MASARYK
UNIVERSITY

FACULTY OF INFORMATICS

Recognition of Reading Disorder Based on Eye-Tracking Data

Master's Thesis

BC. ANDREJ ČERNEK

Advisor: doc. RNDr. Jan Sedmidubský, Ph.D.

Department of Machine Learning and Data Processing

Brno, Spring 2023

FACULTAS ARTIS INFORMATICÆ • UNIVERSITAS MASARYKIANA •

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Bc. Andrej Černek

Advisor: doc. RNDr. Jan Sedmidubský, Ph.D.

Acknowledgements

This endeavour would not have been possible without the guidance, insight and feedback from my supervisor, doc. RNDr. Jan Sedmidubský, Ph.D. My gratitude also extends to the Faculty of Arts research team and the pilot experiment participants for enabling our study. Finally, I am thankful to my family, without whose care and understanding throughout the years, I would not be where I am now.

Abstract

Developmental dyslexia, the disorder related to reading, affects an estimated 5–10 % of the population, making it the most common learning disability. It can severely impact one's academic and occupational performance if unmitigated, so early and easy detection is essential. This thesis explores the use of eye-tracking technology and machine-learning techniques as a fast pre-screening solution.

Based on an analysis of the existing research, we propose a variety of established and new data transformations, including extraction of gaze-event-based statistics in the form of vectors or visualisation of multi-dimensional gaze event sequences as fixed-size images, among others. With our Python implementation, enabled by scikit-learn and PyTorch, we evaluate these representations using appropriate neural networks such as Multilayer perceptron, Gated recurrent unit and Convolutional neural network on four tasks read by Czech children.

Our experiments show that neural network models, specifically the neural network trained on the images with augmentations, achieved a mean balanced accuracy of up to 92 %, six p.p. above the best 1-Nearest neighbour baseline. The results are also analysed individually per dataset and participant with limitations in mind. Finally, suggestions for further research areas are outlined.

Keywords

AI, computer vision, deep learning, eye-tracking, dyslexia

Contents

1 Introduction

Dyslexia is among the most common learning disabilities, affecting 5–10 % of the population [1], children and adults alike. The condition can impair the individual's academic and occupational performance [2, 3], which may be minimised by early detection and support [4]. Therefore, providing fast and reliable diagnostic tools at an early age is of great interest.

Eye-tracking technologies enable us to record eye movements during various activities [5], including reading. The differences in reading are well-studied and involve lower reading speeds or a higher chance of rereading already visited sections [6]. Such dissimilarities raise questions about the viability of machine learning in this area.

The existing research focuses primarily on using Support vector machines on statistics extracted from vectors of gaze event sequences which aggregate the eye movement time series [7]. This thesis contributes to verifying the outlined approaches of dyslexia detection on four tasks read by 35 Czech children. In doing so, we aim to explore a broader range of data representations and neural network models.

Aside from testing the well-established vectors of gaze event statistics, we propose representing the eye movements by gaze event sequences or fixed-size images. We classify these data types with appropriate neural networks, Multilayer perceptron for vectors, Gated recurrent unit for sequences and Convolutional neural network for images, achieving a mean accuracy of up to 92 % compared to the best 1-Nearest neighbour baseline accuracy of around 86 %. We also present an ensemble combination of the four tasks for each model, which did not improve average accuracy but slightly stabilised the results.

This thesis starts by describing the dyslexia detection problem in Chapter 2. In the following Chapter 3, we define the dataset and the proposed representations. Afterwards, Chapter 4 briefly explains the compared machine learning methods. Later, Chapter 5 proposes the methodology of the experiments, the validation and the evaluation process. Finally, in Chapter 6, we look at the results and draw conclusions in Chapter 7.

2 Dyslexia detection

The eleventh edition of the International Classification of Diseases (ICD-11), developed by the World Health Organisation, characterises the Developmental learning disorder with impairment in reading by '*significant and persistent difficulties in learning academic skills related to reading, such as word reading accuracy, reading fluency, and reading comprehension. The individual's performance in reading is markedly below what would be expected for chronological age and level of intellectual functioning*' [8]. While there is a debate over whether reading comprehension is an inherent part of dyslexia or a separate issue [9], this is irrelevant to this thesis as it only focuses on reading fluency.

The prevalence estimates of dyslexia are directly affected by the used definition and diagnostic criteria, ranging from just above 5 % to almost 18 % [1], although the majority falls under 10 %. Studies also show higher occurrence in boys, but this is believed to be caused by generally lower referral rates for girls [10]. Finally, studies on orthographically shallow languages (like Czech) suggest lower prevalence rates than studies on deeper ones (e.g. English), which might be due to differing methodologies [11].

ICD-11 notes that dyslexia '*results in significant impairment in the individual's academic or occupational functioning*' [8]. This can limit academic and professional choices, as people with dyslexia may avoid tasks involving reading [2]. It may also damage the person's self-esteem and lead to other mood disorders [3]. Early detection can lead to improvements in confidence and the development of coping strategies [4].

Section 2.1 looks at the typical method of studying reading through eye movements and its ins and outs. The following Section 2.2 details the existing research on machine learning using this method.

2.1 Eye movements

In general, the motivation for analysing eye movements is that cognitive processes, such as perception, memory, language or decision-making, influence them. This eye-mind link, as it is sometimes called, makes eye-tracking suitable for exploring various mental processes and patterns, particularly disorders [5]. As such, attempts to use it for diagnosing neurodevelopment disorders have been tried as well, including ADHD, autism and learning disabilities [12].

The eye, similar to cameras, gathers, focuses and detects light. The image's detail depends on the region of the retina the light falls onto. The highest acuity, along with colour vision, is primarily a product of the fovea centralis, where lies the highest concentration of cones, the colour-sensitive photoreceptors. The fovea captures the centre of the visual field, approximately 1°20′ out of the full 140° [13]. The rest of the field is first caught by parafovea and then the periphery, which are increasingly less detailed [14], as shown in Figure 2.1.

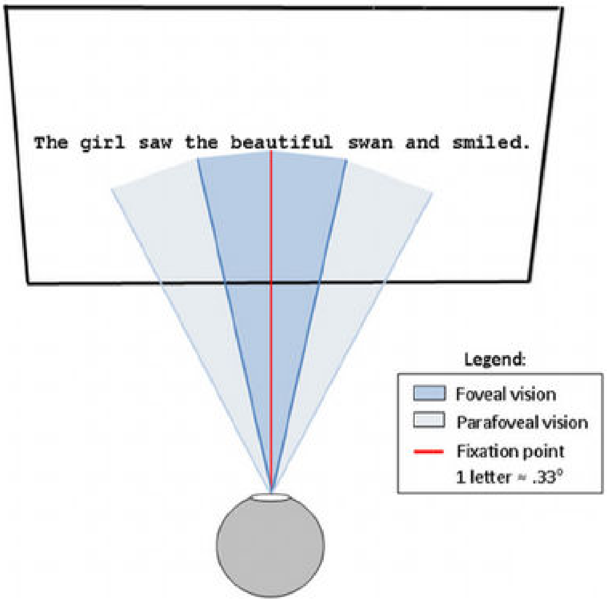


Figure 2.1: Illustration of visual span during reading [15]. The number of letters covered by each vision depends on the distance and font size.

A fixation is an essential type of gaze event during which the eye is fixed on a visual target. Since the fovea is small, the eye cannot take the entirety of the visual field in a single fixation. Instead, the eye makes a sequence of shorter fixations with a slight offset. The specific length depends on various factors, including the nature of the visual stimuli and the task, as well as the individual's skill and mental state. However, the fixations generally last for about 180 to 330 milliseconds [14].

The process of moving between fixations is called a saccade. The duration and velocity again vary based on the task, specifically, the distance travelled. During reading, we want to move the centre of vision by the size of the fovea (a 2° rotation), which usually lasts about 30 milliseconds [14]. For reading tasks, it is also reasonable to differentiate between forward saccades (progressions) and backward saccades (regressions) depending on whether the reader is continuing reading or returning to an earlier part of the text. Moreover, by analogy, this can also be done with fixations, but it is rarely so.

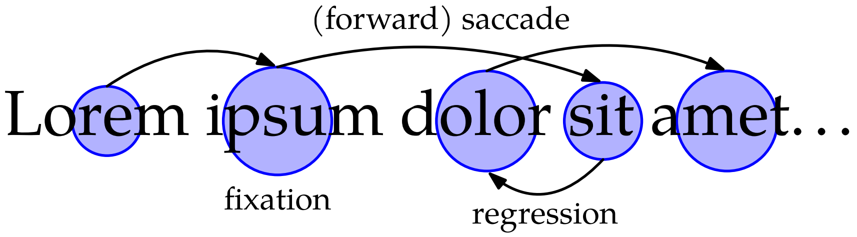


Figure 2.2: Visualisation of fixations, forward and backward saccades.

While saccades and fixations, as seen in Figure 2.2, are the most commonly observed gaze events, some researchers choose to recognise others as well: smooth pursuit (following a moving object), vergence (bringing the eyes together to focus on near objects), glissade (short back and forth saccades before the eyes fully fixate) [16].

The differences in eye movement patterns among people with dyslexia are well-researched, going back to 1958 [6]. The studies generally agree on fixations of dyslexic readers being more frequent and longer, the saccades being shorter, with a higher proportion of regressions. It is also generally accepted that these abnormalities result from the difficulties the person has with reading rather than being the cause [17].

2.1.1 Eye-tracking

From a technical point of view, tracking eye movements involves shining some light source, usually an infrared light, into the eye with a given frequency. The eye-tracking software then identifies the pupil's centre from a cornea's reflection. During the experiments, calibrations, during which users look at a series of pre-determined locations on the screen, are regularly done to establish a baseline from which the relative gaze position is calculated [5].

Stabilisation via a chin rest may be used to maximise accuracy, and a higher frequency device may increase precision. Nonetheless, the real-world outcomes will vary widely due to circumstances beyond the eye-tracker choice, like the experiment setup and procedure, as well as user-specific behaviour and physiology. For this reason, data cleaning ought to be performed during the pre-processing of the data [5].

The raw output of the devices consists primarily of X and Y coordinate time series for each eye (in pixels), although additional measures such as pupil size can also be seen. These series are then aggregated into gaze events, fixations and saccades, and blinks and lost data. For this, a broad selection of commercial and open-source software packages is available, each offering different algorithms and parameters. A noteworthy ability is the recognition of the Areas of interest (AOI), that is, the matching of the fixations to the visual stimuli [5].

Finally, most circumstances do not necessitate analysis of both eyes. We may average or subtract the coordinates in such cases or only consider the eye with better data coverage.

2.2 Related work

The earliest cases of dyslexia detection models came up in the early 2010s [7, 12]. Al-Edaily et al. [18], for example, used simple thresholds on simple statistics such as total fixation duration, mean fixation duration or fixation count. The accuracy of individual features on 14 children (7 with and 7 without dyslexia) aged 10–12 reading Arabic script reached 70 % and above.

The first instance of machine learning usage for eye-tracking analysis was a 2015 paper by Rello et al. [19]. The paper achieved an accuracy of around 80 % on 12 readings from 97 Spanish speakers (48 of whom were dyslexic) aged 11–54 using Support vector machines (SVM). The authors established the method of aggregating gaze events globally (on the entire text) into statistics, stating reading time (time spent in the AOI) and mean fixation duration as the most valuable features while also noting the importance of considering age.

The dataset of 185 Swedish children aged 9–10, 97 classified as high-risk, is one of the largest and most well-studied. Firstly, Nilsson Benfatto et al. [20] achieved an accuracy of 95.6 % by using SVM-RFE to select 48 of their original 168 features relating to fixation duration and saccade position. Secondly, a Portuguese paper [21] explored a range of methods on wavelet transformed data claiming to achieve 97.3 % accuracy using Random Forest. On the other hand, Jothi Prabha et al. [22] ended up ranking the Hybrid SVM-PSO model [23] as best at 96.6 % accuracy, preferring features based on fixation and saccade counts and lengths. Finally, Nerušil et al. [24] tried a Convolutional neural network (CNN) on pre-processed time series resulting in 96.6 %.

