Effect of the Side Effect Machines in Edit Metric Decoding

Sharnendu Banik

Submitted in partial fulfilment
of the requirements for the degree of

Master of Science

Department of Computer Science
Brock University
St. Catharines, Ontario

To my parents, without whom I would not be here today.

Abstract

The development of general edit metric decoders is a challenging problem, especially with the inclusion of additional biological restrictions that can occur in DNA error correcting codes. Side effect machines (SEMs), an extension of finite state machines, can provide efficient decoding algorithms for such edit metric codes. However, finding a good machine poses its own set of challenges and is itself considered as an open problem with no general solution. Previous studies utilizing evolutionary computation techniques, such as genetic algorithms and evolutionary programming to search for good SEMs have found success in terms of decoding accuracy. However, they all worked with extremely constricted problem spaces i.e. a single code or codes of the same length. Therefore a general approach that works well across codes of different lengths is yet to be formalized.

In this research, several codes of varying lengths are used to study the effectiveness of evolutionary programming (EP) as a general approach for finding efficient edit metric decoders. Two classification methods — direct and fuzzy — are compared while also changing some of the EP settings to observe how the decoding accuracy is affected. The final SEMs are verified against an additional dataset to test their general effectiveness. Regardless of the code length, the best results are found using the fuzzy classification methods. For codes of length 10, a maximum accuracy of up to 99.4% is achieved for distance 1 whereas distance 2 and 3 achieve up to 97.1% and 85.9%, respectively. Unsurprisingly, the accuracy suffers for longer codes, as the maximum accuracies achieved by codes of length 14 were 92.4%, 85.7% and 69.2% for distance 1, 2, and 3 respectively. Additionally, the machines are examined for potential bloat by comparing the number of visited states against the number of total states. The study has found some machines with at least one unvisited state. The bloat is seen more in larger machines than it is in smaller machines. Furthermore, the results are analyzed to find potential trends and relationships among the parameters. The trend that is most consistently noticed is that — when allowed, the longer codes generally show a propensity for larger machines.

Acknowledgements

First and foremost, I would like to express my sincerest gratitude to my supervisor Dr. Sheridan Houghten. Her continuous support and inspiring guidance throughout the course of my graduate study ensured the completion of this dissertation.

I would also like to thank the committee members, Dr. Brian Ross, Dr. Robson De Grande for reviewing the draft of this thesis and providing me with invaluable comments and suggestions.

I would like to show my appreciation to Dr. Daniel Ashlock for providing the edit metric codes that were integral to this research. Without his support, this study would not have been possible. I would like to thank Tyler Kennedy Collins for providing me the previous materials of this research.

I would also like to acknowledge the support provided by Compute Canada for granting me permission to run the extremely resource hungry experiments on their computing clusters.

Finally, I would like to convey my deepest gratitude towards my family - my parents Rekha and Ram Krishna Banik for all the sacrifices they made in their lives to get me where I am today, my other parents Ava and Monir Hossain for their unconditional love, my elder brother Sharnav for his guidance, my sister-in-law Srabani for her endless support and my nieces Aurora and Arya for their endless love and joy. Lastly, my special thanks go to Basanti for her consistent support and love. I will forever be in their debt and I am proud to dedicate this dissertation to them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

The constant improvement of computational technologies has paved the way for new disciplines, such as bioinformatics that has permitted in-depth analyses of massive amounts of crude biological data that otherwise would have remained untouched. One of those areas is the study of the genome, in particular, the base pair sequencing of the DNA strands. An array of tools and methods have been developed for identifying genetic markers - DNA sequences that uniquely identify an organism or a trait. However, corruption of these markers is a common phenomenon that occurs due to a variety of reasons. These errors need to be detected and corrected in order to identify the original sequence. Unfortunately, the current set of tools used in biological applications is extremely limited in terms of their ability to correct such errors.

Decoding is a well-known problem across several scientific disciplines. However, the inclusion of the biological restrictions makes it a particularly challenging problem in biological applications. Looking on the bright side, several new methods have been proposed by researchers in the last few years. One of them, the *side effect machine* (SEM), used in conjunction with *evolutionary programming* (EP), has shown promising results in terms of decoding accuracy. However, the results are far from perfect as there remain many gaps in fully understanding the nuances that are involved in developing such a solution.

This study aims to fill some of these gaps and establish a better realization of some of the key parameters involved, in the process potentially improving the accuracy of the decoder. Overall, the goal of this study is to contribute to the ongoing research of developing decoders for biological applications.

1.2 Problem statement

The most common types of errors observed in a DNA sequence are caused by insertion, deletion, and substitution of the base pairs. While the Hamming metric is well suited for detecting substitution errors, it is not useful for decoding errors caused by insertion or deletion of symbols. These errors are best understood using *Levenshtein distance*, also known as *edit distance*. However, the cost of computing edit distance is much higher than it is for *Hamming distance* — $O(n^2)$ vs $O(n)$ respectively. This high run time complexity leads to poor decoding performance and renders it unsuitable for use in real-world biological applications. To minimize this heavy cost, a general decoder using SEMs, that allows linear-time decoding, was proposed in [11]. Having said that, finding an efficient SEM is a very difficult problem in its own right. Previous studies [14, 34, 31] have seen success by applying various evolutionary algorithms to heuristically find effective machines. Even so, due to the large size of the problem space and the probabilistic nature of the process, the fitness of the machines often saturate at a local maximum and an optimal machine is never found. Overall, the problem of finding efficient decoders for an edit metric code is still an open problem with no general solution.

This research aims to contribute by studying a wider range of codes than what was done by previous studies [11, 14, 34, 31]. The goal is to observe the effectiveness of evolutionary programming on these new codes to determine its merit as a generalized edit metric decoder. Furthermore, this study will also perform in-depth analyses of the structures of the successful SEMs, especially with respect to the number of states and their connectivity to accurately identify and measure possible bloat in the SEMs.

1.3 Organization of the thesis

This thesis is organized as follows:

Chapter 2 reviews key concepts related to error correction in DNA sequences. It introduces the reader to the general concepts of errors in data communication and their relevance in biological contexts. It starts by giving a brief overview of errors in data communication and discusses the common methods used today for correction. Some of these concepts include Hamming distance, edit distance, *error correcting code*, and DNA structure.

Chapter 3 reviews relevant past work in relation to the creation of edit metric codes and their use in biological applications. It also gives a brief overview of code creation methods proposed by previous studies, such as comma free code, marker code, and watermark code. Finally, the chapter reviews the method of using SEMs as edit metric decoders and discusses previous approaches for developing and implementing such decoders for DNA error correction.

Chapter 4 provides detailed descriptions of the SEM and its key characteristics. It explains the process of decoding using an SEM and discusses its merits and demerits against other methods of edit metric decoding. It discusses upon the difficulties involved in developing SEMs to work as a general decoder. Finally, it concludes by explaining the optimization techniques and trade-offs associated with using SEMs as general decoders.

Chapter 5 discusses the general concept of *evolutionary algorithms* (EAs) and their application for solving hard optimization problems. It also gives a brief overview of the key features of EAs such as solution representation, initialization, fitness, selection, and genetic operators. After briefly talking about the different types of EAs, it provides an in-depth look at EP as this is the technique used to generate SEMs in this thesis. Finally, the chapter reviews how the different operations of EP are tweaked and tuned for developing SEMs.

Chapter 6 discusses two methodologies — direct classification and fuzzy classification — that are used for decoding error patterns. It describes how the two datasets, training and verification, are generated for this study. Then, it shows the different parameter values, used in direct classification technique to generate SEMs. Later, the fuzzy classification with a tolerance value is implemented to improve the decoding capability of the generated machines.

Chapter 7 shows the results of two different methodologies, direct classification and fuzzy classification, to find the error correction accuracy over nine different codes with different parameter settings. It also analyzes the different aspects of different parameters to find out the relationships between them.

Chapter 8 gives a summary of the methods used with different parameter settings and suggests future work to improve decoding using side effect machines.

Chapter 2

Background

This chapter discusses the key concepts related to error correction. It reviews the general concepts of errors in data communication and their relevance in biological contexts.

2.1 Error in Data Communication

Data transmitted over most communication channels are subject to electrical or electromagnetic noise and other impairments and, as a result, are prone to corruption. Data is considered corrupted when it has undergone unwanted modifications during the course of transmission. In the field of data communication, such modifications are called *errors*. Errors not only degrade the quality of communication but, depending on the amount, can modify the sender's data to such an extent that no meaningful information can be retrieved from it by the receiver. Therefore, detection and correction of errors are of paramount importance in ensuring reliable communication.

2.2 Classification of Errors

Data errors are classified into two types - *single bit error* and *burst error*. A single bit error, as the name suggests, occurs when only one bit of the data unit has changed from 1 to 0 or 0 to 1. Figure 2.1 shows how a single bit error can occur in a noisy communication channel. On the other hand, when two or more bits in sequence have been modified, it is called a burst error.

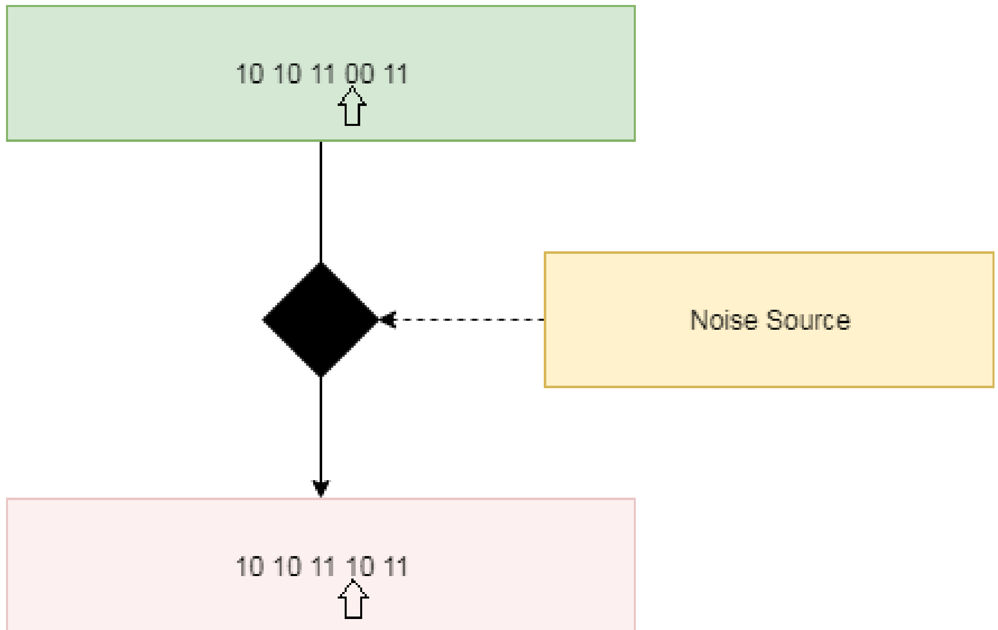

Figure 2.1: Single bit error in data communication

2.3 Error Detection

An error in a discrete signal can be represented as the difference between the original message and the received message. The difference can be calculated using a bitwise exclusive or (XOR) logical operation [41] that outputs true or 1 when the two bits differ and false or 0 otherwise. However, the lengths of the two messages have to be equal to perform this operation. Table 2.1 shows an example of how the XOR operation is used to detect any mismatch between two same-length patterns. If 10010011 is sent through a noisy channel and 10110011 is received by the receiver, the error that occurred during transmission can be represented as the vector [00100000]. This vector also provides information on the number of symbols that do not match, and it is called the Hamming distance between the two words.

	Message
Sent pattern	10010011
Received pattern	10110011
XOR	00100000

Table 2.1: Error detection using bitwise XOR operation

The Hamming distance is the number of unmatched symbols between two words of the same length. It was named after Richard Hamming, an American mathematician, who introduced the idea in his paper on error detection and correction in 1950 [29]. The Hamming distance between two strings, 01010101 and 10101010, is 8 as both strings have eight characters and every character from one string is different from the corresponding character in the other.

2.3.1 Error Detection Using Redundancy

A non-zero Hamming distance between two words indicates a mismatch which can be used to detect a transmission error. However, this can only work if the receiver knows the original message that was sent. In reality, the original message is unknown. The receiver has no point of reference against which it can compare the received message in order to calculate the Hamming distance. This problem is solved by transforming the original message using an algorithm before sending it through the channel. The process usually increases the length of the original message without adding new information to its content, which is why it is considered redundant information. If the message is altered during transmission, it will no longer conform to the same algorithm and the receiver detects it as an error. Redundancy is the central concept behind all error detection schemes used in modern communication.

2.4 Error Detection vs Error Correction

Error detection is not to be confused with error correction. In general, error detection schemes are simpler and are not designed to perform the correction. They are commonly used in digital data communication where the sender can be requested to repeat the message. A few examples of such error detection schemes are parity bits, checksums, and cyclic redundancy check (CRC). Error correction schemes, on the other hand, need to both detect and correct the error without intervention from the sender. This is achieved by using an error correcting code.

2.5 Error Correcting Code (ECC)

An error correcting code (ECC) transforms a sequence of data such that any errors introduced to any of the data in the said sequence can be detected and corrected to a certain extent. Each of the original strings is referred to as a *codeword* and the set of all codewords is called a *code*.

Mathematically, an error correcting code is denoted as $(n, M, d)_q$ where:

- n = the length of the codewords

- d = the minimum distance by which the codewords are separated
- M = the total number of codewords the code contains
- q = the number of symbols that occur in the codewords, e.g. binary codewords consist of two symbols, 0 and 1. Hence, $q = 2$.

It is worth mentioning that M may not always be the *optimal* value for a code. A code is optimal for a specific value of length n and minimum distance d if and only if it has the highest possible number of codewords [22].

Error correcting codes have the ability of finding and correcting errors. When a word is received, it is assumed that the codeword closest to the received word is the original word. Therefore, if a received word completely matches with a codeword, there is no error. If not, the closest codeword replaces the received word to decode the error. This process of decoding is called *maximum-likelihood decoding*. However, there is an error correction bound of a code depending on the radius of the sphere of each codeword. Figure 2.2 shows the bounds of two codewords A and B of radius r. When a word, *E1*, is received, it finds its closest codeword, B. Then, it checks if *E1* is inside the sphere of B (the distance between *E1* and B is less than radius r). As it is true here, the error message *E1* can be corrected by replacing it with codeword B. However, error message *E2* is outside of the spheres of both A and B, and so will not be corrected to either A or B.

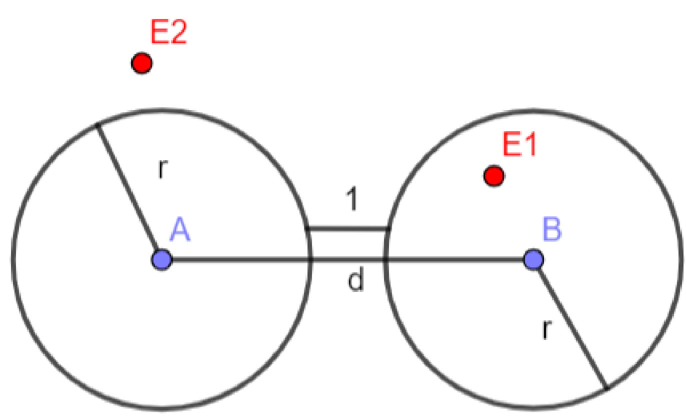

Figure 2.2: View of the Sphere Correction Bounds of Codewords A and B.

Mathematically, an error correcting code can correct up to t errors. The value of t depends on the minimum distance d between each codeword, where $t = \lfloor (d-1)/2 \rfloor$ [33, 43]. For example, with minimum distance 5, an error correcting code can decode error patterns that have up to 2 errors. If an error pattern has more than 2 errors, the code is unable to correct it as the received word gets closer to another codeword than the original word. Similarly, there are some error patterns which are of equal distance from two or more codewords, in which cases the correction becomes ambiguous [12]. Therefore, the codewords are created well separated from each other. The implementation of such codes is practical and effective in a lot of applications involving error correction.

2.6 The Biological Context

Often in science two seemingly unrelated disciplines find a common problem of interest. Coding theory and bioinformatics have found one such common problem in error correction. This has turned out to be a huge challenge in the field of genomics, in particular with respect to DNA sequencing.

2.6.1 DNA Structure

DNA, Deoxyribonucleic acid, is built with two strands of nucleotide molecules running in opposite directions and circling each other forming a double helix. The most important components of a nucleotide are a phosphate group, a sugar, and a nitrogen base. There are four types of nitrogen bases as shown in Figure 2.3. These are adenine (A), thymine (T), guanine (G), and cytosine (C). A and G are examples of purine and C and T are examples of pyrimidine. Because of the chemical structure, A can form two hydrogen bonds with T and G, can form three hydrogen bonds with C [35]. These bases create bonds to form the double helix formation of DNA and the codes are sequenced along with it. As each base can only bond with a specific base partner (A pairs with T and G pairs with C), it is called complementary base pairing. The sequence of these bases determines the genetic instructions encoded in the proteins that determine key characteristics of every living organism.

2.6.2 DNA Sequence

The order in which the nucleotides appear in DNA to create the double helix formation is commonly referred to as the DNA sequence. The nitrogenous bases are used to characterize the nucleotides as they are the only components that differ in them.

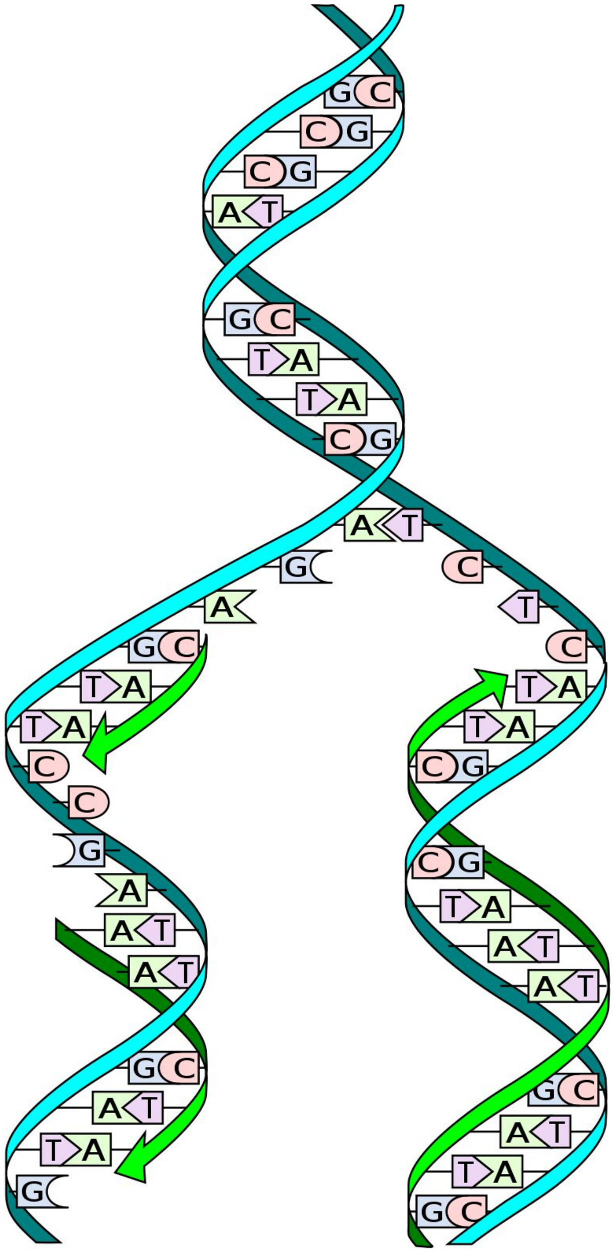

Figure 2.3: DNA structure [8]

2.7 Errors in DNA sequences

Over the years there have been several methods of sequencing DNA. The most modern techniques examine the fluorescent-dye intensity signal generated by automatic sequencing machines to determine the nitrogen bases [46]. However, this process is prone to errors and the sequence obtained by it is not entirely trustworthy. The most common sequencing errors can be classified into the following three basic categories:

1. Insertion: Occurs when a base is wrongly identified in a place where there is none. E.g. AATCAAG in place of AATCAG.
2. Deletion: Occurs when a base is not identified in a place where there is one. E.g. ATCAG in place of AATCAG.
3. Substitution: Occurs when the wrong based type is identified. E.g. ATTCAG in place of AATCAG.

2.7.1 Error correction in a DNA Sequence

Upon observation, it can be seen that the sequencing errors mentioned earlier are fundamentally similar to errors encountered during transmission over a noisy communication channel. This makes the correction schemes discussed in coding theory applicable in the field of genomics.

While the Hamming distance is a decent choice for correcting substitution errors, it is not useful for detecting insertions and deletions. As discussed earlier, the Hamming distance is a measure of unmatched symbols between two codewords of the same length in which only substitution errors are expected. It does not work in situations where insertions and deletions may occur. A different measure called the edit distance must be used in order to identify such cases.

2.7.2 Edit Distance

The edit distance or Levenshtein distance [37] quantifies dissimilarity between two words by counting the minimum number of operations to change one word to another, where the operations are insertions, deletions, and substitutions of symbols. Therefore, the edit distance varies from the Hamming distance. Algorithm 1 shows the procedure to find the edit distance between two strings, which is reproduced from [44].

The algorithm takes two stings, x and y of length n and m respectively and returns the edit distance between them. The problem can also be expressed as finding the minimum operations required to convert x into y and vice versa. The algorithm uses dynamic programming to break up this problem into smaller sub-problems where every sub-problem deals with finding the minimum number of operations required to make a sub-string of x equal to a sub-string of y. This is done by creating a matrix of $n \times m$ where x and y are constructed bottom-up, i.e. from a null string to their complete forms and each cell stores the edit distance between the respective sub-strings up to that point. For a given cell, the three adjacent cells to its upper left — diagonally, vertically and horizontally — can be used to represent substitution, deletion and insertion operations respectively. Hence the distance at any cell can be expressed as the minimum value of the three neighbouring cells plus the cost of converting itself. Using this principle, the algorithm populates the matrix at a run-time cost of $O(n^2)$ and eventually the bottom-right cell returns the edit distance between x and y.

```
Input: Two Strings: x = [x1, x2, ..., xn] and y = [y1, y2, ..., ym]
Output: Edit distance between the strings
int d[0, ..., n][0, ..., m];
for i = 0 to n do
    d[i][0] = i;
end
for j = 0 to m do
    d[0][j] = j;
end
for i = 1 to n do
    for j = 1 to m do
        if x[i] = y[j] then
            cost = 0;
        else
            cost = 1;
        end
        d[i][j] = MIN(d[i − 1][j] + 1, d[i][j − 1] + 1, d[i − l][j − 1] + cost);
    end
end
return d[n][m]
```

Algorithm 1: Dynamic programming algorithm for calculating edit distance [44]

Table 2.2 shows an example of distance measurement between two words. The Hamming distance is 8 as it requires 8 substitutions to convert one word to another. However, the edit distance is 2 as the former can be transformed into the latter by removing the 0 from the beginning (most significant bit) and inserting a 0 at the end (least significant bit). Therefore, codes formed with edit distance are better choice than those with Hamming distance for use in bioinformatics applications because they can correct insertions and deletions along with substitutions, which are all common in sequencing.

Word 1	01010101
Word 2	10101010
Hamming distance	8
Edit distance	2

Table 2.2: Difference between Hamming distance and edit distance of two words

Codes generated using the edit distance as the minimum distance between codewords are called edit metric codes. These codes are particularly suitable for genomic applications due to their ability to account for insertions and deletions as well as substitutions.

2.8 DNA Error Correcting Code

Sequence tags[1] are relatively short DNA sequences which provide identifying information about an organism. These tags are unique and easily detectable in the genome by the polymerase chain reaction (PCR). Therefore, they serve as important elements in a genetic construct. Incidentally, the process of sequencing a genetic construct is prone to error. Errors, such as misreading a base, skipping a base, reading a base that is not in the sequence tags are common in biological applications. However, if the tags are stored well separated from one another, they can be used as codewords to design an edit metric code called the DNA error correcting code that can correct such errors. This code with parameters $(n, M, d)_q$ should have a value of 4 for q, as a DNA sequence is constructed with 4 symbols A, T, C, and G. Construction and decoding of DNA error correcting codes are further discussed in the following chapter.

Chapter 3

Literature Review

This chapter first discusses the general code creation techniques. Then, it discusses some advanced techniques to decode codes. It also talks about the constraints in codes and why general techniques struggle in biological problems. Finally, it shows the previous approaches that use SEMs in edit metric decoding.

3.1 Construction of Codes

Error correcting codes may be generated using Conway's lexicode algorithm [16]. This is a greedy algorithm that creates a code $C(n, d)$ by examining each possible codeword of length n in an ascending lexicographical order and selecting those that have a minimum distance of d from all existing codewords in C. The algorithm begins by initializing C as an empty list and continues by populating it with compatible codewords, i.e. codewords satisfying the aforementioned distance rule. Conway's lexicode algorithm was originally defined with Hamming distance, but edit distance can also be used. It is a slow process as all possible codewords are examined.

In [2] Ashlock made small modifications in the original Conway's lexicode algorithm to create edit metric lexicodes. The goal was to construct DNA codes with a maximum number of codewords. These codes, which may have biological restrictions, were later used as embeddable markers for *cDNA libraries*. An evolutionary algorithm named the *Greedy Closure Evolutionary Algorithm* was used to change the order in which the words were selected. Initially, three random seed codewords were placed in an empty set of codewords to create a parent. The seed codewords maintained the predefined minimum distance between each other. Later, binary genetic operators made a comparison between the seeds of two parents for reproduction. The children were created by first selecting the common seed words from the parents and later randomly distributing the remaining words in them. Afterwards, the code was created by Conway's lexicode algorithm using the seed as a starting point. Fitness was measured for each new child based on the size of the code. If any child violated the minimum distance rule, it was given a fitness value of zero and ignored for further selection. This procedure continued for a predefined number of generations and produced the codes with a maximum number of codewords where each codeword maintained the minimum distance between one another. It was found that in terms of the code size, the greedy fitness evolutionary algorithm [2] performed better than the unmodified lexicode algorithm.

Houghten et al. [32] used a variation of Conway's algorithm to optimize the process of generating edit codes. The method creates a new child code by mixing two known parent codes and appending a new random codeword at the end. Conway's lexicode algorithm is then used on the child code to filter out incompatible codewords. Compared to conventional applications of Conway's lexicode algorithm, this method reduces the computational complexity of generating edit codes and allows codes with longer codewords to be built much faster, albeit not without a trade off i.e. the codes generated in this way are usually smaller.

Ashlock et al. [4] concluded that mutation was a more effective reproduction method than crossover for finding codes. Crossover is overly aggressive in eliminating weak candidates and thus converges to good solutions too quickly. As a result, the population loses its diversity fairly early in the evolution process and begins to produce children identical to the parents. Mutation, on the other hand, allows weaker children to be produced from fitter parents adding more diversity to the population as well as slowing down the rate of convergence to the final solution.

The problem of creating error correcting codes is a well-studied one. Previous studies [5, 3] examined a variety of approaches to create codes with as many codewords as possible. In [5] four different algorithms were attempted for synthesizing error correcting codes over the DNA alphabet. In the end, although the *salmon algorithm* enhanced the performance of the shorter codes capable of correcting single errors and the ES-algorithm provided improvements to the medium-length codes, the most significant improvements came due to the continuous advancements of computer hardware, as was forecasted by *Moore's law*. In [3] three different types of evolutionary algorithms were considered to improve the upper bound of nine DNA error correcting codes using a *ring optimizer* and a *hybridizing evolutionary algorithm*. The hybridizer started with the output of the ring optimizer, that already increased the size bounds of two codes, and managed to increase it further for four more codes. Overall the study managed to push the boundaries of the table of known best code sizes to distance 13 and was able to correct errors with up to 6 edits in a DNA marker of length 14 or more. It also anticipated further improvements with more powerful computers using the same approach.

3.2 Decoding of Codes

Given a corrupted word, decoding is the process of finding the correct codeword from an ECC. The process of decoding may vary based on ECC properties which are usually dictated by applications.

A code is called *comma-free* when no predefined symbol, such as a “comma”, is required to separate its codewords from one another. It was first introduced by Crick et al. [17] in 1957. The code consists of non-overlapping codewords with distinct starts and ends so that they can be distinguished from one another without having to place separators to mark their boundaries. This allows the decoders to catch errors fast and regain synchronization. However, a major drawback of comma-free codes is that they are unable to correct insertion and deletion errors.

Sellers proposed *marker codes* [40] to identify and correct insertion and deletion errors in the edit metric. The code is, in fact, a concatenation of two codes — an inner code that detects errors and an outer code that corrects them. It works by appending to each codeword a unique marker sequence that the outer burst-error-coding code can examine to look for errors. The sequence acts as a synchronization mechanism and allows the code to detect insertion and deletion errors between markers. The error correction capability is proportional to the length of the marker sequence i.e. the longer the sequence, the more errors it is able to correct. However, it introduces redundancy, which reduces the overall throughput of the actual data and limits the rate at which information can be sent.

Watermark codes, introduced by Davey el al. [19] and further studied by Ratzer and MacKay in [39] are similar to marker codes in that they are also concatenated codes that rely upon an inner code to detect insertion and deletion errors and an outer code that is used for correcting substitution errors. A known watermark sequence is added to each codeword. The idea is akin to writing on a sheet of paper that has a watermark on it, where the integrity of the written data can be determined by inspecting the watermark for morphological changes. First, the codewords are examined by the inner code against a known watermark to check for insertion and deletion errors. Once the locations of the errors are identified, and the insertions removed, the codeword is left with nothing but substitution and deletion errors. However, these deletion errors can be interpreted as substitutions by null and be treated as substitution errors. This leaves substitution error as the only type of error remaining in the codewords enabling Hamming distance to be used for their correction. At this point, the codewords are sent to the outer code which is designed to correct these remaining errors.

However, these codes are not suitable in DNA sequencing problems as they maintain a specific structure [12]. In addition, DNA sequencing requires correct hybridization of single DNA strands to their target strands which are constructed with several biological restrictions [9] [42] [45]. Therefore, depending on the problem, DNA error correcting codes are typically constructed with several constraints, such as GC-content constraint, reverse-complement constraint, edit distance constraint, and thermodynamic constraint to name a few. All in all, general edit metric decoding using edit distance is very inefficient and calls for alternatives. Previous studies considered the use of side effect machines in an effort to optimize decoding performance.

3.3 Decoding of Codes using side effect machines

Side effect machines (SEMs) are an offshoot of finite automata. They are described further in Chapter 4. SEMs were first used to decode an edit metric code, with parameters $(12, 55, 7)_4$ in [11]. A quaternary edit metric code ($q = 4$) was chosen due to its suitability for bioinformatics problems. Two approaches were introduced in this study. The first one implemented a general error correction decoder named Single Classifier Machine (SCM) with the help of a genetic algorithm (GA). The SCM converted all codewords of an error correcting code into classification vectors. An error pattern was decoded by converting it into a classification vector and comparing that against the classification vectors of the codewords in order to find the closest match using Euclidean distance. The difference with conventional edit metric decoding was the use of Euclidean distance instead of the classic edit distance, which helped reduce the runtime complexity from $O(n^2)$ to $O(n)$. However, this made the decoding process an approximation. Therefore the fuzzy classification method was introduced as the second approach to improve the decoding accuracy. The main difference between the SCM and the fuzzy classification was the distance function where the latter also used edit distance, as described further in Section 6.6. Error patterns with distance 1 and 2 were created to examine the performance. The result showed that the best SCM corrected around 80% of the errors where fuzzy classification enhanced the performance by another 10%. The study used 3 different fixed sizes of 6, 12, and 18 states for the SCMs where the ones with 6 states performed poorly compared to the other two.

The previous work was expanded in [14] where five different codes of length 12 and a minimum edit distance of 7 were used. The number of codewords in each code ranged between 54 and 56. Each code was tested with machines of size 2 to 30. It was found that the accuracy of decoding increased rapidly up to machine size 12 and then plateaued from there on out. This study also decoded the error patterns using a locking side effect machines (LSEMs) technique where the codes were broken into subclasses. This method used the idea of multiple SEMs working together in a tree structure to classify the codewords in a better way.

In [34] a recentering-restarting evolutionary algorithm was used along with the basic genetic algorithm for generating SEMs. The results were compared with the previous study by testing with similar datasets. The number of codewords in the three codes used were 55, 60 and 60. The recentering-restarting GA algorithm was executed with a direct and indirect transposition representation. The results showed that indirect transposition representation had a strong ability to generalize SEMs when the number of states decreases. Using this method, SEMs with a small number of states (4 and 6) performed significantly better with fuzzy classification than it did with direct. However, this trend was not observed when the SEMs were generated using the direct transposition representation. It achieved results similar to those obtained with the basic GA where the error correcting ability improved as the number of states increased. Further investigation was suggested for larger codes and the number of states.

Brown [13] examined side effect machines to estimate their placement within the Chomsky hierarchy. It also provided a mathematical relationship between an error correcting code and the number of states to correctly map an error pattern to a codeword. If a binary code, over Σ symbols, has length n and a set of codewords w, then there exists an SEM of $\mid \Sigma \mid^n + w$ states which can decode an input string to a codeword. It explains the fact that SEMs for longer codes tend to need more states.

The latest study [31] on this topic examined the use of evolutionary programming (EP) for the creation of such decoders. EP is described in more detail in Section 5.3. The main advantage of using EP over GA was the easy modification of the number of states. The same codes from [34] were used in the study. The results were quite similar to earlier studies especially for error patterns with distance 1. The ability to modify the size of the SEMs using mutation operations that add states or remove states during evolution added a new dimension. It did not restrict the machines to a fixed size, rather allowing them to freely evolve (within a given range) to a size that would produce the best results. The study used a range of 4 to 18 states within which the machines were allowed to evolve. However, the machine sizes obtained over 100 experiments were inconsistent. The size varied from 9 to 18 states. Although larger machines performed better than the smaller ones, the fitness value did not improve significantly after 14 to 16 states. Moreover, although a preference for larger machines was observed, their association to better fitness could not be confirmed due to the potential bloat that might have existed in the machines.

The above studies clearly demonstrate the importance of studying the different aspects of the side effect machines for edit metric decoding. Furthermore, analyzing the best machines to determine the exact count of the used states to simplify the resulting machines is yet to be attempted. Most importantly, all of the previous studies used codes of the same length (12), as well as similar number of codewords, ranging between 55 to 60. Therefore, it is a definite requirement to look at codes of other lengths to find out the effect of SEM in different problem spaces. This thesis will be looking to fill these missing areas while attempting to improve the decoding accuracy of the SEMs in the process.

Chapter 4

Side Effect Machines

This chapter introduces the side effect machine (SEM) and its key characteristics. It explains the process of decoding using an SEM and discusses its benefits against the traditional methods of edit metric decoding. It discusses upon the difficulties involved in developing SEM to work as a general decoder. Finally, it concludes by explaining the optimization techniques and trade-offs associated with using SEMs as general decoders.

A side effect machine [6] is an extension of deterministic finite automata. Each node of the machine represents a state, one of which is preselected to be the start state. A counter is placed on each state and is initialized to zero. The machine takes a string or a sequence of symbols as input. The symbols are read in sequence and each symbol triggers a state transition. As a state is visited, the counter associated with it is incremented. This is what makes it different from a regular finite state machine (FSM). Once the entire sequence has been processed by the machine, the counter values are stored in a classification vector [31].

4.1 Classification Vector

A classification vector is a representation of the number of times each state has been visited in an SEM during processing of a given string. For an SEM with states 0, 1, 2, and 3 a classification vector $c = (c_0, c_1, c_2, c_3)$ can be defined such that every node represents the number of visits for its respective state. Figure 4.1 shows a simple SEM with four states 0, 1, 2, and 3. State 0 is the start state as denoted by the double circle. The sequence of ACTGCCGA produces the transition path 1, 1, 2, 3, 2, 2, 3, 0 and yields $c = (1, 2, 3, 2)$, as state 0 is visited only once, states 1 and 3 twice each and state 2 three times. Similarly, input CCTAGAAT produces a transition path of 0, 0, 3, 0, 2, 0, 1, 2, which makes the classification vector, $c = (4, 1, 2, 1)$.

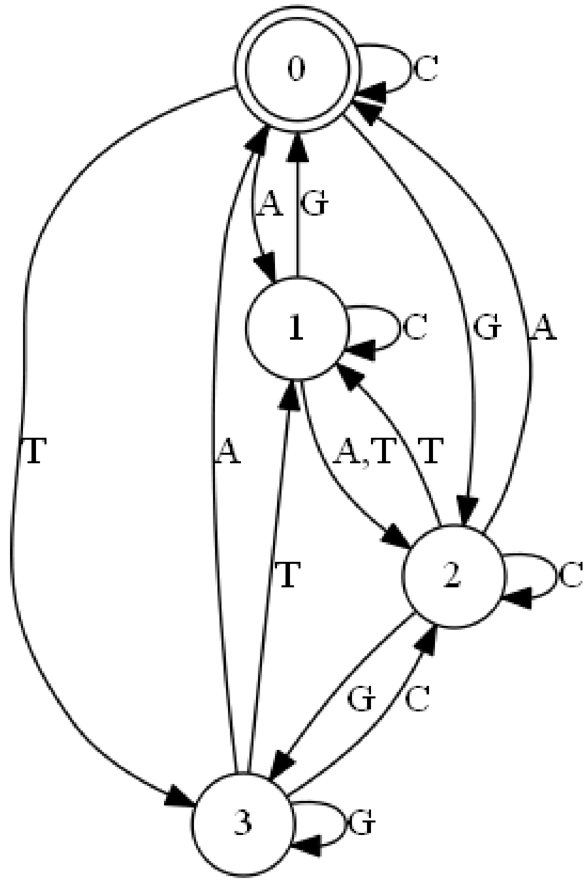

Figure 4.1: A simple side effect machine of four states

4.2 Transition Matrix

A transition matrix [12] is a representation of a state machine in a tabular form. The size of a transition matrix is $S \times \Sigma$, where S is the number of states and Σ is the number of input symbols. In the context of genomics, Σ=4 as there can only be 4 symbols in a DNA sequence, namely A, C, T, and G.

The state machine shown in Figure 4.1 corresponds to the transition matrix of size 4×4 shown in Table 4.1. It can be used to derive the transitions of a state for a given input. For example, state 2 transitions to state 1 upon receiving T whereas state 1 transitions to state 0 when G is received and so on.

Input Symbol	A	C	G	T
State Number				
0	1	0	2	3
1	2	1	0	2
2	0	2	3	1
3	0	2	3	1

Table 4.1: Transition Matrix of the SEM in Figure 4.1

4.3 Euclidean Distance

In an Euclidean space or a n-dimensional space, straight line distance between two points is called Euclidean distance [12]. For example, if $a = \{a_1, ..., a_n\}$ and $b = \{b_1, ..., b_n\}$ are the vector representation of two points of a n-dimensional space, the Euclidean distance between them is $D(a, b) = \sqrt{(a_1 - b_1)^2 + \ldots + (a_n - b_n)^2}$. However, the actual distance is not required when comparing between multiple points, but only their relative distance, i.e. whether point X is closer to point Y or point Z. Therefore, the square root can be ignored to reduce the computational cost [12]. Therefore, the actual calculation performed in this work is $(a_1 - b_1)^2 + \ldots + (a_n - b_n)^2$.

4.4 Decoding using Side Effect Machines

An SEM can be used to quantify how different two words are from each other. The words can be compared by running their symbols through the state machine and finding the Euclidean distance between the classification vectors that are produced. The same word will always take the same path through the states and thus will produce the same classification vector. Therefore, for identical words, the Euclidean distance between their classification vectors will be zero. On the other hand, a larger Euclidean distance generally implies that there are more differences between them.

This principle can be used to decode edit metric codes using an SEM. The idea is to compare an error pattern with each codeword in a code to find out which one it resembles most closely. To do this, the error pattern and the codewords are run through the SEM and their classification vectors are computed. Next, the Euclidean distance between the classification vector of the error pattern and that of the codewords are calculated. The codeword that is associated with the smallest distance is considered to be the original word as long as the distance is within a given *tolerance*, which is the error correction capacity of the code.

		Classification Vector C0, C1, C2, C3				Edit distance with received word	Euclidean distance of classification vector with received word
Codeword1	ACTGCCGA	1	2	3	2	1	2
Codeword2	CCTAGAAT	4	1	2	1	5	22
Codeword3	ATCGACGT	2	2	3	1	3	6
received word	ACTGCCGT	0	3	3	2	-	-

Table 4.2: Comparison of edit distance with the Euclidean distance of the classification vector for a sample word with three other codewords using the SEM of Figure 4.1.

Table 4.2 shows a simple example of how a SEM can be used to decode a received word. Suppose a code has three codewords of length 8. If a word is received, the error correction can be done by finding the smallest edit distance with all the codewords. Here, it is found that received word, ACTGCCGT, has smallest edit distance with Codeword1, ACTGCCGA. A SEM (Figure 4.1) can also be used to find the original word. First, the SEM finds the classification vectors of all codewords and the received word by running them through the machine. Then, the Euclidean distance between the classification vector of the word and that of the codewords are calculated. It is also found that Codeword1 has the smallest Euclidean distance to the received word which leads to a successful decoding.

4.5 Pros and Cons of using SEM in Edit Metric Decoding

The biggest advantage of using an SEM for decoding edit metric codes comes from the avoidance of having to compute the edit distance which, in terms of performance, is the costliest operation in the general decoding algorithms. Calculating the edit distance between two words of length n produces a runtime complexity of $O(n^2)$ [44]. In the general decoding technique, the edit distance from the received word must be calculated for each codeword which makes the entire process of error correction inefficient. This is why optimizing the performance of the decoder remains an area of great interest among researchers across disciplines. Although Hamming distance can be calculated in $O(n)$ time, Hamming distance codes are not appropriate because they are unable to detect insertions and deletions. Therefore, they are insufficient for use in biological applications. The SEM, with the help of its classification vector, provides a way to compare two words for insertion, deletion and substitution and it is able to do so in $O(n)$ time since the state machine makes only as many transitions as the number of symbols in the word. Furthermore, SEMs can handle any structure of code with the addition of biological restrictions. This is why an SEM is better suited to tackle the problem of decoding error correcting codes, particularly in the field of genomics.

The downside of using an SEM as a decoder is its probabilistic characteristic since the Euclidean distance is used as a cheap substitution for edit distance. Therefore, some SEMs perform better than others for a specific code. Furthermore, two different sequence patterns can obtain the same classification vectors or Euclidean distance from a SEM. This makes the decoding process ambiguous. For example, two different words, AACG and ATCG, will produce the same classification vectors when they are passed through the machine in Figure 4.1. This is happening because there is a transition from state 1 to state 2 on both input A and T. Therefore, the goal is to create a generalized SEM using evolutionary techniques that will maximize the error correction capability.

Chapter 5

Evolutionary Computation

This chapter discusses the general concept of evolutionary algorithms (EAs) and their application for solving hard optimization problems. It also gives a brief overview of the key features of EAs such as solution representation, initialization, fitness, selection, and genetic operators. After briefly talking about the different types of EAs, it provides an in-depth look at evolutionary programming (EP) as this is the technique used to generate SEMs. Finally, the chapter reviews how the different operations of EP are tweaked and tuned for developing SEMs.

5.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) form a class of metaheuristic optimization techniques. They are a subset of *evolutionary computation*, a technique that was inspired by Darwin's Theory of Evolution. Darwin proposed a process in [18] called natural selection, also known as the "survival of the fittest". The fittest individuals are those who are best equipped with the abilities to survive in their environment. They are the ones that grow to maturity, reproduce and thus pass on their traits to their offspring. The individuals lacking such *fitness* either do not survive long enough to reproduce or do so at a lower rate. This process, repeated over many a generation, typically results in the "good" qualities to prevail while gradually improving the average fitness of the entire *population*. This idea of biological evolution is applied in evolutionary computation to heuristically find optimal or near-optimal solutions for a variety of computation problems.

The basic idea behind any EA is to create a solution that gradually improves over time and converges toward the best solution in a problem space. The process starts with a population of candidates and selects the stronger individuals based on their fitness to reproduce and create the next generation. Reproduction usually takes place in two ways — mutation and crossover. This process is repeated until an individual gets the (sub-)optimal fitness or the number of iterations exceeds a predefined threshold. It is worth noting that the fitness may plateau after a certain number of iterations without ever reaching optimal fitness.

5.1.1 Solution Representation as Chromosomes

Candidate solutions are represented as *chromosomes* in an organism. Each chromosome is made up of a sequence of genes that encodes the characteristics in that organism. Candidate solutions are also made in such a way that all the necessary information is present in one solution according to the problem requirement. However, the candidate solution is not required to have a direct mapping to the solution. In general, the representation scheme defines how the problem is structured in the evolutionary algorithm. The set of all candidate solutions at any given point during the evolutionary process is called a population.

5.1.2 Initialization

The initial population is generally filled up with random solutions from the entire search space. This increases the chance of gradually evolving towards the best solution and reduces the possibilities of getting confined to a local search space. Seeding is another common method of initialization, where the initial population is constructed with known good chromosomes.

5.1.3 Fitness

Fitness is a quantitative heuristic measure of the effectiveness of a solution in solving a given problem. It can also be conceived as an understanding of how close a solution is to an optimal solution.

A fitness function is an objective function that evaluates the fitness of a solution and gives it a score. The score is calculated based upon certain criteria dictated by key parameters in the problem space. Each individual in a population is given a fitness score which is later used for selecting candidates for reproduction.

5.1.4 Selection

Selection in the evolutionary technique is modeled based on natural selection in biological evolution. The selection process uses fitness to allow the fittest individuals to survive while the others are eliminated. However, it should be noted that just as it happens in nature, sometimes lesser fit parents can also produce relatively fitter children as a result of a favorable genetic modification during reproduction. Therefore the selection process is often devised in ways that allow for a few lower ranked members to be selected along with the ones with the higher fitness scores.

5.1.5 Genetic Operators

In general, two types of genetic operators are used to create new candidate solutions.

Crossover is a genetic operator, also known as recombination, used to combine two individuals' chromosomes and create one or more children which inherit, in a certain way, the genes of both parents.

The *mutation* operator is applied to a single individual in the population that promotes diversity in the population. It changes single or multiple gene values in a chromosome. It helps to explore the neighborhood of current solutions to find the undiscovered regions of the search space.

5.2 Types of Evolutionary Algorithms

Based on implementation details, EAs can be categorized into four major types:

1. Genetic algorithm (GA) : The solutions of the problems, chromosomes, are usually represented with strings or numbers and the data structures are allowed to evolve using genetic operators, such as mutation and crossover.

2. Genetic programming (GP) : Similar to genetic algorithm, except the solutions themselves are computer programs. The programs can be represented as tree structures where traditionally the nodes contain operators and the leaf nodes contain variables.

3. Evolutionary programming (EP) : Similar to genetic programming with one exception - only the numerical parameters of the data structures are allowed to evolve, not the structures themselves. It is described further in Section 5.3.

4. Evolution strategy (ES) : ES is implemented with the goal of solving real-valued function optimization problems. The mutation rate is self adjusted and used for solutions represented as vectors of real numbers which is similar to EP [7].

5.3 Evolutionary Programming

Evolutionary programming was first conceived by Lawrence J. Fogel in 1966 [27]. Although similar to GA and GP, it differs from these algorithms in the way it places the emphasis on evolving the behavior of a population rather than trying to emulate the genetic operations that take place in nature. EP uses mutation alone as the method of reproduction.

EP is generally used as a method of optimization when an analytical search is inefficient and other heuristics are either impossible or ineffective. It had been effectively implemented to numerical and combined optimization challenges [23, 25, 24]. It is also suitable for problems for which there exist many locally optimal solutions[47]. It was first used for solutions represented as finite state machines and was later enhanced to use other representations. No restrictions are imposed on the data types used to define the attributes of a solution. However, the attributes are only allowed to evolve numerically, not structurally, that is to say, no attributes are allowed to be added or removed from the original data structure of the solution. In this thesis, the solutions are represented as SEMs (Section 5.4.2) and EP provides the ability to easily mutate them by adding or removing states and modifying transitions. In addition, the solution space contains many local optima, making it well suited for EP.

Other biological applications of EP include multiple sequence alignment of nucleotide or protein sequences [15], flexible docking and drug design problem [28], reconstruction of DNA sequence information from a simulated DNA chip [26], and classification problems using DNA coding [21]. It has also been used in other disciplines, such as, in mixed wireless controllers to control the direction of transmission [20], in fast voltage stability index based reactive power planning [36], electromagnetic optimization problem [30], and in transparent optical networks for survivable routing and wavelength assignment [10].

5.4 EP using SEMs

5.4.1 General Steps

1. An initial population of a fixed size is created with randomly generated SEMs.

2. Using the fitness function, each member is given a fitness score.

3. Children are created using mutation by changing the start state, modifying a transition or adding/removing a state.

4. The children are given fitness scores and merge with the parents doubling the size of the population.

5. All individuals are given a bout score using a bout system (Section 5.4.4).

6. The population is cut in half and brought back down to its initial size by selecting the fittest members (based on bout score) to form the next generation.

7. Steps 3 to 6 are repeated until a solution with a desired fitness level is found or the number of iterations reaches a predefined threshold.

5.4.2 Representation

Representation defines how a candidate solution is organized in a problem space. It works as a chromosome to hold important information. As discussed in the previous chapter, SEMs are represented using a transition matrix. The transition matrix stores the state numbers or the path it would follow for an input sequence. The size of an SEM is equal to the state number that is initially generated randomly between the minimum and the maximum number of states. Each state has four output transitions for four DNA symbols that can go to another state or itself. When a SEM is given an input sequence, a pointer is placed at a state, named as an initial state, to indicate which state to start from. In this thesis, state 0 has been selected for the initial state for every machine, which can be replaced with another state through mutation. A machine can also add a new state, delete an existing state or change the output transition to another state. However, when an input sequence is passed through a machine, it does not require all its states to be visited. Not visited states, acting as bloat, can stay in the machine without affecting the final fitness value. Bloat is unwanted growth of the structure which increases computational cost and uses more memory.

5.4.3 Initialization

The candidate solutions are initialized (Section 5.1.2) randomly. The number of candidate solutions depends on the population size. Randomization helps to distribute the population over the problem space. Each candidate in the initial population also has a random number of states from within an allowed range.

5.4.4 Bout System

The bout system is a selection (Section 5.1.4) process that is similar to tournament selection [38]. As discussed previously, the best ranked candidates in a population do not always converge to the optimal solution. Therefore, the selection is performed using the bout system to avoid always selecting only the best ranked candidates. Each SEM has a bout score that is initially set to 0. Based on the bout size, this process selects distinct random individuals for each SEM and runs a tournament between them to find out the bout score of that SEM. For example, if the bout size is 5, each SEM in the population selects 5 other SEMs and compares the fitness (Section 5.1.3) value with others. If the fitness value of the SEM is better than another SEM, the bout score increments by one. Thus, each SEM in the population gets an individual score that can later be sorted to select the best SEMs for the next generation. In this thesis, the value of the bout size is 10.

5.4.5 Mutation

Mutation is a unary operation to modify an SEM from the parent population to the child population. It plays an important role in evolutionary programming. It changes the structure of a machine randomly. Four different types of mutation are used in this thesis:

(i) Change a Transition

This operation changes a single transition of one randomly selected state. The program randomly selects a transition from the transition matrix and change its ending state to another state for this mutation. Figure 5.1 shows an example of it where the left SEM is before mutation and the right SEM is after mutation. Here, a transition from state 2 to state 1 using input value of 3 has mutated to go from state 2 to state 0.

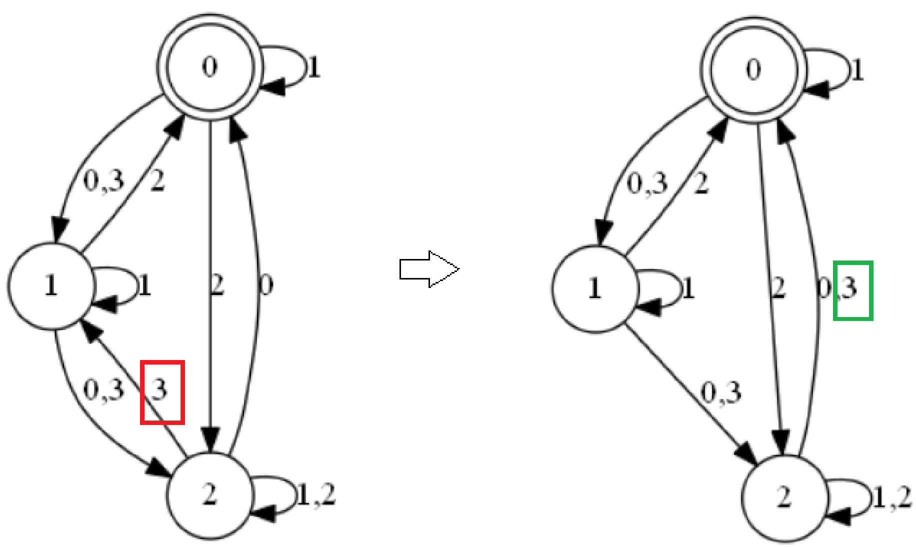

Figure 5.1: Example of changing a transition

(ii) Add a State

This type of mutation adds a new state into the machine at a random position. The state is added only if the resulting size of the machine stays within the upper bound. It is then connected to the rest of the machine by adding new outgoing transitions to randomly selected states. Since no incoming transitions are created, the newly added state stays unreachable from the rest of the machine. For this reason, the addition does not immediately make an impact on the fitness of the machine, but rather relies on future mutations to possibly alter the transitions in ways that allow the new state to be reached. It is also worth noting that a machine with more connecting states does not guarantee a higher fitness value. Figure 5.2 shows adding a new state 3 to the machine.

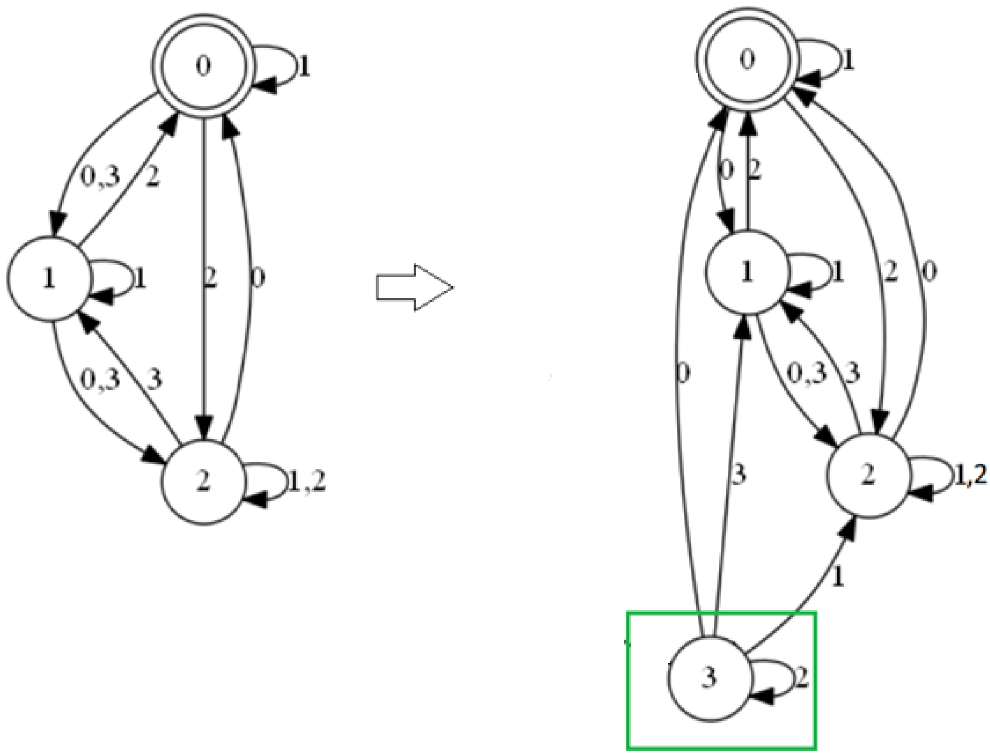

Figure 5.2: Example of adding a new state

(iii) Delete a State

This operation deletes a state from the machine at a random position as long as the resulting size of the machine stays within the lower bound. The input edges from other states to the removed state are connected to its previous state. Furthermore, if the state that is going to be deleted is the initial state, then the next state that comes numerically after that will be the new initial state. Figure 5.3 shows deleting state 1, from the machine. However, deleting state 1 requires other modifications in this machine. There are two transitions from state 0 with input 0 and 3 and one transition from state 2 with input 3 to state 1 in the parent machine. After deleting state 1, the output edges from state 0 and state 2 which were going to state 1, will go to state 0 to maintain connectivity of the whole machine. Afterwards, all states greater than the removed state are decremented by 1 to fill the void left behind by the removed state and thus state 2 now becomes state 1.

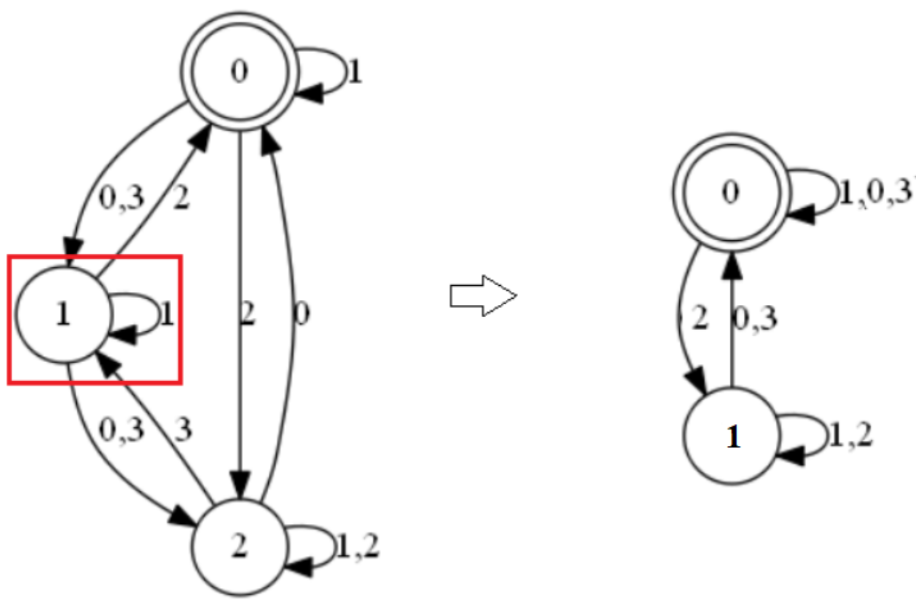

Figure 5.3: Example of deleting a State

(iv) Change Initial State

As mentioned before, state 0 is selected as the initial state at the start of the program. This operation with predefined probability selects a state to make it the new initial state of the machine. Figure 5.4 shows a mutation of changing the initial state from state 0 to state 2.

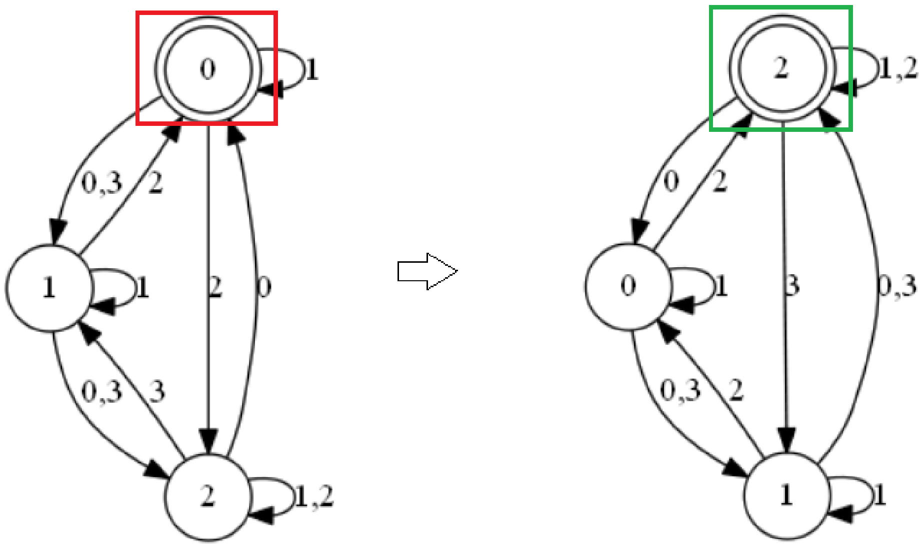

Figure 5.4: Example of changing an initial state

Chapter 6

Methodology

This chapter discusses two methodologies, direct classification (Section 6.3) and fuzzy classification (Section 6.6), to decode an error pattern. Two datasets — training and verification — with identical characteristics are used for this study. The idea is to use the first set to find the best possible solution, i.e. an SEM capable of correcting as many errors as possible, and then verify its accuracy by running it against the second dataset. Each code in the dataset consists of several codewords, from each of which are constructed a number of error patterns with edit distances 1 to 3 from the codeword (Section 6.2). Each individual in the population is an SEM. The fitness of a given SEM is calculated using a predefined fitness function by running all the error patterns through it. Each member of the population is then mutated (Section 5.4.5) to create a population of children which are then merged with the parents. Based on their fitness (Section 6.3.1) only half of the members from the combined population are selected to form the next generation. This process is repeated for a predefined number of generations at which point the SEM with the highest fitness is selected as the final solution. As mentioned above, the SEM is then tested against the second dataset to verify its error-correcting capability across the problem space. Later, fuzzy classification with a tolerance value is implemented to improve the decoding capability on both datasets.

6.1 Dataset

Nine quaternary codes are used for the purpose of this study. All the codes are presented in Appendix A. Each code consists of a set of words of a given length. Three different lengths of codes — 10, 12, and 14 — are used in this experiment, where the three codes of length 12 are taken from [31]. All the codes are sets of strings of DNA sequence tags comprised of symbols A, C, G, and T. However, the symbols of the DNA sequence comprised of symbols

A, C, G, and T are replaced by numbers 0, 1, 2, and 3 respectively in order to reduce the computational load and the memory footprint of the program.

3 1 2 1 1 0 0 0 3 3 3 3 3 0 0 0 2 2 1 1 1 1 1 2 0 2 2 3 2 1 0 0 0 0 0 0 0 0 0 0 2 2 2 3 1 2 1 3 3 1	1 3 2 3 2 1 0 2 0 2 0 2 3 3 3 3 2 2 2 2 3 2 1 1 1 1 1 2 2 0 0 3 0 3 3 1 1 1 1 1	1 1 3 1 0 1 3 1 0 0 0 3 3 2 2 3 3 3 3 0 2 1 0 2 3 3 3 0 1 3 0 1 0 0 1 1 3 3 2 2	2 2 2 2 0 0 0 1 2 2 3 0 1 1 3 2 3 0 0 1 3 0 0 3 0 3 1 2 3 3 0 0 2 1 2 2 2 1 1 3

Table 6.1: Code17-1: a $(10, 17, 7)_4$ code

Recall (from Section 2.5) that an edit metric code is denoted as $(n, M, d)_q$ where n is the length of the codeword, M is the total number of codewords in the code and d is the minimum edit distance of the code. Table 6.1 shows a $(10, 17, 7)_4$ code, labeled as Code17-1 where the length of each codeword is 10, the total number of codewords is 17 and all codewords have a minimum edit distance of 7 from one another. For example, the first codeword (3 1 2 1 1 0 0 0 3 3) sits at an edit distance of 8 and 9 from the second (3 3 3 0 0 0 2 2 1 1) and the third (1 1 1 2 0 2 2 3 2 1) codewords respectively, whereas the edit distance between the second and the third codewords is 7. All of the nine codes used in this study maintain a minimum edit distance of 7, thereby can correct up to $t = (7 - 1)/2 = 3$ errors.

6.2 Creation of Error Patterns

Three sets of error patterns are created from every codeword. As mentioned earlier, there are three types of errors commonly observed in a DNA sequence, namely insertions, deletions, and substitutions. Therefore, the error patterns are generated by applying these modifications to the set of codewords. Each of these operations applied on a given codeword produces an error pattern that is edit distance 1 away from that codeword. Similarly, in order to create an error pattern with an edit distance of d, d modifications to the codeword are required. Because the minimum distance of the code is 7, the maximum number number of errors that can be corrected is 3. As a result, three sets of error patterns of edit distance 1, 2, and 3 are created for every codeword using the above principle. All error patterns are created to be of the same length as the codewords because these are assumed as potential "messages" in this application.

The following methods are used for generating the error patterns. Note that no two operations are made to the same bit positions. In other words, all modifications must be made in different bit positions.

- Distance 1 error pattern: one substitution
- Distance 2 error pattern: two substitutions or one insertion and one deletion
- Distance 3 error pattern: three substitutions or one insertion, one deletion, and one substitution.

For every codeword, n error patterns are created for edit distance 1 through 3, where n = the length of the codeword. For example, every codeword under Code17-1 has 10 error patterns with a single error, 10 with two errors, and 10 with three errors. Therefore, 170 $(17 * 10)$ error patterns are generated for each increment of the edit distance from 1 to 3 resulting in a total of 510 $(170 * 3)$ error patterns for the code.

The idea behind creating an error pattern e from a codeword c by 1, 2 or 3 errors is to check if an SEM is able to correctly decode it. If an SEM correctly decodes all error patterns, whether for the training set or the verification set, then it obtains a perfect score which is equal to the total number of errors. Thus for Code17-1, a perfect score is 170 at each individual distance and 510 overall.

Code	Number of Codewords	Length of each Codeword	Number of errors at an individual distance of 1, 2, and 3	Total number of errors over all distances
Code17-1	17	10	170	510
Code17-2	17	10	170	510
Code18	18	10	180	540
Code55	55	12	660	1980
Code60-1	60	12	720	2160
Code60-2	60	12	720	2160
Code201	201	14	2814	8442
Code205-1	205	14	2870	8610
Code205-2	205	14	2870	8610

Table 6.2: Dataset

While previous studies [11, 14, 34, 31] dealt with codes of length 12, this thesis expands the scope of the investigation by varying key parameters, such as the length of the codes and the number of codewords to understand the effectiveness of this approach on different code lengths. Three codes of length 10, 12 and 14 are used, where the codes of length 12 (Code55, Code60-1, and Code60-2) are taken from [31] to compare the results. Table 6.2 lists all codes with their respective number of codewords and the number of errors that are generated to conduct the experiment.

6.3 Direct Classification

Decoding an error message against a code requires finding the codeword that has the closest resemblance to the message, that is to say the codeword with the smallest edit distance from the message. Yet, calculating the edit distance is an inefficient process that results in a runtime complexity of $O(n^2)$. To avoid this heavy cost, a direct classification method that employs Euclidean distance is used to bring down the runtime complexity to $O(n)$. Instead of comparing the error pattern with the codewords by means of edit distance, this method runs them through an SEM to generate their respective classification vectors and compare these using Euclidean distance to decode the error. However, it should be noted that Euclidean distance is used as a cheap substitution of edit distance and therefore must be considered as an approximation. It is also worth noting that the affinity of Euclidean distance to edit distance depends heavily on the SEM and some machines react better to this method than others. This is why evolutionary programming techniques are used to construct an SEM that provides the maximum accuracy in terms of error correction.

6.3.1 Fitness

The fitness function used here is a simple counter that counts the number of error patterns accurately decoded by an SEM. An error pattern is considered decoded when it is found to be closer to its original codeword as compared to the other codewords. First, the Euclidean distance between the classification vectors of the error pattern and the original codeword is measured. This distance is then compared against distances measured from the other codewords. The fitness score is incremented if the distance measured from the original codeword is found to be the smallest amongst all. The process is repeated for all error patterns and the higher the score, the better the performance of the SEM.

6.4 Parameter Values for Initial Sets of Experiments

Four sets of parameter values are used on all datasets. These values are presented in Table 6.3. The population size, generation number, and bout size are chosen from a past study [31] for consistency. Although initial tests are done with different values for these parameters with Code55, Code60-1, and Code60-2, no significant improvements are observed.

Parameter settings E1 and E2 were first used in [31]. Additionally, two new settings, E3 and E4 are added. The new settings reduce the probabilities of "add state" and "remove state" mutation operations. The motivation is to observe the effect of different mutation

Experiment	E1	E2	E3	E4
Population Size	300	300	300	300
Number of Generations	1250	1250	1250	1250
Bout Size	10	10	10	10
Probability of Changing a Transition	0.6	0.75	0.8	0.85
Probability of Changing the Initial State	0.1	0.05	0.1	0.05
Probability of Adding a State	0.15	0.1	0.05	0.05
Probability of Removing a State	0.15	0.1	0.05	0.05

Table 6.3: Parameter values for four sets of experiments

settings on the resulting machines and the overall decoding accuracy.

6.4.1 Range of States

In earlier studies [11, 14, 34], the effectiveness of an SEM was observed to be related to its size. Therefore with EP, the machines are allowed to shrink or grow within a certain range. The range used in the previous study [31] was 4 to 18. Even though the experiment showed higher accuracy for machines with larger size in general, the behavior was not consistent. The best machines are prevalent in certain sub-ranges of sizes rather than being inclined to just one size. In fact, the positions and the widths of these sub-ranges are also observed to vary from code to code. As a result, a new approach is required to investigate possible relationships among all these variables. In this study, the range of 4 to 18 is divided into eight smaller sub-ranges, as shown in Table 6.4. The idea is to investigate whether certain codes react better to certain sub-ranges and understand how the machines evolve within those sub-ranges to reach their final sizes.

Rnage of States	Minimum Number of States	Maximum Number States
4to6	4	6
4to8	4	8
6to12	6	12
6to18	6	18
8to14	8	14
8to18	8	18
10to16	10	16
14to18	14	18

Table 6.4: Different range of states

6.4.2 Count of Exact Machine Size

At the beginning, the size of an SEM and its transitions are initialized randomly within the bounds of the selected range. Due to the random nature of the transitions, not every state is guaranteed to be visited when a pattern is passed through the machine. Additionally, at the time of reproduction, the location of the mutation is also selected randomly and can alter the transition matrix in such a way that a state which was visited at least once in the parent machine may get removed from the transition path, and hence is never visited in the child machine. A depth-first search is performed to determine which states are reachable or unreachable. The unreachable states can be purged to simplify a machine without altering its behavior and thus help to determine the true machine size required for optimal performance.

6.5 Pseudocode Algorithm to Generate SEM using EP

First, a population of SEMs with random sizes, bound by the selected range, are created. This step is known as the initialization of the population. The initial population is now considered as the first generation of candidates and act as parents for the next. Each parent is then mutated in one of four ways — altering a transition, altering the start state, adding a new state, removing a state — to produce a child. The type of mutation that is applied is determined by its overall probability of being used as shown in the Table 6.3. The reproduction of the parents doubles the population size. All of these candidates are then given fitness scores and ranked using the bout system and only the top half is selected to form the next generation and act as parents for the following generation. This process is repeated for a predefined number of generations — another parameter of the experiment — and the SEM with the highest fitness from the last generation is selected as the final solution. The pseudocode for this process is shown in Algorithm 2.

```
Initialize parent population with randomly generated SEMs
Get fitness of each SEM in Parent population
Get best SEM in parent population
for i = 1 to generation number do
    Make copy of parent population to child population
    for j = 1 to number of child population do
        Mutate
    end
    Get fitness of each SEM in child population
    Get the best SEM in child population
    if child population best > parent population best then
        new best = child population best
    else
        new best = parent population best
    end
    Add all child population with Parent Population
    Sort all population based on bout system
    Select half population for Next Generation
end
```

Algorithm 2: Algorithm for Evolving Side Effect Machine using Evolutionary Programming

6.6 Fuzzy Classification

As discussed earlier, the direct classification method compares the received error pattern with every codeword in a code to find the closest match. The words are compared using the Euclidean distance between their respective classification vectors. The error pattern is considered decoded when it is able to find the original codeword from which it was generated. Using the byproducts of direct classification, fuzzy classification tries to optimize the decoding process by checking the most probable error codes first. This is achieved by first creating a sorted list of all codewords in ascending order of the Euclidean distance of their classification vectors to the error patterns classification vector. The codewords are then compared with the received error pattern using edit distance until a match is found within the correction capacity of the code. Therefore, a tolerance is often used to narrow the search by filtering out codewords whose said Euclidean distances are above the given tolerance. This essentially optimizes the search by shrinking the problem space into a smaller hypersphere with a radius that is equal to the tolerance and the search for valid codewords takes place only within the perimeter of the sphere. The algorithm stops when either the correct codeword is found or the list of all codewords within the given radius is exhausted. In addition to improving runtime, the fuzzy classification method can be used to identify when decoding fails, that is no codewords within the given tolerance have an edit distance less than or equal to $(d-1)/2$ from the error pattern.

Chapter 7

Results and Analysis

This chapter shows the results of two different methodologies — direct classification and fuzzy classification — to find the error correction accuracy over nine different codes with different parameter settings. It also analyzes the different aspects of the parameters in an attempt to look for potential trends and establish meaningful relationships among them.

