Aggregating Algorithm

Candidate 2408208

Submitted for the Degree of Master of Science in

Machine Learning

Department of Computer Science
Royal Holloway University of London
Egham, Surrey TW20 0EX, UK

August 28, 2024

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count:

Candidate Number: 2408208

Date of Submission: 29 August 2024

Signature:

Abstract

This dissertation investigates how well humans generate objectively random binary sequences by evaluating human-generated sequences against true randomness using a variety of statistical methods, as well as assessing the predictability of such sequences using the Aggregating Algorithm, inspired by Markov Chains.

The study is designed with a within-subject approach and makes use of a web application to allow subjects—primarily postgraduate students at Royal Holloway, University of London—to input binary sequences sequentially. As the subject enters their sequence, the bits are analysed by various "Experts", each of whom make their own predictions on the next bit in the sequence prior to the subject pressing a key. These predictions are then mixed using the Aggregating Algorithm to form a single prediction which is then compared against the observed outcome entered by the subject.

This study measured various dependent variables, including the frequences of 0s and 1s (Heads and Tails), run lengths, and predictions from a variety of experts as well as the algorithm itself. The data gathered was evaluated using chi-square goodness-of-fit tests and by analysing the regret associated with each Expert/Learner to compare the generated sequences to statistically random processes. In summary, the Aggregating Algorithm's performance was evaluated, revealing that human-generated sequences do suffer from biases and are, therefore, predictable to a statistically significant extent better than is possible with random guessing.

Keywords: On-line Learning, Prediction with Expert Advice, Aggregating Algorithm for Specialist Experts, Perceived Randomness

Contents

Acknowledgements

While the contents of this report are based on my work, none of this would have been possible without the patience and mentorship of my supervisor to whom I am extremely grateful. It was your advice, clear explanations, and expertise that made this project what it is now and something that I am incredibly proud of. I would also like to express my gratitude to the group of friends who made this academic year possible, namely Cougar Tasker, Einstein Ebereonwu, Hayden Amarr, Mohammadreza Yazdian, Niraj Jain, and Ray Mahbub, without whom I would have struggled to maintain my discipline and motivation.

1 Introduction

1.1 Project Scope and Objectives

This project aims to implement the Aggregating Algorithm for Specialist *(Sleeping)* Experts, a method of Prediction with Expert Advice, to scenarios involving human-generated sequences and evaluate its predictive performance in pre-empting what the human subject will input, effectively testing how well human subjects generate random inputs.

As an introduction to the concepts that will be explored in the following sections, this algorithm allows for the effective pooling of different prediction algorithms, known as 'Experts', to improve the algorithm's prediction accuracy. Aggregating several predictions allows the final prediction outputted by the algorithm to be nearly as accurate as that of the best-performing Expert.

This project will encompass several key areas, including:

- **Explaining the Theory of Perceived Randomness.** The basis of this study revolves around the human perception of randomness, which is different from objective randomness, therefore the underlying psychological mechanisms for how humans judge and perceive randomness must be understood.

- **Explaining the Theory of Prediction with Expert Advice.** The other portion of this study is firmlyon the subject of Prediction with Expert Advice, primarily focussing on the Aggregating Algorithm and the Aggregating Algorithm for Specialists Experts so the underlying theory must be explored with a thorough review of the current literature.

- **Implementing the Aggregating Algorithm.** This project will primarily investigate the Aggregating Algorithm introduced by Vovk (see [1], [2]).

- **Handling Specialist Experts.** Introduced by Freund [3], *Specialist Experts* may refrain from making predictions at certain points, meaning that the Aggregating Algorithm has to be modified slightly [4].

- **Evaluating the Performance of Human Subjects in Generating Statistically Random Sequences.** Through conducting the experiment outlined in this paper, this study will present how well the

subjects were able to generate a "random" sequence when compared to the statistical definition used by statisticians.

- **Evaluating the Performance of the Algorithm in Predicting Human-Generated Outcomes.** The experiment will also compare how well the Aggregating Algorithm for Specialist Experts was able to pre-empt each subject's sequences, in a somewhat adversarial comparison to statistical randomness.

1.2 Motivation and Interest in the Subject Area

The motivation for selecting a project in this subject area is rooted in both my personal and professional interests, as well as the discussions I had with Dr. Yuri Kalnishkan before finalising my selection.

During this academic year, the module that most piqued my interest was CS5200 – On-line Machine Learning because I was interested in the techniques that allowed machine learning models to gradually improve over time as more data became available to them without the need to retrain the model on the entire newly-updated dataset; something that had not been covered previously by other modules. Due to the module's small size and frequent absentees, I was able to gain a deeper understanding of the module, in large part due to Dr. Kalnishkan's willingness to explain portions of the syllabus in extreme detail. Alongside the lectures, I felt like I was strongly suited to the contents of the module because it has strong ties to the field of statistics – another area that I thoroughly enjoyed throughout my education.

Regarding my professional aspirations, I am set to begin my career later this year and I am of the firm belief that the work that I have done in this subject area is highly relevant, not only to the job I am to start in September, but also for my career plan due to its relevance across a variety of industries – including finance, energy, and insurance.

Ultimately, the combination of all of these factors led me to pursue a project investigating on-line prediction, and prediction with expert advice.

1.3 Structure of the Dissertation

This dissertation is split into several distinct chapters that explore a specific aspect of the study. Provided below is an outline, providing an overview of each chapter's contents.

This study is split into several distinct chapters and subsections, each exploring a specific aspect of the study being conducted. The following is

an outline to help guide you through the report and provide a brief overview of each chapter's contents.

The following study is split into several distinct chapters and subsections, each dedicated to exploring a specific aspect of the study being conducted. The following outline guides you, the reader, through the report by providing a brief overview of the contents of each chapter.

Chapter 2 contains the Literature Review that is organised to explain the underlying theory that the practical portion of this study aims to investigate.

Section 2.2 explores the human perception of randomness and how it is, in fact, different from the objective definition of randomness, and aims to provide context to answer the question "Are humans good randomisers?" in both judgement and generation scenarios.

Section 2.3 is diverse in its contents. Subsection 2.3.1 defines the problem of On-line Prediction, outlining the scenarios in which it is applicable, and the protocols that such problems follow. Additionally, it explores how On-line Prediction differs from the traditional Machine Learning frameworks of Batch Learning and Timeseries Analysis and defines concepts that will be critical to understanding the experiment being conducted. Subsection 2.3.2 expands on the information presented in the previous subsection by defining the modifications to the original framework to now incorporate the predictions made by a pool of Experts and how the Learner aggregates those to inform its own. It also gives two examples of merging strategies that could be used to accomplish that task. Subsection 2.3.4 introduces the algorithm that is the main focus of this study, delving into the rationale behind the algorithm and discussing the bounds guaranteed by utilising it. Lastly, Subsection 2.3.5 defines a modification to the original Aggregating Algorithm that allows it to be used in scenarios involving Specialist Experts, a term used to define any Expert that can abstain from making a prediction.

Chapter 3 is centred on the practical applications of the theory and how it was used to derive the experiment that is to be conducted, including how the experiment was designed, how the Aggregating Algorithm was applied to the proposed scenario and how the data gathered from the subjects will be analysed.

Chapter 4 will discuss the findings of the study in detail, ultimately attempting to answer the hypothesised question "Are humans good randomisers?"

Finally, Chapter 5 contains a conclusion that summarises the findings of the study, as well as a self-evaluation of the project.

2 Literature Review

2.1 Introduction

Through exploring the human capacity for judging and generating random binary sequences, this Literature Review will examine a variety of studies that delve into the complex, and somewhat subconscious, relationship between what a human perceives as random and what is truly random according to the statistical definition.

The term 'random" inherently means something that is happening by chance with no cause or reason, however, the method by which statisticians measure randomness is through statistical tests, ultimately meaning that there is some measurable quantity to determine whether something conforms to what we objectively consider random. That being said, humans have a subjective view of randomness which may deviate significantly from what is considered objective randomness. The discrepancy between objective and subjective randomness forms the basis of this study, particularly in the context of whether humans are capable of producing sequences that are statistically indistinguishable from those that are expected of a random process.

A fundamental debate that is found throughout the literature reviewed in this study is whether humans, when asked to perceive and generate random sequences, can truly do so without identifying "patterns" that occur by coincidence regarding judgement tasks, or without introducing subconscious patterns and biases regarding production tasks. Early research conducted by Reichenbach, with subsequent studies performed by Ross, Wagenaar, and Nickerson (to name a few), highlight a complex and often contradictory knowledge base with some of their findings indicating that humans are inherently poor at identifying and generating random sequences, while others suggest the opposite in that humans might approximate randomness better than previously thought.

This review does not aim to provide a definitive answer to the question, however, it does synthesise the current understanding of how humans perceive randomness, as well as connects these findings to the broader context of prediction algorithms—by understanding the limitations of human-generated randomness, insights can be gained into how these subconscious biases might cause algorithms specifically designed for prediction in scenarios with incomplete information, such as the Aggregating Algorithm, to perform better than would be expected of a truly random process. This hypothesis forms the basis of the experimental work performed in this study.

2.2 Perceived Randomness

2.2.1 Are Humans Good Randomisers?

The definition of the term "random" is a contentious area for debate. In essence, randomness is an unobservable characteristic of a generating process therefore the act of trying to define it is somewhat contradictory. To determine if a process or sequence is random, it has to be put through statistical tests for specific properties which are deemed to be "random". However, because these tests are statistical, are the conclusions drawn *objectively* random or purely *subjective*?

Research into the human perception and generation of random sequences is a common topic within psychological papers yet the contradictory nature of findings results in a less-than-satisfactory answer to the question of "Are humans good randomisers?" – much like when defining the term itself.

The origins of such a question can be traced back to an observation made by Hans Reichenbach in The Theory of Probability [5]; he suggested that when asked to produce a series that seemed random to them, people untrained in the theory of probability would be unable to generate such a series and, instead, generate one that would contain patterns and biases, e.g. too many alternations than what was expected. This ultimately suggests that humans are not good randomisers which is the prevalent opinion to date. This behaviour is attributed to the fact that human-generated sequences often reflect the underlying psychological tendencies of subjects, rather than the unpredictability of true randomness.