Smyrnakis et al. [25] proposed a custom score made from mean fixation time, saccade length quartiles, the number of short regressions, fixation count, skipped word count and other event statistics. The score differentiated 69 Greek children, 32 dyslexic, aged 9–13, with an accuracy of 94.2 % on one text and 87.9 % on another. Asvestopoulou et al. [26] then expanded on this work by fitting SVM, *k*-means and Naive Bayes models, with the SVM model coming on top with 97.1 % and 89.39 % accuracies, respectively. The used feature selection, LASSO, also identified features with the most predictive power: the mean saccade length, the number of short forward saccades, and, to a lesser degree, the median saccade length and the number of multiply fixated words.

Another research done by Szalma et al. [27] tested SVM on a group of 48 Hungarian young adults (20–28 y.o.) with accuracies ranging from 70 % to 90 % depending on the level of overfitting. Notable features included an Interquartile range (IRQ) of forward saccade count divided by word, median saccade amplitude or median regression amplitude. The follow-up papers [28, 29] investigated the effect of spacing level on detection, favouring larger spacings. By looking at a broader range of feature selection methods, they also reported the median of progressive and all saccade amplitudes, a median of fixation duration and an IRQ of forward glissade duration as the most important.

Vajs et al. [30, 31] tried detection based on gaze event statistics and raw time series data evaluated on 30 Serbian children (7—13), half dyslexic. In the former case, all four models (Linear Regression, SVM, *k*-NN and Random Forest) achieved an accuracy of around 90 %, with the highest relevance given to the unique event features presented by the authors. In the latter case, they trained CNN on image visualisations of the gaze, reaching accuracies up to 87 % depending on the selected hyper-parameters.

Even among the less studied datasets, SVM tends to dominate the research. Gran Ekstrand et al. [32] attained an average accuracy of 87.9 % on the 2726 Swedish participants aged around 9. While El Hmimdi et al. [33] reached an accuracy of up to 80 % with SVM on 87 French students (12–18), 46 dyslexic, Linear Regression slightly outperformed it at 81.25 %, and both significantly outperformed the MLP model. Raatikainen et al. [34] similarly achieved an accuracy of up to 89.8 % on 165 Finish youngsters (around 12 y.o.), 30 of whom were dyslexic, outmatching the Random Forest by three percentual points.

Neural networks also started to be explored in recent years. One of the earliest attempts was a Master's thesis by Lustig [35] comparing SVM to MLP and LSTM on 18 participants, half dyslexic. The feature-based SVM and MLP models came on top with 83 % accuracy versus the *X* coordinate sequence-based LSTM with 78 %. Similarly, Haller et al. [36] tested the SVM, LSTM and CNN models on 62 Mandarin Chinese children, 33 with dyslexia, reaching 90 % accuracy with all of them. Here, both neural networks used word-by-word sequences of gaze event statistics rather than the events or raw coordinate pairs.

Some studies looked into identifying reading skills more broadly. Zhan et al. [37] tried to assess the reading abilities of 74 Chinese students aged 17–21, among which 38 were deemed to have lower proficiency. The custom model with 95.09 % accuracy helped them determine fixation and saccade rates as most important, followed by regression rate, count, and pupil diameter. In contrast, Lou et al. [38] chose to predict the literacy skills of 61 Chinese undergrad students in their twenties (the class breakdown was not stated). The SVM model with the highest accuracy of 80.03 % utilised the duration of different types of fixations across varying text sections.

While differing methodologies, small and sometimes even imbalanced datasets, and different languages researched make it harder to draw absolute conclusions, existing literature does confirm that

Table 2.1: Overview of ML methods and data types used in existing research: The cells include the number of studies researching the given model on a given data type, except the totals since some studies research multiple approaches.

Algorithm	Global stat.	Per AOI stat.	Time series	Total
SVM	**10**	2	1	**13**
Random Forest	2	1	1	4
LR	3			3
k-NN	2		1	3
CNN		1	2	3
Naive-Bayes	1		1	2
MLP	1		1	2
k-Means	1			1
Decission Tree			1	1
Gauss. Process			1	1
adaBoost			1	1
RNN			1	1
LSTM		1		1
Custom model	2			2
Total	**22**	5	11	

machine learning is a helpful tool for dyslexia detection. SVM stands out as the state-of-the-art method (see Table 2.1), achieving an accuracy of up to 90 % on gaze event statistics. The literature varies in which specific statistics are used, only preferring the ones based on fixation duration and saccade length and noting the importance of progression differentiation. At the same time, alternative neural network approaches also show promising results, potentially superseding SVM, but more research is needed.

3 Data and preprocessing

This chapter describes the data and our approach to representing them. Section 3.1 details the format of the data as received. Section 3.2 continues with the resulting data types we explore in our work.

3.1 Collection

The data for this thesis was provided by a research team from the Faculty of Arts of Masaryk University as part of a pilot experiment on the differences between dyslexic and non-dyslexic readers. This experiment is part of a larger project supported by the Technology Agency of the Czech Republic[1] and approved by the Research Ethics Committee of Masaryk University.

The sample consists of 35 4th-grade children (i.e. aged 9–10), of which 13 were diagnosed with dyslexia by the Pedagogicko-psychologická poradna Brno. The procedure took approximately 30 minutes and involved the participants fixing their heads with a chinrest located 55–60 cm in front of a 22″ LCD monitor (1680 × 1050 resolution) that displayed the stimuli.

The eye-tracking was enabled by commercial solutions from SensoMotoric Instruments:

- Experiment Center 3.7.69, software for the experiment design;
- REDm 250, an eye-tracking device with a 250 Hz sampling rate;
- iViewX to control the device;
- BeGaze 3.7 for the velocity-based gaze event detection and export of the data as described in Subsection 3.1.2.

3.1.1 Tasks

The pilot experiment utilised four verbal tasks from the standardised battery for diagnosing dyslexia in the Czech Republic [39] and five

[1] See `https://starfos.tacr.cz/en/project/TL05000177`.

additional non-verbal. For purposes of machine learning, only three types of reading tasks were considered:

Grid reading of syllables and short words embedded in a 7×7 grid without a time limit, as shown in Figure 3.1;

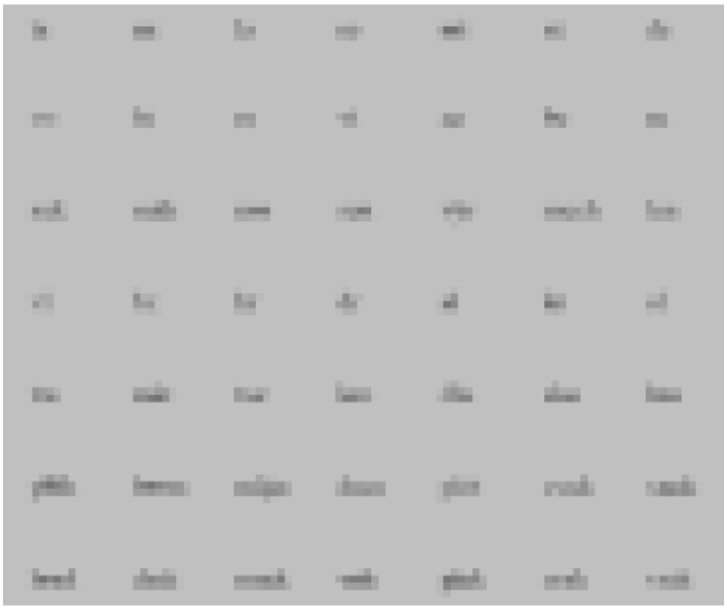


Figure 3.1: Illustration of the *grid* task, adapted from the battery (5×5 initially) [39], obscured at the request of the authors.

Text reading of meaningful continuous texts limited to 2 minutes, as illustrated in Figure 3.2. This task consisted of two trials:

- **Hard text** age appropriate;
- **Easy text** aimed at younger pupils;

Pseudo-text reading of a meaningless continuous text limited to 2 minutes. The structure of the text matched the Czech language, but the words were fictitious.

The remaining tasks were structured differently, so they were deemed out of this thesis's scope.

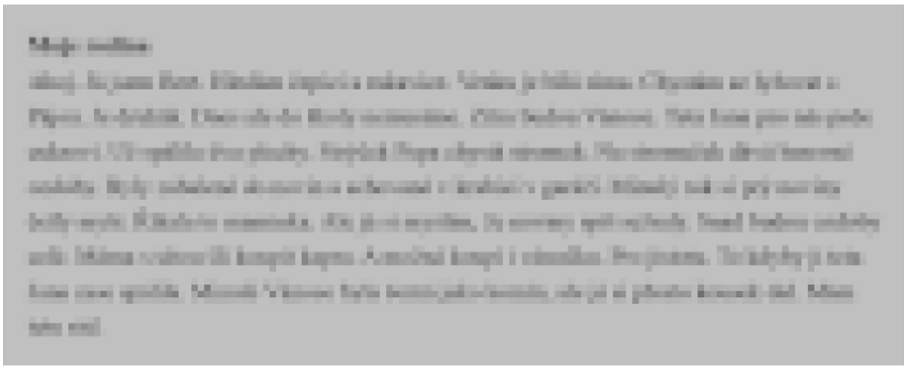


Figure 3.2: Illustration of the *easy text* task [39], obscured at the request of the authors.

3.1.2 Events data format

The BeGaze software transforms the raw time series the eye-tracking device generates into gaze event data (as described in Section 2.1). These are made available in files individually for each participant, providing the data from an eye with better data coverage. Each file line represents a gaze event whose type is defined in the *Category* column: one of *Fixation*, *Saccade* or *Blink*. Explanation of the columns used in this thesis follows:

General trial attributes

- *Participant*: The ID of the participant.
- *Trial Length* (in ms): The total time length of the task for a given participant.

Additionally, the classification of the participants was located in another file.

Common gaze event attributes

All event types shared a set of common columns:

- *Event Start Trial Time* (in ms): The offset from the trial's start until the event's beginning;

- *Event End Trial Time* (in ms): The offset from the trial's start until the event's end;
- *Event Duration* (in ms): This should be the subtraction of the values above.

Fixation attributes

Fixations are aggregated from several raw coordinates, and so aside from the estimated position of the fixation's centre dispersion of the raw values is also included:

- *Fixation Position* X and Y (in px);
- *Fixation Dispersion* X and Y (in px);
- *AOI name*: All tasks, except for the *hard text*, had predefined Areas of interest (AOIs): The grid had an AOI for each cell (49), while text-based tasks had one for each line (9–12 including the heading). The AOIs usually covered multiple fixations, and fixations that were too far from the content were grouped in the *White Space* AOI.

Saccade attributes

- *Saccade Start Position* X and Y (in px);
- *Saccade End Position* X and Y (in px);
- *Saccade Amplitude* (in °): The eye angle between the start and end of the saccade.

Further, *Saccade Length* (in px) can be calculated as a Euclidean distance between the starting and ending positions.

3.2 Feature extraction

Processing raw time series is significantly resource-consuming, which leads us to search for ways to simplify the data. Gaze events are an improvement but, at the same time, high-dimensional. One way to further trivialise this work is to aggregate the gaze events into statistics at different levels of detail. Alternatively, we can focus only on one gaze event type, like fixations, and analyse their sequences.

3.2.1 Custom AOI definitions

Since AOI names for all trials were not made available to us, a need for custom labels arose. We can use a moving grid to account for the potential differences in coordinates between participants.

The idea is to find AOI positions that maximise the number of fixations in them and lump the fixations outside together. To make the search easier, we can split it into finding the bounds on X-axis and then the Y-axis.

For the first task, we can start the search from the bottom right corner, move the bounds to the left corner, and take the first combination of borders that cover the maximum of fixations. Then we can divide the inner area regularly into forty-nine (7×7) cells.

We must also consider that not all participants have finished the entire text in time for the remaining trials. Searching for X-axis bound stays the same, but we search for the lower Y-axis from the top down. For each explored bound, we find the lowest estimated row count that maximises the fixation count. Similarly, we select the lowest maximising lower bound at the end, like in Figure 3.3.