The summary statistics of maximum fitness accuracy for each code, along with their respective range and experiment number, are presented in Tables 7.1 and 7.2. Tables 7.1 and 7.2 show the results of direct classification and fuzzy classification, respectively, for both training and verification datasets. It was found that smaller lengths of codes have better maximum accuracy than larger lengths of codes in both direct classification and fuzzy classification. The only exception was code18 in the verification dataset for direct classification, which has lower accuracy than code55.

The full tables for these results are presented in Appendix B. It has been seen that there is no particular range or experiment setting which outperforms the others. Therefore, further analyses have been made to find relationships in the following sections.

	Direct-training			Direct-verification		
Code	Max Accuracy %	Range	Exp.	Max Accuracy %	Range	Exp.
code17-1	83.5	6 to 18	E3	76.7	14 to 18	E1
code17-2	84.9	14 to 18	E3	74.3	8 to 14	E4
code18	81.3	14 to 18	E2	72.6	14 to 18	E2
code55	75.2	14 to 18	E3	73.7	8 to 14	E2
code60-1	75.2	14 to 18	E4	72.1	10 to 16	E3
code60-2	73.6	8 to 18	E3	71.3	6 to 18	E4
code201	65	14 to 18	E3	64.3	14 to 18	E3
code205-1	65.6	14 to 18	E2	63.5	6 to 18	E3
code205-2	66.4	8 to 18	E2	64.1	14 to 18	E3

Table 7.1: Direct classification maximum accuracy result for each code. It also shows the range and experiment number from where the maximum accuracy has been obtained.

	Fuzzy-training			Fuzzy-verification		
Code	Max Accuracy %	Range	Exp.	Max Accuracy %	Range	Exp.
code17-1	93.1	8 to 18	E4	89.2	6 to 18	E2
code17-2	91.6	6 to 12	E3	88.2	4 to 6	E1
code18	88.3	4 to 6	E4	84.4	4 to 6	E4
code55	88.2	8 to 18	E1	88.1	6 to 18	E2
code60-1	88.2	8 to 14	E2	86.9	8 to 14	E2
code60-2	87.5	6 to 18	E4	87	6 to 18	E4
code201	81.2	8 to 18	E2	80.9	8 to 18	E2
code205-1	80.6	6 to 18	E3	80	6 to 18	E3
code205-2	82.4	8 to 18	E3	82.2	8 to 18	E3

Table 7.2: Fuzzy classification maximum accuracy result for each code. It also shows the range and experiment number from where the maximum accuracy has been obtained.

7.1 Number of States

Previous studies [31, 14] found an inclination towards larger machines with respect to decoding accuracy, a trend that is analyzed further in this study. The extent of this impact appears to vary based on a number of other factors such as the length of the code, the classification method and the dataset used i.e. direct vs fuzzy and training vs verification. To understand this relationship, the median accuracy rate was plotted against the machine size (number of visited states) of all the best machines found across mutation types and ranges. Four such graphs are shown in Figures 7.1 – 7.4 for the different datasets used, direct training, direct verification, fuzzy training, and fuzzy verification respectively. As shown in Figure 7.1, using direct classification with the training dataset, larger machines show a higher and steady rate of improvement for longer codes. This is evident in the steady incline that the codes of length 14 (code201, code205-1, and code205-2) experience as the machine size grew from 8 to 18. On the other hand, the decoding accuracy for the smaller length codes starts plateauing or even declining beyond a certain machine size, namely 13 and 15 respectively for codewords of length 10 (code17-1, code17-2, code18) and 12 (code55, code60-1, code60-2). In contrast to the gradual improvement noticed across codes, a slight dip was noticed for code18 and code201, just as they were approaching the maximum size, which could be an indication of a possible plateau once the machine grows past a certain size. However, the evidence is not consistent enough for making such a conclusion without further experimentation.

As a side note, it should be noted that the fitness did not plunge from machine size 6 to 7, even though it appears that way. The graphs in Figures 7.1 – 7.4 include machines across all ranges. For some of the larger codes, no best machines with 5 or 7 states were created, which formed the peaks at the left of the graphs. A possible explanation is that the larger codes tend to produce larger machines. Therefore, for some of the larger codes, range 4 to 6 only produced machines with 6 states whereas ranges 4 to 8 and 6 to 12 produced machines with 8 states or more.

With the verification dataset as shown in Figure 7.2, the plateau is noticed much sooner for smaller codes while SEMs for codes with length 14 keep improving with a larger number of states till the end, albeit at a lower rate. In general, the results from two datasets with direct classification show a preference for a higher number of states, which is prominent for codes of length 14.

Using the fuzzy classification method (Figures 7.3 and 7.4) for lengths 10 and 12, the improvement in accuracy is negligible once the machine size grows beyond 8. In fact, for codes of length 10, it even seems to decline once it goes past 10 – 12 states. The decoding accuracy for codes of length 14, however, showed a slight but steady improvement as the machine size grows all the way to the maximum allowed size. The above results support the fact found in [31] that machine sizes have little impact on fitness using fuzzy classification once the machines go past a certain number of states. It also finds that the difference in accuracy between the training and verification datasets is much more evident with the direct approach than the fuzzy approach, especially with the machines created for codewords of length 10.

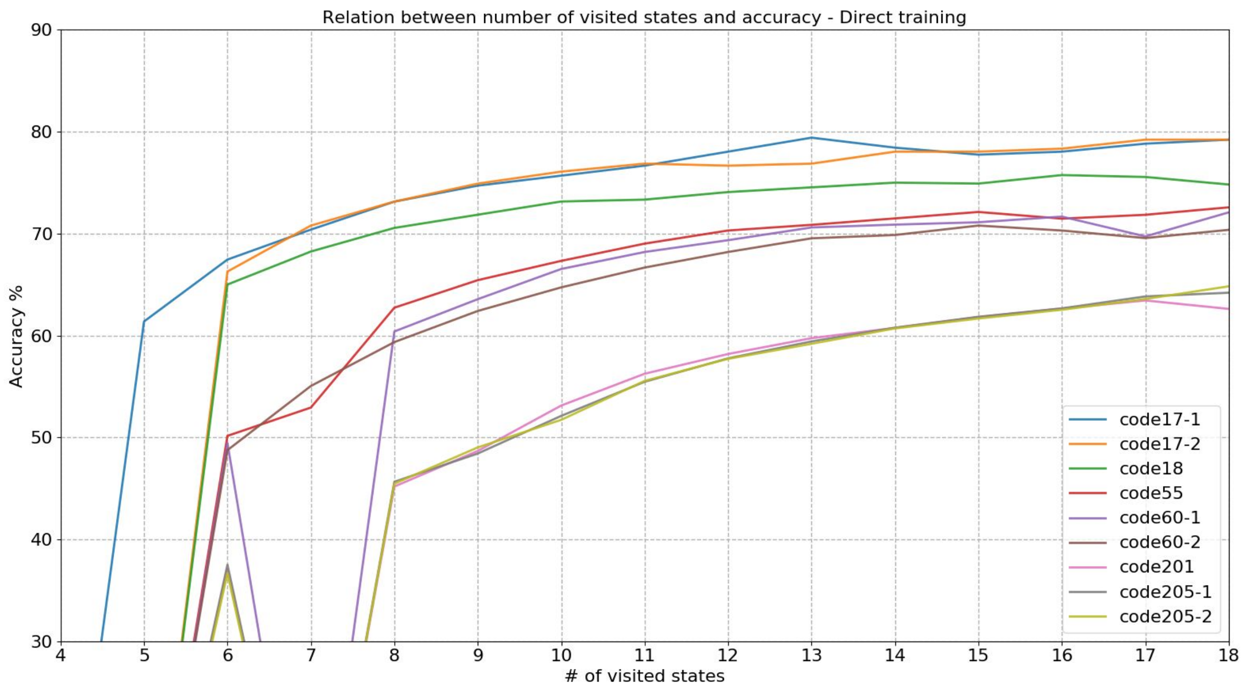

Figure 7.1: Role of machine size (visited number of states) on accuracy over all ranges and all experiments(E1, E2, E3, and E4) for all codes in training dataset with direct classification

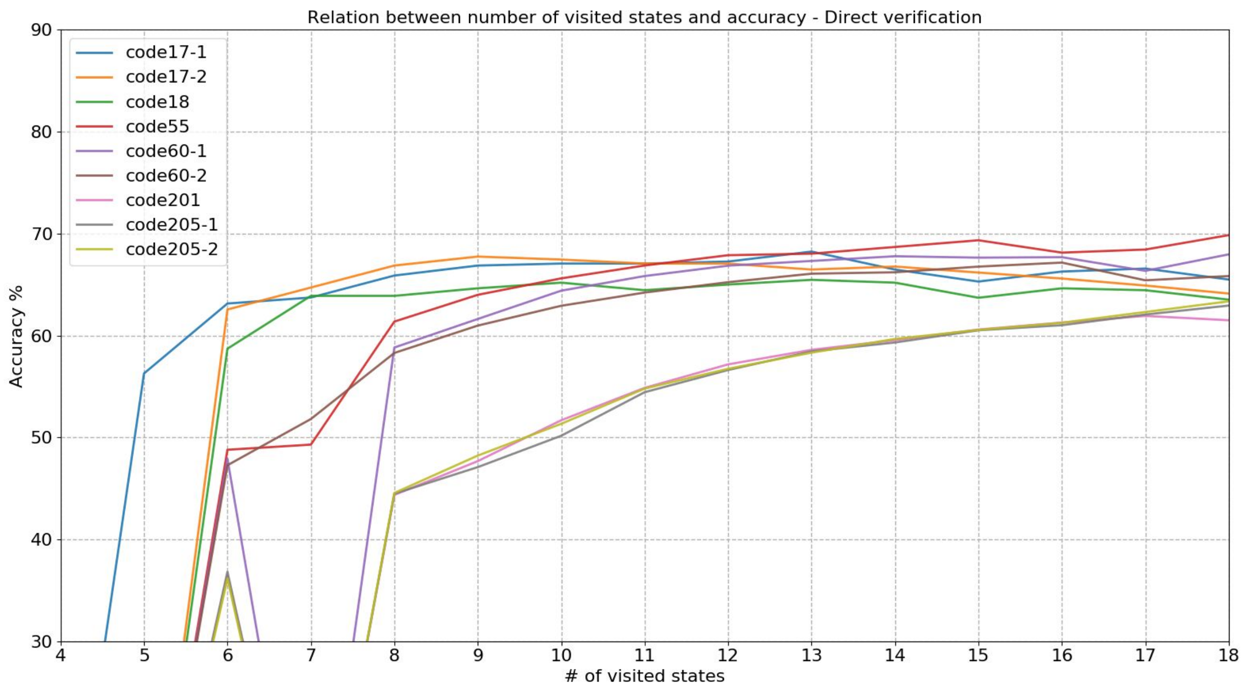

Figure 7.2: Role of machine size (visited number of states) on accuracy over all ranges and all experiments(E1, E2, E3, and E4) for all codes in verification dataset with direct classification

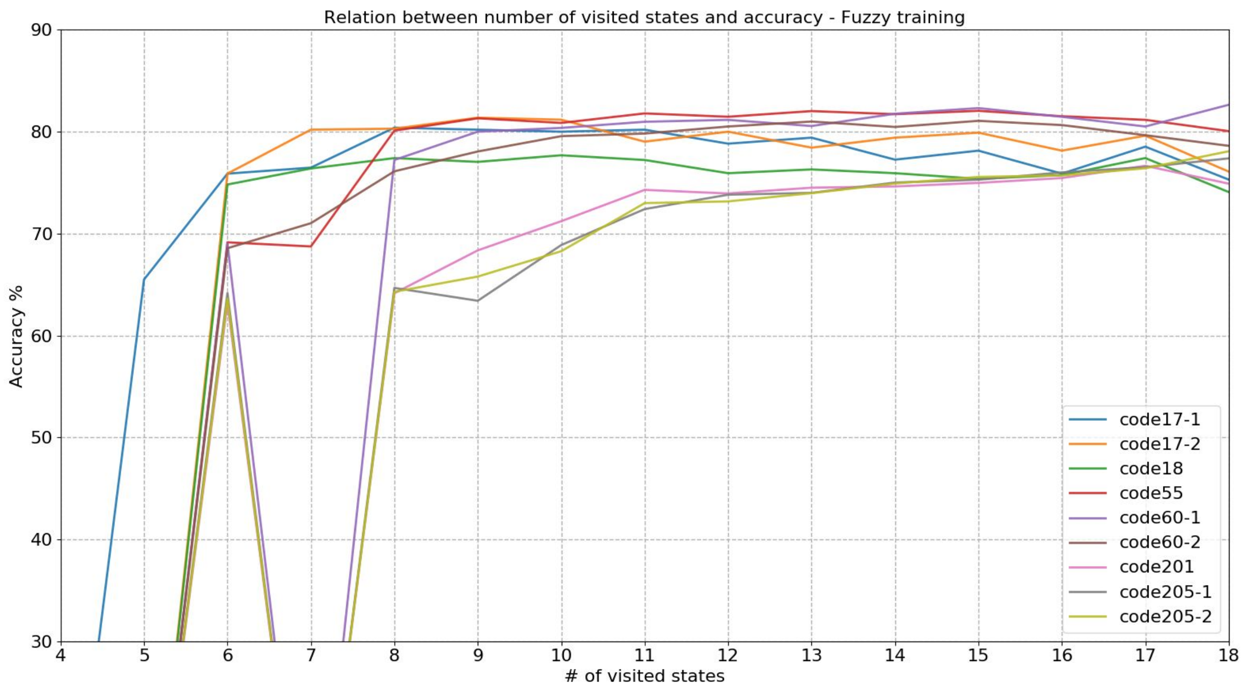

Figure 7.3: Role of machine size (visited number of states) on accuracy over all ranges and all experiments(E1, E2, E3, and E4) for all codes in training dataset with fuzzy classification

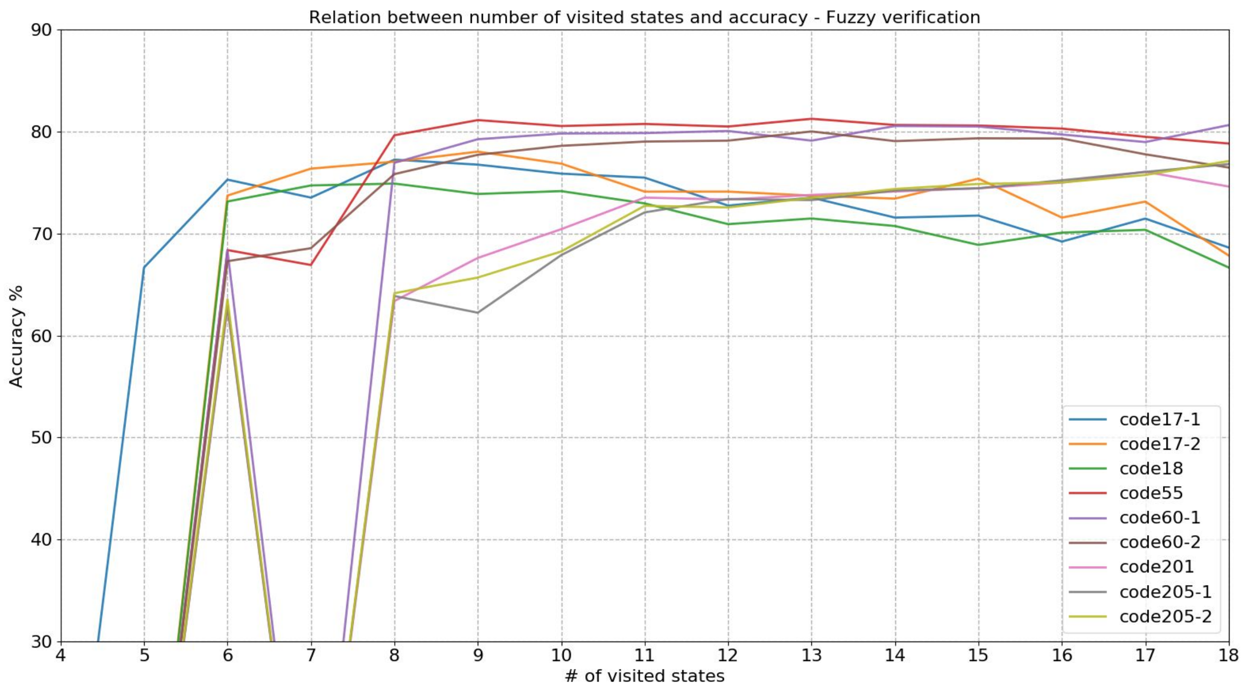

Figure 7.4: Role of machine size (visited number of states) on accuracy over all ranges and all experiments(E1, E2, E3, and E4) for all codes in verification dataset with fuzzy classification

7.2 Different Length of Codes

Figure 7.1 also offers an insight into how machines for codes of different lengths react to EP. For any machine size, the machines for codes with the same lengths achieve very similar results in terms of decoding accuracy. This is noticed for codes of all lengths studied, namely 14 (code201, code205-2, code205-2), 12 (code55, code60-1, code60-2) and 10 (code17-1, code17-2), where the respective curves appear grouped together, often overlapping one another. The only exception to this trend is code18 which, despite having a codeword length of 10, did not achieve the same degree of accuracy as code17-1 and code17-2. This could possibly be due to the fact that the higher number of codewords (18 vs 17) that code18 has makes them packed in tighter i.e. it is harder to distinguish them from one another due to their relatively close proximity. The same effect is noticed, although on a smaller scale, with code55 vs code60-1 and code60-2.

This behavior can be further analyzed by examining Figure 7.5, which is a heatmap representation of the overall accuracy achieved by each range for each code grouped by mutation techniques during direct training. Each block represents the median fitness across 30 experiments, expressed as a percentage of the maximum fitness that equates to 100% decoding success rate. Similar graphs were generated for direct verification, fuzzy training, and fuzzy verification which are displayed in Figures 7.6, 7.7, and 7.8 respectively. As evident by the colors of their respective blocks, machines for codes of length 10 (code17-1, code17-2, code18) and length 14 (code201, code205-1, and code205-2) produce the highest and lowest accuracy scores respectively, whereas length 12 (code55-1, code55-2, code60) fares in the middle of the spectrum. This behavior is consistent with the aforementioned grouping seen in Figure 7.1 and it reinforces the notion that codes of the same length react in a similar manner to SEMs constructed using EP. It is also worth noting that the inverse proportional relationship between the code length and the success rate that is seen in Figure 7.5 is only observed for direct training, as codes of length 12 perform better than those of length 10 in Figures 7.6, 7.7, and 7.8.

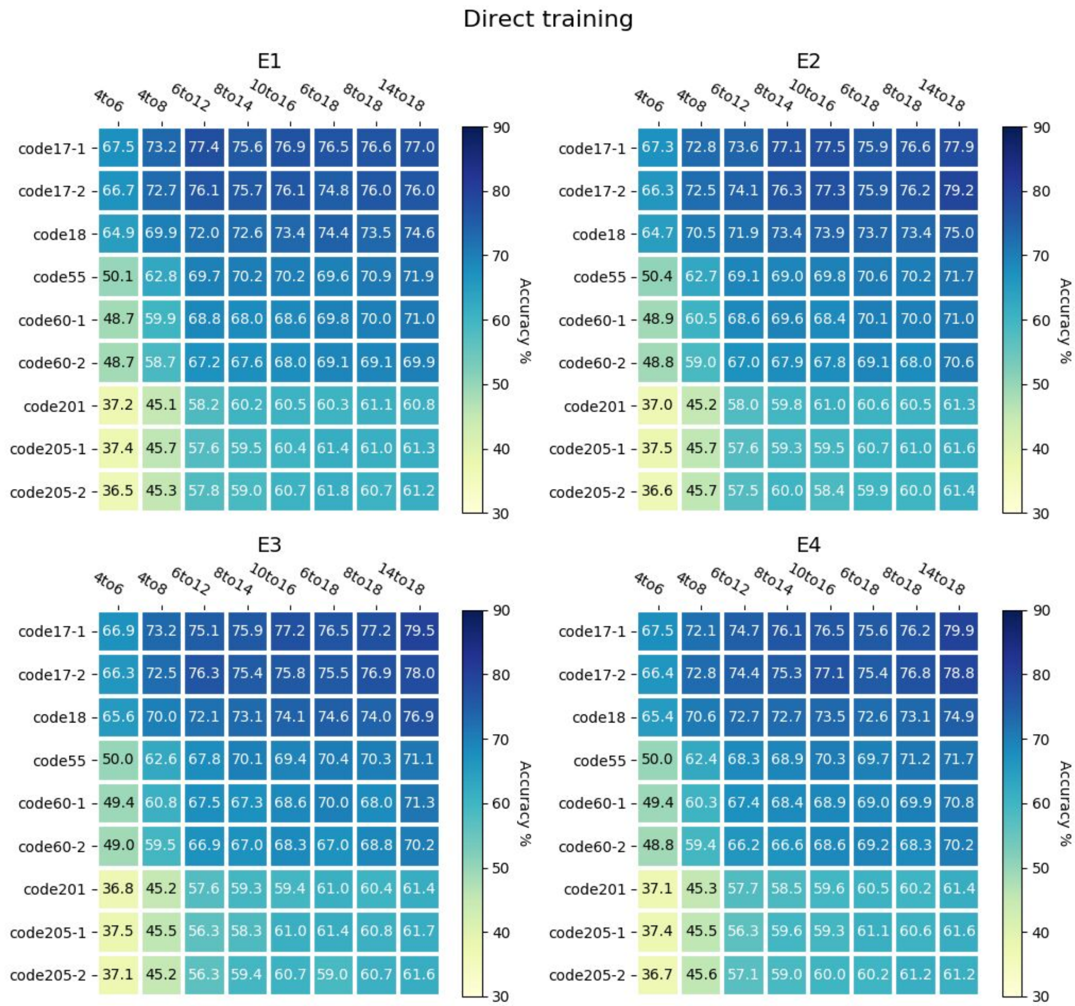

Figure 7.5: Representation of the overall accuracy achieved by each range for each code grouped by mutation techniques during direct training. Each block represents the median fitness across 30 experiments.

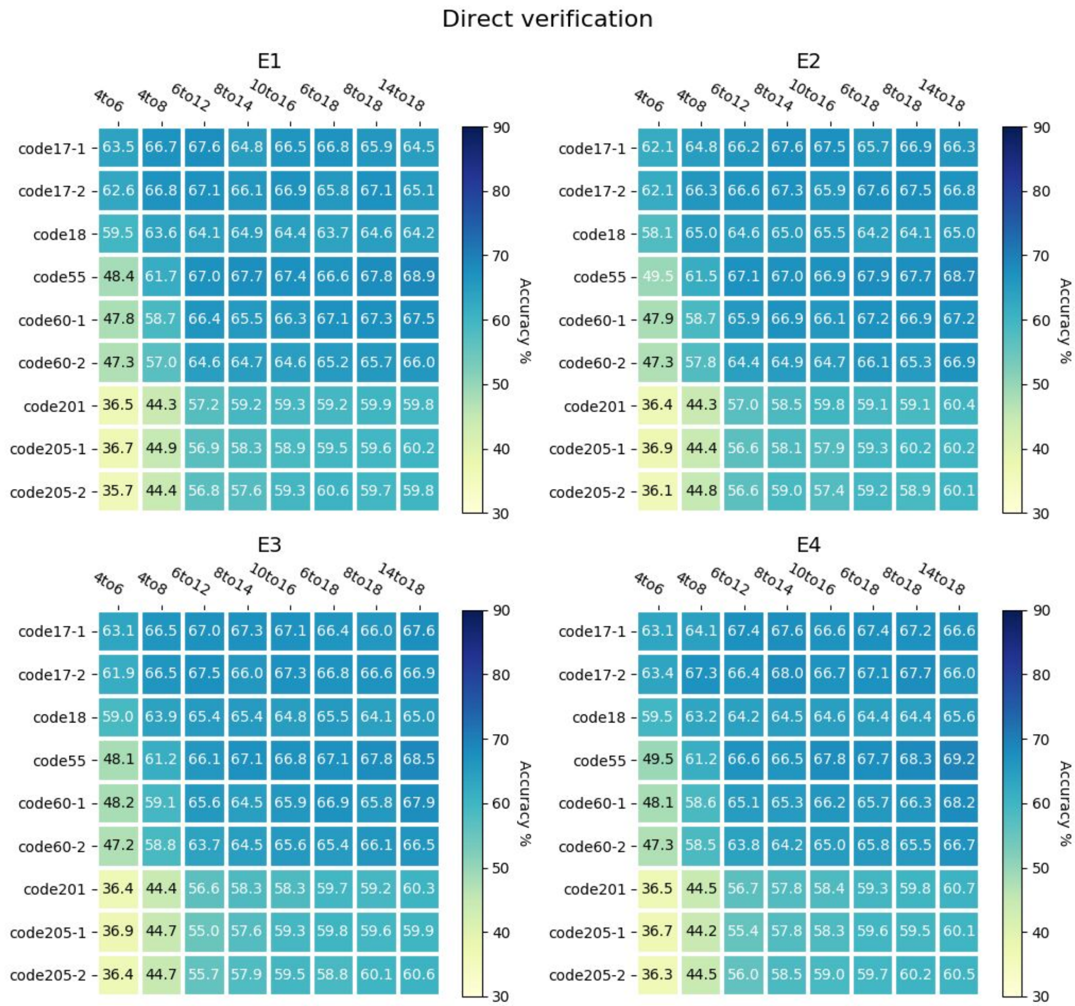

Figure 7.6: Representation of the overall accuracy achieved by each range for each code grouped by mutation techniques during direct verification. Each block represents the median fitness across 30 experiments.

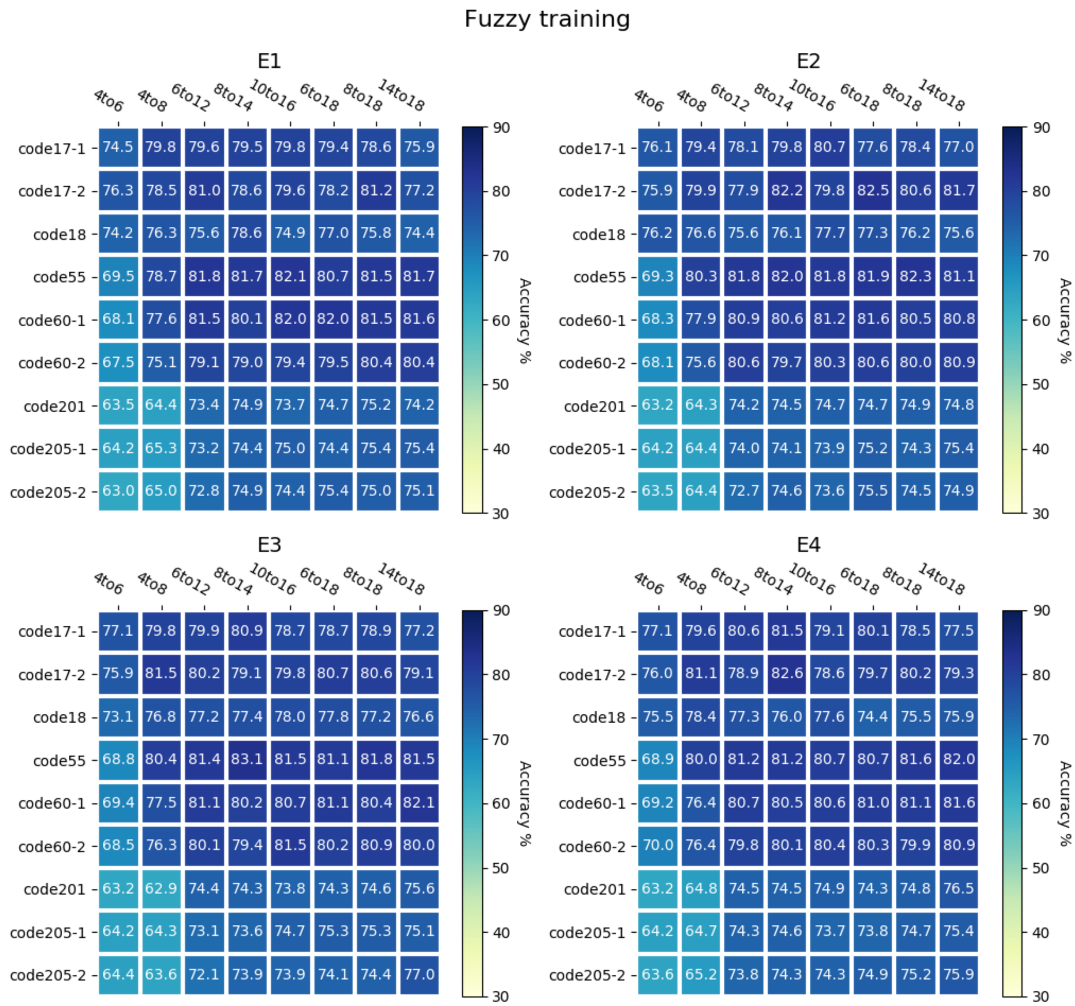

Figure 7.7: Representation of the overall accuracy achieved by each range for each code grouped by mutation techniques during fuzzy training. Each block represents the median fitness across 30 experiments.

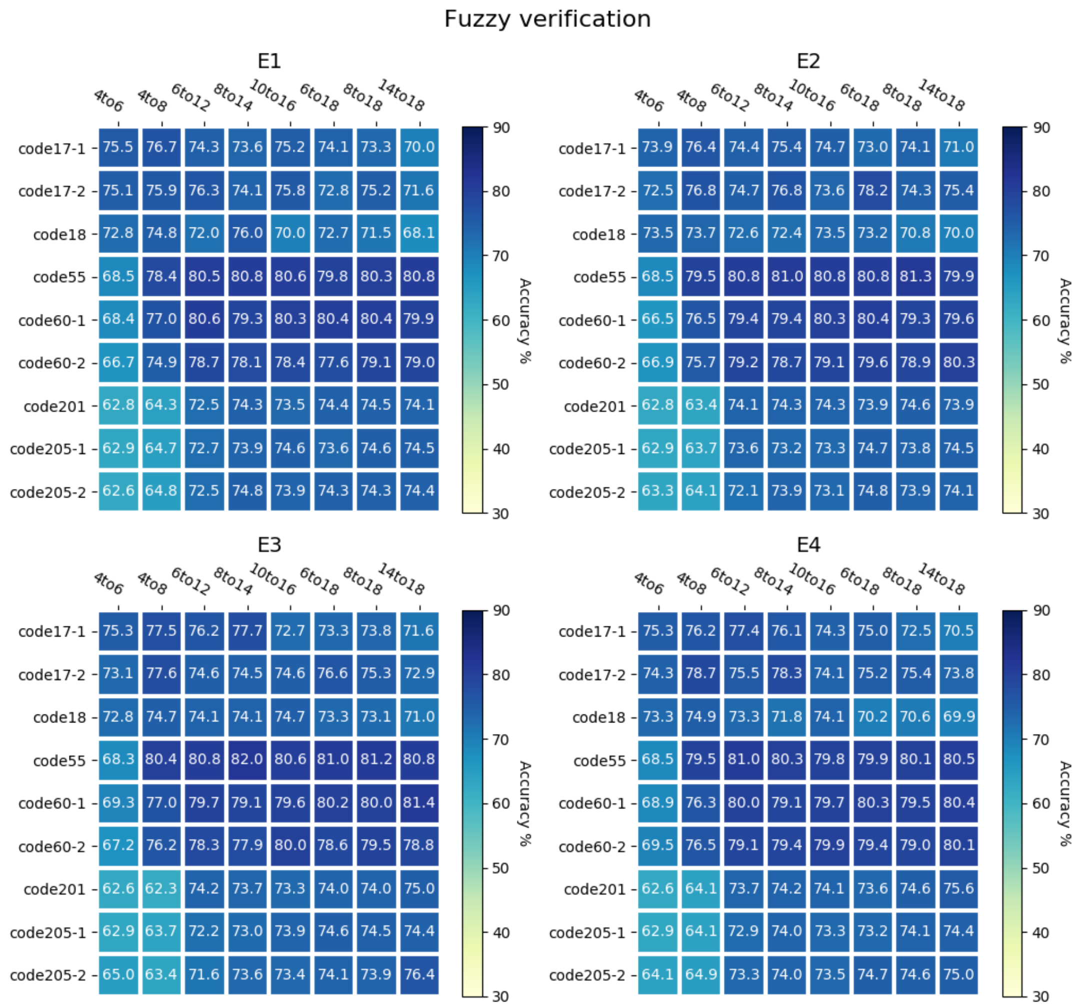

Figure 7.8: Representation of the overall accuracy achieved by each range for each code grouped by mutation techniques during fuzzy verification. Each block represents the median fitness across 30 experiments.

7.3 Mutation

As evident from Figure 7.1, the overall fitness improves with the growth of the machine size before potentially plateauing or dipping. The success rate of an SEM, therefore, depends on the mutation strategy that influences the machine size to grow up to the point that produces peak performance. To understand how the different mutation strategies influence the final machine size, the total number of machines generated for each machine size was plotted for each mutation type. As was expected, Figure 7.9 shows that the mutations that promote machine size growth, marked by the higher weights for addition, produced larger machines more often than their counterparts. In the graph, the four mutation types, denoted as E1, E2, E3, and E4 and as shown in Table 6.3, where E1 (15%) $>$ E2 (10%) $>$ E3, E4 (5%) in terms of how often they add a new state, are represented by different colors. When compared in wider ranges (ones that can accommodate a reasonable growth to take place), it can be seen that E1 consistently produced more machines with larger sizes than the others. In fact, the same trend was noticed when comparing E2 with E3 and E4.

However, the impact of such growth on the overall fitness was not conclusive. Figures 7.10 – 7.21 can be examined in order to understand the role that these mutation settings play on the overall evolution process and in turn the accuracy. Each subplot, further divided by ranges, shows the distribution of the final machine fitness for each mutation type over 30 experiments. Focusing on any particular range in any particular code, no significant difference was noticed among the four mutation types in terms of accuracy. The differences between the respective medians, if any, were slight and did not conform to any noticeable trend. A Kruskal-Wallis H-test was performed to verify this observation. The test compares the accuracy of the best machines generated by the four mutation settings, E1, E2, E3, and E4. Tables 7.3 – 7.6 show the p-values of the Kruskal-Wallis H-test. Those values less than 0.05, indicate a statistically significant difference between the groups compared. With the exception of a few that are highlighted in bold, no significant difference was noticed. In other words, the test proves that the differences in the mutation settings made no significant impact on the overall accuracy of the resulting machines.

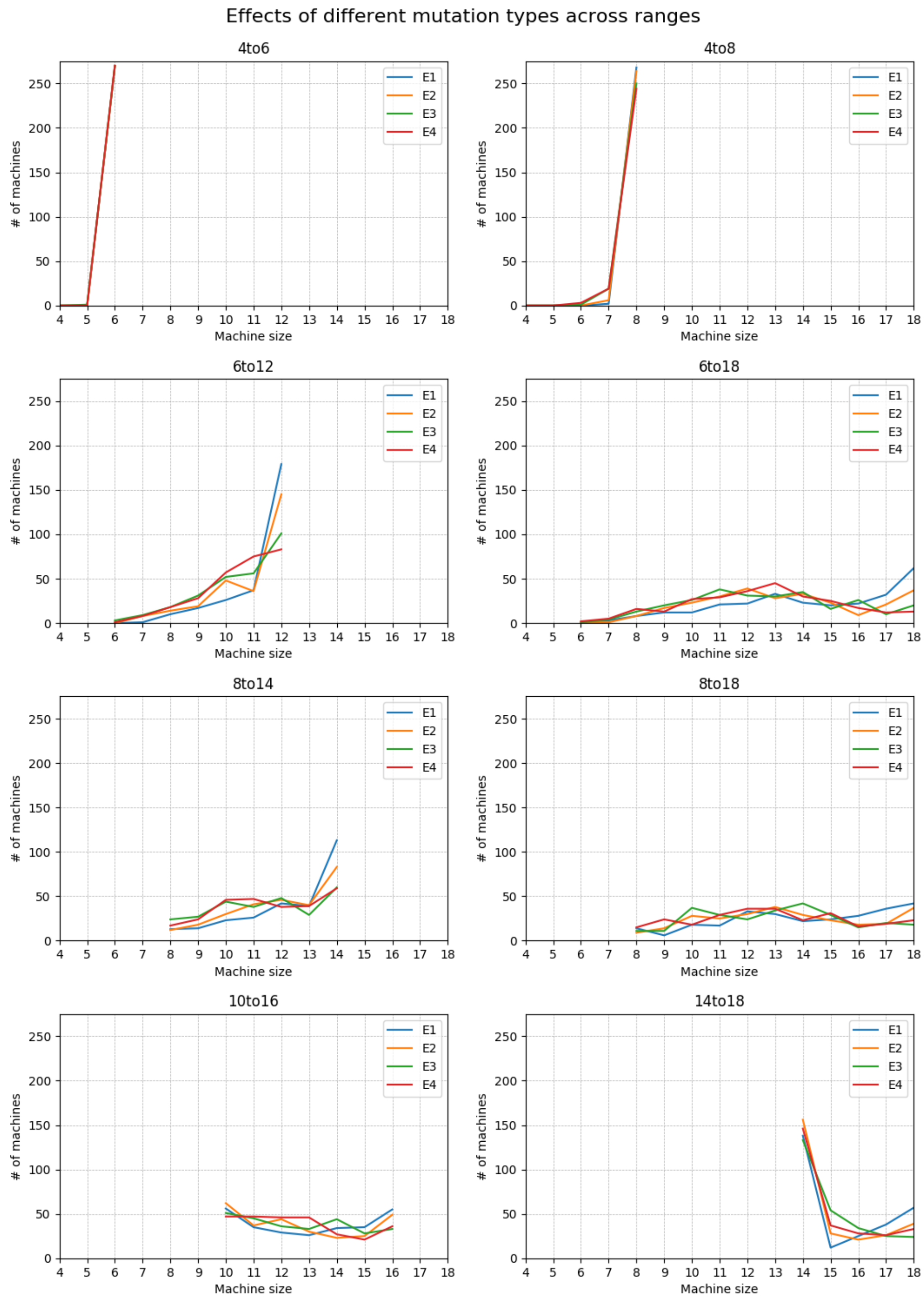

Figure 7.9: Effects of different mutation types across ranges. E1 (Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%) represent four different experiments

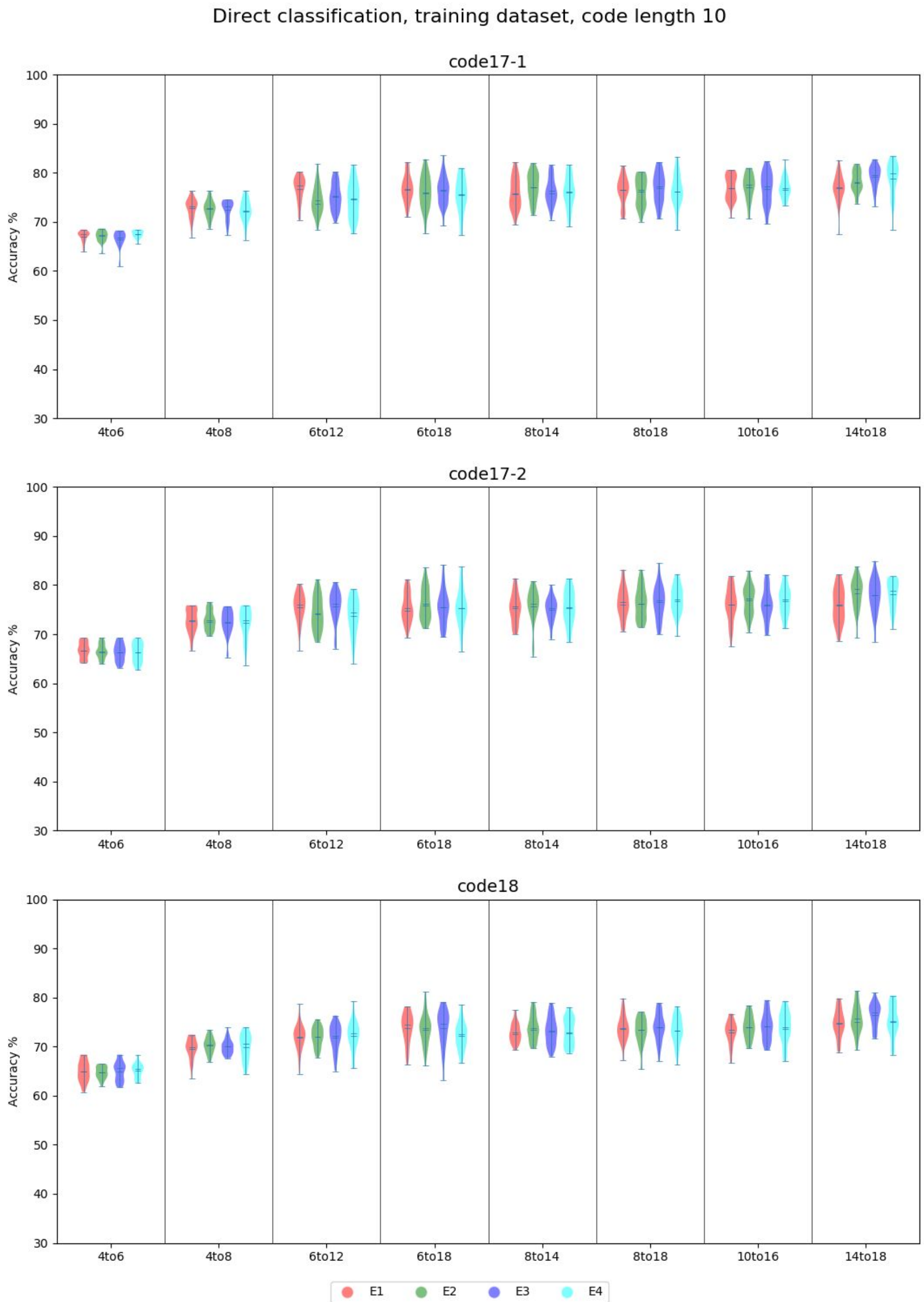

Figure 7.10: Decoding accuracy of SEMs on codes of length 10 with direct classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

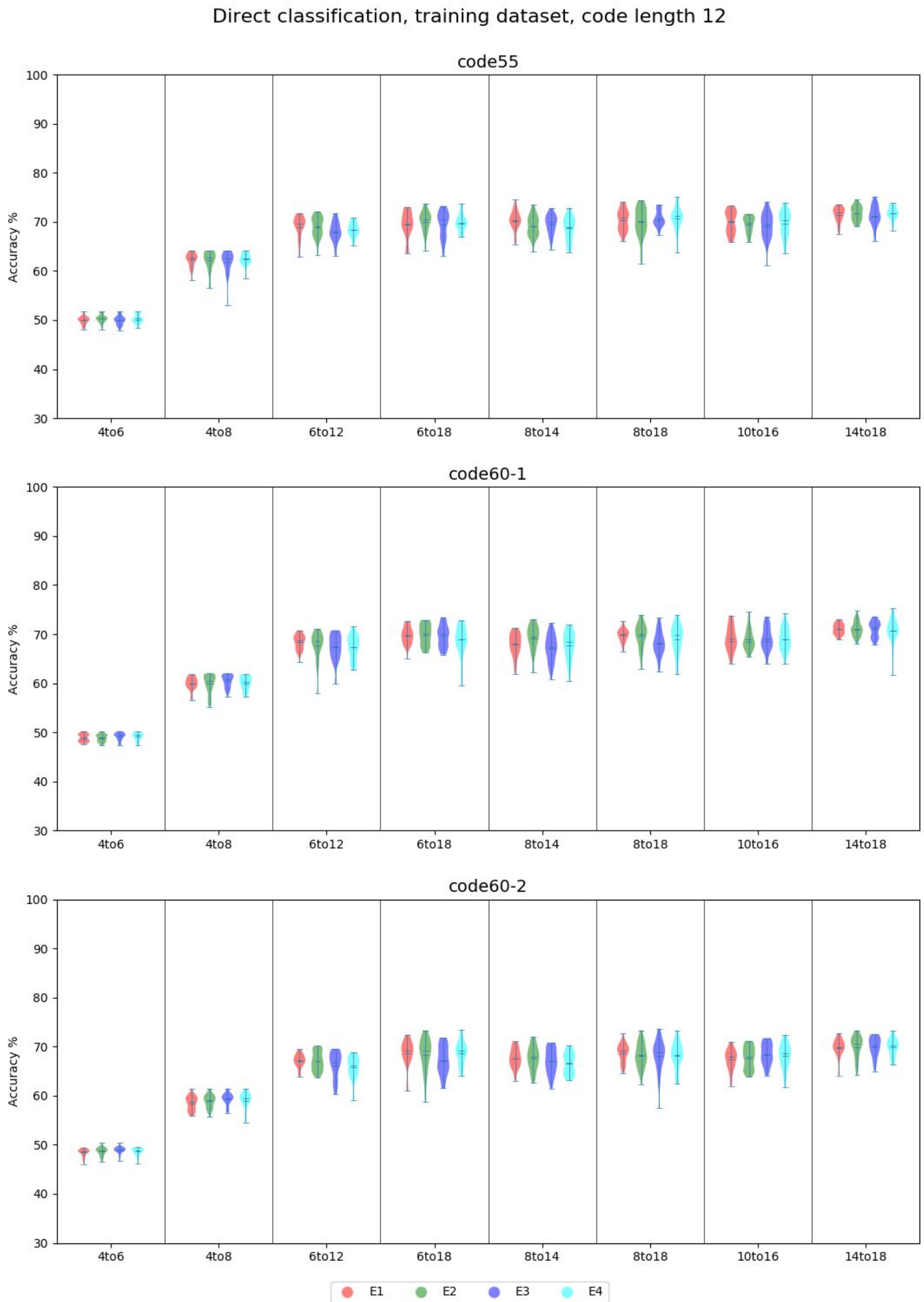

Figure 7.11: Decoding accuracy of SEMs on codes of length 12 with direct classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

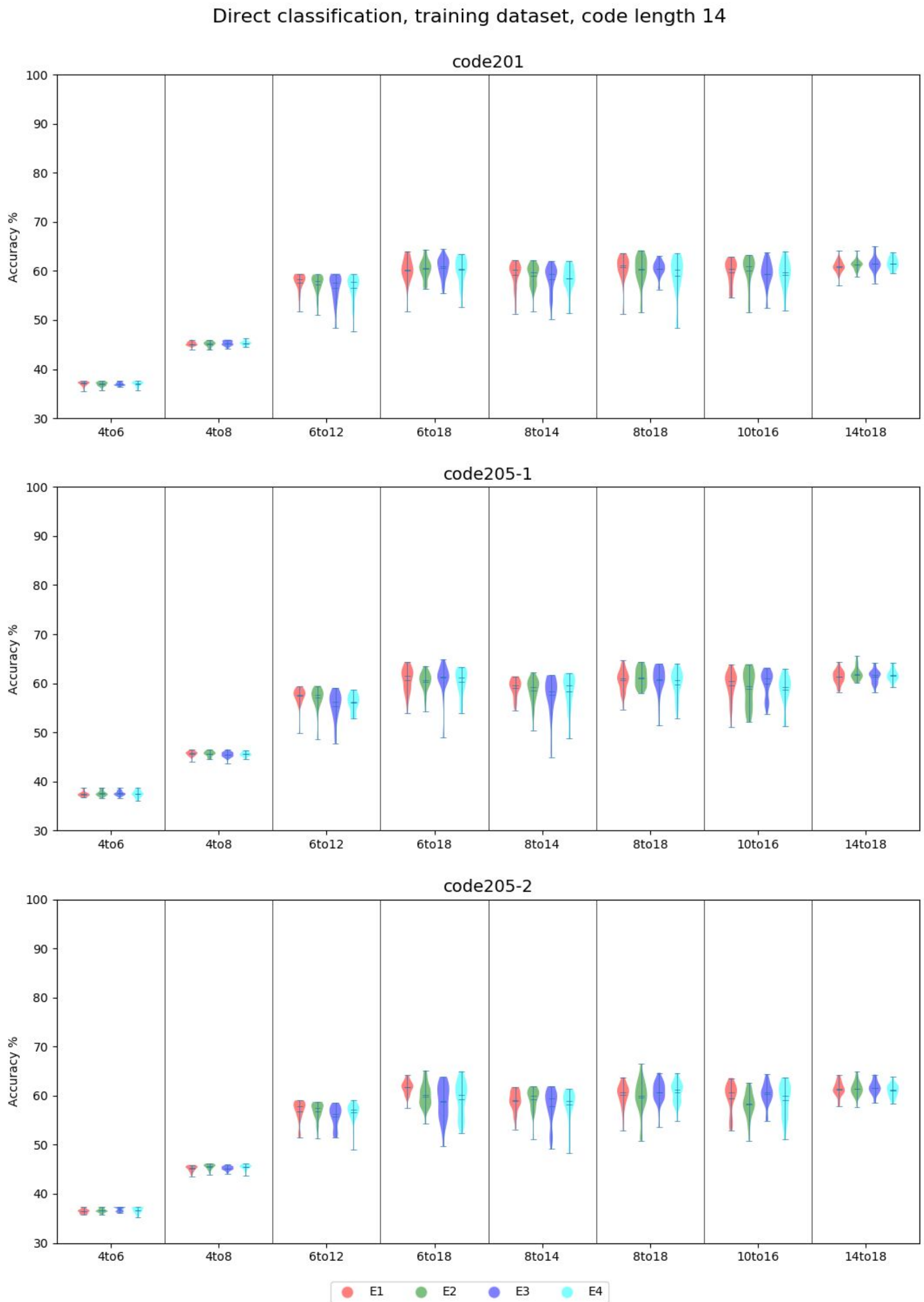

Figure 7.12: Decoding accuracy of SEMs on codes of length 14 with direct classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

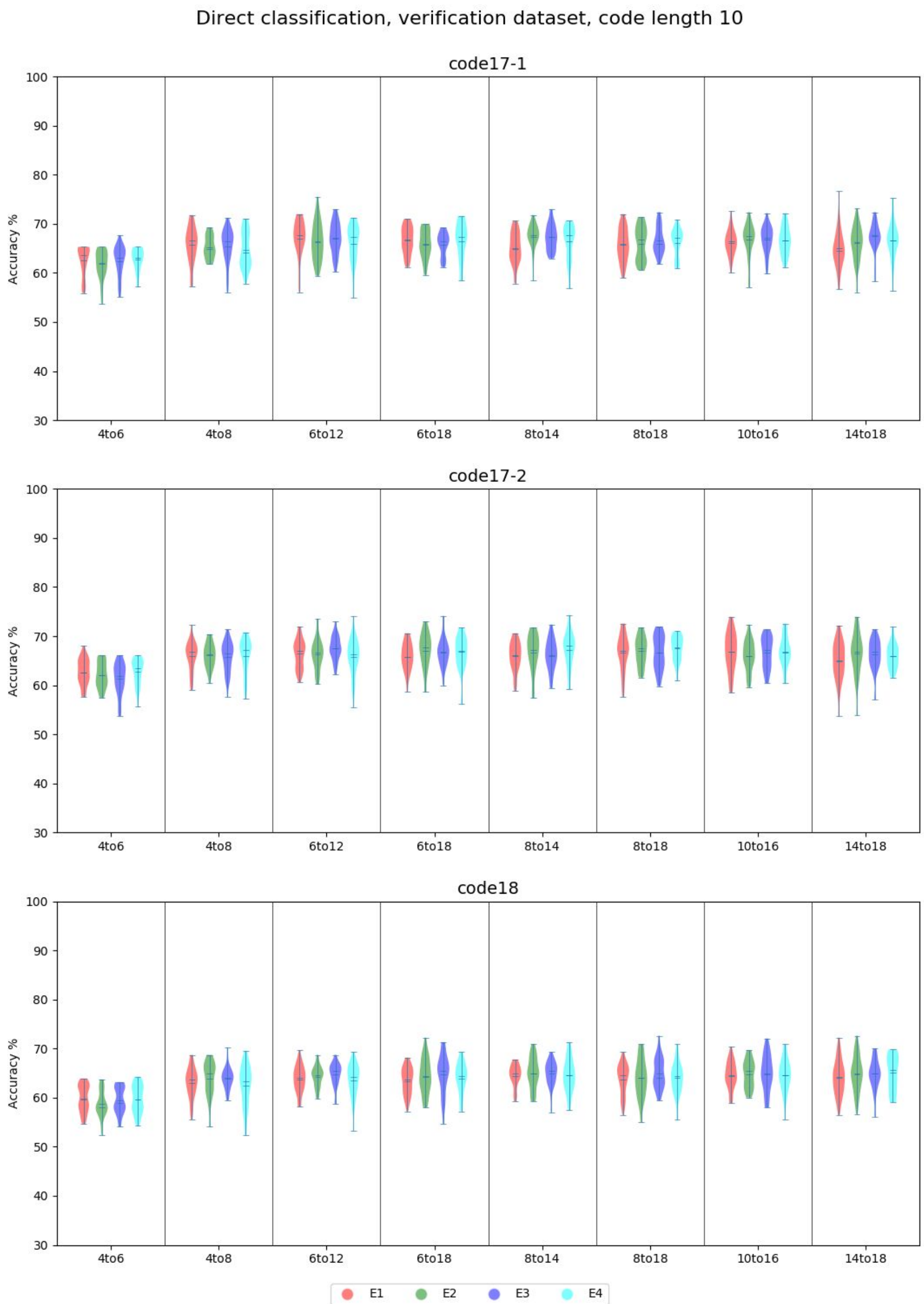

Figure 7.13: Decoding accuracy of SEMs on codes of length 10 with direct classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

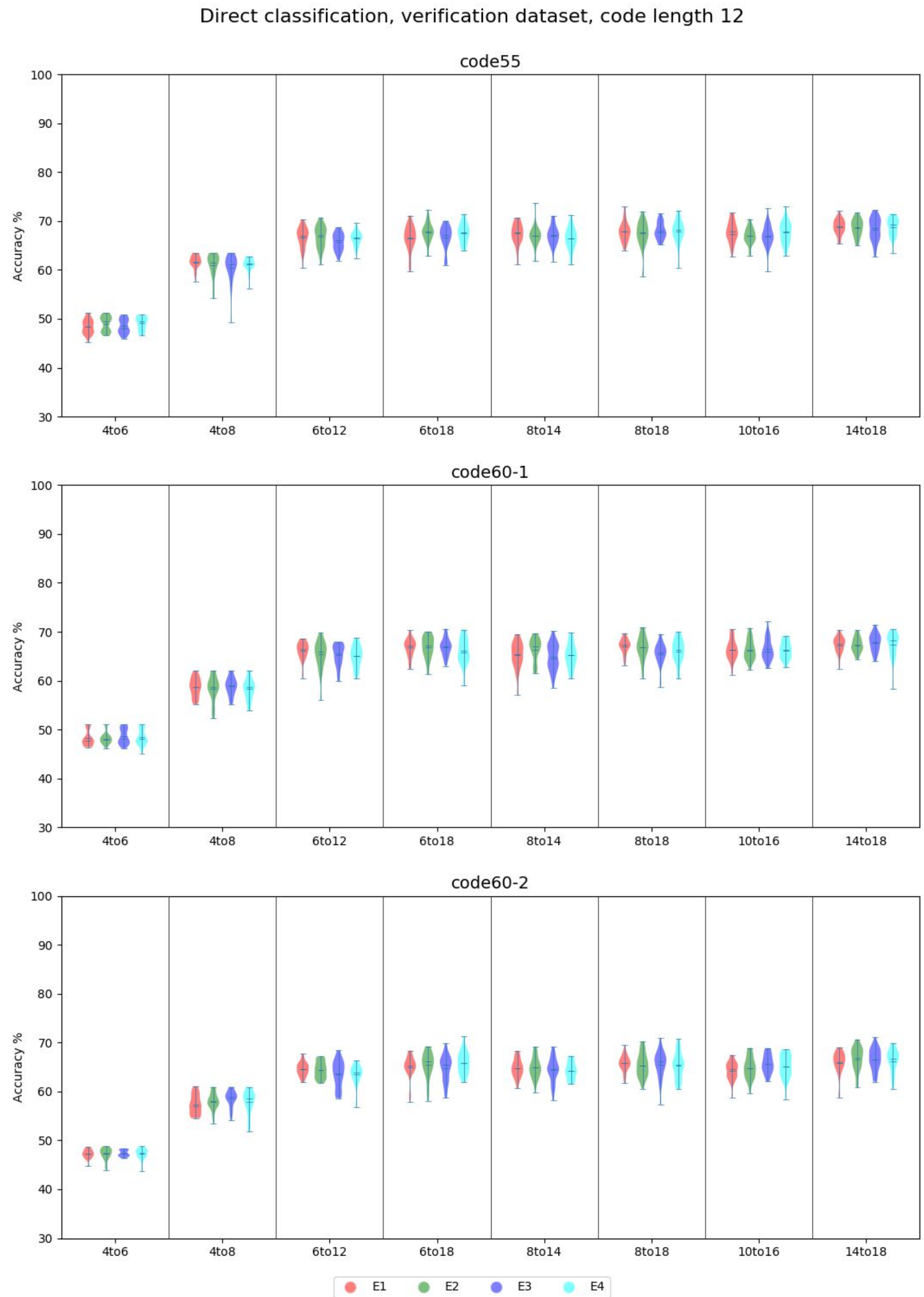

Figure 7.14: Decoding accuracy of SEMs on codes of length 12 with direct classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

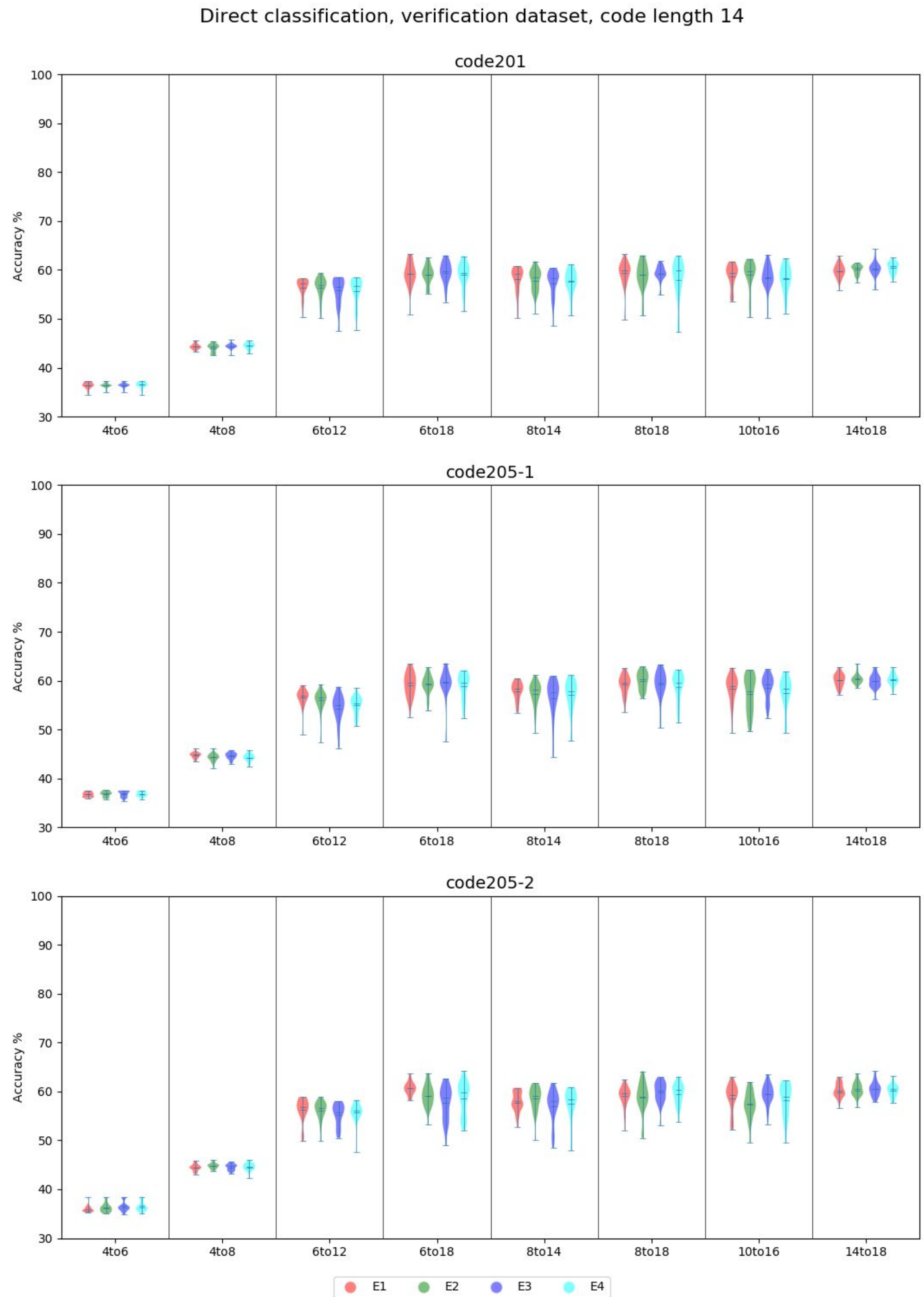

Figure 7.15: Decoding accuracy of SEMs on codes of length 14 with direct classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

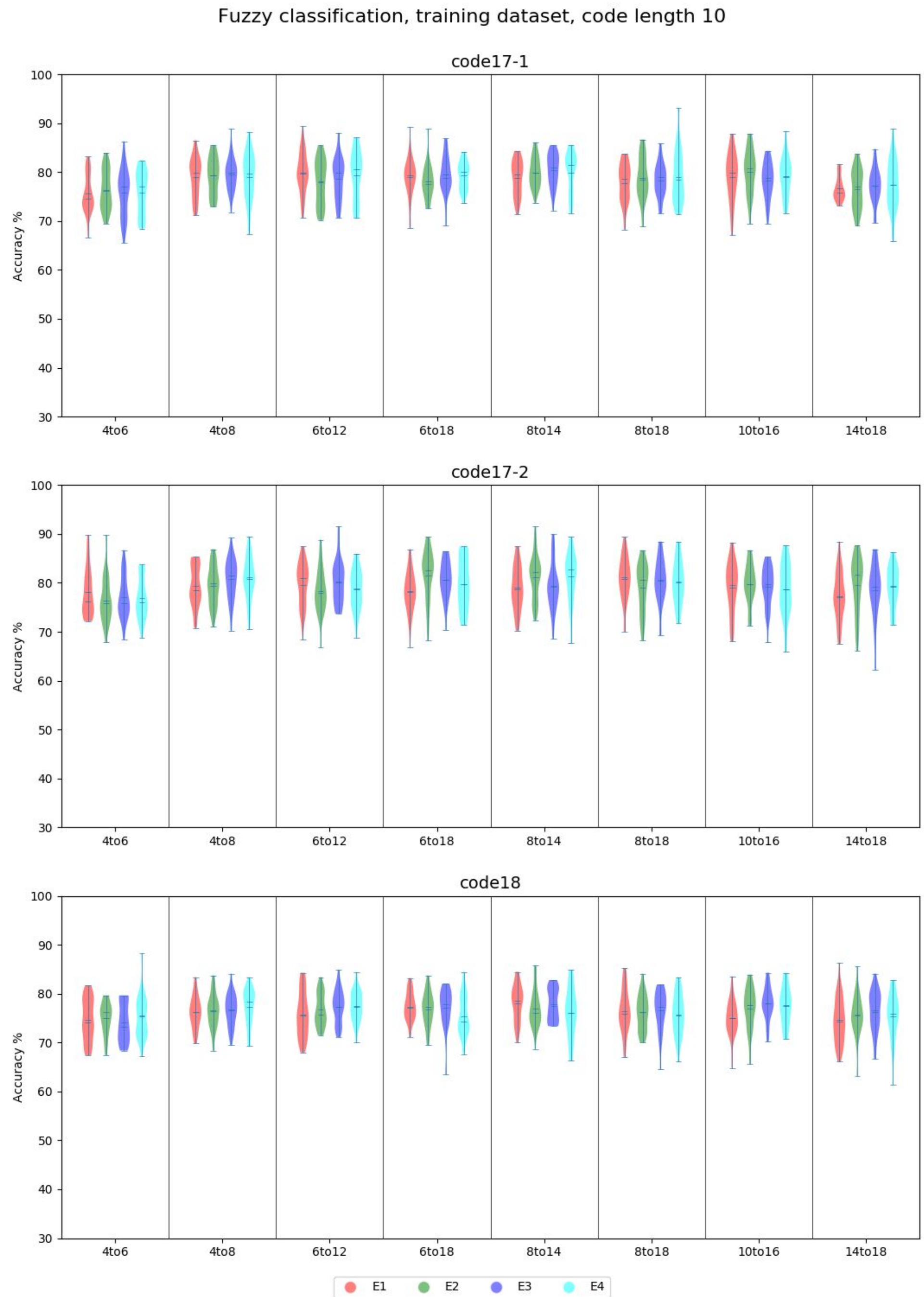

Figure 7.16: Decoding accuracy of SEMs on codes of length 10 with fuzzy classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

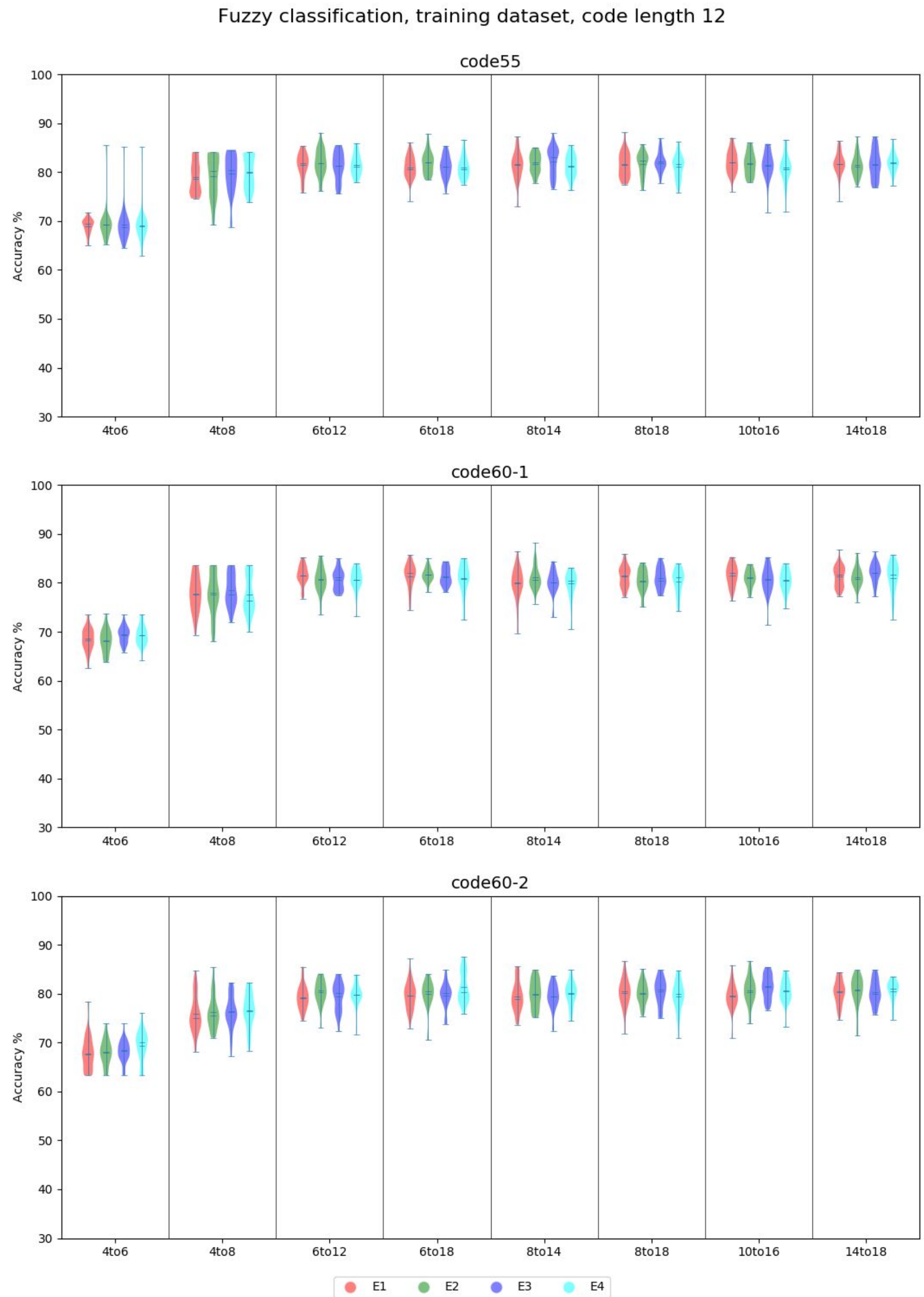

Figure 7.17: Decoding accuracy of SEMs on codes of length 12 with fuzzy classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

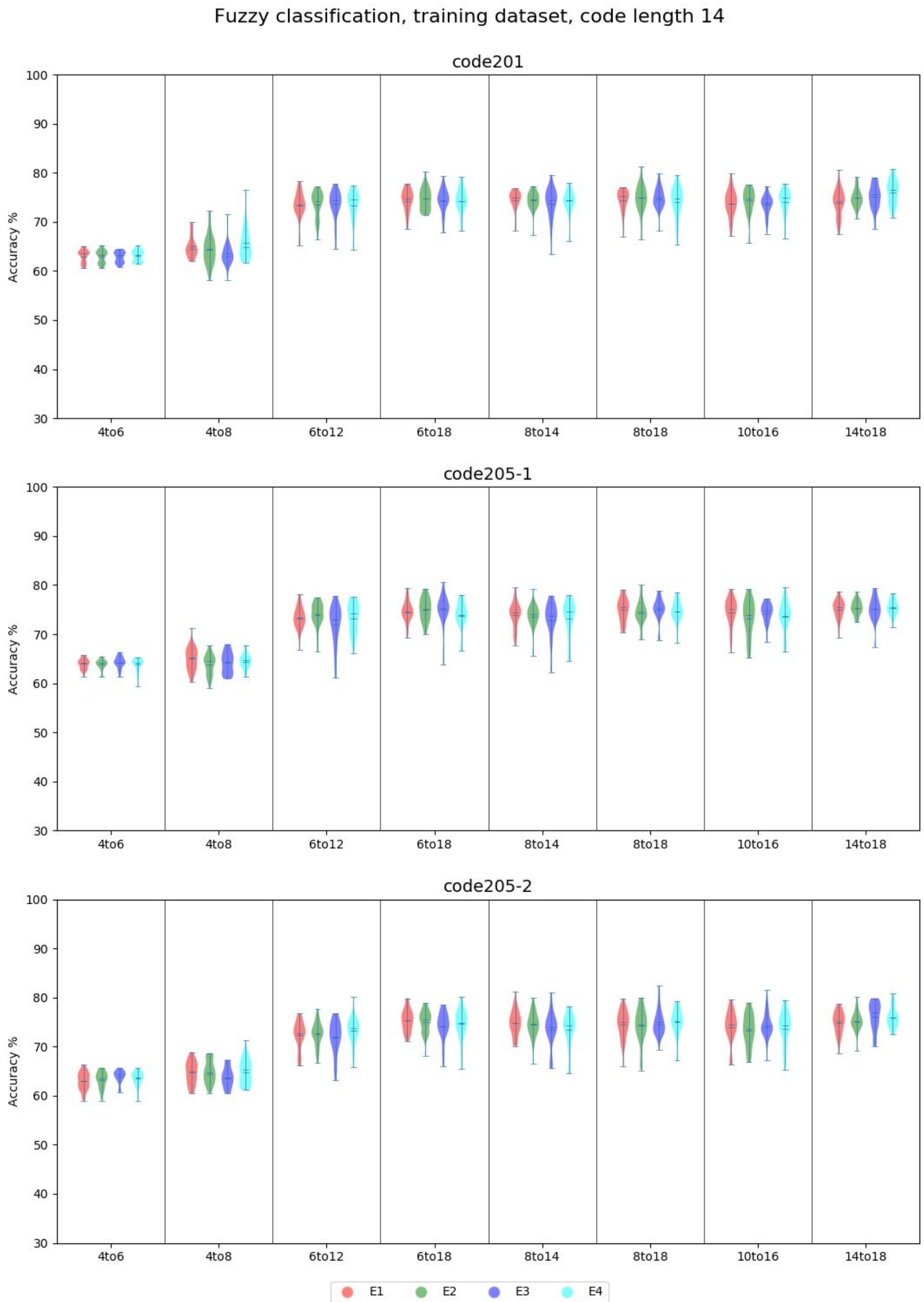

Figure 7.18: Decoding accuracy of SEMs on codes of length 14 with fuzzy classification on training dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

