GAZE BASED MIND WANDERING DETECTION USING DEEP LEARNING

A Thesis

by

SUBROTO SINGHA

Submitted to the Graduate School
of Texas A&M University-Commerce
in partial fulfillment of the requirements
for the degree of
MASTER OF SCIENCE
August 2021

GAZE BASED MIND WANDERING DETECTION

USING DEEP LEARNING

A Thesis

by

SUBROTO SINGHA

Approved by:

Advisor: Derek Harter

Committee: Abdullah N. Arslan
Yuehua Wang
Burchan Aydin

Head of Department: Andrea Graham

Dean of the College: Brent Donham

Dean of the Graduate School: Jennifer L. Schroeder

ABSTRACT

GAZE BASED MIND WANDERING DETECTION
USING DEEP LEARNING

Subroto Singha, MS
Texas A&M University-Commerce, 2021

Advisor: Derek Harter, PhD

Mind wandering (MW) is a phenomenon where a person shifts their attention from task-related to task-unrelated information. Mind Wandering is an omnipresent phenomenon for human beings. The consequence of mind wandering can impact a person's performance negatively. Reorienting the mind's attention using technology shows great promise to improve the performance and productivity of people in learning or other performative tasks. In this research, we investigate 62 eye gaze features by dividing them into four sets of global features: eye movement descriptive features, pupil diameter descriptive features, blink features, and miscellaneous features to detect mind wandering during reading from a computer interface. Our dataset, which was collected from a previous study, contains a mind wandering report where 135 participants were recorded "mind wandering" or "not mind wandering" using self-reporting during a computerized reading task. During this process, a remotely placed eye tracker tool recorded eye gaze data. Models were created using six supervised conventional machine learning (ML) algorithms: logistic regression, k-nearest neighbors (k-NN), support vector machine (SVM), decision tree, random forest and naive Bayes. Machine learning models were trained on eye gaze dataset and evaluated using 5-fold cross validation. We measured the performance using area under the receiver operating characteristics (AUC-ROC) score, AUC-ROC curve, and confusion-matrix. To further improve the AUC-ROC score and other evaluation metrics, we trained standard neural networks and deep learning models using the data. Four sets of deep learning architectures were trained and evaluated. We found that dense neural network with one dimensional convolutional layer (DNN+Conv1D) outperformed the performance of conventional machine learning models. Naïve Bayes achieved mean test AUC-ROC score of 0.6595 and mean test accuracy of 0.6416. DNN+Conv1D beat the AUC-ROC score and achieved a score of 0.8024 and mean test accuracy of 0.7278. Our implementation used missing data values rather than discarding them which in fact improved our results. Our findings also showed that an automated mind wandering detection using deep learning models generalize well for new participants. This finding may help laboratory studies of mind wandering and for building systems to detect attention of inattentive drivers, students, or people in other work contexts that need focus to improve performance on a task.

Keywords: mind wandering, deep learning, eye tracking, LSTM, GRU

ACKNOWLEDGEMENTS

Firstly, I would like to thank Dr. Derek Harter because of his immense support to accomplish this research. I would like to thank Dr. Abdullah Arslan, Dr. Yuehua Wang and Dr. Burchan Aydin for their support as a committee member. Finally, I consider myself one of the fortunate persons to get the opportunity to be a student at Texas A&M University-Commerce.

I would like to dedicate this thesis work to my parents and wife who unconditionally contributed to my life and supported me throughout the whole journey.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

FIGURE

Chapter 1

INTRODUCTION

As a human being, it is a common phenomenon for us to drift away from focusing on an essential task to being distracted by unessential work or information. For example, we may be engaged in listening to a lecture, reading a book, or other task-related jobs but all of a sudden, we may realize that our task-related thoughts deviate from necessary to unnecessary thoughts like what to have for lunch, interacting with social-media, news or other life related anxieties & worries. Mind wandering (MW) refers to such unintentional attentional deviations, from being focused on the task at hand, to drifting to internal thoughts that are unrelated to what is being worked on (Schooler et al., 2011). Previous research demonstrated that this kind of attentional drifting occurred very frequently (Galéra et al., 2012; Kane et al., 2007; Killingsworth & Gilbert, 2010; Stawarczyk et al., 2014). The amount of mind wandering can be mild to severe and depends on the type of task being performed. For instance, one gigantic study, which consisted of 5000 people from 86 countries, found 46.9% of the time mind wandering was occurring using data collected from an iPhone app (Killingsworth & Gilbert, 2010). There are many similar studies like this that found mind wandering using various experiments such as Carciofo et al. (2014), Giambra (1989), Mrazek et al. (2012), Plimpton et al. (2015).

Thus, mind wandering is not a simple incidental thing. The incidents affect person's performance negatively on the task they are trying to perform. The consequences of this negative impact can range from negligible to life threatening for dangerous tasks that require very high levels of concentration, patience, and perseverance. Correlation between mind wandering and decreases in human performance is evident in meta-analysis studies such as D'Mello et al. (2013), Randall et al. (2014). It is obvious that performance will deteriorate if people need to focus on more than one thing simultaneously, especially if focus drifts from task-related to task-unrelated thoughts. In fact, evidence shows because of mind wandering, the rate of error increased when trying to detect a relevant piece of information to a task being performed (Robertson et al., 1997; Smallwood et al., 2004). This performance decrease can have large consequences. It can negatively affect memory tasks such as reading (Feng et al., 2013; Smallwood et al., 2007). Mind wandering can also lead to less precise recall of information (Seibert & Ellis, 1991; Smallwood & Schooler, 2006) when being tested afterwards to see how well a piece of written information was remembered.

The negative effects of mind wandering can be decreased proactively (Aron, 2011) or reactively (Verbruggen et al., 2014). Proactive refers to taking actions to prevent or control mind wandering before it occurs, and reactive refers to taking steps after mind wandering happens. Meditation is one of the popular ways to proactively control mind wandering but it requires months of training and patience to achieve and even sometimes it is not possible for all to adopt with this tactic. Moreover, this method has one major limitation in that it cannot eliminate mind wandering entirely and it has no mechanism to control the issue if it occurs. Therefore, the second method, or reactive intervention seems to have more promise to reduce mind wandering. This is especially true because we can potentially apply reactive interventions by detecting instances of mind wandering and then reactively responding. In this methodology, intervening while mind wandering occurs, is the major step. In fact, a robust and fast detector of mind wandering instances is necessary to perform such interventions. Eye behavior has been shown to be a good indicator of attention or mind wandering in people. In fact, eye gaze features such as fixation duration, pupil diameter, eye blink frequency and other features are useful indicators to detect mind wandering (Bixler & D'Mello, 2016). More research relating to the link between eye gaze behavior and mind wandering are given in detail in the Chapter 2 literature review.

Therefore, there is enough evidence that mind wandering is indeed correlated to a reduction in performance for tasks that require high attention and focus. So, detecting mind wandering in a user-independent and automated fashion is a promising approach to improving human performance and efficiency when engaged with tasks using technological systems.

Statement of the Problem

In this study, the research problem is that mind wandering negatively affects the performance of people who are engaged in jobs like driving, office jobs, studying, reading and other important tasks requiring focus. So, if detection of the attentional shifting is possible at an early stage, it will be possible to improve the performance, perhaps dramatically, by intervening. A promising approach to detecting mind wandering is through using eye gaze features that can in theory be captured continuously and in real time. Previous work using standard ML techniques has shown that it is possible to classify and detect mind wandering behavior from eye gaze data. But more recent techniques applying so-called state of art deep learning methods have not yet been applied to this type of data. Therefore, applying deep learning techniques using these eye gaze data may detect mind wandering in a user-independent and automatic fashion. And further, successful implementation may eventually improve the accuracy and other evaluation metrics (AUC-ROC, confusion-matrix) of the detection as well over existing techniques.

Purpose of the Study

The main purpose of this study is to improve the AUC-ROC score and accuracy of mind wandering detection on eye gaze data using deep learning algorithms. An additional purpose is to check whether the features of eye-gaze data are significant enough to detect mind wandering or not. In the process of achieving this goal, other steps of data preprocessing such data cleaning, visualization, data loading and other popular techniques using data pipelines are implemented. Other mind wandering detecting features such as non-eye gaze-based features are not considered for this study. We successfully replicated and matched performance from past work on six conventional machine learning algorithms support vector machine (SVM), logistic regression, naïve Bayes, k-NN, random forest and decision tree to detect mind wandering in the eye gaze data. Final implementation using deep learning based algorithms such as convolutional neural network, long short-term memory (LSTM) and gated recurrent units (GRU) may outperform the performance of conventional machine learning models.

Hypotheses

In this study, the research hypotheses are as follows:

1. Machine learning and deep learning models will detect mind wandering using 62 global eye gaze features.
2. Deep learning (DL) based algorithms such as Conv1D, LSTM and GRU can detect mind wandering because of the data pattern that contains the time series data.
3. Using Conv1D, LSTM, and GRU layers, we will outperform the performance of conventional machine learning models.

Research Questions

The following questions are expected to be answered from this research implemented using deep learning algorithms for detecting mind wandering:

1. Can the available 62 data features be used for mind wandering detection?
2. Will the deep learning methods outperform the conventional machine learning models?

3. What are the main factors that improve the performance of deep learning algorithms (i.e., dropout, batch normalization or some other parameter)?
4. How much performance improvement will the classifier be able to achieve?
5. What parameters (i.e., number of nodes, batch size, number of layers and so on) of the algorithms will improve the performance most?

Significance of the Study

In this study, we extended the traditional machine learning techniques to deep learning techniques to see if recent advances in deep learning techniques improve capabilities of the detector to improve the performance. The result of this study will be sufficient enough to detect a drift in attention for resolving many issues that decrease human performance where people need to be highly attentive, focused and task oriented. Further the whole process will be implemented in a fast, accurate, user-independent, and automatic fashion, and thus should be suitable for real-time classification and detection. In fact, to best of our knowledge, this is first time mind wondering detection is done using deep learning algorithms from eye gaze data. Furthermore, there are many applications of mind wandering detector that will be discussed in the following applications section.

Definitions of Terms

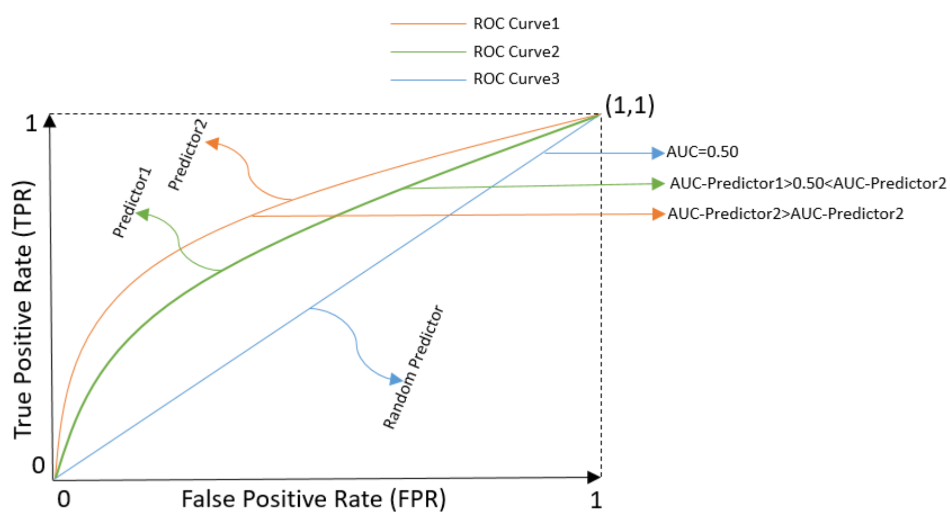
Activation function. Activation functions are the functions that transform the output of weighted sum to feed into other nodes or simply output the classes. In deep learning, non-linear activation functions are used. Non-linear activation functions can capture the complexity of the data. Activation functions must be differentiable due to the necessity of backpropagation.

Accuracy. Accuracy refers to among all true classes how many of them are correctly predicted by the model.

AUC-ROC. ROC stands for receiver operating characteristic. ROC curve is a plot between TPR vs FPR at different classification thresholds. AUC stands for area under the curve. AUC basically measures the area under the ROC curve. AUC-ROC is a performance measurement metrics for classification tasks. Especially, this metric is very useful when we have an imbalanced class in the dataset. Figure 1 shows AUC-ROC curve and a brief explanation.

Figure 1

AUC-ROC Curve



Best estimator. Best estimator refers to the best value or optimum value from parameter search space after performing a grid-search.

Bias. If y =mx +c is an equation of a straight line where m = slope, c=y-cut. Then, in machine learning, this y-cut is known as bias.

Cross validation. This is way of evaluating and train any model. Where first same dataset is randomly chosen for training and a part of the dataset is held back for testing the model.

Classification. When we classify two or more classes using machine learning models, this is known as classification problem.

Confusion-matrix. Confusion-matrix is a performance measurement for classification task. Table 1 shows a confusion matrix for binary classification.

True Positive (TP) = predicted positive and it was actually positive.

True Negative (TN) = predicted negative and it was actually negative.

False Positive (FP) = predicted positive but it was actually negative (Type-I error).

False Negative (FN) = predicted negative but it was actually positive (Type-II error).

Table 1

Confusion-Matrix

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Conventional machine learning. We refer supervised machine learning algorithms as conventional machine learning algorithms which excludes deep learning.

Chance model. Any random model that does not really perform well but can be used for reference to compare with better model.

Dropout. Dropout is technique to randomly drop some percentage of neurons in layers.

Data pipeline. Data pipeline refers to a combination of techniques that help preprocess data.

Data transformer. Data transformer are the techniques that transform a dataset. In our research, we inherited a class from scikit-learn library to introduce data transformer for our data pipeline.

Deep learning. Deep learning is a field of machine learning that imitates human brains process in order to make decisions.

Exploratory data analysis (EDA). Exploratory data analysis (EDA) is a common term in machine learning community in order to visualize, summarize and interpret the hidden information in dataset.

Evaluation metrics. Evaluation metrics are the metrics that are used for evaluation of trained models.

Feature rankings. Feature rankings refer to a process of ranking features according to their importance.

F1-score. F1-score is calculated using the formula: 2x ((Precision*Recall)/ (Precision + Recall)). So, F-1 score is basically the average of precision and recall.

FPR. FPR stands for false positive rate. FPR is the ration between FP and (FP+TN). So, basically, how many times the model got it incorrect.

Feature. In simple words, feature refers to a measurable characteristic of data that describe itself.

Grid-search. Variation in hyper-parameter values produce various outputs thus to search optimum values of hyper parameters we form a grid of values for which we try same models. Grid-search basically try all possible combination of hyper parameters.

Global features. Global features refer to common/existing features in a certain community such human.

Hyper-parameters. Hyper-parameters refer to the parameters that control the learning process in machine learning.

Kernel. Kernel in deep learning refers to filters that are used to convolution. 3x3, 5x5, 7x7 kernels are common in deep learning.

Keras tuner. Keras tuner is a library in keras that search for optimal values from a set of hyper-parameters.

Machine learning. Machine learning is a field of artificial intelligence (AI) where a system can be trained using data and learn various patterns and make decision by its own.