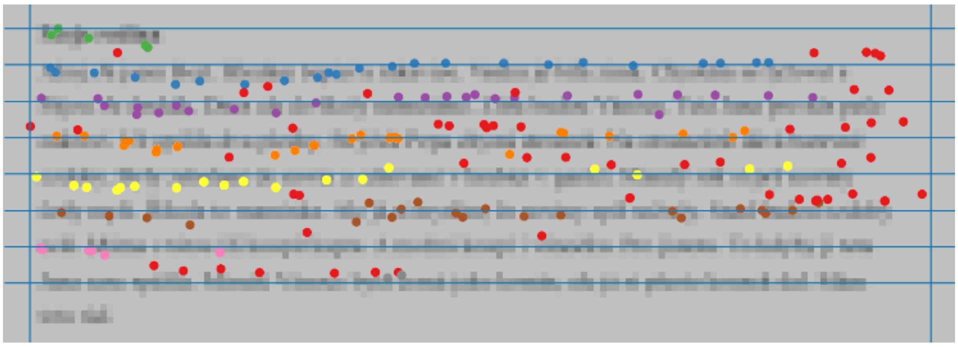


Figure 3.3: The grid for a single participant on *easy text* with fixations coloured by the original AOI labels (red signifies white space).

As a result, we have both the rough AOI labels and estimated row counts. Additionally, we can use these AOIs to categorise the saccades.

If the saccade ends in a region later than it started, we can assume it is a progression and, in the opposite case, a regression. If the saccade does not change the region and the length is small, we say it is a refocus. Finally, in the text-based tasks, if the change in the X coordinate is more considerable (larger than 25 pixels), we categorise it based on the direction.

This algorithm determines only a rough estimate, e.g. it does not reckon with fixations that fall into white space between words. If available, it should be replaced with the native solution from the gaze event extraction software.

3.2.2 Gaze event statistics

Aggregating gaze event attributes is usually done globally, on the whole trial, but we can also examine them on a more granular level: e.g. per AOI or per time window.

Global statistics

The global level statistics are fixed-length vectors of features, each of which is a real number statistic or some other descriptive attribute. These statistics should ideally have different class distributions, like in Figure 3.4.

The backbones of the global statistics are the counts of each gaze event type (fixation, saccade, blink). In the case of saccades, we also recognise progression and regression counts and their proportions of the total and the saccade-to-fixation ratio.

For saccades, we can aggregate their time, length and amplitude using mean, median and *Interquartile range* (IQR), the difference between the 75th and 25th quartile. On top of it, the totals of the time and length can also be considered.

Similarly, we can use the same aggregations for fixations: on their time and X, Y coordinate dispersions. Here, only the time's sum has meaning. We also apply the same statistics to the fixations excluding the non-AOI (i.e. white space) ones: the so-called *visits*. Aside from the total visit count, we are also interested in the mean, median and IQR of the visit counts per AOI.

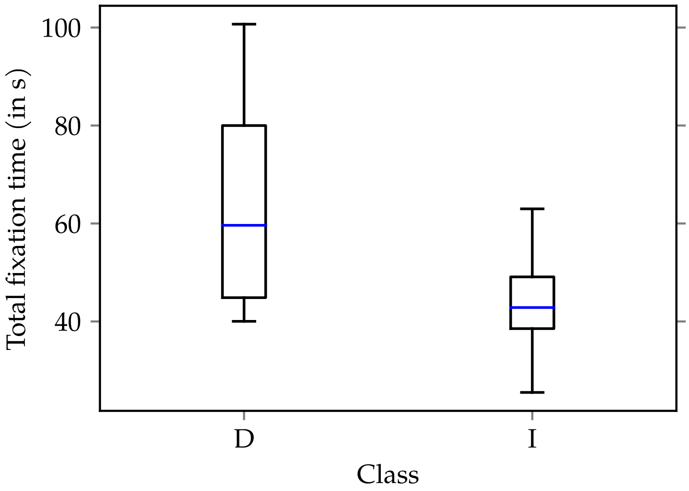


Figure 3.4: Comparison of the fixation time sum between dyslexics (D) and non-dyslexics (I) on *grid*.

Finally, we shall look at the blink rate: the number of blinks per second. Besides, the trial length and estimated row count (as described in the previous subsection) add up to 46 features (45 in the *grid*'s case).

If we exclude AOI-related statistics, trial length, row count, blink statistics and saccade-to-fixation ratio, we get a reduced set of 27 features relevant to smaller parts of the tasks.

Per AOI statistics

Similarly, we can investigate the same statistics per AOI. For the sake of simplicity, we have explored a single statistic and eliminated any global attribute like the trial length. While any statistic from the reduced feature set should work, the sum of fixation time is the most promising based on existing research.

Per time window statistics

Another option is to split the trial into time segments of the same length (e.g. 5 s) and do the aggregations. We need to remember that some events could span multiple segments, in which case we put them into the first relevant segment for convenience. This results in time series of 27 features.

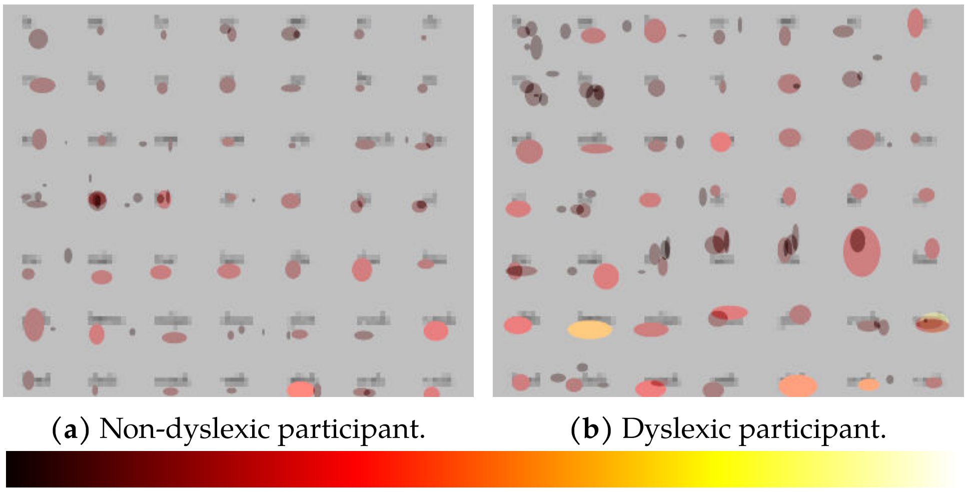
3.2.3 Fixation sequences

We define fixation sequences as sequences of tuples made from all the fixation-related ordinal features (X, Y positions, duration and X, Y dispersions). Alternatively, we can view them as matrices of size $5 \times \mathit{fixation\ count}$. AOI labels were omitted since such categorical features necessitate specific encoding and are no more than a different position representation.

3.2.4 Fixation visualisations

In addition to using raw fixation sequences, we may also represent them as images. Among the various methods available, we have chosen an approach that preserves all five explicit features, even though it does not preserve the order of fixations. Expressly, we represent fixations as ellipses where the fixation coordinates determine their position on the canvas, and the dispersion defines their width and height. The duration of each fixation is represented by its colour, as seen in Figure 3.5.

To map fixation duration to colour, we use Matplotlib's *hot* colourmap, which ranges from black (for minimum values) through red and yellow to white (for maximum values). It is important to note that this normalisation and the choice of canvas boundaries were performed before splitting the datasets, which may have introduced some data leakage.



(a) Non-dyslexic participant.

(b) Dyslexic participant.

(c) The fixation duration colourmap [a].

[a] See `https://matplotlib.org/stable/tutorials/colors/colormaps.html`.

Figure 3.5: Illustrations of the *grid* reading task overlaid by the rescaled fixture visualisations.

4 Machine learning methods

Detecting dyslexia can be viewed as an instance of binary classification. Machine learning, the process of improving a program's performance using data [40], can also be applied to the classification problem. Specifically, providing pre-labelled data to the learning process is called supervised learning. The two methods explored in this thesis are k-Nearest neighbours (Section 4.1) and Neural networks (Section 4.2).

4.1 k-Nearest neighbours

The k-Nearest neighbours (k-NN) is a distance-based supervised method. The idea with this group of models is that the instances of the same class are typically close to each other. Formally, we can use a distance metric $dist\colon X \times X \to \mathbb{R}$, where X is the set of data examples. For all $x, y, z \in X$, the following axioms have to hold [41]:

- Reflexivity: $dist(x, x) = 0$;
- Positivity: $x \neq y \Leftrightarrow dist(x, y) > 0$;
- Symmetry: $dist(x, y) = dist(y, x)$;
- Triangle inequality: $dist(x, y) \leq dist(x, z) + dist(z, y)$.

Suppose we have a training set of n pairs $T = \{(\vec{x_1}, c_1), \ldots, (\vec{x_n}, c_n)\}$, where $\vec{x_i} \in T_x$ is typically a d-dimensional vector ($T_x \subseteq \mathbb{R}^d$), and $c_i \in T_c$ is the category label. For distance metric $dist$ and positive integer k, we can define the set of k nearest neighbours $N_k(\vec{x}) \subseteq T_x$, $|N_k(\vec{x})| = k, \forall \vec{x'} \in N_k(\vec{x}), \forall \vec{x''} \in T_x \setminus N_k(\vec{x})$: $dist(\vec{x}, \vec{x'}) \leq dist(\vec{x}, \vec{x''})$. In that case, we can label $\vec{x}$ with the most common label of the points in $N_k(\vec{x})$ [42].

The most straightforward option for the distance metric is the Euclidean distance, which is the distance between two points in Euclidean space, as seen in Equation 4.1.

$$dist(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} \tag{4.1}$$

4.1.1 Dynamic time warping

In the case of time series or sequential data, we do not have individual inputs in the form of d-dimensional vectors but variable-length sequences of d-dimensional points. Since regular Euclidean distance or its generalisations cannot handle such inputs, we have to look for other metrics, like Dynamic time warping (DTW) [43].

Let us consider two sequences $x = \langle x_1, \ldots, x_m \rangle$ and $y = \langle y_1, \ldots, y_n \rangle$ in the d-dimensional space ($x_i, y_j \in \mathbb{R}^d$) with lengths m and n, respectively. DTW then forms an optimisation problem:

$$DTW(x, y) = \min_{\pi} \sum_{(i,j) \in \pi} dist(x_i, y_j), \tag{4.2}$$

where $dist$ is a distance function (e.g. Euclidean distance) and $\pi = \langle \pi_1, \ldots, \pi_o \rangle$ is an alignment between the two sequences, i.e. it satisfies:

- It is a list of index pairs $\pi_k = (i_k, j_k)$ for $1 \leq i_k \leq m$ and $1 \leq j_k \leq n$;
- It starts at the beginning of the sequences $\pi_1 = (1, 1)$ and ends at their end $\pi_o = (m, n)$;
- It walks through all points in order, i.e. for all $\pi_k = (i_k, j_k)$ and $\pi_{k-1} = (i_{k-1}, j_{k-1})$ where $1 < k \leq o$, $i_{k-1} \leq i_k \leq i_{k-1} + 1$ and $j_{k-1} \leq j_k \leq j_{k-1} + 1$.

Notice that DTW is not technically a proper distance metric, as it does not satisfy the triangle inequality, which is, however, not in conflict with the k-NN algorithm.

4.1.2 Min-max scaling

When using Euclidean distance in machine learning models, we need to consider that it uses absolute distances of features. Therefore it can overfocus on features with large ranges. The simplest solution is to rescale all features into a fixed range [44], such as $[0, 1]$:

$$x'_{ij} = \frac{x_{ij} - \min_k(x_{kj})}{\max_k(x_{kj}) - \min_k(x_{kj})}, \tag{4.3}$$

where i is the sample index and j is the feature index. The extremes are always calculated per feature, even for sequential data, where we calculate the extremes across the entire sequence.

4.2 Neural networks

Artificial neural networks are machine learning techniques inspired by the biological neural networks that constitute brains. They are collections of smaller units, called (artificial) neurons, connected to pass a signal in the form of real numbers. Individual variants differ in either the architecture or the unit design.

4.2.1 Multilayer perceptron

The most straightforward architecture is the Feedforward neural network [45], where the neurons are organised into layers, usually fully connected to the neurons from the previous layer. The input layer consists of n virtual neurons, where n is the input length, that only passes the values into the next layer. The last layer is called the output layer, and its number of neurons determines the output size. We speak of Multilayer perceptron (MLP) if more (hidden) layers exist between the input and output, as shown in Figure 4.1.