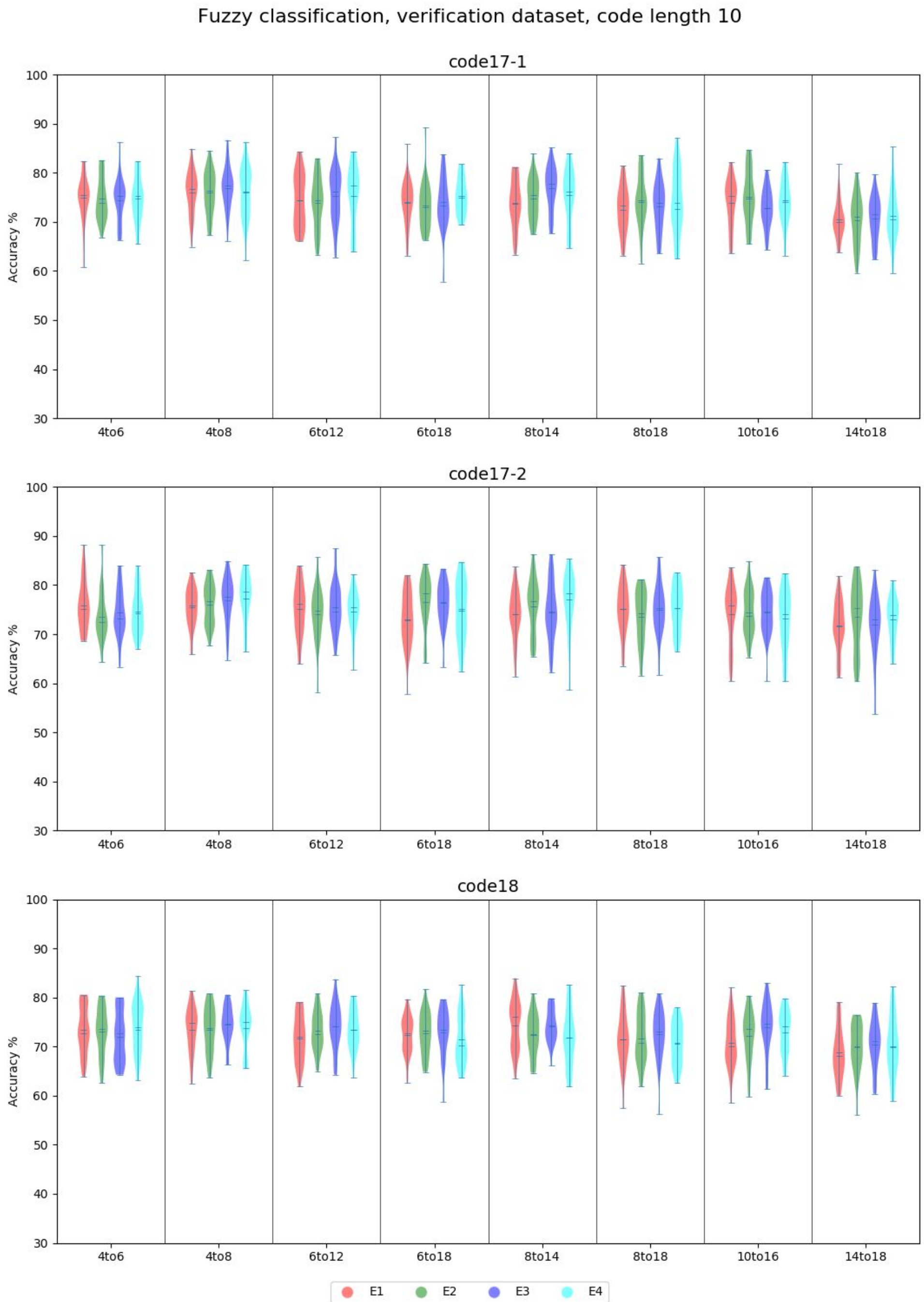

Figure 7.19: Decoding accuracy of SEMs on codes of length 10 with fuzzy classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

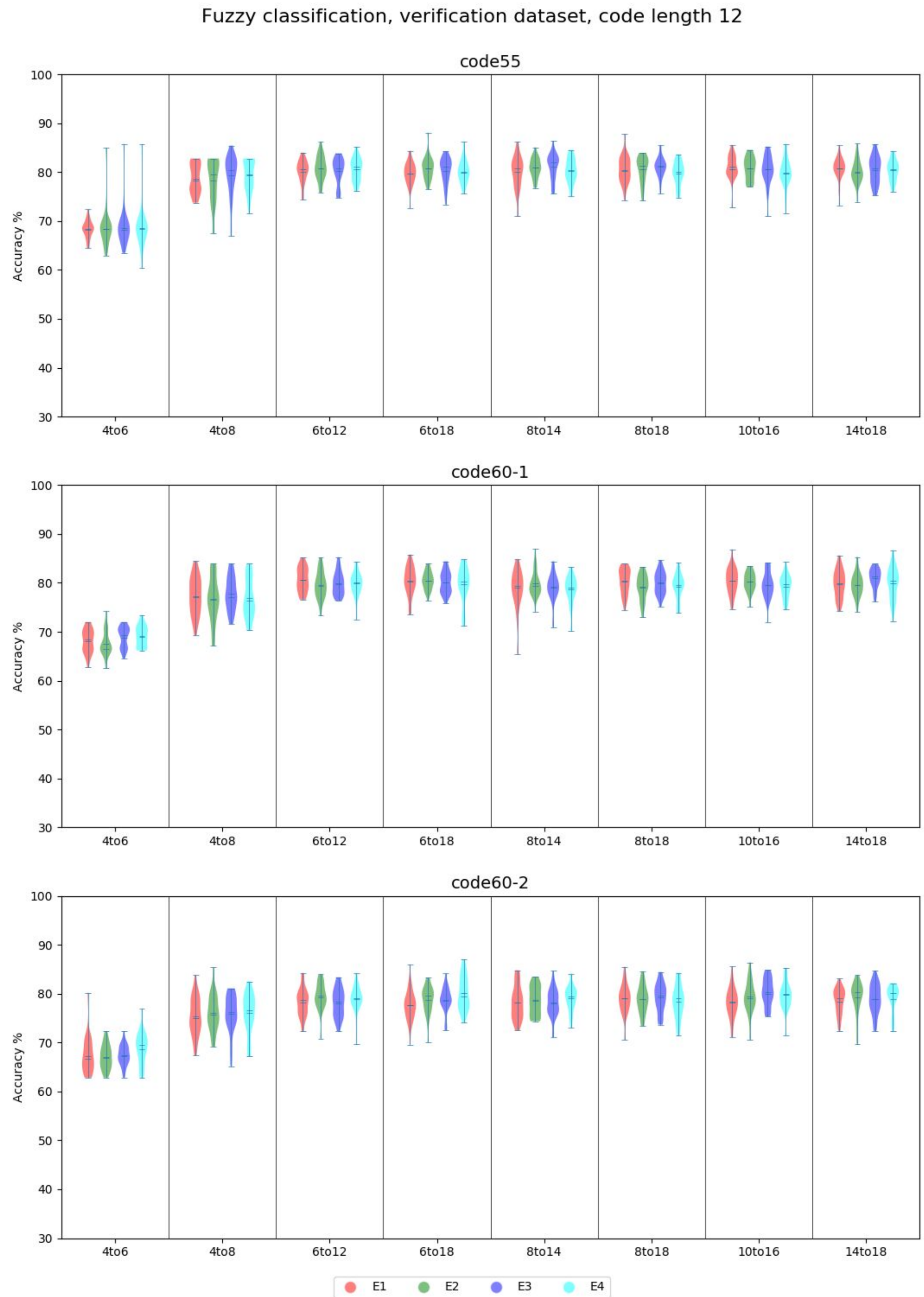

Figure 7.20: Decoding accuracy of SEMs on codes of length 12 with fuzzy classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

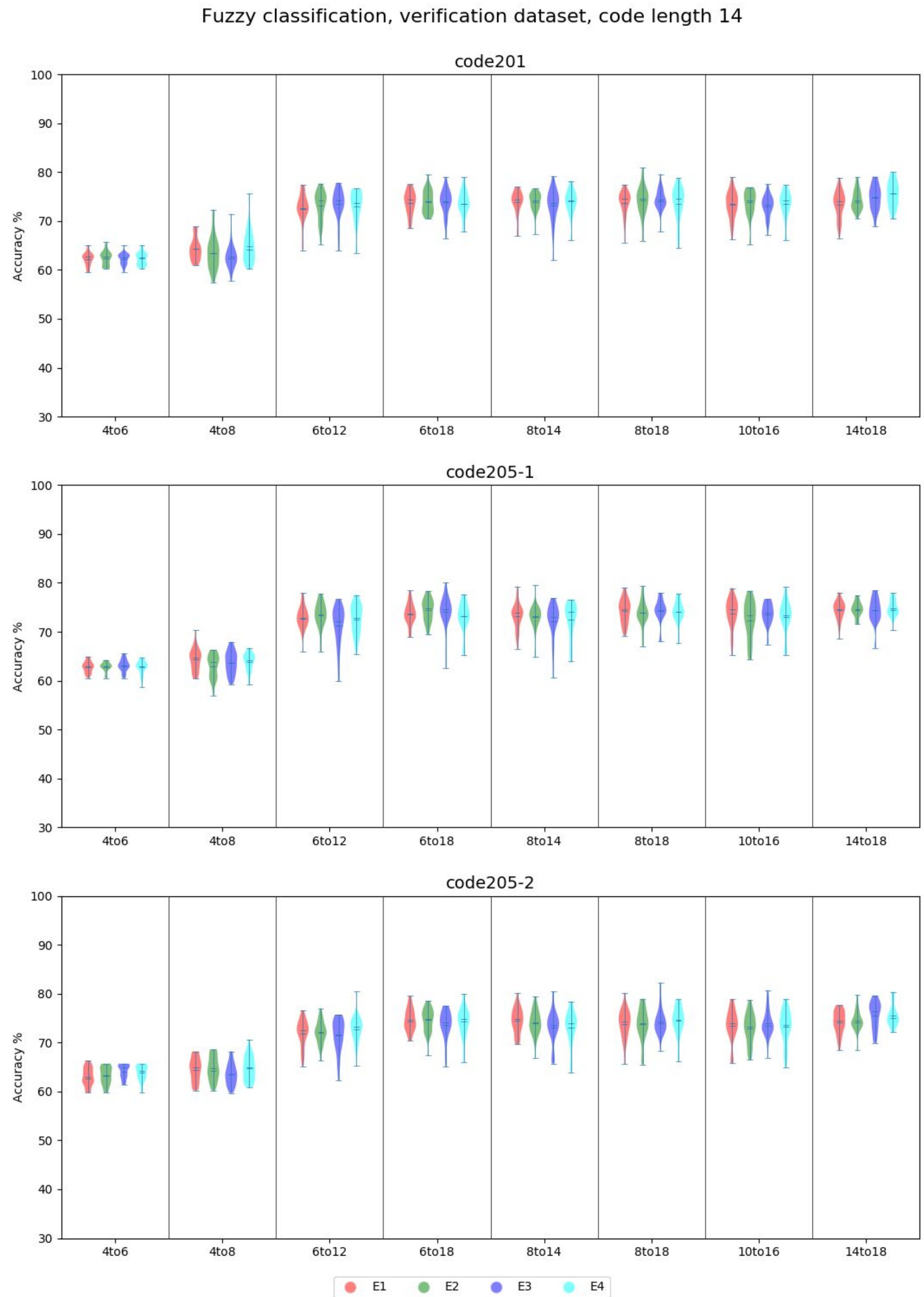

Figure 7.21: Decoding accuracy of SEMs on codes of length 14 with fuzzy classification on verification dataset across four experiments, E1(Edge 60%, Start 10%, Add 15%, Remove 15%), E2 (Edge 75%, Start 5%, Add 10%, Remove 10%), E3 (Edge 80%, Start 10%, Add 5%, Remove 5%), and E4 (Edge 85%, Start 5%, Add 5%, Remove 5%)

	4to6	4to8	6to12	6to18	8to14	8to18	10to16	14to18
code17-1	0.063	0.77	**0.017**	0.625	0.381	0.826	0.948	**0.002**
code17-2	0.803	0.946	0.11	0.772	0.625	0.714	0.707	**0.038**
code18	0.906	0.722	0.908	0.071	0.553	0.773	0.372	0.228
code55	0.2	0.881	0.076	0.794	0.078	0.755	0.498	0.728
code60-1	0.192	0.492	0.326	0.685	**0.034**	**0.043**	0.98	0.993
code60-2	0.056	0.104	0.134	0.183	0.248	0.818	0.276	0.866
code201	0.681	0.44	0.147	0.901	0.302	0.381	0.286	0.2
code205-1	0.641	0.337	**0.025**	0.299	0.643	0.504	0.299	0.529
code205-2	0.063	**0.001**	**0.033**	**0.002**	0.343	0.567	**0.034**	0.436

Table 7.3: P-values of Kruskal-Wallis H-test to observe the impact of mutation settings on accuracy with direct classification (training dataset)

	4to6	4to8	6to12	6to18	8to14	8to18	10to16	14to18
code17-1	0.451	0.606	0.6	0.421	**0.028**	0.82	0.523	**0.013**
code17-2	0.305	0.899	0.239	0.386	0.301	0.828	0.744	0.389
code18	0.5	0.336	0.351	0.333	0.847	0.673	0.957	0.621
code55	0.235	0.459	0.111	0.225	0.3	0.895	0.37	0.855
code60-1	0.746	0.746	0.283	0.66	0.11	**0.021**	0.984	0.601
code60-2	0.912	0.142	0.185	0.452	0.691	0.703	0.109	0.522
code201	0.448	0.227	0.697	0.828	0.359	0.702	0.235	0.227
code205-1	0.919	**0.034**	**0.025**	0.697	0.776	0.495	0.496	0.637
code205-2	**0.009**	0.229	0.111	**0.002**	0.286	0.414	**0.034**	0.825

Table 7.4: P-values of Kruskal-Wallis H-test to observe the impact of mutation settings on accuracy with direct classification (verification dataset)

	4to6	4to8	6to12	6to18	8to14	8to18	10to16	14to18
code17-1	0.951	0.988	0.536	0.209	0.304	0.857	0.4	0.833
code17-2	0.701	0.334	0.411	**0.026**	0.082	0.641	0.948	0.208
code18	0.792	0.501	0.283	0.101	0.389	0.721	**0.026**	0.366
code55	0.886	0.699	0.862	0.459	0.375	0.48	0.548	0.756
code60-1	0.128	0.817	0.3	0.817	0.596	0.519	0.43	0.399
code60-2	0.094	0.668	0.302	0.229	0.756	0.65	0.072	0.79
code201	0.942	**0.028**	0.842	0.982	0.879	0.807	0.264	**0.036**
code205-1	0.666	0.267	0.203	0.199	0.849	0.458	0.377	0.951
code205-2	0.196	0.275	0.365	0.625	0.38	0.745	0.764	0.122

Table 7.5: P-values of Kruskal-Wallis H-test to observe the impact of mutation settings on accuracy with fuzzy classification (training dataset)

	4to6	4to8	6to12	6to18	8to14	8to18	10to16	14to18
code17-1	0.834	0.903	0.729	0.275	0.086	0.598	0.394	0.954
code17-2	0.35	0.215	0.853	0.053	0.101	0.726	0.881	0.439
code18	0.682	0.895	0.262	0.432	0.15	0.474	0.072	0.463
code55	0.987	0.48	0.788	0.388	0.385	0.285	0.348	0.577
code60-1	0.054	0.759	0.434	0.919	0.828	0.24	0.268	0.145
code60-2	0.117	0.722	0.164	**0.041**	0.581	0.71	0.058	0.467
code201	0.976	**0.046**	0.484	0.92	0.824	0.865	0.549	**0.035**
code205-1	0.78	0.241	0.178	0.179	0.92	0.683	0.502	0.982
code205-2	0.082	0.271	0.532	0.748	0.387	0.685	0.83	0.065

Table 7.6: P-values of Kruskal-Wallis H-test to observe the impact of mutation settings on accuracy with fuzzy classification (verification dataset)

7.4 Total States vs Visited States

Previous studies have discussed the bloat that manifests itself in the best machines due to the presence of unused states. In its conclusion, [31] suggested that the best machines should be investigated further to find the visited state count, an important analysis not witnessed in previous work. This information can help simplify the machines by excluding the not visited states. This analysis also bears significance in studying whether the mutation algorithms contribute to further inflating the bloat. The bubble chart shown in Figure 7.22 plots the final SEMs machine size (total states) against the number of states that were visited. Samples across runs for all codes and mutation types were combined and grouped by range. Machines with one or more unused states appear below the x = y line, while the ones which had all of their states visited appear on it. The size of the bubbles reflects the size of the data points, i.e. the number of machines that appear at a given coordinate. No machines ever appear above the line as the number of visited states cannot be larger than the numbers of total states. This graph provides a key insight into the effectiveness of the mutation strategies used — whether they are influencing the machines to evolve towards a truly good solution or simply bloating the machines without improving their fitness.

As can be seen from the graphs, a fair number of the SEMs featured one or more unused states. This bloat was most noticeable in larger machines and was rare in smaller machines. This is evident by the higher occurrences of such machines in ranges with the upper bound of 18 (6 to 18, 8 to 18, and 14 to 18) than others. On the other hand, all states were visited in machines generated with ranges 4 to 6 and 4 to 8. This can be explained by the lack of opportunity a machine has to be able to grow and improve while being constricted by a smaller bound.

This plot also exposes a potential flaw in the mutation algorithm, where growth in number of states does not always lead to improved fitness, but instead adds unnecessary bulk. This is due to the fact that, while adding a new state the mutation algorithm only creates outgoing transitions from it. Therefore, a newly added state can be visited only if the machine survives elimination and is selected for further mutation that creates one or more incoming transitions to it. Even then it might remain not visited unless the transition condition is met. In addition, because of the random nature of which state or transition gets mutated, there is a high possibility that this machine either never gets mutated in a favorable way to include the new state or it gets eliminated due to poor fitness.

Plots for the final SEMs machine size (total states) against the number of visited states across all experiments for each codes are presented in Appendix B.2. Figure 7.23 – 7.25 show the distribution of machine sizes over eight different ranges for each length of code, 10, 12, and 14. It is observed that as the length of code increases, the SEMs are generating machines with bigger size. It supports the fact that SEMs for longer codes tend to need more states which was shown in [13].

Figure 7.26 shows the difference of four experiments (mutation parameter) on machine size. It shows that the mutations that promote machine size growth, marked by the higher weights for addition, produced bloats more often than their counterparts. Experiment 1 (E1), which has 15% chance of adding a state in mutation, creates the most bloat and E3 and E4, with 5%, create the least.

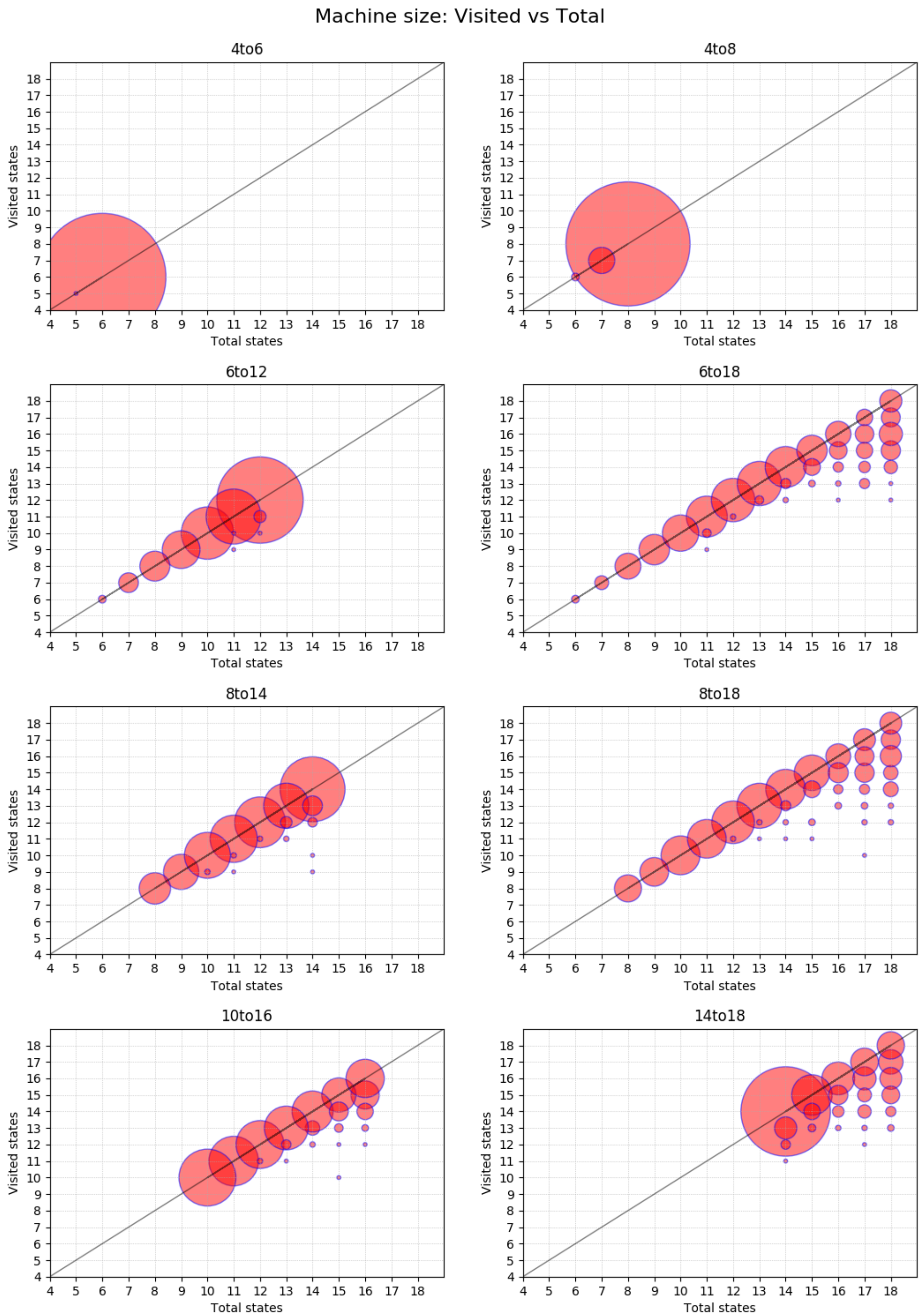

Figure 7.22: The final SEMs machine size (total states) against the number of visited states across all experiments and codes

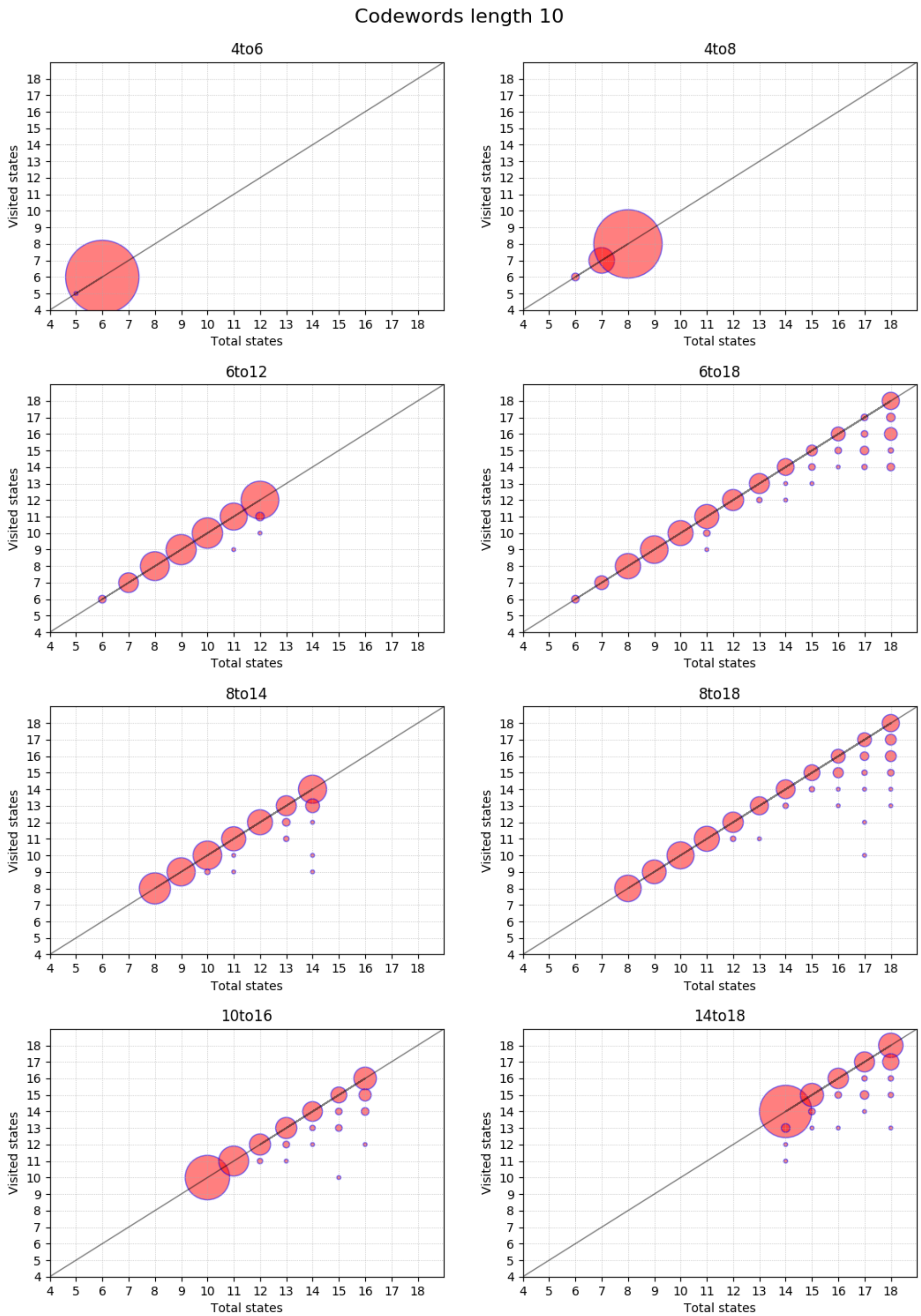

Figure 7.23: The final SEMs machine size (total states) against the number of visited states across all experiments for codewords length 10

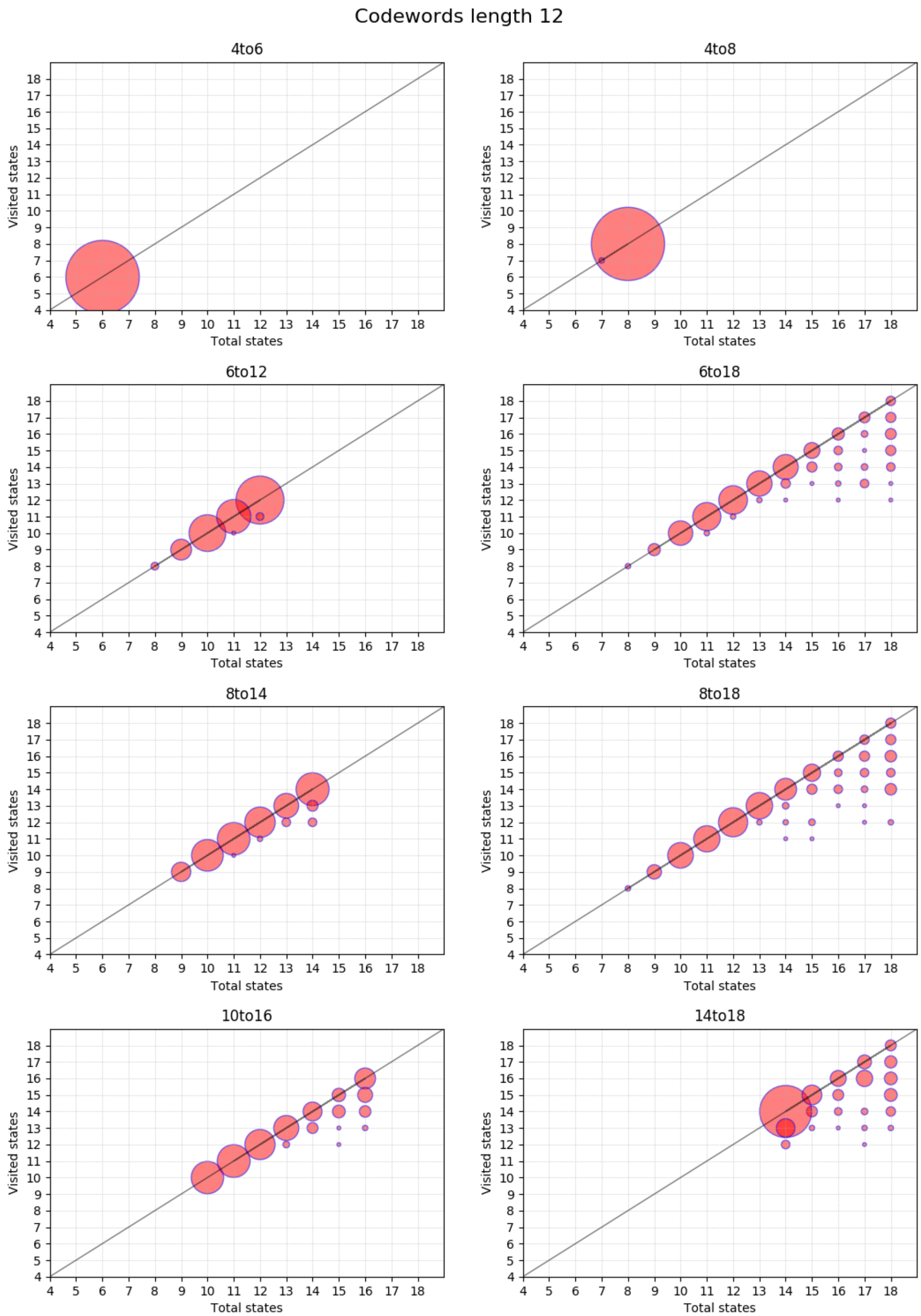

Figure 7.24: The final SEMs machine size (total states) against the number of visited states across all experiments for codewords length 12

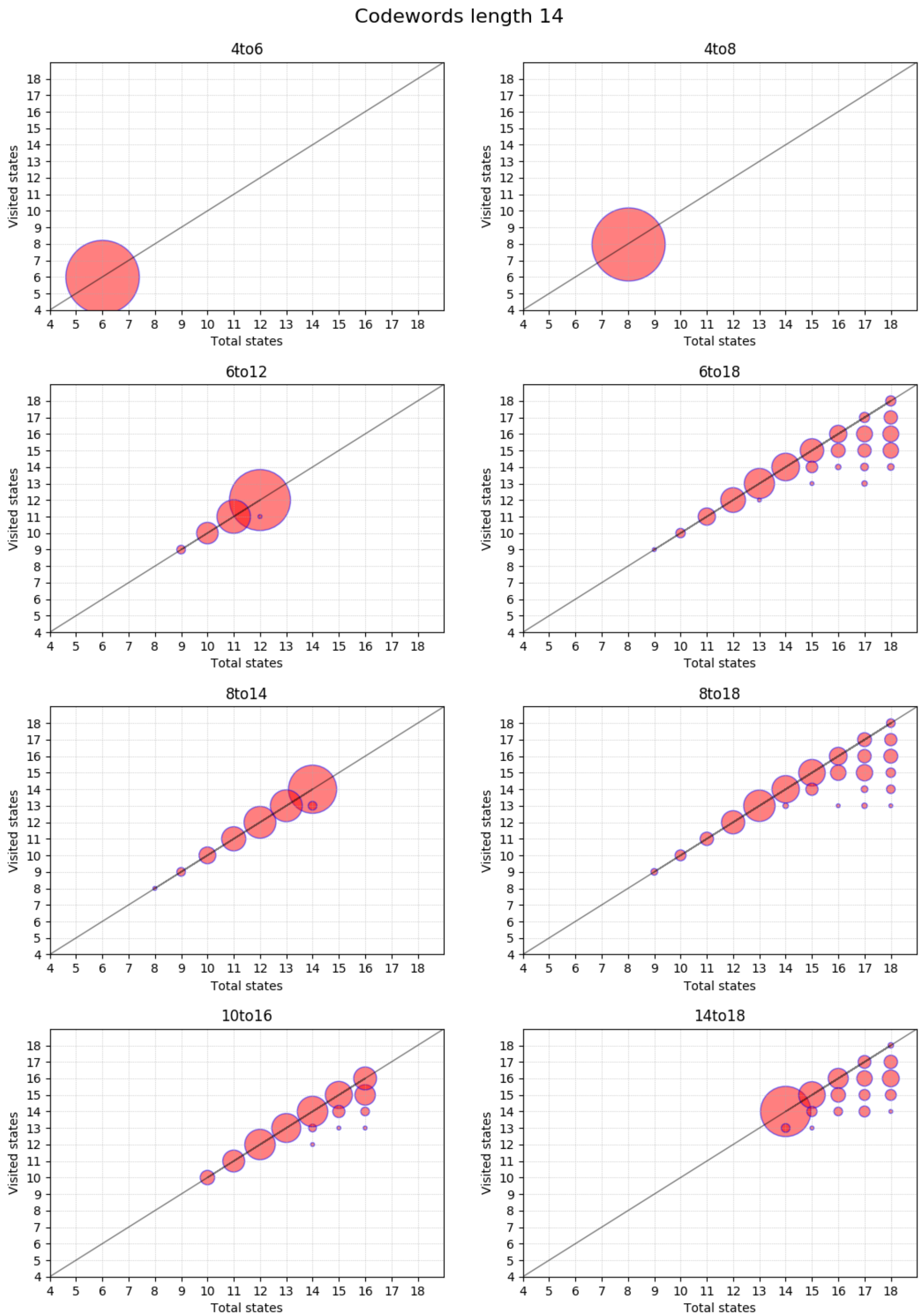

Figure 7.25: The final SEMs machine size (total states) against the number of visited states across all experiments for codewords length 14

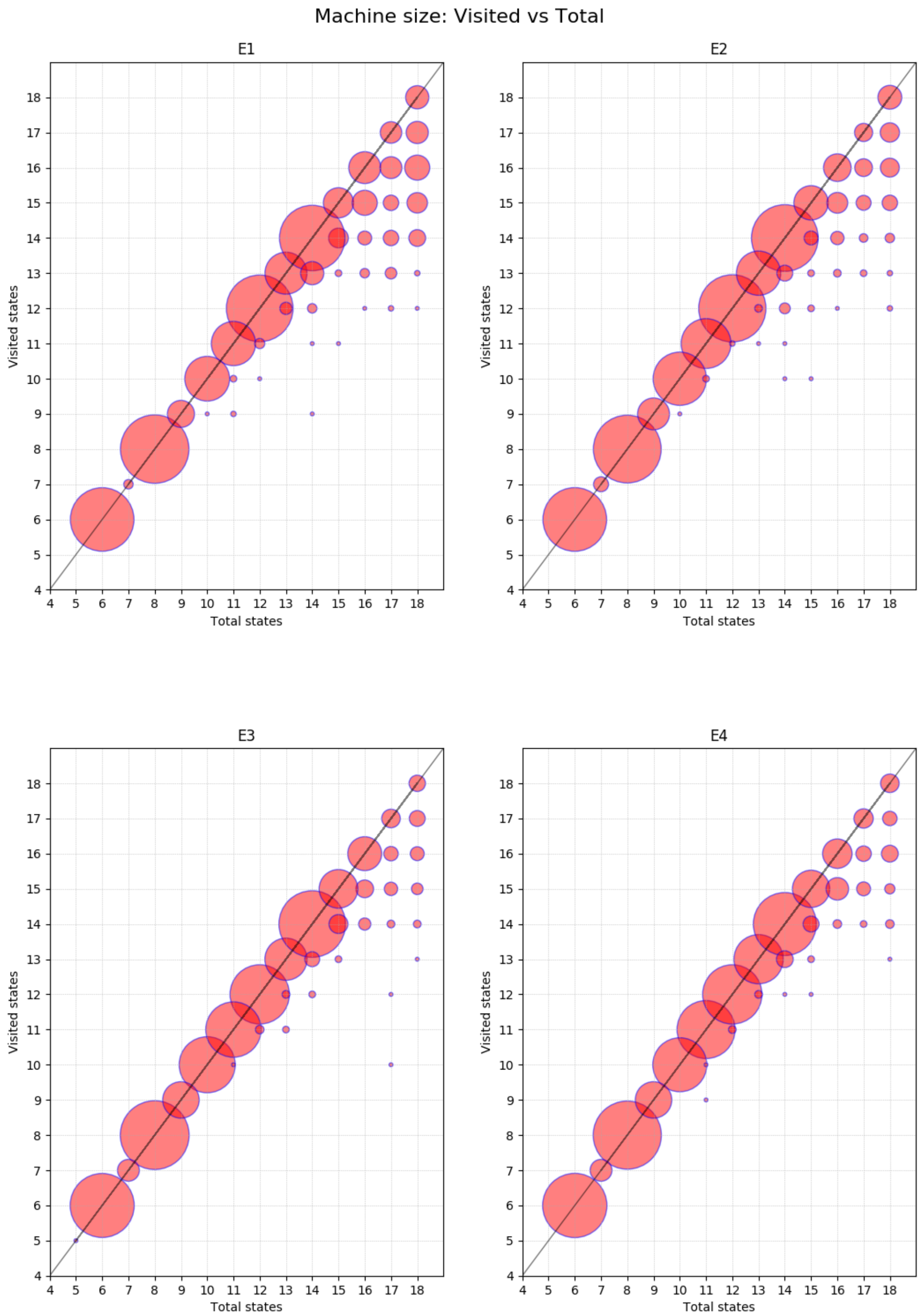

Figure 7.26: The final SEMs machine size (total states) against the number of visited states for four experiments, E1, E2, E3, and E4.

7.5 Ranges of Machine Size

The modification of the machine size within the range of 4 to 18 states was allowed in [31] and observed an inconsistency in the distribution of final machine sizes. This study further examined this range by splitting it into smaller chunks (sometimes overlapping) to see if any of them allowed faster convergence towards the best machines. The sizes of the machines generated over 100 runs in the previous study were somewhat uniformly dispersed from 9 to 18 states with the median varying from code to code between 11 and 13.5 states. Therefore further investigation was proposed to determine if a smaller range would minimize the deviation and to see if doing so would encourage the machines to move towards a particular size. Overall the idea was to understand how different codes reacted to different ranges and to look for potential trends or anomalies.

Figure 7.5 can be analyzed to realize the effectiveness of the different ranges on different codes. When compared to their larger counterparts, the two smallest ranges (4 to 6 and 4 to 8) proved to be much less effective across the board. The general trend that was noticed was ranges with higher maximums tended to produce SEMs with better decoding ability. However, the improvements were not too significant and there were a few inconsistencies noticed between adjacent or overlapping ranges e.g. range 10 to 16 producing slightly better accuracy than 6 to 18 for code60-2 and code205-2 in subplot E3. For direct training, range 14 to 18 can be identified as the overall best as it produced good accuracy across codes. Also notice in Figures 7.10 – 7.15 that for larger codes range 14 to 18 had the smallest deviation in terms of accuracy, making it the most consistent range during training and verification with direct classification. However, range 6 to 18 performed well across classification methods (direct, fuzzy) and datasets (training, verification). This possibly suggests that a wider range with a high upper bound may be better suited to be used across different code lengths and classification methods. However, this is not conclusive for specific code so requires further study.

7.6 Error Correction Capability on Different Distances

The error correcting ability depend on the edit distance of the codeword and the error pattern. Table 7.7 – 7.8 show the maximum accuracy on three different distances with direct and fuzzy classification respectively. The results for each code with different experiments are provided in Appendix B.1. As was expected, the accuracy of the machines decreases as the distance increases. This is also shown in previous studies [31, 14, 34, 11]. Three different codes of three different lengths are provided here as examples to showcase the differences among distance 1, 2, and 3. These codes are code17-1 of code length 10, code55 of code length 12, and code201 for code length 14.

Figures 7.27 – 7.29 demonstrate how the accuracy of the decoders are affected by the distance between the error pattern and the codeword. The violin plots are color coded based on the classification method (direct vs fuzzy) and the dataset (training vs verification) used. In general, the accuracy obtained with training dataset is slightly higher than verification dataset for both direct and fuzzy classifications. This could possibly be due to the fact that the training dataset is used to find the best possible solution and the verification dataset is used to verify its accuracy.

	Max Accuracy % (Direct-training)			Max Accuracy % (Direct-verification)		
Code	Distance 1	Distance 2	Distance 3	Distance 1	Distance 2	Distance 3
code17-1	98.2	91.2	67.6	94.7	83.5	56.5
code17-2	97.6	91.8	72.4	95.3	82.9	55.9
code18	96.1	87.2	67.8	95.0	78.3	52.8
code55	92.4	81.4	57.7	91.7	78.8	55.2
code60-1	91.5	79.6	58.9	90.0	75.8	52.6
code60-2	91.0	78.9	55.7	90.3	75.1	52.2
code201	86.5	68.6	42.6	85.5	67.5	41.0
code205-1	86.9	69.7	42.3	85.6	66.8	40.2
code205-2	87.7	70.5	43.0	86.3	68.7	40.4

Table 7.7: Maximum accuracy with direct classification for each distance

	Max Accuracy % (Fuzzy-training)			Max Accuracy % (Fuzzy-verification)		
Code	Distance 1	Distance 2	Distance 3	Distance 1	Distance 2	Distance 3
code17-1	98.8	96.5	85.9	98.2	94.1	78.8
code17-2	99.4	97.1	82.4	98.2	94.1	75.3
code18	97.2	91.1	78.9	97.8	90.6	70.6
code55	95.5	90.9	80.8	95.5	91.1	79.5
code60-1	94.6	90.3	80.0	94.6	88.8	78.6
code60-2	95.6	90.4	78.2	95.8	89.2	77.9
code201	91.9	83.4	69.2	91.4	84.1	67.8
code205-1	91.9	83.6	66.9	91.8	83.2	67.1
code205-2	92.4	85.7	69.0	92.7	84.5	69.5

Table 7.8: Maximum accuracy with fuzzy classification for each distance

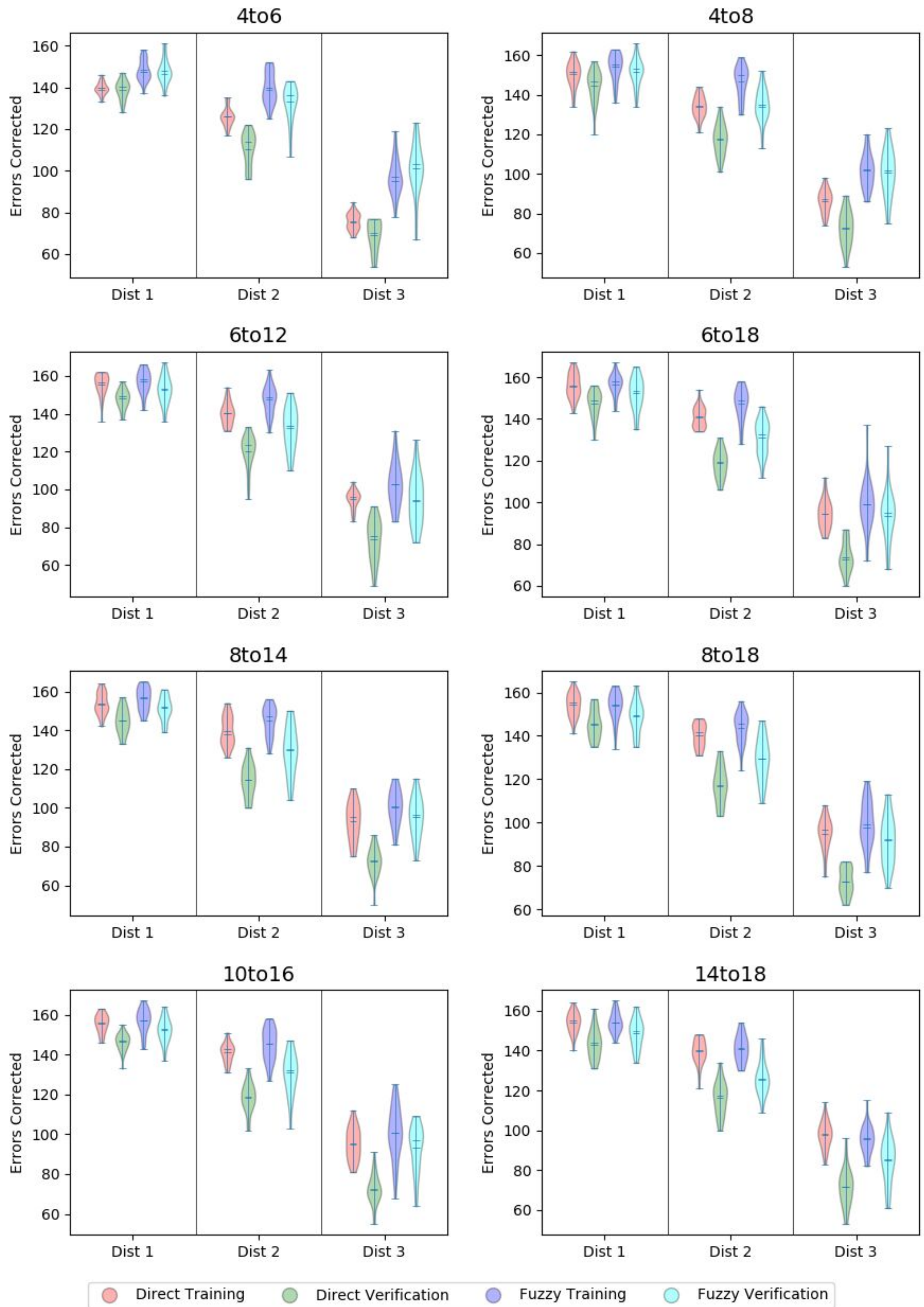

Figure 7.27: Code17-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1. The maximum possible fitness score for each distance is 170

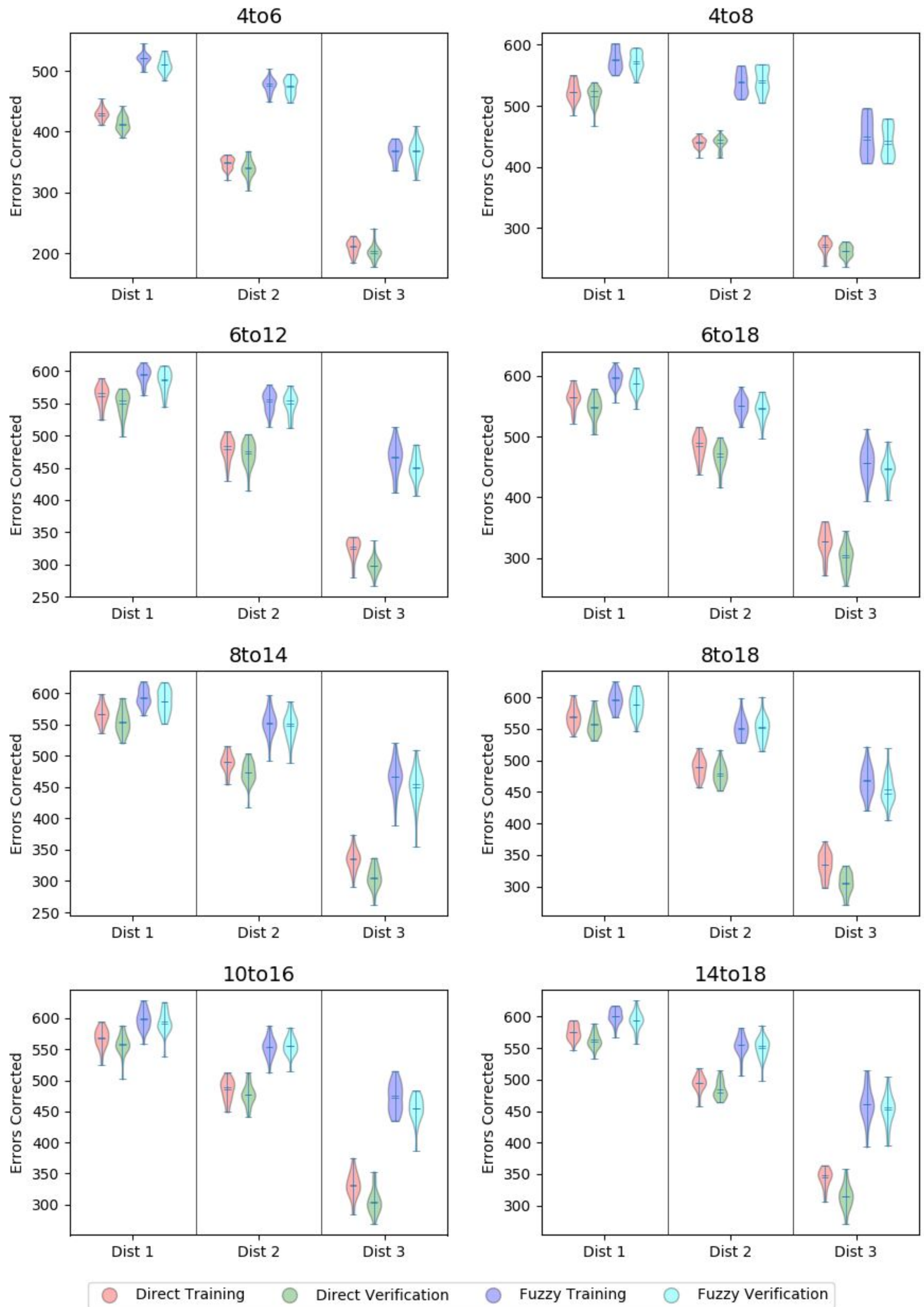

Figure 7.28: Code55, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1. The maximum possible fitness score for each distance is 660

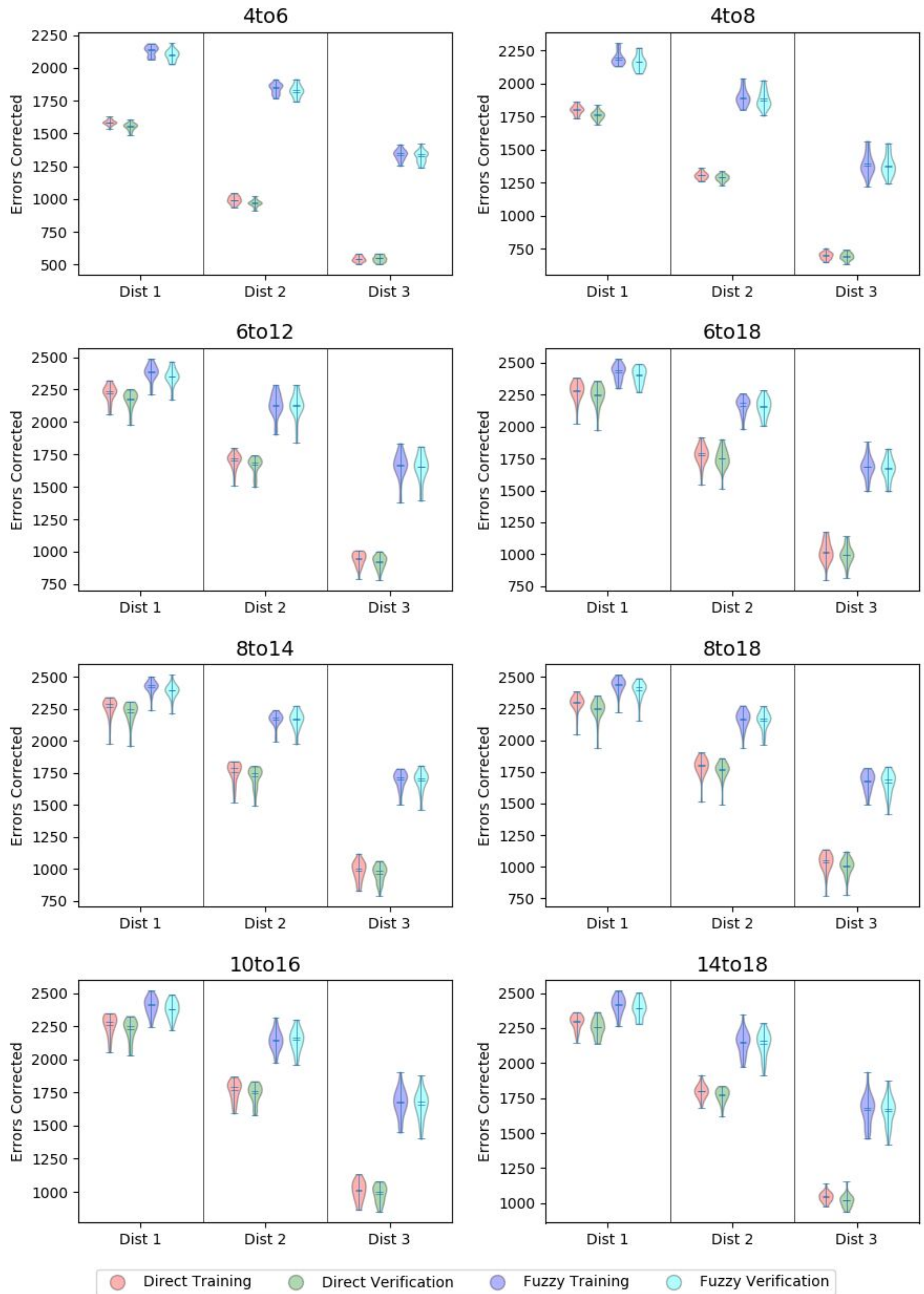

Figure 7.29: Code201, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1. The maximum possible fitness score for each distance is 2814

Chapter 8

Conclusion and Future Work

This study is a continuation of previous studies, in particular [31], that examined SEMs as edit metric decoders. Besides validating previous work, this thesis also extended the scope of the study by investigating the effectiveness of SEMs for decoding codewords of different lengths. In addition to length 12 that was studied by [11, 14, 34, 31], lengths 10 and 14 were used in this study. Also, more ranges were considered as compared to all related previous studies.

Previous studies [14, 31] observed a preference for a higher number of states, a trend that is also witnessed in this study. In fact, the propensity for larger machines is found to be stronger for larger codes i.e. more often than not, the smaller codes find smaller machines and the larger codes find larger machines as best machines. However, the fitness saturates once a certain number of states is reached and this saturation point also appears to depend on the length of the codeword. For example, fitness for codes of length 14 (code201, code205-1, code205-2) do not ever saturate in this study, which begs the question of how far they would improve if allowed to grow more. Future work can include finding a new upper bound for the number of states to be used with larger codes.

Compared to the direct classification method, the fuzzy method greatly improves the accuracy of the decoders, especially for error patterns with larger edit distance. It also produces better decoding accuracy than the direct approach for errors with higher distances as well as for the verification dataset. These improvements are not surprising since the use of edit distance in fuzzy classification greatly enhances the precision of the decoders in predicting the correct codeword. These findings are also consistent with observations made in previous studies. Possible future work can include a study of appropriate tolerance for fuzzy to examine how far away the actual codewords are, on average.

The study also observes a weakness in the algorithms used for evolution where, as the number of generations progresses, the population loses its diversity and gets overpopulated with more of the same SEMs, hindering the evolution towards a consistent well-balanced machine. This trend is stronger in codes of smaller length where duplicate machines start appearing sooner than they do in codes of higher length. This is expected due to the smaller overall search space. There are various ways to increase diversity in the population, which would help make sure that the search for the best machine is performed on the entire solution space rather than getting stuck within a local space. One approach that could be tried in future work is the recentering-restarting algorithm which appeared to have achieved good results in [34].

Incidentally, for larger codewords, the solution space also grows enormously. This larger search space hinders the ability of mutation to find a compact SEM. For these larger codes, as long as the improvement continues, it could be interesting to continue the process for more generations to see if accuracy improves. Future work can also include experimenting with other EP settings, such as bout size and number of generations, as well as examination of codes with different minimum distances.

The study also considers a variety of rates for the different types of mutation and observes how they influence the number of states in the final machines. The results in Figure 7.9 confirm that a higher rate of "add state" operation allowed more machines to grow to max size and vice versa. Further examination of the mutation types, especially other combinations with respect to the rates may prove valuable. Another interesting question that can be asked is whether allowing even faster growth rates will encourage the machines to achieve better accuracy, in particular for larger codes.

This study closely examines the connectivity of the best machines by finding the number of total and visited states. This knowledge can help simplify the machines by excluding the unvisited states. It also conclusively demonstrates the manifestation of the bloat that was observed in [31] and was suggested to be examined further. Therefore, it should be recognized that this study is the first to conduct this investigation and accurately report the actual machine size used for decoding.

A potential flaw in the "add state" mutation operation is also identified where the newly added state is left unreachable by design, which is believed to adding to the bloat. As future work, the algorithm for the said operation can be tweaked to create incoming transitions as soon as a new state is added and the machines should be investigated to see if this helps reduce the bulk.

Bibliography

[1] Mark D Adams, Jenny M Kelley, Jeannine D Gocayne, Mark Dubnick, Mihael H Polymeropoulos, Hong Xiao, Carl R Merril, Andrew Wu, Bjorn Olde, Ruben F Moreno, et al. Complementary dna sequencing: expressed sequence tags and human genome project. *Science*, 252(5013):1651–1656, 1991.

[2] Dan Ashlock, Ling Guo, and Fang Qiu. Greedy closure evolutionary algorithms. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02*, volume 2, pages 1296–1301. IEEE, 2002.

[3] Daniel Ashlock and Sheridan Houghten. Hybridization and ring optimization for larger sets of embeddable biomarkers. In *2017 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8. IEEE, 2017.

[4] Daniel Ashlock and Sheridan K Houghten. Dna error correcting codes: No crossover. In *2009 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 38–45. IEEE, 2009.

[5] Daniel Ashlock, Sheridan K Houghten, Joseph Alexander Brown, and John Orth. On the synthesis of dna error correcting codes. *Biosystems*, 110(1):1–8, 2012.

[6] Daniel Ashlock and Elizabeth Warner. Side effect machines for sequence classification. In *2008 Canadian Conference on Electrical and Computer Engineering*, pages 1453–1456. IEEE, 2008.

[7] Thomas Bäck, Günter Rudolph, and Hans-Paul Schwefel. Evolutionary programming and evolution strategies: Similarities and differences. In *Proceedings of the Second Annual Conference on Evolutionary Programming*, pages 11–22, 1993.

[8] Madeleine Price Ball. Dna replication split, 2013. `https://commons.wikimedia.org/wiki/File:DNA_replication_split.svg`, Accessed on 2020-02-07.

[9] Kenza Guenda Bennenni, Nabil and T. Aaron Gulliver. Greedy construction of dna codes and new bounds. *arXiv preprint arXiv:1505.06262*, 2015.

[10] Urmila Bhanja, Sudipta Mahapatra, and Rajarshi Roy. An evolutionary programming algorithm for survivable routing and wavelength assignment in transparent optical networks. *Information sciences*, 222:634–647, 2013.

[11] Joseph A Brown, Sheridan K Houghten, and Daniel A Ashlock. Edit metric decoding: a new hope. In *Proceedings of the 2nd Canadian Conference on Computer Science and Software Engineering*, pages 233–242. ACM, 2009.

[12] Joseph Alexander Brown. Decoding algorithms using side-effect machines. Master's thesis, Brock University, 2010.

[13] Joseph Alexander Brown. On side effect machines as a representation for evolutionary algorithms. In *2015 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8. IEEE, 2015.

[14] Joseph Alexander Brown, Sheridan K Houghten, and Daniel Ashlock. Side effect machines for quaternary edit metric decoding. In *2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8. IEEE, 2010.

[15] Kumar Chellapilla and Gary B Fogel. Multiple sequence alignment using evolutionary programming. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99*, volume 1, pages 445–452. IEEE, 1999.

[16] John Conway and N. J. A. Sloane. Lexicographic codes: error-correcting codes from game theory. *IEEE Transactions on Information Theory*, 32(3):337–348, 1986.

[17] Francis HC Crick, John Stanley Griffith, and Leslie E Orgel. Codes without commas. *Proceedings of the National Academy of Sciences of the United States of America*, 43(5):416, 1957.

[18] Charles Darwin. *On the origin of species by means of natural selection, or, the preservation of favoured races in the struggle for life*. J. Murray, 1859.

[19] Matthew C Davey and David JC Mackay. Watermark codes: Reliable communication over insertion/deletion channels. In *2000 IEEE International Symposium on Information Theory*, page 477. IEEE, 2000.

[20] Jason B Ernst and Joseph Alexander Brown. An online evolutionary programming method for parameters of wireless networks. In *2011 International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 515–520. IEEE, 2011.

[21] Ting-Cheng Feng, Tzuu-Hseng S Li, and Ping-Huan Kuo. Variable coded hierarchical fuzzy classification model using dna coding and evolutionary programming. *Applied Mathematical Modelling*, 39(23-24):7401–7419, 2015.

[22] Robert Flack and Sheridan Houghten. Generation of good edit codes from classical hamming distance codes. *Congressus Numerantium*, 190:97–108, 2008.

[23] David B Fogel. *System identification through simulated evolution: A machine learning approach to modeling*. Ginn Press, 1991.

[24] David B Fogel. Applying evolutionary programming to selected traveling salesman problems. *Cybernetics and systems*, 24(1):27–36, 1993.

[25] David B Fogel. *Evolving artificial intelligence.* PhD thesis, University of California, San Diego, 1993.

[26] Gary B Fogel, Kumar Chellapilla, and David B Fogel. Reconstruction of dna sequence information from a simulated dna chip using evolutionary programming. In *International Conference on Evolutionary Programming*, pages 427–436. Springer, 1998.

[27] Lawrence J Fogel, Alvin J Owens, and Michael J Walsh. *Artificial intelligence through simulated evolution*. John Wiley & Sons, New York, 1966.

[28] Daniel K Gehlhaar, Gennady M Verkhivker, Paul A Rejto, Christopher J Sherman, David R Fogel, Lawrence J Fogel, and Stephan T Freer. Molecular recognition of the inhibitor ag-1343 by hiv-1 protease: conformationally flexible docking by evolutionary programming. *Chemistry & biology*, 2(5):317–324, 1995.

[29] Richard W Hamming. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160, 1950.

[30] Ahmad Hoorfar. Evolutionary programming in electromagnetic optimization: a review. *IEEE Transactions on Antennas and Propagation*, 55(3):523–537, 2007.

[31] Sheridan Houghten, Tyler K Collins, James Alexander Hughes, and Joseph Alexander Brown. Edit metric decoding: Return of the side effect machines. In *2018 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8. IEEE, 2018.

[32] Sheridan K Houghten, Dan Ashlock, and Jessie Lenarz. Construction of optimal edit metric codes. In *2006 IEEE Information Theory Workshop-ITW'06 Chengdu*, pages 259–263. IEEE, 2006.

[33] W Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge University Press, 2010.

[34] James Alexander Hughes, Joseph Alexander Brown, Sheridan K. Houghten, and Daniel A. Ashlock. Edit metric decoding: Representation strikes back. *2013 IEEE Congress on Evolutionary Computation*, pages 229–236, 2013.

[35] Chris A Kaiser, Monty Krieger, Harvey Lodish, and Arnold Berk. *Molecular cell biology.* WH Freeman, 2007.

[36] SK Nandha Kumar and P Renuga. Fvsi based reactive power planning using evolutionary programming. In *2010 INTERNATIONAL CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING TECHNOLOGIES*, pages 265–269. IEEE, 2010.

[37] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[38] Brad L Miller, David E Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212, 1995.

[39] Edward A Ratzer and David JC MacKay. Codes for channels with insertions, deletions and substitutions. In *2nd International Symposium on Turbo Codes and Related Topics*, 2000.

[40] F Sellers. Bit loss and gain correction code. *IRE Transactions on Information theory*, 8(1):35–38, 1962.

[41] Robert Edmund Simpson. *Introductory electronics for scientists and engineers*. Allyn & Bacon, 1974.

[42] Jing Sun. Bounds on edit metric codes with combinatorial dna constraints. Master's thesis, Brock University, 2010.

[43] Roberto Togneri and JS Christopher. *Fundamentals of information theory and coding design*. Chapman and Hall/CRC, 2003.

[44] Robert A Wagner and Michael J Fischer. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173, 1974.

[45] Xiaopeng Wei Jing Dong Wang, Bin and Qiang Zhang. Improved lower bounds of dna tags based on a modified genetic algorithm. *PloS one*, 10(2):e0110640, 2015.

[46] Wang Xu, Ke Min Chan, and Eric T Kool. Fluorescent nucleobases as tools for studying dna and rna. *Nature chemistry*, 9(11):1043, 2017.

[47] Xin Yao and Yong Liu. Fast evolutionary programming. *Evolutionary programming*, 3:451–460, 1996.