Mean test AUC-ROC. If we measure AUC-ROC score on multiple test dataset then mean of multiple AUC-ROC scores is known as mean test AUC-ROC score.

Mean test accuracy. Mean test accuracy refers to the mean of multiple accuracy on testing dataset.

Model. A trained machine learning algorithm on a certain dataset is known as model. Model sometimes refer to classifiers due to its ability to classify various classes.

Overfitting. When model try to memorize the dataset. For this reason, it performs well during training time but performs low in testing time.

Parameter search space. Parameter search space refers to hyper-parameters or parameters that are defined with a value or range of values.

Precision. Ratio between TP and (TP+FP). Denominator is actually predicted positive values which is the positive rows in confusion-matrix.

Pearson correlation. Pearson correlation is a statistical technique that provides coefficient to measure correlation between features. The coefficient value ranges from -1 to +1 where 1 means high and 0 means no correlation.

ReLU. ReLU stands for Rectified linear unit. The equation of RELU is f (x) = max (0, x)

Regression. When we try to predict numeric values, this is known as regression problem.

Regularization. Regularization is used for preventing overfitting problem, regularization techniques is applied that basically penalizes the cost function of a machine learning algorithm. There are L1 and L2 regularization.

Reference paper. We refer to Faber et al. (2018) as our reference paper. This paper contains the original analysis of the dataset used in this research, where we improved upon their past results in building a mind wandering classifier for the eye tracking data gathered in the reported experiment.

Supervised machine learning. Supervised machine learning is a type of machine learning where the data is labeled.

Sigmoid function. Sigmoid function is an activation functions that maps the output between 0 and 1.

Swish activation function. Swish is an activation function developed by google team that outperforms ReLU activation function. Swish (z) =z x sigmoid (Bz) = z/ (1+e (-Bz)

Tensorflow. Tensorflow is an open-source machine learning library.

TPR. TPR stands for true positive rate. TPR=TP/ (TP+FN). So, basically, how many times the model got it right. Also, this is known as recall.

Winsorization. Winsorization is a statistical technique in order to reduce the effects of outliers in data.

Applications

Learning by reading is a very common way but mind wandering hinders this way of learning (Moreno & Mayer, 2002). Thus, any user interfaces that are connected to reading texts may be improved by detecting mind wandering (Feng et al., 2013). In fact, any tasks that require attention may benefit from MW detection models (Kam et al., 2012). For example, if a driver drives a truck, it is critical, they maintain good focus during the task, thus if there is a system that can automatically detect mind wandering that may improve the quality of driving and play a role to reduce the accidents (Qu et al., 2015). There are more applications of MW detectors, for example, MW detector may be useful to any tasks that need tremendous amounts of concentration to extract critical information from passages (Dixon & Bortolussi, 2013; Smallwood et al., 2007). From a learning content perspective, we can use MW detector to improve the content of learning and informational material. For example, if we can detect in which part of resource material a person's mind often wanders, we can improve that part of the content. In fact, MW detector can be applied to any general subject since features like fixation duration and, saccade lengths are common features to many tasks such as watching movies, attending an e-learning session etc. (Bixler & D'Mello, 2016). Therefore, MW detector can be applied in a wide range of applications to improve performance and efficiency.

Limitations

In our research study, limitations are the following:

1. Implementing deep learning algorithms require a high-performance Graphical Processing Unit (GPU) which is often expensive and not available for all.
2. We are relying on a pre-existing dataset collected by another research group. If we could have collected the data on our own or developed our own dataset, we might

expect better results. In fact, more data is always better, and the size of the data set could be an issue for deep learning methods. Furthermore, familiarity with the details of the dataset could be drawbacks in fully using it to its best potential.

Organization of Thesis Chapters

This research is organized in the following fashion. Chapter 1 consists of an introduction where we discuss the problem, purpose of the study, hypothesis & research questions, and significance of our study and definition of terms. In addition, the limitations and assumptions are discussed in this chapter. Chapter 2 discusses the past literature relating to the research. In this chapter, we discuss the existing methodologies relating to mind wandering detection and further divided into three subsections, namely non-eye & eye gaze-based detection methodologies and comparing this study with the previous studies. In Chapter 3 we discuss method of procedure. In Chapter 4, we describe the findings of the research and finally in Chapter 5, we will summarize the research findings, including conclusions and future recommendation works.

Chapter 2

REVIEW OF THE LITERATURE

Mind wandering detection can be divided into two categories, firstly non-eye gaze based and secondly eye-gaze based mind wandering detection which is our main area of interest in this research. There is a close relation between mind wandering and attentional state. Before going into the topic of interest it is important to know the difference between attentional state estimation and mind wandering as well as how they are correlated to each other.

Attentional State Estimation

Determining attentional focus is commonly known as attentional state estimation. There are various reasons behind estimating the attentional state. For example, knowledge of the attentional state of a person can be used to improve productivity in general (Börner et al., 2014; Buntz, 1981; Kersten & Murphy, 2006; Paul et al., 2010; Wellins, 1991), improve the productivity of employees (Jabar et al., 2011; Jensen, 2011; Sharma & Gupta, 2004), can drive attention thus improving performance (Dong et al., 2010; Kaplan et al., 2015; Su et al., 2006), and in improving interactions in virtual environments (Barbuceanu et al., 2011; Kaur et al., 1998; Li & Zhang, 2014).

Shifting attention due to secondary tasks is one of the major significant factors for 78% of traffic accidents (Choe et al., 2018). Secondary cognitive tasks are things such as texting, conversation, or controlling other elements in a car. Estimating drivers' attentional state is one of the means to minimize the severity of traffic accidents. Driving behaviors (Bellet et al., 2009; Kaber et al., 2012) such as wheel & speed controls and physiological responses (Palinko et al., 2010; Zhang et al., 2004) such as heart rates, EEG, and eye movements are variables that have been studied to estimate the driver's attentional state.

Student’s attention estimation during class lectures has also been studied as an effective way to improve learning performance. Facial expressions (Butko et al., 2011; Calvo & D’Mello, 2010; Whitehill et al., 2014) are used to determine whether a student is tired, interested or confused and attentional state is estimated using face gaze, head motion, body postures etc. (Dinesh & Bijlani, 2016; Won el al., 2014). Zaletelj (2017) proposed a methodology to compute features from Kinect sensor data and use machine learning methods to predict the attentional state estimation of the students.

There are various areas where estimating attention is very important for different reasons. For example, Barbuceanu et al. (2011) proposed a user interface to detect the user’s intention and use the information to improve the virtual environment for user’s satisfaction. Vergara et al. (2019) proposed a methodology to assess the psychophysiological state i.e., attentional state using the finger temperature. Other examples such as head pose estimation (Gollan et al., 2011), dynamic scene viewing (Mital et al., 2011) etc. have been studied in the literature.

Although attentional state is somehow related to mind wandering, it cannot detect mind wandering solely. Mind wandering is a temporal incident while our neurocognitive systems go into a transient decoupling state from external environment because of the drifting from natural thought process (Kam et al., 2013). MW has a close tie with attentional disengagement which is related to boredom (Eastwood et al., 2012). Moreover, detection of mind wandering is tough due to its covert nature during inattention (D’Mello et al., 2016, May). Mind wandering characteristics particularly include looking at an aspect of the external environment and during this event of looking the users gaze location indicates locus of the attention. While attentional state and MW both deal with identifying the focus of user’s attention, MW is nothing but a disconnection between internal thought and external environment i.e., the displayed items. This disconnection is termed as ‘perceptual decoupling’ (Schooler et al., 2011). Therefore, though attentional state is related to MW, it is not enough to detect mind wandering by itself only. Further approaches to detect MW are discussed in the following sections.

Non-Eye Gaze Based Detection

Non-eye gaze based mind wandering has been detected using neurology (Gruberger et al., 2011), physiology (Blanchard et al., 2014), facial features (Stewart et al., 2017), and reading & textual features (Mills & D’Mello, 2015). Two aspects of neural systems, executive and default network regions, are associated during mind wandering in our human brain (Christoff et al., 2009). In fact, in Christoff et al. (2009), they used a functional magnetic resonance imaging (fMRI) data by experience sampling and found that executive network recruitment and default network functions-, two brain systems working in opposition, are able to detect mind wandering. But the cost of an fMRI ranges from 500,000 to 3 million USD depending on the resolution, thus real time mind wandering is not feasible currently using this kind of brain scan technology. In Hosseini and Guo (2019), they used EEG signals to detect mind wandering as EEG signals are much cheaper thus real time detections are feasible.

Physiological aspects such as skin conductance and skin temperature have been used to try and detect mind wandering. Blanchard et al. (2014) used this kind of technique using probe-based self-mind wandering reports data and used supervised machine learning techniques to measure the kappa values for detecting significant features from the collected data. Pham and Wang (2015) used heart rate and lecture content features to detect mind wandering. Their methodology is currently used by mobile phone applications and users and these aspects make their techniques more fascinating as the users are increasing rapidly.

Reading behaviors and textual features are also able to detect mind wandering. In Mills and D'Mello (2015), they achieved Cohen's kappa=0.207 and AUC=0.609 while using reading time and level of difficulty features that were extracted from self-paced log files to classify MW. This study also used supervised machine learning when the users navigated from screen to screen in an attempt to build a MW classifier. Franklin et al. (2011) used reading behavior features as well to detect MW. In this study they used a word-by-word reading paradigm to check whether a user being monitored could detect instances of MW in real time fashion. The reported method was able to achieve 72% accuracy while the expected accuracy was 49%. But this kind of word-by-word paradigm is time consuming and less efficient than standard ML techniques for building a classifier.

Facial features have shown some significant results to detect MW. In Stewart et al. (2017), they introduced the first student-independent facial feature-based MW detector using a lab study consisting of a 32.5 min narrative film show and self-reported mind wandering data. Facial features & additional body movements were extracted using computer vision techniques. Using supervised machine learning technique, their methodology achieved a F1 score of 0.390 which is a 31% improvement over a chance model. Their detector is particularly useful for online learning contexts. Therefore, other MW detecting techniques need to be adopted and these will be discussed in the following sections.

Eye Gaze Based Detection

Tracking a subject's eye gaze is one of the important measures used to determine users' attention during a task (Sibert & Jacob, 2000). For example, Sibert and Jacob (2000) used eye gaze features to detect the interaction between computer and human. Eye gaze features like eye movement play a role for mind wandering during events like reading (Just & Carpenter, 1980;

Rayner, 1998; Reichle et al., 1998). Other characteristics of eye gaze such as blinks (Foulsham et al., 2013; Frank et al., 2015; Smilek et al., 2010), pupil diameters (Hartmann & Fischer, 2014; Unsworth & Robinson, 2018; Vinski & Watter, 2013), eye fixation (Bixler & D'Mello, 2016; Krasich et al., 2018; Uzzaman & Joordens, 2011) & saccades (Bixler &D'Mello, 2014; Hutt et al., 2016; Uzzaman & Joordens, 2011) are responsible for or useful in detecting mind wandering. In Smilek et al. (2010), they found more eye closures (blinks) and fewer fixations on the text occur when the mind is wandering. In this study, eye blinks and mind wandering showed a close relation when blink rates and on-task periods of reading are compared. Their methodology used 15 graduate students reading two passages from a simple general science passage while measuring reading comprehension and attention. Eye blinks were counted using the EyeLink 1000 tool and the experiments continued for 15 min reading. Finally, twelve of the 15 subjects were included and found that three subjects were reported as mind wandering. In fact, more blinks were found while on task, contrary to some reports where blinks seem to indicate mind wandering. In Foulsham et al. (2013), mind wandering was detected using an experiment called controlled stimuli, where slower reading times, longer average fixation duration and an absence of word frequency are features detected as the consequences of MW. In this experiment, participants were told to read a series of sentences including high and low frequency words and by using eye tracking methodology they found whether there were mind wandering or they were on task. Thus, longer fixation is indeed an indication of mind wandering. Another eye behavior, the diameter of the pupil, showed an indication of mind wandering based on its size. In Smallwood et al. (2011), they found decoupling during spontaneous cognitive activity of brain using the measurements of pupil diameter. On a given task if the users gave incorrect responses, then this was assigned as inattention and during this process of inattention the mean pupil diameter was larger than usual from when users were giving correct responses. Similarly, Vinski and Watter (2013) found larger pupil diameters during the periods of detected mind wandering as well. In addition, in Unsworth and Robison (2018) four experiments were performed to examine the association between arousal state and mind wandering states. Their experiments found a smaller pupil diameter and smaller phasic pupillary responses during on task behavior. In the process of experiments the participants attended a sustained attention task and thought probes were used to detect whether the participants were on task or mind wandering. During the whole process the responses of the pupil were continuously recorded to keep track of mind wandering.

However, the patterns of the eye behavior are not consistent all the time as a few studies showed that the pupil diameter was smaller while MW was taking place. For example, Franklin et al. (2013) found exactly the opposite effect, correlating smaller pupil diameter with measures of MW. The possible reason of this inconsistency was discussed in Bixler and D'Mello (2016). They hypothesized that the differences among the tasks and methodologies such as-, some using single fixation during experiments, was mainly responsible for the inconsistency. Similarly, several papers showed that fixation duration was longer than usual during MW (Bixler & D'Mello, 2016; Uzzaman & Joordens, 2011). In fact, some researchers found no effect on pupil diameter during MW. But eye gaze is a tested feature for mind wandering detection in most cases (Bixler & D'Mello, 2016). Moreover, it is an omnipresent characteristic in all human beings. In fact, detecting mind wandering using this feature has proven its importance in the area of e-learning, safety driving and so on (DelSignore et al., 2016; Galéra et al., 2012). There are many existing implementations that have achieved very good accuracy using the eye gaze data for mind wandering detection. For instance, D'Mello et al. (2013) used 17 global and 12 local features collected from 84 participants MW data using classification task and downsampling was performed to achieve 60% accuracy.

Our reference paper used supervised machine learning models: bagging, with REPTree as a base learner, Bayes net, naïve Bayes, logistic regression, support vector machine, k-nearest neighbors, decision table, C4.5 decision tree, random forest, REPTree and random tree. Their implementation included data outlier treatment using Winsorization. They down sampled data set in order to treat the class imbalance issue. Feature importance applied using weak and high correlation techniques with the mind wandering reports. Finally, 25%, 50% or 75% top-ranked features were used for feature rankings. Using leave-one-participant-out validation method, they evaluated their trained models and found 17 best models with each having AUC-ROC>0.63 and all of the best performing models were logistic regression model.

In the following sections we will discuss current studies to further improve the accuracy using eye gaze data.

Current Study

This study uses the dataset extracted from a computerized reading task, and detects mind wandering in a user-independent and automated fashion. Computerized reading data has been a tried and tested resource for detecting mind wandering in many research such as (Al-Balushi & Al-Harthy, 2015; Bixler & D'Mello, 2014; Bixler & D'Mello, 2016; D'Mello et al., 2016; Jackson & Balota, 2012; McVay & Kane, 2012; Mills & D'Mello, 2015). In fact, mind wandering has a negative correlation with reading comprehension (Feng et al., 2013; Franklin et al., 2011; Kopp et al., 2015). Moreover, acquiring eye gaze data from reading is a common existing activity, with good secondary measure of attention using reading comprehension, that will help us generalize our methodologies into other interfaces as well (Bixler & D'Mello, 2016).

