Identifying Device Screen Size using User Eye Movements on Mobile Devices

Abhinav Pandey
University Id : u6724645
Australian National University

Abstract

With the advancement of mobile technology, comes an increase of using such devices for illegal activities. This calls for an advancement in the investigative capabilities of institutions responsible for governing private and public security. This study is a review of the performance of a simple feed-forward network of three layers of processing units[1] for classifying the screen size of a mobile device using a dataset containing eye movement and phone usage data gathered from 18 subjects using devices of varying sizes [2]. In this paper we demonstrate the disadvantages of this method and why it is not appropriate for the task.

Introduction

Neural networks have recently achieved state-of-the-art performance on several classification tasks [4]. Since, they have performed well on a broad variety of tasks and achieved such state-of-the-art results, it is natural to assume that they can be relied upon for other classification tasks. In this paper we will generally assume a feed-forward network of three layers of processing units. All connections are from units in one level to the subsequent and previous one. The hidden layer consists of fewer units than the input layer, thus compressing the data. [1]

The dataset records eye movements and web searching activity by different subjects on three different screen sizes. These are recorded as *S, M and L* corresponding to *Small, Medium and Large screen sizes.* Devices with the screen size (diagonal) less than 4 inches (e.g., Samsung Galaxy S1 and Apple iPhone 3 or 4) are recorded in the *S or small-sized* category. Devices with a screen of 4.5 inches (e.g., Apple iPhone 6) are classified as *M or medium-sized*. Finally, *phablets* (a portmanteau word combining the words phone and tablet) that have a screen size of over 5.4 inches (e.g., Samsung Galaxy Note 4 or Apple iPhone 6 Plus) are recorded as *L or large-sized*. [2]

Web searching is a common activity that phone users indulge in. Identifying a devices dimensional information can be used to isolate and identify which category of device owners does an anonymous individual with criminal intent belong to.

Data Pre-processing

The dataset[2] contains 33 features (excluding the Subject IDs and Screen Size) containing information about usage duration, website selection and the relative positions of these website links on the device screens. 32 of these were numerical features and 1 was of categorical type. *"Screen Size" was treated as the dependent/target feature for the classification task.*

The first step was to reduce the number of features by removing all but one variable from highly correlated pairs/groups of features. The thought behind this approach that was that pairs/groups of highly correlated independent variables carry redundant information and retaining just one of them would be a good way to remove features without much loss of information for the classification model.

Correlation between the independent variables were first calculated and inspected visually. However, the large number of features acted as an impediment to identifying meaningful relationships among these variables.