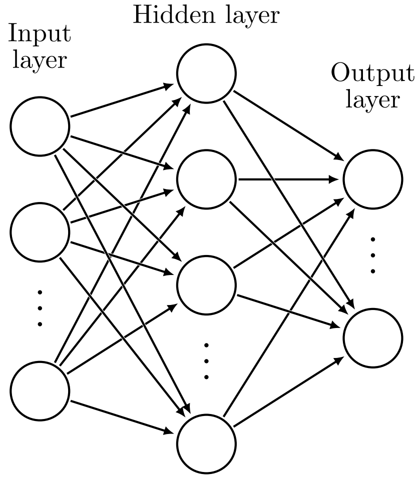


Figure 4.1: Illustration of a fully connected multilayer perceptron with a single hidden layer.

In the basic neuron design, we see the vector of inputs $(x_1, \ldots, x_n) \in \mathbb{R}^n$ combined with the weights $(w_1, \ldots, w_n) \in \mathbb{R}^n$ into the so-called inner potential ξ, almost always by summing the per-element products with the bias $w_o \in \mathbb{R}$, a non-input weight:

$$\xi = w_0 + \sum_{i=1}^{n} w_i x_i. \tag{4.4}$$

The activation function $\sigma\colon \mathbb{R} \to \mathbb{R}$ finally transforms the inner potential into output $y = \sigma(\xi)$.

The activation function can be an arbitrary differentiable function, but for the hidden layers, we generally use non-linear activations like the Rectified linear unit (ReLU):

$$ReLU(\xi) = \max(0, \xi). \tag{4.5}$$

The most commonly used output activation for binary classification is the (logistic) sigmoid:

$$sigmoid(\xi) = \frac{1}{1 + e^{-\xi}}. \tag{4.6}$$

The range of this function is between 0 and 1, meaning we can interpret it as class probability and use a single output neuron.

Alternatively, for the classification of n classes, we use softmax activation on n neurons:

$$softmax(\xi_i) = \frac{e^{\xi_i}}{\sum_{j=1}^{n} \xi_j}, \tag{4.7}$$

where ξ_i is the i-th output inner potential. By scaling the sum of the outputs to 1, softmax again allows for the probability interpretation.

4.2.2 Gated recurrent unit

The typical neural network only accepts vectors of fixed length. Recurrent neural networks (RNNs) [45] implement a form of memory to support sequential data. The network activity can be split into time steps, and in each step, the recurrent units process not only the input in that step but also the output from the previous one.

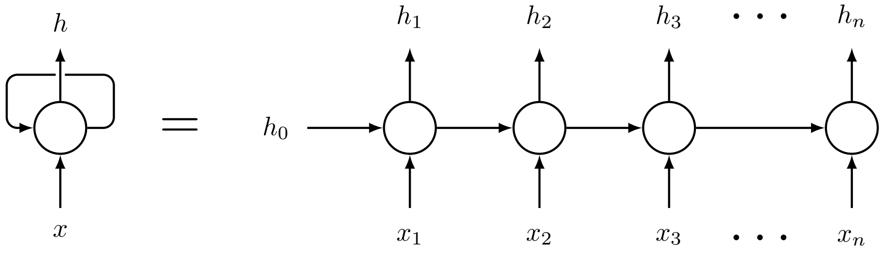


Figure 4.2: Illustration of an unrolling of a recurrent unit.

Just like in Figure 4.2, for a given input sequence $x = \langle x_1, \ldots, x_n \rangle$, $x_t \in \mathbb{R}^d$, the single hidden recurrent neuron outputs a hidden sequence $h = \langle h_1, \dots, h_n \rangle$, $h_t \in \mathbb{R}$:

$$\begin{aligned} h_t &= \sigma(v_0 + \sum_{i=1}^{n} v_i h_{(t-1)i} + w_0 + \sum_{i=1}^{n} w_i x_{ti}) \\ &= \sigma(v_0 + V h_{(t-1)} + w_0 + W x_t), \end{aligned} \tag{4.8}$$

where h_0 would be the initial hidden state (e.g. zeroes). The activation function commonly used in RNNs is the hyperbolic tangent $\sigma = \tanh$. Since these recurrent layers produce sequences of hidden states, we can take the last state h_n as an output for classification purposes.

Unfortunately, this basic architecture is plagued by the exploding/vanishing gradient when learned with gradient-based methods. Several improvements were proposed, such as the Long short-term memory (LSTM) or simpler Gated recurrent unit (GRU) [45]. Unlike RNN, where the hidden state serves as memory, LSTM uses an additional memory channel, and a system of gates that control the information flow more granularly. GRU retains the gating system but without the second memory channel, which speeds up the processing, but potentially leads to worse outcomes.

The structure of GRUs allows the cell to forget its state before updating it:

$$\begin{aligned} r_t &= sigmoid(v_{0,r} + V_r h_{(t-1)} + w_{0,r} + W_r x_t) \\ z_t &= sigmoid(v_{0,z} + V_z h_{(t-1)} + w_{0,z} + W_z x_t) \\ n_t &= \tanh(v_{0,n} + V_n h_{(t-1)} + r_t * (w_{0,n} + W_n x_t)) \\ h_t &= (1 - z_t) * n_t + z_t * h_{t-1}, \end{aligned} \tag{4.9}$$

where $*$ is the Hadamard product and r_t, z_t and n_t are the reset, update and new gates, respectively.

4.2.3 Convolutional neural network

For the processing of high-dimensional data (like images), we often use Convolutional neural networks (CNNs) [45]. Unlike MLPs, layers of which are usually fully-connected, the CNNs include so-called convolutional and pooling layers (Figure 4.3), which utilise weight-sharing to reduce the total weights count and, by proxy, the computational costs.

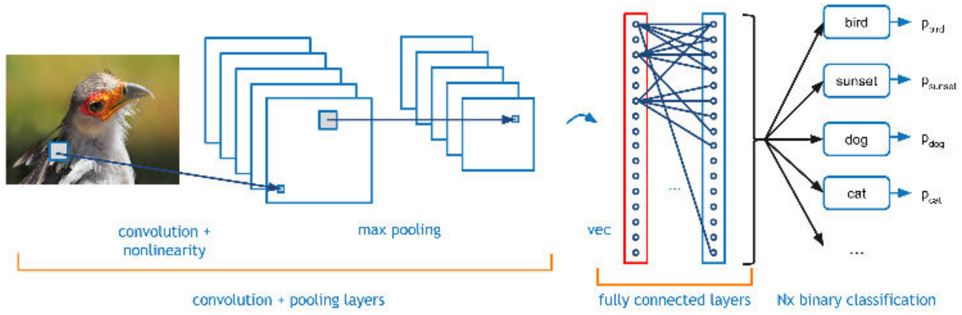


Figure 4.3: Illustration of the typical CNN for classification of images [46].

In each convolutional layer, we have feature maps, groups of neurons that share their weights. The neurons in a given feature map are connected to a subset of the neurons from the previous layer called a receptive field with the same (kernel) size. These fields can intersect, and the distance between them is called stride length. For example, the receptive fields are often squares with some small stride in image processing, and each feature map detects some specific element in the previous layer (e.g. shape or colour).

The pooling layer follows a convolutional layer (or a sequence thereof) and reduces the number of neurons. Similarly to convolutional layers, pooling, split into several channels to match the feature maps, connects its neurons to small clusters from the previous layer. Unlike in convolutional and dense (fully-connected) layers, the neurons in the pooling layer combine the outputs from their receptive layer without any weights:

- **Max pooling**: The neuron calculates the maximum;
- **Average pooling**: The neuron calculates the average.

4.2.4 Learning

The previous subsections detail the inner working of the neuron networks, but the learning itself can be described as an optimisation problem [45]. Specifically, we are minimising the error, i.e. the output of the so-called loss function. The typical loss function for classification (with softmax output) is cross-entropy:

$$E = \frac{1}{n} \sum_{i=1}^{n} (-\log y_{i,c_i}), \tag{4.10}$$

where n is the number of samples, c_i is the category of the i-th sample, and so y_{i,c_i} is the outputted probability of the correct class.

In the case of binary classification (with sigmoid output, i.e. $c_i \in \{0,1\}$), the cross-entropy simplifies to:

$$E = \frac{1}{n} \sum_{i=1}^{n} -(c_i \log y_i + (1 - c_i) \log(1 - y_i)). \tag{4.11}$$

The optimisation is commonly done by the gradient descent algorithms, which adjust the weights with the negative gradient of the error to find a local minimum. The most straightforward iteration implementation is the Stochastic gradient descent:

$$w^{(t)} = w^{(t-1)} - \eta \nabla E, \tag{4.12}$$

where η is the learning rate. SGD is usually not applied to all samples at once, but in each iteration (epoch), we update the weights several times to disjoint subsets (batches). The gradient itself ∇E is calculated using backpropagation.

Many extensions were proposed to deal with various issues the SGD faces, like the exploding/vanishing gradient. These include Momentum, AdaGrad, RMSProp or Adam (Adaptive moment estimation) [45].

4.2.5 Input standardisation

The expected struggle in neural networks is some part of the network increasing and decreasing out of control (e.g. the exploding/vanishing gradient). For this reason, keeping parts of the network around zero is best. Inside the network, this is achieved by an appropriate weight initialisation. Outside, we can standardise the inputs [45]:

$$x'_{ij} = \frac{x_{ij} - \overline{x_j}}{\sigma_j}, \tag{4.13}$$

where i is the sample index, j is the feature index, $\overline{x_j}$ is the average of the feature and σ_j is its standard deviation. The statistics are always calculated per feature, even for sequential data, where we calculate the statistics across the entire sequence.

4.3 Ensemble

The idea behind ensemble learning is that combining multiple learning algorithms could lead to a better outcome. Notably, neural networks can also be viewed as ensembles of neurons and layers.

The most elementary type of ensemble is majority voting [47]. In the case of binary classification and an odd number of classifiers, we select the class that the majority of them predict. For an even number of classifiers and a tie scenario, we either have to give some larger weight or choose the class randomly.

5 Experiment setup

This thesis's central core is examining different machine-learning approaches to dyslexia detection rather than implementing a specific model. As such, great care has to be taken when designing the experiments and evaluating the results. On the other hand, we can exploit Python's rich data-science ecosystem for conducting experiments.

As such, the technologies used are described in Section 5.1, while the project structure in Appendix B. Section 5.2 introduces the validation and evaluation process, including the model details. Additionally, Section 5.3 introduces the data augmentation strategy.

5.1 Machine learning in Python

The backbone of data science in Python is *NumPy*, which offers basic numerical data structures far exceeding the standard library in functionality and performance. This library is then built upon by *SciPy*, providing a broad range of algorithms, *Pandas* for data analysis and manipulation and *Matplotlib* powering the visualisations. The presentation of the code and results can be boosted further with *Jupyter* notebooks.

The most common libraries that enable machine learning are *scikit-learn* [48], which covers the basic models and evaluation, and *PyTorch*, which implements an in-depth neural network interface. When it comes to time series and sequences, there are fewer established packages. However, there are smaller projects like *tsaug*, for time series augmentation, or *tslearn* [49], for time series models utilising scikit API.

5.2 Evaluation

The central choices we have to make during the design of the validation and evaluation are selecting the validation method, the evaluation metric and the hyper-tuning technique. The specific circumstances that drive our decision process are the dataset size and its unbalanced nature.

5.2.1 k-fold cross-validation

One of the most common pitfalls of machine learning is overfitting: Training and evaluating the model on the same data may lead to unfounded confidence in its abilities regardless of the ability to process unseen instances. The solution is to split the data into two subsets: the training and testing sets.

However, this practice needs to be revised in the case of small datasets, as a small training set cannot cover the total variability of the real world. To better represent the available material, we can split the set into k folds and iteratively use one fold for testing and the rest for training [50]. Further, we can repeat this procedure n times with differently shuffled splits, which yields $n \times k$ results. In this thesis, we use ten repeats ($n = 10$) of 5-fold cross-validation ($k = 5$), i.e. we evaluate each model 50 times.

The class imbalance requires another revision. If we used standard shuffled k-fold cross-validation with imbalanced data, we could end up with folds with a testing set of a single class. To avoid this, we can stratify the process by separately sampling each class, as shown in Figure 5.1.

5.2.2 Balanced accuracy

Let us consider *dyslexic* participants as *positive* cases and *non-dyslexic* (intact) as *negative* and look at their classification's correctness. We can sort them into true positives (TPs), false positives (FPs), true negatives (TNs) and false negatives (FNs), as seen in Table 5.1.