In this study, data extraction has been done using a probe-caught method while an experiment was done during reading on a computer screen. After extracting features, we applied conventional supervised learning methodologies to determine which features were most significantly correlated with the labeled mind wandering instances. Further details will be discussed in the data collection section.

In this research, the methodologies are different from existing previously mentioned research. In the research, firstly, we implement deep learning methodologies to try and improve the accuracy of MW detection, rather than typical machine learning algorithms. Secondly, this work covers a wide range of features that are user-independent and can be collected in an automated fashion. Previous studies such as Drummond and Litman (2010) did not consider attentional state as MW. Thirdly, to the best of our knowledge, this is the first-time deep learning algorithms is used in eye-gaze data to detect mind wandering.

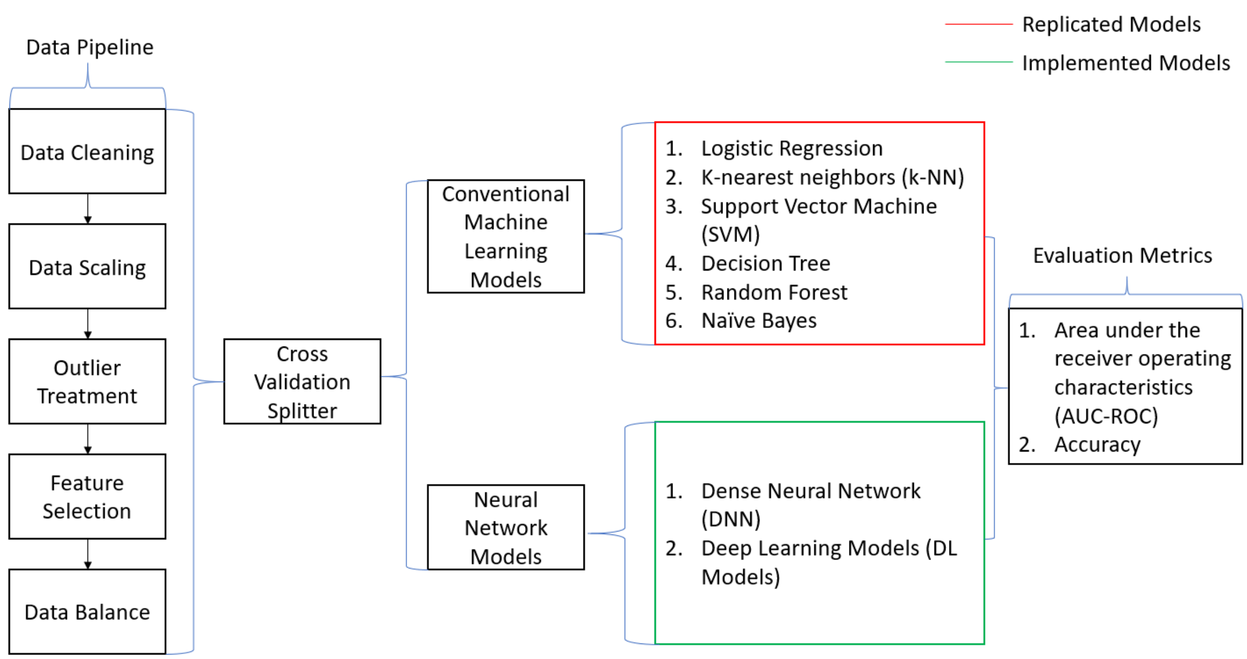
Chapter 3

METHOD OF PROCEDURE

Our methodology consists of three parts. First, we prepared a data pipeline in order to do data preprocessing. Second, we trained the prepared mind wandering dataset on conventional machine learning and neural network based classifier models. Finally, we evaluated our trained models using two evaluation metrics, namely area under the receiver operating characteristics (AUC-ROC) curve and prediction accuracy. Standard 5-fold cross validation was used to train the models and evaluate them on the average performance on the held back test folds. Figure 2 shows our overall implementation methodology and training pipeline.

Figure 2

Diagram for Our Methodology



Dataset

Supervised machine learning and deep learning models were trained on a labeled dataset where targets were labeled as "mind wandering" or "not mind wandering". We used a collected eye-gaze dataset from another study in order to train the models. There were 135 unique participants, and their mind wandering was recorded during a computerized reading. Mind wandering can be detected using either self-caught or probe-caught methods. Our dataset used self-caught labels where each participant reported whether their mind wandered or not during the reading. During the process, various eye-gaze features were recorded for later use. According to our reference paper, global features are the features most likely to have significant impact in being able to detect mind wandering. Thus, we extracted 62 global features using our collected raw dataset. The 62 features were combined from four sets of global features. The four sets of global features were: eye movement descriptive features (48 features), pupil diameter descriptive features (8 features), blink features (2 features), and miscellaneous gaze features (4 features). Tables 2, 3, 4, and 5 show these four sets of features with short descriptions of each from the four categories of global features.

Table 2

Eye Movement Descriptive Features

Eye movement features	Description
Fixation duration	Fixation, also known as visual fixation, is the practice of keeping one's visual sight fixed on a single point.

Table 2 (continued)

Eye movement features	Description
Fixation duration median Fixation duration mean Fixation duration standard deviation Fixation duration minimum Fixation duration maximum Fixation duration range Fixation duration skew Fixation duration kurtosis	Statistics of fixation duration: median, mean, standard deviation, minimum, maximum, range, skewness, kurtosis
Saccade duration	Between two successive fixations, the time in milliseconds
Saccade duration median Saccade duration mean Saccade duration standard deviation Saccade duration minimum Saccade duration maximum Saccade duration range Saccade duration skew Saccade duration kurtosis	Statistics of saccade duration: median, mean, standard deviation, minimum, maximum, range, skewness, kurtosis
Saccade amplitude	In pixels, the distance between two successive fixations.

Table 2 (continued)

Eye movement features	Description
Saccade amplitude median Saccade amplitude mean Saccade amplitude standard deviation Saccade amplitude minimum Saccade amplitude maximum Saccade amplitude range Saccade amplitude skew Saccade amplitude kurtosis	Statistics of saccade amplitude: median, mean, standard deviation, minimum, maximum, range, skewness, kurtosis
Saccade velocity	During a saccade, the eye's highest angular speed
Saccade velocity median Saccade velocity mean Saccade velocity standard deviation Saccade velocity minimum Saccade velocity maximum Saccade velocity range Saccade velocity skew Saccade velocity kurtosis	Statistics of saccade velocity: median, mean, standard deviation, minimum, maximum, range, skewness, kurtosis
Saccade angle	Angle between the x-axis and the saccade in degrees

Table 2 (continued)

Eye movement features	Description
Saccade angle absolute median	Statistics of saccade angle: absolution
Saccade angle absolute mean	and relative measurement of median,
Saccade angle absolute standard deviation	mean, standard deviation, minimum,
Saccade angle absolute minimum	maximum, range, skewness, kurtosis
Saccade angle absolute maximum	
Saccade angle absolute range	
Saccade angle absolute skew	
Saccade angle absolute kurtosis	
Saccade angle relative median	
Saccade angle relative mean	
Saccade angle relative standard deviation	
Saccade angle relative minimum	
Saccade angle relative maximum	
Saccade angle relative range	
Saccade angle relative skew	
Saccade angle relative kurtosis	

Table 3

Pupil Diameter Descriptive Features

Pupil diameter descriptive features	Description
Pupil diameter	pupillary diameter of eye
Pupil diameter median	Statistics of pupil diameter: median, mean, standard deviation, minimum, maximum, range, skewness, kurtosis
Pupil diameter mean	
Pupil diameter standard deviation	
Pupil diameter minimum	
Pupil diameter maximum	
Pupil diameter range	
Pupil diameter skew	
Pupil diameter kurtosis	

Table 4

Blink Features

Blink features	Description
Number of blinks	Number of blinks in the window as a whole
Blink duration mean	Mean of blink duration

Table 5

Miscellaneous Gaze Features

Miscellaneous gaze features	Description
Number of saccades	Number of saccades in the window as a whole
Horizontal saccade proportion	The percentage of saccades that are 30 degrees above or below the horizontal axis.
Fixation dispersion	The root mean square of the distances between each fixation and the window's average fixation position
Fixation saccade duration ratio	Fixation duration to saccade duration ratio

Thus, the entire dataset consists of 62 features. There was a total of 4076 trials of data consisting of trials collected from 135 total experiment participants. Most features had all values for all trials, only one feature had missing data that had to be dealt with (see discussion below). The whole dataset then consisted of a (4076, 62) shaped tensor used for training all models, and a single 4076 sized vector of binary true/false labels indicating if mind wandering occurred or did not occur for each of the 4076 trials.

In the original self-report of mind wandering, subjects could report no mind wandering during a trial. There were 2963 of the 4076 trials therefore with no mind wandering reported. During a trial though, subjects reported the number of instances of when their mind wandered. There were cases with 1, 2, 3 or 4 mind wandering instances during a trial. In the models reported in this research we performed a binary classification task, so all trials with 1 or more instances of mind wandering trials in this data set are classified as mind wandering trials. The dataset was therefore a bit skewed, about ¾ of trials were not mind wandered, and only ¼ represent mind wandered events. We will discuss this skewness more in the following sections.

Model Toolkits and Software

We used the Scikit Learn (Pedregosa et al., 2011), Keras (Chollet, 2017) and Tensorflow (Abadi et al., 2016) machine learning and deep learning software libraries as primary tools, mainly for modeling, data exploration, data cleaning, evaluation, and visualization. All libraries are part of the Python scientific programming and data analysis ecosystem of tools. We used standard pipelines and transformers defined in Scikit Learn for most of the data cleaning and transformations, with some custom cleaning transformations to extract the features of the original data set. Standard ML, neural network and deep learning models and architectures were explored from both Scikit Learn and Keras to build binary classifiers of the mind wandering data. Data cleaning and modeling work are discussed in detail in the following sections.

Data Pipeline

We used a pipeline to preprocess the eye-gaze dataset. Data preprocessing involved five techniques: data cleaning, data scaling, outlier treatment, feature selection and data balancing.

Our raw dataset actually consisted of a shape of (4078, 129), or 4078 trials with 129 collected features. Out of 129 raw features, the first 12 features were experimental meta-data information. A few examples are ‘ParticipantID’, ‘TrialID’, ‘TrialIndex’ and so on. We explored experimental meta-data, if necessary, for our implementation and eventually we dropped them to clean the dataset. In our reference paper, they used 62 features by dividing them into four sets of global features. We extracted and used the same 62 features, cleaning and renaming them for ease of use in our modeling environment pipelines. Out of four sets of global features, only global blink features had missing values. We assumed that missing blink values indicated that there were no blinks during that trial thus we filled missing blink count features as zeros. The data transformation steps we describe in the following sub-section replicated the transformations reported in the original paper on this work, though we added some additional transformations as well that were explored in our effort to improve upon the original reported model performance.

Data Scaling

We scaled/normalized our dataset using standard scaling and min-max scaling. We used both scalers to provide grid-search implementation more search options to explore. Standard scaling techniques scale each feature so that it has a mean of 0 and a unit variance (1). The standard Scikit Learn scaler transformer was used to transform the dataset using standard scaling. The Scikit Learn min-max scaler transformed all features to have values between 0 and 1 for the dataset.

Outlier Treatment

We used the Winsorization technique for outlier treatment. Following our reference paper, where values were 3 standard deviations above or below the mean they were replaced with that value (clipped to be no more than 3 standard deviations from mean). So, for example when using standard scaling, all features were scaled and had mean of 0 and standard deviation of 1, and we thus replaced the values below -3 or above 3 standard deviations away from mean with -3 or 3 respectively to Winsorize the features.

Feature Selection

We used standard Pearson correlation to calculate correlation between features & features (pairwise correlation) and features & target. If scores of correlations between target and features were high, then it was a sign of good features and target variables. However, high scores between features can be bad, indicating highly correlated and thus redundant features. We calculated a weighted average by inverting the correlation score between features. Pandas .corr() method was used to calculate pairwise correlation and then we performed weighted average formula using our own written function.

A cutoff target was selected to perform a type of pre-feature selection, to remove features with the worst combined weighted scores. This weighted correlation feature trimming was introduced again to replicate described work in the original reference paper, though we explored varying the cutoff target percentage, and also trained models without this pre-feature correlation trimming.

To prevent overfitting, 66% of participants were randomly chosen and then we performed feature rankings on this sample. The process was repeated five times to reduce the variance caused by random selection. Then 20%, 25%, 30%, 35%, 40% or 50% of the top-ranked features were kept for training purposes. That means 20% of 62 features, 25% of 62 features, 30% of 62 features equal to 12, 15, 18 and so on. Again, this represented a replication of the primary feature selection method described in the original work to create mind wandering models of this dataset.

Data Balancing

As mentioned previously, the binary labels of this data set are a bit imbalanced, with 2963 trials or 73% reporting no mind wandering, and only 1113 trials or 27% reporting mind wandering occurred. By convention for binary classification, the 27% mind wandering trials are considered as positive instances (it is true that mind wandering occurred in these trials) and the 73% are considered as false labels. This type of skewed labeled dataset can be problematic when training classifiers. Due to this class imbalance, we performed data balancing using two techniques. In the first technique, we performed downsampling of the dataset. Downsampling simply randomly eliminates majority classes to make a dataset balanced with the minority classes. In the second technique, we performed oversampling using Synthetic Minority Oversampling Technique (SMOTE; Chawla et al., 2002) in which a synthetic dataset of the minority class is created. SMOTE represents a type of data augmentation technique, as added minority classes are not simply replications of existing trials, but random combinations of features of randomly selected minority class trials. The original reference paper only performed down sampling balancing; we added the newer up sampling to the mix to see if it would improve performance. Figure 3 shows the technique of down sampling the dataset and Figure 4 shows the technique of over sampling the dataset.