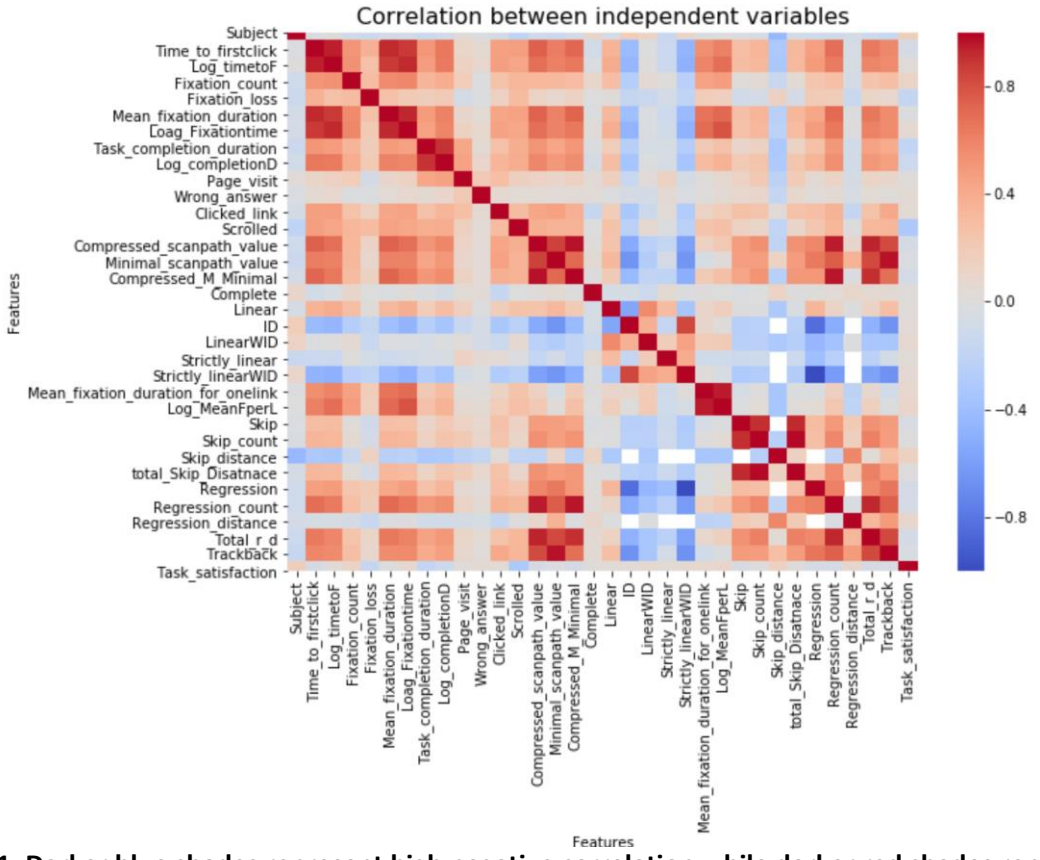


Fig. 1. Darker blue shades represent high negative correlation while darker red shades represent high positive correlation

For that reason, features with high (absolute) correlation (a Pearson correlation greater than 0.85 were marked as highly correlated) were isolated and observed.

