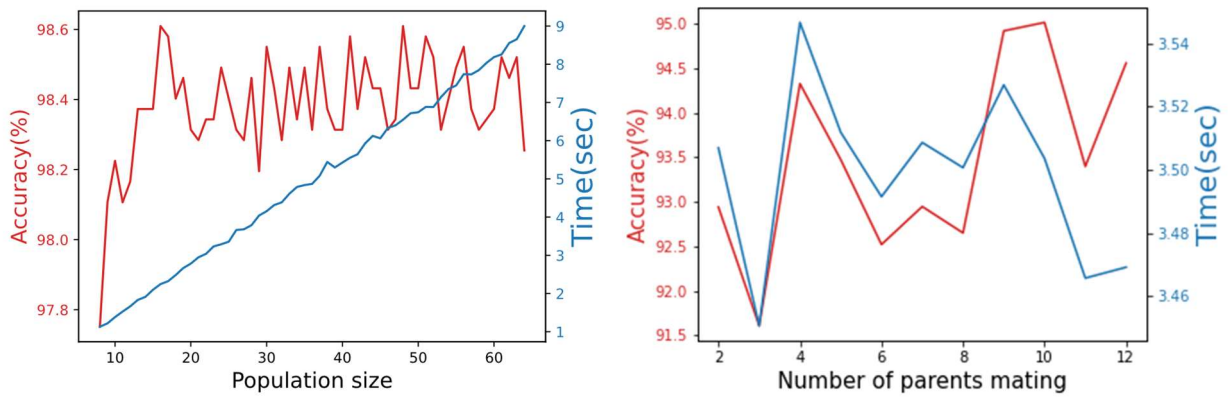

Fig. 3. Plot b/w population, accuracy and time (Left) helps us visualize the changes wrt increase in population size. Plot between mating pool, accuracy and time(Right) helps us visualize the changes wrt increase in size of mating pool. The red splines represent the accuracy in percentage while the blue time in secs.

After mutation, we include all the parents and fill off springs in the vacant space for the next generation. Using the above optimal values obtained at each and every experiment above, our EA implementation achieves its best performance within 50 generations.

Table 2. Experimental results obtained with different mutation rates on training and testing dataset.

	Training data			Testing data
Mutation rate	Precision	Recall	F1-score	Accuracy(%)
0.05	0.91	0.99	0.95	97.43
0.1	0.96	0.98	0.97	98.816
0.2	0.85	0.99	0.92	89.94

2.6 Metrics

Below we will be defining the metrices used in this paper during experiments. These metrices will be based on the terms – True Positive(TP), False Positive(FP), True Negative(TN) and False Negative(FN), from confusion matrix.

2.6.1 Accuracy

Accuracy has been used in this paper while tuning parameters and hyper parameters. Accuracy is given by the formula.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2.6.2 Precision and recall

In pattern recognition and classification, precision(P) explains what fraction of positive predictions by the model are actually positive, while recall(R) explains what fraction of actual positives are correctly classified as positive by the model. These two are more useful when evaluating Binary Classifiers on Imbalanced Datasets [10]. Precision and recall are given by

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

$$Recall = \frac{TP}{TP + FN}$$

2.6.3 F1 score

The F1 score gives a weighted measure of Precision and Recall, defined as a harmonic mean between them. F1-score [6] can be written as

$$F1\ score = \frac{2 * P * R}{P + R}$$

3 Results

Out of the four approaches, the first one is the simplest. We trained and tested the architecture discussed in Simple Neural Network subsection on the imbalanced dataset. This trained model achieved 0.96 precision, 0.98 recall and 0.97 f1-score over the test data. We will be using this as our base result and comparing it with the rest of approaches.

In our second approach, we use the same NN architecture attached with BDR technique. We straight away implemented the approach as discussed in [11] without any major modifications to the configuration. The only parameter that needs tuning is α. Even after setting α to a very low value when considering data from both the classes for removal, this outlier removal technique removes most of the data from the class with less data and training accuracy reaches 100% in 200 iterations. So we had to modify this approach, such that it removes data only from the dataset with more data values. ***Table 3*** tabulates the performance of the second approach on testing data with different sensitivity factor values ranging from 0.1 to 1. The modified approach with 0.5 as sensitivity value achieved 0.98 recall, 0.97 f1-score and a loss of 0.1 over test data. So, this value is considered optimal. Using the metrics and above observations, we can clearly infer that approach 2 performed better than 1 computationally.