Table 5.1: A confusion matrix

		True class	
		Dyslexic	Intact
Predicted class	Dyslexic	TP	FP
	Intact	FN	TN

These counts can be combined into many metrics, among which accuracy is the simplest: it describes the proportion of correctly classified

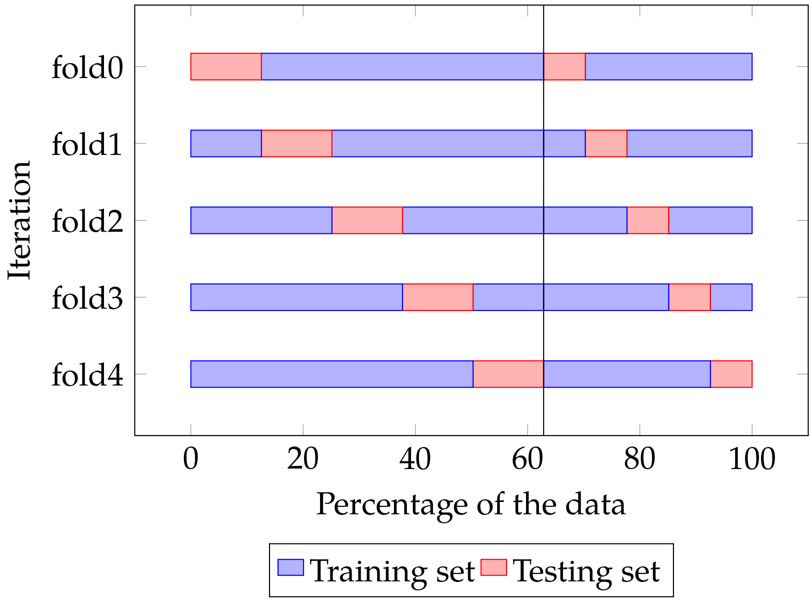


Figure 5.1: Example of stratified 5-fold cross-validation: dyslexic participants on the right of the vertical line, non-dyslexic on the left.

points, as defined in Equation 5.1.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

A baseline model always returning the same class would achieve only 50 % accuracy if we had perfectly balanced data. On the other hand, for imbalanced data, such a model returning the majority class could achieve an accuracy equal to the proportion of that class (i.e. more than 50 %). Since we are typically more interested in detecting the minority class, we have to balance the metric.

One such correction is the balanced accuracy [51], commonly defined as an average of the true positive rate (recall) and the true negative rate (specificity). The most significant advantage of this metric is that the results are fully comparable with accuracy on balanced data, which is the case of most existing research. The slight disadvantage is that insufficient models will achieve 40–60 %; lower values usually only happen due to an implementation mistake (e.g. reversed testing labels).

$$\begin{aligned} recall &= \frac{TP}{TP + FN} \\ specificity &= \frac{TN}{TN + FP} \\ balanced\,accuracy &= \frac{recall + specificity}{2} \end{aligned} \tag{5.2}$$

5.2.3 Hyper-parameter tuning

Most machine learning models come with parameters that can be tuned to improve their performance. The best combination of parameters should be chosen using training data only to avoid overfitting, and the cross-validation techniques can again be used to make the choice more representative.

However, this approach clashes with our decision to use *k*-fold cross-validation for the final evaluation. The synthesis of these practices is called nested *k*-fold cross-validation, where we perform another cross-validation in each fold. The drawback of such a technique is that we practically would not evaluate the performance of a single model (with the same parameters) but of an ensemble of models (with different parameters in each fold).

As a compromise, we decided to conduct the tuning on a single *k*-fold shuffle out of 10. While this does lead to some degree of data leakage (and over-fitting), we believe there is no better way to get stable results on such a small sample.

Since the parameters are usually continuous variables, we cannot test them all, so we have to select and compare only their subset. While an exhaustive search would lead to a more definitive outcome, it is also time-consuming, so a manual search using educated guesses was done instead.

The baseline

As a baseline for our experiments, we have chosen the *k*-nearest neighbours for its simplicity. It has several parameters, like the distance function and, most notably, the *k*. However, since we are utilising it as a baseline, we limit ourselves to 1-nearest neighbour and the Euclidean distance function.

Similarly, the time series variant used Euclidean distance for the DTW calculation. Be aware that tslearn's implementation differs from the definition in Subsection 4.1.1 in that it optimises the root of the sum of squared distances rather than the sum of the distances.

Neural networks

The universal hyper-parameters of neural networks are the learning rate and the number of epochs. The learning rates tested were the negative exponents of 10 $(10^{-1}, \ldots, 10^{-4})$. The epoch number was then chosen with a form of early stopping: the lowest number with the highest accuracy on a given learning rate.

As for the architecture, the global and per AOI MLPs had only a single ReLU hidden layer half the size of the input and single sigmoidal output neuron. The GRUs also had the same output layer and a single hidden layer, but the size was hyper-tuned for each task and ranged between 8 and 32 neurons. Finally, the CNN was the pre-trained resnet18 with two softmax output neurons.

Feature selection was only done for the GRUs (fixation sequences) and is further discussed in Section 6.2.

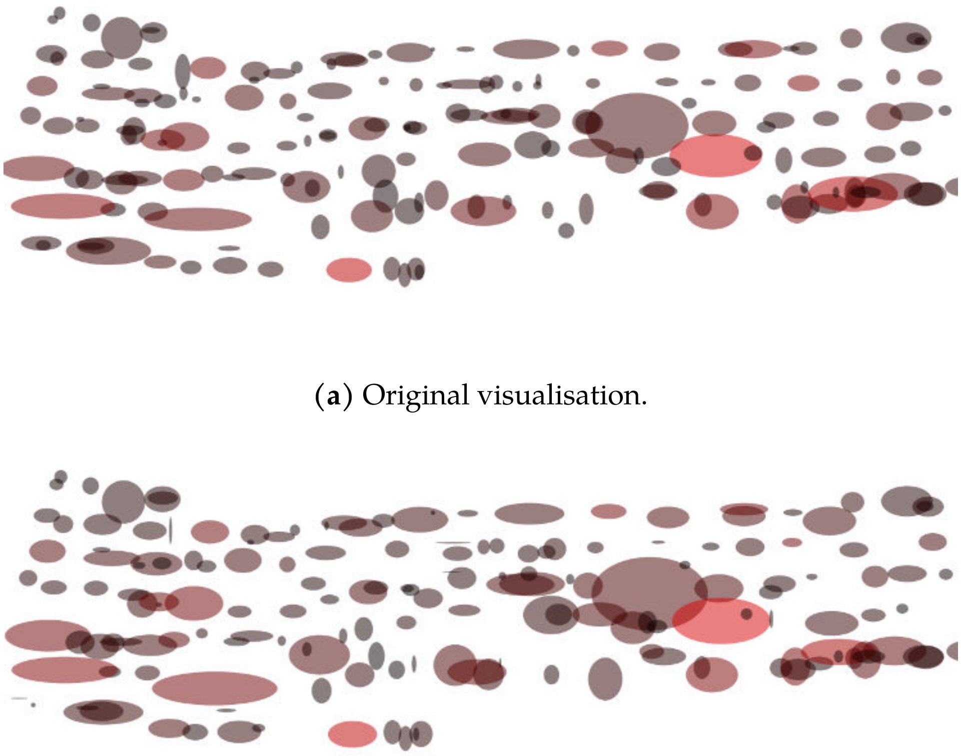
Ensemble

The ensemble is a Majority voting of the same model on different tasks. Since there are four tasks, we must either ignore one task or give two votes to one task. Both approaches were compared on the single 5-fold, where the ignored task was the worst performing, the doubled task was the best performing, and the better performing was used for the final evaluation.

5.3 Data augmentation

Another way to address the data size and imbalance is data augmentation: the idea is to create artificial samples, which can make the training less monotone and balance the data.

Specifically, we will employ augmentation for fixation visualisations by augmenting the fixation sequences. First, we drop some fixations with a probability of 0.1 % and then add gaussian noise (with 0 mean and different standard deviations for each feature). Resulting effect can be seen in Figure 5.2.



(**a**) Original visualisation.

(**b**) Augmented visualisation.

Figure 5.2: Fixation visualisation of the *easy text* reading task.

To keep the running time low, we added three samples per dyslexic participant and 1 per non-dyslexic, which increases the mean proportion of dyslexics in training set from 31.4 % to 47.9 %.

6 Results

Let us remember the experiment setup. We test different representations of gaze event data with two types of models: 1-Nearest neighbour as the baseline and neural networks as our primary focus. We evaluate the models with balanced accuracy on ten shuffles of 5-fold cross-validation, leading to 50 readings that we characterise by their mean and standard deviation. We train each model on four tasks and evaluate a majority voting ensemble built from them.

Section 6.1 explores the statistical-based approach, which aggregates the events on different degrees of granularity. On the other hand, the sequence-based technique is examined in its raw form in Section 6.2 and in image form in Section 6.3. Finally, we conclude by evaluating all 12 models per task and analysing the participants from the point of view of classification error in Section 6.4.

6.1 Gaze event statistics

The three levels of granularity at which we calculate the statistics are on the whole trial (global), on the AOIs and on time windows of equal size.

6.1.1 Global statistics

Looking at Table 6.1, the complete feature set of 45–46 real number statistics shows clear supremacy of the MLP over the baseline, on average by ten p.p. The grid-based task and meaningful text reading have similar results on MLP, around 84 %, but pseudo-text achieves 16 p.p. less. The ensemble of three successful tasks leads to further but negligible improvement. Aside from adam optimiser, L-BFGS was also tried, but the results were five p.p. lower on average (see Appendix A).

Reducing the feature set to 27 features that are also relevant on more granular levels leads to an average performance reduction of almost six p.p. on MLP. Table 6.2 also demonstrates that the neural networks struggled to outperform the baseline.

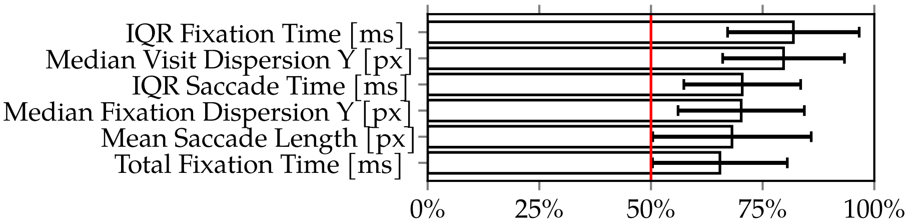
Table 6.1: Ballanced accuracy on global gaze event statistics with complete feature set.