While Reichenbach takes the stance that humans are not good randomisers, the alternative to this conclusion was put forward by the work of Bruce Ross [6]. Ross explores the processes involved in randomising binary sequences and analyses the methods that people use, as well as the typical mistakes that they make when attempting to create random sequences. In his study, Ross got 60 subjects to stamp cards with either an '$\mathcal{O}$' or an '$\mathcal{X}$' and place them singly in a 100-item sequence that they thought to be random in the middle of a table, with item frequencies of either $50-50$, $60-40$, or $70-30$. These sequences were then scored against the expected properties of a random sequence and, based on the analysis conducted, resulted in "the prevalent *a priori* assumption that the human being is a systematically biased randomi[s]er [not being] borne out" [6] and that "[subjects] who are instructed to construct a random series give a fairly good approximation of the expected number of alternations" [7]. This, however, is not sufficient to deem that humans are not systematically biased randomisers.

2.2.2 Judgement vs. Production of Random Binary Sequences

As alluded to by the question posed in the previous subsection, the human perception of randomness is a *subjective*, rather than *objective*, quality. Because of this subjectivity, it will vary from person to person and two natural conclusions can be drawn from this—either that people have an incorrect idea of what randomness is and what it should look like, or that people intuitively know what true randomness should look like, but there is some internal functional limitation that prevents the judgement and production of such sequences [8], being so powerful that individuals may choose to forego an available statistical analysis in favour of this 'gut feeling' [9].

Being that the topic of this dissertation is Prediction with Expert Advice, specifically in the scenario of η-mixable Games—a subject to be introduced in the following section—the primary focus of this literature review will be centred around experiments conducted to explore the judgement and production of random binary sequences. These two categories both make interesting observations about the internal mechanism responsible for the human perception of randomness, namely "that [humans] see clumps or streaks in truly random series and expect more alternation, or shorter runs, than are there", and that "[humans] produce series with higher than expected alternation rates" [9].

2.2.2.1 Judgement of Random Binary Sequences

We will first explore *judgement*. In Willem Wagenaar's study titled "Appreciation of conditional probabilities in binary sequences", Wagenaar examines how people perceive and interpret the likelihood of certain events occurring given previous outcomes revealing a disparity between what was perceived to be random and what was truly random, as well a systematic recency bias that affected subject's judgements of conditional probabilities [8]. The study controlled the conditional probability of a 0 after 0 (1 after 1) as the experimental variable, testing it between the range $0.2 - 0.8$ with 0.1 increments, i.e. 7 values, for first-, second-, and third-orders of dependency. To test this, subjects were shown 16 sets of 7 binary sequences (each generated with one of the conditional probabilities in the range) for each order of dependency and were asked to select and record the sequence in each set that looked the most random to them—explained as the sequence that looked the most likely to be produced when flipping a fair coin.

For reference, in a truly random binary sequence, the conditional probability of 0 after 0 ($\Pr(0|0)$) or 1 after 1 ($\Pr(1|1)$) for the first order of dependency is 0.5. However, Wagenaar identified sequences with conditional

probabilities close to 0.4 were the ones perceived as the most random across all orders of dependency, affirming the position that humans aren't good randomisers. This study also highlights the bias in favour of 'negative recency', more commonly known as the gambler's fallacy wherein gamblers will tend to bet on red after a run of blacks (and vice versa) on a roulette wheel. This observation ultimately caused subjects to favour series with slightly more alternations than is expected of true randomness causing Wagenaar to postulate that this is because subjects "cannot process such a mathematical quantity as 'conditional probability'... Rather, they will look at some other characteristics like, for instance, the run-structure of the sequence" [8].

2.2.2.2 Production of Random Binary Sequences

Having introduced the topic and a systematic bias that affects how humans judge sequences to be random, we can now delve into generation—the category that this project will explore.

Examining Paul Bakan's work titled "Response-Tendencies in Attempts to Generate Random Binary Series", Bakan aimed to "allow for another test of the hypothesis that [a subject] will generate more runs than chance predicts under conditions somewhat different from those reported by Ross" in that biases in motor operations (e.g., favouring to use their dominant hand) was avoided [7]. As stated by Bakan, the main findings of this study are that subjects "exhibit consistent patterns of responses" and "deviate from randomness by having too many alternations in the series" when trying to generate a random binary sequence—a conclusion supported by [10]: "humans-produced sequences have too few symmetries and long runs, too many alternations among events, and too much balancing of event frequencies over relatively short regions" which may be explained by the fact that a human's short-term memory roughly spans 7 (+/- 2) items that constitute the "window" that people try to achieve representative randomness in [11].

Lastly, we explore the work of Raymond Nickerson and Susan Butler titled "On producing random binary sequences" which forms the basis of the experiment that this project will carry out. Nickerson and Butler's experiment varies from previous ones carried out in that, instead of getting subjects to produce a single sequence that would later be aggregated into a larger collective, they got subjects to produce several sequences while attempting to be random since they noted that "randomness does not reveal itself in any single short sequence; it reveals itself in sets of such sequences. Or at least it has a better opportunity to reveal itself in a set of sequences rather than in a single member of such a set." [12]. In their methodology, subjects

were tasked with producing 100 10-item random sequences—explained as the sequences likely to be recorded if 100 individuals were asked to flip a fair coin 10 times each—that would be statistically indistinguishable from if an actual coin were to have been flipped. The notion behind this experiment was that, if subjects' perceptions of randomness were good, then subjects would be able to produce sets of sequences with properties (i.e., number of heads per sequence, number of runs per sequence, run lengths, frequency of alternations and repetitions) that fell within expected percentages. Because each subject was made to produce several sequences, their results provided a stronger justification for a human's ability as a randomiser since it allows subjects time to prove that they can act randomly. What Nickerson and Butler found was that, while subjects weren't any good at producing truly random sequences since "in the aggregate, the sets of sequences produced by our participants differed quantitatively from those expected of a random process, so our results can be seen as supporting the prevailing view that people are not very good randomi[s]ers", the distribution shape produced by the aggregate of participants' sequences were qualitatively similar—"not indistinguishable, but close" [12]—to what was expected (shown in the figures below), suggesting that humans can be effective randomisers when part of a group.

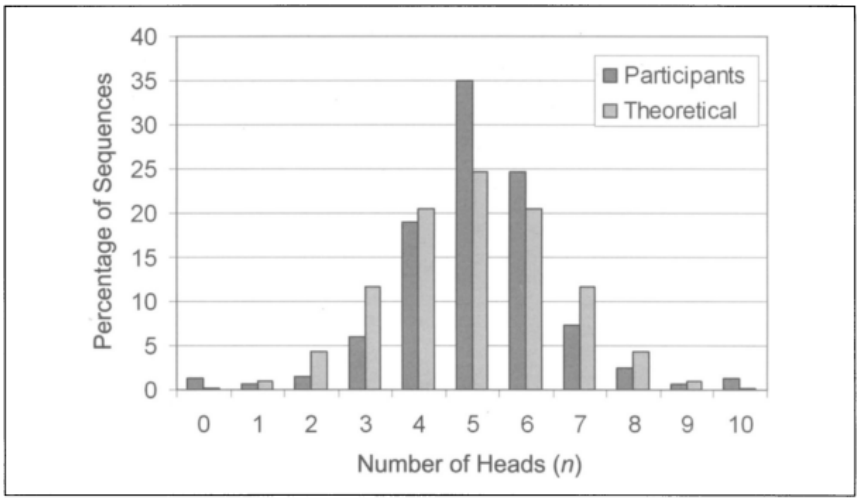

Figure 1: Percentage of 10-toss sequences with n heads, including the theoretical distribution, $X \sim B(10, 0.5)$, for comparison [12].

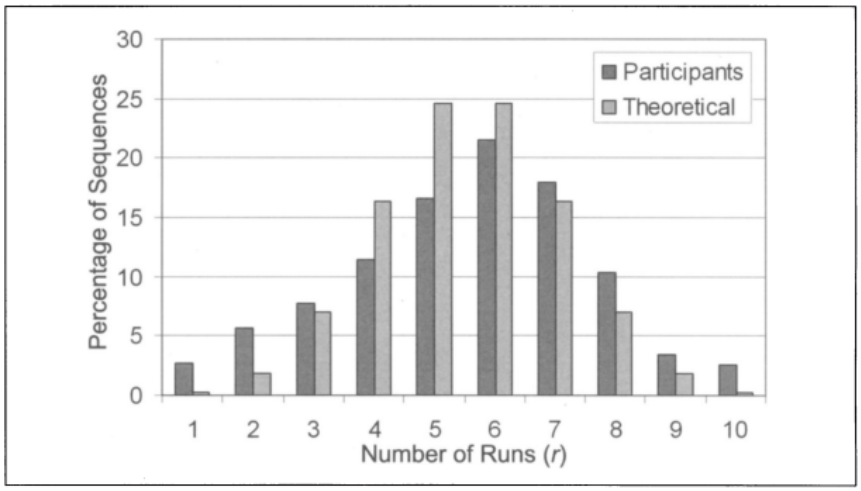

Figure 2: Percentage of all 10-toss sequences with r runs, including the theoretical distribution for comparison [12].

2.3 Prediction with Expert Advice

Having explored Perceived Randomness, which underlines the limitations of human intuition in regards to both recognising and generating random sequences, we can now extend the insights gained to the domain of prediction, particularly in scenarios where decisions must be made with some level of uncertainty. Given that the literature suggests that individuals struggle to produce random sequences, this could manifest in the form of predictable patterns that occur as a result of subconscious biases that the individuals themselves may be unaware of. To test this hypothesis, this study will make use of Prediction with Expert Advice to attempt to pre-empt what a subject is about to generate, prior to them pressing a key.

2.3.1 Introduction to On-line Prediction

Within the areas of Machine Learning and Statistics, there lies the problem of accurately "predicting future events based on past observations" [13] known as On-line Prediction. The problem involves methods in which a model makes predictions sequentially, updating its parameters in real time as new data becomes available, as shown in Protocol 1.