Figure 3

Downsampling of Original Dataset

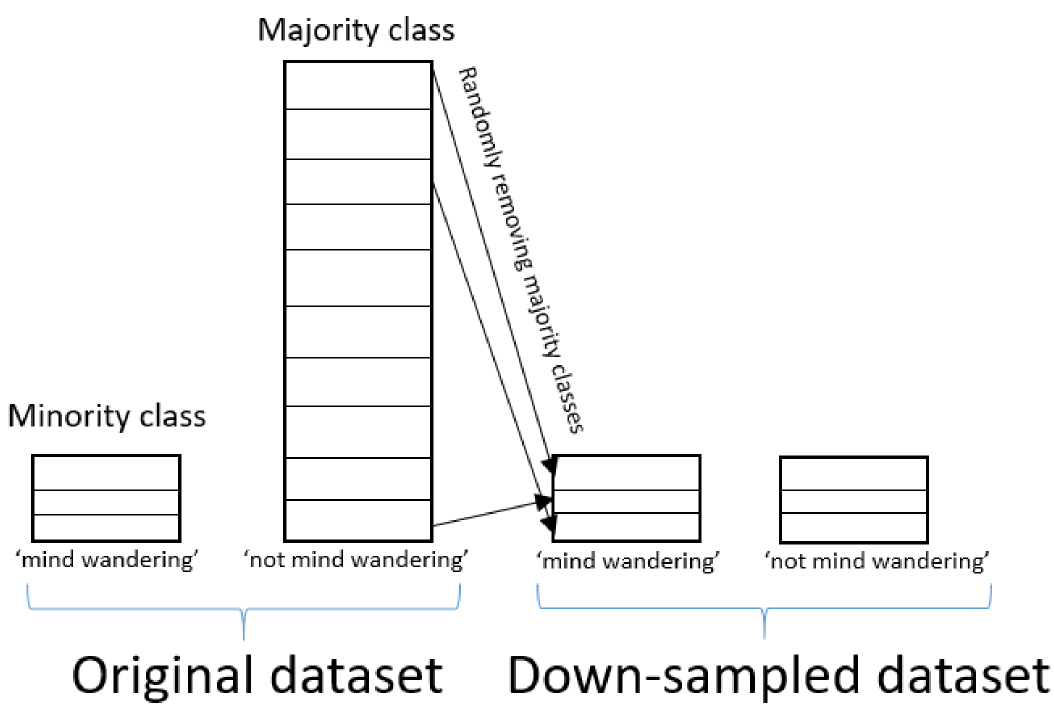
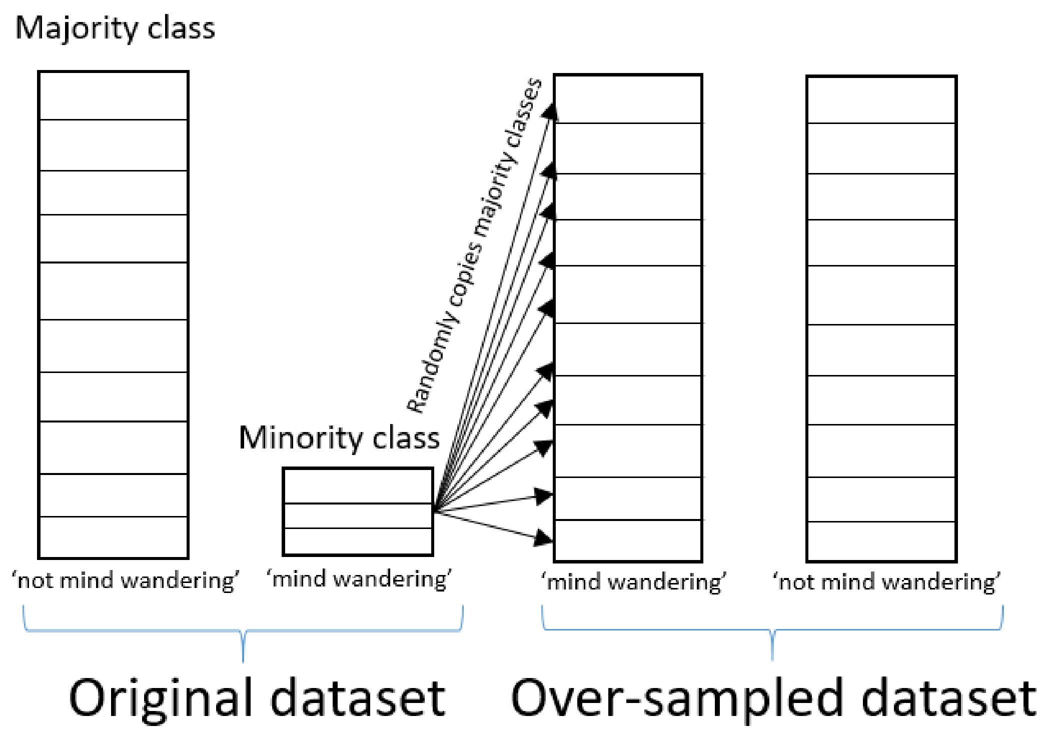


Figure 4

Oversampling of Original Dataset



Model Selection

First, we replicated a list of classifiers that were mentioned in the reference paper. Six conventional machine learning algorithms were selected to implement. The considered machine learning (ML) models were logistic regression, k-nearest neighbors (k-NN), support vector machine (SVM), decision tree, random forest and naïve bayes. Then, we implemented neural network based models divided into two categories: dense neural network (DNN) models and deep learning (DL) models. All of the standard ML models were created using SciKit Learn library objects, and neural network and DNN models were created from Keras library layers and classes.

Models were evaluated using standard 5-fold cross validation. Reports of model performance in the next sections are on the average over the 5 folds evaluation on AUC-ROC and accuracy. Model accuracy is of course the ultimate measure we would like to maximize. This is simply the report of the percentage of predictions the model gets correct. But accuracy can often be a misleading measure of model performance when trying to make a classifier from an imbalanced data set. For example, a model that just always predicts no mind wandering would be expected to be 73% accurate on this data set. A better measure of performance is looking at the confusion matrix and calculating the area under the curve of receiver operating characteristics (AUC-ROC). AUC-ROC is a performance measure of a model under varying thresholds for deciding to output true/false for a binary classification label. An AUC-ROC score of 0.5 is a model only performing at chance level. So, for example, if we always guess true, we will end up with an AUC-ROC score of 0.5 for this dataset, because that is only chance performance. Higher scores represent a better ability to predict mind wandering as mind wandering and no mind wandering as no mind wandering, e.g., to correctly discriminate and to minimize false positives and false negatives. An AUC-ROC score of 1.0 represents perfect performance, making correct predictions on all of the test data for the model.

Logistic Regression

Logistic regression is a technique which uses a sigmoid function, $1/(1 + e^{-value})$ to predict probabilities of the classes. We used Scikit Learn's built-in method to perform logistic regression implementation. A grid of parameters was first defined then we passed these parameters through our pipeline to perform a grid-search to find an optimum logistic regression model. In the process of performing grid-search, we maximized AUC-ROC and accuracy. Then, we printed out top performing models with highest AUC-ROC and we displayed other parameters as a table. We evaluated a trained logistic regression model by plotting a confusion-matrix, AUC-ROC curve and predicting accuracy score on grid-search best estimator. Finally, we plotted a histogram consisting of AUC-ROC score in x-axis and number of models in y-axis which is how many models produced that AUC-ROC score.

K-Nearest Neighbors (k-NN)

k-nearest neighbors (k-NN) supervised machine learning algorithm works both for classification and regression problems. The basic idea of this algorithm is that it labels the data based on measures of distance or similarity. Close data points belong to the same labels if the distance measure truly captures some sense of similarity between different instances. We used the Scikit Learn library to implement the k-NN algorithm. First, we pass the raw dataset through the data pipeline then use grid-search in order to find the optimum k-NN model. In the process of performing grid-search, we maximized AUC-ROC and accuracy. Then, we printed out top performing models with highest AUC-ROC and we displayed other parameters as a table. We evaluated the trained k-NN model by plotting a confusion-matrix, AUC-ROC curve and predicting accuracy score on grid-search best estimator. Finally, we plotted a histogram consisting of AUC-ROC score in x-axis and number of models in y-axis which is how many models produced that AUC-ROC score.

Support Vector Machine (SVM)

Support vector machine (SVM) is another popular supervised machine learning algorithm which is used for classification and regression tasks. The data points are called support vectors. The main idea of SVM is to find a hyperplane that can classify the classes successfully. We implemented SVM using the scikit-learn library. First, we passed the raw dataset through the data pipeline and then performed grid-search in order to find the optimum SVM model. In the process of performing grid-search, we maximized AUC-ROC and accuracy. Then, we printed out top performing models with highest AUC-ROC and we displayed other parameters as a table. We evaluated a trained SVM model by plotting a confusion-matrix, AUC-ROC curve and predicting accuracy score on grid-search best estimator. Finally, we plotted a histogram consisting of AUC-ROC score in x-axis and number of models in y-axis which is how many models produced that AUC-ROC score.

Decision Tree

In decision tree machine learning algorithms, a decision has been taken from a tree-like model based on features defined condition. A decision tree mainly has three components: a decision condition (internal node), branches (edges) and decision (leaf). Similar to previous implementations, we implemented a decision tree algorithm using Scikit Learn and performed a grid-search to find the best decision tree model. Then we evaluated the best model with the best set of parameters. Evaluation metrics AUC-ROC and accuracy were maximized during the training process and then an AUC-ROC histogram was plotted showing the maximum score. As usual, a confusion-matrix was generated and evaluated the model's performance on the best estimator.

Random Forest

Random forest is a powerful machine learning algorithm which can be used both for classification and regression. The main idea of random forest is based on decision trees. This algorithm is basically an ensemble of decision trees. A list of decision tree’s outcomes is assembled together (for example by majority vote) in order to produce the maximum correct model. There are versions of random forest such as bagging or boosting. We used a simple random forest classifier and performed a grid-search after data preprocessing through our mentioned data pipeline to find the best optimum random forest model. Then, we evaluated the best model with the best set of parameters. Evaluation metrics AUC-ROC and accuracy were maximized during the training process and then an AUC-ROC histogram was plotted showing the maximum score. In addition, a confusion-matrix was generated and evaluated the model's performance on the best estimator.

Naïve Bayes

The Naïve Bayes algorithm is based on the Bayes theorem. Bayes theorem states that probability of an event ($event_1$) can be identified given an event ($event_2$) occurred. The theorem can be written as: P ($event_1$) = (P ($event_2|event_1$) P ($event_1$))/P ($event_2$). For a binary classification problem, we can write the formula as: P (y | X) = (P (X | y) P(y))/P(X).

Where y is for our case mind wandered (1) or not mind wandered (0) and X are the eye-gaze features. One assumption is that the events are independent, and features have equal effect on the classification decision. Among various Naïve Bayes implementations, we implemented Gaussian Naïve Bayes that assumes the feature values are from normal distribution. We used Scikit Learn's GaussianNB() method and then used our data pipeline in order to process the data and performed grid-search in order to find the optimum Naïve Bayes model with best parameters. . Then, we evaluated the best model with the best set of parameters. Evaluation metrics AUC-ROC and accuracy were maximized during the training process and then an AUC-ROC histogram was plotted showing the maximum score. In addition, a confusion-matrix was generated and evaluated the model's performance on the best estimator.

Dense Neural Network (DNN)

In dense neural networks (DNN, also known as fully connected neural networks), layers are densely connected to each other. Each layer has one or more neurons. Neurons are also known as units. Neurons are the basic building block of DNN.

We implemented a sequential feed forward DNN using tensorflow and keras. Keras provides keras tuner, which is a hyper parameter tuning library. The tuner function is like grid search from Scikit Learn and is a fast and reliable way to test a range of hyper parameter values to tune DNN model performance. Therefore, we used keras tuner in order to find optimum number of units, activation function, kernel & bias regularization, dropout rate and learning rate of the tuned models. While tuning the hyper parameters, our goal was to maximize the AUC-ROC and accuracy scores. Binary cross entropy was used as the loss function, which is a standard loss function choice for binary classification. Final dense layer output consisted of one unit with sigmoid activation function in order to predict the binary class probabilities. Keras tuner provides three hyper parameter tuning techniques namely Random-search, Hyper-Band-search, and Bayesian Optimization. We used these three techniques in order to get an idea about the hyper parameters. Our final goal was to perform a grid-search exactly similar to the implemented machine learning models. Once we found the best hyper parameters for DNN, including the number of layers, we built our model which consisted of three dense layers and two dropout layers. We used 50 neurons in the first dense layer. Our selected activation function was the swish activation function. We used L1 & L2 kernel regularization with l1=1e-5 and l2=1e-4. L2 regularization was used for activity and bias regularization. Then, a dropout layer of value 0.3 was used to deal with the overfitting issue. Second dense layer consisted of 64 neurons and swish activation. Then, a dropout layer was used with a value of dropout rate of 0.2. Final output layer consisted of a single neuron with sigmoid activation function to produce class probabilities for our mind wandering detection classifier. Table 6 gives a summary of the DNN model layers used.

Table 6

DNN Model Summary

Layer (Type)	Output Shape	Parameters
Dense1(Dense)	(None,50)	3,150
Dropout1(Dropout)	(None,50)	0
Dense2(Dense)	(None,64)	3,264
Dropout2(Dropout)	(None,64)	0
Dense3(Dense)	(None,1)	65
	Total parameters:	6,479
	Trainable parameters:	6,479
	Non-trainable parameters:	0

To perform grid-search cross validation, we first needed to convert the keras model to be compatible for Scikit Learn using keras classifier wrapper function. Then as with our previous machine learning model implementations, we passed the data through our data transformation pipeline. We trained DNN models for various epochs: 100, 150, 200; various batch size: 8, 16, 24, 32, 40, and 48. These ranges were passed into grid-search parameter grid in order to tune for the optimum model performance. We used seven optimization algorithms: adadelta, adagrad, adam, adamax, nadam, RMSprop, stochastic gradient descent (SGD). During training, our goal was to maximize AUC-ROC and accuracy scores. Then, we evaluated the trained DNN model on the testing data set which was an automatic process during cross validation (we used 5-fold cross validation). We tested model accuracy on best estimator and produced confusion-matrix, and AUC-ROC curves. Finally, we generated similar AUC-ROC histogram where x-axis had the AUC-ROC values and how many models AUC-ROC scores were in that range plotted in y-axis.

Deep Learning (DL) Models

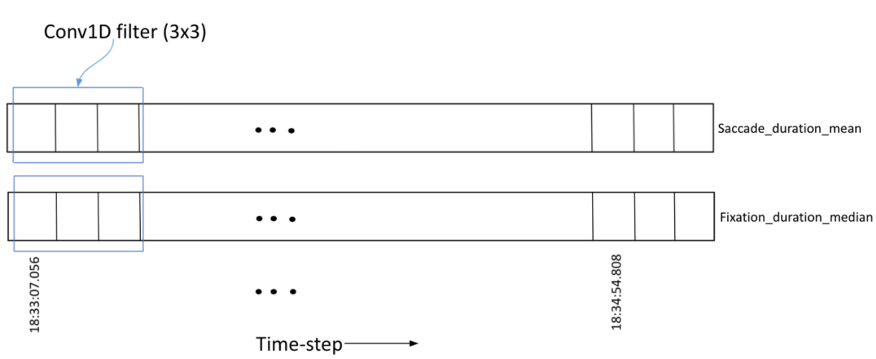
In order to maximize the AUC-ROC and accuracy scores, we further implemented neural network models using convolutional layers (conv_layer) as the first layer. Our main idea was that the data set was in time series format. Thus, convolutional layers may improve the AUC-ROC and accuracy scores as both convolutional networks and recurrent networks are known to be useful at modeling time series data (Fawaz et al., 2019).

Our first challenge was to use the same data transformer pipeline to implement DL models using convolution. In fact, convolution operation modifies the shape of the dataset depending on the shape of the filters. For example, if input to a convolution layer is (None, 12, 1) and if we use 64 convolution filters then the output will be (None, 12, 64). So, we could not use our same data transformer concurrently with convolutional operation. To resolve this issue, we first passed our dataset through the same set of best parameter grids generated by best ML model and then we trained the variations of our DL models. We considered the same set of best parameters generated by the best machine learning model (k-NN) to compare whether using the same setup for DL can outperform the ML models or not. k-NN produced best AUC-ROC score using parameters values: feature selection parameter, k = 12, data balancing techniques ‘allknn’, data scaling technique ‘standard scaling’. So, we scaled the dataset using standard scaling, used k=12 during feature selection and used ‘allknn’ technique for down sampling the dataset.