Task	1-NN	MLP
grid	79.87 ± 16.17 %	**82.23 ± 13.58** %
easy text	72.88 ± 16.66 %	**85.22 ± 13.14** %
hard text	69.35 ± 17.99 %	**85.85 ± 13.70** %
pseudo-text	59.55 ± 15.08 %	**68.17 ± 17.58** %
ensemble	81.05 ± 16.96 %	**85.88 ± 12.37** %

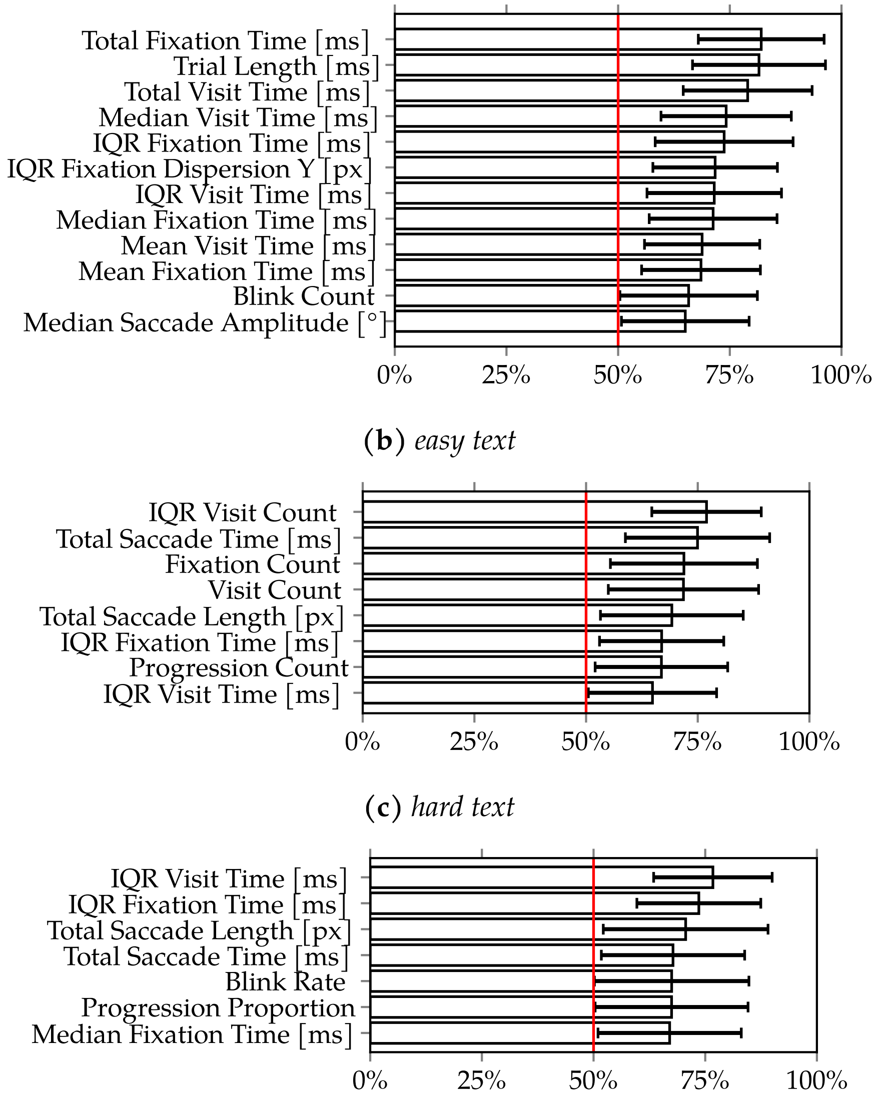
Table 6.2: Ballanced accuracy on global gaze event statistics with reduced feature set.

Task	1-NN	MLP
grid	**73.72 ± 17.81** %	72.87 ± 17.36 %
easy text	73.47 ± 16.64 %	**85.77 ± 13.49** %
hard text	69.50 ± 18.22 %	**72.47 ± 17.21** %
pseudo-text	**68.67 ± 14.68** %	67.33 ± 16.51 %
ensemble	76.78 ± 16.44 %	**84.53 ± 12.75** %

If we look at Figure 6.1, the results from 1-NN models evaluated on each feature separately, the most exciting features across the tasks were IQR of fixation duration, IQR of visit duration (same, but without white space fixations), total fixation duration and total trial duration.



(**a**) *grid*



(**b**) *easy text*

(**c**) *hard text*

(**d**) *pseudo-text*

Figure 6.1: Comparison of the balanced accuracy means of the best 1-NN models trained on single feature; red line shows the worst possible result.

6.1.2 Per AOI statistics

The total fixation duration per AOI (49 grid cells or 9-11 lines of text) follows similar trends to global statistics. Again, Table 6.3 reveals significantly poorer performance on the pseudo-text task, this time by 20 p.p. on MLP. It is also worth mentioning that the training iterations on this task were longer than the rest (over 100 epochs versus below 10). At 80 % accuracy, MLP on per AOI statistics performed worse than on global statistics by four p.p. and was even outperformed by the baseline on the hard task. The L-BFGS results can again be seen in Appendix A.

Table 6.3: Ballanced accuracy on per AOI gaze event statistics.

Task	1-NN	MLP
grid	69.37 ± 14.85 %	**79.60 ± 12.96** %
easy text	73.92 ± 14.87 %	**80.18 ± 14.57** %
hard text	**84.60 ± 14.04** %	81.33 ± 15.21 %
pseudo-text	**61.98 ± 17.35** %	60.83 ± 19.69 %
ensemble	78.53 ± 16.85 %	**80.67 ± 15.08** %

6.1.3 Statistics time series

In addition to the two area-based granularities, time-based was briefly attempted, albeit only with the baseline. As a reminder, we create a series of 5-second windows with 27 statistics, where each event is assigned to the window in which it started. The average accuracy of 71 % is basically the same as the baseline's performance on the global scope with the same features. However, as Table 6.4 presents, this is skewed by exceptional results on the easy task.

6.2 Fixation sequences

While fixation sequences can be represented by up to 5 features (X, Y coordinates, fixation duration and X, Y dispersion), only some combinations were tried. In Table 6.5, we can compare the baseline results on the most interesting ones. We can see a particular usefulness

Table 6.4: Ballanced accuracy on gaze event statistics time series.

Task	1-NN (DTW)
grid	73.22 ± 19.56 %
easy text	83.55 ± 12.58 %
hard text	63.75 ± 17.11 %
pseudo-text	63.42 ± 17.58 %
ensemble	83.78 ± 14.54 %

of the fixation duration, followed by the X coordinate. Interestingly, the Y coordinate seems to play some role in the hard task and pseudo-task.

Table 6.5: Ballanced accuracy of 1-NN on fixation sequences per feature combination (X, Y are coordinates, L is fixation duration and D_X is dispersion on x axis).

	grid	easy text	hard text	pseudo-text
X	49.1 ± 12.4 %	65.7 ± 16.1 %	71.5 ± 15.3 %	71.5 ± 13.8 %
L	67.0 ± 16.0 %	80.2 ± 13.4 %	67.9 ± 16.0 %	61.4 ± 15.5 %
XY	50.1 ± 14.3 %	69.4 ± 16.6 %	86.0 ± 13.0 %	68.5 ± 17.0 %
XL	69.5 ± 15.5 %	77.5 ± 16.5 %	79.7 ± 15.0 %	65.3 ± 16.1 %
XYL	64.9 ± 14.7 %	75.6 ± 17.0 %	82.3 ± 14.1 %	73.8 ± 16.7 %
XLD_X	67.0 ± 16.7 %	80.1 ± 15.9 %	80.7 ± 13.8 %	71.9 ± 15.3 %

The feature combinations for the final evaluation in Table 6.6 were the best combinations on the single 5-fold. The GRU achieves better outcomes on sequences of the hard text and pseud-text than the MLP on global statistics by 2 p.p. and 4 p.p., respectively. However, more feature combinations were tried in this approach. The training also took significantly longer (over 100 epochs compared to below ten on MLPs), which might bring the worth of this approach into question.

6.3 Fixation visualisations

Fixation images visualise the five sequential features, omitting the order. As the results of resnet18 without data augmentation are similar

Table 6.6: Ballanced accuracy on fixation sequences (X, Y are coordinates, L is fixation duration).

Task	1-NN (DTW)		GRU	
grid	69.50 ± 15.48 %	(XL)	**78.95 ± 16.11** %	(XL)
easy text	**80.20 ± 13.43** %	(L)	77.18 ± 15.84 %	(XL)
hard text	85.98 ± 13.01 %	(XY)	**88.07 ± 11.73** %	(XYL)
pseudo-text	**73.82 ± 16.67** %	(XYL)	72.63 ± 15.83 %	(XYL)
ensemble	82.53 ± 15.05 %		**86.20 ± 15.48** %	

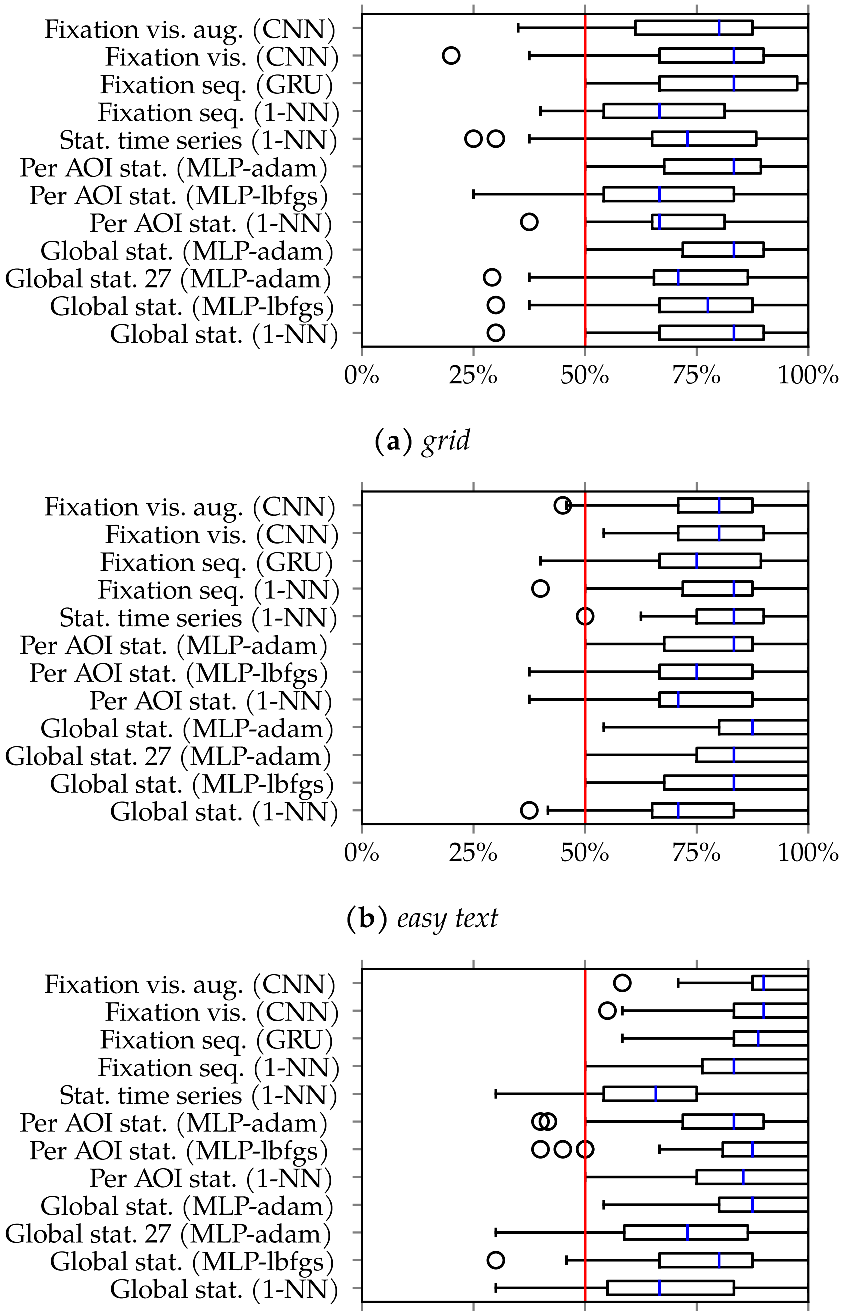
to the fixation sequences, both above 79 % on average, we can conclude that the fixation order is unimportant. Looking at the Table 6.7, the data augmentation has a minimal effect outside of the hard text, only raising the average accuracy by one p.p., but the non-dyslexic points were only doubled. The decline on the grid task could be due to too drastic augmentation of the coordinates, decaying the grid structure.

Table 6.7: Ballanced accuracy on fixation visualisations (augmentation doubles the intact points and quadruples the dyslexics).