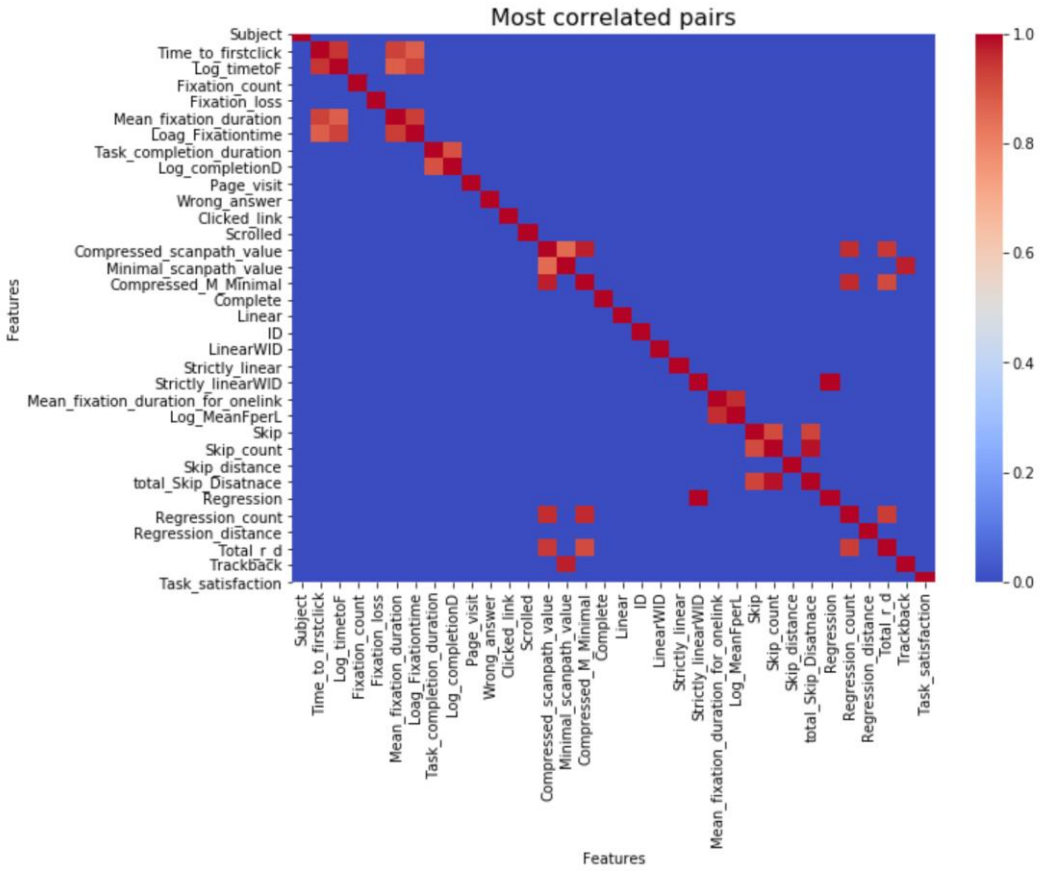


Fig. 2. Dark red shades represent high (positive or negative) correlation

After visually identifying these high correlation groups of features, we further analysed the exact correlation values in these groups. In the case where more than 2 variables (group) had a high correlation with one another, the feature with the highest sum of correlation coefficients with all the other members of the group was retained. In the case of highly correlated *pairs*, one was randomly chosen while the other one was removed.

Identification features (Subject) and features with more than 80% missing data were removed (Skip_distance).

Task Number, the only categorical feature other than the target, contained 9 distinct values. It was encoded by creating dummy features (containing 0 or 1 marking presence or absence of that variable) for 8 of these 9 values. The idea behind keeping 8 and removing 1 of these features was that when 8 features have a value of 0, we get the encoding for the Task Number 9. Thus, only 8 of these are needed to represent 9 features.

The target variable viz. *Screen Size* was encoded with numerical values i.e. S, M and L were replaced with 0, 1 and 2.

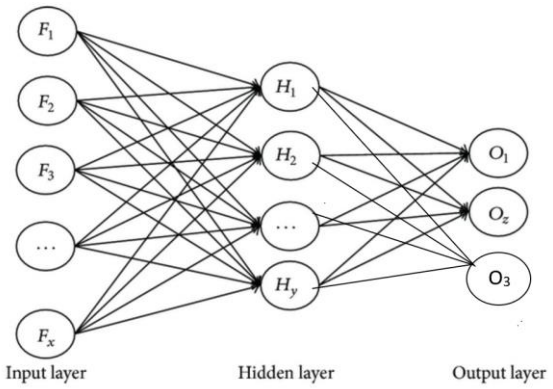
Since, we are using evolutionary algorithms in this paper, it is important to consider the computation cost and the time complexity of our implementation. Therefore, it was important for us to use inputs for our model that had smaller dimensionality, while preserving information in the dataset that is relevant to our classification task. For this purpose, we performed Principal Component Analysis on the dataset to *extract the dominant patterns in the matrix* (9).

This was followed by using forward feature selection on these Principal Components to further reduce our input size to reduce the running time of our Evolutionary Algorithm. *Forward selection is an iterative method in which we start with having no feature in the model. In each iteration, we keep adding the feature which best improves our model till an addition of a new variable does not improve the performance of the model*.(7) However, instead of using neural networks for this selection method we used a decision tree classifier. This was done because decision tree models require less training and testing time compared to neural networks and demonstrate strong classification abilities. It is important to note that this was done under the assumption that (with the *right* architecture and model parameters) a neural network can use the same input features and provide more accurate classification results. The Decision Tree model had the best classification performance (measured using the F1-score on test data) of 0.467 (on a scale of 0 to 1, with 0 being the worst and 1 being the best) using 4 out of the total 29 Principal Components. These were Principal Component number 9, 10, 26 and 27 which we used for our classification task using neural networks.

Modelling

The neural network architecture for our simple classification model can be visually represented as shown in Fig.3[5]. Three fully interconnected layers viz. an input layer with 30 neurons, a hidden layer with 20 neurons and the final layer with 3 output neurons. The *Cross-entropy loss* function was used for measuring loss during training, which is a widely used loss function for classification tasks.

Fig. 3. Our Simple Neural Network (*with 30 input, 20 hidden and 3 output neurons*) [5]



To be able to draw statistically convincing conclusions, it is important to estimate the uncertainty of such estimates.(6) For this reason, the model was evaluated using a 10-fold cross validation on our dataset. The average F1-score was used to evaluate the performance of the model, which in this case was found to be 0.454.