We used a Conv1D layer as the first layer. Convolution is a mathematical operation that brings out the best local features from the dataset. Convolution is widely used for image datasets. Before understanding Conv1D, concepts about Conv2D may help. Conv2D works by using 2D filters which are trainable in DL. This 2D filter moves along the 2 dimensions of an image and keeps extracting spatial features. Features in images may include edges, colors and other higher-level features. For example, if the images are pictures of dogs, and cats; filters may extract eyes, nose etc. Similar to Conv2D, Conv1D slides/moves along one dimension in order to extract local features. Conv1D is most suitable for time series data due to its learning capability. For our dataset, each participant’s mind wandering was recorded on a particular time-step. Each time-step had 62 features such as for participant 1002-UM the start time step was 2013-10-18, 18:34:54.808. Similarly for the same participant for another trial the time-step was 2013-10-18, 18:33:07.056 and so on. So, time-series dataset can be plotted based on time-step in x-axis as shown in Figure 5.

Figure 5

Conv1D Operation on Time-Series Data



First tensorflow and keras models were built using a Conv1D layer as first layer, followed by a flatten layer to prepare the output of the first layer for the next dense layer with 32 units, then a one-unit dense layer with swish activation and finally an action layer with sigmoid activation function. Then this model was converted in order to be compatible for grid-search using Scikit Learn. We then used the same data pipeline for this Conv1D implementation as with previous model tuning. To keep the same methodology as before, we performed a grid-search while tuning models batch size: 32, 48; epochs: 50, 60, 100; filters: 8, 16, 32, 64; kernel size: 3, 5, 7; optimizer: RMSprop, adagrad, adadelta, adam, SGD, adamax. Kernel size refers to the size of the 1D convolutions used in the convolution layer. Then we trained and evaluated the model's performance on the grid-search best estimator value by calculating accuracy, confusion-matrix, and AUC-ROC curve. Finally, a histogram of the AUC-ROC scores was generated in order to compare to the previous histograms. In Table 7, we give a summary of the DL convolutional network settings.

Table 7

Conv1D Model Summary

Layer (Type)	Output Shape	Parameters
Conv1(Conv1D)	(None,12,64)	384
Flatten1(Flatten)	(None,768)	0
Dense1(Dense)	(None,32)	24,608
Dense2(Dense)	(None,1)	33
OutputLayer (Activation)	(None,1)	0

Table 7 (continued)

Layer (Type)	Output Shape	Parameters
	Total parameters:	25,025
	Trainable parameters:	25,025
	Non-trainable parameters:	0

In addition to the Conv1D layer, we implemented a DL model using Conv1D and long short-term memory (LSTM) layers. LSTM is a recurrent network model which performs well on time-series datasets. Similar to previous Conv1D implementation, we performed a grid-search for parameters (parameters search space was the same as Conv1D) and during the training process, we maximized AUC-ROC and accuracy evaluation metrics. We evaluated the trained model on the best estimator for accuracy and generated a confusion-matrix & AUC-ROC curve. Finally, a histogram was generated for AUC-ROC scores. In Table 8, we summarize the parameters used for this LSTM and Conv1D DL model reported in these results.

Table 8

LSTM and Conv1D Model Summary

Layer (Type)	Output Shape	Parameters
conv1d_1(Conv1D)	(None,2,64)	384
Lstm_1(LSTM)	(None,16)	5,184
Flatten_1(Flatten)	(None,16)	0
Dense_2(Dense)	(None,32)	544

Table 8 (continued)

Layer (Type)	Output Shape	Parameters
Dense_3(Dense)	(None,1)	33
Activation_1(Activation)	(None,1)	0
	Total parameters:	6,145
	Trainable parameters:	6,145
	Non-trainable parameters:	0

In addition to the Conv1D and LSTM layer, we implemented a DL model using Conv1D and Gated Recurrent Unit (GRU) layers. GRU is another recurrent algorithm which performs well on time-series datasets. Similar to previous Conv1D implementation, we performed a grid-search for parameters (parameters search space was the same as Conv1D) and during the training process, we maximized AUC-ROC and accuracy evaluation metrics. We evaluated the trained model on the best estimator for accuracy and generated a confusion-matrix & AUC-ROC curve. Finally, a histogram was generated for AUC-ROC scores. Table 9 summarizes the parameters and layer shapes for the GRU and Conv1D DL model.

Table 9

GRU and Conv1D Model Summary

Layer (Type)	Output Shape	Parameters
conv1d_1(Conv1D)	(None,2,64)	384
gru_1(GRU)	(None,2,20)	5,160

Table 9 (continued)

Layer (Type)	Output Shape	Parameters
Flatten_1(Flatten)	(None,40)	0
Dense_2(Dense)	(None,32)	1,312
Dense_3(Dense)	(None,1)	33
Activation_1(Activation)	(None,1)	0
	Total parameters:	6,889
	Trainable parameters:	6,889
	Non-trainable parameters:	0

Chapter 4

PRESENTATION OF FINDINGS

Conventional Machine Learning Classifiers

In six conventional machine learning models, we defined a parameter grid to perform a grid-search which is summarized in Table 10:

Table 10

Conventional Machine Learning Model's Parameter Search Space

Models	Defined parameter search space
Logistic regression	model__solver: ['lbfgs', 'liblinear', 'newton-cg'],
	model__C: [numpy.logspace(1,6,18)],
	model__max_iter:[50000]
k-NN	model__n_neigbors: [3,5,11,19],
	model__weights: ['uniform', 'distance'],
	model__metric: ['euclidean', 'manhattan']
SVM	model__gamma: [0.1,0.01,0.001,0.0001],
	model__C: [0.1,1,10,100],
	model__kernel: ['rbf', 'poly']
Random forest	model__n_estimators: [200,500],
	model__max_depth: ['auto', 'sqrt', 'log2'],
	model__max_depth: [4,5,6,7,8],
	model__criterion: ['gini', 'entropy']

Table 10 (continued)

Models	Defined parameter search space
Decision tree	model__criterion: ['gini', 'entropy'], model__max_depth: numpy.arange(3,10), model__max_leaf_nodes: np.arange(3,10)
Naïve Bayes	model__var_smoothing: [1e-6,1e-7,1e-8,1e-9,1e-10,1e-11,1e-12]

In addition to the above parameter search space, each model's grid-search implementation included data pipeline parameters which were best feature selection parameter (k), data resampling techniques (allknn, smote-enn), outlier treatment thresholding (0,3). The best performing parameters are summarized in Table 11. The best reported model for each standard ML technique was the one with the best average AUC-ROC score obtained on the held back fold of the 5-fold cross validation.

Table 11

The Best Performing Parameters for Machine Learning Models

Models	Mean test AUC-ROC	Mean test accuracy	Param_k	Data Balancer	Other parameters
Logistic regression	0.6318	0.5635	18	allknn	C:10, newton-cg
k-NN	0.6310	0.6278	15	allknn	Euclidean, n_neighbors:19, weights: distance, scaling_type: standard

Table 11 (continued)

Models	Mean test AUC-ROC	Mean test accuracy	Param_k	Data Balancer	Other parameters
SVM	0.6588	0.6067	18	allknn	C:10, kernel: rbf
Random forest	0.6574	0.5967	24	allknn	N_estimators: 200, max_features: log2, max_depth: 8
Decision tree	0.6409	0.5449	15	allknn	Criterion: gini, max_depth: 9
Naïve Bayes	**0.6595**	**0.6416**	12	allknn	Var_smoothing:1e-06, outlier_threshold: 3

As we mentioned, all of our models were evaluated using confusion-matrix, AUC-ROC curves, and accuracy. The three best ML models confusion-matrix and AUC-ROC curves are shown in the following Figures. Figure 6, 7 and 8 show confusion-matrix of SVM, random forest, and naïve bayes classifiers. Detailed comparisons will be given later of these models in the performance section.

Figure 6

Confusion-Matrix for SVM

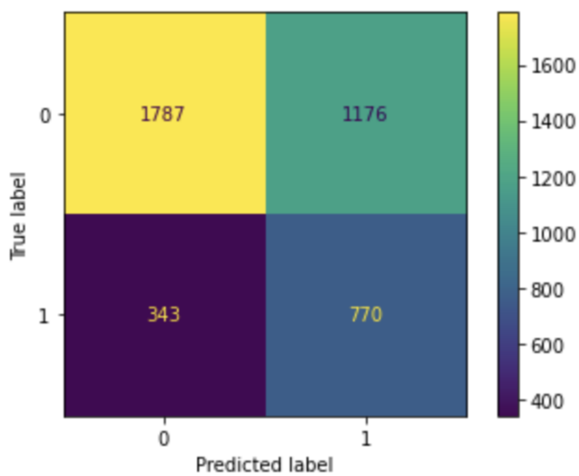
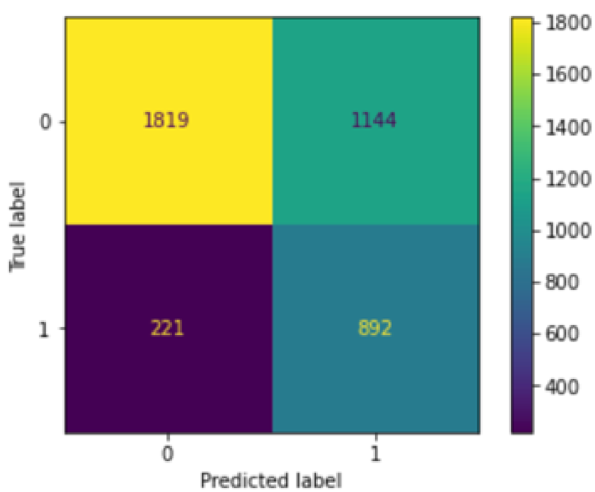


Figure 7

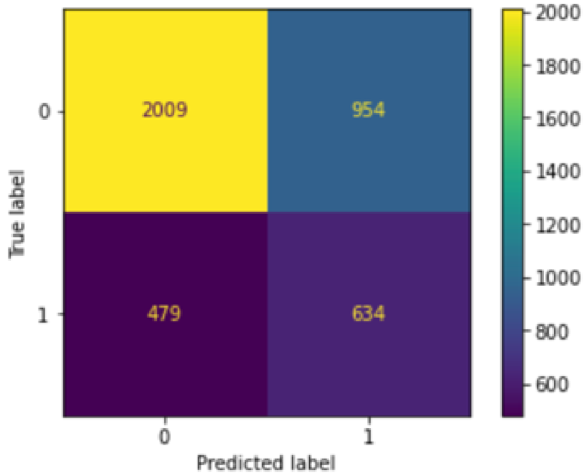
Confusion-Matrix for Random Forest



Since, Naïve Bayes was the winner among all conventional machine learning models thus this is our best performed model considering the evaluation metrics. Figure 8 shows confusion-matrix of our best machine learning model.

Figure 8

Confusion-Matrix for Naïve Bayes



In the next set of figures, three best models AUC-ROC curves are plotted. Figures 9, 10, and 11 show the AUC-ROC curves for the SVM, random forest, and naïve bayes model's performance.

Figure 9

AUC-ROC Curve for SVM

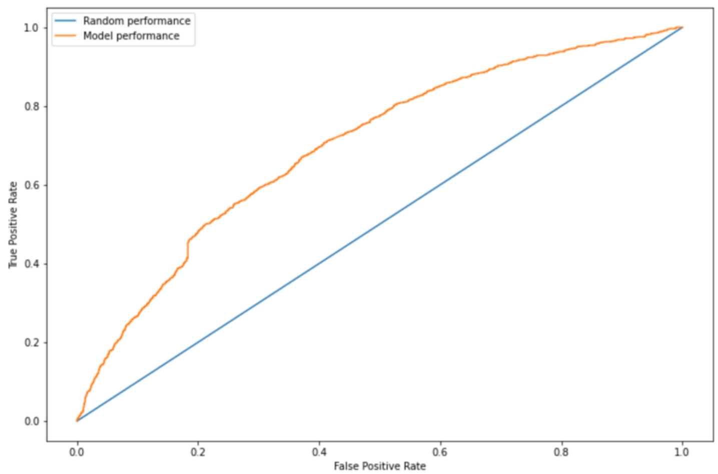
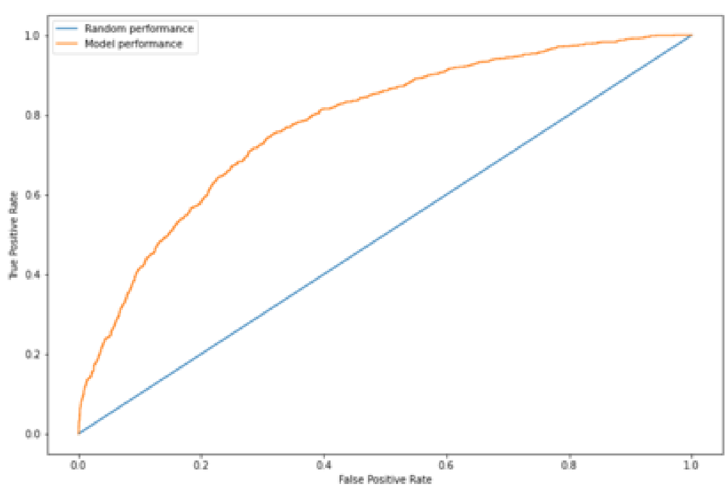


Figure 10

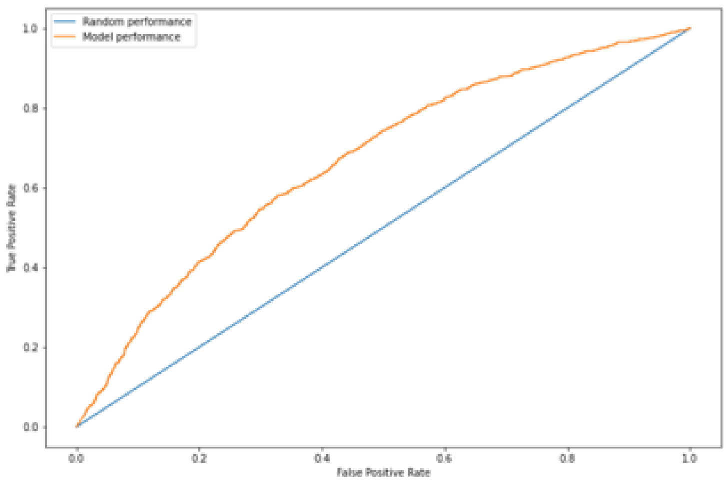
AUC-ROC Curve for Random Forest



As we mentioned previously that Naïve Bayes was the winner model, so we further evaluated this machine learning model using AUC-ROC curve as well. Figure 11 shows best machine learning models AUC-ROC curve.