Task	resnet18	resnet18 + augmented data
grid	**76.83 ± 18.35** %	75.60 ± 18.17 %
easy text	78.40 ± 14.23 %	**78.87 ± 13.50** %
hard text	88.78 ± 13.18 %	**92.03 ± 9.48** %
pseudo-text	74.13 ± 17.03 %	**75.82 ± 15.41** %
ensemble	88.12 ± 13.75 %	**89.50 ± 12.30** %

6.4 Summary

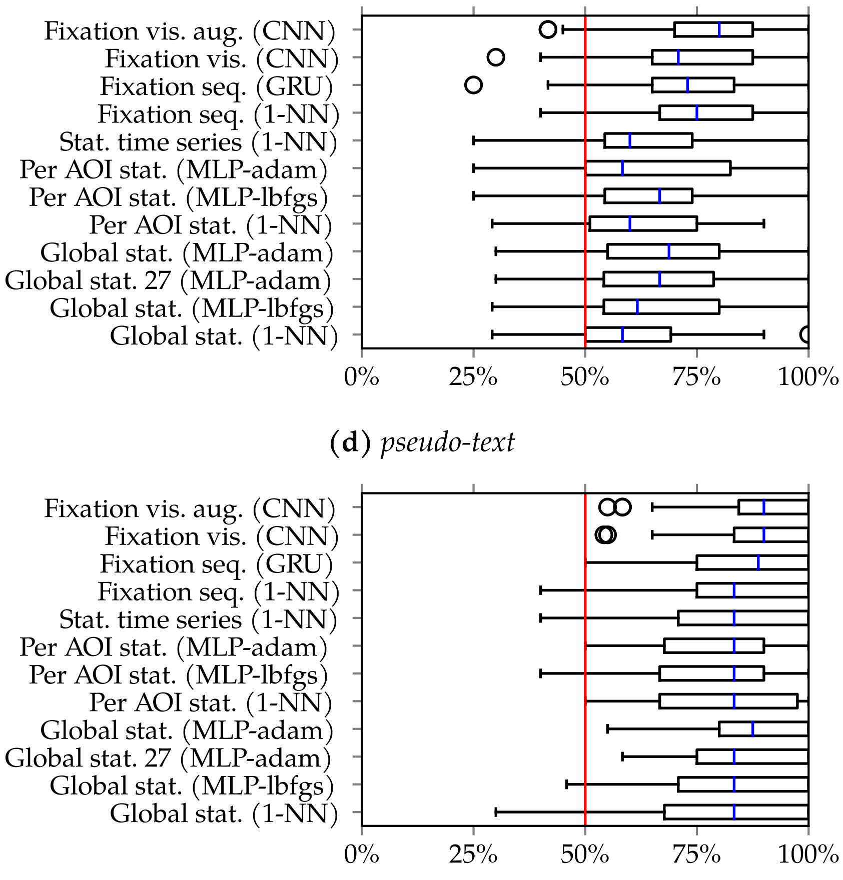
We can reach more interesting observations by looking at the results per task instead of per data type, like in Figure 6.2. First, the boxplots evidently show that models trained on the hard text lead to the best outcomes, followed by the easy text, grid and pseudo-text, respectively. Specifically, pseudo-text fails to achieve more than 80 % on any model. Unless a different method is required, this task is not worthwhile.



(a) *grid*

(b) *easy text*

(c) *hard text*



(**d**) *pseudo-text*

(**e**) *ensemble*

Figure 6.2: Final comparison of the balanced accuracy across the 50 folds per dataset; red line shows the worst possible result.

Ensembles on the same data type generally lead to more stable results but worse than the best model. Notably, the third quartile of the AOI-based models is lower than 100 %, suggesting this is either a less practical approach or that the AOI detection has to be improved. As for the models and data types, the fixation sequences and visualisations worked the best, followed by the global statistics fit with MLP.

If we look at which participants the models failed to classify the most in Figure 6.3, we can see that the most anomalous on the hard text were dyslexic No. 6 and non-dyslexics No. 32 and 37. These participants were considered unusual in reading for their class, which suggests that to improve the classification further, we might need to utilise some non-reading tasks as well.

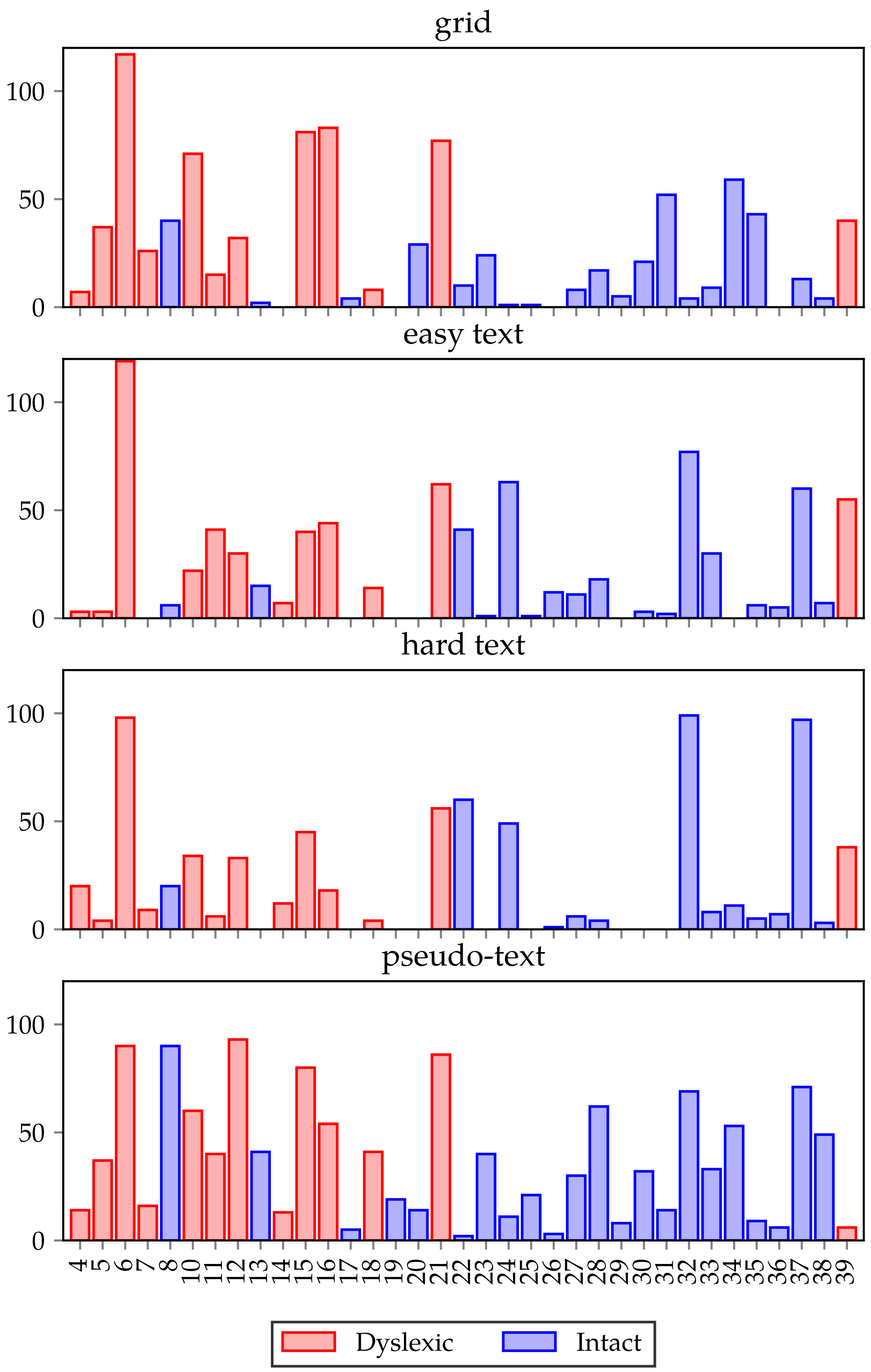


Figure 6.3: The failed experiment count per participant.

7 Conclusion

This thesis aimed to test machine learning on dyslexia detection from text-reading eye movements. For this purpose, we trained a selection of neural network models (MLP, CNN, GRU) and compared them to the 1-NN baseline on various custom data representations. The best model, fine-tuned resnet-18 trained on fixation images with data augmentation, achieved an accuracy of up to 92 %, six p.p. above the best baseline model.

Among the four reading tasks, we identified hard text, the age-appropriate reading task, to work best across models and data representations. The participants the model struggled with had outlying reading abilities for their class, so non-reading tasks should be explored to improve the results further. On the other hand, the pseudo-text failed to prove its usefulness with the methods tried.

The experiments also showed promising results throughout the data representations. The least interesting turned out to be the per AOI statistics, possibly due to only using a single statistic for each AOI. On the opposite side of the spectrum, we have fixation sequences and images that both worked well, even if slower than global statistics (by one and two orders of magnitude, respectively). Data augmentation, only examined on fixation images, only led to improvements on one task, motivating more research.

The results need not be overstated, as the datasets were small and, consequently, the experimental setup had to allow for a degree of data leakage. Similarly, while balanced accuracy accounted for the dataset imbalance in the scoring, the training imbalance could have negatively affected the neural network models. Overall, the model scores were sensitive to the hyper-parameters, so the experiments should be replicated on a more extensive and cleaner dataset.

Additional areas for further research include other data representations (e.g. saccade sequences and images or raw eye-movement time series), pre-processing approaches (e.g. magnitude spectrum of sequences), data augmentation outside the fixation images (e.g. on fixation sequences), feature selection on the statistics and comparing the neural networks to the most researched models (SVMs).

A Results of MLPs with L-BFGS optimiser

The MLPs have also been learnt with Limited-memory BFGS (L-BFGS) in the early stages as it works well on small datasets without hyperparameter tuning. While, in general, it performed worse than adam, there is a rare occurrence of the ensemble noticeably improving the score beyond the best task model on global statistics. Specifically, the accuracy has grown from an average of 78 % on the three tasks (without pseudo-text) to 85 %.

Table A.1: Ballanced accuracy from MLP with L-BFGS optimiser.

Task	Global statistics	Per AOI statistics
grid	**75.80 ± 16.47** %	68.32 ± 18.87 %
easy text	**81.60 ± 14.54** %	76.58 ± 16.31 %
hard text	77.33 ± 15.97 %	**85.48 ± 15.42** %
pseudo-text	**65.07 ± 18.37** %	64.67 ± 15.87 %
ensemble	**85.02 ± 15.14** %	79.13 ± 16.59 %

B Code structure

The implementation of the thesis is included in the SDIPR.zip attachment, which is structered as follows:

```
SDIPR.zip
|__data
|  |__...
|__exports
|  |__data
|  |  |__...
|  |__predicted
|     |__...
|__utils
|  |____init__.py
|  |__aoi.py
|  |__ensenmble.py
|  |__evaluate.py
|  |__io.py
|  |__model.py
|  |__preprocessing.py
|  |__pytorch.py
|  |__transform.py
|__evaluate_fixation_sequences.ipynb
|__evaluate_fixation_visualisations.ipynb
|__evaluate_global_statistics.ipynb
|__evaluate_per_aoi_statistics.ipynb
|__evaluate_statistics_time_series.ipynb
|__extract_features.ipynb
|__final_evaluation.ipynb
|__README.md
|__spec-file.yml
```

The root of the archive contains instructions for environment preparation README.md, the *conda* environment specification spec-file.yml and the Jupyter notebooks *.ipynb. Each evaluate_* notebook contains experiment for the given data representation and final_evaluation has some further analyses. The extract_features.ipynb contains the code for transforming the raw data into the final representations.

`data` directory

This directory would normally contain the gaze event data, but they were excluded at the request of the authors. Note that the names of the tasks are bit different from how they were named in the thesis:

- T1 = grid;
- T3-bert = easy text;
- T3-veverka = hard text;
- T4 = pseudo-text.

`exports` directory

This directory contains both the final data representations (in `data`) and the predicted classes from the experiments (in `predicted`).

`utils` directory

This directory contains all the Python code produced for the thesis, split into multiple modules:

- `aoi.py`: Functions for AOI detection and identification;
- `ensemble.py`: Implements the majority voting ensemble from predictions;
- `evaluate.py`: Contains functions to run the experiments;
- `io.py`: Functions for loading and saving data;
- `model.py`: Classes that represent the neural networks;
- `preprocessing.py`: Contains the min-max scaler for time series data;
- `pytorch.py`: Utility classes and functions to prepare data for PyTorch;
- `transform.py`: Functions for extracting the examined data representations from the gaze event data.