Appendix A

Edit Metric Error Correcting Codes

A.1 Code17-1

3 1 2 1 1 0 0 0 3 3	1 3 2 3 2 1 0 2 0 2	1 1 3 1 0 1 3 1 0 0	2 2 2 2 0 0 0 1 2 2
3 3 3 0 0 0 2 2 1 1	0 2 3 3 3 3 2 2 2 2	0 3 3 2 2 3 3 3 3 0	3 0 1 1 3 2 3 0 0 1
1 1 1 2 0 2 2 3 2 1	3 2 1 1 1 1 1 2 2 0	2 1 0 2 3 3 3 0 1 3	3 0 0 3 0 3 1 2 3 3
0 0 0 0 0 0 0 0 0 0	0 3 0 3 3 1 1 1 1 1	0 1 0 0 1 1 3 3 2 2	0 0 2 1 2 2 2 1 1 3
2 2 2 3 1 2 1 3 3 1			

Table A.1: $(10, 17, 7)_4$ Code - Code17-1

A.2 Code17-2

0 1 1 1 0 1 2 1 1 1 1 2 2 2 0 0 1 1 0 2 1 0 0 3 0 2 1 2 1 3 1 3 3 1 3 0 0 2 2 0 3 2 1 1 1 3 0 0 0 1	3 1 2 2 0 2 3 2 2 1 0 0 0 1 1 1 3 3 2 2 1 1 0 2 1 0 0 3 3 3 2 3 3 0 0 3 3 2 3 3	3 0 3 3 3 1 3 1 1 0 0 2 2 2 3 3 3 3 1 3 2 3 2 3 1 1 2 2 1 2 0 2 1 3 2 2 2 0 2 3	3 3 2 1 2 1 1 1 3 3 0 0 0 0 0 0 0 0 0 0 2 2 0 2 2 1 2 3 0 0 1 0 1 2 3 3 2 0 3 0

Table A.2: $(10, 17, 7)_4$ Code - Code17-2

A.3 Code18

2 2 3 1 3 3 1 3 0 1	2 2 2 2 0 0 2 2 2 0	0 0 0 0 0 0 0 0 0 0	3 3 3 3 1 2 1 1 0 0
0 3 0 1 1 1 3 3 2 0	1 0 3 3 0 1 0 2 1 2	0 1 3 3 2 0 3 2 3 0	3 1 0 1 1 2 2 2 1 1
0 0 0 2 3 3 3 1 1 3	2 1 1 0 0 0 3 1 1 1	1 2 2 0 1 0 1 0 3 3	1 1 1 2 1 3 0 2 2 3
2 0 2 1 1 1 2 0 0 2	3 3 1 3 0 0 2 3 3 3	2 3 0 0 3 2 0 1 3 2	1 1 2 3 2 2 0 0 0 1
3 2 2 2 2 1 1 1 1 1	3 3 2 3 3 3 3 2 2 2		

Table A.3: $(10, 18, 7)_4$ Code - Code18

A.4 Code55

0 1 3 3 3 3 3 1 3 1 1 1	1 1 1 3 2 2 2 1 1 2 0 2	1 1 1 2 0 0 3 1 3 3 0 3	3 0 1 2 1 2 1 2 3 0 2 0
2 0 3 3 2 2 2 0 0 0 3 0	0 0 0 0 0 1 3 1 1 1 3 3	3 3 3 2 2 2 2 1 0 1 2 1	3 3 2 1 1 1 1 0 3 1 3 0
1 2 2 2 0 1 3 2 3 1 1 1	3 3 1 3 3 0 0 0 2 1 0 0	0 0 3 2 2 1 3 2 0 3 3 1	1 0 3 0 3 1 2 3 3 3 3 0
1 2 1 0 0 2 1 0 3 2 2 2	1 1 0 0 3 3 0 0 1 2 3 3	2 0 0 1 1 1 0 2 3 3 0 1	2 2 1 1 0 0 0 3 0 1 0 2
2 0 1 0 2 2 0 2 2 3 2 3	2 3 2 3 3 2 2 3 3 1 3 1	0 0 1 0 0 3 0 0 0 0 1 1	3 2 3 1 0 2 1 0 1 0 0 0
3 2 3 1 2 1 3 2 2 2 2 2	0 1 1 3 1 0 3 3 2 1 3 2	1 3 0 0 0 0 0 2 0 0 2 2	3 1 2 2 2 2 3 3 3 3 3 2
0 0 2 1 2 3 1 1 2 1 2 2	1 3 3 3 1 0 0 0 1 3 3 2	3 1 1 0 3 0 1 1 1 3 1 0	0 1 1 1 3 1 3 0 0 3 0 0
3 2 2 3 3 3 1 1 2 3 3 3	0 0 0 2 2 0 0 0 1 2 2 0	2 2 0 0 1 2 1 2 0 1 3 2	2 0 0 0 0 3 3 3 3 2 0 2
1 1 2 0 3 1 1 2 2 2 0 1	0 2 1 3 3 2 0 1 3 3 2 0	2 1 1 1 2 3 2 1 2 3 3 1	3 3 3 0 1 1 0 0 2 2 1 1
3 3 3 0 3 3 3 2 3 0 0 0	3 1 1 2 3 3 0 3 1 0 2 2	1 3 1 2 1 1 0 2 1 1 2 3	0 3 3 1 1 1 1 2 2 0 0 3
3 3 0 0 2 0 3 1 2 0 3 1	1 2 3 1 1 1 1 3 0 3 2 1	1 2 2 3 2 2 0 2 2 2 1 0	2 0 2 2 3 1 0 0 1 3 1 3
0 3 0 1 3 2 3 2 0 2 1 3	0 1 2 0 0 3 2 2 1 1 0 0	1 0 0 2 1 3 3 0 2 0 0 3	3 2 2 2 0 0 0 0 0 3 3 1
1 0 0 3 3 3 2 2 2 2 2 2	2 0 3 3 0 1 0 3 1 1 2 1	0 2 2 2 3 0 3 2 0 0 0 2	2 1 0 2 1 1 1 1 1 1 1 1
2 1 2 3 0 2 3 0 2 0 1 1	2 2 3 3 3 3 3 3 0 2 2 3	2 2 2 2 1 1 3 1 2 2 3 0	

Table A.4: $(12, 55, 7)_4$ Code - Code55

A.5 Code60-1

1 3 3 1 1 1 0 1 1 2 1 0	3 3 1 2 2 0 3 0 1 1 0 2	1 2 3 2 2 2 2 2 0 0 0 2	3 0 0 1 3 2 2 1 2 0 3 0
1 1 0 2 1 1 2 0 1 3 2 2	3 3 1 0 1 3 1 0 0 2 2 2	1 1 0 0 2 3 2 0 3 1 1 1	1 0 0 1 3 1 1 3 2 2 1 3
2 0 0 3 2 2 2 2 2 2 1 1	1 3 3 3 2 1 0 0 2 3 3 2	0 1 1 1 0 0 0 1 0 1 2 1	2 2 0 0 1 1 3 1 0 2 0 3
3 2 2 1 0 3 3 1 1 1 1 0	2 2 2 3 1 3 0 0 1 1 0 1	3 1 3 0 1 2 3 2 1 0 1 2	3 2 1 0 0 2 0 2 2 3 0 2
2 2 3 0 2 3 1 3 3 1 2 2	0 0 0 2 0 2 1 1 1 3 2 1	3 0 1 2 3 3 3 2 2 1 1 1	2 1 1 3 3 0 1 1 2 2 2 2
2 3 2 0 0 0 1 2 1 1 1 2	0 0 2 0 0 3 3 0 0 1 3 3	2 2 1 1 0 1 2 1 0 3 3 0	3 3 2 2 1 2 3 1 2 3 0 3
3 1 3 3 3 0 0 3 3 0 1 3	0 3 0 3 0 2 2 0 0 3 1 3	0 2 0 3 1 1 1 2 1 2 3 2	1 3 2 1 1 3 3 0 3 0 0 0
2 3 1 2 2 2 0 3 3 3 2 3	1 2 2 2 2 3 2 1 1 2 3 1	3 1 1 2 3 1 1 3 1 3 3 1	0 2 2 2 2 0 0 0 3 3 1 2
3 2 1 2 1 2 1 1 0 0 1 1	2 3 3 1 3 3 0 2 0 1 2 3	3 3 0 0 0 0 1 1 3 3 3 0	0 0 2 2 0 0 0 1 2 2 0 0
1 1 1 1 3 1 2 2 3 0 3 2	2 2 1 1 1 2 3 3 2 2 1 2	1 1 3 3 2 3 3 3 2 2 0 3	0 3 3 0 1 0 0 3 2 0 2 0
3 0 0 0 0 1 0 0 0 2 1 1	1 0 1 1 2 0 3 2 3 3 0 1	2 1 0 1 1 2 2 1 1 1 1 3	2 0 1 0 3 1 1 0 0 0 0 0
0 0 3 3 1 0 2 1 3 0 0 1	0 0 0 3 2 3 0 2 0 0 2 2	3 3 3 1 0 2 2 2 1 2 2 1	2 3 2 3 3 3 2 1 3 1 3 0
1 2 0 0 0 0 3 0 3 2 2 2	3 2 0 1 1 1 0 3 3 1 3 3	3 3 0 3 2 1 3 3 0 3 2 1	0 1 1 2 2 1 3 0 2 2 2 1
2 0 2 2 1 2 1 0 2 1 2 0	2 1 1 0 0 0 0 0 2 0 3 3	0 2 1 0 2 2 3 3 1 0 0 0	1 2 1 3 3 0 0 2 0 2 1 1
0 0 3 3 3 1 3 1 1 1 1 2	1 0 0 3 0 0 2 2 0 1 0 0	1 1 2 3 2 0 0 1 0 0 3 0	0 0 0 2 1 3 3 3 2 3 3 2

Table A.5: $(12, 60, 7)_4$ Code - Code60-1

A.6 Code60-2

2 2 2 0 1 2 2 1 1 0 2 2	0 1 0 1 1 3 3 0 0 0 0 0	3 0 1 3 0 1 3 0 1 3 3 1	3 2 0 2 0 1 1 2 2 2 2 1
0 3 2 0 0 0 0 2 3 2 0 2	2 1 2 3 2 0 0 2 2 3 3 2	1 3 3 0 3 3 1 1 3 1 1 3	0 3 3 0 3 2 0 0 0 3 3 3
1 1 3 3 2 2 1 0 0 0 2 0	2 1 1 1 1 1 2 0 1 1 1 2	2 2 0 3 3 2 2 3 0 3 2 1	0 3 3 3 1 2 1 2 1 3 0 1
3 3 3 1 1 0 3 1 3 3 3 2	3 3 0 0 0 2 2 2 3 3 3 0	0 1 1 0 0 2 2 2 0 1 0 1	2 2 2 2 2 3 2 3 2 2 0 3
3 3 3 3 3 0 1 0 2 2 2 3	1 3 2 1 3 2 2 2 2 2 2 2	1 1 1 3 3 0 0 1 3 2 1 1	3 3 3 2 0 1 1 1 1 0 0 3
1 2 0 2 2 2 1 2 1 3 3 3	1 0 1 0 0 1 1 1 1 1 2 0	1 1 2 1 1 3 1 1 1 0 3 1	3 0 0 2 2 3 2 0 0 3 2 2
0 0 0 0 1 1 0 0 0 1 1 1	2 3 2 1 2 2 1 0 3 3 0 0	3 3 1 3 3 2 3 2 3 0 0 3	3 1 0 3 1 0 3 1 0 1 2 1
2 0 1 3 1 0 3 2 1 2 0 2	1 3 2 2 3 2 3 3 3 1 3 0	1 1 0 0 0 0 0 3 0 3 1 0	1 0 1 2 3 1 2 2 0 1 3 2
2 3 0 2 3 3 1 3 0 0 1 2	1 2 2 3 3 3 1 0 0 3 3 2	0 0 0 3 2 0 2 1 2 2 1 3	1 0 2 2 1 0 0 2 2 1 2 3
3 1 1 0 2 0 3 2 3 1 1 2	1 1 1 1 1 1 1 3 2 2 2 3	1 0 0 3 1 2 0 3 0 0 0 3	3 3 1 2 2 2 1 0 1 1 1 1
3 2 1 1 1 0 0 0 0 0 2 2	3 1 2 2 1 2 1 1 2 3 0 2	2 2 1 3 0 0 1 2 0 3 1 3	2 3 3 3 0 0 2 3 1 0 1 0
0 1 1 2 1 0 2 3 0 3 3 3	0 0 3 3 3 3 3 3 3 0 3 1	0 0 3 0 2 0 1 3 1 2 3 2	1 0 1 1 2 2 3 3 1 2 1 0
1 1 3 0 3 3 3 0 2 2 0 2	2 0 0 1 1 1 3 3 3 1 3 3	3 0 3 2 2 2 0 2 2 0 0 1	1 2 2 2 0 1 3 0 3 0 0 1
2 0 2 0 0 3 3 2 1 1 1 1	2 3 2 2 0 0 0 0 0 1 3 1	3 0 0 0 3 3 1 1 2 0 0 0	1 2 3 1 1 1 2 2 2 1 0 0
2 2 0 0 2 1 0 2 0 2 3 0	0 2 1 3 2 3 0 1 1 0 2 0	0 2 2 1 1 3 3 3 3 2 1 2	0 1 2 3 3 1 2 3 3 2 2 0

Table A.6: $(12, 60, 7)_4$ Code - Code60-2

A.7 Code201

0 3 0 1 0 2 2 1 1 1 0 2 1 0	1 0 3 0 0 0 3 1 2 1 1 1 3 1	2 2 2 0 3 2 3 3 1 1 0 0 1 1	2 2 2 2 2 2 0 1 0 1 1 2 2 0
0 2 0 0 3 0 2 3 1 1 0 1 2 2	1 1 3 3 3 3 2 1 1 3 0 0 3 2	2 2 3 2 3 1 3 2 3 0 0 2 2 3	0 0 1 1 2 1 3 1 1 1 1 1 1 3
0 1 3 2 1 1 1 2 3 0 3 2 3 1	2 3 1 2 0 2 2 1 0 0 0 3 1 1	3 3 3 3 2 0 2 3 3 3 3 3 1 1	2 1 1 0 3 0 1 1 2 2 3 2 1 2
2 0 3 3 1 3 2 1 2 2 2 2 2 0	3 0 3 0 2 2 3 3 0 3 2 2 3 1	2 2 2 0 0 2 3 1 3 2 2 2 3 3	3 1 1 2 0 3 2 3 1 0 2 1 1 2
0 2 0 1 2 3 3 3 0 0 0 3 2 2	1 1 1 0 1 0 1 1 0 0 2 1 1 1	2 2 0 3 2 2 1 3 3 3 3 2 3 0	1 0 1 0 2 1 2 2 1 3 0 2 2 1
1 1 3 1 0 3 3 1 0 0 0 3 3 3	0 1 3 3 0 1 0 0 2 3 2 3 1 3	3 1 1 0 0 2 1 0 3 0 3 3 0 0	2 2 1 1 1 1 3 3 2 3 1 3 0 0
3 3 3 2 2 2 0 3 3 2 1 0 2 3	2 2 1 3 2 1 0 0 3 2 3 2 2 1	2 0 0 1 2 1 2 1 1 0 2 2 2 2	0 2 2 2 1 2 1 3 3 3 2 2 0 3
0 3 2 3 1 1 1 0 0 1 1 3 1 3	3 1 0 0 3 2 1 1 0 0 0 0 2 2	0 3 1 1 2 0 0 0 3 2 3 0 1 0	2 0 0 1 0 0 2 2 1 1 2 3 3 0
0 3 0 3 1 3 2 1 1 2 3 1 1 1	2 2 2 3 3 3 0 2 1 3 3 2 0 2	3 0 0 0 1 2 3 3 0 1 3 0 0 0	1 0 3 2 3 1 0 3 3 1 1 3 2 2
2 1 2 0 3 3 3 1 3 1 1 1 3 0	1 2 0 2 1 1 2 1 3 3 1 1 0 1	2 2 1 3 0 0 0 0 0 1 3 0 3 0	0 1 0 3 2 3 3 1 2 2 0 3 0 0
1 0 0 2 3 2 3 0 1 2 1 3 1 3	1 3 2 2 3 0 2 1 0 1 3 0 0 0	3 3 2 0 0 2 3 2 2 2 2 0 0 2	3 3 3 0 3 1 2 1 1 1 2 2 3 0
1 1 0 2 1 0 0 2 3 2 2 0 3 3	1 2 3 1 1 1 1 1 3 2 0 3 1 1	1 1 1 2 2 3 3 3 3 2 0 1 1 0	3 3 2 3 2 1 1 2 3 3 2 0 1 0
1 3 1 3 3 3 0 0 0 0 2 0 1 2	2 3 0 0 0 3 1 0 2 0 2 1 2 3	0 2 0 3 0 2 0 2 2 3 2 0 2 0	0 1 0 0 0 1 1 1 3 3 2 2 1 1
3 3 1 1 2 2 0 1 1 2 0 3 2 0	1 0 3 0 3 2 2 2 3 2 0 0 3 1	2 2 0 2 0 0 0 2 2 1 3 3 1 1	2 0 1 0 0 3 1 0 0 1 0 1 0 1
0 2 2 2 1 2 0 0 2 0 0 3 1 3	0 0 1 2 1 1 2 2 3 3 1 3 2 0	2 2 0 1 3 3 3 0 3 3 2 3 3 3	3 1 0 1 0 0 3 0 1 2 2 0 1 1
2 2 1 1 3 2 1 2 0 1 1 1 2 3	0 0 3 0 1 1 1 3 2 0 0 0 0 1	2 3 3 3 3 1 0 2 3 2 2 1 0 0	2 2 1 2 0 0 1 1 1 1 2 1 3 1
2 0 3 1 0 1 2 3 0 2 3 3 2 1	3 0 3 2 2 1 2 2 2 1 0 1 2 0	1 3 1 0 0 0 0 1 1 1 1 2 0 0	3 0 0 2 1 2 1 2 3 0 3 1 1 3
0 3 2 0 2 3 3 3 2 1 3 0 0 1	0 3 1 3 3 3 3 3 3 1 1 1 0 3	1 2 0 0 2 2 3 2 2 3 3 3 0 0	3 0 1 2 1 0 2 1 0 0 1 2 3 3
3 0 0 1 2 2 2 1 0 2 0 1 0 1	1 1 2 2 1 1 0 0 0 2 3 3 0 1	0 3 0 0 1 1 3 2 1 3 2 2 0 2	2 3 3 1 2 2 1 0 2 1 3 3 3 2
2 3 0 3 3 3 3 3 2 2 1 3 1 1	2 1 0 3 2 2 0 1 1 0 0 3 3 2	0 3 2 2 0 3 2 2 2 1 3 2 3 3	3 0 1 1 1 0 0 2 1 2 1 3 0 1
1 3 3 1 1 3 1 2 1 0 0 2 1 3	1 3 1 0 2 3 0 3 1 3 2 3 3 0	1 3 1 3 2 2 1 2 1 0 3 0 3 3	1 0 2 2 3 1 1 3 2 2 2 0 0 2
2 0 3 2 1 3 0 2 0 0 2 1 0 1	3 0 2 0 3 1 1 1 3 3 2 1 2 2	3 2 3 0 1 0 0 2 2 3 1 0 1 1	1 2 1 2 0 1 1 2 0 2 0 0 0 3
3 2 1 3 0 3 2 2 0 3 0 3 1 3	0 3 2 1 3 2 3 1 0 0 0 0 3 0	0 1 0 2 2 2 2 0 3 3 2 0 0 0	1 2 2 3 1 2 0 0 3 2 2 1 1 0
3 2 0 2 0 3 1 1 0 2 1 0 3 0	0 0 0 2 3 1 0 3 0 3 3 3 3 1	0 2 3 2 3 3 2 2 3 2 3 3 1 2	3 0 2 1 1 3 0 1 3 0 0 1 3 1
3 3 2 2 0 0 0 0 0 2 2 3 2 0	2 3 0 2 3 2 0 3 2 0 3 2 2 2	3 2 3 0 3 1 3 3 1 0 1 1 3 2	0 3 2 1 0 1 3 0 0 3 0 2 2 3
1 1 1 1 3 1 2 3 3 3 1 3 1 1	1 0 2 0 2 1 0 0 0 3 1 3 2 3	0 0 3 2 0 2 3 2 2 0 2 2 1 3	0 2 3 2 2 2 2 2 3 1 1 1 0 3
0 2 1 3 2 2 2 2 1 2 2 3 2 1	2 2 1 1 0 3 1 3 1 0 3 2 2 0	0 1 0 0 2 0 0 3 0 2 1 1 2 0	2 1 1 2 3 0 0 1 1 0 3 1 0 3
0 2 3 3 3 2 0 1 0 1 0 1 0 0	0 0 2 0 1 0 2 1 2 1 2 0 0 3	2 2 3 1 3 1 1 0 1 1 2 0 0 0	0 1 2 3 3 3 2 3 0 0 0 2 1 1
1 1 1 2 2 0 2 0 0 0 1 0 2 1	0 0 0 0 3 2 2 0 0 3 3 0 1 2	3 3 0 1 0 0 0 3 1 3 1 2 2 2	1 1 3 0 0 3 0 3 3 2 3 2 2 3
0 0 1 1 0 2 0 3 2 2 2 3 3 2	3 2 1 1 3 3 3 3 3 2 0 0 0 1	2 1 0 3 1 0 3 0 1 2 3 1 3 3	1 3 0 3 0 0 1 0 2 2 2 2 0 1
0 0 3 1 0 0 1 3 3 1 2 1 1 0	2 2 3 0 2 3 2 0 0 0 1 1 3 3	3 2 2 1 2 1 3 1 1 3 3 3 3 2	2 3 0 1 3 0 3 3 3 1 0 0 2 3
0 2 1 2 3 3 1 0 3 2 1 0 2 3	2 0 3 3 2 1 2 3 1 3 1 1 1 2	2 1 3 1 0 2 3 3 3 0 2 1 2 2	0 0 2 2 2 0 1 0 3 2 3 3 2 2
0 1 1 0 3 1 1 2 0 1 2 0 3 0	3 1 0 3 0 0 0 3 3 0 0 3 2 0	2 3 3 3 1 1 0 1 0 3 3 0 1 1	2 1 2 2 2 0 3 3 3 1 2 2 2 1
2 1 0 1 1 1 1 1 0 0 1 1 3 0	1 1 0 2 0 1 1 1 1 1 3 3 3 2	1 1 2 2 2 0 1 3 2 3 0 1 3 1	0 0 1 1 3 1 2 1 1 3 3 0 0 0
2 3 0 2 2 0 1 0 3 3 1 2 1 3	2 2 2 2 0 1 1 0 2 3 3 3 3 3	1 1 1 1 2 0 3 0 2 0 2 1 3 3	3 3 1 2 2 0 0 3 1 1 0 0 0 1
3 1 3 3 2 3 0 1 1 2 3 1 2 2	3 3 2 3 3 1 0 0 1 3 0 0 2 2	0 1 3 1 1 0 3 1 3 0 1 0 1 3	2 0 2 0 2 0 0 1 1 3 0 0 0 2
3 3 2 0 1 2 0 3 2 1 1 3 3 1	0 0 0 3 3 1 0 0 1 0 2 3 2 0	1 0 0 0 1 1 0 1 2 2 1 2 2 2	3 0 2 0 3 0 0 1 1 0 2 3 1 1
0 2 1 2 1 0 0 2 0 1 2 0 0 2	0 1 2 3 3 0 1 3 1 3 2 2 2 0	3 1 1 1 2 2 3 2 0 2 2 2 2 2	0 0 1 3 1 3 3 3 0 2 3 3 1 0
1 2 1 3 1 2 0 1 1 1 1 0 1 2	3 2 1 1 1 1 2 1 2 2 0 1 3 2	3 1 2 2 1 3 3 3 3 1 3 0 1 2	3 3 3 0 3 0 3 2 3 2 1 2 0 1
0 3 3 1 3 1 2 0 2 0 0 1 2 2	1 1 0 3 2 2 0 2 3 1 1 2 1 0	3 3 1 0 1 3 2 2 3 1 2 3 0 3	2 2 2 3 0 1 1 2 2 1 1 0 2 2
1 2 3 2 0 1 0 1 0 1 2 2 3 3	1 2 0 2 0 0 2 2 0 0 2 3 2 2	1 1 0 3 0 2 3 3 1 3 3 0 1 3	1 1 2 0 0 1 3 3 1 0 0 2 0 0
0 0 2 3 1 2 3 0 2 1 1 2 2 1	1 1 1 0 1 3 3 2 2 3 0 1 0 2	3 0 3 3 3 0 2 2 0 1 0 0 0 3	1 0 0 1 0 2 0 2 0 3 3 1 0 3
0 3 3 3 0 3 1 1 2 2 0 0 3 3	3 1 1 1 0 0 0 0 3 3 1 1 3 1	3 3 3 3 3 3 3 2 0 3 3 3 0 0	3 2 2 1 0 3 3 1 1 1 1 0 0 3
2 2 3 0 1 3 2 1 2 3 0 3 0 1	2 0 1 0 0 0 0 2 1 2 2 2 2 3	1 1 2 3 3 3 0 2 2 2 2 1 3 1	1 1 2 1 0 2 2 2 1 1 1 3 2 3
2 2 0 0 0 1 3 3 3 0 3 1 1 1	0 2 1 0 0 3 2 0 1 3 1 2 3 1	0 0 2 2 2 0 1 1 0 0 1 1 0 0	2 0 0 2 3 3 2 1 1 1 0 2 0 1
1 2 1 3 3 2 1 2 2 0 1 2 0 1	3 3 1 3 1 1 3 3 2 2 2 2 3 1	1 1 2 3 2 3 2 3 0 3 3 3 0 2	3 3 3 0 0 1 0 2 0 0 1 1 1 0
3 1 0 3 1 1 2 2 2 3 3 3 3 2	0 1 1 3 0 2 2 1 3 3 1 1 1 2	0 0 3 1 3 1 1 2 1 3 2 3 3 3	3 3 2 1 3 3 1 2 2 1 1 3 1 0
1 1 0 0 3 3 1 1 2 3 1 0 0 1	3 3 0 1 3 1 0 1 3 3 1 3 3 0	2 2 1 2 2 2 3 3 2 0 0 0 2 0	2 3 1 2 1 2 1 2 2 2 2 1 1 1
1 3 3 0 0 3 2 3 0 3 0 1 1 2	1 0 2 3 3 0 0 1 3 3 3 3 3 2	3 1 2 0 0 3 3 0 0 1 1 2 2 2	3 3 3 3 2 0 1 1 1 1 2 1 1 3
0 0 2 2 0 2 2 3 0 1 0 3 0 2	3 1 2 1 1 1 2 2 3 0 0 0 0 2	1 1 1 3 2 1 1 1 3 1 2 3 0 0	0 1 1 1 1 0 2 0 0 3 0 3 2 2
3 1 2 2 3 1 3 0 1 0 3 3 2 3			

Table A.7: $(14, 201, 7)_4$ Code - Code201

A.8 Code205-1

1 2 2 3 1 0 3 0 3 1 2 0 1 1	2 2 2 0 0 0 0 0 1 1 3 3 0 2	1 3 0 1 2 0 0 3 2 1 1 3 0 1	1 2 2 2 2 2 1 1 2 2 0 1 1 3
3 0 1 3 1 2 2 2 2 2 2 1 0 0	0 1 3 3 2 3 1 0 1 3 3 0 2 0	3 0 0 3 3 1 2 1 1 1 0 2 2 1	3 1 0 3 2 3 1 3 1 3 1 3 1 3
1 1 0 1 1 1 1 2 3 2 0 3 1 3	0 0 2 3 3 3 0 2 1 3 2 0 1 3	2 2 2 0 2 2 2 1 2 0 0 3 2 2	1 1 2 1 0 1 3 2 3 2 1 0 2 0
2 2 3 3 3 3 0 0 0 2 2 3 2 0	3 1 3 0 0 1 1 1 1 0 3 1 3 0	2 3 1 1 2 2 2 2 3 2 0 2 2 3	0 0 3 1 2 0 0 2 3 0 3 2 0 3
3 1 2 1 2 2 3 3 3 3 3 0 1 3	3 2 2 0 3 3 3 1 0 0 3 2 1 1	1 0 0 1 0 2 3 2 3 0 2 1 3 3	0 0 3 1 2 2 2 2 1 0 0 3 1 2
1 1 1 0 0 2 2 2 1 0 2 2 3 2	0 0 3 3 3 2 3 0 3 1 1 2 3 0	3 3 2 3 2 1 1 1 2 1 3 1 2 2	3 1 3 3 3 2 2 0 0 3 2 2 1 0
3 1 0 1 0 0 2 3 2 3 3 0 3 0	0 3 1 3 0 3 1 2 0 1 1 1 2 1	1 3 2 1 0 2 2 1 1 1 1 1 1 1	1 0 2 2 2 3 3 2 0 0 1 2 0 2
3 2 0 3 2 1 0 0 2 1 1 1 0 0	1 0 0 0 0 0 3 3 0 0 1 1 2 2	2 3 2 0 0 2 0 0 2 2 0 3 3 3	2 0 2 3 3 0 1 0 2 3 1 1 0 3
1 1 1 3 3 2 0 0 2 2 1 1 2 1	2 2 3 0 1 3 0 3 1 1 1 1 3 0	2 0 0 0 3 3 1 3 3 1 3 1 1 2	3 2 2 2 1 0 2 3 3 3 0 0 2 2
2 0 3 2 2 2 3 2 1 1 1 1 3 2	1 3 1 0 3 0 3 0 3 3 2 2 3 3	0 2 0 1 3 0 3 0 3 3 0 1 1 3	1 1 1 1 2 0 3 1 0 1 0 2 3 3
2 2 2 2 1 0 1 1 3 0 2 1 0 0	0 2 2 3 0 0 1 1 1 0 0 2 3 2	3 0 1 0 0 0 2 1 0 3 2 1 1 2	1 1 3 1 1 1 1 3 0 2 1 1 0 1
0 3 3 2 3 3 2 1 3 3 3 1 3 3	1 0 2 2 3 0 2 2 3 1 3 3 1 2	1 1 3 2 0 1 2 3 3 2 3 3 1 0	0 2 1 2 2 1 2 2 1 1 1 3 3 3
0 3 2 1 1 3 2 2 3 1 3 3 2 1	0 2 3 3 1 1 3 3 0 3 3 2 0 0	0 3 2 1 1 0 0 3 1 0 2 0 0 3	3 1 3 3 2 2 1 2 3 1 0 0 0 1
1 2 1 3 1 3 2 2 2 3 3 1 3 2	1 3 2 3 1 1 2 1 0 1 0 3 0 3	2 2 3 0 0 2 3 2 1 0 2 0 2 1	2 3 2 3 2 3 3 2 2 0 1 3 1 1
2 1 0 0 1 2 0 3 0 2 0 1 1 1	2 0 0 1 0 1 1 2 0 3 3 0 2 2	0 3 2 2 0 2 0 0 3 3 2 1 1 1	2 3 3 3 2 0 2 2 0 2 1 2 0 0
0 1 0 2 2 1 1 2 1 1 2 2 1 2	1 2 1 2 2 0 1 1 1 0 3 3 1 2	0 1 2 3 1 0 1 0 0 1 3 2 2 2	2 2 0 0 0 1 2 3 3 3 1 1 0 1
1 1 3 0 3 1 3 1 2 2 2 1 2 2	0 1 1 1 3 0 1 2 2 2 0 1 0 1	2 2 2 0 1 2 2 0 0 2 1 1 2 3	3 3 0 3 0 3 1 2 0 2 0 0 3 0
3 3 1 0 1 1 3 2 0 3 3 1 1 3	2 2 0 2 0 1 0 0 2 3 2 3 2 1	0 0 1 1 0 2 0 2 3 1 1 2 2 2	2 1 3 3 1 0 1 0 3 0 0 2 1 3
1 2 1 1 0 3 2 1 3 0 2 3 1 0	1 0 1 2 3 3 3 1 1 2 2 3 0 3	0 2 2 1 0 3 0 0 0 0 0 3 2 2	2 1 3 0 3 2 0 1 2 3 3 3 0 3
0 1 1 2 0 3 3 1 3 0 3 3 2 3	1 2 2 0 2 1 1 3 3 2 0 0 3 0	0 0 3 3 0 1 2 3 1 2 2 3 3 3	2 1 3 2 1 1 3 0 3 3 0 3 0 1
2 0 0 0 2 0 1 3 0 3 2 2 0 2	3 3 0 3 0 0 0 3 0 0 2 2 0 1	3 2 1 2 2 2 1 3 0 0 1 0 0 3	3 3 2 2 0 0 1 3 3 1 2 3 2 0
3 0 2 0 3 2 2 2 2 3 1 0 2 3	1 0 3 3 0 2 0 1 1 0 3 1 1 3	2 3 3 1 1 0 1 2 1 2 1 0 1 3	3 2 2 2 2 2 3 0 1 3 2 2 3 3
0 2 2 0 2 2 0 2 0 1 0 1 3 0	0 3 1 0 0 1 1 3 3 2 2 1 0 0	0 0 0 2 1 0 0 1 0 0 1 3 3 3	2 3 0 1 1 3 1 1 1 3 2 2 2 0
3 1 1 1 2 2 0 0 0 1 2 0 2 1	0 0 2 1 0 3 3 1 3 1 2 2 3 1	0 1 3 0 0 0 0 0 3 1 2 3 1 3	2 2 1 1 0 0 3 3 0 1 1 1 1 1
2 3 2 2 1 1 3 3 1 0 3 3 3 3	2 2 1 3 3 0 2 3 1 3 1 2 3 3	1 1 2 0 3 0 0 0 2 3 0 2 1 1	2 2 1 0 0 0 1 2 0 2 2 1 3 3
0 1 2 3 2 0 0 1 1 2 1 1 3 3	2 0 3 0 2 1 1 2 2 2 2 1 1 1	3 3 3 0 1 2 1 1 3 2 0 2 0 2	1 0 0 3 2 2 1 0 0 0 0 3 2 1
1 2 0 1 1 2 2 1 3 3 3 2 2 1	0 2 0 1 3 1 0 1 0 0 2 2 0 0	1 2 2 1 2 3 1 3 0 1 1 1 2 2	1 0 0 0 0 0 2 2 2 2 2 0 3 1
0 0 1 3 1 3 1 3 2 0 3 0 1 0	1 0 1 1 1 1 0 0 0 2 3 0 3 0	3 3 3 3 0 0 2 2 1 0 0 1 1 1	1 1 1 1 1 3 1 3 1 1 3 3 0 2
2 1 1 1 1 2 2 1 0 1 3 0 0 0	1 2 3 0 1 0 0 1 3 0 0 0 3 1	2 1 0 0 2 2 3 2 1 2 1 3 1 0	2 0 3 0 2 3 2 3 2 2 0 0 0 3
1 3 0 0 0 2 3 1 1 0 0 3 0 2	2 2 2 2 0 3 3 0 2 0 2 0 1 2	1 3 1 0 0 0 0 1 1 1 1 2 0 0	1 2 2 2 0 0 2 0 1 2 2 2 0 0
0 0 0 2 2 0 1 3 1 0 0 0 1 1	3 1 2 1 0 3 1 2 2 0 0 0 1 3	1 1 1 0 0 3 0 0 3 3 3 1 1 0	3 2 3 3 0 3 3 3 3 2 3 0 3 1
2 1 2 2 2 2 1 0 3 2 2 1 2 2	3 1 1 2 1 2 1 2 0 1 2 3 3 0	3 3 1 1 1 2 0 0 2 2 3 3 2 2	1 1 1 1 1 2 0 2 3 2 2 2 0 1
0 2 2 3 1 0 3 2 2 1 0 3 0 0	3 2 1 1 3 3 3 1 2 3 0 1 0 2	1 2 2 1 1 2 0 0 0 0 2 1 3 1	3 0 0 1 0 2 2 0 1 2 2 0 0 3
0 3 0 3 1 1 3 0 1 2 0 1 2 3	2 1 1 0 3 2 1 1 1 1 0 0 2 3	2 0 2 2 1 3 3 3 2 3 1 1 1 0	2 1 1 3 0 0 0 3 3 3 2 0 2 3
0 2 2 3 1 3 1 1 0 2 1 3 1 1	2 0 2 3 2 1 0 0 3 3 0 1 3 2	1 1 2 3 3 1 2 2 1 1 1 1 0 0	3 0 0 3 1 1 1 3 3 3 1 0 0 2
2 3 0 3 2 2 2 1 3 3 0 0 1 2	0 1 1 0 3 1 3 3 3 3 3 0 0 1	0 0 0 1 1 2 2 1 3 1 1 1 0 3	3 3 1 2 2 1 1 2 2 2 3 2 3 3
2 0 0 1 1 1 3 0 2 1 2 3 3 3	3 0 0 1 3 2 2 3 2 0 1 2 1 2	3 0 0 0 0 2 2 1 1 3 1 3 3 1	1 1 0 3 3 3 3 1 2 1 1 3 3 1
3 0 3 3 1 3 3 0 2 2 1 3 0 2	0 0 2 2 2 2 1 2 2 2 3 3 2 0	2 0 0 3 3 0 3 1 0 1 2 0 0 1	0 3 3 2 1 2 3 3 0 0 1 0 3 0
3 0 1 1 1 3 1 1 3 3 0 0 3 3	0 1 2 2 1 1 1 1 1 0 3 0 0 1	2 1 1 1 2 3 1 2 3 3 1 1 1 1	3 1 1 1 0 1 1 1 0 0 0 2 2 2
3 3 0 0 2 2 0 2 0 2 1 0 2 2	0 2 0 0 1 3 3 2 2 2 1 1 3 3	3 3 1 0 2 1 3 1 3 0 1 2 2 1	1 1 1 0 0 0 1 2 3 2 3 0 0 2
1 3 3 3 3 2 1 2 1 2 2 2 3 0	2 2 3 3 1 2 1 2 1 1 3 2 1 2	1 1 1 3 3 0 3 3 1 1 0 0 1 2	3 3 2 0 0 3 0 3 2 1 3 0 1 0
1 2 0 0 3 3 1 1 0 2 2 2 2 3	1 3 2 3 3 2 2 2 0 0 0 1 2 2	0 2 0 0 0 0 3 3 3 3 3 3 2 0	0 3 1 2 3 1 2 2 1 2 2 0 0 2
2 1 1 2 1 3 3 2 2 0 2 1 0 3	3 3 3 1 0 0 2 0 3 0 1 3 3 2	2 0 1 1 3 2 0 2 2 3 0 3 1 1	3 1 0 0 3 3 0 1 1 2 0 1 3 2
3 3 3 2 1 3 2 1 0 2 3 0 1 0	1 0 1 2 1 0 2 2 0 3 0 0 0 1	1 2 3 2 2 2 0 3 0 2 3 3 3 1	0 3 1 0 1 3 0 0 0 2 1 2 3 2
1 0 0 0 1 1 3 0 3 0 1 2 1 0	3 0 0 2 2 0 3 3 0 1 3 3 0 0	0 0 2 0 2 1 3 1 1 3 2 3 2 3	1 0 0 2 3 3 1 0 3 0 0 0 0 3
0 3 0 1 1 2 3 3 3 2 1 3 1 2	3 0 2 1 1 0 0 1 1 1 3 1 2 1	3 1 2 3 0 2 2 3 3 1 1 0 3 2	3 0 3 0 2 2 3 3 2 3 2 2 2 1
0 1 2 0 1 1 2 3 0 0 3 1 2 0	3 1 0 0 0 1 0 0 2 0 3 0 2 3	0 2 2 3 3 3 3 3 3 3 2 1 2 2	3 3 1 2 3 3 3 0 3 0 0 3 3 1
0 3 2 2 3 0 1 2 0 3 3 0 0 3	0 2 1 1 1 0 2 1 1 2 0 1 1 0	3 0 3 3 3 1 3 2 3 3 0 3 2 2	2 3 1 2 0 2 0 3 2 0 3 0 0 2
1 1 2 2 2 3 0 0 1 1 3 1 0 1	2 0 1 3 1 1 1 2 2 1 1 2 3 1	1 2 3 1 2 3 1 2 0 0 3 2 3 0	0 2 0 2 1 1 0 1 2 3 0 0 0 2
2 2 1 2 1 1 1 2 3 3 3 3 2 3	2 3 2 2 3 1 1 0 1 0 0 1 1 0	0 1 0 0 3 2 2 0 2 2 0 2 0 0	1 3 0 2 3 2 0 0 1 0 3 2 2 2
1 3 2 0 3 3 3 3 3 0 1 1 0 0	3 3 3 1 3 3 1 1 3 1 1 2 1 0	0 1 3 2 0 3 0 3 2 3 1 0 2 2	2 0 3 0 2 3 0 0 3 1 3 3 3 1
0 2 1 3 2 0 2 0 2 2 2 2 2 3			

Table A.8: $(14, 205, 7)_4$ Code - Code205-1

A.9 Code205-2

0 0 3 0 1 0 2 3 1 2 0 2 2 2	2 2 1 3 1 1 0 0 3 1 0 1 0 0	3 2 3 2 2 0 2 0 0 2 2 2 2 3	3 0 3 0 2 2 2 2 2 3 3 3 3 3
2 1 2 2 1 3 2 0 3 3 1 3 2 2	3 0 1 3 1 2 1 0 3 2 1 0 2 3	1 0 1 2 3 3 2 1 0 1 1 1 0 0	3 3 1 3 1 0 0 0 2 3 2 2 2 2
0 0 1 2 0 2 1 1 2 1 1 1 3 3	3 3 3 2 0 3 2 0 0 3 0 1 2 1	2 3 2 2 2 2 1 0 2 0 3 0 0 3	3 3 0 2 2 0 0 2 1 0 1 0 0 2
1 0 0 0 0 0 0 1 3 1 0 2 1 3	3 2 2 3 1 0 2 1 1 3 1 0 3 2	3 0 2 0 0 2 2 1 2 2 2 0 1 2	2 3 1 3 3 0 2 2 2 2 1 3 3 2
0 0 0 3 1 0 3 3 3 3 1 1 1 0	0 3 3 1 1 0 0 2 2 2 1 3 1 0	0 2 2 2 3 3 1 0 2 2 2 0 2 1	1 0 3 1 1 1 1 1 2 2 3 2 2 1
0 3 2 3 1 0 3 2 3 2 0 0 0 2	3 3 1 0 1 0 1 2 3 1 0 0 1 3	3 2 3 0 0 3 3 1 2 1 0 1 1 1	1 2 2 0 1 1 0 0 1 2 2 0 2 2
1 3 0 0 3 1 2 0 2 0 2 3 3 2	1 2 3 2 0 0 1 1 0 1 1 3 3 2	0 0 1 3 2 2 3 0 0 2 0 1 1 0	1 2 1 3 1 0 2 2 0 0 2 3 1 0
3 0 0 2 3 2 3 3 2 0 3 1 3 1	0 1 2 2 2 2 2 2 1 1 3 0 3 1	1 0 0 0 1 2 3 1 2 2 0 0 0 0	2 0 3 0 0 0 0 2 1 0 1 3 3 3
0 2 3 3 1 1 2 1 3 1 3 0 1 1	1 3 3 3 2 1 2 3 0 1 0 0 2 2	0 1 2 0 1 3 2 0 0 3 2 2 1 3	2 1 1 1 2 1 1 0 1 3 2 1 2 2
1 2 2 3 2 3 3 3 0 0 2 3 3 2	0 2 1 2 3 2 2 2 1 0 2 1 3 2	3 1 0 0 2 0 2 2 3 3 1 2 1 0	0 2 3 1 0 3 3 1 0 0 2 1 3 0
0 3 2 3 3 2 1 1 0 3 2 0 1 0	2 0 0 2 3 1 0 3 1 1 2 2 2 3	2 0 0 0 0 3 3 3 1 1 0 0 1 2	3 1 2 1 1 3 1 1 3 2 2 2 2 2
1 1 3 3 3 3 3 1 1 2 1 3 0 0	3 0 3 2 0 2 2 2 1 3 0 0 0 0	0 2 3 3 3 2 0 2 3 3 3 3 2 0	1 0 1 3 3 3 1 3 3 2 2 3 1 3
2 0 1 2 1 1 3 0 2 2 1 1 2 1	0 2 0 3 0 2 3 2 0 0 3 0 0 1	1 1 0 1 3 3 2 0 2 0 0 0 1 3	1 1 2 2 2 1 3 0 2 0 0 1 3 0
2 1 3 1 3 2 3 1 1 2 0 1 2 1	3 2 3 3 3 0 1 2 0 2 0 3 0 3	3 3 3 2 2 2 3 3 2 2 2 3 0 0	0 1 1 2 3 1 0 1 1 1 1 3 2 1
2 1 1 1 3 0 2 0 3 2 0 0 3 0	0 3 0 0 0 1 2 1 1 3 1 2 1 2	3 3 3 3 3 3 0 2 1 1 2 2 2 0	0 2 2 1 0 3 0 0 0 3 0 3 2 3
1 2 0 2 2 0 0 2 2 3 1 2 3 3	3 3 2 0 1 1 1 1 0 3 1 2 2 1	3 3 3 0 0 1 1 3 2 3 2 0 3 2	2 0 0 0 1 2 0 0 1 0 3 2 0 2
0 2 1 1 2 3 3 3 1 2 3 0 0 0	1 1 0 0 0 3 2 2 2 3 1 0 2 2	1 1 1 0 1 3 0 0 2 0 2 1 2 1	3 3 1 2 3 1 1 0 1 0 0 0 3 2
2 1 2 2 0 2 1 3 1 3 3 2 1 1	1 1 0 0 1 3 1 1 3 3 3 3 1 2	3 2 2 0 3 2 1 1 2 2 1 3 3 0	2 0 2 1 2 0 3 0 3 3 2 3 0 3
1 1 2 3 1 1 1 0 0 3 1 3 3 3	2 3 0 0 1 0 2 2 3 1 2 2 3 1	0 2 0 2 0 0 1 0 3 2 1 3 1 3	2 0 2 2 3 1 2 1 2 1 2 3 0 2
2 3 2 1 0 2 3 3 3 0 1 1 0 0	2 0 3 3 3 3 1 0 3 1 1 0 3 1	2 2 3 2 3 0 3 2 2 1 1 3 0 0	1 0 3 3 2 1 2 0 3 3 2 1 1 2
0 3 0 1 2 2 0 3 0 3 1 0 3 0	1 2 2 1 2 0 3 1 0 0 0 1 0 1	0 0 0 3 3 2 2 3 1 2 3 0 3 3	3 1 0 1 3 1 3 2 1 1 0 3 3 1
3 3 1 1 3 0 2 2 1 0 3 2 3 3	2 3 2 3 1 3 1 1 3 2 1 1 1 3	2 3 1 1 2 3 2 0 2 2 0 2 2 2	0 3 3 1 2 3 3 1 2 2 3 3 3 1
1 3 2 2 2 3 1 1 1 3 3 2 0 2	3 0 2 0 3 3 0 2 1 0 0 2 3 2	2 0 0 3 2 0 1 1 0 3 3 3 1 0	0 2 3 3 2 2 2 0 1 1 1 3 3 3
1 1 3 1 3 1 2 2 1 3 3 1 0 1	1 0 1 3 2 3 2 2 1 3 1 3 2 3	1 2 1 1 0 2 1 3 2 3 0 3 3 2	0 0 0 0 2 1 2 0 3 3 3 0 2 3
1 0 3 1 2 3 0 0 1 3 0 3 1 1	2 2 0 2 2 2 2 0 1 3 2 2 1 2	1 1 0 3 3 0 2 3 0 1 3 2 2 3	2 1 1 1 0 0 1 2 3 1 1 3 0 3
2 1 2 0 0 1 2 2 1 3 2 3 3 0	2 2 0 0 0 0 0 3 2 3 0 0 3 1	0 1 0 0 2 1 2 2 2 2 2 3 2 1	2 1 0 3 0 2 2 1 1 1 0 0 2 3
2 2 1 0 0 3 2 1 2 3 3 1 2 3	3 3 2 0 1 0 3 3 3 0 2 3 2 3	3 1 2 2 2 2 2 3 1 2 1 2 2 2	1 3 3 1 0 0 2 0 1 2 2 0 3 3
0 0 1 1 3 0 1 0 2 1 2 3 0 0	0 1 3 3 0 0 0 3 1 1 1 1 0 1	2 2 1 1 0 3 3 3 1 3 1 3 1 1	0 2 0 2 0 3 3 1 1 1 3 2 0 3
0 0 1 1 3 2 3 2 3 3 3 0 1 2	1 3 2 2 0 0 0 2 2 3 0 3 0 0	1 0 1 0 2 3 0 3 3 1 2 0 3 2	3 1 2 3 2 3 1 0 0 3 3 1 0 0
3 1 0 0 0 0 1 3 1 3 2 3 0 1	2 3 3 3 3 1 3 2 1 3 3 3 3 3	2 3 1 1 3 3 0 0 1 0 2 0 0 1	0 3 3 2 3 3 2 2 0 0 0 0 3 3
2 2 1 1 1 1 3 1 2 0 2 2 3 3	1 0 2 2 1 0 1 3 3 1 1 2 2 0	3 1 0 2 2 1 1 0 3 0 3 2 1 3	2 3 0 2 3 3 2 2 0 2 3 1 1 0
0 3 3 3 1 2 0 1 1 2 0 0 3 0	2 2 0 3 3 3 1 1 3 0 2 2 1 0	1 2 1 2 3 3 3 0 3 3 0 1 1 2	3 3 2 1 3 0 1 3 0 3 0 0 0 1
3 1 1 2 1 3 3 3 0 0 1 2 3 1	3 0 2 2 1 3 3 0 1 3 3 3 3 1	0 0 3 0 2 2 2 3 3 1 1 1 1 1	2 2 2 0 0 2 3 0 2 3 2 3 2 1
3 1 0 1 1 2 2 2 0 0 2 2 0 2	1 3 1 2 1 2 3 0 3 1 1 3 1 3	1 1 1 1 1 0 2 1 1 1 2 2 2 0	3 3 3 3 0 1 1 1 1 1 0 2 3 3
1 1 0 1 2 2 1 2 1 2 0 2 1 1	1 0 1 3 0 0 2 3 3 0 1 0 0 0	1 3 2 1 3 2 0 1 1 1 2 2 3 1	0 1 3 3 1 1 1 3 0 2 3 3 0 0
0 0 2 2 0 1 3 2 3 0 0 1 1 1	2 1 2 2 0 0 1 0 0 0 0 1 1 2	1 2 3 0 1 0 1 2 0 3 3 2 2 0	2 0 2 0 2 0 2 1 1 0 1 1 2 1
1 1 2 0 3 2 2 0 1 0 3 3 0 3	2 2 1 2 2 1 1 1 2 1 1 0 0 1	2 2 1 1 2 2 3 2 0 0 1 1 2 3	3 2 2 2 1 0 0 2 0 1 0 2 1 1
1 0 3 0 3 0 1 0 0 3 0 0 3 3	1 2 0 1 0 1 0 1 1 1 1 0 0 0	0 1 2 0 3 3 3 2 3 2 1 0 1 0	2 3 3 0 1 3 3 3 2 2 1 2 0 1
0 1 0 0 3 3 1 3 3 0 0 0 2 1	2 2 2 2 3 2 0 0 0 0 2 1 0 0 2	3 0 0 1 1 3 1 2 2 2 3 1 1 3	3 1 3 3 3 3 0 0 0 2 1 1 3 3
3 1 1 3 2 2 3 3 1 3 3 1 1 3	2 2 0 0 0 3 0 0 2 1 1 2 1 0	2 2 2 0 3 0 0 1 3 1 1 1 3 3	3 3 1 1 3 1 1 2 3 0 1 1 2 0
3 0 2 1 0 0 0 3 1 1 3 3 2 2	3 3 0 0 1 1 0 0 0 1 3 3 1 0	3 0 1 1 0 2 1 1 3 3 3 2 0 0	2 3 2 2 0 1 3 1 1 0 3 0 3 0
3 1 3 1 0 0 3 3 2 0 3 1 2 0	3 2 2 2 2 2 1 0 3 1 2 3 3 1	1 3 3 0 2 3 1 3 1 3 0 1 3 0	2 2 1 3 3 3 3 2 1 0 0 3 2 2
2 2 3 3 2 0 2 3 3 0 2 1 3 3	3 1 3 0 1 2 2 1 0 1 1 3 2 2	3 1 1 0 2 0 0 1 1 2 1 2 0 3	0 3 3 0 3 0 3 0 0 1 3 3 0 2
0 1 1 1 1 1 2 3 3 0 0 3 1 1	2 2 0 1 0 3 2 2 2 2 0 3 0 3	0 2 2 1 2 1 1 3 3 3 3 3 0 2	1 0 2 1 1 1 1 1 0 0 1 0 0 2
3 2 1 0 3 2 2 0 0 1 2 0 0 0	2 0 0 3 0 1 3 0 0 1 2 1 3 1	2 3 0 3 2 1 3 1 3 1 0 2 0 2	2 3 1 1 3 1 0 3 2 3 3 2 1 1
1 1 0 2 1 3 1 0 0 2 2 2 0 0	1 1 1 2 3 2 0 0 0 3 2 3 2 0	0 1 1 1 3 0 3 1 3 0 1 1 0 2	3 2 2 1 2 1 2 2 1 1 2 0 2 3
0 1 1 0 0 2 0 0 0 1 3 3 3 3	1 1 1 2 0 0 2 0 3 3 3 1 3 1	0 0 0 2 2 2 1 1 1 3 0 1 2 0	3 3 3 3 3 3 3 3 1 0 0 0 1 1
1 0 3 2 0 0 0 3 0 0 2 0 2 2	3 1 1 2 1 1 2 2 2 2 1 0 3 0	0 0 3 2 1 2 1 3 3 2 2 0 0 1	2 3 3 0 1 2 0 1 0 0 0 2 1 1
0 0 2 0 1 1 1 2 0 2 0 2 3 0	0 0 2 2 2 0 0 3 2 2 0 3 3 2	0 2 2 3 3 0 0 0 1 3 2 2 0 0	3 2 1 2 2 0 1 3 3 2 0 0 2 2
1 2 3 2 3 2 3 2 2 2 2 0 1 3	2 1 1 3 2 2 1 2 2 2 3 2 2 0	0 1 3 3 0 2 2 3 2 3 0 2 2 1	1 3 0 3 0 3 0 1 2 2 2 2 1 1
3 0 1 0 0 3 1 0 1 2 2 1 1 0	0 0 3 3 0 3 1 0 1 0 1 2 2 0	0 2 0 0 3 3 3 2 0 3 2 2 2 2	0 3 2 1 0 2 1 0 2 2 3 1 1 2
2 2 0 3 3 0 0 0 0 3 3 1 2 2			

Table A.9: $(14, 205, 7)_4$ Code - Code205-2

Appendix B

Results

B.1 Direct vs Fuzzy Analysis for Each Code

The summary statistics of fitness value for four different combination of mutations, named as experiments (E1, E2, E3, and E4) with eight different ranges of sates have been presented from table B.1 to B.72 for each code. The results of both direct and fuzzy classification for each set of mutations are shown synchronously. Each table has maximum fitness (number of corrected errors), median fitness, interquartile range of 30 runs and the percentage of maximum fitness of both training and verification dataset for every eight ranges of states. The fitness values in each table are measured for all distances that are the sum of the fitness value of distance 1, 2 and 3 for an individual machine and also the fitness value of distance 1, 2 and 3 explicitly.

The violin plots demonstrate the distribution of fitness of distance 1, 2, and 3 for training and verification, for both direct and fuzzy classification for each range of state. The x-axis indicates each distance and the y-axis represents the fitness value or the corrected number of errors for each distance. The figure for each range of state is generated from the 30 runs for each experiment.

B.1.1 Code of length 12

Code55

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1024	991	19	51.7	1013	958	44	51.2
	1	454	427.5	10	68.8	442	412	19	67
	2	362	349.5	16	54.8	368	339.5	17	55.8
	3	229	213	18	34.7	241	201	7	36.5
4to8	All	1270	1243	27	64.1	1257	1222	33	63.5
	1	550	523.5	23	83.3	539	524	22	81.7
	2	455	442	10	68.9	460	444	9	69.7
	3	288	273	11	43.6	278	263	13	42.1
6to12	All	1421	1380	35	71.8	1391	1327	55	70.3
	1	589	566.5	18	89.2	573	554	22	86.8
	2	506	483	19	76.7	502	476	26	76.1
	3	343	328	19	52	337	298	12	51.1
6to18	All	1446	1378.5	59	73	1408	1319	54	71.1
	1	593	564.5	15	89.8	579	550	24	87.7
	2	516	489	25	78.2	499	472.5	23	75.6
	3	360	328	28	54.5	344	304.5	30	52.1
8to14	All	**1475**	1390.5	37	74.5	1399	1340	57	70.7
	1	598	566	20	90.6	592	553.5	28	89.7
	2	515	490	14	78	503	473	28	76.2
	3	374	336	16	56.7	337	304.5	24	51.1
8to18	All	1467	1404	62	74.1	**1445**	1342.5	49	73
	1	603	569	23	91.4	595	556	24	90.2
	2	520	490	25	78.8	517	476.5	22	78.3
	3	372	334.5	26	56.4	333	305	24	50.5
10to16	All	1451	1389	69	73.3	1419	1334.5	58	71.7
	1	594	569	23	90	588	557	15	89.1
	2	512	488.5	26	77.6	512	477.5	15	77.6
	3	375	330.5	21	56.8	353	302.5	23	53.5
14to18	All	1457	**1423**	44	73.6	1426	**1364**	42	72
	1	594	575	19	90	589	559.5	15	89.2
	2	518	494.5	11	78.5	514	479	22	77.9
	3	364	347.5	17	55.2	358	315.5	26	54.2

Table B.1: Code55, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1421	1376	40	71.8	1433	1355.5	46	72.4
	1	545	521	7	82.6	533	510	16	80.8
	2	503	479	14	76.2	495	476	21	75
	3	389	369	20	58.9	409	369	27	62
4to8	All	1664	1558.5	96	84	1639	1551.5	111	82.8
	1	602	575.5	28	91.2	595	570	28	90.2
	2	566	540	28	85.8	568	539	36	86.1
	3	496	445	46	75.2	479	437.5	46	72.6
6to12	All	1690	1620	61	85.4	1663	1594.5	67	84
	1	613	596	15	92.9	608	588	25	92.1
	2	579	555.5	25	87.7	577	554.5	20	87.4
	3	513	468	30	77.7	486	449	23	73.6
6to18	All	1704	1597	61	86.1	1670	1579.5	62	84.3
	1	622	597.5	20	94.2	613	588	20	92.9
	2	582	551	20	88.2	574	548	18	87
	3	513	456.5	36	77.7	492	447	16	74.5
8to14	All	1727	1618	68	87.2	1706	1600	74	86.2
	1	618	591.5	22	93.6	617	587	31	93.5
	2	596	551	22	90.3	586	551	26	88.8
	3	520	466	25	78.8	509	454.5	26	77.1
8to18	All	**1746**	1614.5	89	88.2	**1739**	1589	89	87.8
	1	625	597	22	94.7	619	589.5	25	93.8
	2	599	550.5	23	90.8	601	553	28	91.1
	3	522	468.5	32	79.1	519	448	30	78.6
10to16	All	1723	**1625**	72	87	1694	1595	58	85.6
	1	628	599	23	95.2	626	591.5	22	94.8
	2	588	554.5	18	89.1	585	555.5	19	88.6
	3	515	475.5	34	78	483	455.5	26	73.2
14to18	All	1710	1618	44	86.4	1694	**1600.5**	44	85.6
	1	617	600	15	93.5	625	594	13	94.7
	2	582	554.5	18	88.2	586	553	18	88.8
	3	514	460.5	27	77.9	504	456	21	76.4

Table B.2: Code55, Fuzzy Classification Fitness Result For Experiment 1

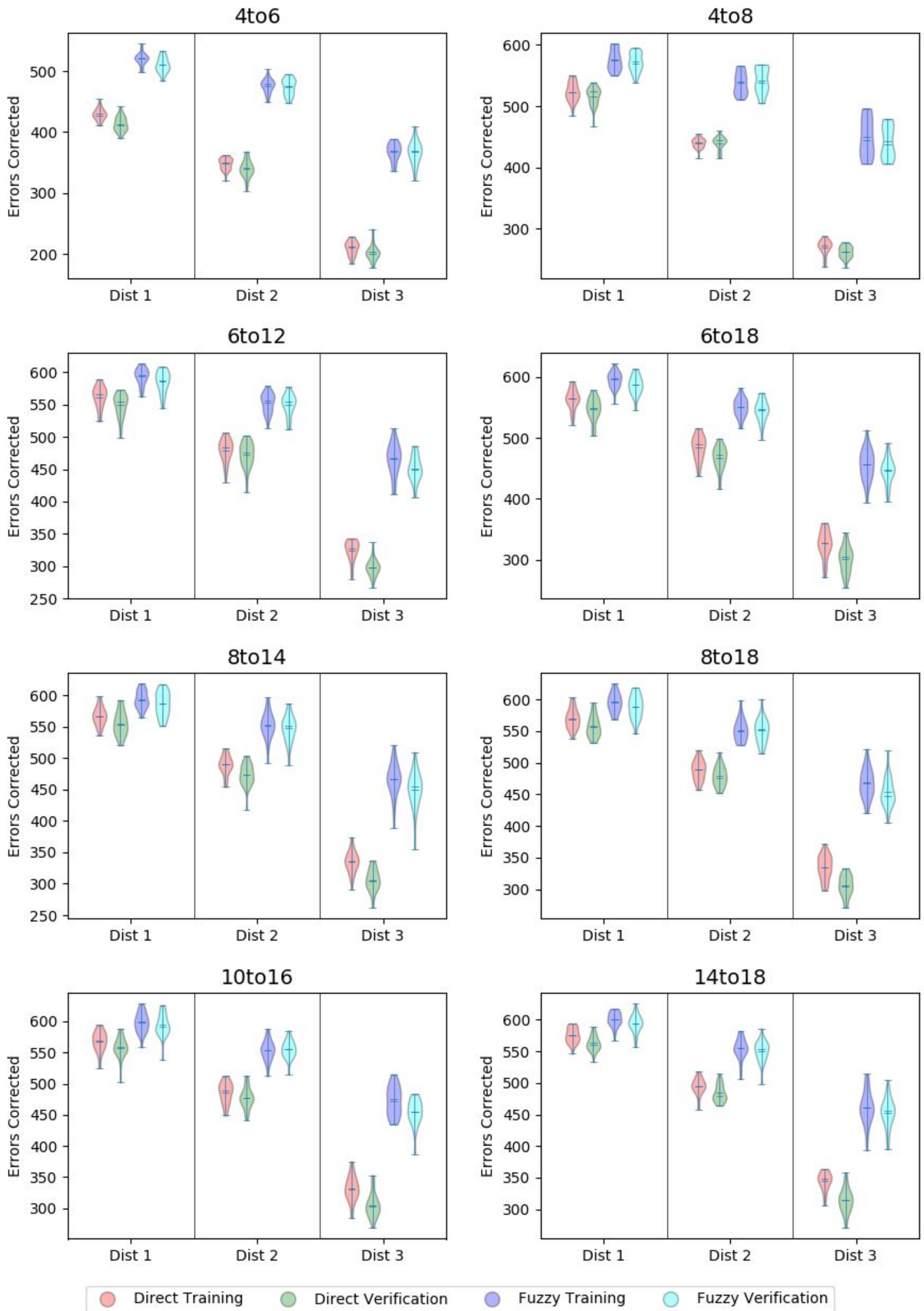

Figure B.1: Code55, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1024	998	15	51.7	1013	980	56	51.2
	1	446	432	13	67.6	442	422	20	67
	2	362	350	13	54.8	368	342	16	55.8
	3	230	217.5	21	34.8	241	205.5	25	36.5
4to8	All	1270	1242	51	64.1	1257	1218.5	70	63.5
	1	550	522	30	83.3	546	517	32	82.7
	2	455	441	10	68.9	448	440.5	24	67.9
	3	288	273	14	43.6	277	269	14	42
6to12	All	1428	1368.5	65	72.1	1400	1329.5	66	70.7
	1	598	562	21	90.6	593	553	22	89.8
	2	502	479.5	19	76.1	497	472.5	16	75.3
	3	360	328	25	54.5	336	296.5	33	50.9
6to18	All	1460	1397	46	73.7	1432	1343.5	46	72.3
	1	600	573	21	90.9	590	562.5	18	89.4
	2	514	485.5	20	77.9	506	480.5	15	76.7
	3	357	334.5	27	54.1	347	304	25	52.6
8to14	All	1455	1366.5	62	73.5	1460	1327	36	73.7
	1	590	563.5	24	89.4	589	549.5	19	89.2
	2	506	483.5	18	76.7	520	473	14	78.8
	3	363	324	25	55	361	299.5	16	54.7
8to18	All	1473	1389.5	91	74.4	**1425**	1340.5	65	72
	1	596	567.5	18	90.3	589	560	19	89.2
	2	515	487	27	78	505	476	24	76.5
	3	377	336.5	32	57.1	337	306.5	18	51.1
10to16	All	1418	1382	54	71.6	1394	1325	46	70.4
	1	588	564	15	89.1	586	555	15	88.8
	2	499	485	17	75.6	504	470	23	76.4
	3	360	329	32	54.5	328	298	19	49.7
14to18	All	**1476**	**1419.5**	51	74.5	1420	**1360**	52	71.7
	1	593	575.5	15	89.8	590	564	17	89.4
	2	522	497	17	79.1	513	482	16	77.7
	3	372	345.5	22	56.4	351	317	17	53.2

Table B.3: Code55, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1692	1373	40	85.5	1682	1355.5	58	84.9
	1	615	522.5	7	93.2	596	510.5	23	90.3
	2	578	472.5	14	87.6	574	472	21	87
	3	499	370	23	75.6	512	367.5	29	77.6
4to8	All	1664	1589.5	172	84	1639	1575	168	82.8
	1	602	584	44	91.2	601	572	38	91.1
	2	566	541	48	85.8	568	539	56	86.1
	3	497	462.5	80	75.3	479	460	60	72.6
6to12	All	**1744**	1619.5	95	88.1	1706	1600	73	86.2
	1	623	595.5	17	94.4	625	589.5	21	94.7
	2	596	553	29	90.3	584	554	27	88.5
	3	530	475.5	35	80.3	499	461	36	75.6
6to18	All	1740	1622.5	68	87.9	**1744**	1600.5	61	88.1
	1	628	598	18	95.2	623	590.5	17	94.4
	2	582	552	20	88.2	596	553.5	18	90.3
	3	533	471	33	80.8	525	453.5	22	79.5
8to14	All	1684	1624.5	61	85.1	1684	1603	68	85.1
	1	620	599	21	93.9	623	593	26	94.4
	2	583	552.5	19	88.3	579	561	20	87.7
	3	497	473	29	75.3	500	456.5	23	75.8
8to18	All	1697	**1630**	65	85.7	1662	**1610.5**	78	83.9
	1	619	603	24	93.8	613	594	18	92.9
	2	582	555	16	88.2	579	557.5	25	87.7
	3	506	473.5	33	76.7	489	452	31	74.1
10to16	All	1703	1619.5	75	86	1674	1600.5	73	84.5
	1	621	596.5	29	94.1	617	594	30	93.5
	2	582	550	22	88.2	581	552	18	88
	3	512	468	37	77.6	491	457	32	74.4
14to18	All	1728	1605.5	52	87.3	1699	1581.5	58	85.8
	1	629	601	18	95.3	626	593.5	20	94.8
	2	593	552	19	89.8	585	549	21	88.6
	3	519	454.5	25	78.6	500	441	25	75.8

Table B.4: Code55, Fuzzy Classification Fitness Result For Experiment 2

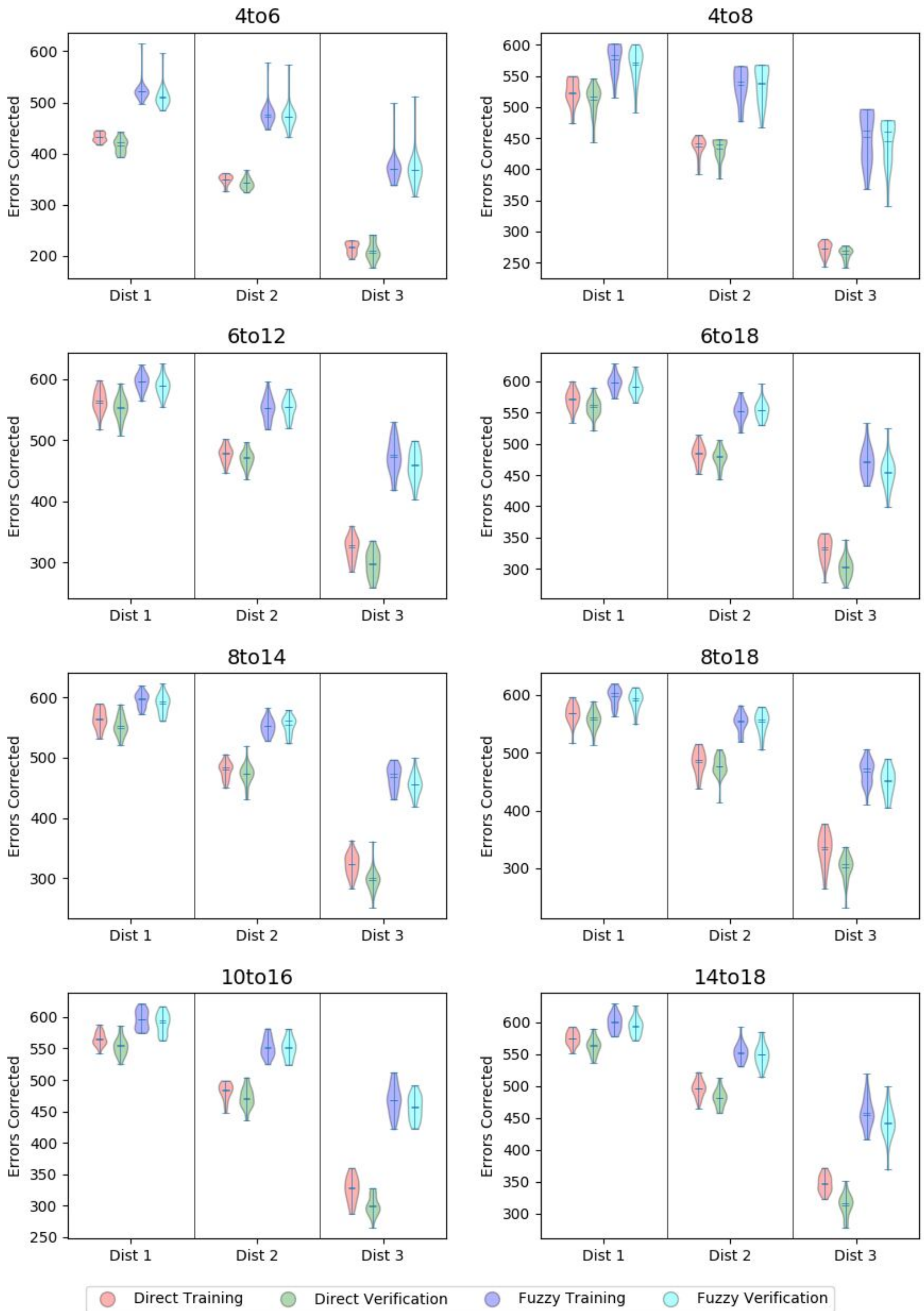

Figure B.2: Code55, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1024	990.5	25	51.7	1006	951.5	47	50.8
	1	454	432	17	68.8	442	409.5	20	67
	2	362	350.5	14	54.8	368	341.5	12	55.8
	3	230	211.5	16	34.8	241	203	26	36.5
4to8	All	1270	1239	47	64.1	1257	1211.5	46	63.5
	1	550	519	23	83.3	544	515	22	82.4
	2	455	440	8	68.9	452	432	24	68.5
	3	288	271	12	43.6	282	259	20	42.7
6to12	All	1420	1343	45	71.7	1360	1308	55	68.7
	1	587	559.5	24	88.9	572	550	22	86.7
	2	496	473.5	26	75.2	487	466.5	16	73.8
	3	347	314	24	52.6	333	287.5	16	50.5
6to18	All	1450	1394.5	85	73.2	1387	1329.5	56	70.1
	1	590	564	27	89.4	582	552.5	21	88.2
	2	514	488.5	27	77.9	503	477.5	28	76.2
	3	358	334.5	35	54.2	329	302	17	49.8
8to14	All	1440	1387	58	72.7	1407	1329.5	47	71.1
	1	587	569.5	23	88.9	585	557	23	88.6
	2	508	483	12	77	504	473.5	23	76.4
	3	359	329	28	54.4	337	299.5	24	51.1
8to18	All	1455	1392	45	73.5	1416	1341.5	54	71.5
	1	597	571.5	20	90.5	592	560.5	19	89.7
	2	515	489	22	78	514	479	20	77.9
	3	367	334	21	55.6	333	307.5	22	50.5
10to16	All	1467	1373.5	78	74.1	**1438**	1323.5	72	72.6
	1	595	568.5	23	90.2	585	556	23	88.6
	2	511	478.5	25	77.4	510	469.5	24	77.3
	3	372	322.5	34	56.4	347	296.5	20	52.6
14to18	All	**1488**	**1408**	55	75.2	1430	**1356**	75	72.2
	1	610	572	18	92.4	605	565.5	26	91.7
	2	521	493	21	78.9	505	479	22	76.5
	3	379	343.5	19	57.4	342	310.5	26	51.8

Table B.5: Code55, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1687	1362.5	50	85.2	1698	1352	53	85.8
	1	607	521.5	15	92	607	511	20	92
	2	578	474.5	22	87.6	595	472	22	90.2
	3	502	366	28	76.1	496	366.5	36	75.2
4to8	All	1674	1592	129	84.5	1691	1591.5	96	85.4
	1	607	584.5	32	92	596	587	27	90.3
	2	566	544	33	85.8	586	550.5	38	88.8
	3	504	457.5	65	76.4	509	465	45	77.1
6to12	All	1692	1611	82	85.5	1660	1600.5	64	83.8
	1	614	595.5	28	93	611	590.5	26	92.6
	2	575	549.5	28	87.1	573	554.5	22	86.8
	3	508	464.5	37	77	490	457	31	74.2
6to18	All	1690	1605.5	64	85.4	1668	1604.5	72	84.2
	1	614	598	17	93	611	592.5	21	92.6
	2	575	548.5	26	87.1	587	556.5	27	88.9
	3	501	464.5	35	75.9	492	457	36	74.5
8to14	All	**1741**	**1645**	98	87.9	**1711**	**1623.5**	67	86.4
	1	621	603	17	94.1	619	591	28	93.8
	2	600	562	30	90.9	596	559	27	90.3
	3	520	476	39	78.8	516	463	25	78.2
8to18	All	1721	1620.5	34	86.9	1692	1608	54	85.5
	1	619	599	16	93.8	621	596.5	18	94.1
	2	592	554	21	89.7	585	556	16	88.6
	3	513	470	35	77.7	496	455	27	75.2
10to16	All	1698	1614	62	85.8	1687	1595	76	85.2
	1	630	597.5	22	95.5	624	589	26	94.5
	2	587	548.5	21	88.9	587	551.5	25	88.9
	3	517	464	27	78.3	500	454	30	75.8
14to18	All	1727	1614.5	105	87.2	1697	1600	93	85.7
	1	628	601.5	16	95.2	630	592.5	25	95.5
	2	592	548	35	89.7	595	554	31	90.2
	3	515	463.5	40	78	505	450.5	41	76.5

Table B.6: Code55, Fuzzy Classification Fitness Result For Experiment 3

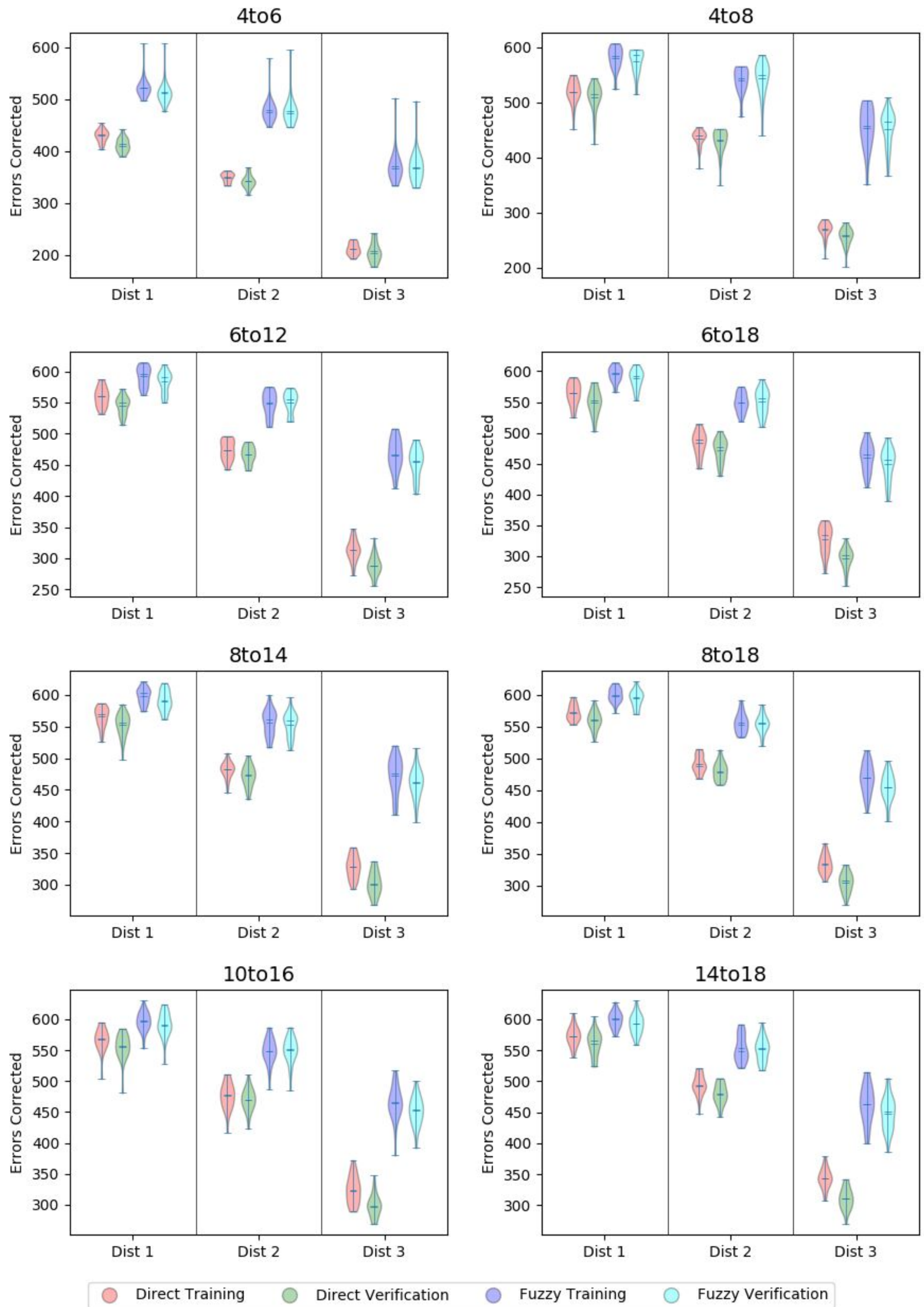

Figure B.3: Code55, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1024	990.5	23	51.7	1006	980.5	41	50.8
	1	454	435	11	68.8	442	422	15	67
	2	362	355.5	13	54.8	368	342	11	55.8
	3	227	212	26	34.4	241	206	17	36.5
4to8	All	1270	1235	30	64.1	1243	1212	40	62.8
	1	544	522.5	7	82.4	529	516	19	80.2
	2	455	442	10	68.9	448	439.5	18	67.9
	3	288	272.5	13	43.6	275	264.5	20	41.7
6to12	All	1402	1352.5	33	70.8	1377	1319.5	31	69.5
	1	575	559	22	87.1	569	550.5	19	86.2
	2	500	477.5	17	75.8	492	471.5	15	74.5
	3	353	315.5	22	53.5	338	297	14	51.2
6to18	All	1460	1380	32	73.7	1413	1340.5	55	71.4
	1	596	562.5	22	90.3	588	554.5	30	89.1
	2	517	483	20	78.3	509	477	22	77.1
	3	360	332	19	54.5	330	305.5	16	50
8to14	All	1440	1365	80	72.7	1411	1316	59	71.3
	1	584	566	22	88.5	575	550.5	23	87.1
	2	507	480	25	76.8	492	469.5	25	74.5
	3	361	327.5	44	54.7	357	295.5	23	54.1
8to18	All	**1487**	1409.5	36	75.1	1429	1351.5	45	72.2
	1	593	574	15	89.8	594	556.5	19	90
	2	537	489	18	81.4	504	481.5	16	76.4
	3	381	341	31	57.7	352	309	30	53.3
10to16	All	1463	1392.5	75	73.9	**1445**	1342	75	73
	1	594	561	28	90	590	555	25	89.4
	2	513	482.5	22	77.7	514	477.5	24	77.9
	3	372	326.5	36	56.4	364	307	25	55.2
14to18	All	1462	**1419**	36	73.8	1415	**1371**	55	71.5
	1	605	575	13	91.7	582	561.5	21	88.2
	2	520	497	15	78.8	508	484	23	77
	3	378	348.5	18	57.3	349	312.5	28	52.9