Table 3. Experimental results for varying sensitivity values(α).

Sensitivity value	Precision	Recall	F1 score
0.1	0.96	0.96	0.96
0.25	0.96	0.97	0.96
0.5	0.97	0.98	0.97
1	0.98	0.97	0.96

For the final approach, we combined NN, BDR and EA. The one major difference compared to the rest of the two approaches is that this method doesn't use back propagation for weight calculation, rather calculates weight using EA. The other major thing to be noted is that this approach straight away uses BDR for outlier removal, and not the modified one used in second approach. Because of this, we could halt the training before the model overfits. This approach achieves an accuracy of 99% on training data in less than 10 seconds. So, this clearly outperforms the rest two methods in computational time. Each generation of EA is computationally expensive than each epoch, but EA reaches global set of weights in less number of generations whereas back propagation needs more epochs. **Table 4** helps us understand the difference in the performance between three approaches for the classification task using three important metrices -precision, recall and f1-score.

Table 4. Comparing the performance of the three approaches using precision , recall and f1-score as metrices.

Approach	Precision	Recall	F1 score
Simple NN – Approach 1	0.96	0.98	0.97
NN with BDR	0.97	0.99	0.97
NN with BDR and EA	0.98	0.99	0.98

In the first and second plot of ***Fig. 4***, the accuracy and loss start changing after 250 iterations approx. The first approach converges after 1500 iterations, while second approach converges after 1000. In case of third plot, the

model converges after 45 generations. But if we are able to make both backprop and EA algorithms parallel, then the execution time difference becomes negligible between second and third approach [16]. The one with EA achieves a classification loss of 0.35 while the rest converge with a loss of above 5.

The metric used in the dataset paper [13] is Classification Error(CLE). The NN with DR and GA discussed in this paper has a CLE of 4% for test patterns considering 2 class CLE as the metric. Considering the resources available at the time period when the dataset was published, our approach works better because we have high computational power.

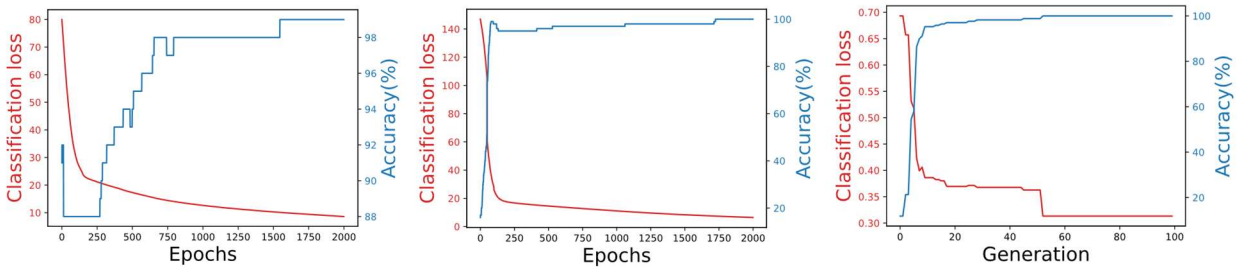

Fig. 4 Three plots visualize the performance of three different approaches on the training dataset. (Left) Represents the simple NN approach, (center) represents the NN with BDR and (right) represents the NN with BDR and GA. Blue spline indicates accuracy, and red line indicates classification loss.

4 Conclusion

Recall and f1-score are considered best metrices for the task of classifying an imbalanced data. So, we haven't used CLE elsewhere in this paper. BDR with minor modifications improved the performance, and it won't be the same when we have no data to remove. [11] method would have been much more efficient with balanced dataset.
Probabilistically selecting chromosomes to become a parent in the next generation can discover globally optimal solution. Whereas back propagation can lead to overfitting with addition of few extra nodes. In a resource constrained environment, our approach NN with BDR and EA can achieve better results than remaining two approaches Clearly from **Table 4**, approach 3 outperforms all approaches for our problem statement.