Protocol 1 On-line Prediction Framework

1: FOR $t = 1, 2, \ldots$
2: learner L outputs $\gamma_t \in \Gamma$
3: nature outputs $\omega_t \in \Omega$
4: learner L suffers loss $\lambda(\gamma_t, \omega_t)$
5: END FOR

There is a particular class of algorithm that is designed to solve this problem, with one of the most notable—forming the basis of this study—being the "Strong" Aggregating Algorithm proposed by Volodymyr Vovk [1]. "Strong" is emphasised with speech marks to assist in distinguishing it from the "Weak" Aggregating Algorithm proposed by Yuri Kalnishkan and Michael Vyugin [14] that will not be explore in detailed in this study.

Given that this study will primarily be investigating the domain of On-line Prediction, this subsection aims to lay a comprehensive foundation, explaining the key concepts and frameworks, to conduct an effective analysis in Chapter 4.

2.3.1.1 On-line Prediction, Batch Learning and Timeseries Analysis

To begin, let us compare and contrast On-line Prediction with alternative frameworks commonly used in Machine Learning: Batch Learning and Timeseries Analysis.

We will first examine the distinction between the On-line Prediction and Batch Learning frameworks. With Batch Learning, a whole training set of labelled examples of the form (x_i, y_i) is given to the Learner at once and used to train a model. In contrast, On-line Prediction involves gradually feeding the Learner information over time, requiring any models made to continuously adapt to the new data that it is given, whilst also requiring the Learner to take actions based on the (possibly incomplete) information it currently possesses rather than waiting for a complete picture [4]. This forced adaptability ensures that the predictions outputted by an On-line Prediction model remain accurate to the information that the model deems to be relevant and stores as it continues to gain additional information. Because of this, these models are particularly valuable in applications that require immediate responses and flexibility in predictions, such as with financial market analysis, and weather forecasting.

Secondly, we will explore the distinction between On-line Prediction and Timeseries Analysis since, while both are ways of handling sequential data, they are unique. On-line Prediction is based on processing data points se-

quentially, updating a predictive model in real time while doing so whereas Timeseries Analysis is based on modelling and forecasting data that is collected over successive time intervals. The prior approach does not impose any strict assumptions about the underlying data-generating process, even going so far as to not assume the existence of such a process [15], while the latter assumes a strictly structured approach in which the current observation is dependent on the previous observation(s). This assumption leads to the data-generating processes being modelled with stochastic models, such as *autoregressive integrated moving average (ARIMA)* or *state-space models* [16].

Contrastingly, the majority of On-line Prediction literature takes a similar stance that no assumptions can be made about the sequence of outcomes observed, therefore the analyses are done according to the worst-case scenario as a result and may, in fact, be better in reality [13].

2.3.1.2 Notation

Having defined the On-line Prediction Framework and compared it to the alternative frameworks of Batch Learning and Timeseries Analysis, we can now formalise the notation presented in Protocol 1.

Consider a scenario where the elements of a sequence, known as ***outcomes*** ω_t occur at discrete time steps $t \in T$, denoted $\omega_1, \omega_2, \ldots, \omega_T$. We assume that these outcomes are drawn from a known ***outcome space*** Ω. In this scenario, the Learner is tasked with making ***predictions*** γ_t about the outcomes prior to their occurrence and, similarly to the outcomes, we assume that the predictions are drawn from a known ***prediction space*** Γ that may or may not be the same as Ω.

After the Learner has made their prediction for the next outcome in the sequence, the true outcome is revealed, and the quality of the Learner's prediction is measured by a loss function, denoted $\lambda(\gamma_t, \omega_t)$. This function measures the discrepancy between the prediction and the outcome that is known as "***regret***", quantifying the effect of the prediction γ_t being confronted with the outcome ω_t in hindsight by mapping the input space $\Gamma \times \Omega$ to a subset of the real number line $\mathbb{R}$, typically $[0, +\infty)$ [17].

$$\lambda(\gamma_t, \omega_t) : \Gamma \times \Omega \to [0, +\infty) \tag{1}$$

Over multiple time steps, the Learner will suffer multiple Losses which is often referred to collectively as the Cumulative Loss up until time T, as shown in Equation 2. The Learner's performance is effectively measured by

this Cumulative Loss, so their natural objective is to try to minimise this value.

$$\text{Loss}_T(L) = \sum_{t=1}^{T} \lambda(\gamma_t, \omega_t) \tag{2}$$

2.3.2 Introduction to Prediction with Expert Advice

Having established the On-line Prediction Framework and its notation, we can investigate a more nuanced approach to prediction: Prediction with Expert Advice. This approach extends the framework presented in Protocol 1 by introducing a technique for leveraging multiple prediction algorithms, known as "Experts", to make more informed and accurate predictions.

Protocol 2 Prediction with Expert Advice Framework

FOR $t = 1, 2, \ldots$
    experts $\mathcal{E}_1, \ldots, \mathcal{E}_N$ output predictions$\gamma_t^1, \ldots, \gamma_t^N \in \Gamma$
    learner L outputs $\gamma_t \in \Gamma$
    nature outputs $\omega_t \in \Omega$
    experts $\mathcal{E}_1, \ldots, \mathcal{E}_N$ suffer losses $\lambda(\gamma_t^1, \omega_t), \ldots, \lambda(\gamma_t^N, \omega_t)$
    learner L suffers loss $\lambda(\gamma_t, \omega_t)$
END FOR

The theoretical foundation for this framework lies in the theory that no single Expert, no matter how knowledgeable, can consistently outperform a well-constructed aggregate of predictions from multiple Experts.

Imagine a scenario where a "Learner" now has access a pool of N Experts, denoted $\mathcal{E}_1, \ldots, \mathcal{E}_N$, that each make a prediction for the outcome of a sequence at time step $t \in T$, denoted $\gamma_t^1, \ldots, \gamma_t^N$, as shown in Protocol 2. On each time step, each Expert suffers Loss, $\text{Loss}_{\mathcal{E}_n}(t) = \lambda(\gamma_t^n, \omega_t)$. Now, suppose that each Expert's predictions are made available to the Learner prior to them making their own, denoted γ_t. A natural objective for the Learner would then be to construct a *merging strategy* that effectively combines the Experts' predictions to minimise their Cumulative Loss, $\text{Loss}_L(T) = \sum_{t=1}^{T} \lambda(\gamma_t, \omega_t)$, if good Experts are present.

In a non-adversarial scenario, we can assume that each Expert's goal is also to minimise their Cumulative Loss similar to the Learner, however, practically, we cannot safely assume that each Expert in the pool will behave

in this manner. Because of this, we must develop a framework that treats each Expert as a black box, meaning that the Learner has no knowledge of the internal prediction mechanism of each Expert nor what their goal is, to guarantee that the Cumulative Loss suffered by the Learner is almost as good as the cumulative loss suffered by the best expert.

$$\forall n, \forall T : \text{Loss}_T(L) \lesssim \text{Loss}_T(E_n) \tag{3}$$

2.3.2.1 Halving Algorithm

Before delving into the Halving Algorithm, we must first define the Simple Prediction Game which involves the prediction and outcome spaces consisting of bits, i.e., $\Gamma = \Omega = \{0, 1\}$ and the Loss Function defined as:

$$\lambda(\gamma_t, \omega_t) = \begin{cases} 0 & \text{if } \gamma_t = \omega_t, \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

Intuitively, this means that the Learner's Cumulative Loss corresponds to the number of mistakes that they have made over T steps and, as before, we want to minimise this value. Now, suppose that we know in advance that among the pool of Experts, there is a perfect Expert that never makes mistakes. Intuitively, this means that their loss will always be zero because they never make a mistake. Despite knowing that the perfect Expert will never make a mistake, this does not guarantee that the Learner will not because they do not know which of the N Experts is the correct one to listen to. Because of this, a strategy must be devised to find them while minimising the Cumulative Loss suffered as a result and this comes in the form of Majority Voting wherein the Learner actively follows the majority of Experts by mimicking their vote. This strategy works because we know in advance that there is a perfect Expert and, thus, do not need to pay attention to any Experts that have made a mistake. To do this, the Learner must maintain two lists: a ***whitelist*** and a ***blocklist***. Initially, all N Experts begin in the whitelist and make predictions as written previously, however, as soon as an Expert makes a mistake, i.e., $\gamma_y^n \neq \omega_t$, they are moved to the blocklist with no way of returning. As a result, we only consider the predictions from Experts that are currently memos of the whitelist on each time step t.

By following this method, every time step t that the Learner makes a mistake, the size of the whitelist is at least halved from the size of the previous time step $t-1$ since the majority ($\geq 50\%$) of Experts also made a mistake and get moved to the blocklist accordingly.

Given that the size of the whitelist on time step t is denoted W_t, this is formally written as $W_t \leq {}^{W_{t-1}}/_{2}$, with $W_0 = N$. Generally, this means that if by time T, the Learner has made m mistakes, then $W_T \leq {}^{W_0}/_{2^m}$ and is at least Size 1 due to the presence of at least one perfect Expert.

$$1 \leq W_T \leq \frac{W_0}{2^m} = \frac{N}{2^m} \tag{5}$$

By rearranging this formula, a Learner following the Halving Algorithm for the Simple Prediction Game will satisfy the following inequality for every sequence of outcomes, provided that there is a perfect Expert.

$$\text{Loss}_L(T) \leq \lfloor \log_2 N \rfloor \tag{6}$$

2.3.2.2 Weighted Majority Algorithm

We now look at an extension to the Halving Algorithm that is still played with $\Gamma = \Omega = \{0, 1\}$ and

$$\lambda(\gamma_t, \omega_t) = \begin{cases} 0 & \text{if } \gamma_t = \omega_t, \\ 1 & \text{otherwise} \end{cases}$$

but removes the restrictive requirement that there be a perfect Expert. To do so, the Learner needs maintains a vector of weights for each expert at each time step, denoted as w_t^n which is a measure of the Learner's trust in each Expert.

Initially, each expert begins with the same weight, $w_0^n = 1$, but instead of moving Experts from the whitelist to the blocklist whenever they make a mistake, the Learner updates the respective expert's weight as follows.

$$w_t^n = w_{t-1}^n \beta^{\lambda(\gamma_t^n, \omega_t)} \tag{7}$$

In this equation, $\beta < 1$ is a parameter of the algorithm that can be written as $\beta = e^{-\eta}$, where $\eta > 0$ is referred to as the learning rate as it controls how severely the trust in an Expert is eroded upon them making a mistake. Conversely, if an Expert does not make a mistake, then $\lambda(\gamma_t^n, \omega_t) = 0$ and $\beta^{\lambda(\gamma_t^n, \omega_t)} = 1$. With this notation, it can be seen how the Halving Algorithm is a special case of the Weighted Majority Algorithm where $\beta = e^{-\eta} = 0$ as the trust in an Expert gets reduced to zero when they make a mistake.