Since, our neural network model performed no better than the Decision Tree model (F1-score = 0.467) that we used earlier, we employed evolutionary algorithms to find the best parameters for our model. Since, they *can self-adapt and find optimal solutions on the fly(8*), while taking less time than a brute force search of the solution space. The parameters of interest were number of hidden layers, number of neurons per layer and the dropout rate for each hidden layer.

The evolutionary algorithm was built with the following specifications (the meaning of these parameters is explained in detail later) –

Initial Population = 100

Number of generations = 20

Fitness Function = average F1-score using 10-fold cross validation

Selection operator = Elitism (the 30 *fittest* networks were selected for producing the next generation)

Reproduction method = Single point crossover

Mutation probability = 100% i.e. each child network had to undergo a 10% alteration in one of the parameters in their *genetic structure.*

Here, is an explanation of these parameters. The maximum size of the population was capped at 100. Each of these networks was trained and evaluated using 10-fold cross validation and 20 of these networks with the highest average F1-scores were selected for producing the next generation of neural networks. The parents were selected in a hierarchical manner i.e. the fittest network mated with the 2nd fittest, the 2nd fittest mated with the 3rd fittest and so on. Single point crossover was the method of reproduction which means that one parameter was randomly selected from the 3 and those were interchanged, resulting in two *child* networks for each parent pair. Following this, the *children* underwent a 10% mutation of one of these parameters. The choice of the mutated parameter was random. To conserve the size of the population at 100, only 60 of these *children* networks survived (by random chance) and passed onto the next generation. 20 of the weakest members of the population also underwent a 10% mutation and survived onto the next generation. This was done to sufficiently increase the variation in the gene pool to ensure that the algorithm did not converge at a local maximum. After all the generations were completely created, we selected the best performing model from their history.

The algorithm took roughly 3 hours to run and arrived at a model with a 10-fold cross validation score of 0.489, which was found in the 16th generation of this evolutionary interpretation of parameter tuning.

Figure 4 can help us visualise the improvement in the performance with each generation with the number of better models (darker cells indicating higher F1-scores).