Figure 11

AUC-ROC Curve for Naïve Bayes

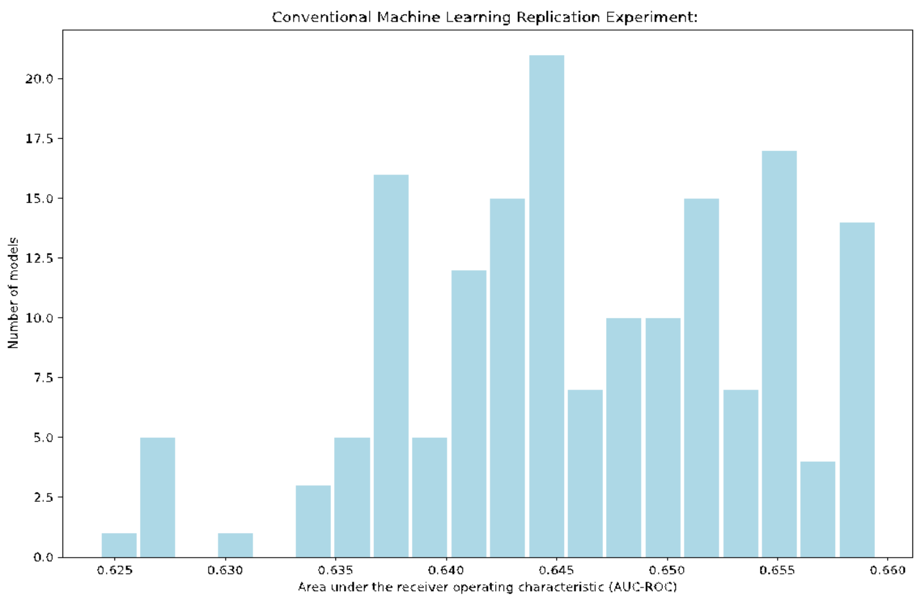


Finally, to get a feel overall for all of the models being trained in our parameter search tuning for the preliminary replication, we generated a combined histogram of AUC-ROC scores from the trained saved machine learning models for all 6 standard ML types tried to this point in our reporting of the results. Figure 12 shows the combined histogram of AUC-ROC scores. A total of 7,176 models were searched among the 6 standard ML techniques. The best model of all of these, as shown in Table 11, was a Naïve Bayes classifier, achieving an AUC-ROC score of almost 0.66, and an overall accuracy of about 0.64. For comparison, the original reported work on this data set reported the best classification they achieved was an AUC-ROC score of 0.64

with a logistic regression model. We will discuss further in our summary the comparison of our replication efforts with the original research.

Figure 12

AUC-ROC Scores of All Trained Conventional Machine Learning Algorithms



Performance Measurement and Findings

The confusion-matrix of the best implemented machine learning models shows performance that determines how many predicted labels were actually accurate labels or instead false negatives or positives. For example, Naïve Bayes model predicted 2643 true labels correctly and misclassified 1433 labels. The confusion matrix gives a good picture of the discrimination abilities of the best standard ML model found in each of the 6 classes explored in this work.

The AUC-ROC curves show the performance of our trained machine learning models. The AUC-ROC curve better indicates a model's discrimination performance than raw accuracy and is based on discrimination as the threshold value for determining true/false is varied. An AUC-ROC score of 0.5 indicates no discrimination power, and scores higher than this are better. The AUC-ROC is actually a measure of the total area under the model performance curve in the figure (orange line), which changes as the decision threshold is varied from 0.0 to 1.0. As you can see from the figure, the AUC-ROC score is actually a plot of the false positive rate to the true positive rate as the threshold is varied. In our AUC-ROC Figures, we first plotted (blue line) a random model's performance which has AUC-ROC of 0.50 (baseline no discrimination performance) then anything above (orange line) this straight line indicates better performance than no ability to discriminate.

In Figure 12, a combined histogram of all standard ML trained models has been plotted. The histogram and Table 11 show that Naïve Bayes was the model that achieved the highest mean AUC-ROC score of 0.6595. In fact, the 32 best models (each with AUC-ROC> 0.655) were all Naïve Bayes models. Therefore, Naïve Bayes performed better than any chance model (AUC-ROC=0.50) since the model achieved a well-above AUC-ROC score than the random model. And Naïve Bayes, since it was predominant among the top of the list of final AUC-ROC scores, appears to have been our best performer among the standard ML techniques explored.

Deep Learning Based Classifiers

Our deep learning implementation consisted of four variations: DNN, DNN+Conv1D, DNN+Conv1D+LSTM, DNN+Conv1D+GRU. Similar to conventional machine learning implementations, we performed a grid-search for these four variations. Table 12 shows the defined parameter space that was searched for our deep learning classifier models.

Table 12

Parameter Space for Deep Learning Models

Models	Defined parameter search space
DNN	model__batch_size: [32, 48], model__epochs: [50, 60], model__units: [8, 16, 32, 64], model__activation: [‘relu’, ‘swish’, ‘tanh’], model__optimizer: [‘RMSprop’, ‘Adagrad’, ‘Adadelta’, ‘Adam’, ‘SGD’, ‘Adamax’, ‘Nadam’], model_learning_rate: [0.01, 0.001, 0.1], model__momentum: [0.0, 0.2, 0.3, 0.4, 0.5]
DNN+Conv1D	model__batch_size: [32, 48], model__epochs: [50, 60], model__filters: [8, 16, 32, 64], model__kernel_size: [3, 5, 7], model__optimizer: [‘RMSprop’, ‘Adagrad’, ‘Adadelta’, ‘Adam’, ‘SGD’, ‘Adamax’]
DNN+Conv1D+LSTM	model__batch_size: [32, 48], model__epochs: [50, 60], model__filters: [8, 16, 32, 64], model__kernel_size: [3, 5, 7], model__optimizer: [‘RMSprop’, ‘Adagrad’, ‘Adadelta’, ‘Adam’, ‘SGD’, ‘Adamax’]

Table 12 (continued)

Models	Defined parameter search space
DNN+Conv1D+GRU	model__batch_size: [32, 48], model__epochs: [50, 60], model__filters: [8, 16, 32, 64], model__kernel_size: [3, 5, 7], model__optimizer: ['RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'SGD', 'Adamax']

In addition to the above parameter search space, each model's grid-search implementation included data pipeline parameters which were best feature selection parameter (k), data resampling techniques (allknn, smote-enn), outlier treatment thresholding (0,3). The best performing parameters for each of the four DN architectures tried are given in Table 13.

Table 13

The Best Performing Parameters for Deep Learning Models

Models	Mean test AUC-ROC	Mean test accuracy	Param_k	Data Balancer	Other parameters
DNN	0.6563	0.6004	12	allknn	model__epochs:100, outlier_threshold: 3
DNN+Conv1D	**0.8024**	**0.7278**	12	allknn	Batch_size: 48, filters: 8, kernel_size: 3, optimizer: RMSprop

Table 13 (continued)

Models	Mean test AUC-ROC	Mean test accuracy	Param_k	Data Balancer	Other parameters
DNN+Conv1D+LSTM	0.7389	0.4391	12	allknn	Batch_size: 64, filters: 8, kernel_size: 5, optimizer: RMSprop, model_epochs: 60
DNN+Conv1D+GRU	0.7405	0.5827	12	allknn	Batch_size: 64, filters: 8, kernel_size: 5, optimizer: RMSprop, Model_epochs: 60

The confusion-matrix of our four model variations was plotted. Here, we show the final confusion-matrix of DNN, and DNN+Conv1D models in Figure 13 and 14 respectively. Figures 15 and 16 show the confusion-matrix of our other two deep learning model variations, DNN+Conv1D+LSTM, and DNN+Conv1D+GRU.

Figure 13

Confusion-Matrix for DNN

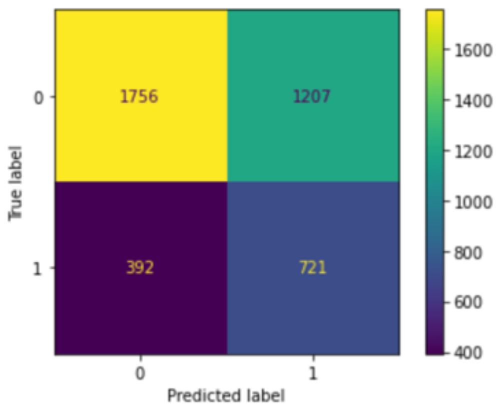
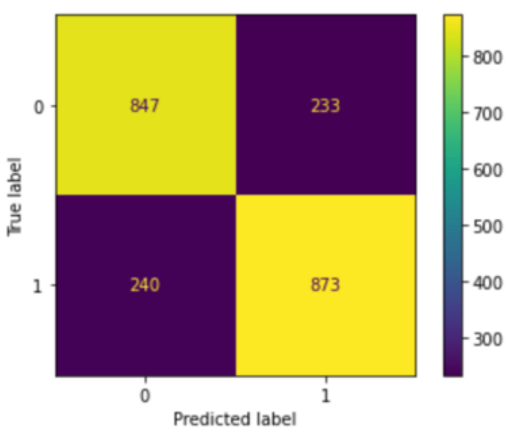


Figure 14

Confusion-Matrix for DNN+Conv1D



Confusion-matrix of DNN+Conv1D+LSTM, and DNN+Conv1D+GRU were generated using ‘allknn’ down sampling technique where the feature selection strategy was applied as well.

Figures 15 and 16 show the confusion-matrix of two variations that indicate DNN+Conv1D was the better performer since it predicted the labels more correctly than these two models.

Figure 15

Confusion-Matrix for DNN+Conv1D+LSTM

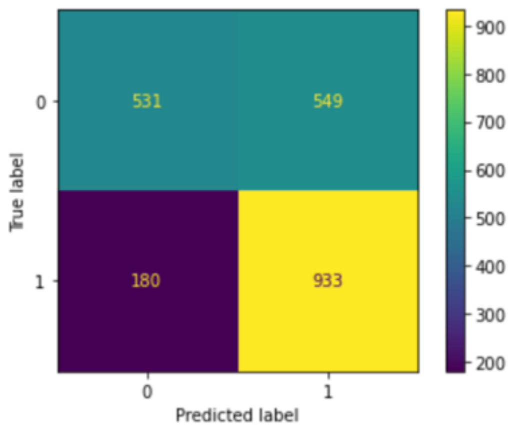
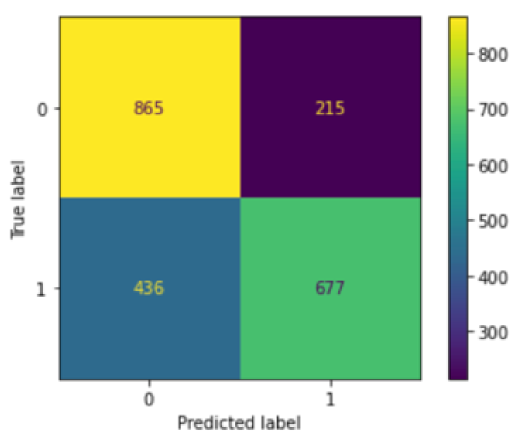


Figure 16

Confusion-Matrix for DNN+Conv1D+GRU



In the following figures, we show the final AUC-ROC curves for the final best model of our deep learning model results. Figures 17 and 18 show the plotted AUC-ROC curves for two variations DNN and DNN+Conv1D, and Figures 19 and 20 demonstrate AUC-ROC curves for two variations namely DNN+Conv1D+LSTM and DNN+Conv1D+GRU.

Figure 17

AUC-ROC Curve for DNN

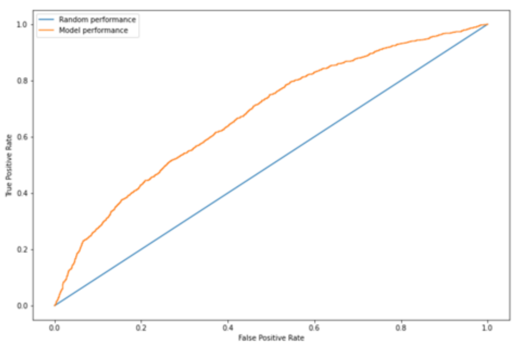
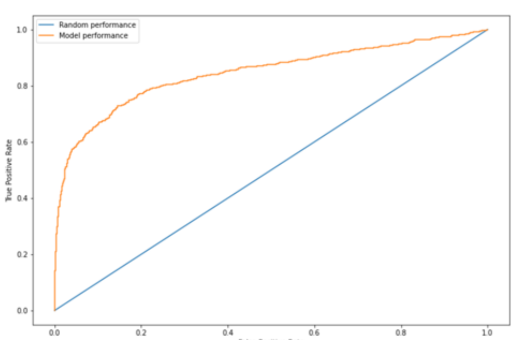


Figure 18

AUC-ROC Curve for DNN+Conv1D



AUC-ROC curves are indicators that how models have the capability to distinguish between classes. The curve more towards the TPR region, it indicates more accurate model. Figures 19 and 20 demonstrate AUC-ROC curves for two variations namely DNN+Conv1D+LSTM, and DNN+Conv1D+GRU.

Figure 19

AUC-ROC Curve for DNN+Conv1D+LSTM

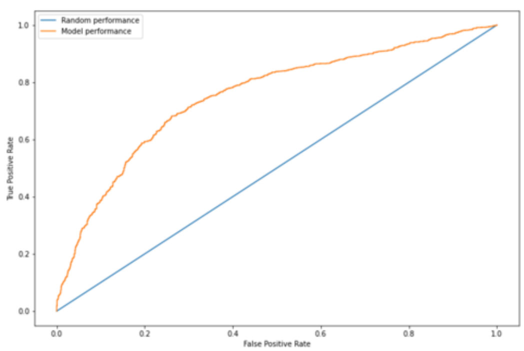
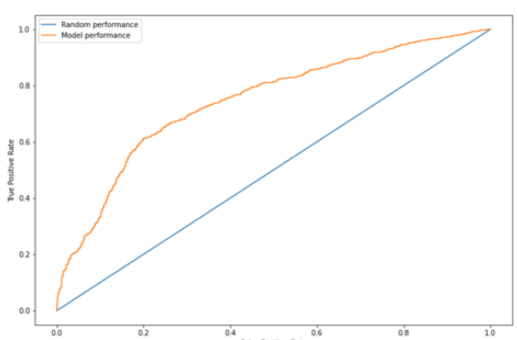


Figure 20

AUC-ROC Curve for DNN+Conv1D+GRU



Finally, we generated a combined histogram of AUC-ROC scores from the trained saved deep learning models. Again, the purpose of this histogram is to get a feel overall for the general range of performance seen while parameter tuning the various DL architectures that we explored. We split the histogram into two separate figures. In Figure 21, we show the histogram of the 216 models tried in our search of the traditional fully connected standard DNN architecture. In Figure 22, we show the histogram of the 240 models explored for the various combinations of convolutions and deep learning techniques applied to the mind wandering classification data task.