The Weighted Majority Algorithm can be represented with the following algorithm:

Algorithm 1 Weighted Majority Algorithm

initialise weights $w_0^i = 1, i = 1, 2, \ldots, N$
FOR $t = 1, 2, \ldots$
    read the experts' predictions $\gamma_t^i, i = 1, 2, \ldots, N$
    calculate the sum of weights $v_t^0 = \sum_{n:\gamma_t^n=0} w_{t-1}^n$
        and $v_t^1 = \sum_{n:\gamma_t^n=1} w_{t-1}^n$
    if $v_t^0 > v_t^1$, predict $\gamma_t = 0$; otherwise predict $\gamma_t = 1$
    observe the outcome ω_t
    update the experts' weights $w_t^i = w_{t-1}^i \beta^{\lambda(\gamma_t^i, \omega_t)}, i = 1, 2, \ldots, N$
END FOR

Given Algorithm 1, we can determine the upper bound for the Weighted Majority Algorithm as follows. Notice that the bound is in terms of β, this is simply due to formatting. The proof for this bound is more complicated than that of the Halving Algorithm and will, therefore, not be discussed in this study but can be found in detail within [18].

$$\text{Loss}_L(T) \leq \frac{\ln(\frac{1}{\beta})}{\ln(\frac{2}{1+\beta})}\text{Loss}_{\mathcal{E}_i}(T) + \frac{\ln(N)}{\ln(\frac{2}{1+\beta})} \tag{8}$$

Alternatively, this upper bound has a simpler notation of the form:

$$\text{Loss}_L(T) \leq c(\beta)\text{Loss}_{E_n}(T) + \alpha(\beta)\ln N \tag{9}$$

Where $c(\beta)$ and $\alpha(\beta)$ are coefficients that depend on the parameter $\beta = e^{-\eta}$, with $\eta > 0$.

In the next subsection, we will explore the Aggregating Algorithm, the final method of merging Experts' predictions and a concrete implementation of the Prediction with Expert Advice that formalises how a Learner's predictions are made as accurate as possible.

2.3.3 Games and Mixability

Previously, we have referred to the simple prediction game which is defined as $\Gamma = \Omega = \{0, 1\}$ and

$$\lambda(\gamma_t, \omega_t) = \begin{cases} 0 & \text{if } \gamma_t = \omega_t, \\ 1 & \text{otherwise} \end{cases}$$

However the term "game" can be generalised to the triple $\langle \Gamma, \Omega, \lambda \rangle$. This triple refers to a specific prediction space, outcome space, and loss function.

Informally, this triple is referred to as a game because it encapsulates the interactive, yet adversarial, nature of the problem as a result of the conflicting goals of the Learner and Nature which closely resembles the ***Repeated Game Framework*** discussed in Game Theory [19].

For our purposes, the Learner must perform sequential decision-making and must output a prediction $\gamma \in \Gamma$ without knowing the true outcome $\omega \in \Omega$ in advance. This highlights the imbalance in the game and the potential for adversarial games to be played. In these scenarios, Nature "has it out" for the Learner and seeks to inflict as much loss as possible by selecting ωs far from γ. The Learner's goal remains the same in trying to minimise the loss that they suffer over time and, therefore, must develop strategies to optimise their performance based on the information available to them, and the actions they have observed. This study will primarily look at the discrete binary game, where $\gamma = [0, 1]$, $\omega = \{0, 1\}$ with the square-loss function $\lambda(\gamma, \omega) = (\omega - \gamma)^2$. This is an example of an η-mixable game.

A game $\langle \Gamma, \Omega, \lambda \rangle$ is called η-mixable if the following condition holds:

For any N predictions, $\gamma^1, \ldots, \gamma^N$, and any probability distribution, $p^1, \ldots, p^N$, there exists a constant $\eta > 0$ and an aggregated ("mixed") prediction $\gamma \in \Gamma$ such that, for all $\omega \in \Omega$,

For any N predictions, $\gamma^1, \ldots, \gamma^N$, and any probability distribution, $p^1, \ldots, p^N$, there exists a constant $\eta > 0$ and an aggregated ("mixed") prediction $\gamma \in \Gamma$ such that, for all $\omega \in \Omega$,

$$\lambda(\gamma_t, \omega_t) \leq -\frac{1}{\eta} \sum_{n=1}^{N} p_{t-1}^n e^{-\eta \lambda(\gamma_t^n, \omega_t)} \tag{10}$$

Where $p_{t-1}^n = \frac{1}{N} e^{-\eta \text{Loss}_{\mathcal{E}_i}(t-1)} / \sum_{n=1}^{N} \frac{1}{N} e^{-\eta \text{Loss}_{\mathcal{E}_i}(t-1)}$

The significance of this equation is that it allows the Learner to effectively combine the predictions from multiple sources, ensuring that the aggregated prediction suffers a loss that is close, if not equal, to the best possible weighted combination of individual predictions. Ultimately, this means that mixable prediction strategies can compete with, or outperform, any fixed prediction strategy in hindsight.

Having discussed Games and Mixability, we can move on to discussing the Aggregating Algorithm which aims to leverage the strength of each Expert's predictions to make the Learner's own more accurate while also minimising the impact of inaccurate predictions. It does so by hedging so that, if adversarial experts are present, the Learner wouldn't suffer a large cumulative loss due to poor reliability.

2.3.4 Aggregating Algorithm (AA)

Having introduced the concept of mixability and the discrete binary game, we can now explore the Aggregating Algorithm that forms the basis of this study. As noted in the previous Subsection, this game allows for predictions to be made from the prediction space $\Gamma = [0, 1]$, rather than $\Gamma = \{0, 1\}$.

Similarly to the Weighted Majority Algorithm, this algorithm maintains a list of weights corresponding to the Learner's confidence in the accuracy of each Expert's Prediction. However, unlike the Weighted Majority Algorithm, the weights are normalised on every time step in order to prevent them from becoming too small.

In order to maintain the mixability condition discussed in Subsection 2.3.3, a mixability constant $C(\eta)$ that is associated with the learning rate η is introduced. The constant determines how effectively each of the Experts' predictions can be combined, specifically in the inequality that bounds the Learner's loss.

As this study is concerned with the *discrete binary game* which has been proven to be mixable in [15], [14], and [20], we can take $C(\eta) = 1$ such that:

$$\lambda(\gamma, \omega) \leq -\frac{1}{\eta} \ln \sum_{n=1}^{N} p^n e^{-\eta\lambda(\gamma^n, \omega)} \tag{11}$$

and

$$e^{-\eta \mathrm{Loss}_L(T)} \geq \sum_{n=1}^{N} \frac{1}{N} e^{-\eta \mathrm{Loss}_{\mathcal{E}_i}(T)} \tag{12}$$

Finally, by dropping all terms except the n^{th} from the sum and taking the logarithm, we get the following inequality:

$$\mathrm{Loss}_L(T) \leq \mathrm{Loss}_{\mathcal{E}_i}(T) + \frac{1}{\eta} \ln \frac{1}{q_i} \tag{13}$$

where q_i is an arbitrary weight set when initialising the experts.

The Aggregating Algorithm is formalised on the following page.

Algorithm 2 Aggregating Algorithm (AA)

initialise weights $w_0^i = q_i, i = 1, 2, \ldots, N$
FOR $t = 1, 2, \ldots$
    read the experts' predictions $\gamma_t^i, i = 1, 2, \ldots, N$
    normalise the experts' weights $p_{t-1}^i = w_{t-1}^i / \sum_{j=1}^N w_{t-1}^j$
    output $\gamma_t \in \Gamma$ that satisfies the inequality for all $\omega \in \Omega$:
        $\lambda(\gamma_t, \omega) \leq -\frac{C}{\eta} \ln \sum_{i=1}^N p_{t-1}^i e^{-\eta\lambda(\gamma_t^i, \omega)}$
    observe the outcome ω_t
    update the experts' weights $w_t^i = w_{t-1}^i e^{-\eta\lambda(\gamma_t^i, \omega_t)}, i = 1, 2, \ldots, N$
END FOR

2.3.5 Aggregating Algorithm for Specialist Experts (AASE)

While the Aggregating Algorithm provides a robust framework for incorporating Expert advice, certain scenarios require a more nuanced approach. This is where the Aggregating Algorithm for Specialist Experts comes into play, which will be the approach that this study's experiment is centred on.

The use of the term 'Specialist' was first introduced by the work of Avrim Blum [21] for the Winnow and Weighted-Majority algorithms, and can be thought of as a natural extension to traditional Experts insofar as it enables these 'Specialists' to abstain from making a prediction "when the current Expert does not fall into their '[speciality]''. While the criteria for an Expert to abstain from making a prediction is sufficient in our context, it can also be extended to allow for other scenarios like those suggested in [20], namely if "a prediction algorithm [sees] that its internal confidence is low and [decides] to skip a turn in order to re-train" or if a prediction algorithm breaks down, as would be the case if a regression algorithm "[has] its matrix very close to singular."