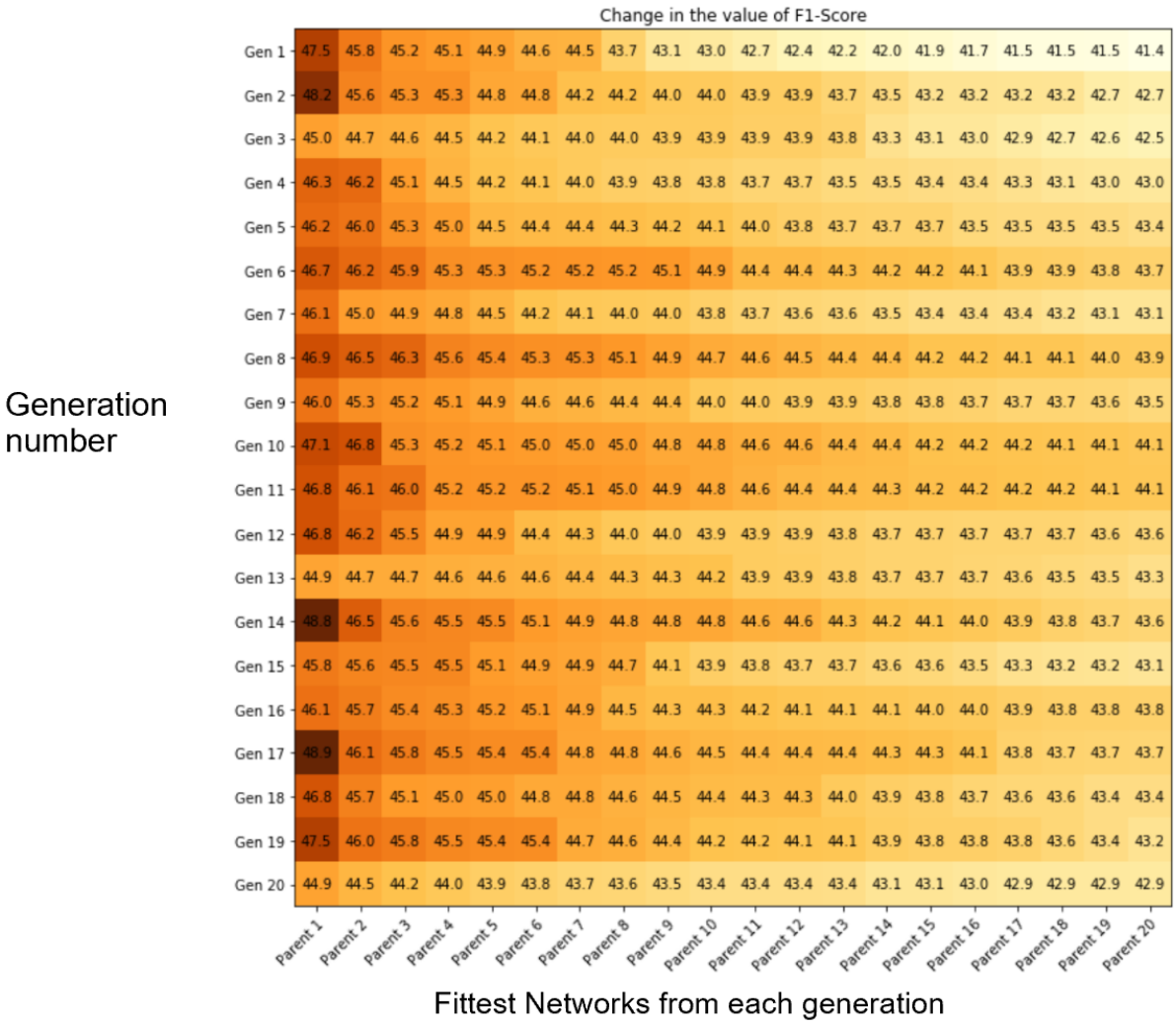


Fig. 4. Visualizing the overall improvement of model performances with each passing generation.

Results

The best performing model had the following parameters –
Number of hidden layers = 2
Number of neurons per layer = 93
Dropout rate = 0.357

It had an F1 score of 0.489, which backs our assumption that the neural network model with the *right* parameters would give us better performance than a decision tree classifier.

Conclusions

However, these results obtained by the model are subpar at best and are not ready for use in a real-world setup, yet. We believe that creating a new dataset with additional features could potentially be helpful for us in improving the classification performance of the models

Future Work

This classification solution could further be improved by a detailed analysis of all the features in the dataset and understanding their relationships to the screen sizes. Also, this paper could be furthered by feature augmentation or transformations, such as log transforms or normalising.
The evolutionary algorithm could be further used to include different activation functions and optimizers for tuning. Also, one could modify the algorithm to have different number of neurons and dropouts for each layer in the network (which is not the case in this review).

References

1. Gedeon, T.D. and Harris, D., 1992, June. Progressive image compression. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks* (Vol. 4, pp. 403-407). IEEE.
2. Kim, J., Thomas, P., Sankaranarayana, R., Gedeon, T. and Yoon, H.J., 2016. Understanding eye movements on mobile devices for better presentation of search results. *Journal of the Association for Information Science and Technology*, *67*(11), pp.2607-2619.
3. Gedeon, T.D., 1998, October. Stochastic bidirectional training. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)* (Vol. 2, pp. 1968-1971). IEEE.
4. Erhan, D., Szegedy, C., Toshev, A. and Anguelov, D., 2014. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2147-2154).
5. Saputri, Theresia & Khan, Adil & Lee, Seok-Won. (2014). User-Independent Activity Recognition via Three-Stage GA-Based Feature Selection. International Journal of Distributed Sensor Networks. 2014. 1-15. 10.1155/2014/706287.
6. Bengio, Y., & Grandvalet, Y. (2004). No unbiased estimator of the variance of k-fold cross-validation. *Journal of machine learning research*, *5*(Sep), 1089-1105.
7. I. and Kaushik, S., 2020. *Feature Selection Methods With Example (Variable Selection Methods)*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/> [Accessed 31 May 2020].
8. Fogel, D. B. (1997, January). The Advantages of Evolutionary Computation. In *BCEC* (pp. 1-
9. Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, *2*(1-3), 37-52.