Bibliography

1. SHAYWITZ, S. E. Dyslexia. *New England Journal of Medicine*. 1998, vol. 338, no. 5, pp. 307–312. Available from DOI: `10.1056/NEJM199801293380507`. PMID: 9445412.
2. UNDHEIM, A. M. A thirteen-year follow-up study of young Norwegian adults with dyslexia in childhood: reading development and educational levels. *Dyslexia*. 2009, vol. 15, no. 4, pp. 291–303. Available from DOI: `10.1002/dys.384`.
3. GLAZZARD, J. The impact of dyslexia on pupils' self-esteem. *Support for Learning*. 2010, vol. 25, no. 2, pp. 63–69. Available from DOI: `10.1111/j.1467-9604.2010.01442.x`.
4. SNOWLING, M. J.; HULME, C. Interventions for children's language and literacy difficulties. *International Journal of Language & Communication Disorders*. 2012, vol. 47, no. 1, pp. 27–34. Available from DOI: `10.1111/j.1460-6984.2011.00081.x`.
5. CARTER, B. T.; LUKE, S. G. Best practices in eye tracking research. *International Journal of Psychophysiology*. 2020, vol. 155, pp. 49–62. ISSN 0167-8760. Available from DOI: `10.1016/j.ijpsycho.2020.05.010`.
6. TINKER, M. A. Recent studies of eye movements in reading. *Psychological bulletin*. 1958, vol. 55, no. 4, p. 215. Available from DOI: `10.1037/h0041228`.
7. KAISAR, S. Developmental dyslexia detection using machine learning techniques : A survey. *ICT Express*. 2020, vol. 6, no. 3, pp. 181–184. ISSN 2405-9595. Available from DOI: `10.1016/j.icte.2020.05.006`.
8. *ICD-11: International Classification of Diseases 11th Revision* [online]. World Health Organization. 02/2022 [visited on 2023-02-01]. Available from: `https://icd.who.int/browse11/l-m/en#/http://id.who.int/icd/entity/1008636089`.
9. SHAYWITZ, B. A.; SHAYWITZ, S. E. The American experience: towards a 21st century definition of dyslexia. *Oxford Review of Education*. 2020, vol. 46, no. 4, pp. 454–471. Available from DOI: `10.1080/03054985.2020.1793545`.

10. PETERSON, R. L.; PENNINGTON, B. F. Developmental dyslexia. *The Lancet*. 2012, vol. 379, no. 9830, pp. 1997–2007. ISSN 0140-6736. Available from DOI: 10.1016/S0140-6736(12)60198-6.
11. HOEFT, F.; MCCARDLE, P.; PUGH, K. The myths and truths of dyslexia in different writing systems. *The Examiner*. 2015. Available also from: https://dyslexiaida.org/the-myths-and-truths-of-dyslexia/.
12. KLAIB, A. F. et al. Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies. *Expert Systems with Applications*. 2021, vol. 166, p. 114037. ISSN 0957-4174. Available from DOI: 10.1016/j.eswa.2020.114037.
13. HENDRICKSON, A. Fovea: Primate. In: SQUIRE, L. R. (ed.). *Encyclopedia of Neuroscience*. Oxford: Academic Press, 2009, pp. 327–334. ISBN 978-0-08-045046-9. Available from DOI: 10.1016/B978-008045046-9.00920-7.
14. RAYNER, K. The 35th Sir Frederick Bartlett Lecture: Eye movements and attention in reading, scene perception, and visual search. *Quarterly Journal of Experimental Psychology*. 2009, vol. 62, no. 8, pp. 1457–1506. Available from DOI: 10.1080/17470210902816461. PMID: 19449261.
15. VASILEV, M. R.; ANGELE, B. Parafoveal preview effects from word N+ 1 and word N+ 2 during reading: A critical review and Bayesian meta-analysis. *Psychonomic Bulletin & Review*. 2017, vol. 24, pp. 666–689. Available from DOI: 10.3758/s13423-016-1147-x.
16. HOLMQVIST, K. et al. *Eye Tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011. ISBN 9780191625428. Available also from: https://books.google.sk/books?id=5rIDPV1EoLUC.
17. RAYNER, K. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*. 1998, vol. 124, no. 3, p. 372. Available from DOI: 10.1037/0033-2909.124.3.372.
18. AL-EDAILY, A.; AL-WABIL, A.; AL-OHALI, Y. Dyslexia Explorer: A Screening System for Learning Difficulties in the Arabic Language Using Eye Tracking. In: HOLZINGER, A. et al. (eds.). *Human Factors in Computing and Informatics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 831–834. Available from DOI: 10.1007/978-3-642-39062-3_63.

19. RELLO, L.; BALLESTEROS, M. Detecting Readers with Dyslexia Using Machine Learning with Eye Tracking Measures. In: *Proceedings of the 12th International Web for All Conference*. Florence, Italy: Association for Computing Machinery, 2015. W4A '15. ISBN 9781450333429. Available from DOI: 10.1145/2745555.2746644.
20. NILSSON BENFATTO, M. et al. Screening for Dyslexia Using Eye Tracking during Reading. *PLOS ONE*. 2016, vol. 11, no. 12, pp. 1–16. Available from DOI: 10.1371/journal.pone.0165508.
21. CERAVOLO, I. d. A.; BRASIL, A. R. A.; KOMATI, K. S. Classifying readers with dyslexia from eye movements using machine learning and wavelets. In: *Proceedings of 8th Brazilian Conference on Intelligent Systems (BRACIS)*. 2019. Available also from: https://sol.sbc.org.br/index.php/eniac/article/download/9342/9244.
22. JOTHI PRABHA, A.; BHARGAVI, R. Predictive Model for Dyslexia from Fixations and Saccadic Eye Movement Events. *Computer Methods and Programs in Biomedicine*. 2020, vol. 195, p. 105538. ISSN 0169-2607. Available from DOI: 10.1016/j.cmpb.2020.105538.
23. JOTHI PRABHA, A.; BHARGAVI, R. Prediction of Dyslexia from Eye Movements Using Machine Learning. *IETE Journal of Research*. 2019, vol. 68, no. 2, pp. 814–823. Available from DOI: 10.1080/03772063.2019.1622461.
24. NERUŠIL, B. et al. Eye tracking based dyslexia detection using a holistic approach. *Scientific Reports*. 2021, vol. 11, no. 1, pp. 1–10. Available from DOI: 10.1038/s41598-021-95275-1.
25. SMYRNAKIS, I. et al. RADAR: A novel fast-screening method for reading difficulties with special focus on dyslexia. *PLOS ONE*. 2017, vol. 12, no. 8, pp. 1–26. Available from DOI: 10.1371/journal.pone.0182597.
26. ASVESTOPOULOU, T. et al. *DysLexML: Screening Tool for Dyslexia Using Machine Learning*. arXiv, 2019. Available from DOI: 10.48550/ARXIV.1903.06274.
27. SZALMA, J.; WEISS, B. Data-Driven Classification of Dyslexia Using Eye-Movement Correlates of Natural Reading. In: *ACM Symposium on Eye Tracking Research and Applications*. Stuttgart, Germany: Association for Computing Machinery, 2020. ETRA '20

Short Papers. ISBN 9781450371346. Available from DOI: 10.1145/3379156.3391379.

28. SZALMA, J. et al. Investigating the Effect of Inter-letter Spacing Modulation on Data-Driven Detection of Developmental Dyslexia Based on Eye-Movement Correlates of Reading: A Machine Learning Approach. In: DEL BIMBO, A. et al. (eds.). *Pattern Recognition. ICPR International Workshops and Challenges*. Cham: Springer International Publishing, 2021, pp. 467–481. ISBN 978-3-030-68796-0. Available from DOI: 10.1007/978-3-030-68796-0_34.
29. WEISS, B.; SZALMA, J.; VIDNYÁNSZKY, Z. *Data-driven detection of developmental dyslexia: A machine learning approach based on behavioral and eye-movement features*. PsyArXiv, 2022-06-20. Version 4. Available from DOI: 10.31234/osf.io/qasnc.
30. VAJS, I. et al. Spatiotemporal Eye-Tracking Feature Set for Improved Recognition of Dyslexic Reading Patterns in Children. *Sensors*. 2022, vol. 22, no. 13. ISSN 1424-8220. Available from DOI: 10.3390/s22134900.
31. VAJS, I. et al. Dyslexia detection in children using eye tracking data based on VGG16 network. In: *2022 30th European Signal Processing Conference (EUSIPCO)*. 2022, pp. 1601–1605. Available from DOI: 10.23919/EUSIPCO55093.2022.9909817.
32. GRAN EKSTRAND, A. C.; NILSSON BENFATTO, M.; ÖQVIST SEIMYR, G. Screening for Reading Difficulties: Comparing Eye Tracking Outcomes to Neuropsychological Assessments. In: *Frontiers in Education*. Frontiers Media SA, 2021, vol. 6, p. 643232. Available from DOI: 10.3389/feduc.2021.643232.
33. EL HMIMDI, A. E. et al. Predicting Dyslexia and Reading Speed in Adolescents from Eye Movements in Reading and Non-Reading Tasks: A Machine Learning Approach. *Brain Sciences*. 2021, vol. 11, no. 10. ISSN 2076-3425. Available from DOI: 10.3390/brainsci11101337.
34. RAATIKAINEN, P. et al. Detection of developmental dyslexia with machine learning using eye movement data. *Array*. 2021, vol. 12, p. 100087. ISSN 2590-0056. Available from DOI: 10.1016/j.array.2021.100087.
35. LUSTIG, J. *Identifying dyslectic gaze pattern: Comparison of methods for identifying dyslectic readers based on eye movement patterns*. 2016. Available also from: https://www.diva-portal.org/smash/

record.jsf?pid=diva2%3A955646. MA thesis. KTH, School of Computer Science and Communication.

36. HALLER, P. et al. *Eye-tracking based classification of Mandarin Chinese readers with and without dyslexia using neural sequence models*. arXiv, 2022. Available from DOI: 10.48550/ARXIV.2210.09819.
37. ZHAN, Z. et al. Online Learners' Reading Ability Detection Based on Eye-Tracking Sensors. *Sensors*. 2016, vol. 16, no. 9. ISSN 1424-8220. Available from DOI: 10.3390/s16091457.
38. LOU, Y. et al. Using support vector machines to identify literacy skills: Evidence from eye movements. *Behavior research methods*. 2017, vol. 49, no. 3, pp. 887–895. Available from DOI: 10.3758/s13428-016-0748-7.
39. BEDNÁŘOVÁ, J. *Diagnostika schopností a dovedností v oblasti čtení a psaní: Varianta pro pedagogy škol a školní poradenská pracoviště, 3. a 4. ročník*. Brno: Pedagogicko-psychologická poradna Brno, 2015.
40. MITCHELL, T. M. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997. ISBN 0070428077. Available also from: http://www.cs.cmu.edu/~tom/mlbook.html.
41. ZEZULA, P. et al. *Similarity Search: The Metric Space Approach*. 2005th ed. New York, NY 10013, USA: Springer, 2005. ISBN 0-387-29146-6.
42. COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*. 1967, vol. 13, no. 1, pp. 21–27. Available from DOI: 10.1109/TIT.1967.1053964.
43. SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 1978, vol. 26, no. 1, pp. 43–49. Available from DOI: 10.1109/TASSP.1978.1163055.
44. PANDEY, A.; JAIN, A. Comparative analysis of KNN algorithm using various normalization techniques. *International Journal of Computer Network and Information Security*. 2017, vol. 11, no. 11, pp. 36–42. Available from DOI: 10.5815/ijcnis.2017.11.04.
45. GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. MIT Press, 2016. Available also from: https://www.deeplearningbook.org/.
46. HESS, R. *Introducing: Flickr PARK or BIRD* [online]. Flickr, 2014 [visited on 2023-04-17]. Available from: https://code.flickr.net/2014/10/20/introducing-flickr-park-or-bird/.

47. DOGAN, A.; BIRANT, D. A Weighted Majority Voting Ensemble Approach for Classification. In: *2019 4th International Conference on Computer Science and Engineering (UBMK)*. 2019, pp. 1–6. Available from DOI: 10.1109/UBMK.2019.8907028.

48. PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, no. 85, pp. 2825–2830. Available also from: http://jmlr.org/papers/v12/pedregosa11a.html.

49. TAVENARD, R. et al. Tslearn, A Machine Learning Toolkit for Time Series Data. *Journal of Machine Learning Research*. 2020, vol. 21, no. 118, pp. 1–6. Available also from: http://jmlr.org/papers/v21/20-091.html.

50. KOHAVI, R. et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *Proceedings of the Fourteenth International Joint Conference on Artificial IntelligenceIjcai*. 1995, vol. 14, pp. 1137–1145. No. 2. Available also from: https://www.ijcai.org/Proceedings/95-2/Papers/016.pdf.

51. BRODERSEN, K. H. et al. The Balanced Accuracy and Its Posterior Distribution. In: *2010 20th International Conference on Pattern Recognition*. 2010, pp. 3121–3124. Available from DOI: 10.1109/ICPR.2010.764.