To accommodate these Specialist Experts, the Prediction with Expert Advice Framework given in Protocol 1 has to be modified:

Protocol 3 Modified Prediction with Expert Advice Framework

FOR $t = 1, 2, \ldots$
    nature chooses a subset of experts $\mathcal{E}_i \in \mathcal{E}$ that are awake
    awake experts $\mathcal{E}_1, \ldots, \mathcal{E}_N$ output predictions $\gamma_t^1, \ldots, \gamma_t^N \in \Gamma$
    learner L outputs $\gamma_t \in \Gamma$
    nature outputs $\omega_t \in \Omega$
    awake experts $\mathcal{E}_1, \ldots, \mathcal{E}_N \in \mathcal{E}_i$ suffer losses $\lambda(\gamma_t^1, \omega_t), \ldots, \lambda(\gamma_t^N, \omega_t)$
    learner L and sleeping experts $\mathcal{E}_j \notin \mathcal{E}_i$ suffers loss $\lambda(\gamma_t, \omega_t)$
END FOR

As referenced above, another colloquial way of referring to 'Specialist Experts' is 'sleeping Experts'; Freund postulated that "a Specialist is awake when it makes a prediction and that it is asleep otherwise", going so far as to refer to the traditional On-line Prediction framework as "the insomniac framework since it is a special case in which all Specialists are awake all the time." [3] This colloquialism is useful when adapting the bounds of the base Aggregating Algorithm because a natural interpretation of what happens when an Expert is sleeping is that it simply "joins the crowd" [20], meaning that it mimics the learner's prediction on the time steps that it is asleep because the learner's prediction is formed based on the weighted majority of Experts' predictions. Given this definition, it can be seen that on some time steps t, the learner's prediction and the Expert $\mathcal{E}_i$'s predictions are the same; $\gamma_t = \gamma_t^i$. Recall that, in the mixable case, the Aggregating Algorithm guarantees that the following inequality is satisfied:

$$\sum_{t=1}^{T} \lambda(\gamma_t, \omega_t) \leq \sum_{t=1}^{T} \lambda(\gamma_t^i, \omega_t) + \frac{1}{\eta} \ln \frac{1}{q_i} \tag{14}$$

Typically, the Aggregating Algorithm's performance is measured in terms of the learner's cumulative loss compared to the best Expert's cumulative loss but given that, on certain time steps t, $gamma_t = \gamma_t^i$, it is clear that the corresponding terms in both sums cancel out and what is left are the sums over the time steps where the learner's and the Expert's predictions are different, i.e. where Expert $\mathcal{E}_i$ is awake. What follows from this is that, instead of wanting the learner's loss to be nearly as good as the best Expert's loss over a period of time T, we judge the AASE's performance based on the learner's loss compared to the best Expert's $\mathcal{E}_i$ loss over the steps in which it was awake. A learner following the algorithm achieves a cumulative loss that satisfies the following inequality:

Algorithm 3 Aggregating Algorithm for Specialist Experts (AASE)

initialise weights $w_0^i = q_i, i = 1, 2, \ldots, N$
FOR $t = 1, 2, \ldots$
    read the awake experts' predictions $\gamma_t^i, i = 1, 2, \ldots, N$
    normalise the awake experts' weights
        $p_{t-1}^i = w_{t-1}^i / \sum_{j:\mathcal{E}_j \text{ is awake}} w_{t-1}^j$
    output $\gamma_t \in \Gamma$ that satisfies the inequality for all $\omega \in \Omega$:
        $\lambda(\gamma_t, \omega) \le -\frac{C}{\eta} \ln \sum_{i:\mathcal{E}_i \text{ is awake}} p_{t-1}^i e^{-\eta\lambda(\gamma_t^i, \omega)}$
    observe the outcome ω_t
    update the awake experts' weights $w_t^i = w_{t-1}^i e^{-\eta\lambda(\gamma_t^i, \omega_t)}$
    update the sleeping experts' weights $w_t^i = w_{t-1}^i e^{-\eta\lambda(\gamma_t, \omega_t)/C(\eta)}$
END FOR

$$\sum_{\substack{t=1,2,\ldots,T: \\ \mathcal{E}_i \text{ is awake} \\ \text{on step } t}} \lambda(\gamma_t, \omega_t) \le \sum_{\substack{t=1,2,\ldots,T: \\ \mathcal{E}_i \text{ is awake} \\ \text{on step } t}} \lambda(\gamma_t^i, \omega_t) + \frac{1}{\eta} \ln \frac{1}{q_i} \tag{15}$$

As is the case for the traditional Aggregating Algorithm, we make no assumptions about the outcome-generating mechanism (including the existence of such a mechanism) and this bound holds for *any* adversarial strategy, meaning that the adversary cannot inflict a large loss on the learner without inflicting a large loss on the Specialists and ensuring that the performance will be good whenever there is a good mixture of Specialists.

2.4 Conclusion

Having gone through the established knowledge base on these topics, this Literature Review has now traced how the perception of randomness in both judging and generating random sequences has changed over time, highlighting key studies that have informed the current understanding of these cognitive processes with the conflicting findings highlighting the complexity of the task. Ultimately, the disagreement suggests that while humans may not be inherently good randomisers individually when compared to the statistical definition of random, their performance can vary significantly depending on the conditions in which they are tested with an interesting observation being made in that when sequences generated by multiple individuals are aggregated, the distribution becomes more like what is expected of a truly random process.

In the context of Prediction with Expert Advice, these insights are particularly interesting because of the imperfections of human-generated "random" sequences, primarily marked by the tendency to favour sequences with a greater number of alternations, or that contain longer runs than what is expected, might be more predictable than realised by making use of On-line Prediction algorithms, providing evidence for the fact that humans are, in fact, bad randomisers.

Having contextualised the experiment being conducted in this study with the frameworks established by the various papers cited above, this Literature Review sets the for the the subsequent analysis of human-generated "random" sequences. The findings from the experiment will assist in providing a deeper understanding of the human perception of randomness, and the potential implications that may have on predictive computational models.

3 Experiment Design and Methodology

3.1 Introduction

As discussed in Subsection 2.2.2.2, the experimental design used in this study closely aligns with the methodology established by [12], serving as the foundational basis for this experiment. By adapting their established methodology to the current study and its novel application, this project not only compares the sequences generated by subjects with those expected from a random process according to the statistical definition but also evaluates the predictability of each subject's responses using Prediction with Expert Advice and Vovk's Aggregating Algorithm [1]. In theory, the more random a subject's sequence, the less predictable their responses, leading to greater losses for both the Learner and each of the relevant Experts.

3.2 Experimental Design

This section outlines the experimental design, which aims to assess how well individuals can generate random binary sequences when compared to theoretical randomness and their previous inputs. The study employs a within-subject design, where each participant is exposed to all conditions of the independent variable which, in his case, means they are required to repeat the generation process several times. This approach provides a comprehensive view of each individual's performance in generating random sequences.

The independent variable in this experiment is the method by which the participants generate their binary sequences, as each participant is allowed to enter their sequences independently of one another. The dependent variables include the frequencies of 0s and 1s in each sequence, the number and length of runs within each sequence, and the predictions generated by the Experts and the Learner, which will be discussed in further detail in the following sections. The control variables of this study include the instructions given to the subject before beginning the experiment, the length of each binary sequence inputted, and the total number of sequences entered. These control variables were devised to facilitate a more accurate comparison and analysis of the collected data between participants.

This study's subjects primarily consisted of postgraduate students from the Computer Science Department at Royal Holloway, University of London, with additional participants from the English Department to create a more representative sample. Each sample was tasked with generating several 10-

item sequences intended to mimic the results expected from a random process. These sequences were entered into a web application hosted on GitHub Pages. Each 10-item sequence consisted of 0s and 1s (representing Heads and Tails) arranged in any order that the subject chose, and participants were allowed to enter the sequences at their own pace. Before beginning the experiment, subjects were presented with the following instructions on a modal screen shown upon loading the web page:

> *Your task is to create a table of sequences each consisting of 10 items, either 0 or 1.*
>
> *Imagine that several people have each tossed a fair coin 10 times and the results of their tosses are recorded in a table, with each row recording the outcomes of the 10 tosses by one person.*
>
> *Your goal is to produce this table in such a way that if compared with a table of the results of actual coin tosses, it would not be possible to distinguish which table represented the actual coin tosses with statistical tests and which didn't.*

Herein lies the first divergence from Nickerson and Butler's original design because the sequences entered by the subjects are always displayed and were concatenated into a single, continuous sequence (as shown in Figure 3), which is then passed to the Aggregating Algorithm as ωs. In the original experiment, the sequence would only remain visible to the subject until they had entered 10 items, at which point it would disappear. The modification in this study allows the Aggregating Algorithm to better identify patterns in the user's inputs as an interval length of 10 bits would be insufficient in allowing the algorithm to determine which Experts should be given higher weighting in forming the Learner's predictions, thereby improving the learner's prediction accuracy. While the underlying algorithm treats the sequence differently to [12], the sequences are still presented to the subject in 10-item chunks (as shown in Figure 4), consistent with the original method, to better align with the human short-term memory span of approximately 7 ± 2 items cited in Subsection 2.2.2.2. This chunking allows subjects to quickly review their previous inputs and continue generating sequences that they perceive as random.

Given this foundation, we can now discuss how the Aggregating Algorithm for Specialist Experts (AASE) was applied to this experiment.

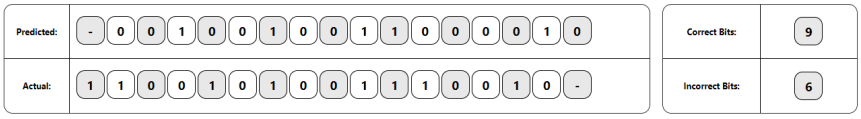

Figure 3: Inputted Sequence Displayed in the Web Application

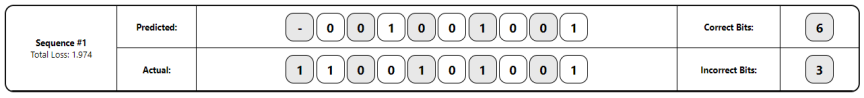

Figure 4: Past Sequences Displayed in the Web Application.

3.3 Applying the Aggregating Algorithm

The Aggregating Algorithm as well as the broader framework of Prediction with Expert Advice, are central to this study. To apply these methods to this scenario, we first define the η-mixable game $G = \langle \Gamma, \Omega, \lambda \rangle$, as defined in Protocol 2. The primary focus of this project, as suggested by Chapter 2, is Prediction with Expert Advice for the Discrete Binary Game. Formally, this game is defined by the ***outcome space*** $\Omega = \{0, 1\}$, the ***prediction space*** $\Gamma = [0, 1]$, and the ***loss function*** given by Brier's (or Square) Loss, $\lambda_{\mathrm{SQ}}(\gamma_t, \omega_t) = (\gamma_t - \omega_t)^2$. In practical terms, this means that Nature generates binary outcomes, either 0 or 1, while both the Learner and the Experts predict values within the range $[0, 1]$.