Table B.7: Code55, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1687	1364.5	49	85.2	1698	1356	44	85.8
	1	607	521	8	92	607	511	19	92
	2	578	476	22	87.6	595	472	19	90.2
	3	502	369	16	76.1	496	371.5	29	75.2
4to8	All	1664	1583.5	127	84	1639	1574	106	82.8
	1	602	579	32	91.2	595	571.5	22	90.2
	2	566	546.5	35	85.8	570	545.5	38	86.4
	3	496	455.5	61	75.2	479	460	50	72.6
6to12	All	1700	1607.5	59	85.9	1687	**1604.5**	65	85.2
	1	618	592.5	20	93.6	613	586	14	92.9
	2	575	551	18	87.1	583	552.5	27	88.3
	3	511	470	31	77.4	501	458	33	75.9
6to18	All	1716	1597	65	86.7	**1707**	1582.5	58	86.2
	1	622	592.5	17	94.2	617	587	19	93.5
	2	572	542.5	20	86.7	580	550	24	87.9
	3	524	462.5	25	79.4	510	450	28	77.3
8to14	All	1693	1607.5	65	85.5	1674	1590.5	79	84.5
	1	617	595.5	17	93.5	612	588	23	92.7
	2	579	552	20	87.7	579	553	27	87.7
	3	508	461.5	30	77	488	451.5	32	73.9
8to18	All	1706	1615.5	68	86.2	1656	1585	72	83.6
	1	623	595	19	94.4	617	591	26	93.5
	2	587	553	26	88.9	578	550.5	16	87.6
	3	504	459.5	30	76.4	486	442.5	27	73.6
10to16	All	1714	1597	59	86.6	1698	1579.5	55	85.8
	1	626	590.5	20	94.8	625	583.5	17	94.7
	2	587	551.5	16	88.9	578	547.5	21	87.6
	3	514	464.5	29	77.9	495	449	27	75
14to18	All	**1717**	**1623**	47	86.7	1668	1594.5	55	84.2
	1	630	600.5	13	95.5	626	593	20	94.8
	2	588	553.5	10	89.1	578	551	26	87.6
	3	501	468	28	75.9	488	450.5	27	73.9

Table B.8: Code55, Fuzzy Classification Fitness Result For Experiment 4

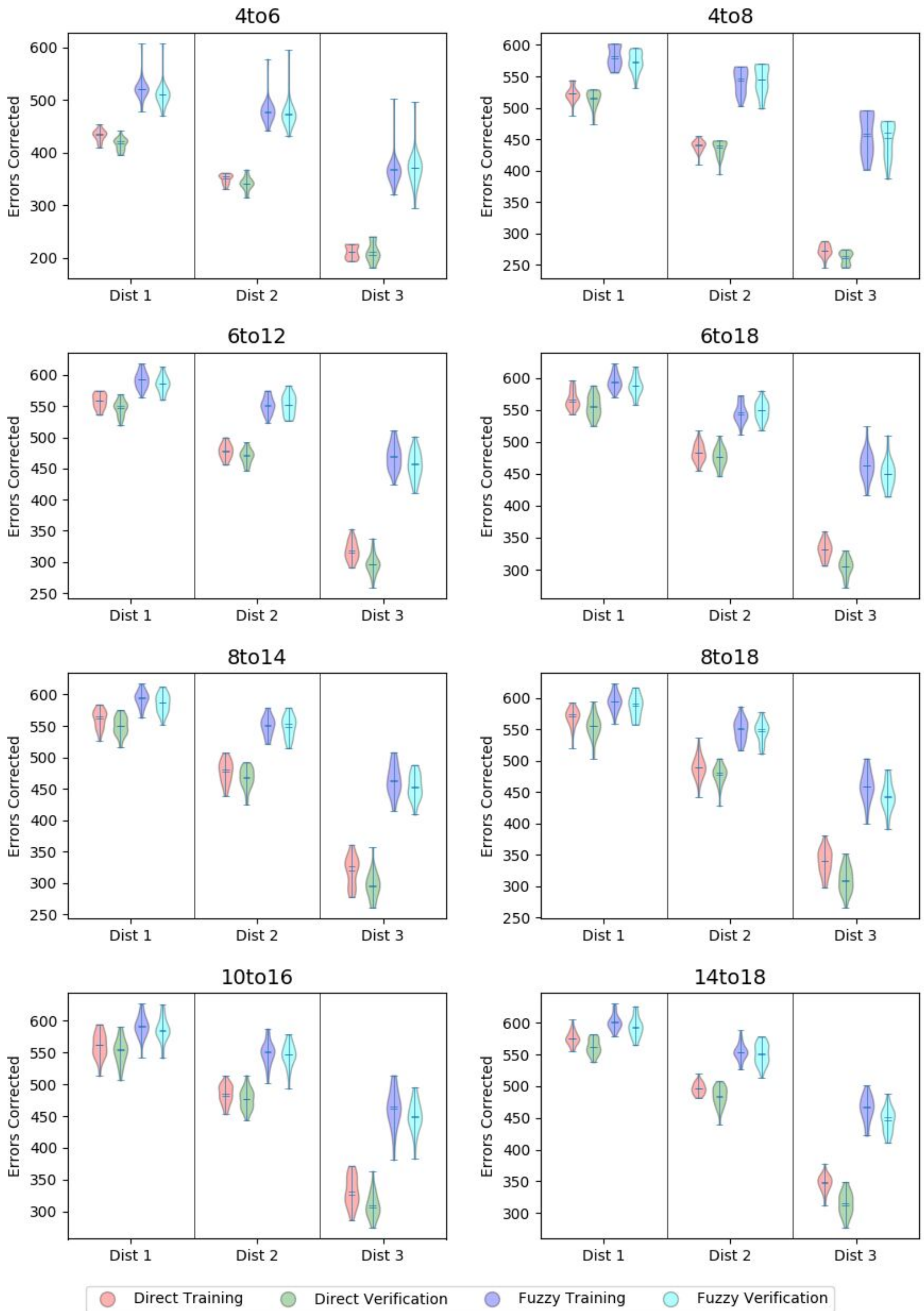

Figure B.4: Code55, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code60-1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1085	1051	27	50.2	1102	1032	42	51
	1	484	470	8	67.2	502	477	31	69.7
	2	378	361.5	11	52.5	387	362.5	28	53.8
	3	249	223.5	12	34.6	243	208	21	33.8
4to8	All	1335	1294.5	34	61.8	1340	1268.5	70	62
	1	586	561	27	81.4	587	553.5	42	81.5
	2	487	454.5	12	67.6	479	454	20	66.5
	3	301	283	12	41.8	281	260.5	19	39
6to12	All	1529	1486	55	70.8	1482	1434.5	57	68.6
	1	650	615	20	90.3	626	608.5	10	86.9
	2	535	511.5	21	74.3	542	504.5	26	75.3
	3	380	349	22	52.8	350	315.5	30	48.6
6to18	All	1570	1508.5	53	72.7	1518	1450	54	70.3
	1	640	619.5	22	88.9	648	618.5	18	90
	2	559	523	19	77.6	540	508.5	18	75
	3	388	363	30	53.9	357	320.5	29	49.6
8to14	All	1540	1469.5	65	71.3	1501	1414.5	70	69.5
	1	646	613	27	89.7	629	602	26	87.4
	2	539	508.5	26	74.9	532	499	29	73.9
	3	382	349	28	53.1	356	314	32	49.4
8to18	All	1570	1512	34	72.7	1505	1454	43	69.7
	1	658	622	19	91.4	632	612	18	87.8
	2	548	516	20	76.1	536	510.5	20	74.4
	3	401	364	23	55.7	369	324.5	29	51.2
10to16	All	**1594**	1482.5	78	73.8	**1523**	1432	53	70.5
	1	646	618	17	89.7	635	611.5	18	88.2
	2	543	516	23	75.4	537	503.5	27	74.6
	3	413	354	44	57.4	366	316	39	50.8
14to18	All	1578	**1533.5**	39	73.1	1520	**1458.5**	45	70.4
	1	650	625.5	15	90.3	640	622	20	88.9
	2	560	535	15	77.8	532	508	19	73.9
	3	402	375.5	17	55.8	361	327.5	16	50.1

Table B.9: Code60-1, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1587	1472	68	73.5	1555	1477	78	72
	1	606	568	27	84.2	590	566	16	81.9
	2	542	509.5	23	75.3	546	515	26	75.8
	3	439	399.5	16	61	422	392	31	58.6
4to8	All	1805	1676.5	118	83.6	1824	1664	141	84.4
	1	654	626.5	29	90.8	655	621	36	91
	2	616	578.5	39	85.6	628	572.5	46	87.2
	3	545	480	59	75.7	542	469	65	75.3
6to12	All	1840	1760	54	85.2	1840	**1741.5**	92	85.2
	1	675	645	17	93.8	666	642.5	20	92.5
	2	631	603	18	87.6	625	594.5	30	86.8
	3	549	513.5	32	76.2	553	510.5	49	76.8
6to18	All	1851	**1771.5**	63	85.7	1853	1736.5	90	85.8
	1	675	650.5	16	93.8	676	651	26	93.9
	2	636	607.5	24	88.3	638	597	22	88.6
	3	549	512.5	30	76.2	544	490	43	75.6
8to14	All	1868	1729.5	73	86.5	1832	1713.5	86	84.8
	1	674	641.5	33	93.6	668	637	24	92.8
	2	632	596.5	30	87.8	620	587	32	86.1
	3	565	499	36	78.5	545	492	33	75.7
8to18	All	1856	1760	72	85.9	1813	1737	87	83.9
	1	679	647.5	27	94.3	677	648.5	24	94
	2	634	598	28	88.1	627	596.5	39	87.1
	3	544	510	27	75.6	539	496	35	74.9
10to16	All	1842	1770.5	67	85.3	**1876**	1735.5	89	86.9
	1	669	649	22	92.9	679	640.5	23	94.3
	2	634	603.5	20	88.1	639	595	24	88.8
	3	548	507	38	76.1	558	498.5	42	77.5
14to18	All	**1873**	1762.5	83	86.7	1849	1725	91	85.6
	1	671	648	16	93.2	674	648.5	25	93.6
	2	650	608.5	26	90.3	636	588.5	23	88.3
	3	553	505.5	41	76.8	541	494	46	75.1

Table B.10: Code60-1, Fuzzy Classification Fitness Result For Experiment 1

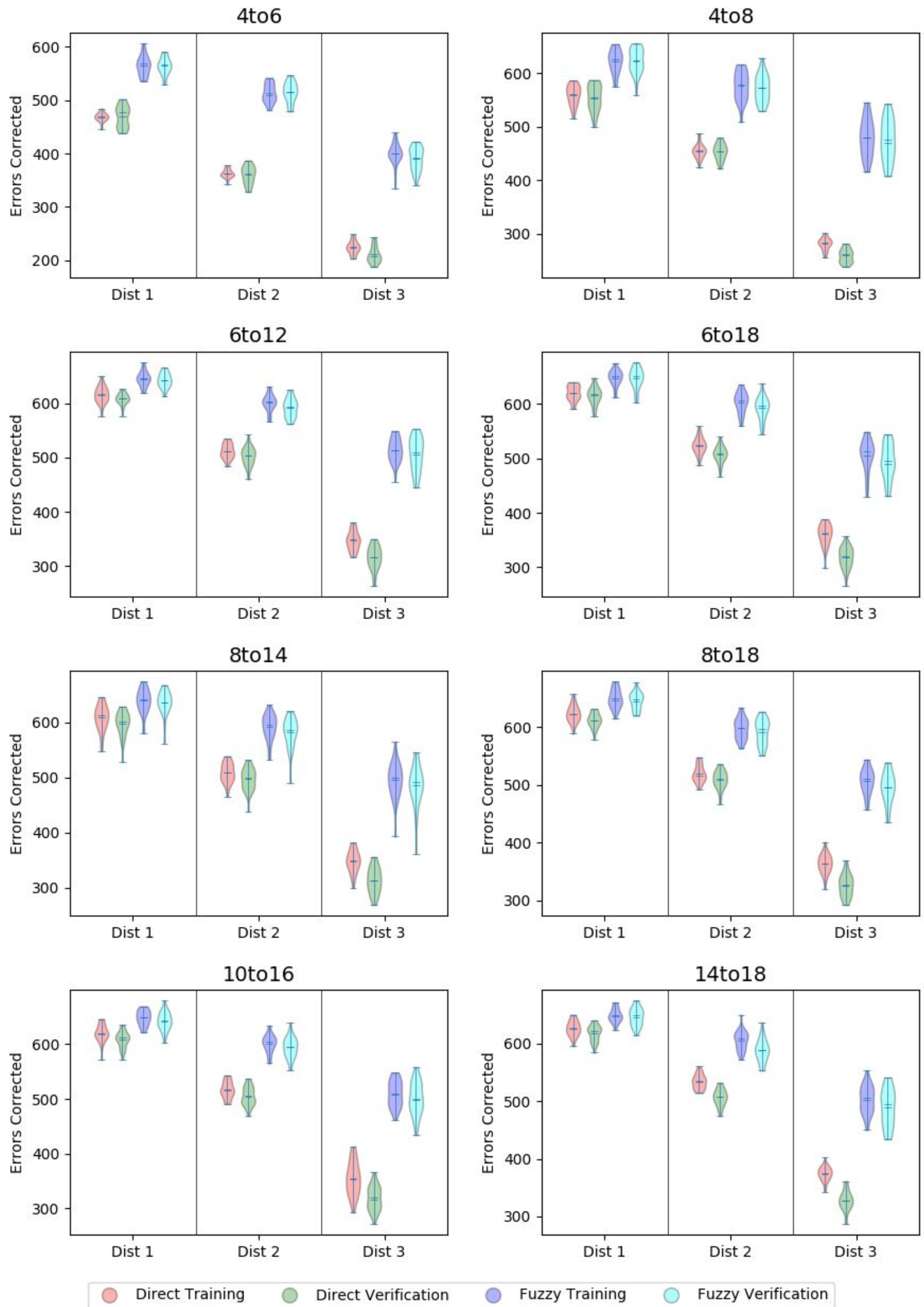

Figure B.5: Code60-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1085	1056	26	50.2	1102	1035.5	32	51
	1	494	472	5	68.6	502	475	28	69.7
	2	375	363.5	12	52.1	384	360	16	53.3
	3	249	222	11	34.6	243	208	18	33.8
4to8	All	1339	1306.5	59	62	1340	1267	47	62
	1	582	561.5	27	80.8	587	555	33	81.5
	2	487	455	14	67.6	479	456	24	66.5
	3	302	286	15	41.9	279	265	26	38.8
6to12	All	1536	1482	62	71.1	1508	1422.5	73	69.8
	1	637	610.5	24	88.5	632	606	19	87.8
	2	533	514	26	74	530	496.5	20	73.6
	3	390	353	35	54.2	365	315.5	39	50.7
6to18	All	1575	1514	87	72.9	1513	**1452**	69	70
	1	641	622.5	20	89	643	614.5	13	89.3
	2	557	524	27	77.4	532	503	25	73.9
	3	415	368.5	40	57.6	369	328	45	51.2
8to14	All	1578	1502.5	80	73.1	1505	1445.5	62	69.7
	1	656	626	37	91.1	638	608	20	88.6
	2	555	521	20	77.1	533	504.5	27	74
	3	410	357	29	56.9	369	323.5	41	51.2
8to18	All	1597	1512	63	73.9	**1532**	1445	58	70.9
	1	649	619.5	24	90.1	635	607	20	88.2
	2	567	525	22	78.8	535	511	28	74.3
	3	401	372	35	55.7	375	327	30	52.1
10to16	All	1610	1478	59	74.5	1529	1427.5	64	70.8
	1	644	614.5	20	89.4	639	610	15	88.8
	2	543	514.5	12	75.4	540	496.5	22	75
	3	424	351	36	58.9	367	317.5	41	51
14to18	All	**1614**	**1533**	47	74.7	1518	1450.5	46	70.3
	1	653	625.5	19	90.7	638	611.5	20	88.6
	2	558	531	19	77.5	531	512	20	73.8
	3	411	372	27	57.1	379	327	23	52.6

Table B.11: Code60-1, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1592	1474.5	70	73.7	1602	1437	57	74.2
	1	589	565	23	81.8	609	559	19	84.6
	2	550	517	20	76.4	550	510	24	76.4
	3	456	398	26	63.3	443	383	41	61.5
4to8	All	1805	1682	113	83.6	1812	1653	139	83.9
	1	654	626	32	90.8	652	624	37	90.6
	2	616	582	49	85.6	619	573.5	42	86
	3	545	473	52	75.7	542	464	66	75.3
6to12	All	1847	1746.5	64	85.5	1840	1716	83	85.2
	1	676	638.5	24	93.9	681	637	28	94.6
	2	637	597	21	88.5	626	586	27	86.9
	3	552	502.5	33	76.7	547	488	50	76
6to18	All	1838	**1762.5**	39	85.1	1813	**1737**	67	83.9
	1	669	646.5	13	92.9	672	650	16	93.3
	2	632	604	14	87.8	620	589.5	16	86.1
	3	546	507.5	26	75.8	533	497	38	74
8to14	All	**1906**	1741	55	88.2	**1877**	1714.5	55	86.9
	1	681	641	27	94.6	676	636.5	20	93.9
	2	649	596	17	90.1	636	587.5	33	88.3
	3	576	504.5	38	80	565	487	37	78.5
8to18	All	1817	1738	83	84.1	1798	1712	93	83.2
	1	665	642.5	20	92.4	664	638	15	92.2
	2	627	594	23	87.1	616	585.5	15	85.6
	3	550	501	29	76.4	537	491.5	57	74.6
10to16	All	1811	1753	52	83.8	1802	1734.5	56	83.4
	1	663	644.5	20	92.1	672	641.5	19	93.3
	2	625	599.5	26	86.8	625	585.5	24	86.8
	3	552	513	27	76.7	531	504	26	73.8
14to18	All	1860	1745	59	86.1	1842	1719.5	66	85.3
	1	680	652.5	14	94.4	667	641	23	92.6
	2	638	599	23	88.6	622	587	26	86.4
	3	543	503.5	25	75.4	553	489.5	28	76.8

Table B.12: Code60-1, Fuzzy Classification Fitness Result For Experiment 2

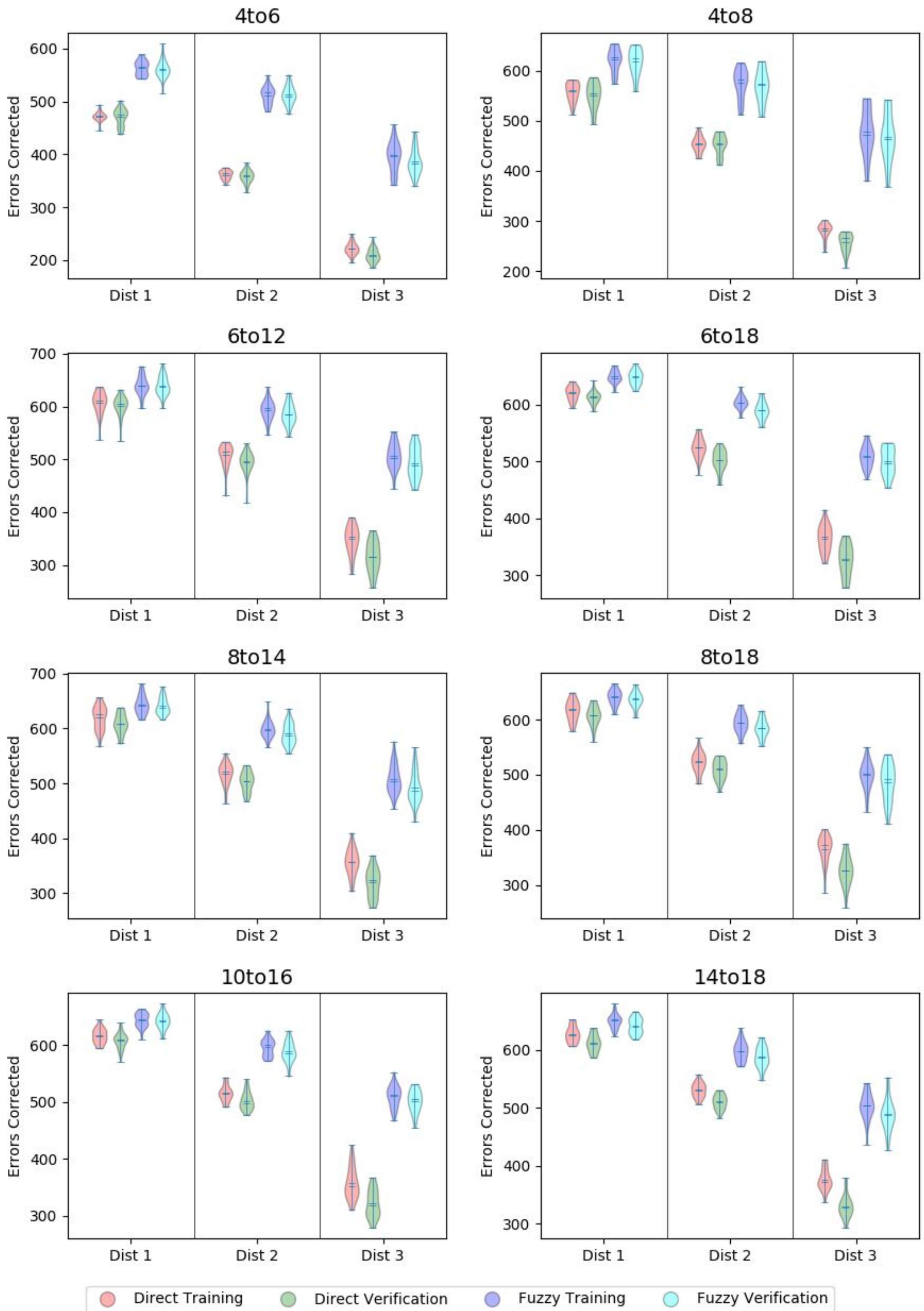

Figure B.6: Code60-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1085	1068	23	50.2	1102	1040.5	69	51
	1	488	472	7	67.8	502	480	35	69.7
	2	375	363	9	52.1	387	364.5	12	53.8
	3	249	225	16	34.6	243	211	27	33.8
4to8	All	1339	1314	51	62	1340	1276	63	62
	1	582	566.5	26	80.8	587	557	32	81.5
	2	487	455	25	67.6	479	454	9	66.5
	3	302	284.5	14	41.9	281	261	21	39
6to12	All	1529	1459	79	70.8	1468	1417.5	70	68
	1	639	610	21	88.8	621	603	21	86.2
	2	544	501.5	26	75.6	523	497	24	72.6
	3	387	343.5	43	53.8	346	314	42	48.1
6to18	All	1586	1513	69	73.4	1525	1445.5	55	70.6
	1	646	613	26	89.7	633	610	18	87.9
	2	570	526	32	79.2	537	509	21	74.6
	3	400	361.5	33	55.6	374	325	31	51.9
8to14	All	1560	1453.5	88	72.2	1515	1393.5	89	70.1
	1	635	607.5	22	88.2	644	600.5	35	89.4
	2	553	508	24	76.8	517	491.5	33	71.8
	3	411	342.5	42	57.1	354	309	43	49.2
8to18	All	1584	1469.5	73	73.3	1502	1422	49	69.5
	1	648	611.5	24	90	630	606	18	87.5
	2	551	511	29	76.5	523	496	18	72.6
	3	414	340	43	57.5	362	315.5	33	50.3
10to16	All	1589	1481	87	73.6	**1558**	1423	61	72.1
	1	648	617.5	23	90	637	610	18	88.5
	2	563	518.5	29	78.2	545	500.5	25	75.7
	3	414	360.5	41	57.5	376	313.5	30	52.2
14to18	All	**1590**	**1541**	63	73.6	1543	**1466.5**	58	71.4
	1	651	632.5	20	90.4	637	621	15	88.5
	2	559	532	18	77.6	541	516	22	75.1
	3	414	374	29	57.5	375	331	28	52.1

Table B.13: Code60-1, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1587	1500	55	73.5	1555	1497.5	84	72
	1	606	571	19	84.2	590	569	20	81.9
	2	542	520.5	22	75.3	546	527	29	75.8
	3	448	410	28	62.2	441	403	34	61.2
4to8	All	1805	1675	110	83.6	1812	1664	125	83.9
	1	654	626	28	90.8	652	632	32	90.6
	2	616	587	49	85.6	619	574	35	86
	3	545	483.5	49	75.7	542	471	47	75.3
6to12	All	1835	1751	80	85	**1841**	1721	80	85.2
	1	665	642	22	92.4	664	637	24	92.2
	2	620	592	24	86.1	629	583.5	24	87.4
	3	567	509	34	78.8	566	502.5	28	78.6
6to18	All	1823	1752	68	84.4	1823	1731.5	76	84.4
	1	672	645.5	22	93.3	667	642.5	20	92.6
	2	633	600	16	87.9	618	589	23	85.8
	3	554	504.5	33	76.9	557	503.5	45	77.4
8to14	All	1823	1731.5	51	84.4	1823	1709	61	84.4
	1	664	643	15	92.2	670	633.5	22	93.1
	2	623	593	19	86.5	614	585	18	85.3
	3	554	492.5	35	76.9	557	489	31	77.4
8to18	All	1838	1737.5	61	85.1	1829	1728	76	84.7
	1	676	641	22	93.9	665	641	15	92.4
	2	623	600.5	27	86.5	620	582.5	17	86.1
	3	556	504	35	77.2	547	508.5	51	76
10to16	All	1841	1744	78	85.2	1817	1720	63	84.1
	1	666	644.5	23	92.5	670	644.5	29	93.1
	2	639	600.5	28	88.8	624	584.5	25	86.7
	3	545	500	29	75.7	543	481.5	39	75.4
14to18	All	**1867**	**1773.5**	70	86.4	1812	**1758**	56	83.9
	1	673	657	18	93.5	676	651	13	93.9
	2	648	607.5	21	90	623	590.5	25	86.5
	3	548	513	35	76.1	539	501	30	74.9

Table B.14: Code60-1, Fuzzy Classification Fitness Result For Experiment 3

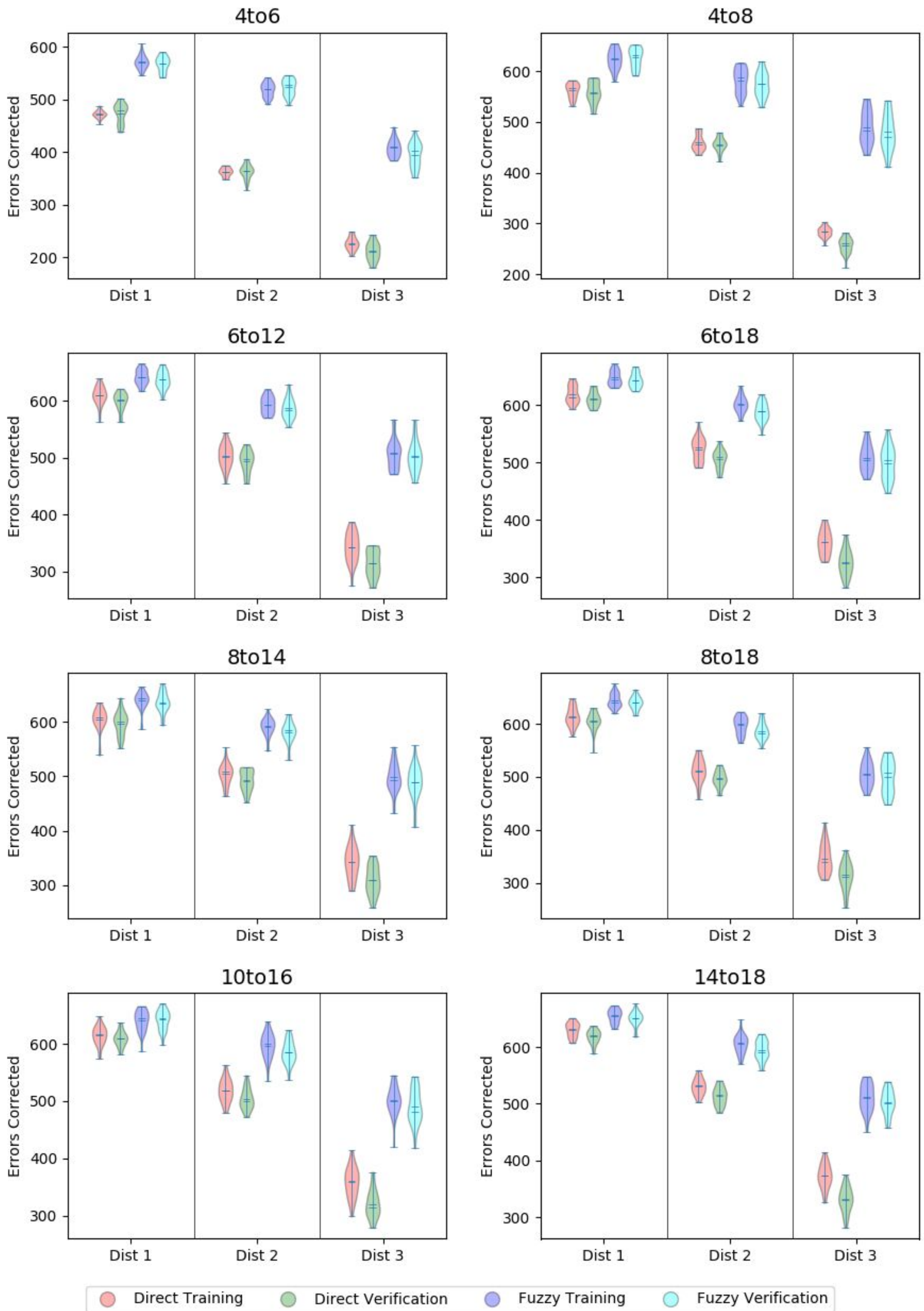

Figure B.7: Code60-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1085	1068	26	50.2	1102	1039	65	51
	1	488	470.5	10	67.8	502	479	30	69.7
	2	375	365	12	52.1	387	363.5	27	53.8
	3	249	225	11	34.6	243	208	28	33.8
4to8	All	1335	1302.5	52	61.8	1340	1266	56	62
	1	582	559.5	31	80.8	587	545.5	49	81.5
	2	487	459	15	67.6	479	454	20	66.5
	3	299	282	13	41.5	281	256	23	39
6to12	All	1545	1455.5	87	71.5	1484	1407	60	68.7
	1	638	608.5	16	88.6	633	596.5	18	87.9
	2	536	507.5	25	74.4	516	498.5	23	71.7
	3	390	336.5	51	54.2	374	312	48	51.9
6to18	All	1572	1490.5	69	72.8	1521	1419.5	77	70.4
	1	641	619	15	89	636	604.5	20	88.3
	2	553	523.5	27	76.8	535	500	27	74.3
	3	400	354	32	55.6	374	319.5	46	51.9
8to14	All	1553	1476.5	89	71.9	1507	1410.5	62	69.8
	1	651	608	23	90.4	626	603	26	86.9
	2	545	514	24	75.7	534	495.5	26	74.2
	3	402	346.5	39	55.8	372	314.5	29	51.7
8to18	All	1596	1509	98	73.9	1513	1431.5	66	70
	1	644	617	20	89.4	634	607	28	88.1
	2	560	517	20	77.8	525	504	23	72.9
	3	402	365.5	47	55.8	371	320	33	51.5
10to16	All	1603	1488	79	74.2	1494	1429.5	59	69.2
	1	636	618.5	20	88.3	631	604.5	21	87.6
	2	565	518	23	78.5	522	502.5	14	72.5
	3	421	360.5	33	58.5	366	322.5	29	50.8
14to18	All	**1625**	**1529**	53	75.2	**1522**	**1472.5**	69	70.5
	1	659	622	21	91.5	646	621.5	24	89.7
	2	573	534	23	79.6	546	510.5	21	75.8
	3	419	374	21	58.2	355	332	22	49.3

Table B.15: Code60-1, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1587	1495.5	58	73.5	1586	1488	86	73.4
	1	610	570.5	20	84.7	597	570.5	20	82.9
	2	543	521.5	24	75.4	558	522.5	20	77.5
	3	439	407	20	61	444	404.5	31	61.7
4to8	All	1805	1650	153	83.6	1812	1648	143	83.9
	1	654	620.5	34	90.8	651	618.5	41	90.4
	2	615	574	56	85.4	619	566	44	86
	3	545	467.5	67	75.7	542	463.5	60	75.3
6to12	All	1812	1743	56	83.9	1822	1729	64	84.4
	1	656	640	21	91.1	664	637	17	92.2
	2	619	596.5	25	86	619	589	28	86
	3	549	515	32	76.2	546	506.5	29	75.8
6to18	All	1836	1749	70	85	1831	1735	82	84.8
	1	671	643	17	93.2	673	640	21	93.5
	2	632	602.5	22	87.8	623	593	29	86.5
	3	565	509.5	45	78.5	550	503.5	50	76.4
8to14	All	1795	1739	53	83.1	1797	1708	66	83.2
	1	669	637.5	12	92.9	664	634	23	92.2
	2	622	591.5	22	86.4	611	582	19	84.9
	3	537	500.5	29	74.6	533	488.5	34	74
8to18	All	1813	1751.5	71	83.9	1818	1717.5	57	84.2
	1	665	640.5	18	92.4	665	636.5	17	92.4
	2	626	599	25	86.9	620	586.5	31	86.1
	3	539	507	46	74.9	548	492	39	76.1
10to16	All	1813	1742	64	83.9	1821	1721	59	84.3
	1	668	646	18	92.8	659	640	21	91.5
	2	628	590.5	19	87.2	616	583	24	85.6
	3	539	502	32	74.9	546	494	45	75.8
14to18	All	**1853**	**1763**	95	85.8	**1872**	**1737**	113	86.7
	1	678	648.5	21	94.2	679	650.5	17	94.3
	2	644	605.5	29	89.4	637	592	31	88.5
	3	545	502.5	41	75.7	556	493	61	77.2

Table B.16: Code60-1, Fuzzy Classification Fitness Result For Experiment 4

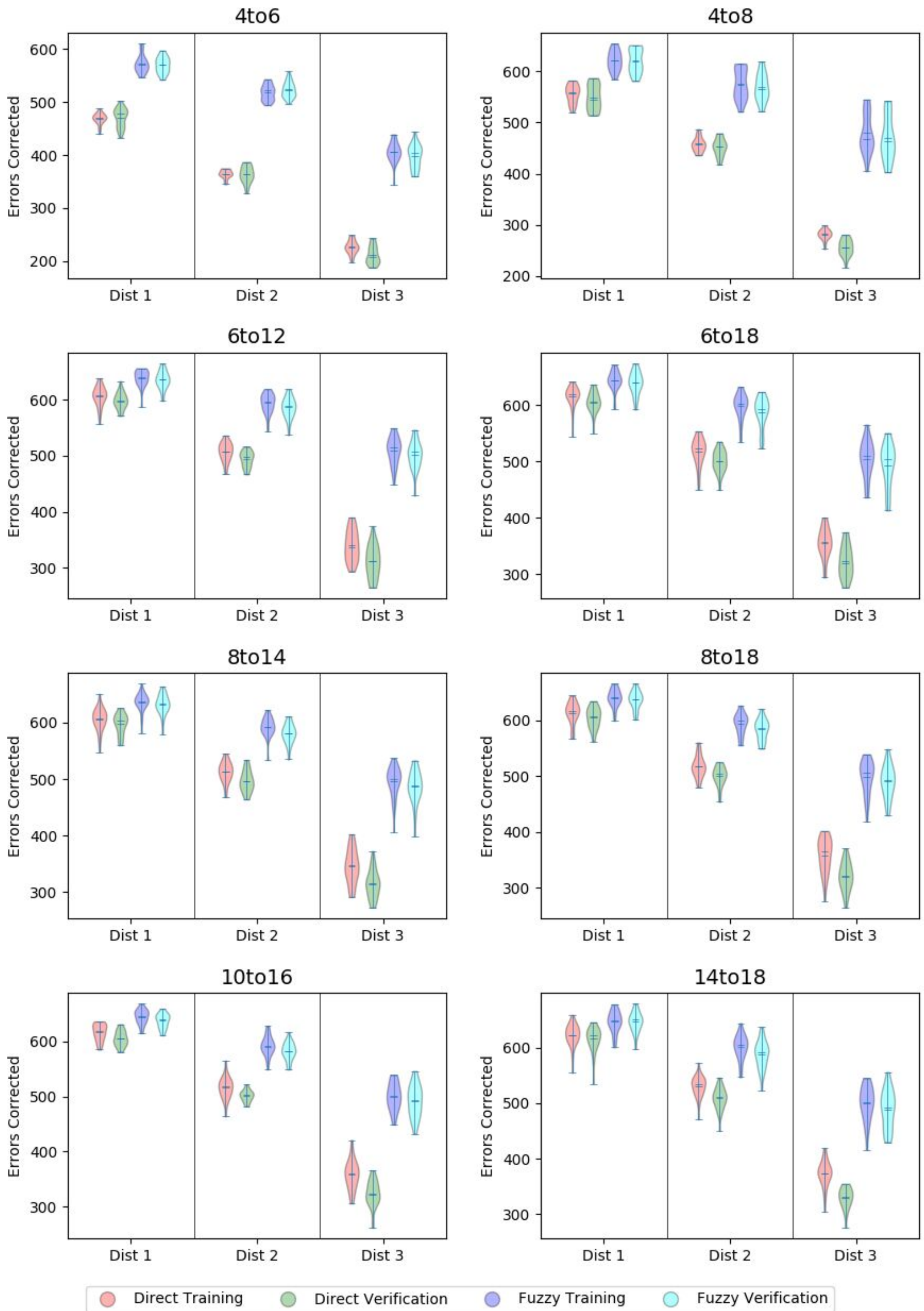

Figure B.8: Code60-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code60-2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1068	1051.5	17	49.4	1052	1022	29	48.7
	1	485	456	26	67.4	495	471	20	68.8
	2	374	364	7	51.9	364	350	15	50.6
	3	249	223	26	34.6	218	203	23	30.3
4to8	All	1327	1267.5	56	61.4	1319	1232	83	61.1
	1	585	545	16	81.2	584	536.5	24	81.1
	2	476	452	14	66.1	474	438.5	25	65.8
	3	292	270	21	40.6	290	257	23	40.3
6to12	All	1500	1451.5	34	69.4	1463	1395	41	67.7
	1	631	598.5	20	87.6	626	596.5	17	86.9
	2	543	514.5	16	75.4	522	489	17	72.5
	3	353	337.5	20	49	343	306.5	18	47.6
6to18	All	1564	1492.5	56	72.4	1476	1408	37	68.3
	1	643	612	29	89.3	626	601.5	23	86.9
	2	557	524	22	77.4	533	497.5	21	74
	3	391	353	30	54.3	337	316.5	25	46.8
8to14	All	1534	1461	70	71	1475	1398	55	68.3
	1	645	603	28	89.6	628	600.5	12	87.2
	2	541	512	20	75.1	517	495	17	71.8
	3	377	342.5	27	52.4	354	305	31	49.2
8to18	All	1569	1493	61	72.6	**1500**	1419.5	37	69.4
	1	655	614.5	16	91	631	610.5	21	87.6
	2	547	520.5	22	76	521	492.5	26	72.4
	3	390	353.5	26	54.2	352	314.5	17	48.9
10to16	All	1533	1468	54	71	1457	1395.5	53	67.5
	1	631	606	22	87.6	621	598	17	86.2
	2	544	515	28	75.6	519	489.5	18	72.1
	3	374	333.5	23	51.9	346	302	24	48.1
14to18	All	**1570**	**1510.5**	36	72.7	1491	**1426**	58	69
	1	638	618	17	88.6	626	609.5	17	86.9
	2	557	529.5	18	77.4	533	505	25	74
	3	397	362.5	21	55.1	345	316	28	47.9

Table B.17: Code60-2, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1693	1458	125	78.4	1732	1440	102	80.2
	1	620	553	28	86.1	641	559.5	30	89
	2	588	516	31	81.7	591	499	40	82.1
	3	485	396.5	37	67.4	500	386	51	69.4
4to8	All	1830	1621.5	97	84.7	1810	1618	119	83.8
	1	663	607	33	92.1	651	611	37	90.4
	2	635	561.5	33	88.2	615	557	41	85.4
	3	532	454	41	73.9	544	451.5	47	75.6
6to12	All	1846	1707.5	79	85.5	1819	1700	79	84.2
	1	673	637.5	21	93.5	666	635	24	92.5
	2	641	596	22	89	620	582.5	26	86.1
	3	536	481.5	32	74.4	543	473	39	75.4
6to18	All	**1882**	1717.5	77	87.1	**1857**	1677	81	86
	1	683	641.5	28	94.9	666	633	24	92.5
	2	647	596	25	89.9	630	577	21	87.5
	3	552	481.5	44	76.7	561	463	50	77.9
8to14	All	1847	1705.5	69	85.5	1831	1687.5	101	84.8
	1	688	639	14	95.6	690	635	24	95.8
	2	637	588.5	26	88.5	638	582.5	31	88.6
	3	541	481.5	44	75.1	541	461.5	56	75.1
8to18	All	1872	**1737.5**	78	86.7	1846	**1708.5**	65	85.5
	1	675	643.5	22	93.8	672	642	20	93.3
	2	650	596	28	90.3	638	587	26	88.6
	3	547	490	30	76	536	482	38	74.4
10to16	All	1854	1715	65	85.8	1848	1692.5	74	85.6
	1	676	640	16	93.9	674	634	15	93.6
	2	633	591.5	19	87.9	635	582	27	88.2
	3	549	483	30	76.2	541	468.5	51	75.1
14to18	All	1822	1736.5	64	84.4	1794	1707	94	83.1
	1	680	649.5	17	94.4	667	642.5	22	92.6
	2	636	601	26	88.3	617	591	28	85.7
	3	533	487.5	36	74	518	474.5	47	71.9

Table B.18: Code60-2, Fuzzy Classification Fitness Result For Experiment 1

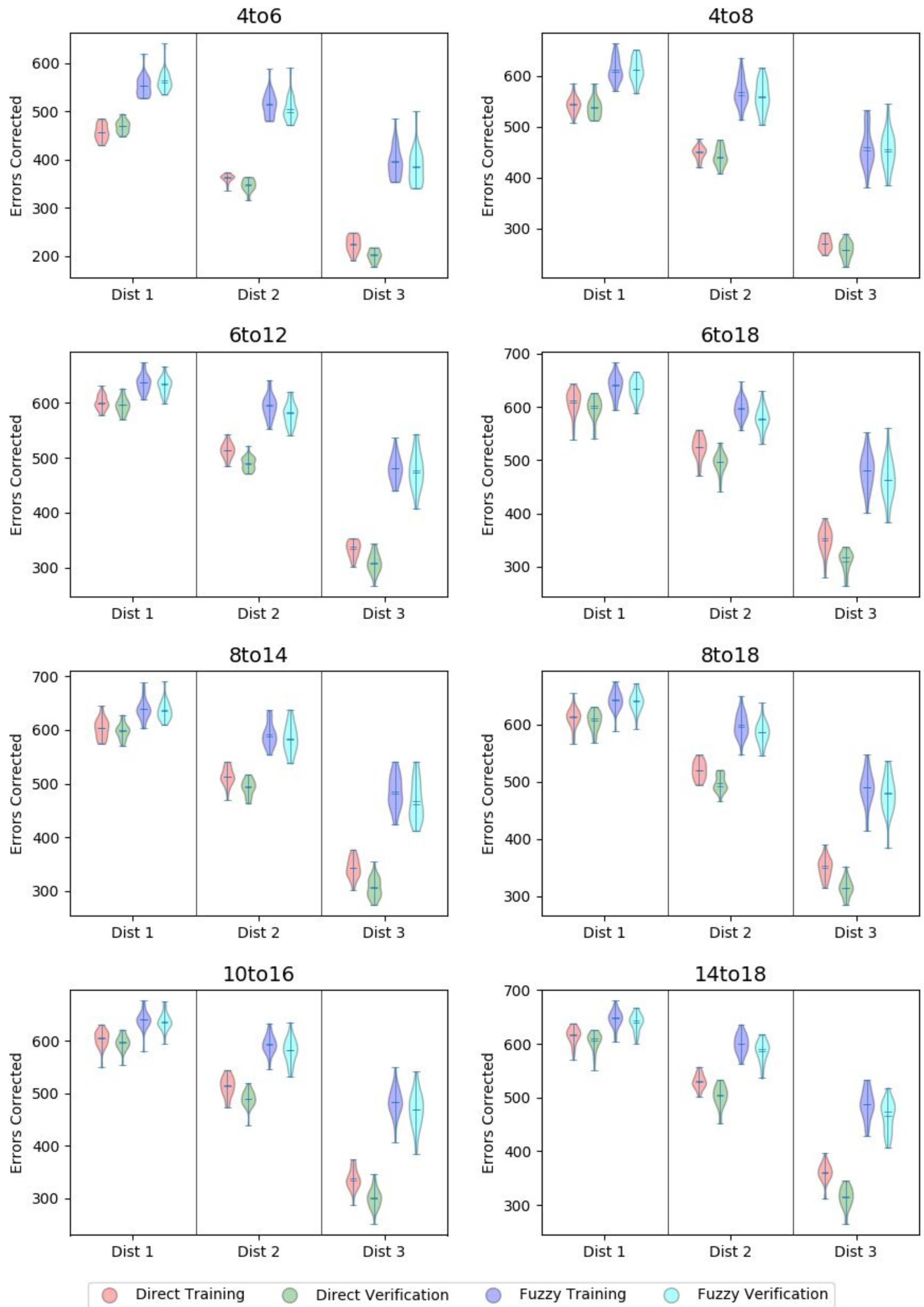

Figure B.9: Code60-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1089	1053	24	50.4	1053	1022.5	31	48.8
	1	488	464	19	67.8	492	473	20	68.3
	2	381	363	7	52.9	371	352	15	51.5
	3	249	219.5	21	34.6	221	203	21	30.7
4to8	All	1327	1274.5	45	61.4	1316	1248	37	60.9
	1	585	547.5	27	81.2	584	542	31	81.1
	2	476	450.5	17	66.1	471	444	17	65.4
	3	292	272.5	15	40.6	282	264	18	39.2
6to12	All	1515	1447.5	83	70.1	1450	1390	59	67.1
	1	623	604.5	14	86.5	621	594.5	16	86.2
	2	541	512	31	75.1	517	489.5	22	71.8
	3	378	336	32	52.5	334	304	34	46.4
6to18	All	**1583**	1493	106	73.3	1495	1428	70	69.2
	1	654	613.5	26	90.8	633	604	24	87.9
	2	557	526	34	77.4	536	501.5	34	74.4
	3	400	352	46	55.6	356	309	20	49.4
8to14	All	1555	1466.5	69	72	1493	1402.5	63	69.1
	1	632	605	22	87.8	617	598	23	85.7
	2	545	516	21	75.7	534	491.5	24	74.2
	3	390	342	32	54.2	347	310.5	23	48.2
8to18	All	**1583**	1469	84	73.3	1517	1410	67	70.2
	1	639	612.5	21	88.8	646	602.5	31	89.7
	2	556	519.5	21	77.2	535	493.5	22	74.3
	3	392	343.5	41	54.4	351	311.5	39	48.8
10to16	All	1537	1465.5	99	71.2	1488	1398.5	72	68.9
	1	630	607	18	87.5	631	605	25	87.6
	2	554	518	36	76.9	522	491.5	31	72.5
	3	373	339	33	51.8	356	305.5	30	49.4
14to18	All	**1583**	**1524.5**	66	73.3	**1526**	**1444.5**	74	70.6
	1	647	626	16	89.9	635	616	22	88.2
	2	568	535	25	78.9	538	506.5	24	74.7
	3	388	360	29	53.9	360	322.5	35	50

Table B.19: Code60-2, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1595	1471.5	60	73.8	1561	1445	65	72.3
	1	594	559	26	82.5	592	556.5	21	82.2
	2	554	523	25	76.9	544	504	23	75.6
	3	447	396	28	62.1	427	385.5	45	59.3
4to8	All	1845	1633	69	85.4	1844	1635	121	85.4
	1	664	610	33	92.2	659	613	36	91.5
	2	637	564.5	22	88.5	642	561.5	43	89.2
	3	544	455.5	50	75.6	543	460	34	75.4
6to12	All	1816	1741	63	84.1	1813	1710.5	69	83.9
	1	661	645	21	91.8	665	640.5	22	92.4
	2	629	600	23	87.4	626	595.5	25	86.9
	3	542	492	32	75.3	536	487	35	74.4
6to18	All	1815	1740	60	84	1798	1718.5	76	83.2
	1	679	645.5	24	94.3	670	642.5	24	93.1
	2	632	602	22	87.8	625	595	24	86.8
	3	527	491	26	73.2	533	472.5	41	74
8to14	All	1833	1721	106	84.9	1803	1700	130	83.5
	1	669	638.5	25	92.9	666	633.5	33	92.5
	2	643	599.5	36	89.3	622	587	41	86.4
	3	530	492.5	35	73.6	528	480.5	41	73.3
8to18	All	1837	1727	64	85	1826	1703.5	89	84.5
	1	670	646.5	19	93.1	675	642	21	93.8
	2	631	598	24	87.6	638	582	28	88.6
	3	539	489.5	26	74.9	537	481	33	74.6
10to16	All	**1872**	1734	72	86.7	**1866**	1708.5	87	86.4
	1	678	646.5	23	94.2	672	646	27	93.3
	2	640	602.5	28	88.9	640	589	21	88.9
	3	563	490	28	78.2	554	475	38	76.9
14to18	All	1832	**1746.5**	76	84.8	1809	**1734.5**	74	83.8
	1	678	654.5	16	94.2	675	651.5	18	93.8
	2	628	605.5	23	87.2	627	595.5	18	87.1
	3	535	488	39	74.3	521	479	32	72.4

Table B.20: Code60-2, Fuzzy Classification Fitness Result For Experiment 2

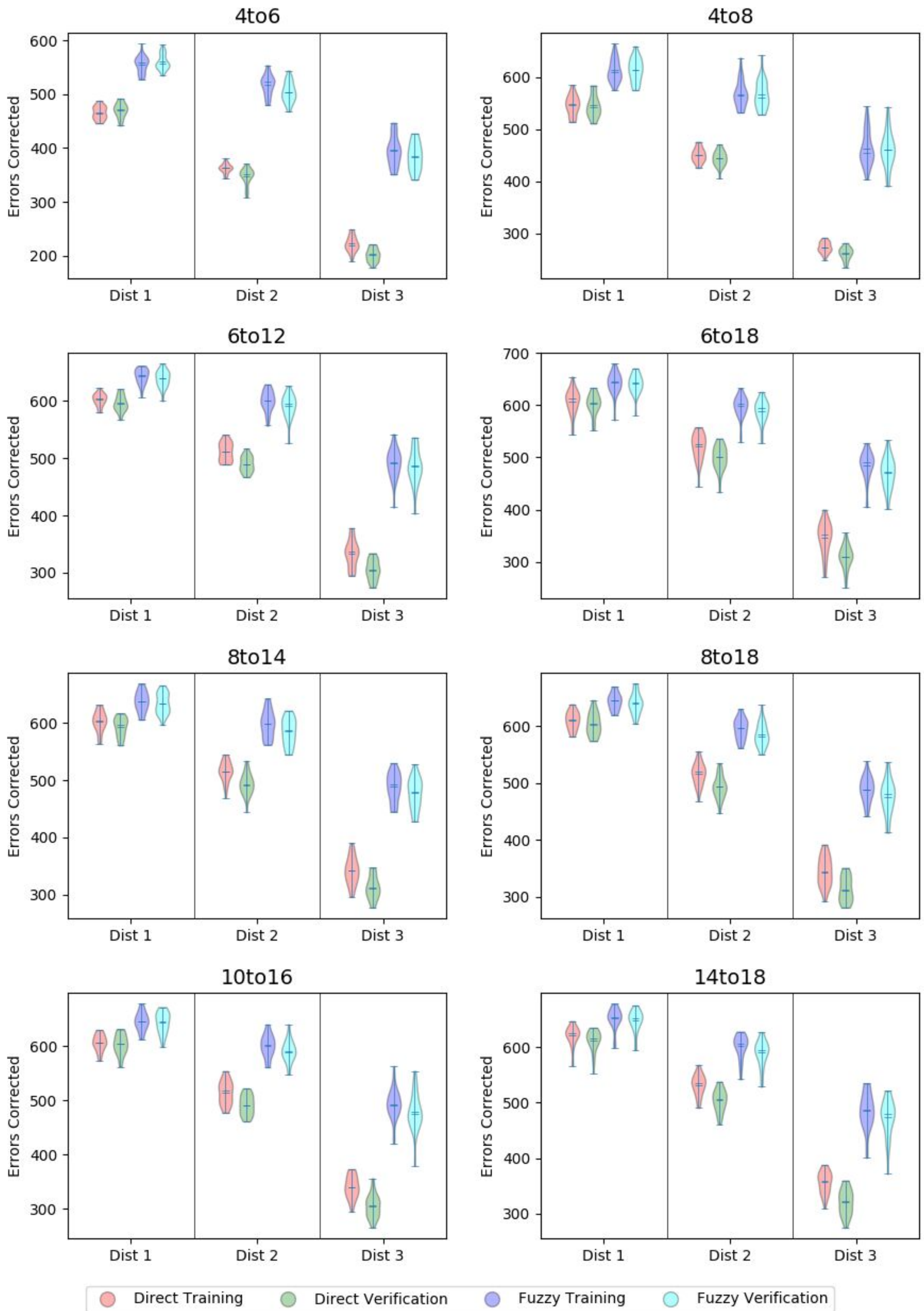

Figure B.10: Code60-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1089	1057.5	14	50.4	1044	1018.5	23	48.3
	1	488	465.5	13	67.8	492	469	10	68.3
	2	382	364	7	53.1	371	353	9	51.5
	3	249	223	14	34.6	217	202	11	30.1
4to8	All	1327	1285.5	33	61.4	1316	1271	40	60.9
	1	585	557.5	20	81.2	584	551	30	81.1
	2	476	456	12	66.1	471	452.5	21	65.4
	3	292	270.5	19	40.6	278	266	19	38.6
6to12	All	1502	1445.5	99	69.5	1477	1375.5	101	68.4
	1	632	600.5	32	87.8	629	594	35	87.4
	2	538	508.5	28	74.7	519	487	29	72.1
	3	370	333	33	51.4	342	295	34	47.5
6to18	All	1551	1447.5	107	71.8	1508	1412	90	69.8
	1	646	602.5	28	89.7	627	599.5	23	87.1
	2	547	512.5	37	76	535	499	33	74.3
	3	375	337.5	41	52.1	346	303.5	29	48.1
8to14	All	1529	1447	100	70.8	1494	1393	48	69.2
	1	641	604	26	89	634	597.5	30	88.1
	2	540	505.5	35	75	521	489.5	30	72.4
	3	372	337	34	51.7	358	305.5	32	49.7
8to18	All	**1590**	1487	78	73.6	1533	1428	67	71
	1	648	618	27	90	641	610.5	25	89
	2	562	519	27	78.1	533	497	31	74
	3	400	345	39	55.6	359	316.5	33	49.9
10to16	All	1549	1475.5	75	71.7	1487	1416	57	68.8
	1	643	610	23	89.3	633	608.5	21	87.9
	2	552	523.5	31	76.7	523	502	20	72.6
	3	388	347	30	53.9	354	315	28	49.2
14to18	All	1566	**1515.5**	59	72.5	**1536**	**1436.5**	70	71.1
	1	642	622	22	89.2	640	612	20	88.9
	2	561	534	26	77.9	540	505	28	75
	3	397	362	26	55.1	376	324	25	52.2

Table B.21: Code60-2, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1595	1479	60	73.8	1561	1452	71	72.3
	1	594	560	12	82.5	592	558	27	82.2
	2	554	519	17	76.9	544	505.5	26	75.6
	3	447	391	30	62.1	431	390.5	28	59.9
4to8	All	1776	1648.5	93	82.2	1751	1647	106	81.1
	1	650	613.5	33	90.3	642	613	33	89.2
	2	608	567	24	84.4	600	567.5	35	83.3
	3	518	459.5	34	71.9	517	459	35	71.8
6to12	All	1814	1731	90	84	1800	1692	95	83.3
	1	665	640.5	27	92.4	664	634	35	92.2
	2	624	599	31	86.7	612	582	29	85
	3	533	494.5	30	74	536	463.5	39	74.4
6to18	All	1833	1731.5	55	84.9	1817	1697.5	56	84.1
	1	673	640	17	93.5	664	636	17	92.2
	2	638	596.5	18	88.6	613	586.5	22	85.1
	3	522	488.5	31	72.5	548	474.5	28	76.1
8to14	All	1807	1716	71	83.7	1828	1683	76	84.6
	1	661	641	21	91.8	659	634	26	91.5
	2	635	594	24	88.2	622	579	31	86.4
	3	526	483.5	38	73.1	551	469	43	76.5
8to18	All	1832	1746.5	82	84.8	1823	1717.5	80	84.4
	1	681	650.5	33	94.6	679	649.5	28	94.3
	2	635	602	26	88.2	625	588	40	86.8
	3	529	497	23	73.5	544	479.5	36	75.6
10to16	All	**1845**	**1760**	90	85.4	**1833**	**1728**	84	84.9
	1	670	651.5	19	93.1	664	644.5	28	92.2
	2	640	605	28	88.9	632	595	26	87.8
	3	546	501	35	75.8	542	497	53	75.3
14to18	All	1833	1727.5	79	84.9	1829	1703	91	84.7
	1	672	650	22	93.3	668	646	27	92.8
	2	634	601.5	31	88.1	620	587.5	28	86.1
	3	532	482.5	32	73.9	546	471.5	59	75.8

Table B.22: Code60-2, Fuzzy Classification Fitness Result For Experiment 3

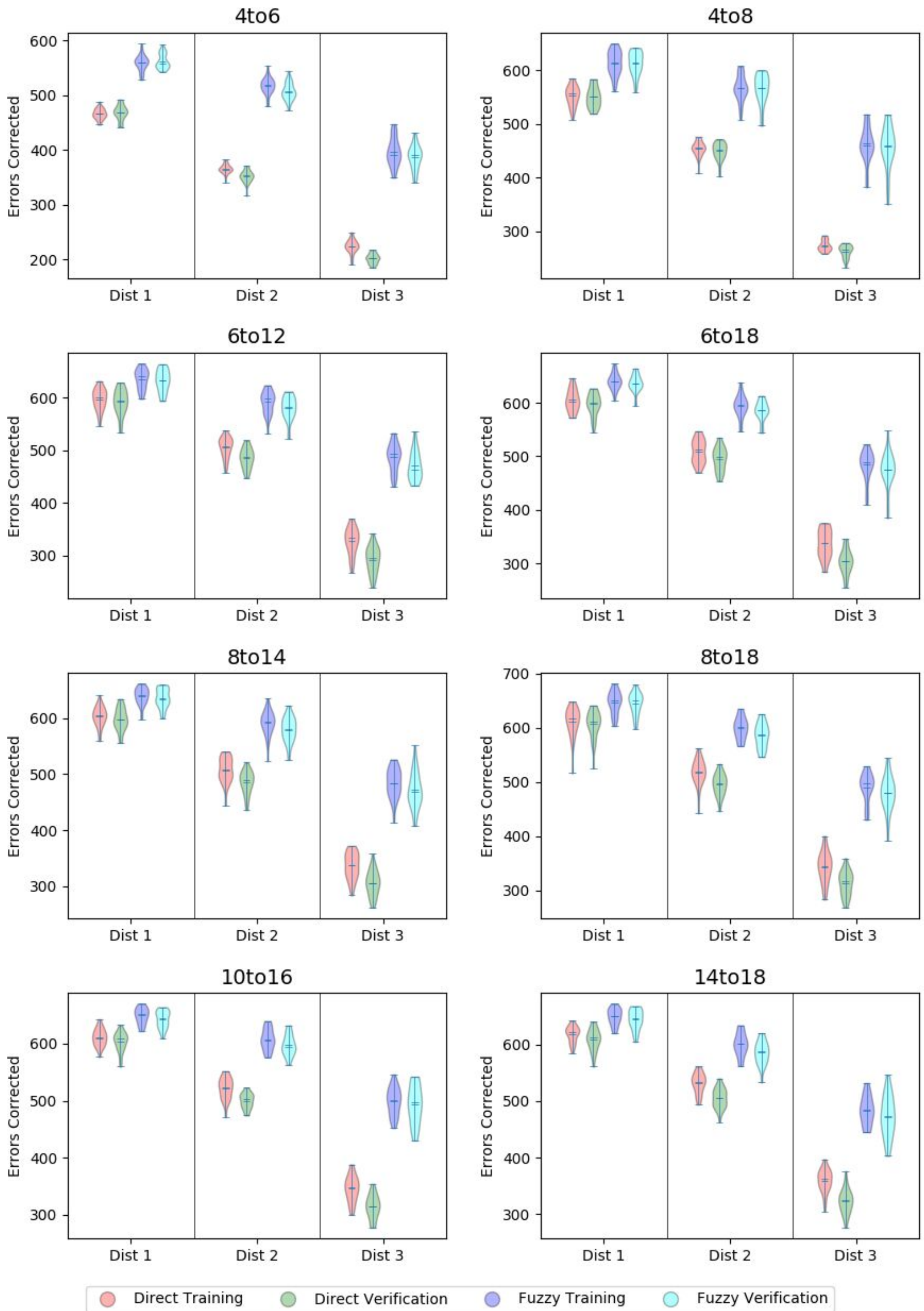

Figure B.11: Code60-3, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1069	1053.5	24	49.5	1053	1022.5	29	48.8
	1	478	464.5	19	66.4	492	469	18	68.3
	2	377	363	11	52.4	367	354	12	51
	3	245	225	21	34	221	199	14	30.7
4to8	All	1327	1283.5	34	61.4	1316	1264.5	61	60.9
	1	585	548.5	22	81.2	584	545.5	19	81.1
	2	476	456.5	12	66.1	471	452.5	25	65.4
	3	292	271.5	16	40.6	283	260	17	39.3
6to12	All	1488	1429	64	68.9	1432	1379	46	66.3
	1	620	596.5	25	86.1	614	594	11	85.3
	2	529	504.5	23	73.5	523	485.5	30	72.6
	3	353	327.5	37	49	315	298	16	43.8
6to18	All	**1586**	1494	73	73.4	**1541**	1422	71	71.3
	1	644	617	30	89.4	650	609.5	29	90.3
	2	558	523.5	22	77.5	541	500.5	28	75.1
	3	391	350	26	54.3	364	315	21	50.6
8to14	All	1515	1439.5	80	70.1	1453	1387.5	69	67.3
	1	631	604.5	22	87.6	633	600.5	15	87.9
	2	536	507	28	74.4	516	488	19	71.7
	3	371	326	35	51.5	334	298	25	46.4
8to18	All	1580	1476	66	73.1	1527	1415	78	70.7
	1	635	608	18	88.2	636	603.5	25	88.3
	2	559	518	19	77.6	541	494	28	75.1
	3	398	350	31	55.3	358	311.5	40	49.7
10to16	All	1563	1481	74	72.4	1481	1405	77	68.6
	1	645	609.5	20	89.6	632	605.5	28	87.8
	2	551	521	25	76.5	529	501	29	73.5
	3	380	342.5	31	52.8	348	307	31	48.3
14to18	All	1580	**1515.5**	47	73.1	1509	**1440.5**	74	69.9
	1	634	622	16	88.1	645	610.5	24	89.6
	2	549	535	18	76.2	529	508.5	22	73.5
	3	401	361	26	55.7	356	327	31	49.4

Table B.23: Code60-2, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	1643	1511.5	90	76.1	1660	1501.5	105	76.9
	1	605	566	18	84	629	578	23	87.4
	2	562	525	29	78.1	563	525.5	35	78.2
	3	476	409.5	43	66.1	468	395	51	65
4to8	All	1776	1651	93	82.2	1782	1652.5	107	82.5
	1	650	616	33	90.3	648	615	32	90
	2	613	570	28	85.1	604	571	35	83.9
	3	518	463	37	71.9	530	465.5	37	73.6
6to12	All	1812	1723.5	34	83.9	1820	1708	43	84.3
	1	667	640	24	92.6	666	637	14	92.5
	2	635	593.5	16	88.2	618	587	21	85.8
	3	519	493	19	72.1	546	483.5	27	75.8
6to18	All	**1891**	1735	123	87.5	**1879**	1714.5	113	87
	1	685	653.5	29	95.1	680	652.5	29	94.4
	2	651	603	38	90.4	638	592.5	31	88.6
	3	562	487	54	78.1	561	477	65	77.9
8to14	All	1834	1730.5	58	84.9	1816	1716	65	84.1
	1	672	641	27	93.3	676	642	19	93.9
	2	633	597.5	24	87.9	626	588	23	86.9
	3	549	492	31	76.2	545	481	33	75.7
8to18	All	1830	1725.5	67	84.7	1820	1706.5	68	84.3
	1	668	642.5	22	92.8	662	643.5	23	91.9
	2	633	596	23	87.9	629	584	32	87.4
	3	529	482	29	73.5	538	473.5	36	74.7
10to16	All	1831	1737	73	84.8	1842	1726.5	67	85.3
	1	674	648	23	93.6	674	642.5	19	93.6
	2	628	601	20	87.2	632	593	23	87.8
	3	544	492	33	75.6	549	486.5	37	76.2
14to18	All	1803	**1748**	45	83.5	1774	**1729.5**	51	82.1
	1	679	655	14	94.3	668	648.5	17	92.8
	2	625	606	19	86.8	615	591	25	85.4
	3	526	491	29	73.1	512	476	41	71.1

Table B.24: Code60-2, Fuzzy Classification Fitness Result For Experiment 4

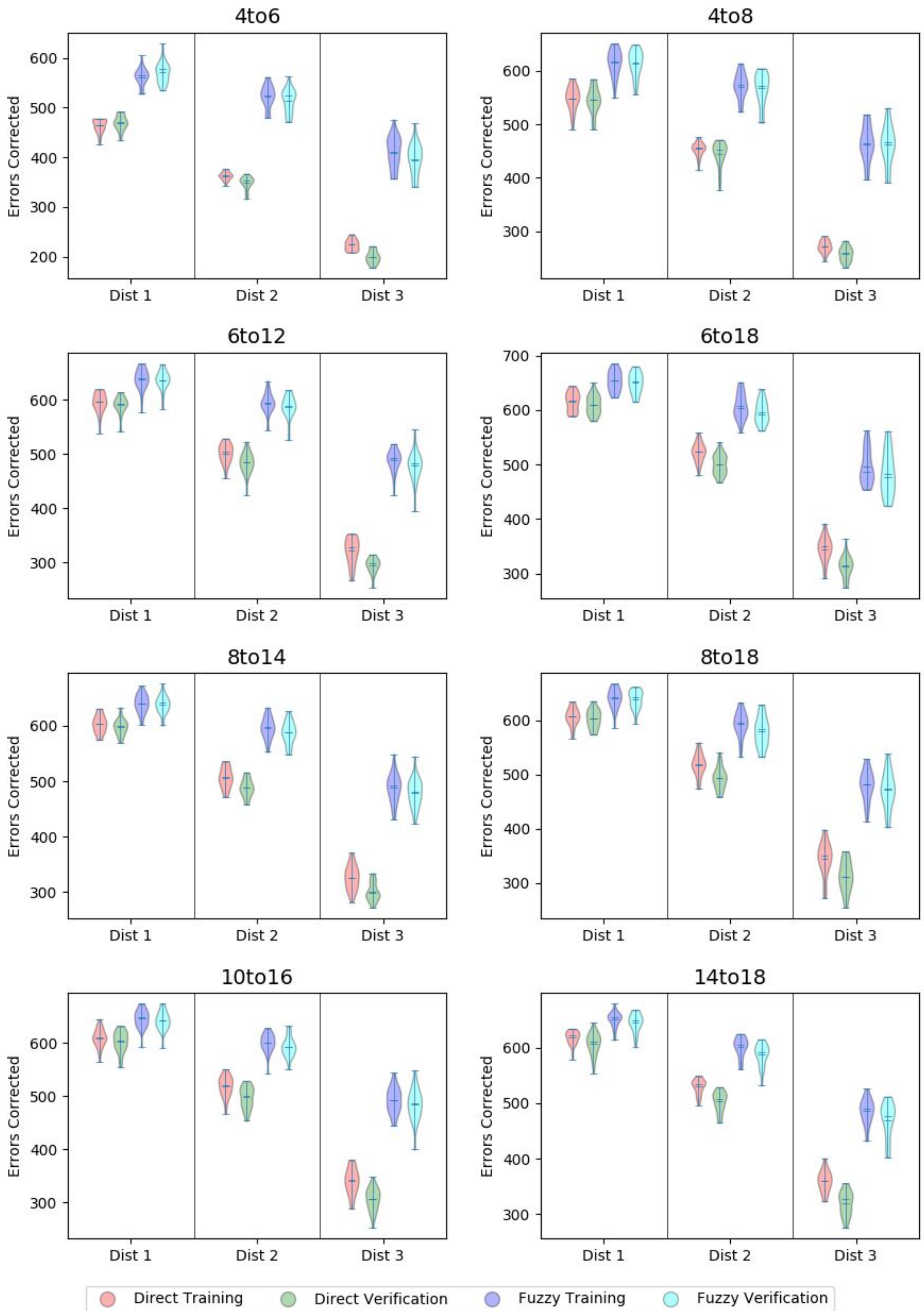

Figure B.12: Code60-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

B.1.2 Codes of Length 10

Code17-1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	349	344	7	68.4	333	324	24	65.3
	1	146	139	3	85.9	147	140.5	4	86.5
	2	135	126	3	79.4	122	114	9	71.8
	3	85	75.5	6	50	77	70	11	45.3
4to8	All	389	373.5	14	76.3	366	340	21	71.8
	1	162	151.5	7	95.3	157	147	11	92.4
	2	144	134.5	7	84.7	134	117.5	12	78.8
	3	98	87	7	57.6	89	73	14	52.4
6to12	All	409	**394.5**	18	80.2	367	**345**	22	72
	1	162	156.5	7	95.3	157	149	6	92.4
	2	154	140.5	9	90.6	133	123.5	7	78.2
	3	104	96	4	61.2	91	75	15	53.5
6to18	All	419	390	16	82.2	362	340.5	20	71
	1	167	155.5	9	98.2	156	149	9	91.8
	2	154	141.5	8	90.6	131	119.5	9	77.1
	3	112	94.5	9	65.9	87	72.5	10	51.2
8to14	All	419	385.5	26	82.2	360	330.5	24	70.6
	1	164	153	9	96.5	157	145	10	92.4
	2	154	138	13	90.6	131	114.5	12	77.1
	3	110	95.5	14	64.7	86	73	9	50.6
8to18	All	415	390.5	15	81.4	367	336	29	72
	1	165	155.5	8	97.1	157	145	11	92.4
	2	148	141.5	9	87.1	133	117	12	78.2
	3	108	96.5	8	63.5	82	72.5	11	48.2
10to16	All	411	392	25	80.6	370	339	15	72.5
	1	163	156	6	95.9	155	147	5	91.2
	2	151	142.5	6	88.8	133	118.5	10	78.2
	3	112	95	14	65.9	91	72	7	53.5
14to18	All	**421**	392.5	17	82.5	**391**	329	25	76.7
	1	164	155	7	96.5	161	143	11	94.7
	2	148	140	9	87.1	134	117.5	12	78.8
	3	114	98	8	67.1	96	71.5	11	56.5

Table B.25: Code17-1, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	424	380	31	83.1	420	385	25	82.4
	1	158	147.5	10	92.9	161	146.5	4	94.7
	2	152	139	13	89.4	143	136	10	84.1
	3	119	95	10	70	123	103	11	72.4
4to8	All	441	**407**	31	86.5	433	**391**	32	84.9
	1	163	155.5	8	95.9	166	153	7	97.6
	2	159	150	11	93.5	152	134	10	89.4
	3	120	102.5	11	70.6	123	101.5	15	72.4
6to12	All	**456**	406	29	89.4	430	379	50	84.3
	1	166	158	9	97.6	167	153	10	98.2
	2	163	148.5	12	95.9	151	133.5	17	88.8
	3	131	102.5	14	77.1	126	94	18	74.1
6to18	All	455	405	18	89.2	**438**	378	33	85.9
	1	167	158	5	98.2	165	153.5	11	97.1
	2	158	149	9	92.9	146	132.5	14	85.9
	3	137	99	13	80.6	127	95	11	74.7
8to14	All	430	405.5	31	84.3	414	375.5	34	81.2
	1	165	157	11	97.1	161	152	8	94.7
	2	156	147	12	91.8	150	130	18	88.2
	3	115	100.5	14	67.6	115	96.5	12	67.6
8to18	All	427	401	29	83.7	415	374	33	81.4
	1	163	154.5	8	95.9	163	149.5	10	95.9
	2	156	145.5	10	91.8	147	129.5	15	86.5
	3	119	97.5	14	70	113	92.5	18	66.5
10to16	All	448	**407**	34	87.8	419	383.5	31	82.2
	1	167	157	7	98.2	164	153	8	96.5
	2	158	145.5	14	92.9	147	132	14	86.5
	3	125	101	16	73.5	109	97	16	64.1
14to18	All	416	387	15	81.6	417	357	21	81.8
	1	165	154	6	97.1	162	150	9	95.3
	2	154	140.5	8	90.6	146	125.5	7	85.9
	3	115	96	9	67.6	109	85.5	15	64.1