Figure 21

AUC-ROC Scores of DNN Parameter Search Results

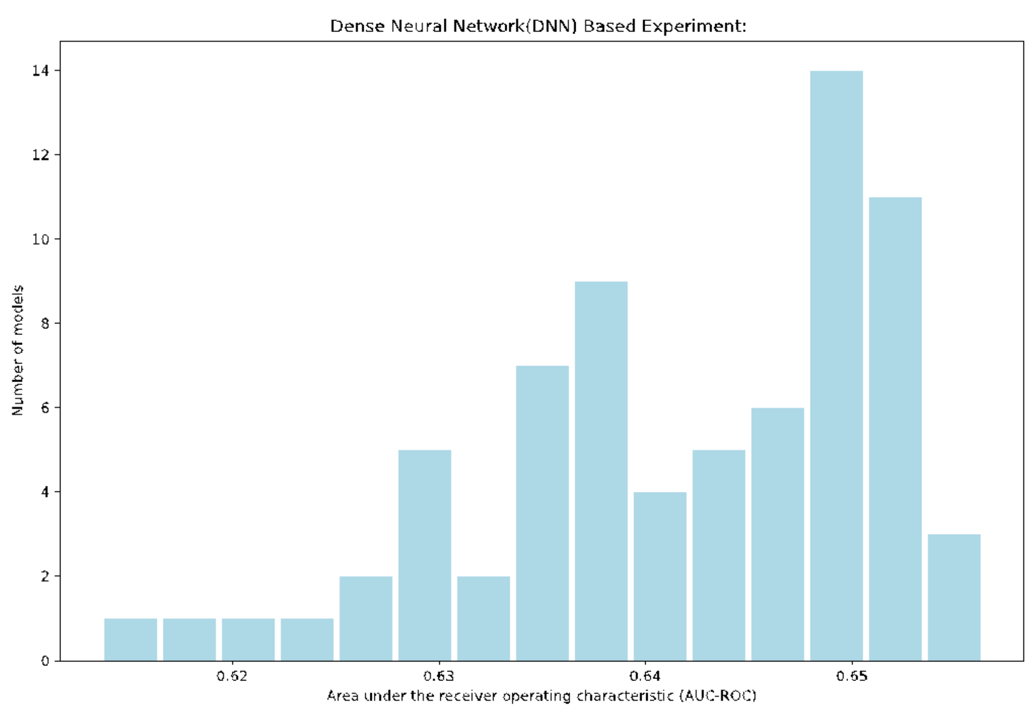
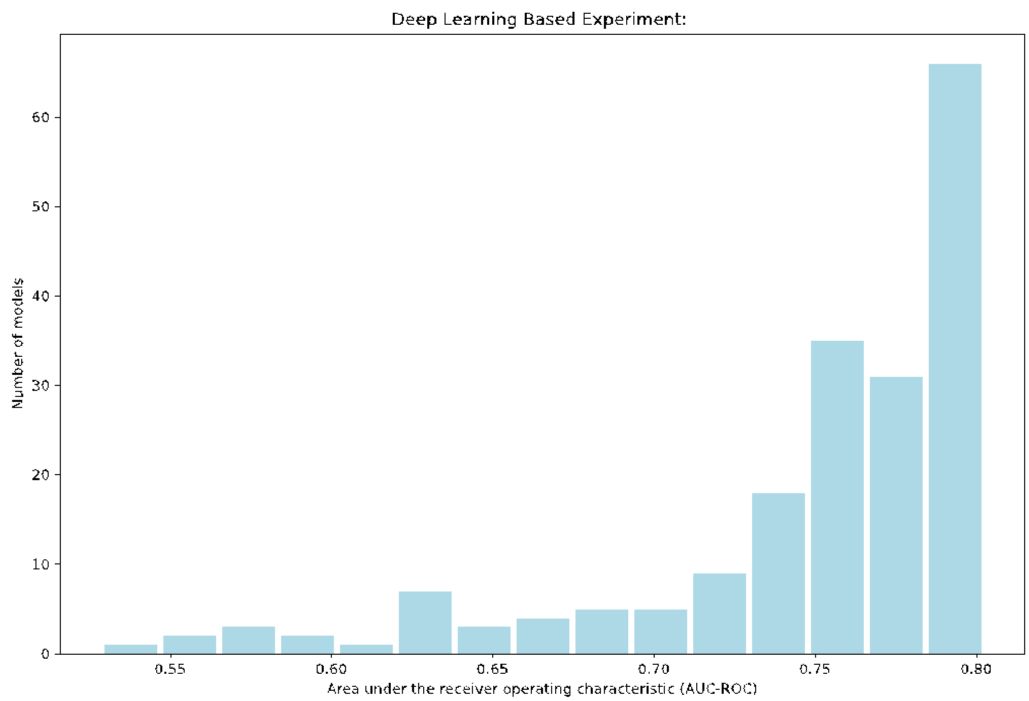


Figure 22

AUC-ROC Scores of DNN+Conv1D, DNN+Conv1D+LSTM, DNN+Conv1D+GRU Parameter Tuning Search



Performance Measurement and Findings

The confusion-matrices of our four variations of neural network and deep learning models show the discrimination performance of our best found model in each class. For example, the DNN model predicted 2477 true labels correctly and misclassified 1599 labels. The confusion matrix is helpful in understanding how well the binary classifiers are doing at discriminating and what kinds of mistakes, false positives and false negatives, they are making.

Figures 15 through 18 consist of AUC-ROC curves that show the discrimination performance of our four variations of neural network and deep learning models. The AUC-ROC curve for DNN+Conv1D model was the best model among the four models architecture explored, as shown by the model performance curve (orange line) and results reported in Table 13.

In Figures 21 and 22, combined histograms of the models explored have been plotted. The histograms give an idea of how the parameter search space performed across multiple models in the various classes of architectures explored. The histogram and Table 13 show that DNN optimized with SGD was the best model that achieved a highest mean AUC-ROC score of 0.6563 among the 216 DNN fully connected network models explored. In fact, the 2 best DNN models (each with above AUC-ROC> 0.655) were all DNN models optimized with SGD. Notice that the best DNN model was approaching our best standard ML (Naive Bayes) model, though it did not exceed its performance. We will discuss further the comparison between all tried models and the previous reference paper work in our results section next.

We further implemented three variations of deep learning models in order to beat the highest AUC-ROC scores achieved by conventional machine learning models. Figures 22 shows combined histogram of the three variations of deep learning architectures that we explored with parameter space tuning. The histogram and Table 13 show that DNN+Conv1D was the best model that achieved a highest mean AUC-ROC score of 0.8024, definitively beating the best performance of standard ML techniques. Approximately 4 best models (each with above AUC-ROC> 0. 80) were all DNN+Conv1D models. Therefore, DNN+Conv1D performed better than any chance model (AUC-ROC=0.50) and conventional machine learning models since the model achieved a higher AUC-ROC score than the random model & conventional machine learning models.

Overall Discussion for DL Models

DNN using one dimensional convolution outperformed the best AUC-ROC score of conventional ML model (Naïve Bayes). Generated confusion-matrix and AUC-ROC curves further strengthened the performance of DNN+Conv1D model. DNN+Conv1D model used same set of best parameters that were produced by best ML model. Therefore, the implementation of best DL model used a trimmed dataset due to down sampling for data balancing and this was addressed in our previous sections of this research. Thus, the confusion-matrix shows the performance on down sampled dataset in combination with feature selection and data scaling technique. In fact, the confusion-matrix of DNN+Conv1D model shows that the trained DL models can classify 1720 instances of labels correctly and 473 instances incorrectly. Further, the AUC-ROC curve of best DL model showed that it skewed more towards the TPR region that the ML best model. Finally, histogram of AUC-ROC scores showed that the best AUC-ROC score was 0.80 that was produced by DNN+Conv1D model.

Our assumption of DL implementation was that if Naïve Bayes was the best ML classifier, then it should produce the best set of parameters. Our overall best findings are shown in Table 14.

Table 14

Overall Findings of Best Models

	Models searched	How many models	Model	Mean test AUC-ROC	Mean test accuracy
Best ML	7,176	32 (AUC-ROC>0.655)	Naïve Bayes	0.6595	0.6416
Best DNN	216	2 (AUC-ROC>0.655)	Optimized with SGD	0.6563	0.6004
Best DL	240	4 (AUC-ROC>0.801)	DNN+Conv1D	**0.8024**	**0.7278**

Chapter 5

SUMMARY OF THE STUDY AND THE FINDINGS, CONCLUSIONS, IMPLICATIONS, AND RECOMMENDATIONS FOR FUTURE RESEARCH

In this research, our goal was to develop an automated mind wandering system based on eye-gaze dataset. We first replicated the conventional machine learning implementation from a reference paper. We successfully implemented six machine learning models. In order to implement these models, we first developed a data pipeline to transform and preprocess the data. Data transformers included data scaling, data cleaning, outlier treatment, feature selection, and data balancing. We automated these transformation processes for each model. We defined a parameter grid to implement grid-search as there was no prior knowledge of which model would best fit our task. We used 5-fold cross validation technique in order to prevent overfitting in all reported results. Then, we trained six conventional models and generated a mean test AUC-ROC, and mean test accuracy. In addition to AUC-ROC and accuracy, performance of trained models was evaluated using confusion-matrix, AUC-ROC curve, and prediction accuracy on best estimator. Finally, we produced a histogram of AUC-ROC scores from the trained models and saved the trained models into a pickle file (.pkl) for later combined histogram production. We produced a combined histogram from the six trained models. Combined histogram showed that we achieved a highest AUC-ROC score of 0.6595 with a test accuracy of 0.6416 using a Naïve Bayes model. Our first contribution from this research was to beat the reference paper's AUC-ROC score of 0.64 that they achieved using a logistic regression models on the same data.

To further improve the AUC-ROC score and accuracy, we implemented four variations of deep learning models which were referred to as DNN, DNN+Conv1D, DNN+Conv1D+LSTM, DNN+Conv1D+GRU. DNN was a simple implementation of densely connected layers of feed forward neural network models where we varied six optimizers (SGD, RMSprop, Adagrad, Adadelta, Adamax, Adam). Then, we implemented a one dimensional convolutional neural network (Conv1D) layer with DNN. Our logic was that the dataset represents a time-series of subject activities, thus Conv1D layer would better detect the local features from the time-series data. Similarly, LSTM and GRU were added to the DNN as layers since these are known to also help with time-series data prediction. Then, using simple DNN, we achieved an AUC-ROC of 0.6563 and test accuracy of 0.6004. Finally, using Conv1D and DNN, we outperformed the previous AUC-ROC scores and achieved an AUC-ROC score of 0.8024 with mean test accuracy of 0.7278. Therefore, our implementation was able to beat the conventional machine learning models performance considering both our evaluation metrics of AUC-ROC score and accuracy.

Our dataset was collected from another study, thus we had to work with the existing collected information and apply useful cleaning and transformation to try and improve performance. In future a dataset of our own may produce better improvement from existing work. The given dataset was based on a computerized reading task, thus mind wandering in other events may not be feasible for this particular mind wandering detector. Our assumption of time-series dataset needs to be investigated further in future. There are more variations possible of both machine learning and deep learning models. In fact, hyper parameter tuning of deep learning models is itself a huge research area. In addition, there are various possibilities of model architectures yet to be explored, thus future model architecture tuning may produce improved results. Model deployment was not considered, so future strategy may include model deployment in production.

In conclusion, our contribution from this research was three-fold. First, we automated the data pipeline thus it needed less work to implement the model. Second, we successfully beat our reference paper's performance using the same data and data transformation setup. Finally, we further implemented deep learning models and outperformed the best reported conventional machine learning models performance.

The application of this work spans a wide range of fields such as psychological research, driver's attention detection or even student's attention detection in a classroom and so on. We can take advantage of our system beyond the lab settings. For example, there are many available eye tracker devices (Eye Tribe, Tobii Eye X etc.) but for automating mind wandering detection, eye tracking information is a noisy source of information at best, and presents a difficult detection and classification task for a machine learning model. Our model showed that it can perform smoothly if there is missing data or low quality data thus it may be helpful for mind wandering research. In addition, due to our cross-validation strategy, model performance was tested in unseen data which can be thought of as new participants, and it generalized well for these unseen participants. Therefore, our model will be a step forward towards detecting a ubiquitous psychological phenomenon "mind wandering".

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Al-Balushi, S. M., & Al-Harthy, I. S. (2015). Students' mind wandering in macroscopic and submicroscopic textual narrations and its relationship with their reading comprehension. *Chemistry Education Research and Practice*, *16*(3), 680-688.

Aron, A. R. (2011). From reactive to proactive and selective control: developing a richer model for stopping inappropriate responses. *Biological psychiatry*, *69*(12), e55-e68.

Barbuceanu, F., Antonya, C., Duguleana, M., & Rusak, Z. (2011, July). Attentive user interface for interaction within virtual reality environments based on gaze analysis. In *International Conference on Human-Computer Interaction* (pp. 204-213). Springer, Berlin, Heidelberg.

Bellet, T., Bailly-Asuni, B., Mayenobe, P., & Banet, A. (2009). A theoretical and methodological framework for studying and modelling drivers' mental representations. *Safety Science*, *47*(9), 1205-1221.

Bixler, R., & D'Mello, S. (2014, July). Toward fully automated person-independent detection of mind wandering. In *International Conference on User Modeling, Adaptation, and Personalization* (pp. 37-48). Springer, Cham.

Bixler, R., & D'Mello, S. (2016). Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*, *26*(1), 33-68.

Blanchard, N., Bixler, R., Joyce, T., & D'Mello, S. (2014, June). Automated physiological-based detection of mind wandering during learning. In *International Conference on Intelligent Tutoring Systems* (pp. 55-60). Springer, Cham.

Börner, D., Kalz, M., & Specht, M. (2014). Lead me gently: Facilitating knowledge gain through attention-aware ambient learning displays. *Computers & Education*, *78*, 10-19.

Buntz, C. G. (1981). Problems and issues in human service productivity improvement. *Public Productivity Review*, 299-320.

Butko, N. J., Theocharous, G., Philipose, M., & Movellan, J. R. (2011, March). Automated facial affect analysis for one-on-one tutoring applications. In *Face and Gesture 2011* (pp. 382-387). IEEE.

Calvo, R. A., & D'Mello, S. (2010). Affect detection: An interdisciplinary review of models, methods, and their applications. *IEEE Transactions on affective computing*, *1*(1), 18-37.

Carciofo, R., Du, F., Song, N., & Zhang, K. (2014). Mind wandering, sleep quality, affect and chronotype: an exploratory study. *PloS one*, *9*(3).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

Choe, S., Jeon, H. J., & Kwon, O. S. (2018, October). Estimating Attentional State of a Driver: Interacting Effects of Task Demands and Cognitive Capacities. In 2018 *International Conference on Information and Communication Technology Convergence (ICTC) (pp. 999-1004).* IEEE.

Chollet, F. (2017). *Deep learning with Python*. Manning Publications.

Christoff, K., Gordon, A. M., Smallwood, J., Smith, R., & Schooler, J. W. (2009). Experience sampling during fMRI reveals default network and executive system contributions to mind wandering. *Proceedings of the National Academy of Sciences*, *106*(21), 8719-8724.

DelSignore, L. A., Wolbrink, T. A., Zurakowski, D., & Burns, J. P. (2016). Test-enhanced e-learning strategies in Postgraduate Medical Education: A randomized cohort study. *Journal of medical Internet research*, *18*(11), e299.

Dinesh, D., & Bijlani, K. (2016, August). Student analytics for productive teaching/learning. In *2016 International Conference on Information Science (ICIS)* (pp. 97-102). IEEE.

Dixon, P., & Bortolussi, M. (2013). Construction, integration, and mind wandering in reading. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, *67*(1), 1.

D'Mello, S., Cobian, J., & Hunter, M. (2013, July). Automatic gaze-based detection of mind wandering during reading. In *Educational Data Mining 2013*.