With the game formally established, it is essential to define the roles of Nature, the Learner and the Experts within our experimental context. In this project, which evaluates a subject's ability to generate random binary sequences, one of the key metrics is the predictability of their inputs before pressing a key. Therefore, each subject assumes the role of Nature, producing the sequence. The Aggregating Algorithm functions as the Learner attempting to pre-empt Nature. As the experiment involves multiple subjects, this supports the premise that "we make no assumptions about the outcome-generating mechanism (including the existence of such a mechanism)." Each subject possesses a unique internal concept of randomness, and the Aggregating Algorithm must generalise across all participants and all sequences.

Next, we define the concept of an Expert, as well as the rationale for making use of Specialist Experts and the AASE. For our purposes, an Expert can be thought of as a function designed to predict the next outcome in a sequence based on the presence of a specific scenario. Since the experiment

evaluates each subject's concept of randomness by statistically analysing conditional probabilities for different orders of dependency, it is natural to conceptualise the group of Experts as functions that search for specific prefixes within the sequence. As the subject inputs their sequence to the application, the last n bits are passed to the group of Experts, who then assess whether the sequence matches their prefix, signifying their "area of expertise".

This approach draws inspiration from the concept of Markov Chains, which are mathematical models that describe sequences of events where the probability of each event depends solely on the current state. In our context, the subject's sequence of bits can be viewed as a Markov Chain, where the last n bits represent the current state, with the Experts estimating the subject's underlying transition rules—the hypothetical process by which they "decide" on the next bit—based on observed patterns.

However, unlike a fixed Markov Chain, where the depth of the chain is predefined, we do not know how deep each subject's decision-making process is. Specialist Experts address this problem by focusing on different prefix lengths: a subset of experts search for shorter prefixes, while others may look for longer ones. This variety allows the Aggregating Algorithm to adaptively estimate the length of each subject's Markov Chain, even if it varies over time. Although this study does not delve into the full theory of Markov Chains, this framework offers a valuable perspective on how humans might generate binary sequences. For readers familiar with Markov Chains, this connection highlights how the experiment models human decision-making as a probabilistic process with potentially varying levels of complexity.

Given that not every prefix will be relevant to each new subject input, the use of Specialist Experts is justified. If the current sequence does not match a Specialist's prefix (i.e., their "area of expertise"), that Specialist is considered "asleep" and abstains from making a prediction, effectively "joining the crowd". Conversely, Specialists whose prefixes match the sequence's ending are considered "awake" and make predictions accordingly. Notably, there will never be a scenario where all Specialists are asleep—at least one is awake at all times, as there is always an Expert searching for the last bit of the sequence.

Finally, the details of how an Expert functions within this experiment must be explained. To generate the Experts, binary sequences up to length 4 were generated and assigned to individual Experts. The decision to limit the prefix length to 4 bits balances computational efficiency with predictive accuracy, reflecting the established cognitive constraints of human short-term memory, typically spanning 7 ± 2 items. With a different implementation, a

higher prefix length could be considered. Each Expert tracks the frequency of 0s and 1s following their specific prefix and makes predictions based on the ratio $\#1/(\#0 + \#1)$. As subjects input their sequences, each Expert checks the last x bits (corresponding to the length of their prefix) to determine whether they are awake. If awake, the Specialist makes a prediction, which is then fed into the Aggregating Algorithm to inform the Learner's prediction for the next time step. For transparency, each Expert's last prediction, current prediction and status are displayed at the bottom of the application for the subject's reference.

3.4 Data Analysis

After the experiment was conducted, the data was aggregated and analysed using various methods. The primary method was a chi-square goodness-of-fit test, comparing each subject's generated sequences to the distribution expected from a random process. To evaluate the Learner's performance, a secondary analysis was conducted, comparing both the cast (rounded) and uncast (unrounded) predictions against the actual outcomes.

Given the experiment involves playing Brier's Game with $\Omega = \{0, 1\}$ and $\Gamma = [0, 1]$, the simplest strategy for the Learner would be to predict $1/2$ for every time step, which is the minimax prediction, resulting in a loss of $1/4$ each time. Over 10 steps, the maximum loss for the Learner would be $10/4 = 2.5$, which serves as a benchmark for assessing the Aggregating Algorithm's performance – loss less than 2.5 indicates that a Learner using the Aggregating Algorithm predicts better than this naive strategy.

These methods of analysis are, in fact, somewhat adversarial, as the more statistically random a subject's input is, the more challenging it should become for the Experts, and thus the Learner, to make accurate predictions. Consequently, a better fit to statistical randomness should result in poorer Learner performance as the subject's sequences should exhibit no discernable patterns.

3.5 Procedure

This chapter provides an outline of the procedure followed to conduct the experiment as a summary of what was outlined previously.

1. Participants were selected from the student community at Royal Holloway, University of London. Those recruited included postgraduate students from the Computer Science Department, as well as students from the English Department, to create a diverse sample.

2. Participants were directed to the application hosted on GitHub Pages designed to collect and analyse generated binary sequences. Its development and the issues faced will be outlined in Chapter 5.

3. During the experiment, subjects were tasked with generating several 10-item binary sequences by entering 0s and 1s into the application such that the results would appear random if subjected to statistical tests. For transparency, the application displayed the internal workings of each Expert at the bottom of the screen, though participants could disable this feature if they desired.

4. After finishing the experiment, the subjects were asked to send their results to the researcher to be subjected to the methods outlined in the previous section, namely chi-square goodness-of-fit and comparison to the loss inflicted by following the naive strategy.

4 Analysis of Perceived Randomness

As mentioned in Subsection 3.4, the analysis of this study's findings is performed using two methods: chi-square goodness-of-fit, and by comparing the differences in the cumulative loss of the Learner L and each Expert $\mathcal{E}_i \in \{\mathcal{E}_1, \ldots, \mathcal{E}_N\}$. With these methods, this Chapter delves into the results to provide evidence for the hypothesised question "Are humans good randomisers?"

4.1 Chi-Square Goodness-of-Fit

We begin with the Chi-Squared (χ^2) Goodness-of-Fit, a statistical method for determining if observed results are similar to what is expected from the null hypothesis—that humans are good randomisers.

4.1.1 Distribution of the Number of Heads

Figure 5 shows that the distribution produced by the subjects is qualitatively similar to the theoretical distribution in that it is bell-shaped, however the results of $\chi^2(10, N = 175) = 155.14, p = 3.26\text{e}{-28} < 0.0001$ strongly indicate that the binomial distribution is not a good fit, with large discrepancies found at either extreme, and for sequences with 7+ heads. As noted in [12], "in a randomly produced set of toss sequences, about 25% of the sequences should have five heads and about 65% should have four to six heads." In this study's sequences, 34% had exactly five, and 73% had four to six heads.

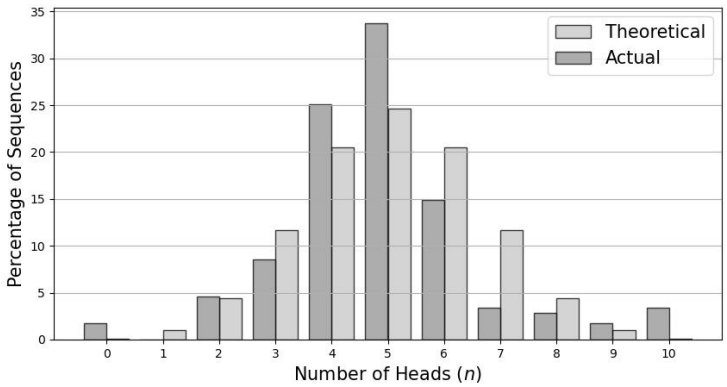

Figure 5: Percentage of 10-Item Sequences with n Heads, Compared to the Theoretical Distribution, $X \sim B(10, 0.5)$

Overall, these findings are more than significant enough to reject the null hypothesis and support the conclusion noted by Nickerson and Butler in that "when trying to emulate a random generator of binary sequences, people are likely to produce a larger percentage of sequences that contain the two elements (e.g., heads and tails) in approximately equal proportions than is a random process." [12]

4.1.2 Distribution of the Number of Runs

Unlike with the Distribution of the Number of Heads, the distribution produced by the subjects shown in Figure 6 is both qualitatively and quantitatively different to what was expected, $\chi^2(9, N = 175) = 908.46, p = 9.29\text{e}{-190} < 0.0001$. The produced distribution does not have a discernable bell shape, and instead, seems to steadily rise to $r = 8$ before significantly dropping off.

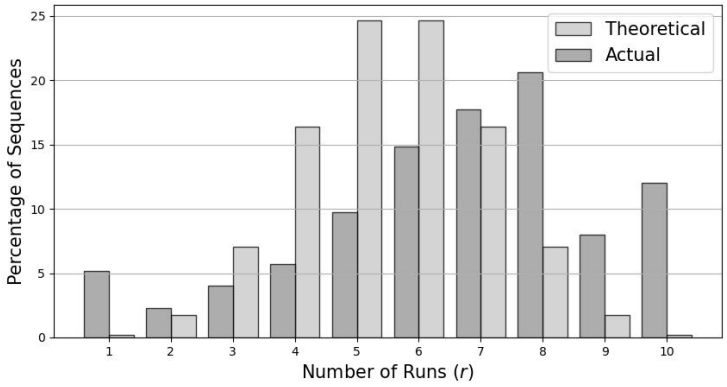

Figure 6: Percentage of 10-Item Sequences with r Runs, Compared to the Theoretical Distribution

Despite looking different to that produced by Nickerson and Butler, this data still supports their conclusion that "people are likely to fail when trying to emulate a random process – in this case by being less likely than a random process to produce sequences with an intermediate number of runs and more likely to produce sequences with close to the minimum or maximum number possible" [12]. The discrepancies that follow the peak at $r = 8$ also support Bar-Hillel and Wagenaar's conclusion that "humans produce series with higher than expected alternation rates" [9].

4.1.3 Distribution of Run Lengths

As previously defined, a *run* is a consecutive sequence of the same outcome (either all heads or all tails), meaning that a run length of 1 occurs when only a single head or tail appears before the outcome changes. While the distribution produced by subjects appears qualitatively similar to the theoretical distribution in that the frequency of run lengths decreases exponentially, a significant percentage of runs more than expected had a length of 1, while all other run lengths were underrepresented in the data, $\chi^2(9, N = 1, 160) = 33.25, p = 0.0001$.