Table B.26: Code17-1, Fuzzy Classification Fitness Result For Experiment 1

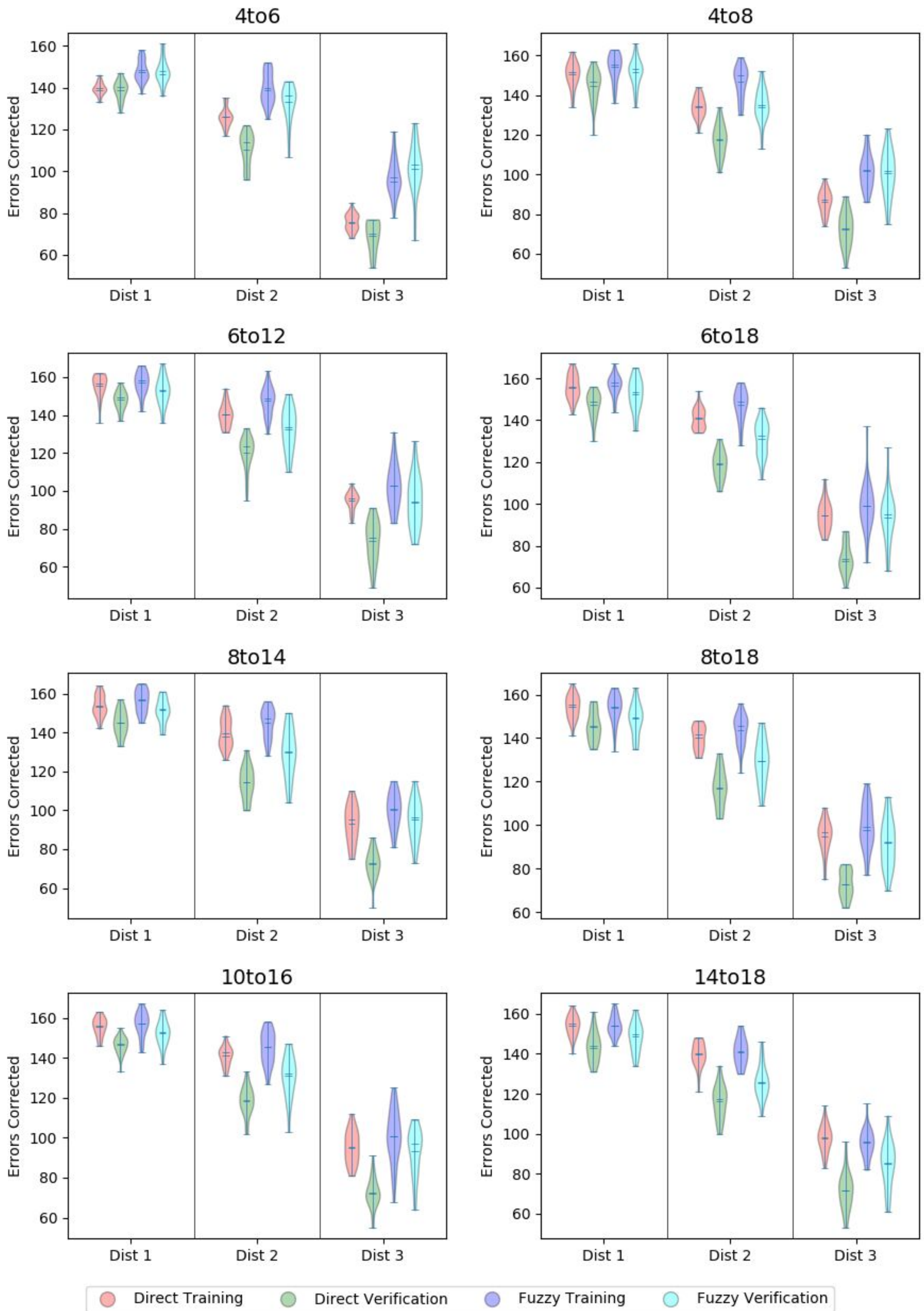

Figure B.13: Code17-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	350	343	9	68.6	333	316.5	20	65.3
	1	150	140	8	88.2	147	140	13	86.5
	2	135	126	5	79.4	123	113	8	72.4
	3	85	76.5	6	50	77	67	7	45.3
4to8	All	389	371.5	11	76.3	353	330.5	20	69.2
	1	159	151	8	93.5	152	143	7	89.4
	2	144	135	5	84.7	132	118.5	12	77.6
	3	94	86.5	6	55.3	83	72	11	48.8
6to12	All	417	375.5	21	81.8	**385**	337.5	30	75.5
	1	162	151	10	95.3	158	145.5	7	92.9
	2	150	137	7	88.2	134	116	11	78.8
	3	107	92	10	62.9	94	75	16	55.3
6to18	All	**422**	387	27	82.7	357	335	24	70
	1	165	156	9	97.1	161	145.5	8	94.7
	2	151	141.5	7	88.8	136	117	6	80
	3	115	92	13	67.6	86	72	9	50.6
8to14	All	418	393	23	82	366	**345**	14	71.8
	1	162	157	6	95.3	155	147	6	91.2
	2	154	145.5	13	90.6	132	122	10	77.6
	3	110	94	10	64.7	87	76	10	51.2
8to18	All	409	390.5	27	80.2	364	341	32	71.4
	1	164	157	8	96.5	158	149.5	11	92.9
	2	150	141	9	88.2	130	119	11	76.5
	3	104	92.5	10	61.2	84	69	12	49.4
10to16	All	413	395.5	18	81	369	344	15	72.4
	1	166	158.5	7	97.6	157	148	6	92.4
	2	150	139	7	88.2	130	121	12	76.5
	3	109	96.5	11	64.1	94	72	13	55.3
14to18	All	417	**397.5**	16	81.8	373	338	23	73.1
	1	163	156.5	5	95.9	158	148	8	92.9
	2	151	142.5	6	88.8	133	119	8	78.2
	3	110	98.5	9	64.7	88	73	9	51.8

Table B.27: Code17-1, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	428	388	36	83.9	421	377	34	82.5
	1	159	148	11	93.5	161	147	13	94.7
	2	155	143.5	13	91.2	147	131.5	8	86.5
	3	116	97	13	68.2	123	97.5	18	72.4
4to8	All	436	405	34	85.5	431	**389.5**	30	84.5
	1	166	156	8	97.6	165	152	9	97.1
	2	159	147.5	12	93.5	155	134.5	14	91.2
	3	121	103	15	71.2	117	99	14	68.8
6to12	All	436	398.5	42	85.5	423	379.5	42	82.9
	1	166	154	12	97.6	165	151	8	97.1
	2	155	145.5	11	91.2	150	130.5	14	88.2
	3	120	101.5	18	70.6	120	94.5	19	70.6
6to18	All	**453**	396	24	88.8	**455**	372.5	35	89.2
	1	165	155.5	6	97.1	165	151	7	97.1
	2	161	144	7	94.7	156	130	12	91.8
	3	127	98.5	16	74.7	134	92.5	16	78.8
8to14	All	439	407	26	86.1	428	384.5	31	83.9
	1	166	157	6	97.6	162	152	6	95.3
	2	160	148	9	94.1	155	134	9	91.2
	3	119	101.5	12	70	117	95.5	15	68.8
8to18	All	442	400	36	86.7	426	378	32	83.5
	1	168	155	12	98.8	166	152	12	97.6
	2	163	145.5	9	95.9	151	133	14	88.8
	3	122	99	15	71.8	115	93	18	67.6
10to16	All	448	**411.5**	34	87.8	432	381	35	84.7
	1	168	161	7	98.8	165	152.5	11	97.1
	2	161	148	10	94.7	152	133.5	10	89.4
	3	127	103	18	74.7	121	94.5	23	71.2
14to18	All	427	392.5	33	83.7	408	362	38	80
	1	163	156	9	95.9	158	151	11	92.9
	2	155	142.5	11	91.2	142	126.5	15	83.5
	3	114	93.5	21	67.1	110	81.5	20	64.7

Table B.28: Code17-1, Fuzzy Classification Fitness Result For Experiment 2

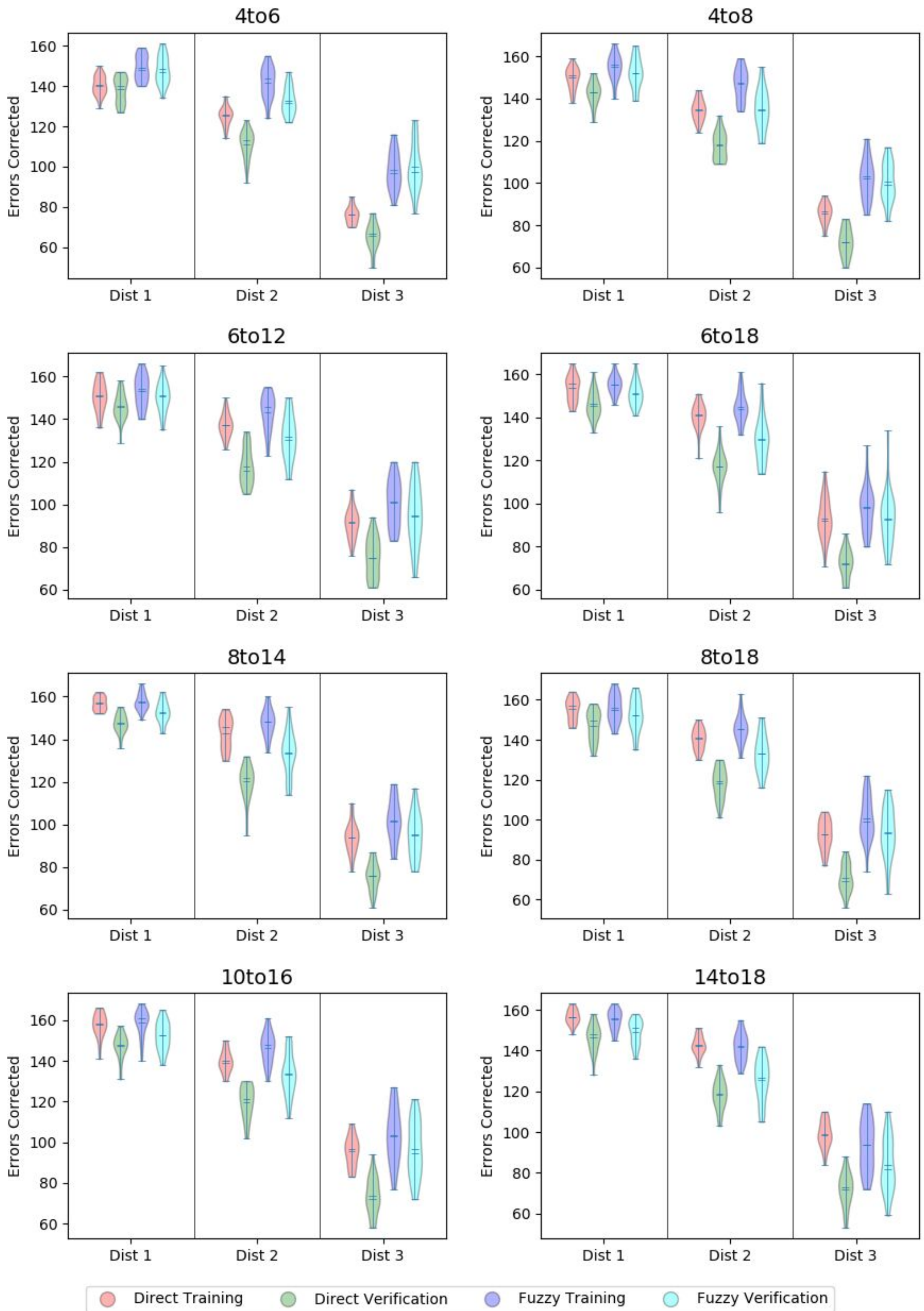

Figure B.14: Code17-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	348	341	9	68.2	345	322	22	67.6
	1	150	142	5	88.2	149	139.5	10	87.6
	2	135	124.5	5	79.4	123	114	10	72.4
	3	84	74.5	8	49.4	81	67.5	8	47.6
4to8	All	380	373.5	16	74.5	363	339	22	71.2
	1	159	151.5	8	93.5	154	144.5	7	90.6
	2	140	133.5	4	82.4	132	118	11	77.6
	3	97	86	9	57.1	85	74	14	50
6to12	All	409	383	27	80.2	**372**	341.5	21	72.9
	1	163	154	11	95.9	159	145.5	8	93.5
	2	149	139	8	87.6	131	120	10	77.1
	3	105	92.5	11	61.8	93	77	9	54.7
6to18	All	**426**	390	20	83.5	353	338.5	22	69.2
	1	165	157	7	97.1	159	147	7	93.5
	2	149	139.5	7	87.6	129	118	7	75.9
	3	112	94	10	65.9	86	71.5	6	50.6
8to14	All	416	387	18	81.6	**372**	343	23	72.9
	1	166	154.5	6	97.6	160	150	6	94.1
	2	149	139	6	87.6	133	119.5	9	78.2
	3	112	94	13	65.9	92	72.5	8	54.1
8to18	All	419	393.5	25	82.2	369	336.5	22	72.4
	1	165	157	5	97.1	156	148.5	8	91.8
	2	151	142	6	88.8	137	118.5	13	80.6
	3	112	90	16	65.9	86	72.5	9	50.6
10to16	All	420	393.5	30	82.4	368	342	20	72.2
	1	164	156	6	96.5	157	149	7	92.4
	2	153	143.5	9	90	131	121	7	77.1
	3	109	94.5	11	64.1	87	74	11	51.2
14to18	All	422	**405.5**	13	82.7	369	**345**	16	72.4
	1	165	160	6	97.1	158	149.5	5	92.9
	2	153	145	7	90	131	121	9	77.1
	3	111	99	10	65.3	89	75.5	9	52.4

Table B.29: Code17-1, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	440	393	39	86.3	440	384	30	86.3
	1	161	149.5	11	94.7	162	150.5	8	95.3
	2	159	142	13	93.5	152	131	11	89.4
	3	121	94	13	71.2	126	98	18	74.1
4to8	All	**453**	407	25	88.8	442	395	26	86.7
	1	166	155.5	6	97.6	165	152.5	4	97.1
	2	163	147	9	95.9	157	137.5	16	92.4
	3	124	101	14	72.9	123	104	18	72.4
6to12	All	449	407.5	31	88	**445**	388.5	42	87.3
	1	166	153	10	97.6	164	151	11	96.5
	2	162	147.5	14	95.3	153	136	15	90
	3	123	100.5	15	72.4	129	99	15	75.9
6to18	All	443	401.5	25	86.9	427	374	33	83.7
	1	167	158.5	6	98.2	167	153	9	98.2
	2	162	146	11	95.3	147	133.5	16	86.5
	3	122	99.5	11	71.8	117	90.5	15	68.8
8to14	All	436	**412.5**	27	85.5	434	**396.5**	33	85.1
	1	166	156.5	8	97.6	167	155	7	98.2
	2	158	148.5	11	92.9	154	138	12	90.6
	3	120	106	11	70.6	116	102	17	68.2
8to18	All	438	402.5	24	85.9	423	376.5	43	82.9
	1	165	157	7	97.1	164	151	10	96.5
	2	154	144.5	6	90.6	145	131.5	13	85.3
	3	120	98.5	16	70.6	122	92	18	71.8
10to16	All	430	401.5	24	84.3	411	371	26	80.6
	1	164	158	9	96.5	160	152.5	9	94.1
	2	156	146	10	91.8	148	129	12	87.1
	3	116	99	13	68.2	115	91	15	67.6
14to18	All	432	393.5	20	84.7	406	365	30	79.6
	1	166	156	6	97.6	162	151.5	8	95.3
	2	156	142.5	9	91.8	143	125.5	15	84.1
	3	114	95	11	67.1	105	82.5	18	61.8

Table B.30: Code17-1, Fuzzy Classification Fitness Result For Experiment 3

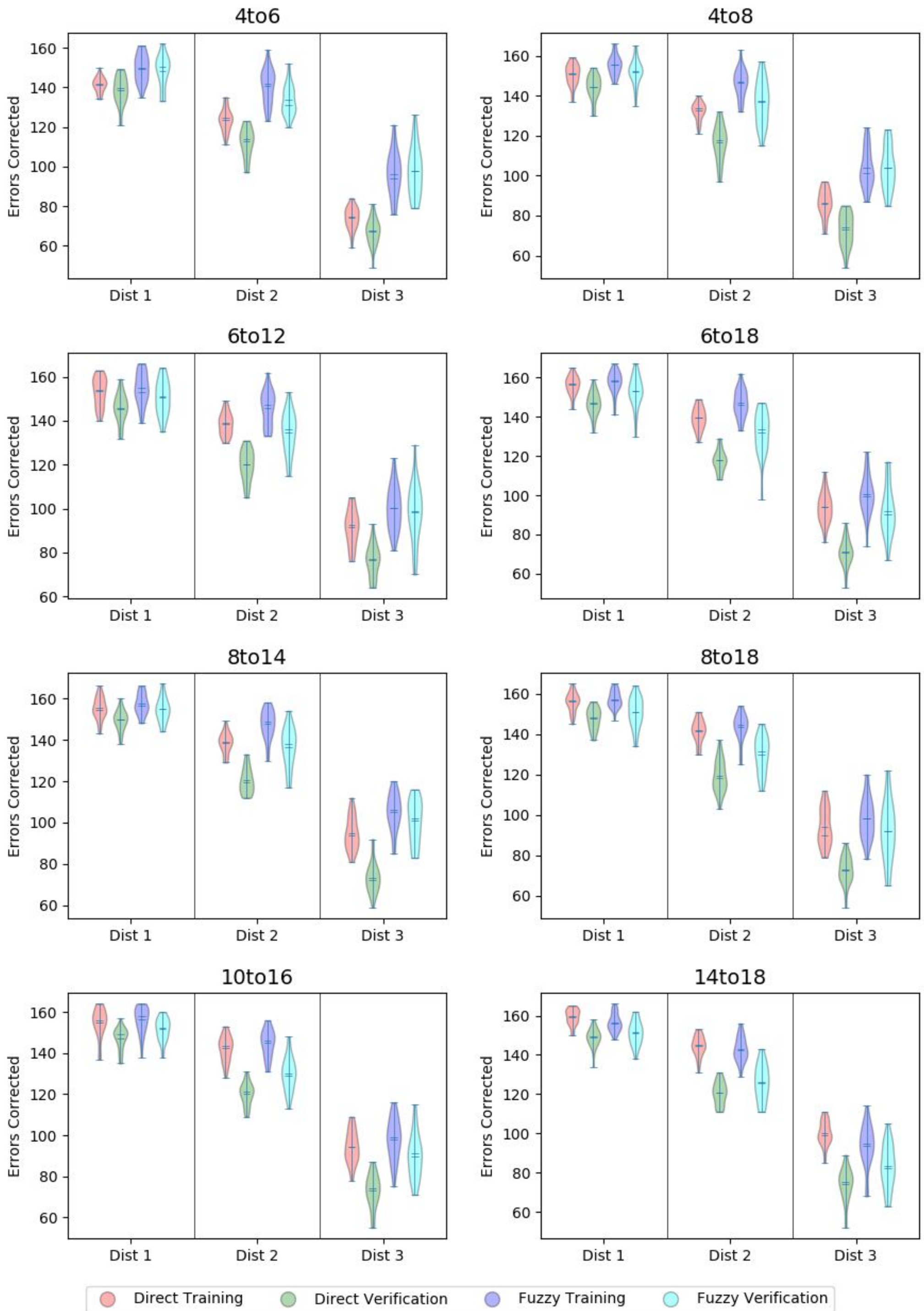

Figure B.15: Code17-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	349	344.5	6	68.4	333	322	18	65.3
	1	150	141.5	8	88.2	149	140	13	87.6
	2	141	128	9	82.9	121	113	5	71.2
	3	85	75	7	50	78	68	6	45.9
4to8	All	389	367.5	15	76.3	362	327	32	71
	1	162	148	7	95.3	155	144	9	91.2
	2	144	133	10	84.7	135	115	12	79.4
	3	94	86	9	55.3	86	69	11	50.6
6to12	All	416	381	32	81.6	363	343.5	27	71.2
	1	162	152.5	6	95.3	155	148	15	91.2
	2	155	136	9	91.2	130	118	12	76.5
	3	107	92	15	62.9	89	71.5	15	52.4
6to18	All	413	385.5	22	81	365	343.5	29	71.6
	1	164	157	9	96.5	154	147	11	90.6
	2	150	138	12	88.2	134	120.5	11	78.8
	3	108	92	14	63.5	90	74.5	13	52.9
8to14	All	416	388	24	81.6	360	**345**	29	70.6
	1	164	155.5	6	96.5	160	147	12	94.1
	2	150	140.5	13	88.2	128	119	10	75.3
	3	104	91.5	7	61.2	90	73.5	9	52.9
8to18	All	424	388.5	21	83.1	361	342.5	24	70.8
	1	165	158	7	97.1	158	146.5	10	92.9
	2	151	139	10	88.8	131	120	11	77.1
	3	115	91.5	11	67.6	87	70	11	51.2
10to16	All	422	390	13	82.7	368	339.5	25	72.2
	1	164	157	5	96.5	156	148	6	91.8
	2	150	141	8	88.2	130	121.5	13	76.5
	3	112	95	11	65.9	87	71.5	11	51.2
14to18	All	**425**	**407.5**	19	83.3	**384**	339.5	16	75.3
	1	165	159	9	97.1	158	147	10	92.9
	2	154	144.5	10	90.6	142	117	6	83.5
	3	115	101.5	10	67.6	91	74.5	10	53.5

Table B.31: Code17-1, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	420	393	41	82.4	420	384	30	82.4
	1	161	153	13	94.7	161	149.5	15	94.7
	2	154	144.5	12	90.6	147	132	6	86.5
	3	115	94	15	67.6	123	98	15	72.4
4to8	All	450	406	37	88.2	440	388.5	38	86.3
	1	167	156.5	14	98.2	164	153	13	96.5
	2	159	148	11	93.5	153	136.5	14	90
	3	124	103.5	14	72.9	123	100	18	72.4
6to12	All	444	411	38	87.1	430	**394.5**	36	84.3
	1	165	155.5	8	97.1	167	153.5	12	98.2
	2	159	146.5	15	93.5	152	138.5	19	89.4
	3	129	103	22	75.9	119	98	20	70
6to18	All	429	408.5	25	84.1	417	382.5	30	81.8
	1	164	158.5	7	96.5	162	152	7	95.3
	2	152	143	6	89.4	153	134	12	90
	3	121	104	15	71.2	116	96	10	68.2
8to14	All	436	**415.5**	33	85.5	428	388	39	83.9
	1	165	158.5	8	97.1	163	154.5	8	95.9
	2	158	147	11	92.9	151	135.5	12	88.8
	3	120	105	13	70.6	118	99.5	21	69.4
8to18	All	**475**	400.5	38	93.1	**444**	370	43	87.1
	1	165	158.5	9	97.1	164	152.5	13	96.5
	2	164	144.5	15	96.5	160	131	14	94.1
	3	146	99	19	85.9	120	90	23	70.6
10to16	All	451	403.5	30	88.4	419	379	34	82.2
	1	168	159.5	6	98.8	164	153	8	96.5
	2	161	143.5	11	94.7	152	133	12	89.4
	3	127	98	12	74.7	106	93	14	62.4
14to18	All	453	395	36	88.8	435	359.5	39	85.3
	1	167	156.5	10	98.2	165	148.5	10	97.1
	2	159	141.5	14	93.5	153	124	15	90
	3	127	97	19	74.7	122	86	17	71.8

Table B.32: Code17-1, Fuzzy Classification Fitness Result For Experiment 4

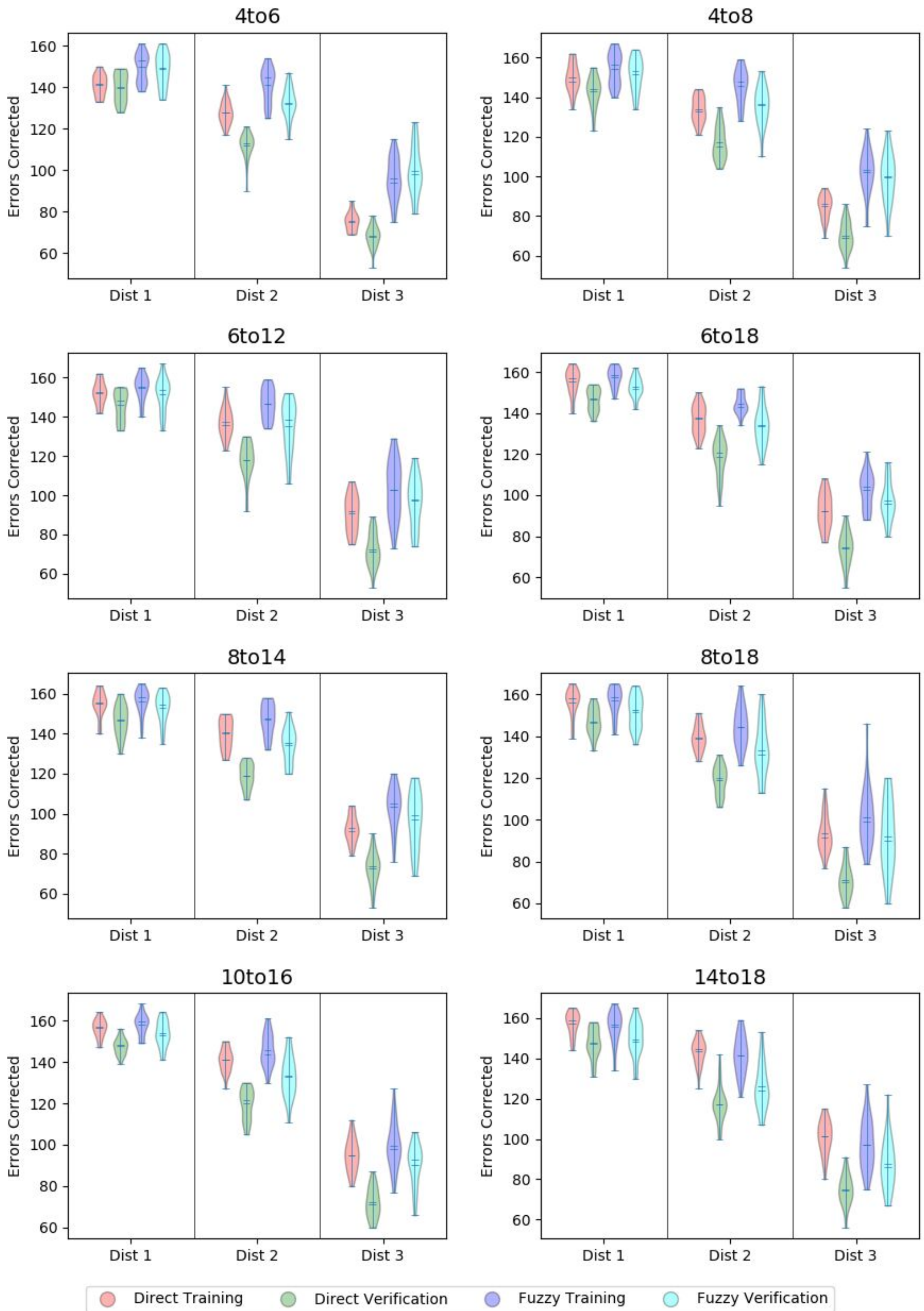

Figure B.16: Code17-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code17-2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	353	340	18	69.2	347	319.5	19	68
	1	150	139.5	8	88.2	151	132.5	10	88.8
	2	129	119.5	11	75.9	132	118	7	77.6
	3	91	82	8	53.5	88	66.5	11	51.8
4to8	All	387	371	21	75.9	369	340.5	19	72.4
	1	155	148	7	91.2	149	142.5	9	87.6
	2	141	130.5	9	82.9	132	124	9	77.6
	3	106	94	10	62.4	90	76.5	11	52.9
6to12	All	409	**388**	17	80.2	367	**342**	25	72
	1	158	151	8	92.9	152	142	10	89.4
	2	145	134	10	85.3	134	122.5	10	78.8
	3	114	102.5	11	67.1	85	74	10	50
6to18	All	414	381.5	24	81.2	360	335.5	20	70.6
	1	162	148.5	10	95.3	155	141	12	91.2
	2	145	133	11	85.3	133	122	6	78.2
	3	114	102	8	67.1	88	72	8	51.8
8to14	All	415	386	25	81.4	360	337	20	70.6
	1	166	150	10	97.6	157	140.5	12	92.4
	2	151	135	9	88.8	131	123.5	5	77.1
	3	112	100.5	10	65.9	83	73.5	9	48.8
8to18	All	**424**	387.5	20	83.1	370	**342**	26	72.5
	1	165	151.5	6	97.1	157	143.5	9	92.4
	2	150	134.5	11	88.2	134	124	14	78.8
	3	119	101.5	10	70	95	73	7	55.9
10to16	All	417	**388**	19	81.8	**377**	341	27	73.9
	1	162	151.5	10	95.3	156	143.5	12	91.8
	2	152	133.5	12	89.4	140	125.5	11	82.4
	3	114	105	12	67.1	90	76	10	52.9
14to18	All	419	387.5	27	82.2	368	332	26	72.2
	1	165	151	6	97.1	162	140.5	16	95.3
	2	147	137	8	86.5	133	120	6	78.2
	3	121	99	15	71.2	89	71.5	11	52.4

Table B.33: Code17-2, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	**458**	389	35	89.8	**450**	383	32	88.2
	1	160	147	12	94.1	163	143	10	95.9
	2	163	138.5	15	95.9	160	140	12	94.1
	3	135	107.5	14	79.4	127	102	12	74.7
4to8	All	435	400.5	37	85.3	421	387	29	82.5
	1	159	150.5	6	93.5	155	147	10	91.2
	2	156	142.5	13	91.8	149	138.5	10	87.6
	3	130	107.5	17	76.5	124	101	12	72.9
6to12	All	446	413	35	87.5	428	**389**	42	83.9
	1	163	152.5	7	95.9	158	148.5	11	92.9
	2	154	144	13	90.6	154	139.5	14	90.6
	3	136	115.5	16	80	121	98	20	71.2
6to18	All	443	399	26	86.9	418	371.5	40	82
	1	159	150.5	8	93.5	158	145.5	10	92.9
	2	157	141	12	92.4	149	134	11	87.6
	3	128	107.5	12	75.3	116	92.5	17	68.2
8to14	All	446	401	35	87.5	427	378	28	83.7
	1	169	152	10	99.4	164	147.5	15	96.5
	2	158	141	12	92.9	151	136.5	9	88.8
	3	127	109	13	74.7	115	97.5	12	67.6
8to18	All	456	**414**	27	89.4	429	383.5	36	84.1
	1	166	154	8	97.6	165	149.5	12	97.1
	2	159	145	13	93.5	152	138	12	89.4
	3	131	114.5	15	77.1	117	98	13	68.8
10to16	All	450	406	35	88.2	426	386.5	35	83.5
	1	163	153	13	95.9	161	149.5	14	94.7
	2	158	144.5	14	92.9	152	139	15	89.4
	3	129	110.5	14	75.9	120	97.5	20	70.6
14to18	All	451	393.5	37	88.4	417	365	31	81.8
	1	168	150.5	10	98.8	165	144	11	97.1
	2	155	138.5	9	91.2	146	134	9	85.9
	3	133	105	16	78.2	115	85.5	15	67.6

Table B.34: Code17-2, Fuzzy Classification Fitness Result For Experiment 1

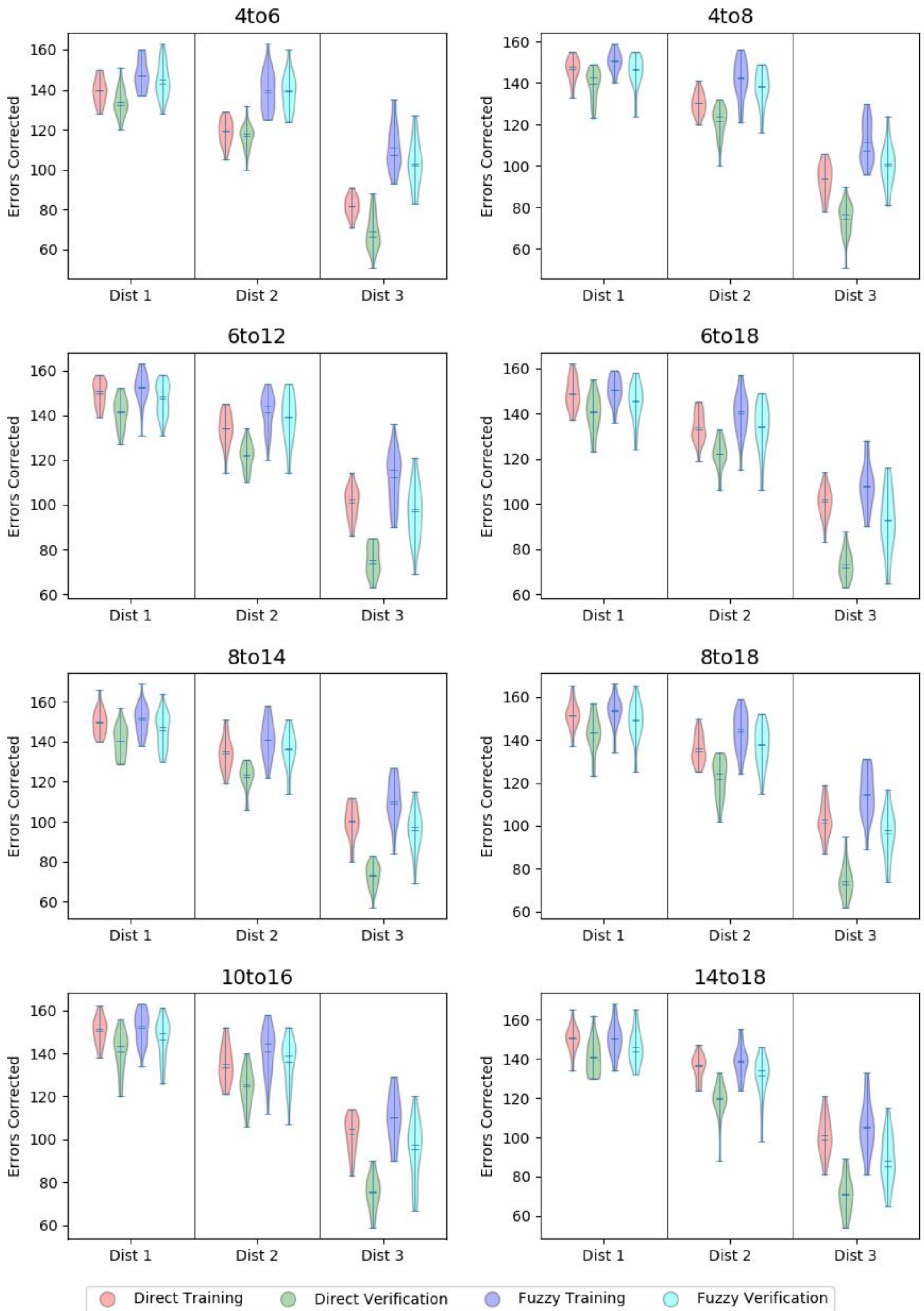

Figure B.17: Code17-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	353	338	10	69.2	337	316.5	22	66.1
	1	150	137	3	88.2	151	131	6	88.8
	2	129	119.5	11	75.9	132	116	8	77.6
	3	89	83	8	52.4	84	70.5	11	49.4
4to8	All	390	370	18	76.5	359	338	18	70.4
	1	159	146.5	10	93.5	151	142	6	88.8
	2	138	130.5	9	81.2	131	121.5	6	77.1
	3	105	95	4	61.8	88	75.5	9	51.8
6to12	All	414	378	31	81.2	375	339.5	18	73.5
	1	160	148	13	94.1	156	139.5	11	91.8
	2	147	133.5	10	86.5	136	125	7	80
	3	116	98	11	68.2	92	74	5	54.1
6to18	All	426	387	31	83.5	372	**345**	27	72.9
	1	163	152	12	95.9	155	144.5	8	91.2
	2	154	136	10	90.6	138	123	8	81.2
	3	119	101.5	11	70	91	76.5	9	53.5
8to14	All	412	389	17	80.8	366	343	25	71.8
	1	162	152	8	95.3	157	143	13	92.4
	2	148	134	10	87.1	136	125	10	80
	3	113	101	8	66.5	88	76	8	51.8
8to18	All	424	388.5	23	83.1	366	344.5	23	71.8
	1	159	150.5	10	93.5	155	142	10	91.2
	2	151	135.5	11	88.8	136	124	9	80
	3	118	101	12	69.4	91	75.5	8	53.5
10to16	All	423	394	23	82.9	369	336	26	72.4
	1	164	151	9	96.5	159	142	8	93.5
	2	156	137	9	91.8	132	123	9	77.6
	3	119	102.5	10	70	84	74	8	49.4
14to18	All	**427**	**404**	20	83.7	**377**	340.5	24	73.9
	1	164	155	9	96.5	156	145	10	91.8
	2	153	139.5	7	90	136	122	9	80
	3	117	106.5	8	68.8	88	74.5	11	51.8

Table B.35: Code17-2, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	458	387	39	89.8	**450**	370	28	88.2
	1	160	144.5	4	94.1	163	142	6	95.9
	2	163	136	19	95.9	160	133.5	8	94.1
	3	135	106.5	16	79.4	127	97.5	18	74.7
4to8	All	443	407.5	32	86.9	424	391.5	39	83.1
	1	162	151	10	95.3	159	149	10	93.5
	2	155	142	11	91.2	153	138.5	11	90
	3	128	112.5	10	75.3	117	101	11	68.8
6to12	All	453	397.5	30	88.8	437	381	39	85.7
	1	162	148	10	95.3	158	145	10	92.9
	2	158	140	13	92.9	156	137	10	91.8
	3	136	110	13	80	123	98.5	17	72.4
6to18	All	456	**421**	37	89.4	430	**399**	35	84.3
	1	164	157	11	96.5	161	153	6	94.7
	2	165	146	11	97.1	153	141	13	90
	3	133	119	17	78.2	117	100.5	13	68.8
8to14	All	**467**	419	35	91.6	440	391.5	48	86.3
	1	167	154.5	9	98.2	167	149.5	12	98.2
	2	162	146.5	15	95.3	156	141.5	17	91.8
	3	138	117	12	81.2	117	101	21	68.8
8to18	All	442	411	40	86.7	414	379	50	81.2
	1	162	152	14	95.3	158	150	12	92.9
	2	156	142.5	13	91.8	151	138	10	88.8
	3	128	113.5	19	75.3	109	93	21	64.1
10to16	All	442	407	24	86.7	433	375.5	36	84.9
	1	165	151.5	9	97.1	165	147	12	97.1
	2	159	145	9	93.5	151	135.5	11	88.8
	3	129	110.5	11	75.9	125	92.5	14	73.5
14to18	All	447	416.5	36	87.6	427	384.5	61	83.7
	1	163	156	8	95.9	163	150	13	95.9
	2	158	142.5	14	92.9	152	136	15	89.4
	3	135	112.5	20	79.4	123	96.5	22	72.4

Table B.36: Code17-2, Fuzzy Classification Fitness Result For Experiment 2

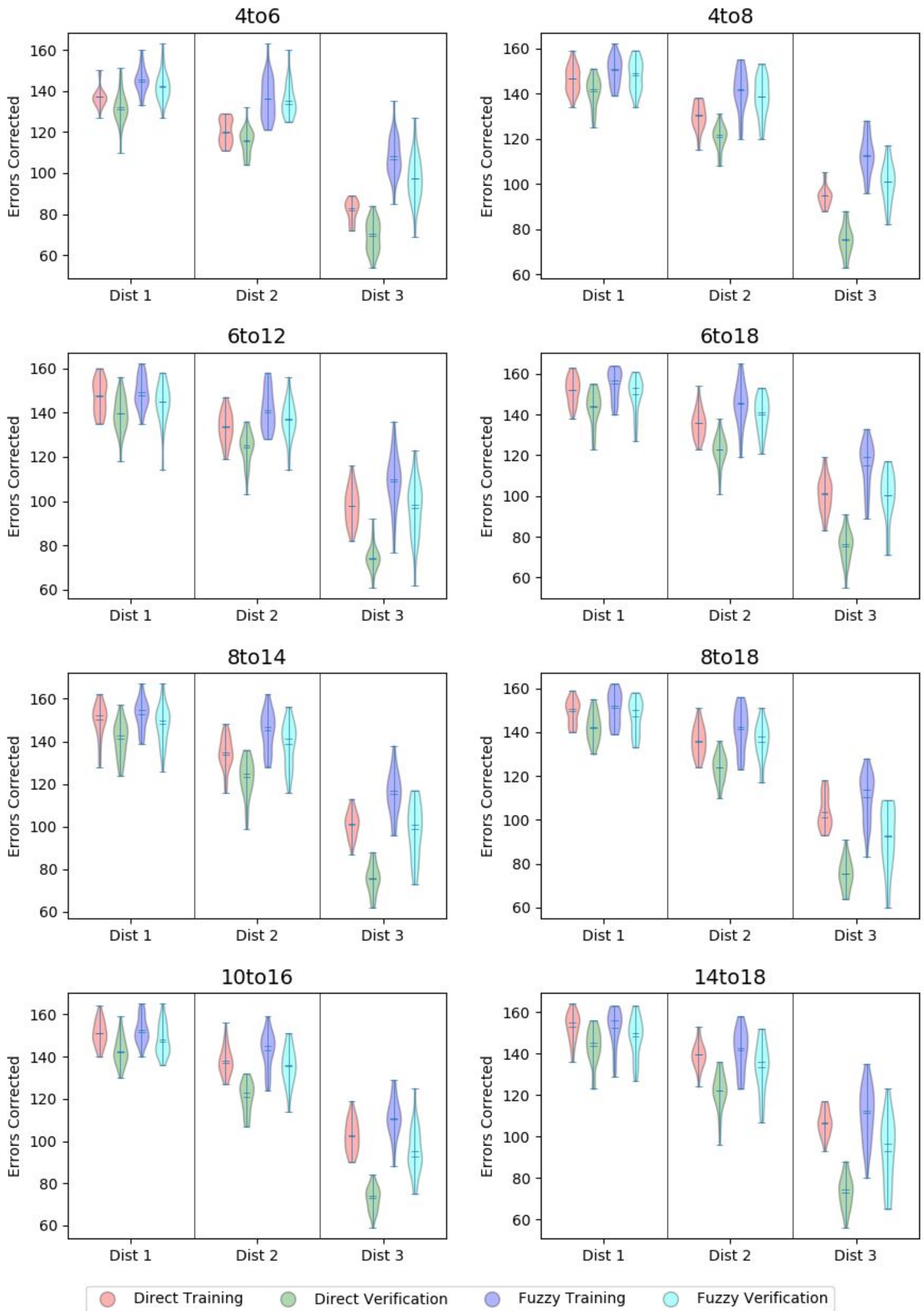

Figure B.18: Code17-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	353	338	16	69.2	337	315.5	20	66.1
	1	150	137	7	88.2	145	131	8	85.3
	2	131	119.5	9	77.1	122	115.5	11	71.8
	3	90	80	9	52.9	83	66.5	9	48.8
4to8	All	386	369.5	16	75.7	364	339	20	71.4
	1	156	146.5	8	91.8	152	142	10	89.4
	2	138	130.5	6	81.2	133	122	10	78.2
	3	106	95	11	62.4	88	75	11	51.8
6to12	All	411	389	21	80.6	372	**344**	17	72.9
	1	166	150.5	8	97.6	159	142	8	93.5
	2	146	137.5	10	85.9	133	126	6	78.2
	3	112	102	8	65.9	91	74.5	8	53.5
6to18	All	429	385	26	84.1	**378**	340.5	15	74.1
	1	161	149	8	94.7	152	142.5	10	89.4
	2	148	135.5	12	87.1	141	124.5	10	82.9
	3	120	99	10	70.6	86	74	6	50.6
8to14	All	408	384.5	14	80	369	336.5	20	72.4
	1	162	148.5	6	95.3	156	140.5	7	91.8
	2	143	135	10	84.1	141	122.5	9	82.9
	3	115	99.5	10	67.6	90	73.5	8	52.9
8to18	All	431	392	25	84.5	367	339.5	31	72
	1	166	151.5	7	97.6	156	144	10	91.8
	2	149	139	10	87.6	135	124	11	79.4
	3	119	103.5	14	70	93	74.5	11	54.7
10to16	All	419	386.5	21	82.2	364	343	29	71.4
	1	161	149.5	6	94.7	154	142	10	90.6
	2	148	133	10	87.1	138	124.5	9	81.2
	3	119	102.5	13	70	87	76	6	51.2
14to18	All	**433**	**398**	24	84.9	364	341	22	71.4
	1	165	153	8	97.1	153	142	10	90
	2	152	138.5	9	89.4	133	122	11	78.2
	3	123	106	13	72.4	85	74	8	50

Table B.37: Code17-2, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	442	387	31	86.7	428	373	33	83.9
	1	159	144.5	8	93.5	158	141	10	92.9
	2	153	139	17	90	154	134	9	90.6
	3	136	106.5	15	80	120	101	16	70.6
4to8	All	455	**415.5**	30	89.2	433	**396**	33	84.9
	1	158	152	7	92.9	160	148	11	94.1
	2	160	147	8	94.1	153	142.5	11	90
	3	138	116	17	81.2	124	104.5	10	72.9
6to12	All	**467**	409	30	91.6	**446**	380.5	32	87.5
	1	169	152	9	99.4	166	146	10	97.6
	2	160	142.5	10	94.1	156	139.5	9	91.8
	3	139	113.5	15	81.8	124	101	16	72.9
6to18	All	441	411.5	30	86.5	425	390.5	27	83.3
	1	161	152	8	94.7	159	148	11	93.5
	2	159	148	12	93.5	155	141.5	9	91.2
	3	132	112.5	12	77.6	119	101.5	12	70
8to14	All	459	403.5	27	90	440	380	38	86.3
	1	166	150.5	7	97.6	165	147	12	97.1
	2	159	143	10	93.5	156	139	13	91.8
	3	140	111.5	12	82.4	123	96.5	21	72.4
8to18	All	451	411	27	88.4	437	384	30	85.7
	1	168	152	9	98.8	165	148	9	97.1
	2	162	146	13	95.3	150	137.5	13	88.2
	3	131	113.5	13	77.1	122	96.5	18	71.8
10to16	All	435	407	29	85.3	416	380.5	32	81.6
	1	162	152	7	95.3	159	147	11	93.5
	2	154	141.5	11	90.6	144	134	11	84.7
	3	127	114	13	74.7	120	99	16	70.6
14to18	All	443	403.5	31	86.9	424	372	38	83.1
	1	165	152.5	7	97.1	166	146	12	97.6
	2	154	142	11	90.6	144	132.5	14	84.7
	3	135	109.5	16	79.4	114	90.5	16	67.1

Table B.38: Code17-2, Fuzzy Classification Fitness Result For Experiment 3

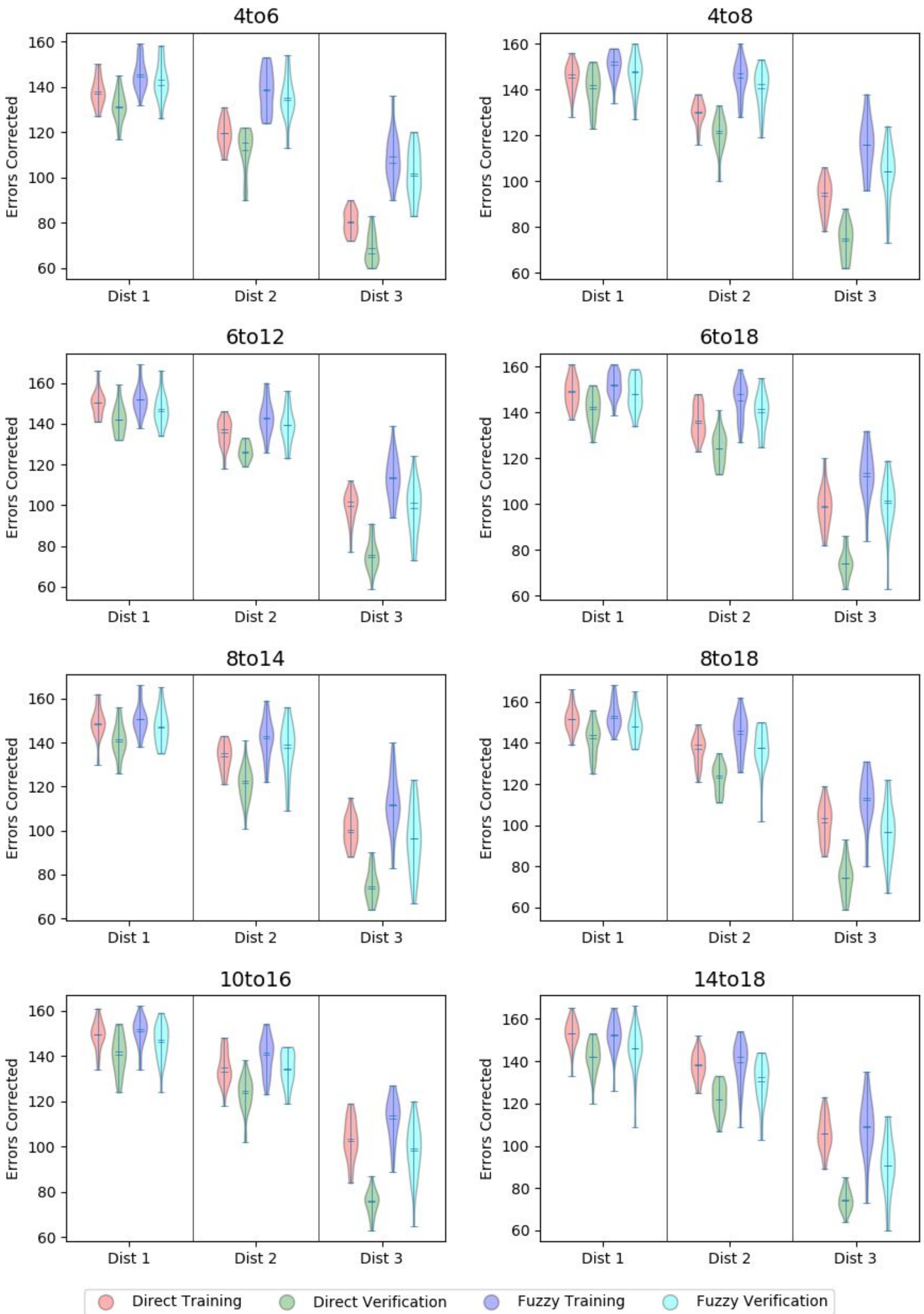

Figure B.19: Code17-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	353	338.5	16	69.2	337	323.5	21	66.1
	1	150	137	9	88.2	145	132	10	85.3
	2	129	119	9	75.9	125	117	8	73.5
	3	90	80.5	4	52.9	84	71	11	49.4
4to8	All	387	371.5	24	75.9	361	343	21	70.8
	1	160	146	9	94.1	156	142.5	10	91.8
	2	139	131	9	81.8	131	123.5	9	77.1
	3	106	94	7	62.4	90	74	7	52.9
6to12	All	404	379.5	29	79.2	378	338.5	19	74.1
	1	159	148	11	93.5	153	139.5	8	90
	2	146	131.5	8	85.9	139	125	7	81.8
	3	111	99	13	65.3	90	73.5	10	52.9
6to18	All	**427**	384.5	24	83.7	366	342	24	71.8
	1	163	150	9	95.9	155	143.5	13	91.2
	2	148	132	12	87.1	131	122	9	77.1
	3	123	100.5	9	72.4	85	77.5	7	50
8to14	All	415	384	28	81.4	**379**	**347**	22	74.3
	1	165	150.5	11	97.1	158	144.5	10	92.9
	2	151	136	7	88.8	136	123.5	8	80
	3	113	99.5	7	66.5	88	75	10	51.8
8to18	All	419	391.5	22	82.2	362	345.5	21	71
	1	166	154	7	97.6	155	145.5	8	91.2
	2	151	138	8	88.8	137	124.5	9	80.6
	3	117	102	13	68.8	92	74.5	9	54.1
10to16	All	418	393	22	82	370	340	19	72.5
	1	163	152.5	7	95.9	158	143	10	92.9
	2	151	138	10	88.8	136	124.5	10	80
	3	113	101.5	12	66.5	93	74	9	54.7
14to18	All	417	**402**	25	81.8	367	336.5	19	72
	1	165	154.5	11	97.1	159	143	6	93.5
	2	149	139	9	87.6	136	123	9	80
	3	121	106.5	10	71.2	81	72	10	47.6

Table B.39: Code17-2, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	427	387.5	29	83.7	428	379	30	83.9
	1	159	145.5	10	93.5	158	141.5	15	92.9
	2	153	137	14	90	154	136	15	90.6
	3	121	107.5	15	71.2	118	103.5	16	69.4
4to8	All	**456**	413.5	25	89.4	429	**401.5**	28	84.1
	1	166	151.5	8	97.6	165	148.5	10	97.1
	2	159	146.5	11	93.5	155	143.5	9	91.2
	3	136	115.5	14	80	128	106	13	75.3
6to12	All	438	402.5	29	85.9	419	385	26	82.2
	1	162	148	9	95.3	156	147	8	91.8
	2	158	142	13	92.9	151	138	11	88.8
	3	125	110	16	73.5	118	98.5	14	69.4
6to18	All	446	406.5	45	87.5	432	383.5	52	84.7
	1	165	151	9	97.1	163	151.5	15	95.9
	2	156	141.5	18	91.8	152	136	14	89.4
	3	130	113	17	76.5	120	96.5	21	70.6
8to14	All	**456**	**421.5**	32	89.4	**435**	399.5	35	85.3
	1	168	155	12	98.8	165	149.5	11	97.1
	2	158	147	11	92.9	155	142	13	91.2
	3	134	117.5	19	78.8	118	105.5	13	69.4
8to18	All	451	409	34	88.4	421	384.5	43	82.5
	1	167	153.5	10	98.2	164	150	10	96.5
	2	159	145	13	93.5	149	139	13	87.6
	3	135	112	16	79.4	121	98	12	71.2
10to16	All	447	401	41	87.6	420	378	45	82.4
	1	167	153.5	10	98.2	166	148	9	97.6
	2	156	142.5	16	91.8	149	136	18	87.6
	3	133	109	19	78.2	114	92	16	67.1
14to18	All	440	404.5	30	86.3	413	376.5	33	81
	1	166	152.5	12	97.6	166	148	11	97.6
	2	156	140	14	91.8	145	133	14	85.3
	3	125	111.5	18	73.5	106	92.5	14	62.4

Table B.40: Code17-2, Fuzzy Classification Fitness Result For Experiment 4

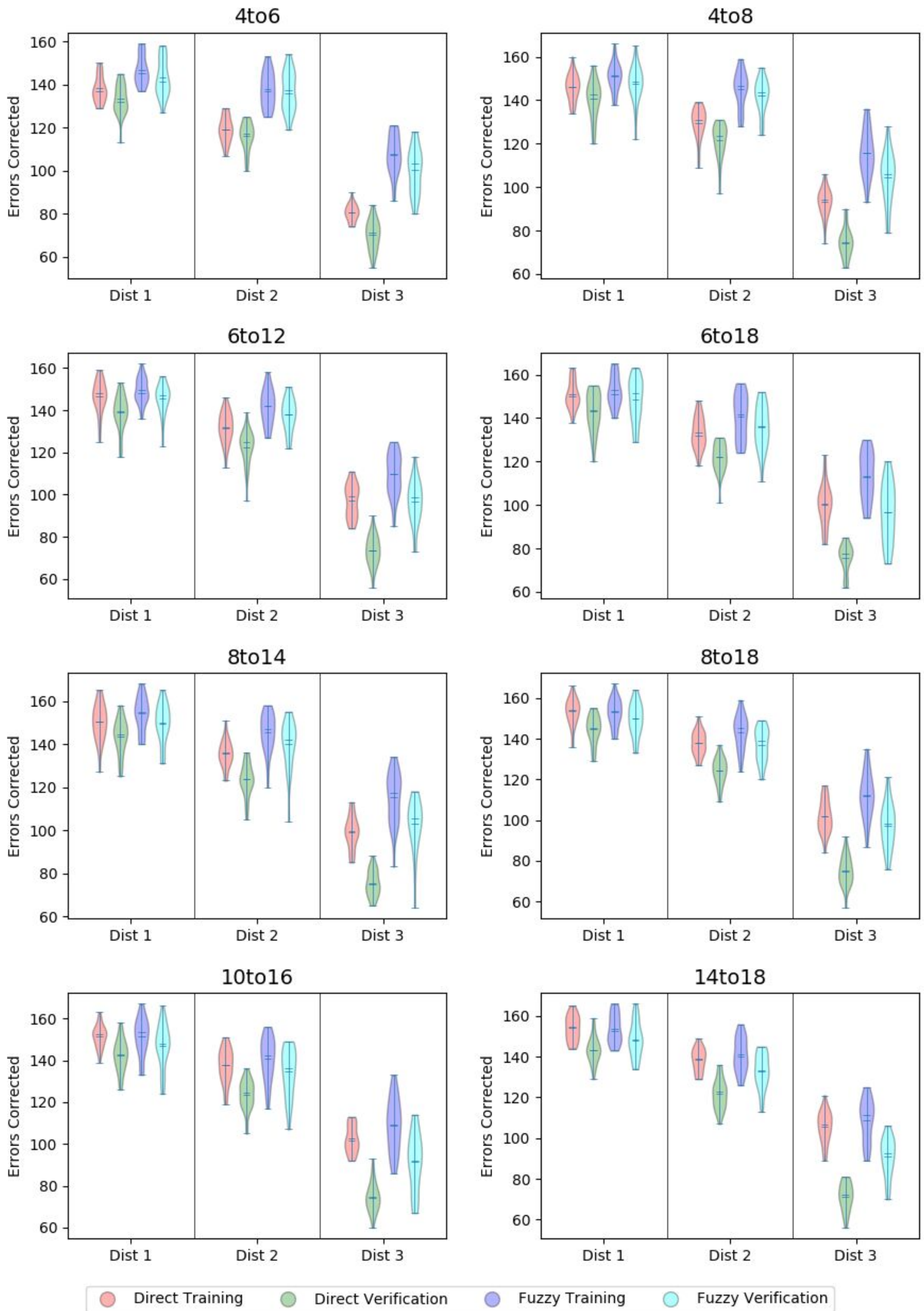

Figure B.20: Code17-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code18

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	369	350.5	17	68.3	345	321.5	23	63.9
	1	156	147	7	86.7	145	137.5	6	80.6
	2	130	124	7	72.2	131	115	10	72.8
	3	92	85	8	51.1	85	69	8	47.2
4to8	All	391	377.5	11	72.4	371	343.5	23	68.7
	1	166	155	12	92.2	161	147.5	8	89.4
	2	140	132	9	77.8	133	122.5	14	73.9
	3	99	88	9	55	84	73.5	11	46.7
6to12	All	425	389	16	78.7	376	346	20	69.6
	1	168	155	10	93.3	160	147.5	10	88.9
	2	149	137.5	7	82.8	138	122	11	76.7
	3	112	96.5	7	62.2	87	74	9	48.3
6to18	All	422	402	21	78.1	368	344	24	68.1
	1	171	160	9	95	163	148.5	9	90.6
	2	152	142.5	14	84.4	137	121.5	11	76.1
	3	116	98.5	9	64.4	85	73.5	10	47.2
8to14	All	418	392	17	77.4	366	**350.5**	14	67.8
	1	166	160	7	92.2	163	152.5	8	90.6
	2	152	137	6	84.4	132	123	8	73.3
	3	109	95.5	13	60.6	88	74	6	48.9
8to18	All	**431**	397	17	79.8	374	349	25	69.3
	1	168	161	8	93.3	165	147.5	9	91.7
	2	154	137	11	85.6	138	125	10	76.7
	3	118	96.5	12	65.6	84	74	11	46.7
10to16	All	414	396.5	13	76.7	380	348	19	70.4
	1	167	157.5	6	92.8	164	150.5	10	91.1
	2	147	138	10	81.7	138	122	10	76.7
	3	109	99.5	8	60.6	84	77	10	46.7
14to18	All	**431**	**403**	18	79.8	**390**	346.5	24	72.2
	1	168	161	10	93.3	160	149	10	88.9
	2	156	141	8	86.7	139	122	10	77.2
	3	121	104.5	9	67.2	95	74.5	12	52.8

Table B.41: Code18, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	441	400.5	40	81.7	435	393	53	80.6
	1	169	154	7	93.9	166	154.5	6	92.2
	2	147	140.5	11	81.7	156	134	21	86.7
	3	138	109	20	76.7	123	107	18	68.3
4to8	All	450	412	25	83.3	439	404	32	81.3
	1	170	157.5	9	94.4	174	157	9	96.7
	2	156	143	13	86.7	156	141.5	15	86.7
	3	132	109	18	73.3	126	102	12	70
6to12	All	455	408	44	84.3	427	389	43	79.1
	1	170	156.5	10	94.4	167	154.5	13	92.8
	2	161	141	12	89.4	154	136	13	85.6
	3	138	111	18	76.7	115	96.5	22	63.9
6to18	All	449	416	20	83.1	430	392.5	29	79.6
	1	173	160	10	96.1	167	156.5	11	92.8
	2	156	146	8	86.7	150	137	13	83.3
	3	125	115.5	15	69.4	117	98.5	11	65
8to14	All	456	**424.5**	28	84.4	**453**	**410.5**	43	83.9
	1	174	163	5	96.7	174	160	9	96.7
	2	158	147	9	87.8	160	142	19	88.9
	3	142	115	18	78.9	123	101.5	17	68.3
8to18	All	460	409.5	29	85.2	445	386	32	82.4
	1	171	159	10	95	170	155.5	11	94.4
	2	161	142.5	11	89.4	156	135	16	86.7
	3	138	106	21	76.7	120	96	12	66.7
10to16	All	451	404.5	27	83.5	443	378	37	82
	1	169	159.5	7	93.9	169	155	10	93.9
	2	157	141	8	87.2	159	132.5	16	88.3
	3	138	107	17	76.7	117	92.5	16	65
14to18	All	**466**	401.5	40	86.3	427	368	34	79.1
	1	170	158	10	94.4	168	153	8	93.3
	2	158	140	12	87.8	157	131	17	87.2
	3	142	105	16	78.9	120	87	15	66.7

Table B.42: Code18, Fuzzy Classification Fitness Result For Experiment 1

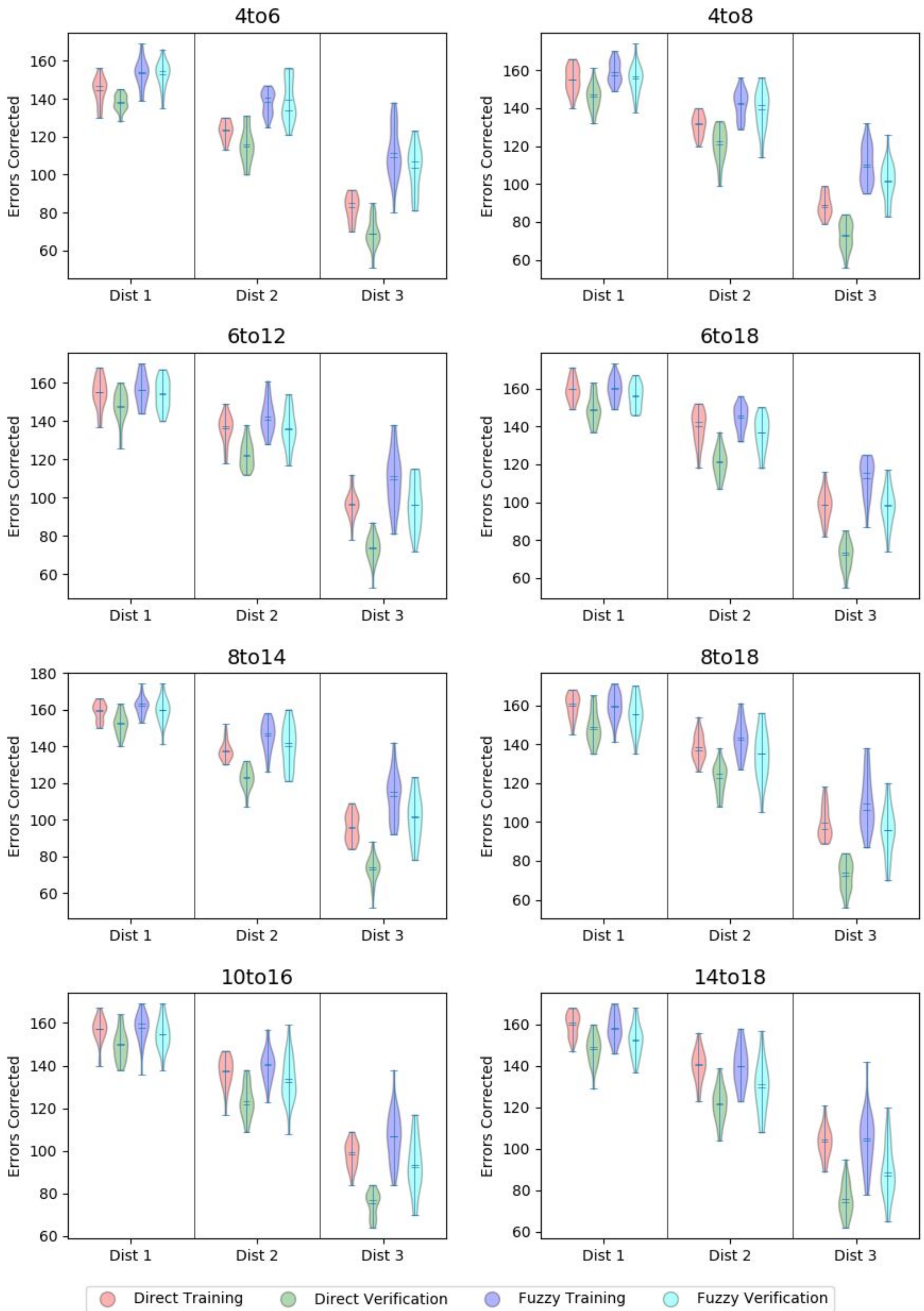

Figure B.21: Code18, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	359	349.5	10	66.5	344	313.5	22	63.7
	1	156	148	12	86.7	146	136	7	81.1
	2	130	123	7	72.2	131	113	14	72.8
	3	88	79	7	48.9	85	65	7	47.2
4to8	All	396	380.5	12	73.3	371	351	26	68.7
	1	168	155	8	93.3	161	147.5	11	89.4
	2	142	135	5	78.9	136	123	13	75.6
	3	104	90	11	57.8	93	73	8	51.7
6to12	All	408	388.5	20	75.6	371	349	17	68.7
	1	166	156.5	8	92.2	164	147	9	91.1
	2	153	136.5	10	85	135	126	7	75
	3	105	96	10	58.3	91	74	5	50.6
6to18	All	438	398	19	81.1	390	346.5	23	72.2
	1	169	160	12	93.9	167	151.5	7	92.8
	2	154	139	7	85.6	136	121	11	75.6
	3	118	97	12	65.6	93	74.5	12	51.7
8to14	All	427	396.5	23	79.1	383	351	29	70.9
	1	169	159	10	93.9	166	152	9	92.2
	2	154	138	6	85.6	139	125	10	77.2
	3	115	99.5	18	63.9	91	75.5	11	50.6
8to18	All	416	396.5	19	77	383	346	28	70.9
	1	170	160	6	94.4	167	151.5	14	92.8
	2	153	138.5	11	85	135	122.5	15	75
	3	108	100.5	8	60	87	75	8	48.3
10to16	All	423	399	20	78.3	376	**353.5**	26	69.6
	1	170	162	6	94.4	166	152	14	92.2
	2	157	139.5	9	87.2	137	124.5	8	76.1
	3	114	100	12	63.3	88	75.5	9	48.9
14to18	All	**439**	**405**	20	81.3	**392**	351	26	72.6
	1	171	159.5	11	95	168	152	12	93.3
	2	157	143	13	87.2	137	124.5	17	76.1
	3	117	106	10	65	87	74.5	14	48.3

Table B.43: Code18, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	430	411.5	18	79.6	434	397	45	80.4
	1	167	156.5	11	92.8	166	154	6	92.2
	2	147	141.5	4	81.7	156	139	18	86.7
	3	130	110	11	72.2	119	104	18	66.1
4to8	All	452	413.5	21	83.7	436	**398**	45	80.7
	1	170	158.5	6	94.4	168	157	11	93.3
	2	158	142.5	7	87.8	160	141	17	88.9
	3	139	112	15	77.2	123	103	18	68.3
6to12	All	450	408.5	33	83.3	436	392	32	80.7
	1	170	158.5	6	94.4	167	158	8	92.8
	2	159	146	10	88.3	155	141	15	86.1
	3	134	110.5	19	74.4	123	97.5	15	68.3
6to18	All	452	417.5	22	83.7	**441**	395.5	38	81.7
	1	169	160	11	93.9	167	159	6	92.8
	2	155	144	8	86.1	158	137	14	87.8
	3	134	111.5	15	74.4	121	97	14	67.2
8to14	All	**463**	411	29	85.7	436	391	32	80.7
	1	168	161.5	7	93.3	166	157	11	92.2
	2	161	144	9	89.4	151	138.5	10	83.9
	3	137	109	20	76.1	121	99.5	16	67.2
8to18	All	454	411.5	29	84.1	437	382.5	44	80.9
	1	169	161	6	93.9	171	154.5	13	95
	2	162	143	12	90	152	134	16	84.4
	3	134	109	12	74.4	121	96	15	67.2
10to16	All	453	**419.5**	37	83.9	434	397	36	80.4
	1	174	163.5	11	96.7	174	158.5	10	96.7
	2	156	143.5	9	86.7	157	137.5	12	87.2
	3	131	109	18	72.8	113	99	22	62.8
14to18	All	462	408	22	85.6	413	378	36	76.5
	1	175	160.5	11	97.2	168	155	11	93.3
	2	164	142.5	9	91.1	148	134.5	16	82.2
	3	127	108.5	9	70.6	107	89.5	18	59.4