D'Mello, S., Kopp, K., Bixler, R. E., & Bosch, N. (2016, May). Attending to attention: Detecting and combating mind wandering during computerized reading. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (pp. 1661-1669).

D'Mello, S. K. (2016). Giving eyesight to the blind: Towards attention-aware AIED. *International Journal of Artificial Intelligence in Education*, *26*(2), 645-659.

Dong, Y., Hu, Z., Uchimura, K., & Murayama, N. (2010). Driver inattention monitoring system for intelligent vehicles: A review. *IEEE transactions on intelligent transportation systems*, *12*(2), 596-614.

Drummond, J., & Litman, D. (2010, June). In the zone: Towards detecting student zoning out using supervised machine learning. In *International Conference on Intelligent Tutoring Systems* (pp. 306-308). Springer, Berlin, Heidelberg.

Eastwood, J. D., Frischen, A., Fenske, M. J., & Smilek, D. (2012). The unengaged mind: Defining boredom in terms of attention. *Perspectives on Psychological Science*, *7*(5), 482-495.

Faber, M., Bixler, R., & D'Mello, S. K. (2018). An automated behavioral measure of mind wandering during computerized reading. *Behavior Research Methods, 50*(1), 134-150.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P. A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, *33*(4), 917-963.

Feng, S., D'Mello, S., & Graesser, A. C. (2013). Mind wandering while reading easy and difficult texts. *Psychonomic bulletin & review*, *20*(3), 586-592.

Foulsham, T., Farley, J., & Kingstone, A. (2013). Mind wandering in sentence reading: Decoupling the link between mind and eye. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, *67*(1), 51.

Frank, D. J., Nara, B., Zavagnin, M., Touron, D. R., & Kane, M. J. (2015). Validating older adults' reports of less mind-wandering: An examination of eye movements and dispositional influences. *Psychology and Aging*, *30*(2), 266.

Franklin, M. S., Broadway, J. M., Mrazek, M. D., Smallwood, J., & Schooler, J. W. (2013). Window to the wandering mind: Pupillometry of spontaneous thought while reading.

Franklin, M. S., Smallwood, J., & Schooler, J. W. (2011). Catching the mind in flight: Using behavioral indices to detect mindless reading in real time. *Psychonomic bulletin & review*, *18*(5), 992-997.

Galéra, C., Orriols, L., M'Bailara, K., Laborey, M., Contrand, B., Ribéreau-Gayon, R., ... & Maury, B. (2012). Mind wandering and driving: responsibility case-control study. *Bmj*, *345*, e8105.

Giambra, L. M. (1989). Task-unrelated thought frequency as a function of age: A laboratory study. *Psychology and aging*, *4*(2), 136.

Gollan, B., Wally, B., & Ferscha, A. (2011). Automatic human attention estimation in an interactive system based on behavior analysis. *Proc. EPIA 2011*.

Gruberger, M., Simon, E. B., Levkovitz, Y., Zangen, A., & Hendler, T. (2011). Towards a neuroscience of mind-wandering. *Frontiers in human neuroscience*, *5*, 56.

Hartmann, M., & Fischer, M. H. (2014). Pupillometry: the eyes shed fresh light on the mind. *Current Biology*, *24*(7), R281-R282.

Hosseini, S., & Guo, X. (2019, September). Deep Convolutional Neural Network for Automated Detection of Mind Wandering using EEG Signals. In *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics* (pp. 314-319).

Hutt, S., Mills, C., White, S., Donnelly, P. J., & D'Mello, S. K. (2016). The Eyes Have It: Gaze-Based Detection of Mind Wandering during Learning with an Intelligent Tutoring System. *International Educational Data Mining Society*.

Jabar, M. A., Sidi, F., Selamat, M. H., Ghani, A. A. A., Ibrahim, H., & Baharom, S. (2011). Capturing tacit knowledge for assessing employees' competency and productivity. *American Journal of Economics and Business Administration*, *3*(2), 358-362.

Jackson, J. D., & Balota, D. A. (2012). Mind-wandering in younger and older adults: Converging evidence from the sustained attention to response task and reading for comprehension. *Psychology and aging*, *27*(1), 106.

Jensen, J. D. (2011). Can worksite nutritional interventions improve productivity and firm profitability? A literature review. *Perspectives in Public Health*, *131*(4), 184-192.

Just, M. A., & Carpenter, P. A. (1980). A theory of reading: From eye fixations to comprehension. *Psychological review*, *87*(4), 329.

Kaber, D. B., Liang, Y., Zhang, Y., Rogers, M. L., & Gangakhedkar, S. (2012). Driver performance effects of simultaneous visual and cognitive distraction and adaptation behavior. *Transportation research part F: traffic psychology and behaviour*, *15*(5), 491-501.

Kam, J. W., Dao, E., Blinn, P., Krigolson, O. E., Boyd, L. A., & Handy, T. C. (2012). Mind wandering and motor control: off-task thinking disrupts the online adjustment of behavior. *Frontiers in human neuroscience*, *6*, 329.

Kam, J. W., Dao, E., Stanciulescu, M., Tildesley, H., & Handy, T. C. (2013). Mind wandering and the adaptive control of attentional resources. *Journal of Cognitive Neuroscience*, *25*(6), 952-960.

Kane, M. J., Brown, L. H., McVay, J. C., Silvia, P. J., Myin-Germeys, I., & Kwapil, T. R. (2007). For whom the mind wanders, and when: An experience-sampling study of working memory and executive control in daily life. *Psychological science*, *18*(7), 614-621.

Kaplan, S., Guvensan, M. A., Yavuz, A. G., & Karalurt, Y. (2015). Driver behavior analysis for safe driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *16*(6), 3017-3032.

Kaur, K., Sutcliffe, A., & Maiden, N. (1998). Improving interaction with virtual environments.

Kersten, M., & Murphy, G. C. (2006, November). Using task context to improve programmer productivity. In *Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering* (pp. 1-11).

Killingsworth, M. A., & Gilbert, D. T. (2010). A wandering mind is an unhappy mind. *Science*, *330*(6006), 932-932.

Kopp, K., D’Mello, S., & Mills, C. (2015). Influencing the occurrence of mind wandering while reading. *Consciousness and cognition*, *34*, 52-62.

Krasich, K., McManus, R., Hutt, S., Faber, M., D'Mello, S. K., & Brockmole, J. R. (2018). Gaze-based signatures of mind wandering during real-world scene processing. *Journal of Experimental Psychology: General*, *147*(8), 1111.

Li, S., & Zhang, X. (2014, August). Implicit human intention inference through gaze cues for people with limited motion ability. In *2014 IEEE International Conference on Mechatronics and Automation* (pp. 257-262). IEEE.

McVay, J. C., & Kane, M. J. (2012). Why does working memory capacity predict variation in reading comprehension? On the influence of mind wandering and executive attention. *Journal of experimental psychology: general*, *141*(2), 302.

Mills, C., & D'Mello, S. (2015). Toward a Real-Time (Day) Dreamcatcher: Sensor-Free Detection of Mind Wandering during Online Reading. *International Educational Data Mining Society*.

Mital, P. K., Smith, T. J., Hill, R. L., & Henderson, J. M. (2011). Clustering of gaze during dynamic scene viewing is predicted by motion. *Cognitive computation*, *3*(1), 5-24.

Moreno, R., & Mayer, R. E. (2002). Verbal redundancy in multimedia learning: When reading helps listening. *Journal of educational psychology*, *94*(1), 156.

Mrazek, M. D., Smallwood, J., & Schooler, J. W. (2012). Mindfulness and mind-wandering: finding convergence through opposing constructs. *Emotion*, *12*(3), 442.

Palinko, O., Kun, A. L., Shyrokov, A., & Heeman, P. (2010, March). Estimating cognitive load using remote eye tracking in a driving simulator. In *Proceedings of the 2010 symposium on eye-tracking research & applications* (pp. 141-144).

Paul, S. M., Mytelka, D. S., Dunwiddie, C. T., Persinger, C. C., Munos, B. H., Lindborg, S. R., & Schacht, A. L. (2010). How to improve R&D productivity: the pharmaceutical industry's grand challenge. *Nature reviews Drug discovery*, *9*(3), 203-214.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research, 12*, 2825-2830.

Pham, P., & Wang, J. (2015, June). AttentiveLearner: improving mobile MOOC learning via implicit heart rate tracking. In *International Conference on Artificial Intelligence in Education* (pp. 367-376). Springer, Cham.

Plimpton, B., Patel, P., & Kvavilashvili, L. (2015). Role of triggers and dysphoria in mind-wandering about past, present and future: A laboratory study. *Consciousness and Cognition*, *33*, 261-276.

Qu, W., Ge, Y., Xiong, Y., Carciofo, R., Zhao, W., & Zhang, K. (2015). The relationship between mind wandering and dangerous driving behavior among Chinese drivers. *Safety Science*, *78*, 41-48.

Randall, J. G., Oswald, F. L., & Beier, M. E. (2014). Mind-wandering, cognition, and performance: A theory-driven meta-analysis of attention regulation. *Psychological bulletin*, *140*(6), 1411.

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*, *124*(3), 372.

Reichle, E. D., Pollatsek, A., Fisher, D. L., & Rayner, K. (1998). Toward a model of eye movement control in reading. *Psychological review*, *105*(1), 125.

Robertson, I. H., Manly, T., Andrade, J., Baddeley, B. T., & Yiend, J. (1997). Oops!': performance correlates of everyday attentional failures in traumatic brain injured and normal subjects. *Neuropsychologia*, *35*(6), 747-758.

Schooler, J. W., Smallwood, J., Christoff, K., Handy, T. C., Reichle, E. D., & Sayette, M. A. (2011). Meta-awareness, perceptual decoupling and the wandering mind. *Trends in cognitive sciences*, *15*(7), 319-326.

Seibert, P. S., & Ellis, H. C. (1991). Irrelevant thoughts, emotional mood states, and cognitive task performance. *Memory & Cognition*, *19*(5), 507-513.

Sharma, S. K., & Gupta, J. N. (2004). Improving workers' productivity and reducing Internet abuse. *Journal of Computer Information Systems*, *44*(2), 74-78.

Sibert, L. E., & Jacob, R. J. (2000, April). Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 281-288).

Smallwood, J., Brown, K. S., Tipper, C., Giesbrecht, B., Franklin, M. S., Mrazek, M. D., ... & Schooler, J. W. (2011). Pupillometric evidence for the decoupling of attention from perceptual input during offline thought. *PloS one*, *6*(3).

Smallwood, J., Davies, J. B., Heim, D., Finnigan, F., Sudberry, M., O'Connor, R., & Obonsawin, M. (2004). Subjective experience and the attentional lapse: Task engagement and disengagement during sustained attention. *Consciousness and cognition*, *13*(4), 657-690.

Smallwood, J., Fishman, D. J., & Schooler, J. W. (2007). Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance. *Psychonomic bulletin & review*, *14*(2), 230-236.

Smallwood, J., & Schooler, J. W. (2006). The restless mind. *Psychological bulletin*, *132*(6), 946.

Smilek, D., Carriere, J. S., & Cheyne, J. A. (2010). Out of mind, out of sight: eye blinking as indicator and embodiment of mind wandering. *Psychological science*, *21*(6), 786-789.

Stawarczyk, D., Majerus, S., Catale, C., & D'Argembeau, A. (2014). Relationships between mind-wandering and attentional control abilities in young adults and adolescents. *Acta psychologica*, *148*, 25-36.

Stewart, A., Bosch, N., Chen, H., Donnelly, P., & D'Mello, S. (2017, June). Face forward: Detecting mind wandering from video during narrative film comprehension. In *International Conference on Artificial Intelligence in Education* (pp. 359-370). Springer, Cham.

Su, M. C., Hsiung, C. Y., & Huang, D. Y. (2006, October). A simple approach to implementing a system for monitoring driver inattention. In *2006 IEEE International Conference on Systems, Man and Cybernetics* (Vol. 1, pp. 429-433). IEEE.

Unsworth, N., & Robison, M. K. (2018). Tracking arousal state and mind wandering with pupillometry. *Cognitive, Affective, & Behavioral Neuroscience*, *18*(4), 638-664.

Uzzaman, S., & Joordens, S. (2011). The eyes know what you are thinking: eye movements as an objective measure of mind wandering. *Consciousness and cognition*, *20*(4), 1882-1886.

Verbruggen, F., Stevens, T., & Chambers, C. D. (2014). Proactive and reactive stopping when distracted: An attentional account. *Journal of Experimental Psychology: Human Perception and Performance*, *40*(4), 1295.

Vergara, R. C., Moënne-Loccoz, C., Ávalos, C., Egaña, J., & Maldonado, P. E. (2019). Finger Temperature: A Psychophysiological Assessment of the Attentional State. *Frontiers in Human Neuroscience*, *13*.

Vinski, M. T., & Watter, S. (2013). Being a grump only makes things worse: a transactional account of acute stress on mind wandering. *Frontiers in psychology*, *4*, 730.

Wellins, R. S. (1991). *Empowered teams: Creating self-directed work groups that improve quality, productivity, and participation*. Jossey-Bass Inc., 350 Sansome Street, San Francisco, CA 94104-1310.

Whitehill, J., Serpell, Z., Lin, Y. C., Foster, A., & Movellan, J. R. (2014). The faces of engagement: Automatic recognition of student engagementfrom facial expressions. *IEEE Transactions on Affective Computing*, *5*(1), 86-98.

Won, A. S., Bailenson, J. N., & Janssen, J. H. (2014). Automatic detection of nonverbal behavior predicts learning in dyadic interactions. *IEEE Transactions on Affective Computing*, *5*(2), 112-125.

Zaletelj, J. (2017, September). Estimation of students' attention in the classroom from kinect features. In *Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis* (pp. 220-224). IEEE.

Zhang, Y., Owechko, Y., & Zhang, J. (2004, October). Driver cognitive workload estimation: A data-driven perspective. In *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No. 04TH8749)* (pp. 642-647). IEEE.

VITA

I am Subroto Singha, completed Master of Science in Computational Sciences at Texas A&M University-Commerce (TAMUC). Before starting master's degree, I completed Bachelor of Science in Electrical and Electronic Engineering from Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

Alongside my graduate degree, I served as a Graduate Research Assistant in the Department of Engineering and Technology at TAMUC. I helped the department implement computer vision technology for detecting fire & drones using Unmanned Aerial Vehicle (UAV). In addition, I helped write proposal for grants. Before this job, I served as a Graduate Teaching Assistant for one year in the department of Computer Science and department of electrical & computer engineering at Boise State University, Idaho, USA.

Along the journey of my academic career, I presented my poster publication in the Pathway Research Symposium'2019 at Texas A&M International University. I was awarded third place in STEM category in Federation Research Symposium'2021 for presenting poster titled "Automated drone detecting using YOLO neural network". I published one IEEE conference paper as a co-author titled "Contextualizing Consensus Protocols in Blockchain: A Short Survey".

My research interests are implementing machine learning and deep learning algorithms in the areas of computer vision, natural language processing and cognitive psychology.

Department of Computer Science and Information Systems
Texas A&M University-Commerce
PO Box 3011
Commerce, Texas 75428
Email: ssingha@leomail.tamuc.edu
subroto.singha4@gmail.com