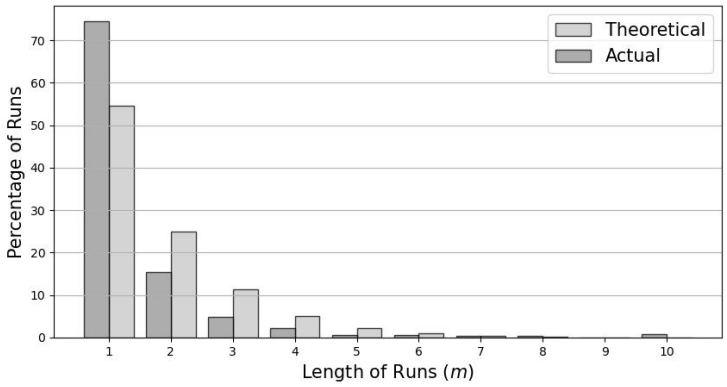

Figure 7: Percentage of Runs with Length m, Compared to the Theoretical Distribution

Once again, this supports the conclusion drawn by [9] in that human-generated sequences favour alternation over continuation with 90% of runs across all sequences being up to length 2, and 97% being up to length 4.

4.2 Regret Analysis

Given the evidence shown in Subsection 4.1 supporting the conclusion that humans are not good randomisers, we now delve into an evaluation of the Aggregating Algorithm's performance at being able to predict the next bit of human-generated binary sequence.

As previously discussed, a Learner makes use of the predictions from several Experts to form their own about the next bit in a sequence entered by subjects after the fact (acting as Nature in this context). Continuing to assume the null hypothesis that humans are good randomisers, the Aggre-

gating Algorithm should perform no better than randomly guessing the next bit of a sequence, i.e., having a 50% success rate, however, the plot below shows that this is not the case for any of the subjects tested.

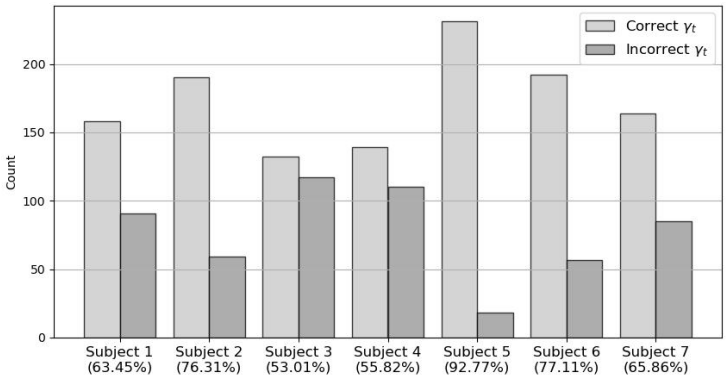

Figure 8: Correct vs. Incorrect Predictions for Each Subject

The performance of the Aggregating Algorithm exceeded that of random guessing across all subjects, although the improvement was only marginal for Subjects 3 and 4. On average, the Aggregating Algorithm achieved an accuracy of 69.19%, reinforcing the notion that human-generated sequences exhibit predictable patterns, even when only considering prefixes up to length 4.

4.2.1 Statistical Significance of the Aggregating Algorithm's Performance

To further investigate whether the Aggregating Algorithm performed significantly better than random guessing, a binomial significance test was conducted. The null hypothesis of the test presumes that the Aggregating Algorithm performs no better than random guessing, i.e., a success rate of 50%—the probability of correctly predicting the next bit in the sequence by chance. The test was applied to the results from the seven subjects individually and in aggregate, using a significance level of $\alpha = 0.01$ to provide a comprehensive assessment of the algorithm's performance.

Individually, the results reveal that the Aggregating Algorithm performed significantly better than random guessing for five out of the seven subjects. For Subjects 1, 2,5, 6, and 7, the p-values were well below the significance threshold, with Subjects 5 and 6 having exceptionally low p-

values of 1.35e−48 and 1.40e−18 respectively. These values strongly indicate that the algorithm's predictions are extremely unlikely to be due to random chance. Additionally, Subjects 1, 2, and 7 had p-values of 1.30e−5, 1.53e−17, and 3.13e−7 respectively, further reinforcing the robustness of the algorithm's predictive performance. With a more lenient threshold of $\alpha = 0.05$, Subject 5's p-value of 0.00379 would have also been significant. Conversely, Subjects 3 and 4 did not achieve statistically significant results, with p-values of 0.1875 and 0.0379 respectively, furthering the notion that individuals have different perceptions of randomness.

When analysing the results in aggregate, the algorithm's performance also remained significantly better than random guessing, with $P(X \geq 1206) < 0.00000001$. This indicates that observing 1,206 or more successes out of 1,743 trials by random chance alone is virtually impossible, ultimately emphasising the predictability of human-generated sequences and highlighting the strength of the Aggregating Algorithm in detecting the subconscious patterns of such sequences.

4.2.2 Comparative Analysis of Cumulative Losses

An examination of the difference in cumulative losses between individual Experts and the Learner across all time steps reveals patterns in the data. When the difference between an Expert's and the Learner's cumulative loss is positive (i.e., the line is above the x-axis), it indicates that the Expert's predictions are less accurate than those of the aggregated model. This implies that the Learner is more effectively aggregating the given predictions, and suggests that the subject is less predictable when inputting a certain prefix, as the relevant Expert's predictions are less accurate.

Conversely, when the difference is negative (i.e., the line is below the x-axis), the individual Expert's predictions outperform the Learner's aggregate. This scenario implies that the Learner's aggregation is less effective and that the subject's behaviour is more predictable for a given prefix as the relevant Expert's predictions are more accurate.

Reviewing the difference plots for all participants, shown below, it is noteworthy that only the Experts designed to detect a Markov Chain of Length 1—a first-order Markov Chain—displayed a difference greater than 2. This observation supports the existing literature: humans tend to avoid immediate repetition and exhibit subconscious longer-term patterns and dependencies that a first-order Markov Chain cannot capture. This is due to inherent memory constraints. For example, if a participant alternates between bits every two or three time steps, a first-order Markov Chain will be

unable to detect the pattern and cause inaccurate predictions,

Furthermore, the results indicate that as a participant's sequences display more qualitative similarities to true randomness, the Aggregating Algorithm's predictive accuracy decreases. This is reflected in a greater proportion of lines closer to the x-axis, as well as alternating signs, signifying an increase in randomness due to the appearance of more inconsistent patterns.