Table B.44: Code18, Fuzzy Classification Fitness Result For Experiment 2

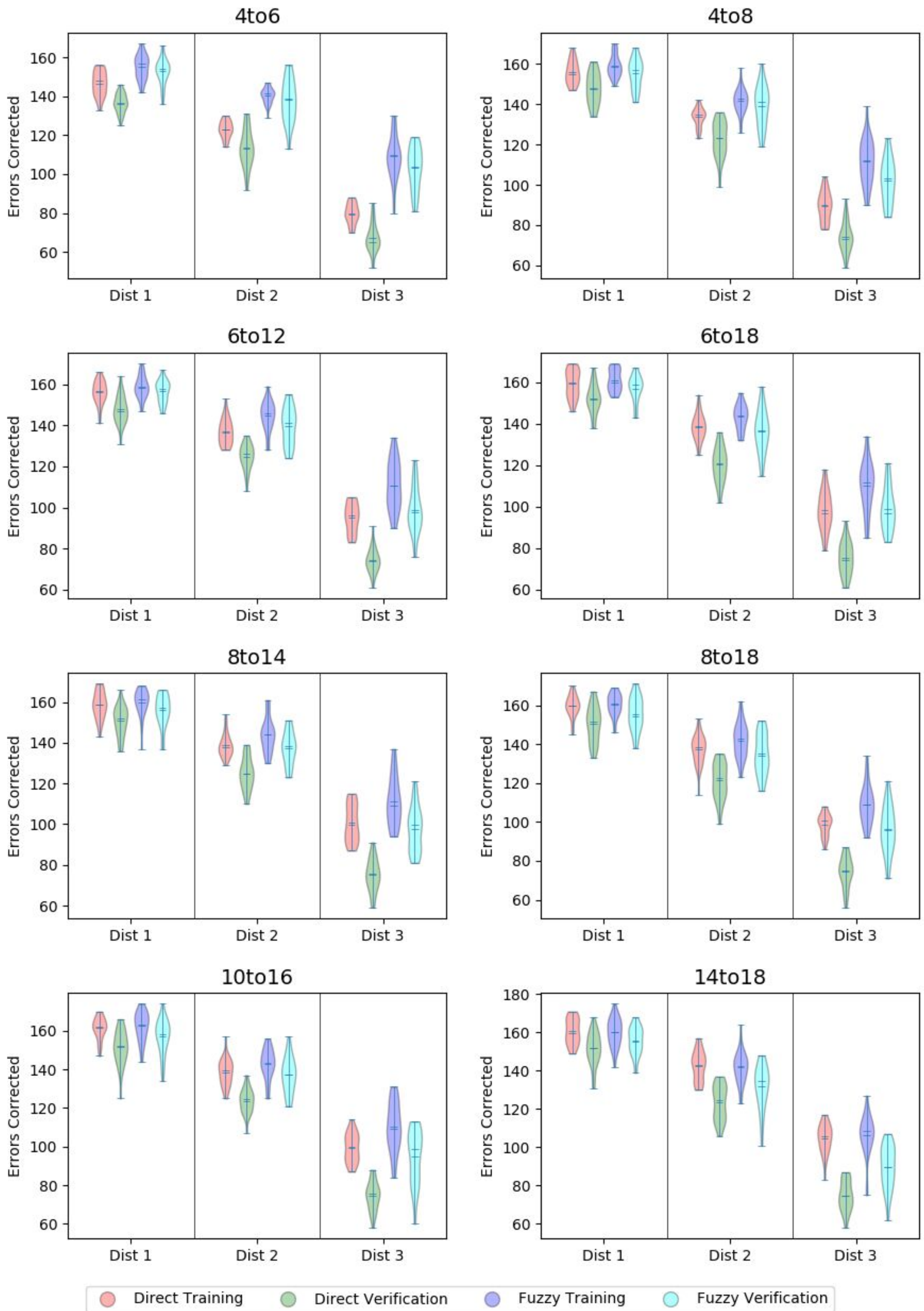

Figure B.22: Code18, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	369	354	17	68.3	341	318.5	25	63.1
	1	154	147	8	85.6	148	137	6	82.2
	2	130	122.5	5	72.2	131	113.5	12	72.8
	3	92	84	9	51.1	79	69	7	43.9
4to8	All	399	378	9	73.9	379	345	15	70.2
	1	165	156	7	91.7	164	150.5	10	91.1
	2	146	133	5	81.1	132	122	10	73.3
	3	100	88.5	9	55.6	88	72.5	12	48.9
6to12	All	412	389.5	20	76.3	371	353	13	68.7
	1	168	157.5	11	93.3	163	151	9	90.6
	2	154	137	11	85.6	137	124	9	76.1
	3	116	94.5	14	64.4	86	76	7	47.8
6to18	All	427	403	24	79.1	385	**353.5**	22	71.3
	1	172	162	7	95.6	169	150	11	93.9
	2	154	139	11	85.6	141	125.5	10	78.3
	3	117	97.5	16	65	92	77.5	13	51.1
8to14	All	426	394.5	23	78.9	374	353	18	69.3
	1	170	158	8	94.4	164	152.5	10	91.1
	2	156	138	10	86.7	139	124	7	77.2
	3	114	98	15	63.3	86	74	8	47.8
8to18	All	426	399.5	22	78.9	**392**	346	27	72.6
	1	172	163	9	95.6	167	152.5	7	92.8
	2	154	140	10	85.6	139	125	13	77.2
	3	115	98	9	63.9	88	76	12	48.9
10to16	All	429	400	24	79.4	389	350	26	72
	1	172	160	6	95.6	171	154.5	12	95
	2	154	140.5	10	85.6	139	122	13	77.2
	3	115	99	10	63.9	89	73	10	49.4
14to18	All	**437**	**415.5**	20	80.9	378	351	21	70
	1	171	163.5	7	95	170	153	7	94.4
	2	155	142	8	86.1	139	122.5	13	77.2
	3	122	105.5	14	67.8	90	75.5	10	50

Table B.45: Code18, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	430	395	46	79.6	432	393	62	80
	1	161	155	7	89.4	164	152.5	6	91.1
	2	148	136.5	11	82.2	156	133.5	17	86.7
	3	130	109	22	72.2	125	104	23	69.4
4to8	All	454	414.5	29	84.1	435	**403.5**	26	80.6
	1	171	160.5	7	95	168	158	8	93.3
	2	161	143	10	89.4	156	141.5	13	86.7
	3	131	110	16	72.8	120	105	9	66.7
6to12	All	**458**	417	30	84.8	**452**	400	28	83.7
	1	171	160	10	95	168	157	4	93.3
	2	159	144.5	7	88.3	163	142	14	90.6
	3	133	111.5	14	73.9	123	100.5	18	68.3
6to18	All	443	420	27	82	430	396	28	79.6
	1	173	164.5	7	96.1	168	156.5	10	93.3
	2	156	145.5	9	86.7	156	138.5	9	86.7
	3	126	109.5	12	70	120	98	16	66.7
8to14	All	447	418	34	82.8	431	400	28	79.8
	1	169	161.5	6	93.9	167	159	8	92.8
	2	158	146	8	87.8	157	138.5	10	87.2
	3	134	114.5	20	74.4	122	101.5	14	67.8
8to18	All	442	417	30	81.9	436	394.5	35	80.7
	1	172	163.5	11	95.6	172	157	7	95.6
	2	159	145.5	12	88.3	153	138	18	85
	3	124	110.5	14	68.9	114	100.5	18	63.3
10to16	All	455	**421**	20	84.3	448	**403.5**	35	83
	1	171	161.5	8	95	176	161.5	12	97.8
	2	161	147	11	89.4	155	139	12	86.1
	3	132	111.5	12	73.3	125	101.5	15	69.4
14to18	All	454	413.5	33	84.1	426	383.5	34	78.9
	1	175	160.5	11	97.2	170	157.5	8	94.4
	2	155	143.5	9	86.1	154	134	11	85.6
	3	126	107	20	70	112	92.5	20	62.2

Table B.46: Code18, Fuzzy Classification Fitness Result For Experiment 3

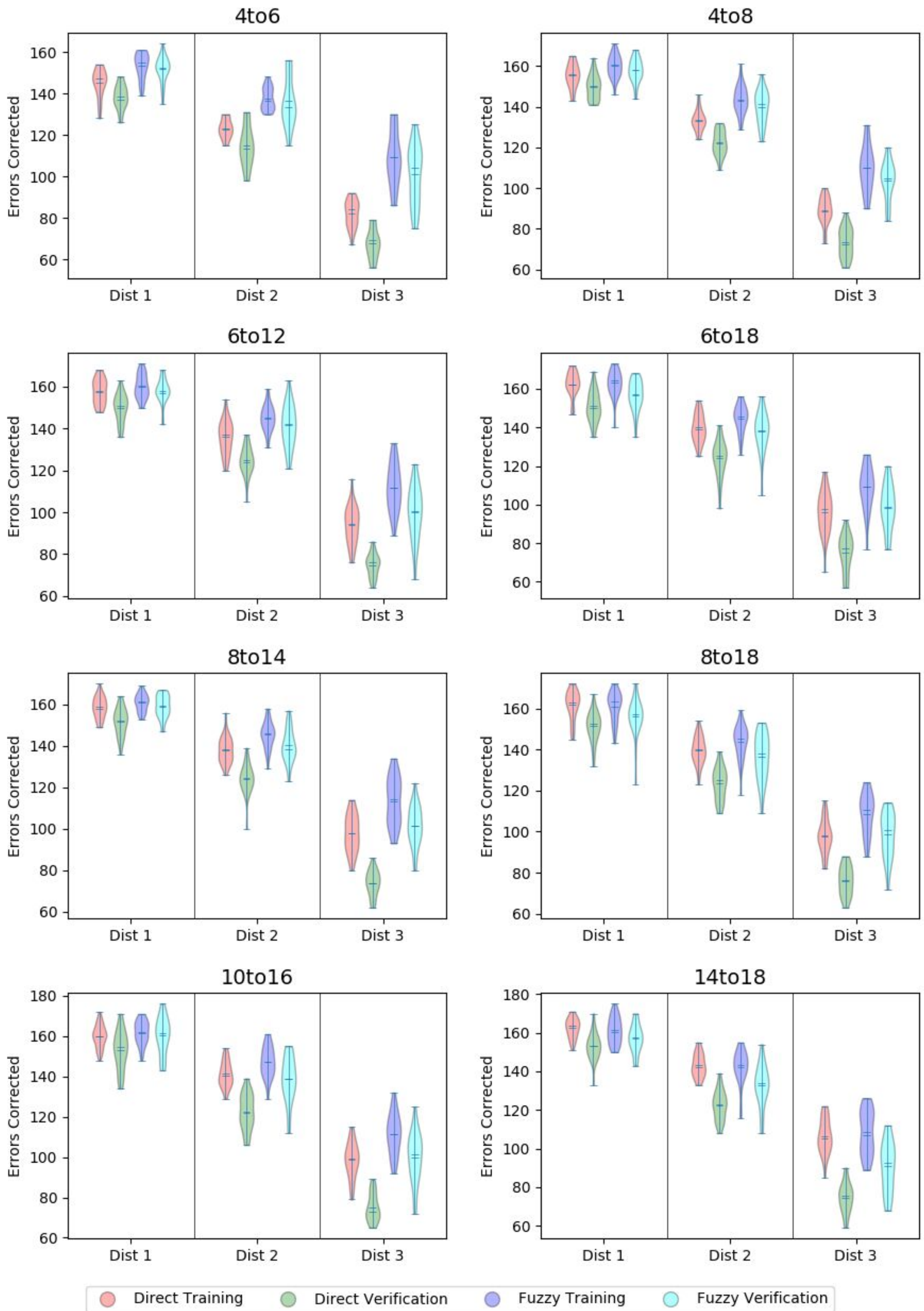

Figure B.23: Code18, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	369	353	14	68.3	347	321.5	25	64.3
	1	155	148	9	86.1	150	138	6	83.3
	2	131	125	6	72.8	131	114.5	8	72.8
	3	92	81.5	10	51.1	85	68	9	47.2
4to8	All	399	381	22	73.9	375	341.5	34	69.4
	1	165	155	8	91.7	166	146	10	92.2
	2	150	135	10	83.3	135	122	18	75
	3	104	89	9	57.8	83	71	13	46.1
6to12	All	428	392.5	19	79.3	374	346.5	26	69.3
	1	172	156	8	95.6	165	150	9	91.7
	2	153	138	10	85	136	123	12	75.6
	3	112	96	12	62.2	87	72	9	48.3
6to18	All	424	392	16	78.5	374	347.5	21	69.3
	1	165	158.5	8	91.7	161	147.5	9	89.4
	2	151	137.5	10	83.9	138	123.5	10	76.7
	3	111	97	12	61.7	93	75	12	51.7
8to14	All	421	392.5	29	78	**385**	348.5	21	71.3
	1	170	160	8	94.4	161	150.5	11	89.4
	2	151	139	11	83.9	137	122	12	76.1
	3	113	98	12	62.8	88	77	10	48.9
8to18	All	422	395	21	78.1	383	347.5	24	70.9
	1	167	159	8	92.8	163	150	6	90.6
	2	149	140	7	82.8	134	123	11	74.4
	3	113	98	9	62.8	88	74	10	48.9
10to16	All	428	397	23	79.3	383	349	27	70.9
	1	173	161	10	96.1	163	150	10	90.6
	2	152	140	11	84.4	136	124.5	8	75.6
	3	118	100.5	11	65.6	92	73.5	18	51.1
14to18	All	**434**	**404.5**	27	80.4	377	**354**	33	69.8
	1	171	159	7	95	170	153	18	94.4
	2	153	141	11	85	139	121.5	8	77.2
	3	122	105.5	8	67.8	92	77.5	10	51.1

Table B.47: Code18, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	**477**	407.5	33	88.3	**456**	396	48	84.4
	1	172	156.5	7	95.6	172	153	9	95.6
	2	164	141.5	9	91.1	157	137	15	87.2
	3	141	109	16	78.3	127	108.5	16	70.6
4to8	All	450	**423.5**	27	83.3	440	**404.5**	27	81.5
	1	170	160.5	10	94.4	172	156.5	11	95.6
	2	160	147	12	88.9	156	143	8	86.7
	3	128	114.5	15	71.1	117	103.5	17	65
6to12	All	456	417.5	25	84.4	434	396	37	80.4
	1	172	159	9	95.6	169	157	7	93.9
	2	155	144.5	10	86.1	159	139.5	13	88.3
	3	133	113.5	13	73.9	119	99.5	15	66.1
6to18	All	456	401.5	36	84.4	446	379	48	82.6
	1	168	161	13	93.3	170	153	12	94.4
	2	161	139	14	89.4	159	134.5	17	88.3
	3	133	108.5	21	73.9	117	95.5	14	65
8to14	All	458	410.5	40	84.8	446	387.5	50	82.6
	1	171	162	12	95	171	156	12	95
	2	162	142	12	90	154	135	12	85.6
	3	136	106	20	75.6	127	96	20	70.6
8to18	All	450	407.5	35	83.3	421	381	42	78
	1	168	160	6	93.3	170	153.5	12	94.4
	2	158	144	13	87.8	152	133	16	84.4
	3	129	107	18	71.7	110	96	14	61.1
10to16	All	455	419	29	84.3	431	400	35	79.8
	1	175	161	11	97.2	170	157.5	9	94.4
	2	161	145	10	89.4	151	138	15	83.9
	3	127	112	15	70.6	115	97.5	13	63.9
14to18	All	447	410	30	82.8	444	377.5	44	82.2
	1	170	160	7	94.4	174	155.5	13	96.7
	2	159	141	11	88.3	150	134	20	83.3
	3	123	109.5	20	68.3	120	87.5	17	66.7

Table B.48: Code18, Fuzzy Classification Fitness Result For Experiment 4

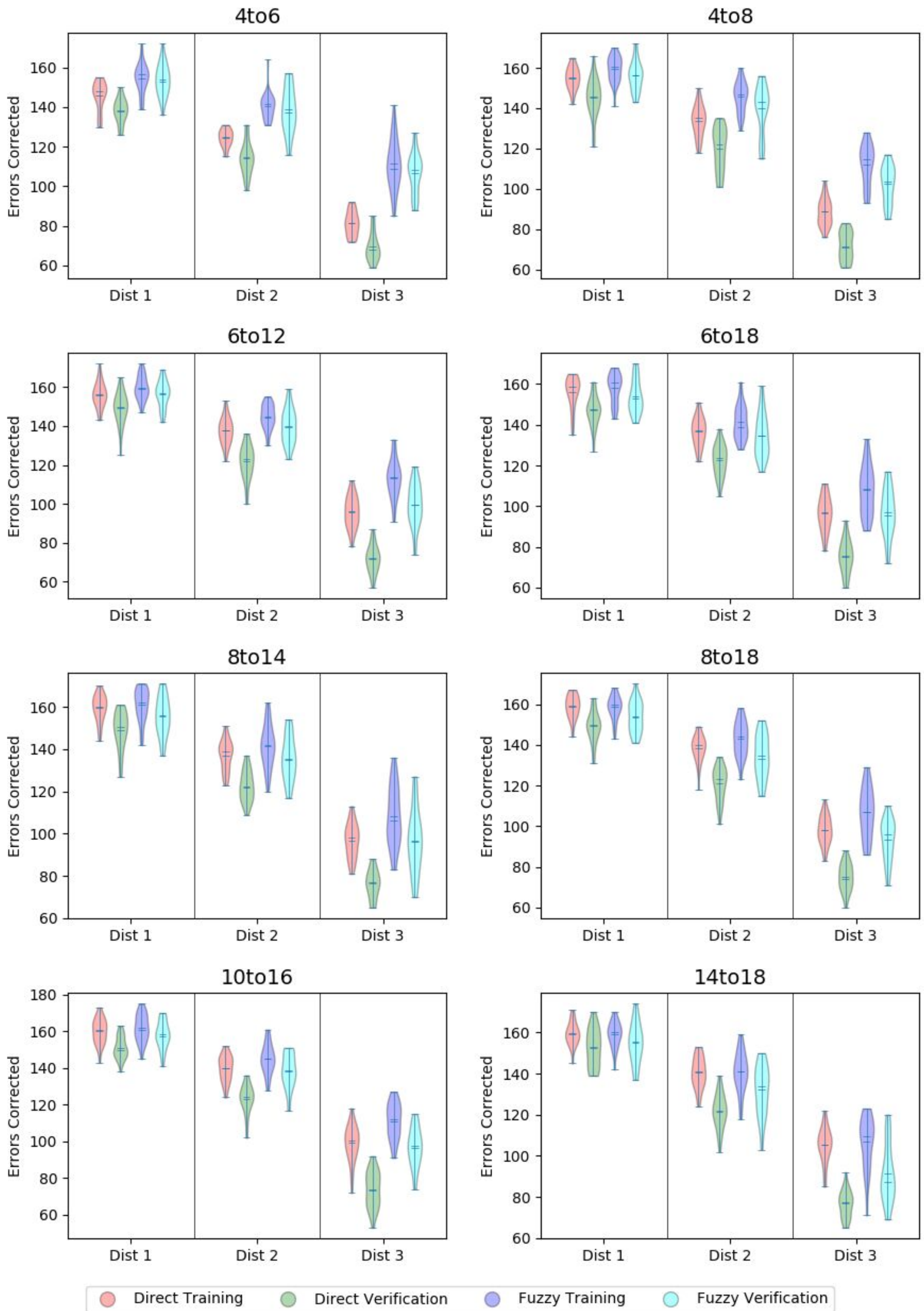

Figure B.24: Code18, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

B.1.3 Codes of Length 14

Code201

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3174	3140	44	37.6	3145	3077.5	92	37.3
	1	1630	1587	20	57.9	1607	1559	29	57.1
	2	1046	994	54	37.2	1020	973	15	36.2
	3	583	544	26	20.7	584	548.5	47	20.8
4to8	All	3882	3805	49	46	3846	3736.5	53	45.6
	1	1864	1806	45	66.2	1839	1766	39	65.4
	2	1362	1305	36	48.4	1337	1291	34	47.5
	3	756	703	40	26.9	741	690	33	26.3
6to12	All	5015	4917	138	59.4	4916	4828.5	164	58.2
	1	2321	2236	63	82.5	2251	2185	63	80
	2	1798	1718	60	63.9	1742	1686	47	61.9
	3	1009	950.5	86	35.9	1003	930.5	61	35.6
6to18	All	5403	5089.5	251	64	**5341**	4998.5	208	63.3
	1	2381	2287	73	84.6	2355	2249.5	98	83.7
	2	1915	1787.5	82	68.1	1900	1747.5	99	67.5
	3	1175	1008.5	112	41.8	1139	993	64	40.5
8to14	All	5252	5086	223	62.2	5130	4999	285	60.8
	1	2340	2287	75	83.2	2306	2243.5	86	81.9
	2	1839	1788.5	76	65.4	1801	1746.5	99	64
	3	1118	1000.5	94	39.7	1062	981.5	83	37.7
8to18	All	5371	**5161**	233	63.6	5333	**5059.5**	223	63.2
	1	2385	2305.5	46	84.8	2353	2257	59	83.6
	2	1902	1806	79	67.6	1859	1775.5	64	66.1
	3	1138	1052	93	40.4	1121	1012	66	39.8
10to16	All	5308	5108	325	62.9	5211	5005	282	61.7
	1	2348	2284	103	83.4	2327	2254	106	82.7
	2	1868	1796.5	110	66.4	1835	1762	92	65.2
	3	1133	1018.5	112	40.3	1079	999	108	38.3
14to18	All	**5413**	5133.5	102	64.1	5309	5044.5	153	62.9
	1	2363	2302	66	84	2365	2258.5	83	84
	2	1915	1799	61	68.1	1835	1778.5	65	65.2
	3	1141	1047.5	43	40.5	1153	1021.5	55	41

Table B.49: Code201, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5491	5363	189	65	5493	5300	118	65.1
	1	2187	2140	77	77.7	2189	2101	59	77.8
	2	1914	1859	65	68	1914	1819	49	68
	3	1418	1348	45	50.4	1424	1344.5	67	50.6
4to8	All	5901	5438.5	175	69.9	5816	5425	231	68.9
	1	2307	2184	48	82	2272	2164.5	66	80.7
	2	2040	1892	73	72.5	2026	1872	95	72
	3	1566	1376.5	87	55.7	1550	1374.5	100	55.1
6to12	All	6608	6200	251	78.3	6535	6118.5	158	77.4
	1	2486	2394	61	88.3	2466	2351	52	87.6
	2	2286	2122	111	81.2	2285	2129.5	73	81.2
	3	1836	1671.5	106	65.2	1811	1657	111	64.4
6to18	All	6567	6309	267	77.8	6542	6277	257	77.5
	1	2528	2440	87	89.8	2489	2408.5	116	88.5
	2	2257	2185	91	80.2	2284	2164.5	96	81.2
	3	1879	1682.5	83	66.8	1826	1674.5	78	64.9
8to14	All	6487	6324.5	187	76.8	6497	6274	167	77
	1	2501	2433	31	88.9	2514	2396.5	48	89.3
	2	2239	2178	45	79.6	2273	2170	75	80.8
	3	1782	1710.5	88	63.3	1805	1708	83	64.1
8to18	All	6505	**6347.5**	228	77.1	6538	**6290.5**	230	77.4
	1	2516	2447.5	56	89.4	2490	2419.5	57	88.5
	2	2274	2174	73	80.8	2269	2174	69	80.6
	3	1780	1685	108	63.3	1791	1687	97	63.6
10to16	All	6740	6220.5	248	79.8	**6666**	6203	296	79
	1	2518	2421.5	80	89.5	2488	2379.5	84	88.4
	2	2317	2141	92	82.3	2299	2166.5	101	81.7
	3	1905	1682.5	118	67.7	1879	1678	129	66.8
14to18	All	**6801**	6261	426	80.6	6658	6252	338	78.9
	1	2516	2421	91	89.4	2502	2394.5	92	88.9
	2	2348	2153.5	130	83.4	2284	2156.5	91	81.2
	3	1937	1680.5	129	68.8	1872	1673	126	66.5

Table B.50: Code201, Fuzzy Classification Fitness Result For Experiment 1

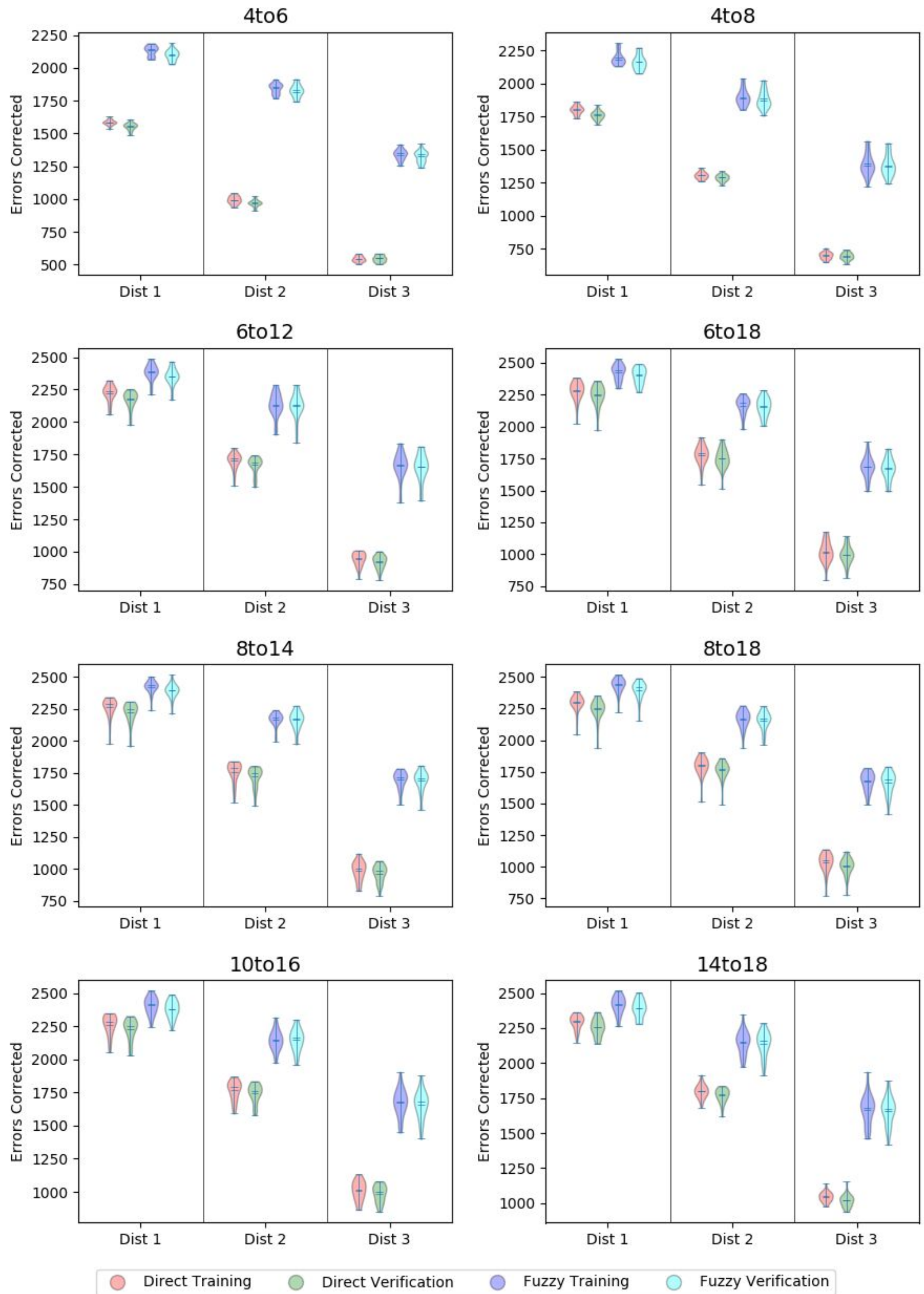

Figure B.25: Code201, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3174	3125.5	50	37.6	3145	3071	16	37.3
	1	1630	1590.5	32	57.9	1597	1559	29	56.8
	2	1046	985	31	37.2	1020	974	24	36.2
	3	583	550	21	20.7	584	549	34	20.8
4to8	All	3876	3814.5	59	45.9	3833	3737	96	45.4
	1	1848	1796	43	65.7	1817	1738.5	57	64.6
	2	1389	1313	31	49.4	1355	1282	50	48.2
	3	761	701	24	27	756	698.5	26	26.9
6to12	All	5004	4895	185	59.3	5014	4809	210	59.4
	1	2322	2224.5	110	82.5	2291	2197	87	81.4
	2	1765	1705	46	62.7	1747	1680	63	62.1
	3	994	951.5	55	35.3	990	940	59	35.2
6to18	All	**5432**	5117.5	193	64.3	5279	4986.5	180	62.5
	1	2387	2287	55	84.8	2333	2254	47	82.9
	2	1917	1795.5	113	68.1	1865	1756.5	70	66.3
	3	1198	1036.5	123	42.6	1106	992	113	39.3
8to14	All	5248	5044.5	279	62.2	5211	4937.5	299	61.7
	1	2335	2276	89	83	2321	2223.5	90	82.5
	2	1847	1779	111	65.6	1817	1732	107	64.6
	3	1070	983	116	38	1087	967.5	105	38.6
8to18	All	5418	5106.5	279	64.2	**5316**	4986	258	63
	1	2428	2287	86	86.3	2407	2248	95	85.5
	2	1916	1798.5	113	68.1	1892	1756.5	100	67.2
	3	1168	1027	145	41.5	1113	978	102	39.6
10to16	All	5336	5146	220	63.2	5257	5047	207	62.3
	1	2358	2301	53	83.8	2353	2275.5	70	83.6
	2	1905	1795.5	92	67.7	1855	1772.5	91	65.9
	3	1109	1044	92	39.4	1112	1017.5	67	39.5
14to18	All	5411	**5173**	79	64.1	5185	**5098**	124	61.4
	1	2390	2306	43	84.9	2347	2273.5	66	83.4
	2	1908	1822	41	67.8	1815	1782.5	42	64.5
	3	1182	1056	50	42	1118	1021	55	39.7

Table B.51: Code201, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5509	5336	190	65.3	5547	5300	198	65.7
	1	2173	2143.5	65	77.2	2167	2101	72	77
	2	1928	1849.5	66	68.5	1926	1851	73	68.4
	3	1414	1331	76	50.2	1454	1320	62	51.7
4to8	All	6095	5425	324	72.2	6104	5348.5	361	72.3
	1	2371	2181.5	90	84.3	2344	2145.5	103	83.3
	2	2106	1877	93	74.8	2125	1858.5	129	75.5
	3	1618	1355	153	57.5	1635	1338.5	159	58.1
6to12	All	6515	6263.5	380	77.2	6555	6257.5	354	77.6
	1	2481	2412.5	89	88.2	2463	2383.5	87	87.5
	2	2242	2176	128	79.7	2289	2161.5	95	81.3
	3	1792	1685	162	63.7	1807	1695	181	64.2
6to18	All	6776	6306	324	80.3	6715	6235.5	367	79.5
	1	2529	2423	80	89.9	2501	2401.5	93	88.9
	2	2311	2177.5	102	82.1	2342	2158	127	83.2
	3	1936	1709	138	68.8	1872	1674	106	66.5
8to14	All	6524	6291.5	193	77.3	6470	6271.5	226	76.6
	1	2476	2425.5	56	88	2451	2408	60	87.1
	2	2269	2175.5	49	80.6	2265	2156.5	76	80.5
	3	1837	1693.5	119	65.3	1783	1701	73	63.4
8to18	All	**6855**	**6320.5**	339	81.2	**6833**	**6301**	268	80.9
	1	2565	2437	83	91.2	2561	2416.5	77	91
	2	2343	2191.5	98	83.3	2364	2167.5	90	84
	3	1947	1702.5	112	69.2	1908	1706	131	67.8
10to16	All	6551	6303	278	77.6	6493	6269	306	76.9
	1	2502	2430	63	88.9	2476	2407	79	88
	2	2260	2183.5	83	80.3	2250	2170.5	99	80
	3	1830	1682.5	132	65	1803	1693.5	101	64.1
14to18	All	6681	6317.5	198	79.1	6670	6239.5	259	79
	1	2542	2428	41	90.3	2533	2399.5	68	90
	2	2298	2178	79	81.7	2301	2165.5	94	81.8
	3	1847	1702.5	81	65.6	1836	1677.5	114	65.2

Table B.52: Code201, Fuzzy Classification Fitness Result For Experiment 2

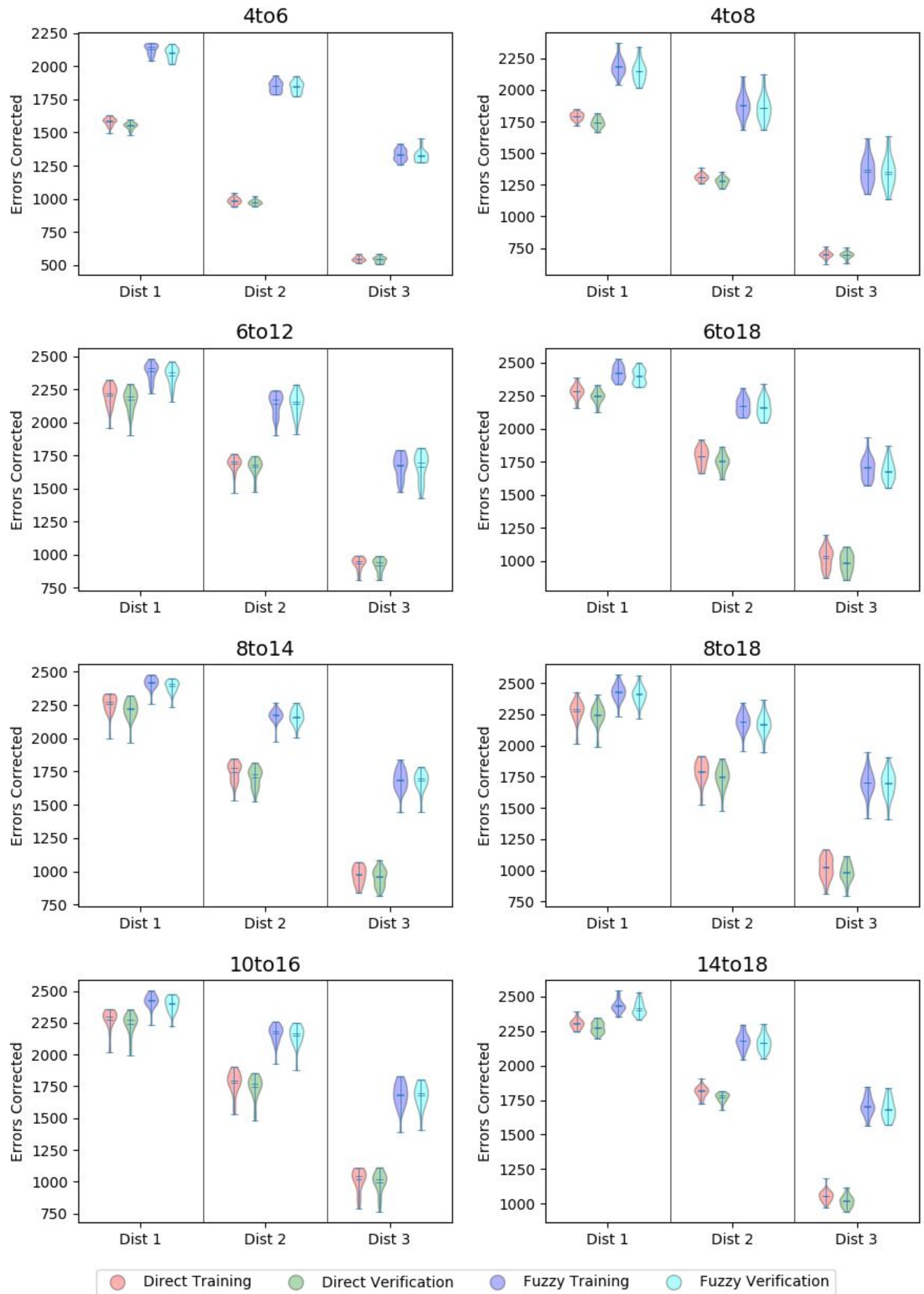

Figure B.26: Code201, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3175	3110	43	37.6	3145	3076.5	20	37.3
	1	1630	1594.5	20	57.9	1597	1561	36	56.8
	2	1046	980.5	42	37.2	1020	968	22	36.2
	3	583	540	22	20.7	584	537.5	30	20.8
4to8	All	3882	3812.5	71	46	3860	3745	38	45.7
	1	1835	1779.5	44	65.2	1807	1753.5	44	64.2
	2	1362	1324.5	56	48.4	1338	1302.5	31	47.5
	3	764	711.5	34	27.1	748	706	26	26.6
6to12	All	5015	4859	308	59.4	4941	4775.5	321	58.5
	1	2307	2240.5	124	82	2285	2187.5	113	81.2
	2	1762	1679	93	62.6	1735	1676.5	112	61.7
	3	1009	923	76	35.9	1003	906.5	71	35.6
6to18	All	5437	5147	259	64.4	5315	5044	223	63
	1	2391	2294	65	85	2343	2260.5	80	83.3
	2	1891	1811.5	93	67.2	1884	1765	87	67
	3	1155	1046	113	41	1115	1012.5	96	39.6
8to14	All	5234	5007.5	296	62	5101	4918.5	342	60.4
	1	2343	2270.5	99	83.3	2318	2224	107	82.4
	2	1837	1734	88	65.3	1807	1723.5	117	64.2
	3	1082	985.5	105	38.5	1046	956.5	91	37.2
8to18	All	5319	5103	138	63	5217	5000	131	61.8
	1	2377	2297.5	58	84.5	2344	2269.5	35	83.3
	2	1877	1791.5	68	66.7	1823	1748	59	64.8
	3	1119	1025	64	39.8	1097	995	49	39
10to16	All	5380	5016.5	237	63.7	5326	4924	253	63.1
	1	2358	2262	72	83.8	2345	2226	86	83.3
	2	1903	1771	79	67.6	1856	1732	91	66
	3	1130	1012	111	40.2	1135	974.5	96	40.3
14to18	All	**5484**	**5187.5**	116	65	**5428**	**5091**	117	64.3
	1	2416	2298.5	66	85.9	2390	2277	65	84.9
	2	1920	1821.5	44	68.2	1894	1781	50	67.3
	3	1161	1059	49	41.3	1144	1024	54	40.7

Table B.53: Code201, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5442	5336	181	64.5	5493	5283	135	65.1
	1	2173	2123	52	77.2	2156	2097.5	47	76.6
	2	1914	1838.5	60	68	1913	1834	47	68
	3	1392	1337	63	49.5	1424	1341.5	70	50.6
4to8	All	6036	5308.5	178	71.5	6033	5258.5	167	71.5
	1	2341	2151.5	41	83.2	2313	2132	64	82.2
	2	2073	1846	47	73.7	2086	1835	65	74.1
	3	1622	1320.5	126	57.6	1634	1317.5	70	58.1
6to12	All	6561	6277	240	77.7	6569	6265	262	77.8
	1	2488	2419.5	72	88.4	2456	2391	75	87.3
	2	2263	2155	97	80.4	2290	2172	93	81.4
	3	1810	1689	88	64.3	1823	1686.5	96	64.8
6to18	All	6696	6272.5	183	79.3	6670	6250.5	209	79
	1	2526	2432	56	89.8	2518	2405.5	94	89.5
	2	2303	2175.5	74	81.8	2316	2162.5	87	82.3
	3	1867	1701.5	96	66.3	1836	1678	78	65.2
8to14	All	6718	6273.5	308	79.6	6679	6223	349	79.1
	1	2562	2416.5	64	91	2535	2381.5	74	90.1
	2	2293	2154	122	81.5	2316	2156.5	112	82.3
	3	1863	1695	147	66.2	1828	1679	153	65
8to18	All	**6740**	6296	236	79.8	**6719**	6250	211	79.6
	1	2564	2434.5	68	91.1	2530	2407	67	89.9
	2	2315	2167.5	73	82.3	2346	2165.5	77	83.4
	3	1861	1697.5	96	66.1	1843	1696.5	98	65.5
10to16	All	6521	6233	169	77.2	6542	6189.5	209	77.5
	1	2491	2415.5	71	88.5	2480	2384.5	68	88.1
	2	2248	2157	55	79.9	2247	2152	85	79.9
	3	1818	1655.5	78	64.6	1815	1650.5	93	64.5
14to18	All	6670	**6378.5**	280	79	6674	**6329.5**	311	79.1
	1	2552	2445.5	75	90.7	2524	2426	85	89.7
	2	2314	2189	81	82.2	2332	2191.5	121	82.9
	3	1846	1719.5	119	65.6	1832	1711.5	127	65.1

Table B.54: Code201, Fuzzy Classification Fitness Result For Experiment 3

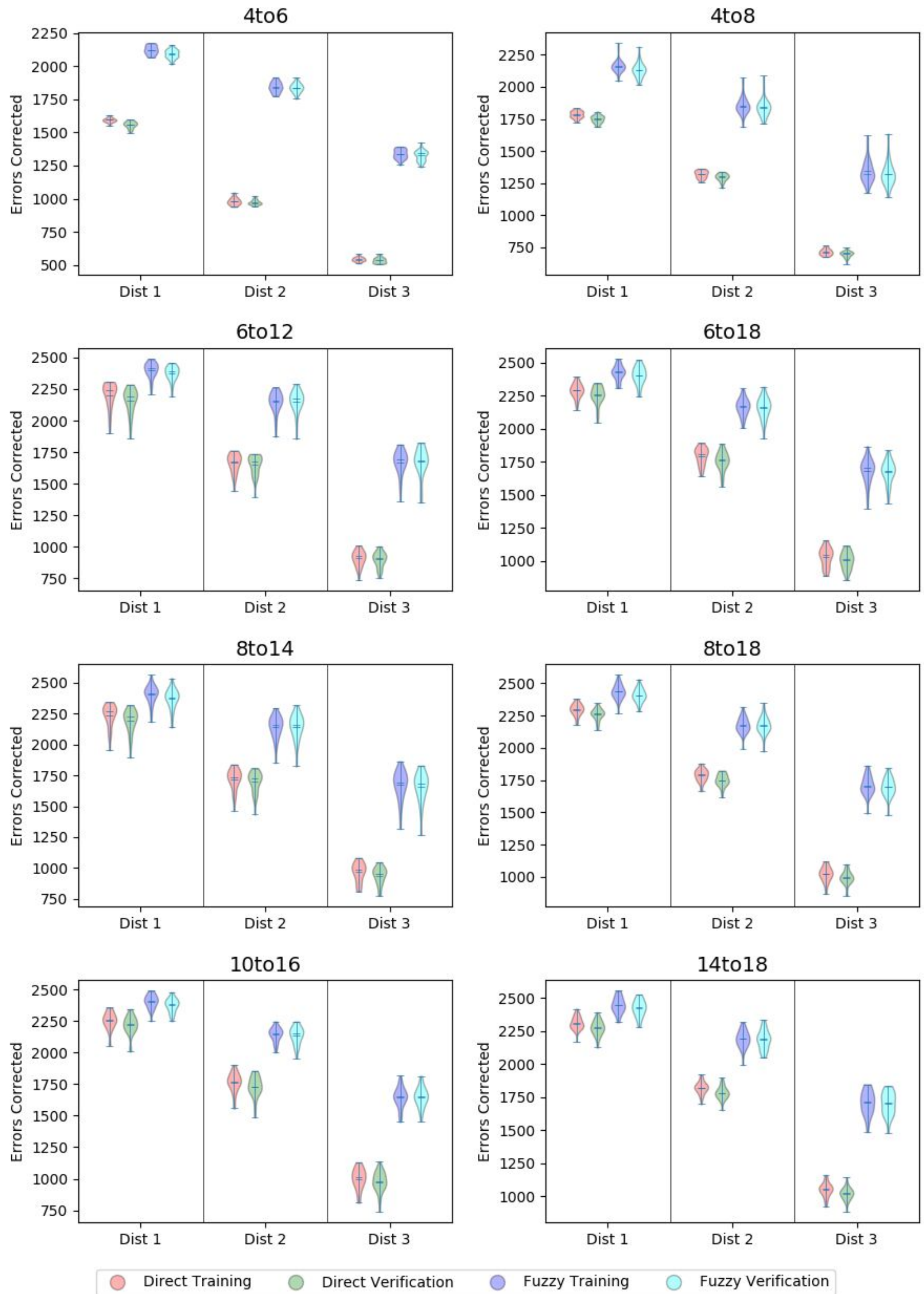

Figure B.27: Code201, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3174	3131	60	37.6	3145	3082	65	37.3
	1	1630	1590	18	57.9	1607	1571.5	22	57.1
	2	1046	987	38	37.2	1020	976	23	36.2
	3	583	539	17	20.7	584	545.5	44	20.8
4to8	All	5320	3827.5	75	63	5252	3766.5	92	62.2
	1	2374	1815	73	84.4	2327	1770.5	79	82.7
	2	1842	1317.5	58	65.5	1835	1300.5	34	65.2
	3	1104	712	29	39.2	1090	697	38	38.7
6to12	All	5006	4871	195	59.3	4933	4784	244	58.4
	1	2317	2232.5	123	82.3	2265	2195	111	80.5
	2	1744	1704.5	86	62	1733	1669	84	61.6
	3	1011	902.5	86	35.9	1003	913.5	61	35.6
6to18	All	5354	5105.5	230	63.4	5301	5010	232	62.8
	1	2396	2279	92	85.1	2369	2247	93	84.2
	2	1929	1800	102	68.6	1854	1761	98	65.9
	3	1135	1039	116	40.3	1088	999	59	38.7
8to14	All	5239	4939	254	62.1	5162	4875.5	239	61.1
	1	2350	2266	51	83.5	2342	2233	64	83.2
	2	1817	1730	86	64.6	1820	1709.5	95	64.7
	3	1115	960.5	129	39.6	1084	953.5	120	38.5
8to18	All	5363	5082.5	366	63.5	**5305**	5052	383	62.8
	1	2387	2295.5	119	84.8	2348	2249	114	83.4
	2	1904	1777	142	67.7	1865	1768	125	66.3
	3	1154	1001	155	41	1111	992	105	39.5
10to16	All	**5403**	5035	301	64	5258	4926	295	62.3
	1	2383	2271.5	101	84.7	2333	2227.5	82	82.9
	2	1914	1768	128	68	1877	1730.5	127	66.7
	3	1125	1006	139	40	1075	968	122	38.2
14to18	All	5388	**5186.5**	156	63.8	5279	**5124.5**	150	62.5
	1	2433	2322.5	55	86.5	2402	2295.5	65	85.4
	2	1909	1819	59	67.8	1846	1786.5	46	65.6
	3	1121	1048	65	39.8	1124	1022.5	61	39.9

Table B.55: Code201, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5504	5336	189	65.2	5493	5286.5	187	65.1
	1	2173	2133.5	56	77.2	2156	2101	50	76.6
	2	1914	1865	53	68	1913	1839	67	68
	3	1464	1331	64	52	1424	1342	74	50.6
4to8	All	6558	5539.5	467	77.7	6527	5483.5	557	77.3
	1	2501	2209.5	105	88.9	2479	2179.5	131	88.1
	2	2243	1910	156	79.7	2262	1903.5	158	80.4
	3	1818	1424	217	64.6	1818	1400.5	240	64.6
6to12	All	6527	6288.5	243	77.3	6470	6223	291	76.6
	1	2467	2420	67	87.7	2438	2385.5	68	86.6
	2	2267	2171	119	80.6	2257	2156	114	80.2
	3	1793	1689	132	63.7	1805	1682	149	64.1
6to18	All	6676	6269	196	79.1	6666	6211	223	79
	1	2546	2425	58	90.5	2522	2396	75	89.6
	2	2282	2178.5	60	81.1	2301	2152	73	81.8
	3	1855	1673	117	65.9	1851	1671	144	65.8
8to14	All	6575	6287	195	77.9	6600	6265.5	247	78.2
	1	2489	2417	35	88.5	2487	2396	60	88.4
	2	2273	2172	60	80.8	2307	2167.5	73	82
	3	1829	1697.5	61	65	1808	1702.5	99	64.3
8to18	All	6707	6312.5	324	79.4	6650	6300	339	78.8
	1	2552	2430.5	74	90.7	2528	2390.5	95	89.8
	2	2307	2168.5	109	82	2319	2176.5	111	82.4
	3	1848	1695	142	65.7	1821	1701.5	102	64.7
10to16	All	6568	6324.5	241	77.8	6526	6257.5	228	77.3
	1	2531	2428.5	59	89.9	2499	2409	67	88.8
	2	2262	2172	74	80.4	2277	2177	105	80.9
	3	1795	1705.5	112	63.8	1810	1670.5	97	64.3
14to18	All	**6820**	**6459**	345	80.8	**6753**	**6381**	303	80
	1	2586	2478.5	87	91.9	2572	2449.5	79	91.4
	2	2348	2223	88	83.4	2366	2202.5	123	84.1
	3	1905	1748.5	152	67.7	1905	1751.5	119	67.7

Table B.56: Code201, Fuzzy Classification Fitness Result For Experiment 4

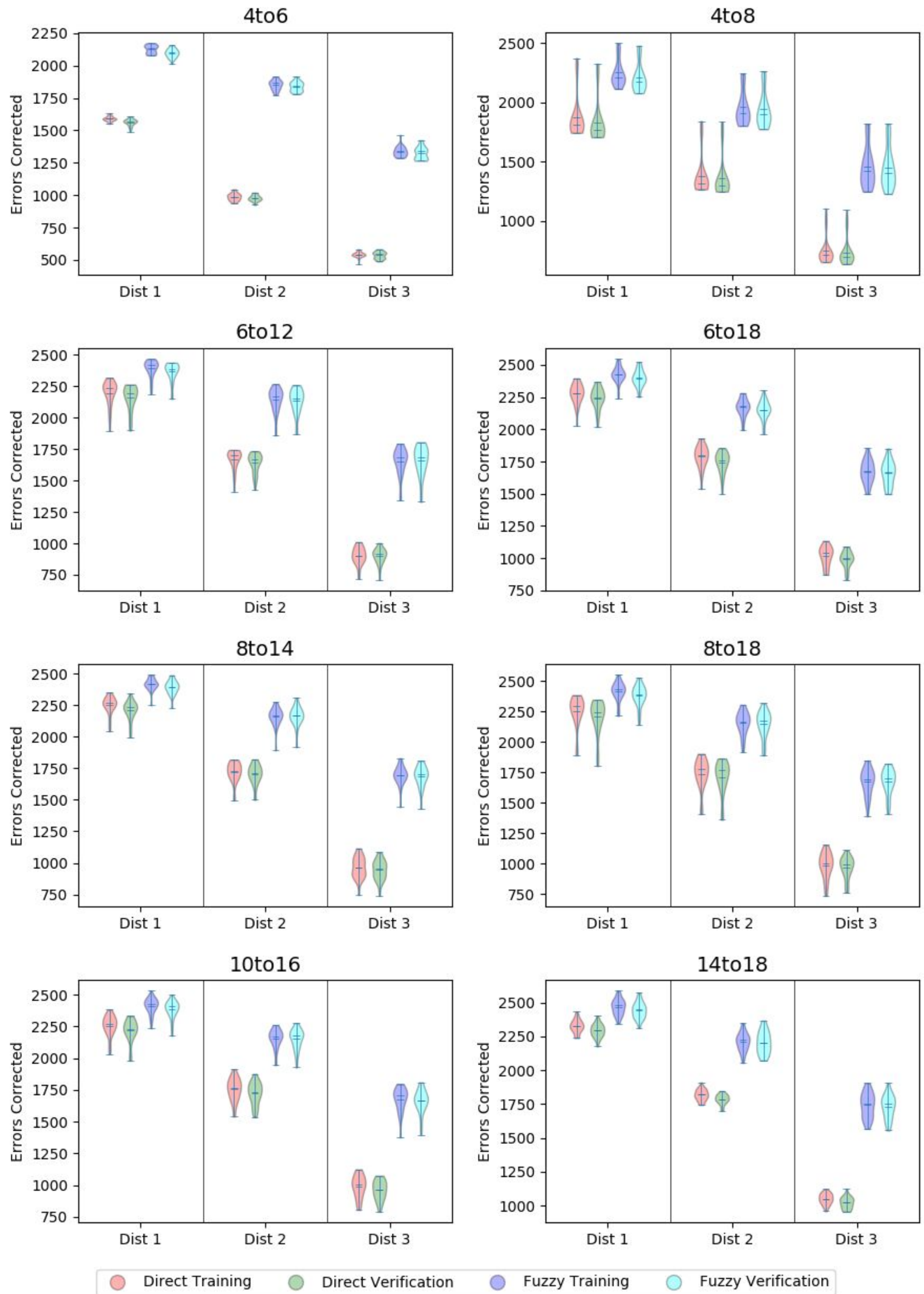

Figure B.28: Code201, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code205-1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3333	3217.5	44	38.7	3231	3159	77	37.5
	1	1679	1628	32	58.5	1651	1601	44	57.5
	2	1097	1007	30	38.2	1039	1005.5	30	36.2
	3	609	571.5	22	21.2	574	559	23	20
4to8	All	4005	3937	55	46.5	3980	3863	56	46.2
	1	1881	1840	26	65.5	1872	1829	37	65.2
	2	1386	1339.5	36	48.3	1377	1319	37	48
	3	782	745.5	28	27.2	774	701	35	27
6to12	All	5107	4961.5	145	59.3	5086	4900.5	172	59.1
	1	2331	2262.5	46	81.2	2325	2255.5	52	81
	2	1788	1738	45	62.3	1781	1715	52	62.1
	3	1042	962	74	36.3	1018	925.5	54	35.5
6to18	All	5537	**5290.5**	287	64.3	5460	5122	339	63.4
	1	2461	2354.5	84	85.7	2447	2330	104	85.3
	2	1967	1854.5	102	68.5	1909	1787	116	66.5
	3	1195	1070	109	41.6	1120	993	95	39
8to14	All	5288	5125.5	192	61.4	5207	5018	229	60.5
	1	2384	2329.5	85	83.1	2375	2303.5	57	82.8
	2	1866	1789	84	65	1836	1748.5	51	64
	3	1095	1019.5	70	38.2	1035	958	96	36.1
8to18	All	**5571**	5248.5	226	64.7	5386	5132.5	168	62.6
	1	2451	2347.5	90	85.4	2449	2340	63	85.3
	2	1972	1840	79	68.7	1875	1782	80	65.3
	3	1150	1031.5	107	40.1	1095	1005	109	38.2
10to16	All	5496	5203	279	63.8	5392	5075	287	62.6
	1	2432	2348	65	84.7	2420	2317	111	84.3
	2	1921	1818.5	100	66.9	1893	1779	117	66
	3	1143	1032	90	39.8	1100	973.5	95	38.3
14to18	All	5540	5282	168	64.3	**5400**	**5182**	177	62.7
	1	2435	2363.5	67	84.8	2413	2353	71	84.1
	2	1955	1857	69	68.1	1896	1807.5	55	66.1
	3	1158	1066	49	40.3	1116	1033.5	48	38.9

Table B.57: Code205-1, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5662	5530.5	123	65.8	5593	5417	140	65
	1	2243	2197	34	78.2	2235	2168	66	77.9
	2	1931	1883	49	67.3	1938	1886	74	67.5
	3	1494	1423.5	60	52.1	1439	1347	27	50.1
4to8	All	6131	5621	317	71.2	6051	5566.5	302	70.3
	1	2367	2244.5	84	82.5	2373	2247	73	82.7
	2	2111	1916	114	73.6	2076	1930.5	89	72.3
	3	1653	1440	114	57.6	1602	1387.5	139	55.8
6to12	All	6725	6306	254	78.1	6708	6255.5	209	77.9
	1	2562	2434.5	61	89.3	2536	2431	64	88.4
	2	2327	2175	87	81.1	2327	2159.5	76	81.1
	3	1878	1698.5	123	65.4	1853	1683	107	64.6
6to18	All	6839	6409	168	79.4	6755	6333	221	78.5
	1	2608	2485.5	50	90.9	2594	2473.5	52	90.4
	2	2398	2208	66	83.6	2326	2193.5	91	81
	3	1833	1726.5	103	63.9	1839	1678	85	64.1
8to14	All	**6843**	6405	271	79.5	**6826**	6360	226	79.3
	1	2575	2480.5	73	89.7	2559	2457.5	73	89.2
	2	2366	2204.5	101	82.4	2389	2184.5	77	83.2
	3	1902	1723	90	66.3	1878	1695	105	65.4
8to18	All	6809	**6495**	273	79.1	6804	6421.5	293	79
	1	2590	2504.5	91	90.2	2613	2496	110	91
	2	2375	2242	106	82.8	2348	2214.5	125	81.8
	3	1878	1734.5	117	65.4	1864	1705	122	64.9
10to16	All	6812	6461.5	244	79.1	6794	**6424**	321	78.9
	1	2593	2503.5	70	90.3	2594	2476	91	90.4
	2	2361	2231.5	105	82.3	2365	2215	122	82.4
	3	1858	1730.5	109	64.7	1835	1714	100	63.9
14to18	All	6775	6491.5	252	78.7	6719	6418.5	234	78
	1	2581	2501.5	51	89.9	2585	2483	55	90.1
	2	2353	2244.5	63	82	2311	2221.5	53	80.5
	3	1865	1745	88	65	1863	1728	119	64.9

Table B.58: Code205-1, Fuzzy Classification Fitness Result For Experiment 1

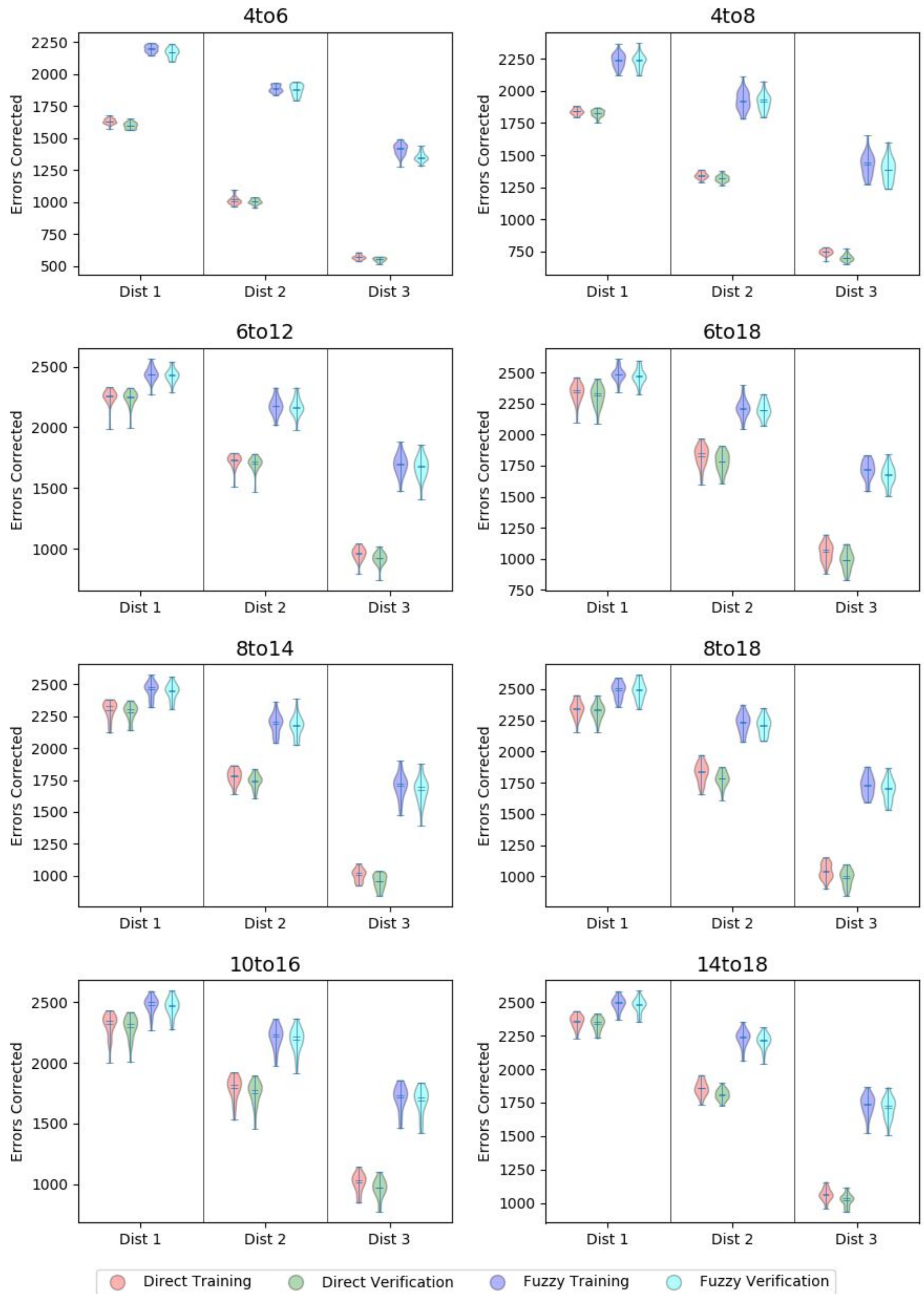

Figure B.29: code205-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3333	3229	91	38.7	3239	3181	97	37.6
	1	1679	1628	35	58.5	1651	1604.5	42	57.5
	2	1097	1014.5	34	38.2	1050	1008.5	32	36.6
	3	609	572	29	21.2	582	559	22	20.3
4to8	All	4005	3933	57	46.5	3980	3822.5	71	46.2
	1	1889	1825	33	65.8	1872	1810	25	65.2
	2	1386	1356.5	37	48.3	1377	1310	38	48
	3	782	754	35	27.2	774	693.5	25	27
6to12	All	5120	4958	195	59.5	5098	4876.5	130	59.2
	1	2346	2283	87	81.7	2325	2256	58	81
	2	1798	1730	45	62.6	1771	1700	55	61.7
	3	1021	943.5	68	35.6	1025	909	48	35.7
6to18	All	5467	5226.5	197	63.5	5397	5109	189	62.7
	1	2440	2342	66	85	2417	2331	75	84.2
	2	1931	1832	70	67.3	1916	1793.5	67	66.8
	3	1146	1052	81	39.9	1108	985	93	38.6
8to14	All	5351	5103.5	270	62.1	5262	5005.5	227	61.1
	1	2395	2298.5	88	83.4	2393	2288.5	90	83.4
	2	1861	1769	99	64.8	1845	1744	98	64.3
	3	1135	1009	89	39.5	1087	938	79	37.9
8to18	All	5538	5252.5	274	64.3	5413	**5185**	250	62.9
	1	2469	2348.5	97	86	2457	2331.5	69	85.6
	2	1950	1839	93	67.9	1890	1812	96	65.9
	3	1214	1056.5	69	42.3	1155	1021	96	40.2
10to16	All	5500	5120.5	555	63.9	5358	4981	550	62.2
	1	2444	2327.5	225	85.2	2414	2292.5	178	84.1
	2	1943	1796.5	189	67.7	1897	1742	196	66.1
	3	1138	1014.5	143	39.7	1093	957	175	38.1
14to18	All	**5648**	**5305**	110	65.6	**5458**	5184.5	93	63.4
	1	2492	2378	46	86.8	2446	2354	52	85.2
	2	2000	1851	72	69.7	1918	1806.5	59	66.8
	3	1199	1079	63	41.8	1111	1030.5	42	38.7

Table B.59: Code205-1, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5638	5524	86	65.5	5530	5417	80	64.2
	1	2243	2198	41	78.2	2219	2169	42	77.3
	2	1930	1891	44	67.2	1933	1886	57	67.4
	3	1465	1403	57	51	1415	1358	30	49.3
4to8	All	5822	5548.5	252	67.6	5710	5487	359	66.3
	1	2296	2199.5	73	80	2308	2221.5	89	80.4
	2	2010	1904	87	70	1970	1883	120	68.6
	3	1516	1422.5	134	52.8	1471	1363	126	51.3
6to12	All	6672	6375	254	77.5	6696	6338	257	77.8
	1	2530	2461.5	74	88.2	2539	2448.5	61	88.5
	2	2309	2187	89	80.5	2314	2190.5	114	80.6
	3	1833	1716	94	63.9	1844	1704	120	64.3
6to18	All	6824	6473	352	79.3	6746	**6434**	297	78.4
	1	2588	2482.5	96	90.2	2581	2477.5	89	89.9
	2	2374	2238	126	82.7	2341	2224	88	81.6
	3	1875	1743	123	65.3	1852	1731.5	125	64.5
8to14	All	6819	6382	215	79.2	**6847**	6306.5	178	79.5
	1	2575	2474.5	85	89.7	2580	2445.5	48	89.9
	2	2372	2186.5	59	82.6	2342	2182.5	56	81.6
	3	1872	1707.5	86	65.2	1925	1693	100	67.1
8to18	All	**6893**	6396	201	80.1	6836	6357	168	79.4
	1	2606	2468	70	90.8	2620	2457	68	91.3
	2	2394	2209.5	81	83.4	2346	2193.5	62	81.7
	3	1909	1724	69	66.5	1870	1707.5	71	65.2
10to16	All	6815	6366.5	420	79.2	6747	6311	469	78.4
	1	2638	2475	121	91.9	2635	2443.5	124	91.8
	2	2353	2181.5	122	82	2341	2178	162	81.6
	3	1861	1703.5	169	64.8	1802	1663	184	62.8
14to18	All	6770	**6488.5**	179	78.6	6669	6414	141	77.5
	1	2622	2509	60	91.4	2593	2487.5	50	90.3
	2	2367	2238.5	54	82.5	2311	2208.5	57	80.5
	3	1830	1742.5	83	63.8	1828	1722.5	85	63.7

Table B.60: Code205-1, Fuzzy Classification Fitness Result For Experiment 2

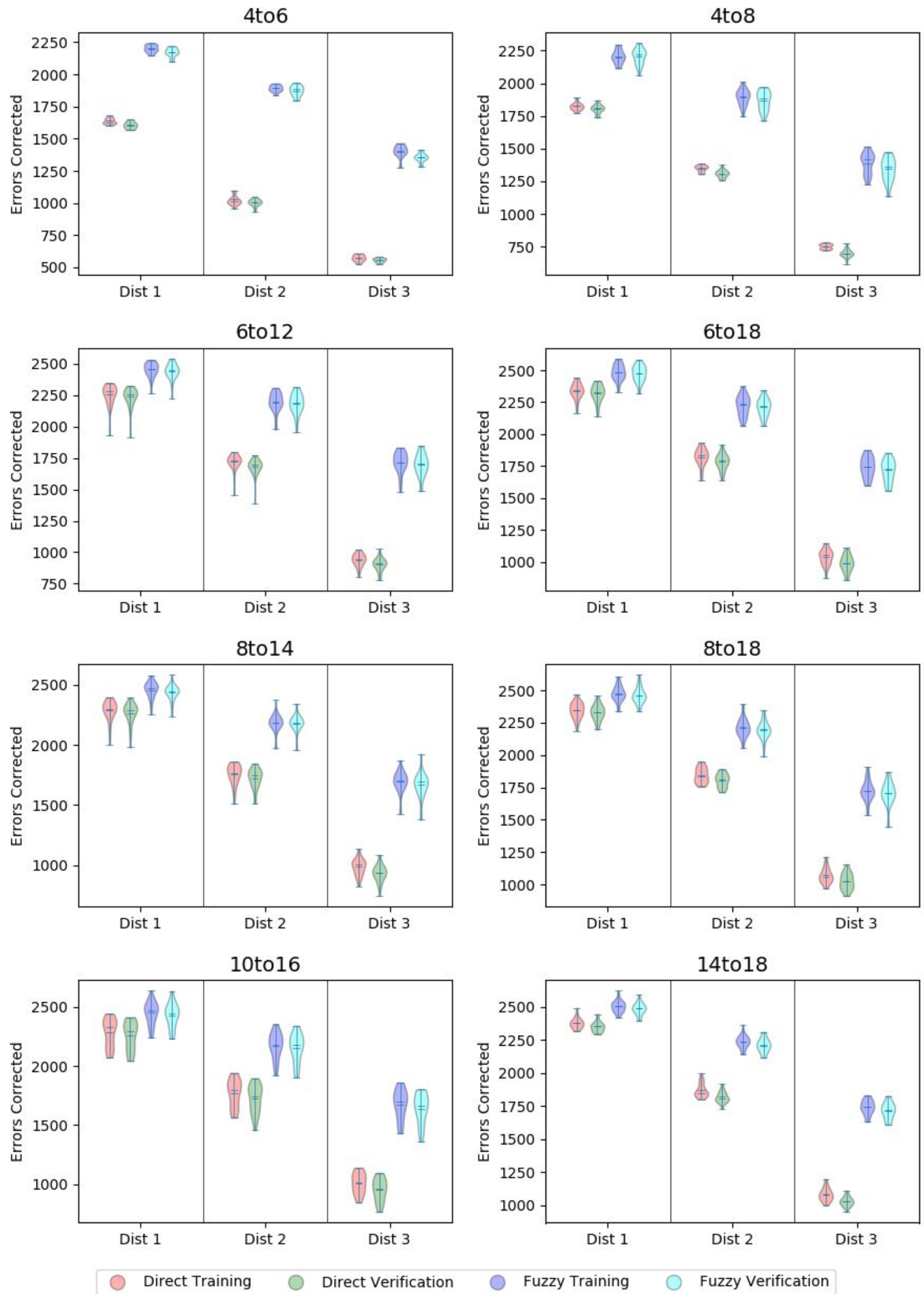

Figure B.30: code205-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3333	3230	74	38.7	3231	3181	106	37.5
	1	1679	1639	32	58.5	1651	1615.5	53	57.5
	2	1097	1015.5	26	38.2	1040	995	32	36.2
	3	609	572	28	21.2	572	562	22	19.9
4to8	All	4005	3915.5	68	46.5	3939	3846	81	45.7
	1	1881	1835	29	65.5	1871	1813	45	65.2
	2	1402	1348.5	31	48.9	1374	1317	46	47.9
	3	781	740	35	27.2	748	704	20	26.1
6to12	All	5085	4843.5	286	59.1	5058	4735	301	58.7
	1	2343	2210	121	81.6	2312	2211	130	80.6
	2	1790	1695	84	62.4	1781	1641.5	95	62.1
	3	1053	917	77	36.7	1009	857	54	35.2
6to18	All	**5582**	5289	183	64.8	**5470**	5149	170	63.5
	1	2493	2363	44	86.9	2447	2333	57	85.3
	2	1958	1853.5	75	68.2	1940	1807	86	67.6
	3	1187	1062.5	86	41.4	1129	1010.5	57	39.3
8to14	All	5306	5021	317	61.6	5247	4960	343	60.9
	1	2389	2289	110	83.2	2383	2274.5	102	83
	2	1869	1756	136	65.1	1859	1723	114	64.8
	3	1103	999	120	38.4	1089	953	157	37.9
8to18	All	5507	5236	238	64	5451	5131	301	63.3
	1	2440	2371.5	56	85	2451	2345.5	42	85.4
	2	1962	1835	105	68.4	1925	1788.5	105	67.1
	3	1164	1044.5	97	40.6	1127	993.5	128	39.3
10to16	All	5439	5250	367	63.2	5369	5101.5	307	62.4
	1	2432	2356.5	128	84.7	2431	2320.5	86	84.7
	2	1924	1830.5	134	67	1894	1788	133	66
	3	1119	1033.5	117	39	1089	995.5	102	37.9
14to18	All	5521	**5311.5**	155	64.1	5403	**5155.5**	159	62.8
	1	2444	2361	59	85.2	2432	2333.5	67	84.7
	2	1969	1852.5	54	68.6	1902	1810.5	66	66.3
	3	1141	1073	70	39.8	1138	1011	73	39.7

Table B.61: Code205-1, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5714	5530.5	123	66.4	5645	5417	135	65.6
	1	2248	2218	44	78.3	2260	2203	46	78.7
	2	1987	1902	45	69.2	1966	1872.5	74	68.5
	3	1490	1415	66	51.9	1422	1358	54	49.5
4to8	All	5851	5538.5	311	68	5845	5481	279	67.9
	1	2306	2206.5	78	80.3	2304	2216	68	80.3
	2	2035	1927.5	98	70.9	2020	1900.5	94	70.4
	3	1516	1391.5	124	52.8	1525	1370.5	149	53.1
6to12	All	6696	6290.5	425	77.8	6599	6216.5	429	76.6
	1	2532	2427.5	106	88.2	2492	2420.5	110	86.8
	2	2316	2177	139	80.7	2279	2158	151	79.4
	3	1848	1672	161	64.4	1838	1658	157	64
6to18	All	**6940**	**6480**	218	80.6	**6888**	**6419**	193	80
	1	2626	2504.5	71	91.5	2597	2489.5	61	90.5
	2	2394	2239.5	58	83.4	2376	2214	87	82.8
	3	1920	1734.5	98	66.9	1922	1709.5	90	67
8to14	All	6702	6339.5	320	77.8	6621	6288	268	76.9
	1	2556	2449	69	89.1	2518	2440	85	87.7
	2	2327	2188	112	81.1	2294	2175	93	79.9
	3	1840	1703	114	64.1	1824	1681	84	63.6
8to18	All	6787	6480	137	78.8	6714	6412.5	186	78
	1	2581	2504.5	53	89.9	2566	2478	61	89.4
	2	2335	2237.5	57	81.4	2338	2215	52	81.5
	3	1879	1748	71	65.5	1861	1720.5	46	64.8
10to16	All	6646	6435	220	77.2	6604	6363	252	76.7
	1	2584	2491.5	70	90	2586	2463.5	56	90.1
	2	2322	2212	66	80.9	2288	2196.5	73	79.7
	3	1831	1722.5	87	63.8	1802	1700.5	105	62.8
14to18	All	6831	6467	234	79.3	6754	6408	284	78.4
	1	2608	2503	62	90.9	2593	2489	80	90.3
	2	2369	2232.5	91	82.5	2336	2207.5	94	81.4
	3	1898	1735	74	66.1	1870	1704.5	121	65.2