The Precision and recall with respect to face perception are close to equal in each approach, but they have a huge difference between approaches when calculated with respect to text reading. We aren't concentrating much on this, as out primary objective to correctly classify face perceptions.

5 Future work

The approaches discussed in this paper are executed serially without the help of GPU. So in future we will parallelizing algorithms and compare their performance. We haven't discussed data balancing techniques, as balancing by random up sampling and down sampling didn't perform better. But there are a few state of the art approaches, which could help us balance our imbalanced data. So we will try to implement them from scratch without using inbuilt libraries, so that we have many parameters to play with for the sake of increase in performance. Eye moves from one fixation point to another in a certain time. This creates a pattern, in future we will be analyzing on ways to predict this pattern using RNN.

References

1. Cashon, C. Cohen, L. Pascalis, O. Slater, A. (2003). The construction, deconstruction, and reconstruction of infant face perception. The development of face processing in infancy and early childhood: Current perspectives. (pp. 55–68). New York: NOVA Science Publishers.
2. Cerf, M. Cleary, D. Peters, R. Einhäuser, W. Koch, C. (2007). Observers are consistent when rating image conspicuity. Vision Research, 47, 3052–3060.
3. Cerf, M. Frady, E. Koch, C. (2008). Using semantic content as cues for better scanpath prediction. Proceedings of the 2008 symposium on Eye tracking research & applications (pp. 143–146). New York, NY, USA: ACM.
4. Einhäuser, W. Kruse, W. Hoffmann, K. König, P. (2006). Differences of monkey and human overt attention under natural conditions. Vision Research, 46, 1194–1209.
5. Foulsham, T. Underwood, G. (2008). What can saliency models predict about eye movements Spatial and sequential aspects of fixations during encoding and recognition. Journal of Vision, 8, (2):6, 1–17, http://journalofvision.org/8/2/6/, doi:10.1167/8.2.6.
6. Nancy, C. 1(992). MUC-4 evaluation metrics. In Proceedings of the 4th conference on Message understanding (pp. 22–29).USA: ACL.
7. Oliva, A. Torralba, A. Castelhano, M. Henderson, J. (2003). Top–down control of visual attention in object detection. Proceedings of the 2003 International Conference on Image Processing (p. 1).
8. Parkhurst, D. Law, K. Niebur, E. (2002). Modeling the role of salience in the allocation of overt visual attention. Vision Research, 42, 107–123.
9. Rizzolatti, G. Riggio, L. Dascola, I. Umilta, C. (1987). Reorienting attention across the horizontal and vertical meridians: Evidence in favor of a premotor theory of attention. Neuropsychologia, 25, 31–40.
10. Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PloS one, 10(3).
11. Slade, P., & Gedeon, T. D. (1993, June). Bimodal distribution removal. In International Workshop on Artificial Neural Networks (pp. 249-254). Springer, Berlin, Heidelberg.
12. Yarbus, A. (1967). Eye Movements and Vision. Plenum Press, New York.
13. Zhu, D., Mendis, B. S. U., Gedeon, T., Asthana, A., & Goecke, R. (2008, November). A hybrid fuzzy approach for human eye gaze pattern recognition. In International Conference on Neural Information Processing (pp. 655-662). Springer, Berlin, Heidelberg.
14. Althoff, R.R., Cohen, N.J. (1999). Eye-movement Based Memory Effect: A Reprocessing Effect in Face Perception. Journal of Experimental Psychology: Learning, Memory, and Cognition 25, 997–1010.
15. James, W. (1890). The principles of psychology. New York: Holt.
16. Fogel, D. B., Fogel, L. J., & Porto, V. W. (1990). Evolving neural networks. Biological cybernetics, 63(6), 487-493.
17. Nannen, V., Smit, S. K., & Eiben, A. E. (2008, September). Costs and benefits of tuning parameters of evolutionary algorithms. In International Conference on Parallel Problem Solving from Nature (pp. 528-538). Springer, Berlin, Heidelberg.
18. Berry, M.J.A., & Linoff, G. (1997). Data Mining Techniques, NY. John Wiley & Sons.
19. Blum, A. (1992). Neural Networks in C++, NY. Wiley.