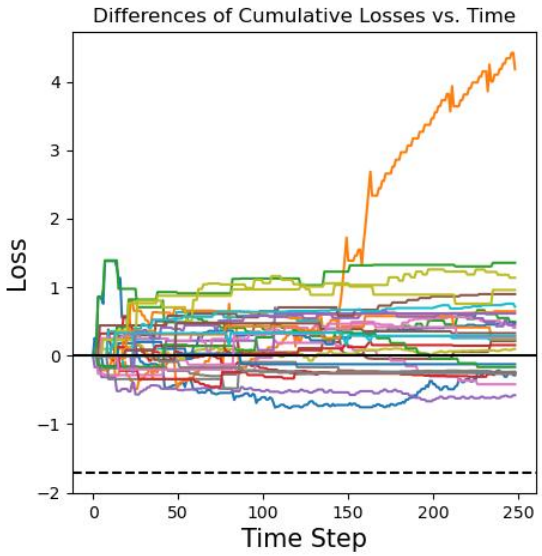

Figure 9: Cumulative Loss Differences: Experts vs. Learner for Subject 1

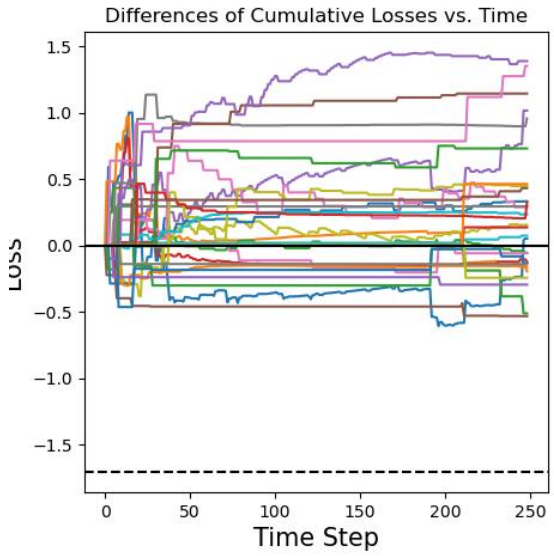

Figure 10: Cumulative Loss Differences: Experts vs. Learner for Subject 2

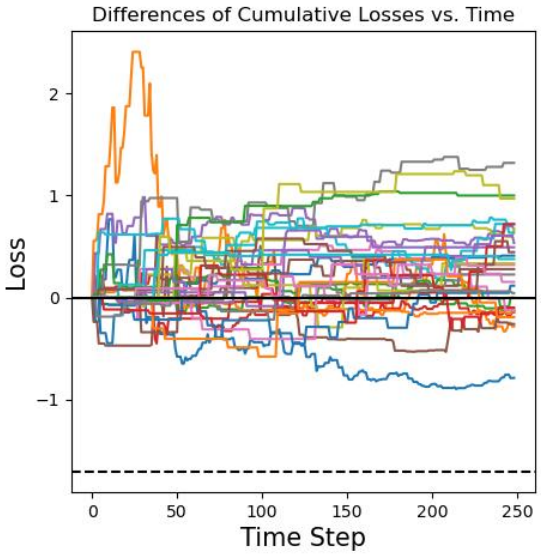

Figure 11: Cumulative Loss Differences: Experts vs. Learner for Subject 3

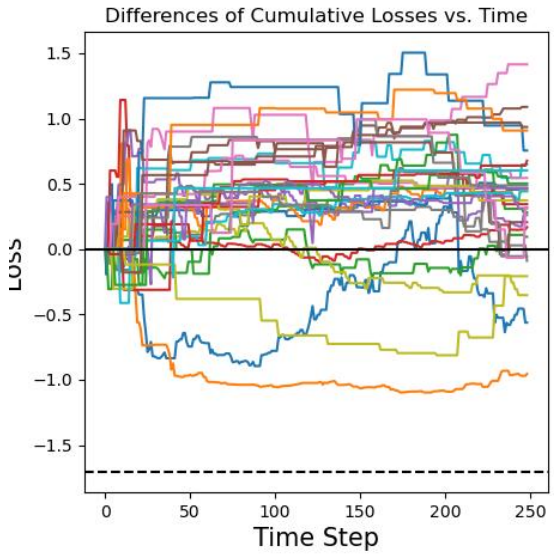

Figure 12: Cumulative Loss Differences: Experts vs. Learner for Subject 4

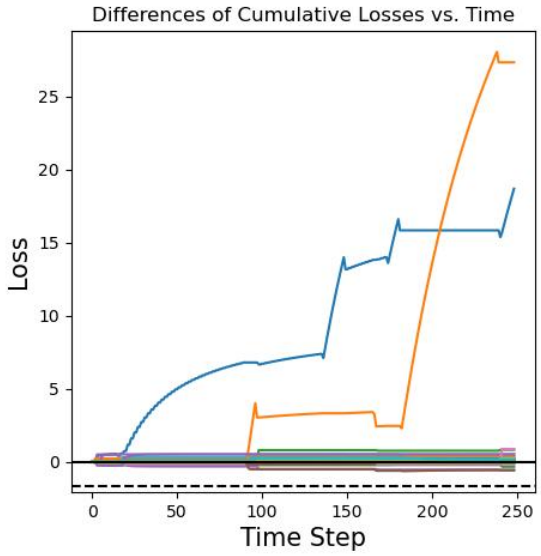

Figure 13: Cumulative Loss Differences: Experts vs. Learner for Subject 5

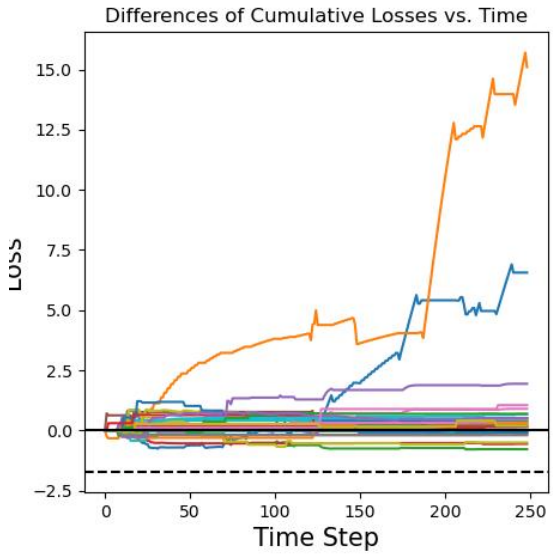

Figure 14: Cumulative Loss Differences: Experts vs. Learner for Subject 6

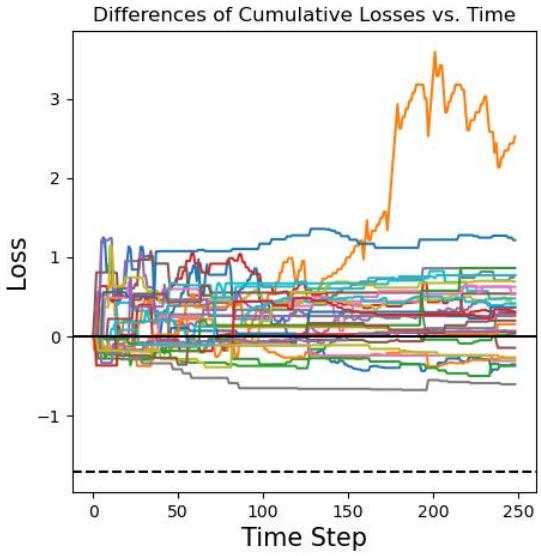

Figure 15: Cumulative Loss Differences: Experts vs. Learner for Subject 7

5 Conclusion

5.1 Summary of Findings

This study aimed to use the Aggregating Algorithm for Specialist Experts to provide a novel insight into the well-documented understanding that humans have difficulty in producing truly random sequences due to cognitive biases and subconscious patterns.

The chi-square tests detailed in Section 4.1 consistently rejected the null hypothesis that humans are good randomisers, indicating deviations from objective randomness in line with previous research [12]. The plots of difference further confirmed this, achieving a prediction accuracy of 69.19% on average and implying predictability. This was reflected in the plots consistently below the x-axis, where an individual Expert's predictions were consistently more accurate than the Learner's.

While the individual data deviated from what was expected of true randomness, when the participants' data was aggregated, it showed qualitative similarities. While these similarities do not equate to true randomness, the aggregate tended towards the distribution of a random process, confirming that by aggregating data, the effects of individual biases become increasingly smaller. Ultimately, these results align with the Cognitive Psychology literature, reinforcing the idea that humans struggle to produce truly random outputs, and contribute to the body of research illustrating that human-generated sequences are often more predictable than expected, despite a conscious effort to be random.

5.2 Limitations

While this study's findings are significant, certain limitations must be acknowledged and addressed in future research.

The first limitation relates to computational power; The AASE implementation was restricted to analysing prefixes up to length 4 as a compromise between computational efficiency and predictive performance. However, considering longer prefixes—extending to the limits of short-term memory—could uncover more intricate and nuanced patterns in human-generated sequences that were previously undetectable.

A second limitation is the small sample size. With only seven participants, individual data heavily influenced the overall results, which limits the study's generalisability. A larger, more diverse sample would minimise this effect and offer a broader evaluation of human randomisation behaviour.

Additionally, future work could extend the Game to consider a *prediction* and *outcome space* that encompasses the entire alphabet, rather than just binary sequences. This would allow the exploration of more complex randomisation tasks, such as the ability to generate random passwords. This could have significant implications for cybersecurity, exposing the weaknesses of human-generated randomness in contexts where security is essential.

5.3 How to Use the Project

As mentioned in Subsection 3.5, the application is hosted on GitHub Pages, however, it can be run locally from `./aggregating_algorithm/` with:

```
npm run start
```

To use the project, navigate to the website and follow the instructions provided on-screen to conduct the experiment. Once the experiment has concluded, the JSON will be available to download and stored within the directory `./aggregating_algorithm/results/data`. Once moved, one can run the Python script for generating the plots shown in this report by executing the following command from `./aggregating_algorithm/results/`:

```
python3 generate_plots.py
```

5.4 Self-Assessment

This dissertation has been an incredible journey and I am incredibly proud of the result. Not only has it deepened my understanding of the Aggregating Algorithm, but it has also taught me how to manage a personal project. Overall, I can confidently say that the project was a success, both in achieving the objectives outlined in Chapter 1 and in preparing for my career.

This is not to say that the project was not without its challenges. While I was familiar with Python, developing a web application with React.js was a new experience and required more time to learn than originally anticipated which caused delays in my project's timeline that could have been avoided with more cautious planning.

Despite this, the experience has significantly boosted my confidence as a future Machine Learning Engineer. I have learned the value of setting realistic deadlines, how to effectively use online resources, and how to stay current with research—all invaluable skills in the future. Looking ahead, I plan to explore the **Mixture of Experts (MoE)** ensemble technique as it is conceptually similar to Prediction with Expert Advice, with applications in neural networks which are extremely relevant currently.

References

[1] V. Vovk, "Aggregating strategies," in *Colt Proceedings 1990*, pp. 371–383, San Francisco: Morgan Kaufmann, 1990.

[2] V. Vovk, "A game of prediction with expert advice," *Journal of Computer and System Sciences*, vol. 56, no. 2, pp. 153–173, 1998.

[3] Y.Freund, R. Schapire, Y. Singer, and M. Warmuth, "Using and combining predictors that specialize," *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, 01 1997.

[4] Y. Kalnishkan, D. Adamskiy, A. Chernov, and T. Scarfe, "Specialist experts for prediction with side information," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pp. 1470–1477, 2015.

[5] H. Reichenbach, *The Theory of Probability.* Berkeley: University of California Press, 1949.

[6] B. M. Ross, "Randomization of a binary series," *The American journal of psychology*, vol. 68, no. 1, pp. 136–138, 1955.

[7] P. Bakan, "Response-tendencies in attempts to generate random binary series," *The American journal of psychology*, vol. 73, no. 1, pp. 127–131, 1960.

[8] W. Wagenaar, "Appreciation of conditional probabilities in binary sequences," *Acta Psychologica*, vol. 34, pp. 348–356, 1970.

[9] M. Bar-Hillel and W. A. Wagenaar, "The perception of randomness," *Advances in applied mathematics*, vol. 12, no. 4, pp. 428–454, 1991.

[10] L. L. Lopes and G. C. Oden, "Distinguishing between random and nonrandom events.," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 13, no. 3, p. 392, 1987.

[11] M. Kubovy and D. Gilden, "More random than random - a study of scaling noises," in *Bulletin of the Psychonomic Society*, vol. 26, pp. 494–494, Psychonomic Soc Inc 1710 Fortview Rd, Austin, TX 787704, 1988.

[12] R. S. Nickerson and S. F. Butler, "On producing random binary sequences," *The American Journal of Psychology*, vol. 122, no. 2, pp. 141–151, 2009.

[13] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth, "How to use expert advice," *J. ACM*, vol. 44, p. 427–485, may 1997.

[14] Y. Kalnishkan and M. Vyugin, "The weak aggregating algorithm and weak mixability," *Journal of Computer and System Sciences*, vol. 74, no. 8, pp. 1228–1244, 2008. Learning Theory 2005.

[15] V. Vovk, "Competitive on-line statistics," *International Statistical Review*, vol. 69, 2001.

[16] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[17] Y. Kalnishkan, "The aggregating algorithm and laissez-faire investment," Tech. Rep. CLRC-TR-09-02, Royal Holloway, University of London, 2009.

[18] N. Littlestone and M. Warmuth, "The weighted majority algorithm," *Information and Computation*, vol. 108, no. 2, pp. 212–261, 1994.

[19] J.-F. Mertens, "Repeated games," in *Game Theory and Applications* (T. Ichiishi, A. Neyman, and Y. Tauman, eds.), Economic Theory, Econometrics, and Mathematical Economics, pp. 77–130, San Diego: Academic Press, 1990.

[20] Y. Kalnishkan, "Prediction with expert advice for a finite number of experts: A practical introduction," *Pattern Recognition*, vol. 126, p. 108557, 2022.

[21] A. Blum, "Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain," *Machine Learning*, vol. 26, pp. 5–23, 1997.