Table B.62: Code205-1, Fuzzy Classification Fitness Result For Experiment 3

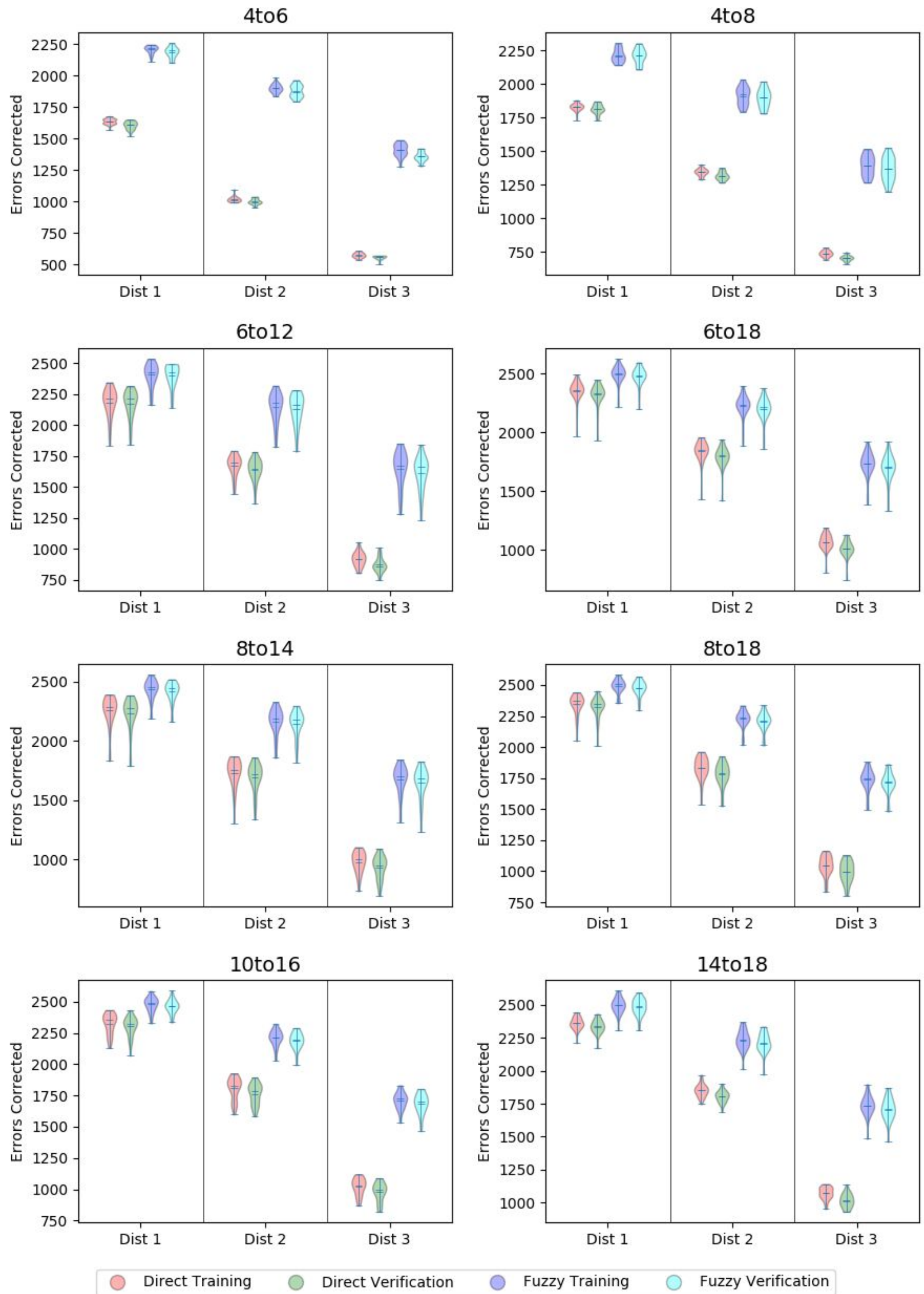

Figure B.31: code205-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3333	3224	107	38.7	3231	3163	79	37.5
	1	1679	1627	50	58.5	1651	1615.5	34	57.5
	2	1097	1025	32	38.2	1044	993	26	36.4
	3	609	572	32	21.2	589	559.5	28	20.5
4to8	All	3991	3920.5	65	46.4	3939	3806	59	45.7
	1	1904	1837	22	66.3	1871	1808.5	43	65.2
	2	1402	1340	32	48.9	1377	1311.5	61	48
	3	775	744	32	27	734	697.5	37	25.6
6to12	All	5052	4844.5	238	58.7	5040	4767	221	58.5
	1	2344	2238.5	101	81.7	2329	2219.5	124	81.1
	2	1795	1691.5	70	62.5	1741	1668	63	60.7
	3	981	917	48	34.2	970	882	54	33.8
6to18	All	5454	5261	286	63.3	5347	5131.5	381	62.1
	1	2441	2344.5	83	85.1	2393	2324	94	83.4
	2	1934	1841.5	100	67.4	1884	1786	132	65.6
	3	1152	1049	112	40.1	1103	1003	105	38.4
8to14	All	5343	5131	381	62.1	5260	4974	286	61.1
	1	2386	2317	117	83.1	2367	2289.5	141	82.5
	2	1886	1786	147	65.7	1836	1740.5	103	64
	3	1102	1011	116	38.4	1076	940	95	37.5
8to18	All	5509	5219.5	286	64	5362	5122.5	320	62.3
	1	2465	2338.5	90	85.9	2426	2330.5	84	84.5
	2	1942	1828	107	67.7	1885	1798	127	65.7
	3	1185	1046.5	133	41.3	1112	1010	124	38.7
10to16	All	5422	5105.5	319	63	5327	5023	305	61.9
	1	2432	2313.5	94	84.7	2424	2295.5	92	84.5
	2	1905	1797.5	105	66.4	1856	1746.5	117	64.7
	3	1091	1008	110	38	1095	984	110	38.2
14to18	All	**5525**	**5304**	119	64.2	**5403**	**5177.5**	126	62.8
	1	2449	2376	45	85.3	2431	2356.5	61	84.7
	2	1952	1865.5	48	68	1886	1810	50	65.7
	3	1154	1061.5	69	40.2	1116	1019.5	47	38.9

Table B.63: Code205-1, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5619	5530.5	146	65.3	5575	5417	137	64.8
	1	2246	2218	64	78.3	2235	2203	46	77.9
	2	1949	1902	47	67.9	1940	1857	63	67.6
	3	1454	1406	57	50.7	1420	1358	39	49.5
4to8	All	5826	5568.5	148	67.7	5744	5520	234	66.7
	1	2309	2213	50	80.5	2289	2222.5	47	79.8
	2	2013	1908.5	66	70.1	1971	1921	80	68.7
	3	1517	1425.5	63	52.9	1508	1382	124	52.5
6to12	All	6681	6393.5	446	77.6	6672	6273	420	77.5
	1	2530	2457	103	88.2	2527	2437	113	88
	2	2300	2190	148	80.1	2319	2177	133	80.8
	3	1865	1719.5	179	65	1826	1660.5	178	63.6
6to18	All	6716	6354	202	78	6686	6301.5	231	77.7
	1	2589	2478.5	73	90.2	2577	2462	61	89.8
	2	2319	2197	73	80.8	2298	2181	82	80.1
	3	1848	1690	78	64.4	1852	1672.5	94	64.5
8to14	All	6714	6426	270	78	6590	6371	321	76.5
	1	2561	2485	113	89.2	2572	2460.5	137	89.6
	2	2330	2200.5	79	81.2	2280	2194	114	79.4
	3	1823	1726	103	63.5	1795	1692.5	160	62.5
8to18	All	6764	6428	244	78.6	6699	6381.5	225	77.8
	1	2597	2475.5	70	90.5	2597	2464	84	90.5
	2	2349	2221	82	81.8	2337	2205.5	64	81.4
	3	1885	1732	81	65.7	1841	1714	67	64.1
10to16	All	**6852**	6347	193	79.6	**6817**	6312.5	216	79.2
	1	2622	2462	81	91.4	2606	2456	66	90.8
	2	2394	2197.5	71	83.4	2345	2188.5	94	81.7
	3	1836	1694	80	64	1879	1688.5	97	65.5
14to18	All	6739	**6488**	198	78.3	6714	**6405**	232	78
	1	2577	2499	44	89.8	2583	2493	64	90
	2	2333	2252	76	81.3	2327	2225.5	78	81.1
	3	1847	1745.5	70	64.4	1828	1706.5	89	63.7

Table B.64: Code205-1, Fuzzy Classification Fitness Result For Experiment 4

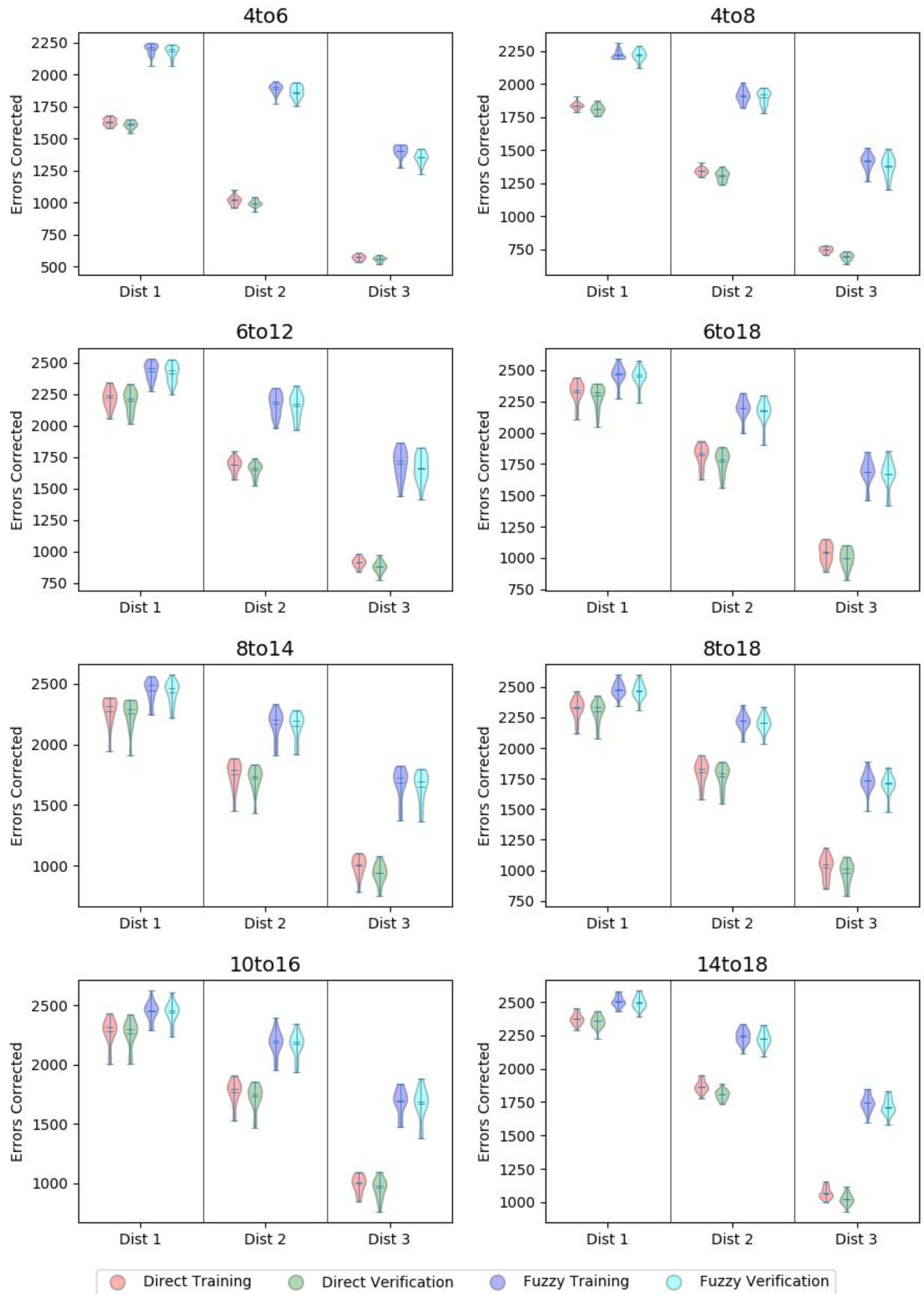

Figure B.32: code205-1, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

Code205-2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3212	3144	61	37.3	3301	3070	68	38.3
	1	1611	1576	41	56.1	1624	1566	30	56.6
	2	1053	993	41	36.7	1088	980.5	25	37.9
	3	584	558	28	20.3	589	528	19	20.5
4to8	All	3949	3898	74	45.9	3942	3824	59	45.8
	1	1869	1825	40	65.1	1881	1818	48	65.5
	2	1396	1344.5	39	48.6	1415	1326.5	49	49.3
	3	779	716.5	35	27.1	712	682.5	34	24.8
6to12	All	5081	4975.5	196	59	5064	4887	232	58.8
	1	2319	2246	77	80.8	2307	2237.5	75	80.4
	2	1818	1742.5	66	63.3	1806	1725.5	74	62.9
	3	1049	957	86	36.6	1005	928.5	71	35
6to18	All	**5528**	**5321**	167	64.2	**5477**	**5220.5**	118	63.6
	1	2429	2367	58	84.6	2435	2348	76	84.8
	2	1978	1881.5	66	68.9	1946	1838	53	67.8
	3	1171	1083.5	76	40.8	1123	1024.5	56	39.1
8to14	All	5316	5080	219	61.7	5228	4959	279	60.7
	1	2400	2307	67	83.6	2377	2288.5	88	82.8
	2	1877	1785	90	65.4	1850	1757	94	64.5
	3	1110	995.5	77	38.7	1067	956	122	37.2
8to18	All	5486	5226.5	212	63.7	5376	5137	148	62.4
	1	2446	2327	65	85.2	2409	2315	60	83.9
	2	1965	1840.5	79	68.5	1924	1805.5	58	67
	3	1156	1025.5	86	40.3	1090	988.5	76	38
10to16	All	5474	5223.5	232	63.6	5416	5109	286	62.9
	1	2423	2329	103	84.4	2408	2307	104	83.9
	2	1951	1842.5	94	68	1883	1824	119	65.6
	3	1130	1041.5	64	39.4	1146	1001	92	39.9
14to18	All	5526	5273.5	159	64.2	5419	5152.5	200	62.9
	1	2435	2359.5	62	84.8	2434	2343.5	64	84.8
	2	1928	1865	79	67.2	1943	1828	68	67.7
	3	1225	1069.5	52	42.7	1137	1004	72	39.6

Table B.65: Code205-2, Direct Classification Fitness Result For Experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5707	5427	225	66.3	5705	5391.5	264	66.3
	1	2232	2148	63	77.8	2255	2164	78	78.6
	2	1985	1900	74	69.2	1994	1890.5	83	69.5
	3	1516	1370.5	90	52.8	1474	1352	83	51.4
4to8	All	5927	5595.5	322	68.8	5870	5582	279	68.2
	1	2295	2224	70	80	2319	2248	83	80.8
	2	2062	1952	120	71.8	2074	1946	113	72.3
	3	1582	1410.5	152	55.1	1512	1390.5	128	52.7
6to12	All	6610	6265.5	242	76.8	6590	6244.5	307	76.5
	1	2516	2402	61	87.7	2512	2404.5	87	87.5
	2	2300	2174.5	93	80.1	2276	2164.5	83	79.3
	3	1814	1689	101	63.2	1802	1657	133	62.8
6to18	All	6874	**6489.5**	292	79.8	6848	6399	295	79.5
	1	2615	2495.5	68	91.1	2617	2495	85	91.2
	2	2404	2250	105	83.8	2397	2204	102	83.5
	3	1908	1733.5	114	66.5	1853	1705.5	122	64.6
8to14	All	**6993**	6447.5	296	81.2	6899	**6436**	334	80.1
	1	2630	2478	93	91.6	2610	2470	72	90.9
	2	2404	2239	93	83.8	2398	2217.5	104	83.6
	3	1959	1732	138	68.3	1891	1723	133	65.9
8to18	All	6866	6459	399	79.7	**6901**	6397.5	330	80.2
	1	2592	2477	82	90.3	2597	2479.5	87	90.5
	2	2393	2232	89	83.4	2374	2228	117	82.7
	3	1906	1725	115	66.4	1930	1691.5	110	67.2
10to16	All	6846	6406.5	314	79.5	6794	6362	322	78.9
	1	2575	2479	94	89.7	2583	2467	81	90
	2	2355	2233	94	82.1	2352	2198	111	82
	3	1942	1694	114	67.7	1908	1684.5	122	66.5
14to18	All	6775	6469	294	78.7	6689	6404	315	77.7
	1	2563	2491.5	83	89.3	2580	2482	60	89.9
	2	2344	2243.5	78	81.7	2329	2234.5	97	81.1
	3	1880	1739.5	136	65.5	1817	1686	159	63.3

Table B.66: Code205-2, Fuzzy Classification Fitness Result For Experiment 1

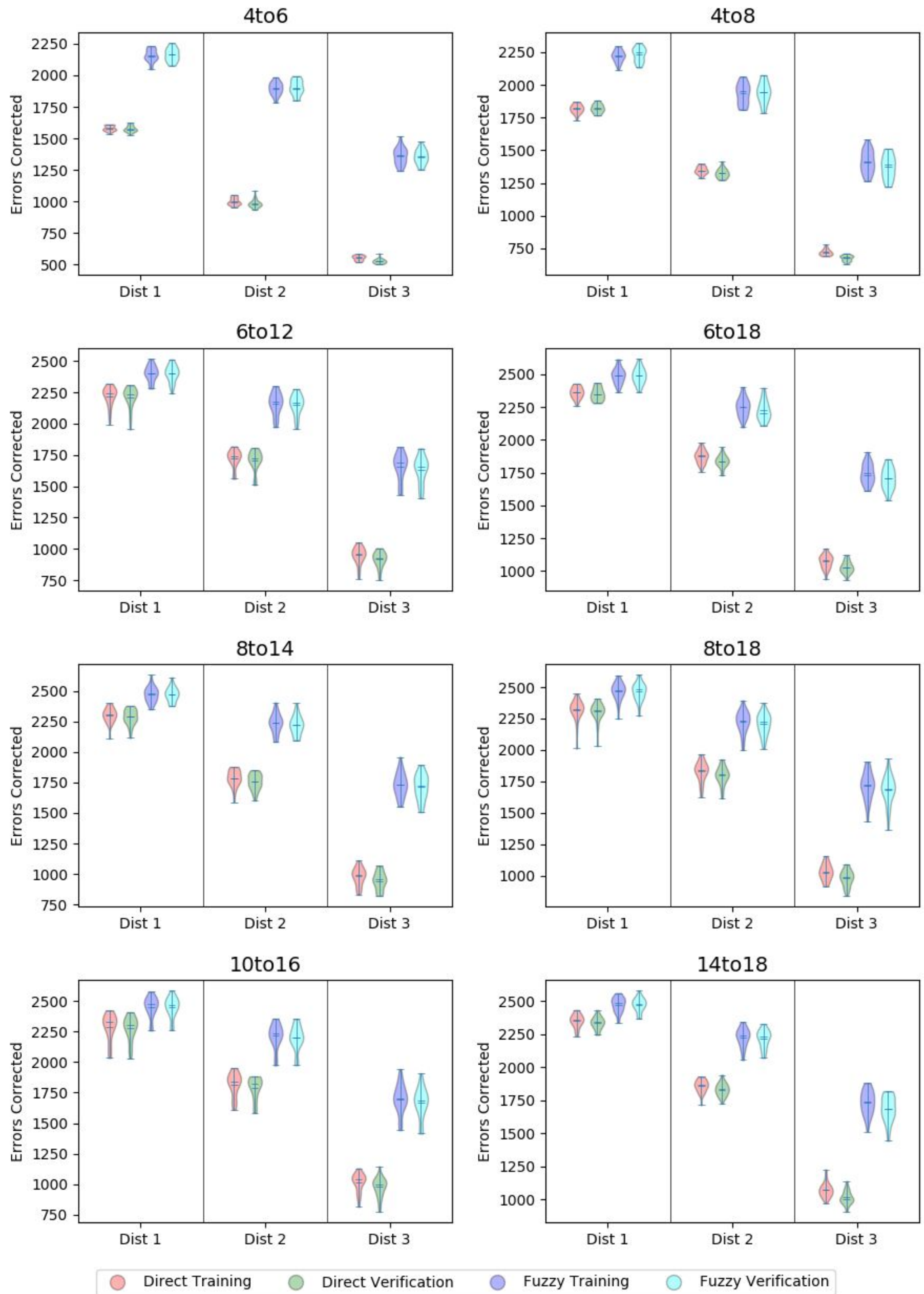

Figure B.33: code205-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 1

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3212	3147.5	69	37.3	3301	3106	97	38.3
	1	1611	1584.5	35	56.1	1644	1593	44	57.3
	2	1053	1005	55	36.7	1088	990	54	37.9
	3	602	558	40	21	589	526	28	20.5
4to8	All	3976	3934	60	46.2	3957	3858.5	53	46
	1	1879	1834	65	65.5	1881	1826.5	28	65.5
	2	1406	1363	40	49	1415	1348	46	49.3
	3	771	721	35	26.9	718	686	26	25
6to12	All	5055	4947.5	171	58.7	5066	4872.5	209	58.8
	1	2307	2237.5	82	80.4	2309	2237.5	61	80.5
	2	1809	1740	70	63	1803	1710	74	62.8
	3	1037	942	58	36.1	980	916	67	34.1
6to18	All	5606	5155	266	65.1	5487	5093	237	63.7
	1	2484	2341.5	98	86.6	2462	2325	73	85.8
	2	1981	1818	111	69	1967	1787.5	100	68.5
	3	1158	1001.5	99	40.3	1091	971	91	38
8to14	All	5326	5166.5	263	61.9	5308	5080.5	286	61.6
	1	2399	2309	85	83.6	2403	2302	86	83.7
	2	1912	1830	113	66.6	1896	1804	113	66.1
	3	1125	1012	95	39.2	1071	972.5	107	37.3
8to18	All	**5719**	5163	292	66.4	**5519**	5073.5	280	64.1
	1	2489	2325	100	86.7	2458	2316	107	85.6
	2	2023	1819.5	104	70.5	1972	1781	78	68.7
	3	1207	1014	75	42.1	1105	975	111	38.5
10to16	All	5388	5028.5	275	62.6	5331	4945	317	61.9
	1	2413	2280	127	84.1	2401	2249	146	83.7
	2	1904	1770	84	66.3	1916	1749	93	66.8
	3	1112	974	77	38.7	1060	945	109	36.9
14to18	All	5589	**5285.5**	172	64.9	5482	**5178**	171	63.7
	1	2438	2368	44	84.9	2429	2346	59	84.6
	2	1966	1861.5	66	68.5	1964	1826.5	53	68.4
	3	1201	1066	75	41.8	1134	1022	63	39.5

Table B.67: Code205-2, Direct Classification Fitness Result For Experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5654	5464	225	65.7	5654	5447	264	65.7
	1	2232	2188	56	77.8	2245	2170	85	78.2
	2	1985	1935	77	69.2	1985	1918	92	69.2
	3	1476	1374	94	51.4	1429	1359	107	49.8
4to8	All	5910	5547	277	68.6	5901	5523	350	68.5
	1	2298	2221.5	93	80.1	2325	2225	91	81
	2	2072	1945	99	72.2	2092	1935	119	72.9
	3	1569	1396.5	134	54.7	1524	1379	126	53.1
6to12	All	6690	6256.5	250	77.7	6628	6207.5	233	77
	1	2522	2415	60	87.9	2517	2419	68	87.7
	2	2314	2182.5	118	80.6	2300	2165	116	80.1
	3	1854	1665	101	64.6	1811	1620.5	83	63.1
6to18	All	6790	**6504.5**	315	78.9	6761	**6443.5**	342	78.5
	1	2598	2487	79	90.5	2575	2486	79	89.7
	2	2353	2248	108	82	2351	2249.5	120	81.9
	3	1908	1742.5	133	66.5	1887	1721	142	65.7
8to14	All	6880	6421.5	246	79.9	6840	6358.5	252	79.4
	1	2592	2471.5	63	90.3	2605	2469	82	90.8
	2	2391	2229	89	83.3	2359	2219	90	82.2
	3	1918	1719	124	66.8	1897	1695	133	66.1
8to18	All	6877	6415.5	336	79.9	6795	6360	303	78.9
	1	2604	2461	114	90.7	2598	2472	85	90.5
	2	2390	2235.5	110	83.3	2371	2220.5	119	82.6
	3	1920	1716	145	66.9	1860	1681	161	64.8
10to16	All	6797	6336.5	446	78.9	6777	6297.5	427	78.7
	1	2557	2441.5	124	89.1	2577	2443	115	89.8
	2	2355	2197	147	82.1	2340	2215	127	81.5
	3	1885	1701.5	156	65.7	1860	1655	130	64.8
14to18	All	**6900**	6453	170	80.1	**6867**	6379.5	168	79.8
	1	2589	2480	55	90.2	2573	2482.5	59	89.7
	2	2380	2244.5	56	82.9	2378	2223.5	77	82.9
	3	1931	1743.5	71	67.3	1921	1691	61	66.9

Table B.68: Code205-2, Fuzzy Classification Fitness Result For Experiment 2

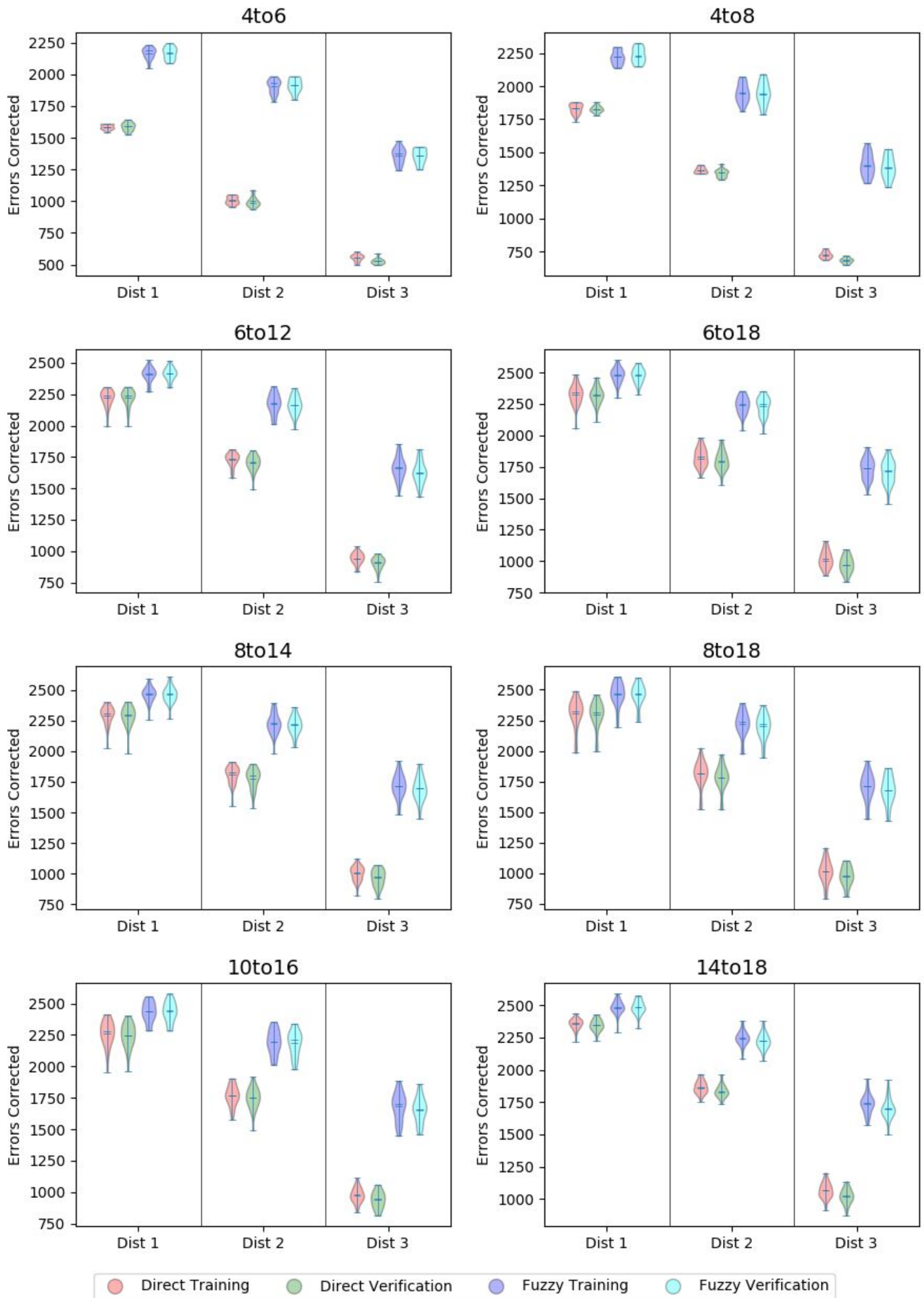

Figure B.34: code205-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 2

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3212	3194.5	82	37.3	3301	3130	51	38.3
	1	1611	1594.5	31	56.1	1624	1593.5	30	56.6
	2	1053	1041	25	36.7	1088	990	59	37.9
	3	584	558	45	20.3	589	543	34	20.5
4to8	All	3955	3888.5	55	45.9	3925	3848	88	45.6
	1	1868	1826	33	65.1	1863	1816	27	64.9
	2	1398	1344	38	48.7	1396	1325	38	48.6
	3	755	724.5	27	26.3	747	688	31	26
6to12	All	5037	4848	280	58.5	4995	4799.5	334	58
	1	2278	2206.5	125	79.4	2292	2202.5	109	79.9
	2	1810	1703	117	63.1	1766	1688	130	61.5
	3	1005	917	61	35	959	896	97	33.4
6to18	All	5490	5079	493	63.8	5392	5063	451	62.6
	1	2460	2297	132	85.7	2415	2283	148	84.1
	2	1935	1792	192	67.4	1910	1801	168	66.6
	3	1144	996	171	39.9	1092	961	140	38
8to14	All	5327	5111	310	61.9	5310	4989	276	61.7
	1	2375	2296	99	82.8	2374	2281	119	82.7
	2	1906	1798	104	66.4	1885	1753	106	65.7
	3	1125	979.5	114	39.2	1075	938	74	37.5
8to18	All	**5566**	5229	256	64.6	5417	5178	240	62.9
	1	2497	2345	90	87	2462	2328.5	78	85.8
	2	1939	1856.5	85	67.6	1937	1832	98	67.5
	3	1173	1037	97	40.9	1133	1007	100	39.5
10to16	All	5539	5223.5	188	64.3	5462	5124	214	63.4
	1	2453	2329.5	55	85.5	2419	2310	77	84.3
	2	1935	1832.5	86	67.4	1945	1804	73	67.8
	3	1233	1044	76	43	1159	1002	102	40.4
14to18	All	5534	**5300.5**	146	64.3	**5521**	**5215.5**	138	64.1
	1	2445	2363.5	55	85.2	2443	2351	56	85.1
	2	1972	1876	59	68.7	1945	1836.5	59	67.8
	3	1150	1060	68	40.1	1133	1017	88	39.5

Table B.69: Code205-2, Direct Classification Fitness Result For Experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5654	5544	130	65.7	5654	5593	164	65.7
	1	2232	2195	82	77.8	2245	2207	50	78.2
	2	1985	1935	3	69.2	1985	1971	63	69.2
	3	1476	1413	96	51.4	1428	1410.5	63	49.8
4to8	All	5794	5473	248	67.3	5869	5462	328	68.2
	1	2281	2201	59	79.5	2306	2204	71	80.3
	2	2048	1898.5	88	71.4	2042	1899	102	71.1
	3	1500	1368	103	52.3	1526	1342.5	98	53.2
6to12	All	6608	6204	368	76.7	6517	6168	407	75.7
	1	2520	2404.5	78	87.8	2521	2408	76	87.8
	2	2307	2163.5	110	80.4	2258	2139	153	78.7
	3	1806	1655	145	62.9	1764	1647.5	179	61.5
6to18	All	6755	6384	259	78.5	6676	6381.5	251	77.5
	1	2647	2463.5	82	92.2	2604	2461.5	89	90.7
	2	2346	2218	103	81.7	2338	2227	94	81.5
	3	1848	1717	89	64.4	1841	1690	94	64.1
8to14	All	6968	6360	275	80.9	6928	6335.5	244	80.5
	1	2600	2447.5	71	90.6	2600	2449	57	90.6
	2	2410	2217.5	103	84	2406	2204.5	110	83.8
	3	1958	1704	101	68.2	1922	1684.5	108	67
8to18	All	**7093**	6404.5	255	82.4	**7076**	6359.5	235	82.2
	1	2651	2459	89	92.4	2661	2454	80	92.7
	2	2461	2213.5	92	85.7	2421	2210.5	77	84.4
	3	1981	1709	127	69	1994	1705	99	69.5
10to16	All	7016	6359	288	81.5	6948	6321	288	80.7
	1	2632	2462.5	62	91.7	2625	2454	95	91.5
	2	2432	2211.5	106	84.7	2426	2191	81	84.5
	3	1952	1699	115	68	1897	1678	93	66.1
14to18	All	6873	**6626**	349	79.8	6854	**6579.5**	372	79.6
	1	2609	2522	91	90.9	2600	2518.5	80	90.6
	2	2390	2282.5	132	83.3	2386	2280	135	83.1
	3	1947	1805	138	67.8	1911	1771.5	134	66.6

Table B.70: Code205-2, Fuzzy Classification Fitness Result For Experiment 3

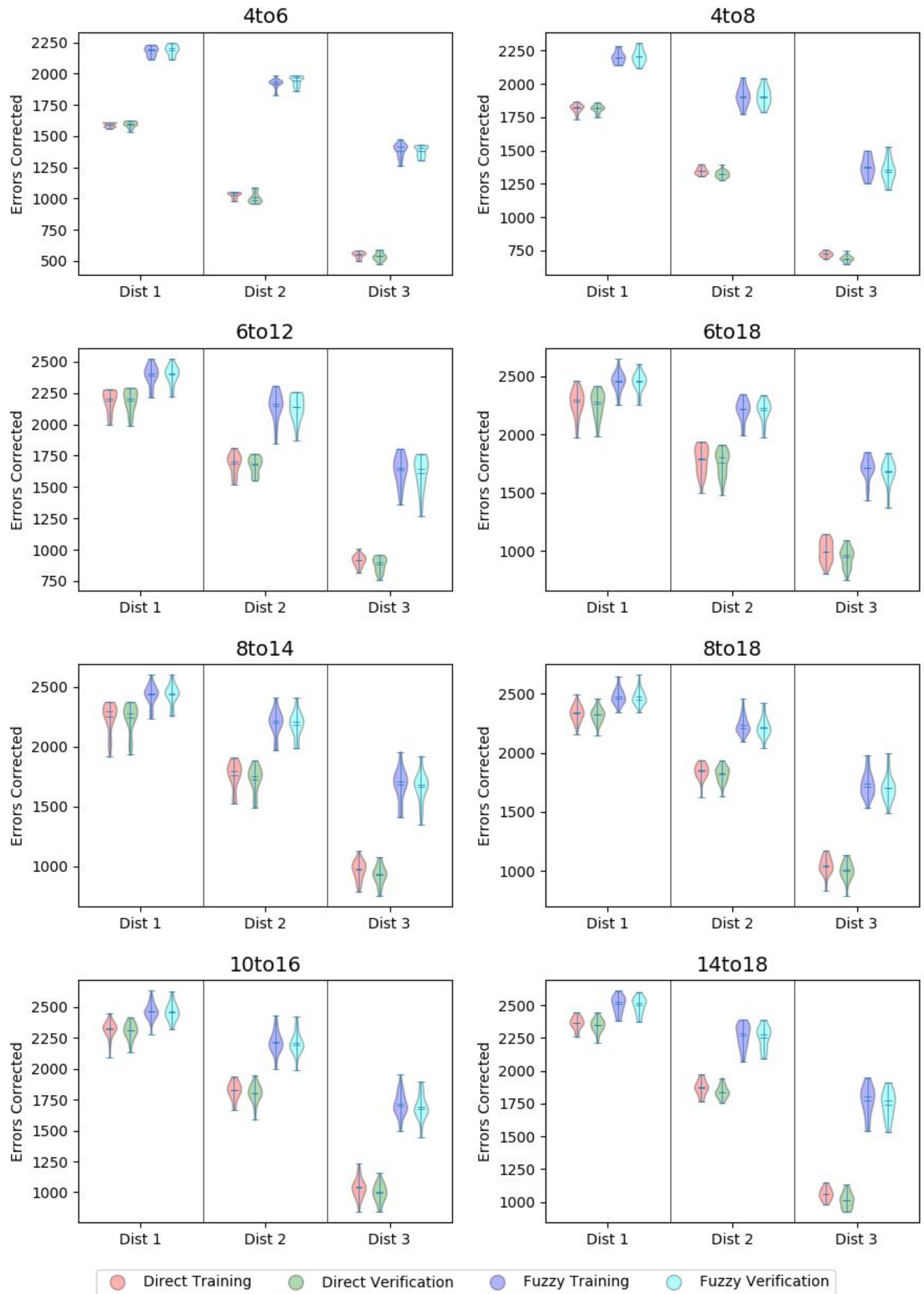

Figure B.35: code205-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 3

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	3212	3157	77	37.3	3301	3126.5	135	38.3
	1	1611	1581.5	34	56.1	1644	1598.5	30	57.3
	2	1053	1021	48	36.7	1088	990	113	37.9
	3	602	558	32	21	589	533.5	33	20.5
4to8	All	3974	3922	45	46.2	3957	3830.5	74	46
	1	1868	1837.5	39	65.1	1882	1837	64	65.6
	2	1406	1362	45	49	1415	1340.5	38	49.3
	3	771	719.5	32	26.9	732	661	30	25.5
6to12	All	5081	4918.5	157	59	5010	4822	125	58.2
	1	2331	2246.5	75	81.2	2367	2219.5	68	82.5
	2	1795	1732	56	62.5	1760	1701	53	61.3
	3	1005	935	71	35	1001	897	43	34.9
6to18	All	**5593**	5185.5	499	65	**5522**	5143.5	549	64.1
	1	2477	2316	175	86.3	2476	2315	155	86.3
	2	1968	1833	208	68.6	1967	1801.5	161	68.5
	3	1148	1027	159	40	1114	1009.5	142	38.8
8to14	All	5280	5078.5	198	61.3	5242	5033	245	60.9
	1	2395	2290	89	83.4	2400	2303	71	83.6
	2	1880	1802	63	65.5	1860	1784	109	64.8
	3	1093	993.5	83	38.1	1045	974	86	36.4
8to18	All	5557	**5267.5**	303	64.5	5422	5186.5	242	63
	1	2517	2336.5	88	87.7	2473	2346	76	86.2
	2	1969	1853.5	75	68.6	1898	1819.5	94	66.1
	3	1163	1047.5	100	40.5	1107	992	82	38.6
10to16	All	5482	5163	483	63.7	5360	5077.5	456	62.3
	1	2436	2320	191	84.9	2427	2307	166	84.6
	2	1944	1810.5	168	67.7	1903	1786.5	152	66.3
	3	1161	1028	134	40.5	1102	978.5	122	38.4
14to18	All	5502	5266.5	167	63.9	5438	**5206**	135	63.2
	1	2440	2345.5	76	85	2440	2344.5	85	85
	2	1956	1857	59	68.2	1928	1827.5	45	67.2
	3	1128	1049	57	39.3	1112	1014	58	38.7

Table B.71: Code205-2, Direct Classification Fitness Result For Experiment 4

State Range	Error Dist.	Training Max	Training Median	Training IQR	Training Max %	Verification Max	Verification Median	Verification IQR	Verification Max %
4to6	All	5654	5476	120	65.7	5654	5522.5	207	65.7
	1	2232	2193	46	77.8	2245	2188	59	78.2
	2	1985	1935	42	69.2	1985	1940.5	57	69.2
	3	1476	1381	82	51.4	1429	1381.5	98	49.8
4to8	All	6139	5613	372	71.3	6076	5586	366	70.6
	1	2328	2226	85	81.1	2366	2219.5	116	82.4
	2	2154	1944	123	75.1	2123	1951	109	74
	3	1657	1419	175	57.7	1587	1378.5	122	55.3
6to12	All	6895	6355	244	80.1	**6933**	6311	228	80.5
	1	2575	2443	85	89.7	2593	2434	76	90.3
	2	2415	2204.5	93	84.1	2402	2189.5	97	83.7
	3	1905	1701	99	66.4	1938	1669	108	67.5
6to18	All	6900	6445	293	80.1	6880	6435	256	79.9
	1	2579	2475.5	92	89.9	2600	2477	81	90.6
	2	2417	2247.5	105	84.2	2351	2227.5	101	81.9
	3	1939	1737.5	126	67.6	1929	1726	112	67.2
8to14	All	6733	6400.5	282	78.2	6749	6370	300	78.4
	1	2559	2454	63	89.2	2586	2457	67	90.1
	2	2347	2225.5	77	81.8	2336	2212	92	81.4
	3	1827	1711.5	147	63.7	1827	1678	127	63.7
8to18	All	6819	6477	231	79.2	6786	6425.5	287	78.8
	1	2634	2486	83	91.8	2629	2494.5	69	91.6
	2	2396	2255.5	105	83.5	2363	2227	117	82.3
	3	1885	1734	112	65.7	1871	1697	149	65.2
10to16	All	6842	6400	289	79.5	6793	6328.5	360	78.9
	1	2586	2462	118	90.1	2579	2452.5	98	89.9
	2	2349	2210.5	148	81.8	2329	2189	109	81.1
	3	1922	1727.5	70	67	1885	1690	137	65.7
14to18	All	**6952**	**6538.5**	194	80.7	6918	**6455**	170	80.3
	1	2623	2486	66	91.4	2613	2485.5	72	91
	2	2408	2261	90	83.9	2399	2235.5	70	83.6
	3	1925	1751.5	96	67.1	1928	1739.5	99	67.2

Table B.72: Code205-2, Fuzzy Classification Fitness Result For Experiment 4

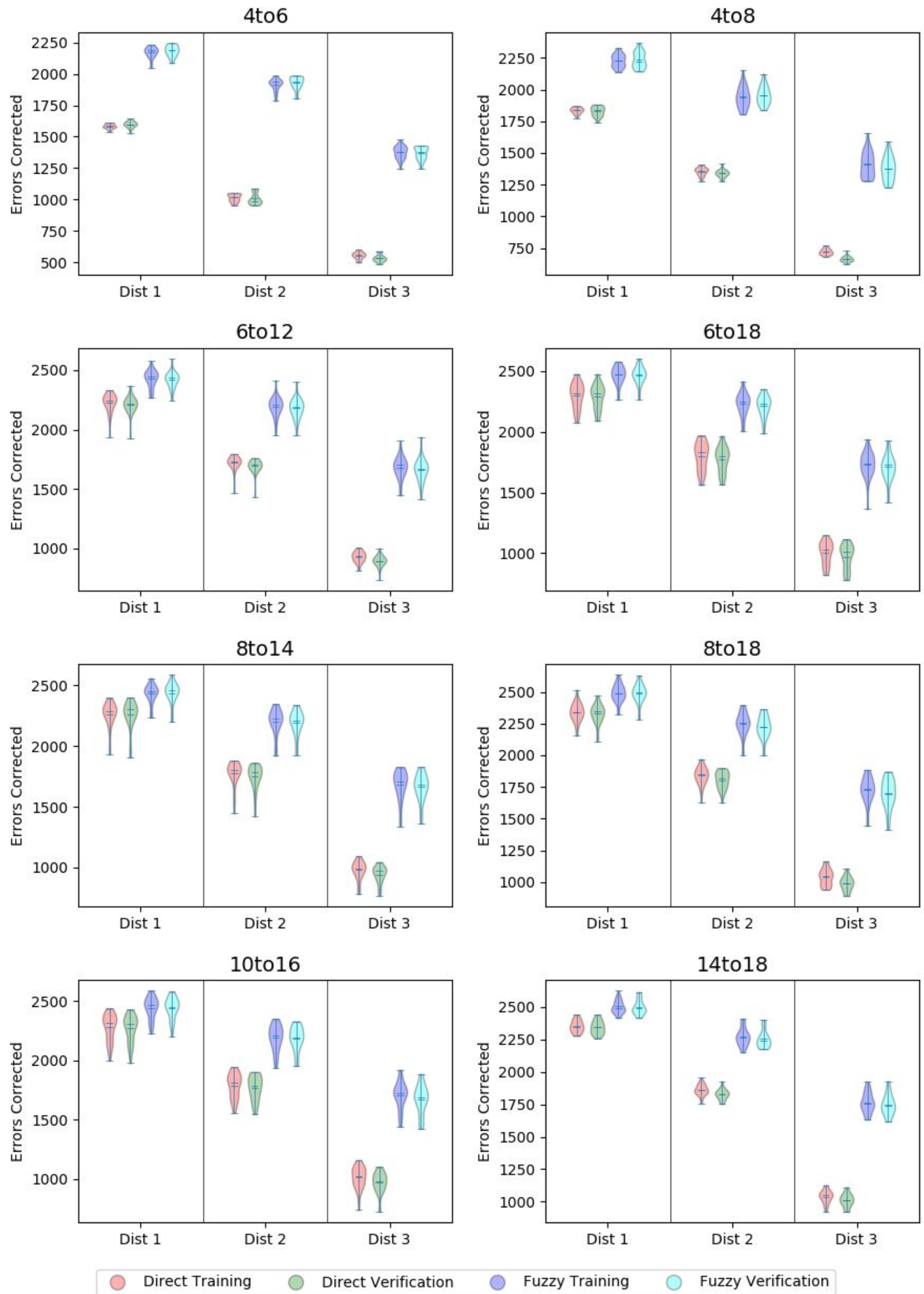

Figure B.36: code205-2, violin plots representing the distribution of correctly decoded error patterns for different ranges of states for the 30 runs for experiment 4

B.2 Difference between Total and Visited number of states

B.2.1 Codes of Length 10

Code17-1

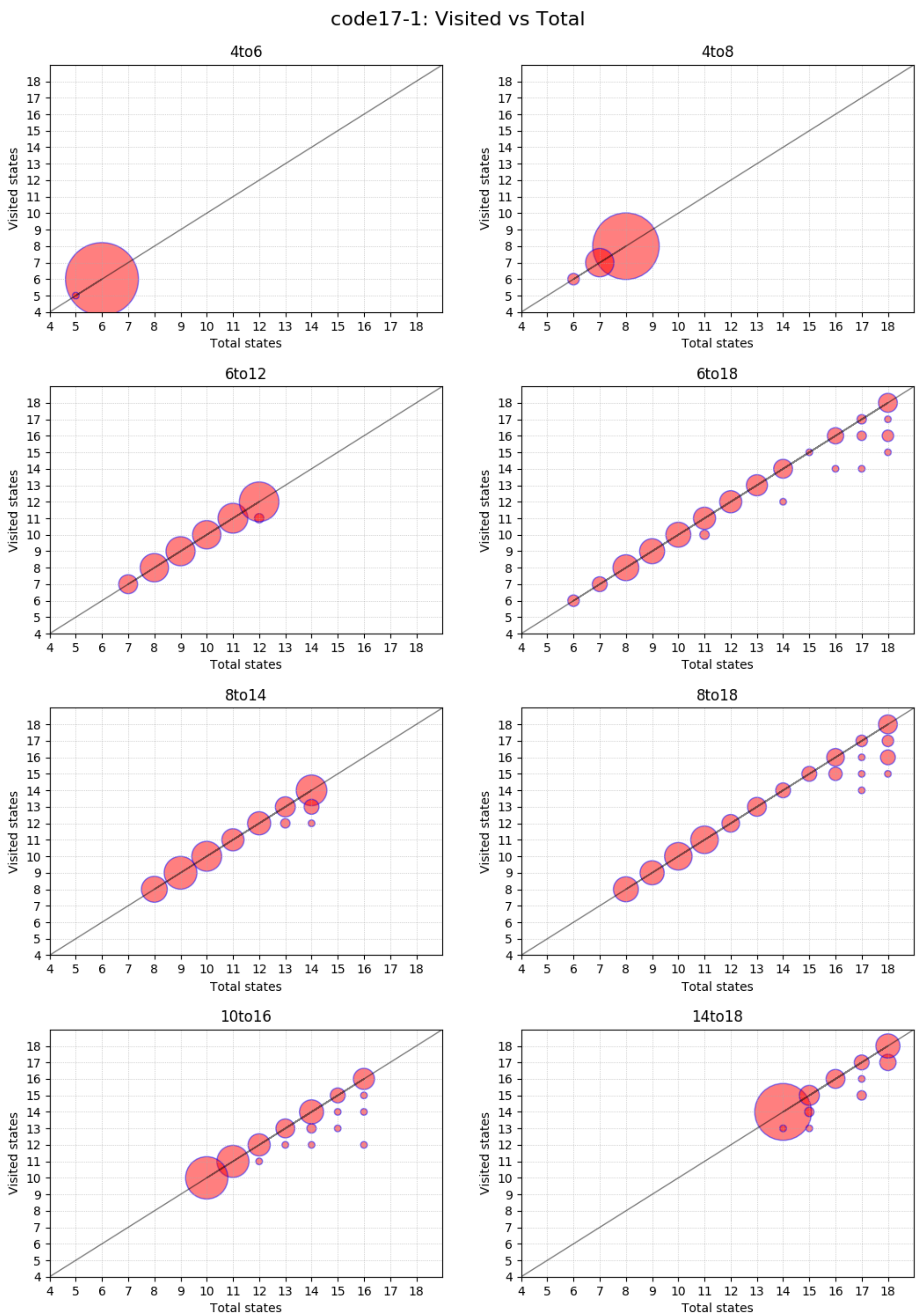

Figure B.37: The final SEMs machine size (total states) against the number of visited states across all experiments for code17-1

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	10.8	12	12	3	9.7	12	9.5	3
6to12	Visited	10.7	12	11.5	3	9.7	12	9.5	3
6to18	Total	13.2	18	13	8	11.6	18	10	4
6to18	Visited	12.7	18	13	7	11.4	18	10	4
8to14	Total	11.3	14	11.5	5	10.7	14	11	3
8to14	Visited	11	14	10.5	4	10.5	14	11	3
8to18	Total	12.6	18	12	8	12.3	18	12	5
8to18	Visited	12	18	12	6	12.1	18	12	4
10to16	Total	12.8	16	12.5	4	12.3	16	11	5
10to16	Visited	12.5	16	12	4	12.1	16	11	4
14to18	Total	15.9	18	16	4	15.1	18	14	2
14to18	Visited	15.5	18	15.5	3	14.9	18	14	1

Table B.73: Code17-1, Difference Between Total and Visited Number of States For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	7.8	8	8	0	7.6	8	8	1
4to8	Visited	7.8	8	8	0	7.6	8	8	1
6to12	Total	9.9	12	9	4	9.9	12	10	3
6to12	Visited	9.8	12	9	4	9.9	12	10	3
6to18	Total	11.6	18	11	5	10.3	16	10	4
6to18	Visited	11.4	17	11	5	10.2	16	10	4
8to14	Total	10.5	14	10	3	10.6	14	10	3
8to14	Visited	10.4	14	10	3	10.5	14	10	3
8to18	Total	12.4	18	12.5	4	11.8	18	11	6
8to18	Visited	12.3	18	12.5	4	11.6	18	11	6
10to16	Total	12	16	12	4	11.5	15	11	2
10to16	Visited	11.9	16	12	3	11.4	15	11	2
14to18	Total	14.9	18	14	2	15.3	18	14	3
14to18	Visited	14.8	18	14	2	15.1	18	14	3

Table B.74: Code17-1, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code17-2

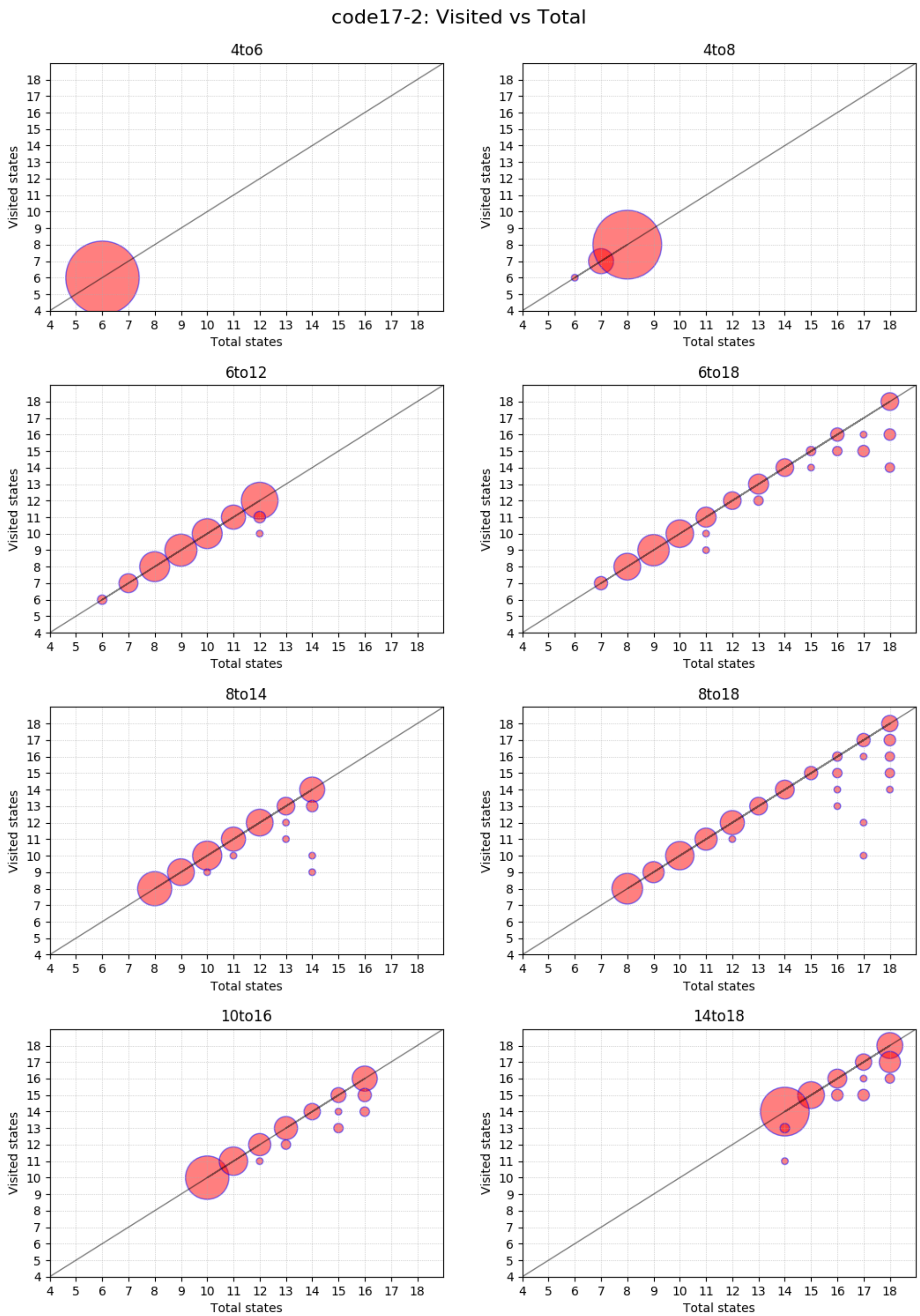

Figure B.38: The final SEMs machine size (total states) against the number of visited states across all experiments for code17-2

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	10.8	12	12	3	9.7	12	9.5	3
6to12	Visited	10.7	12	11.5	3	9.7	12	9.5	3
6to18	Total	13.2	18	13	8	11.6	18	10	4
6to18	Visited	12.7	18	13	7	11.4	18	10	4
8to14	Total	11.3	14	11.5	5	10.7	14	11	3
8to14	Visited	11	14	10.5	4	10.5	14	11	3
8to18	Total	12.6	18	12	8	12.3	18	12	5
8to18	Visited	12	18	12	6	12.1	18	12	4
10to16	Total	12.8	16	12.5	4	12.3	16	11	5
10to16	Visited	12.5	16	12	4	12.1	16	11	4
14to18	Total	15.9	18	16	4	15.1	18	14	2
14to18	Visited	15.5	18	15.5	3	14.9	18	14	1

Table B.75: Code17-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	7.7	8	8	1	7.8	8	8	0
4to8	Visited	7.7	8	8	1	7.8	8	8	0
6to12	Total	9.8	12	10	3	9.2	12	9	2
6to12	Visited	9.8	12	10	2	9.2	12	9	2
6to18	Total	10.6	18	10	3	11.1	18	10	5
6to18	Visited	10.5	16	10	3	10.9	15	10	5
8to14	Total	10.5	14	10	4	10.1	14	10	2
8to14	Visited	10.5	14	10	4	10.1	14	10	2
8to18	Total	11.8	18	11	5	11.9	18	11	4
8to18	Visited	11.5	18	11	5	11.7	18	11	4
10to16	Total	11.3	15	11	2	12.6	16	12.5	5
10to16	Visited	11.3	15	11	2	12.4	16	12	4
14to18	Total	15.5	18	15	3	15.3	18	15	3
14to18	Visited	15.4	18	15	3	15.1	18	15	2

Table B.76: Code17-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code18

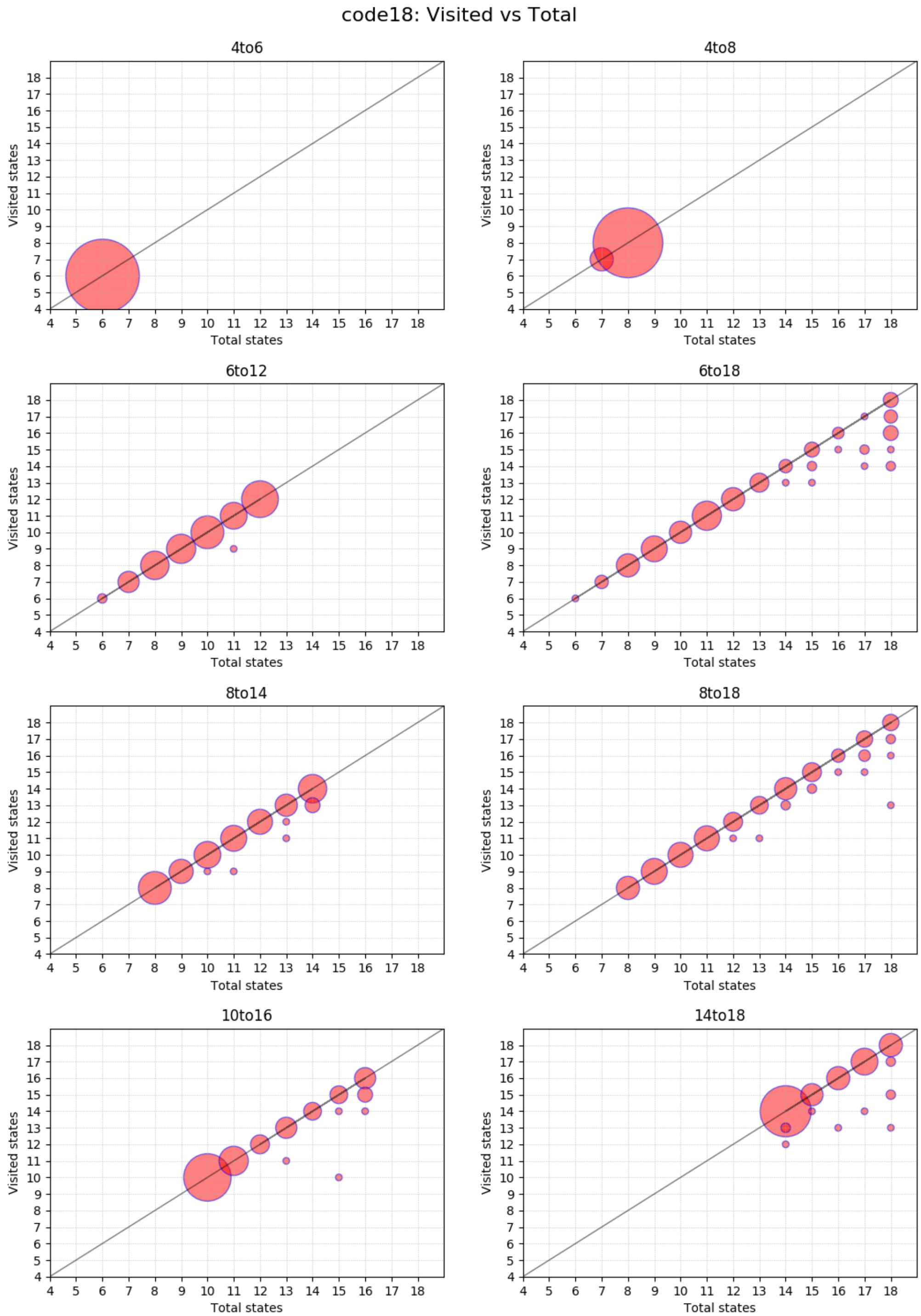

Figure B.39: The final SEMs machine size (total states) against the number of visited states across all experiments for code18

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	7.9	8	8	0	8	8	8	0
4to8	Visited	7.9	8	8	0	8	8	8	0
6to12	Total	10.6	12	11	3	10.2	12	10	3
6to12	Visited	10.6	12	11	3	10.2	12	10	3
6to18	Total	13.4	18	13	8	12.5	18	11	5
6to18	Visited	12.7	18	13	5	12.2	17	11	5
8to14	Total	11.1	14	11	4	11.6	14	12	4
8to14	Visited	10.8	14	11	4	11.6	14	12	4
8to18	Total	13.3	18	13.5	7	12.4	18	12.5	4
8to18	Visited	13.1	18	13	6	12.2	18	12	4
10to16	Total	12.1	16	10.5	5	12.1	16	11.5	4
10to16	Visited	11.9	16	10.5	5	11.9	16	11	3
14to18	Total	15.5	18	14	4	15.2	18	14	2
14to18	Visited	15.2	18	14	3	15.1	18	14	2

Table B.77: Code18, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	7.8	8	8	0	7.8	8	8	0
4to8	Visited	7.8	8	8	0	7.8	8	8	0
6to12	Total	9.1	12	9	2	9.6	12	10	3
6to12	Visited	9.1	12	9	2	9.5	12	9.5	3
6to18	Total	11.8	18	12	5	10.9	18	10.5	4
6to18	Visited	11.7	18	12	5	10.8	18	10.5	4
8to14	Total	10.1	14	10	4	11	14	11	4
8to14	Visited	10.1	14	10	3	11	14	11	4
8to18	Total	12.3	18	11	5	11.9	18	11.5	5
8to18	Visited	12	18	11	5	11.9	18	11.5	5
10to16	Total	11.9	16	11	3	11.8	16	11	3
10to16	Visited	11.8	16	11	3	11.8	16	11	3
14to18	Total	15	17	15	2	15.5	18	15	3
14to18	Visited	15	17	15	2	15.1	18	14	3

Table B.78: Code18, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

B.2.2 Codes of Length 12

Code55

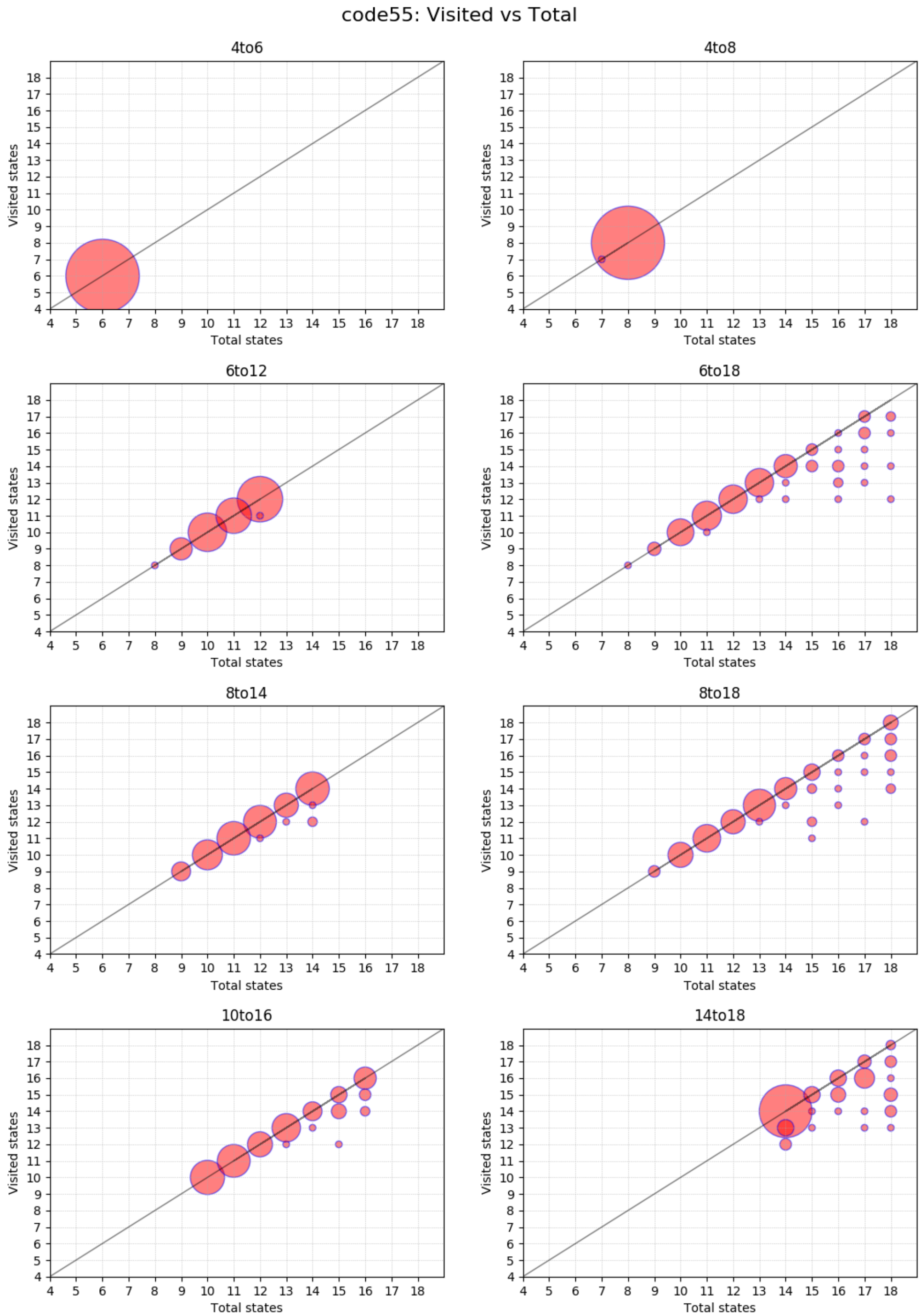

Figure B.40: The final SEMs machine size (total states) against the number of visited states across all experiments for Code55

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.4	12	12	1	11.1	12	11.5	2
6to12	Visited	11.4	12	12	1	11.1	12	11.5	2
6to18	Total	14.2	18	15	6	12.6	18	12.5	3
6to18	Visited	13.2	17	13	4	12.1	16	12	2
8to14	Total	12.8	14	13	2	11.6	14	11.5	2
8to14	Visited	12.7	14	12.5	2	11.5	14	11.5	2
8to18	Total	13.3	18	13	5	13.1	18	13	4
8to18	Visited	12.7	17	12	3	12.7	18	13	3
10to16	Total	13	16	13	4	12.4	16	12	2
10to16	Visited	12.8	16	13	4	12.3	16	12	2
14to18	Total	15.5	18	14.5	3	15	18	14	3
14to18	Visited	14.8	18	14	2	14.4	17	14	1

Table B.79: Code55, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	10.5	12	10.5	1	10.6	12	11	1
6to12	Visited	10.5	12	10.5	1	10.6	12	11	1
6to18	Total	12.1	16	12.5	4	12.6	18	12	3
6to18	Visited	12.1	16	12	4	12.4	17	12	2
8to14	Total	11.6	14	11.5	3	11.4	14	11	3
8to14	Visited	11.5	14	11	2	11.4	14	11	3
8to18	Total	13.2	18	13	2	13.8	18	13	6
8to18	Visited	13	18	13	2	13.5	18	13	4
10to16	Total	12.2	16	11	4	12.4	16	12.5	3
10to16	Visited	12	16	11	4	12.3	16	12.5	3
14to18	Total	14.8	18	14	1	15.2	18	14	2
14to18	Visited	14.3	17	14	1	14.6	18	14	1

Table B.80: Code55, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code60-1

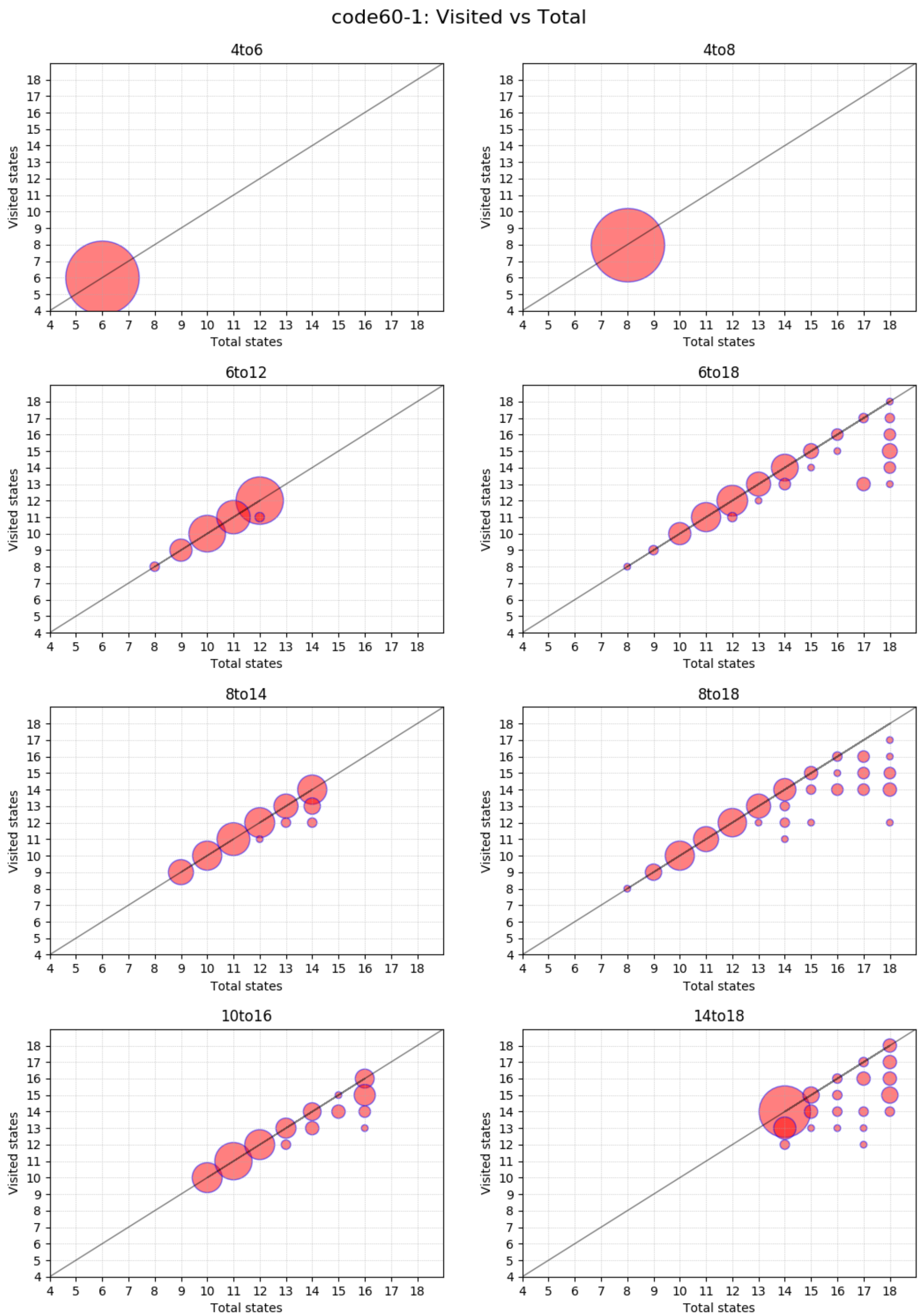

Figure B.41: The final SEMs machine size (total states) against the number of visited states across all experiments for Code60-1

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.4	12	12	1	11.1	12	12	2
6to12	Visited	11.3	12	12	1	11.1	12	12	2
6to18	Total	14.2	18	14	5	13.8	18	13	6
6to18	Visited	13.3	17	13	2	12.9	18	13	3
8to14	Total	12.2	14	12	3	12.1	14	12	3
8to14	Visited	12	14	12	2	12	14	12	2
8to18	Total	13.8	18	14	3	13.9	18	13.5	5
8to18	Visited	12.8	15	13	2	13	16	13	2
10to16	Total	12.8	16	12.5	4	12.1	16	11.5	2
10to16	Visited	12.5	16	12	3	11.9	15	11.5	2
14to18	Total	15.8	18	16	4	14.6	18	14	0
14to18	Visited	14.6	18	14	1	14	17	14	0

Table B.81: Code60-1, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	10.8	12	11	2	10.5	12	11	1
6to12	Visited	10.7	12	11	2	10.5	12	11	1
6to18	Total	12.7	18	12	3	12.4	18	12	3
6to18	Visited	12.5	16	12	3	12.3	16	12	3
8to14	Total	10.9	14	11	3	11.6	14	11.5	3
8to14	Visited	10.9	14	11	3	11.5	14	11.5	3
8to18	Total	11.6	18	11	3	12.8	18	12	5
8to18	Visited	11.4	14	11	3	12.4	17	12	4
10to16	Total	12.7	16	12	3	12.5	16	12	3
10to16	Visited	12.5	16	12	3	12.3	16	12	3
14to18	Total	15.4	18	15	3	14.6	18	14	1
14to18	Visited	14.8	18	14	2	14.2	18	14	0

Table B.82: Code60-1, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code60-2

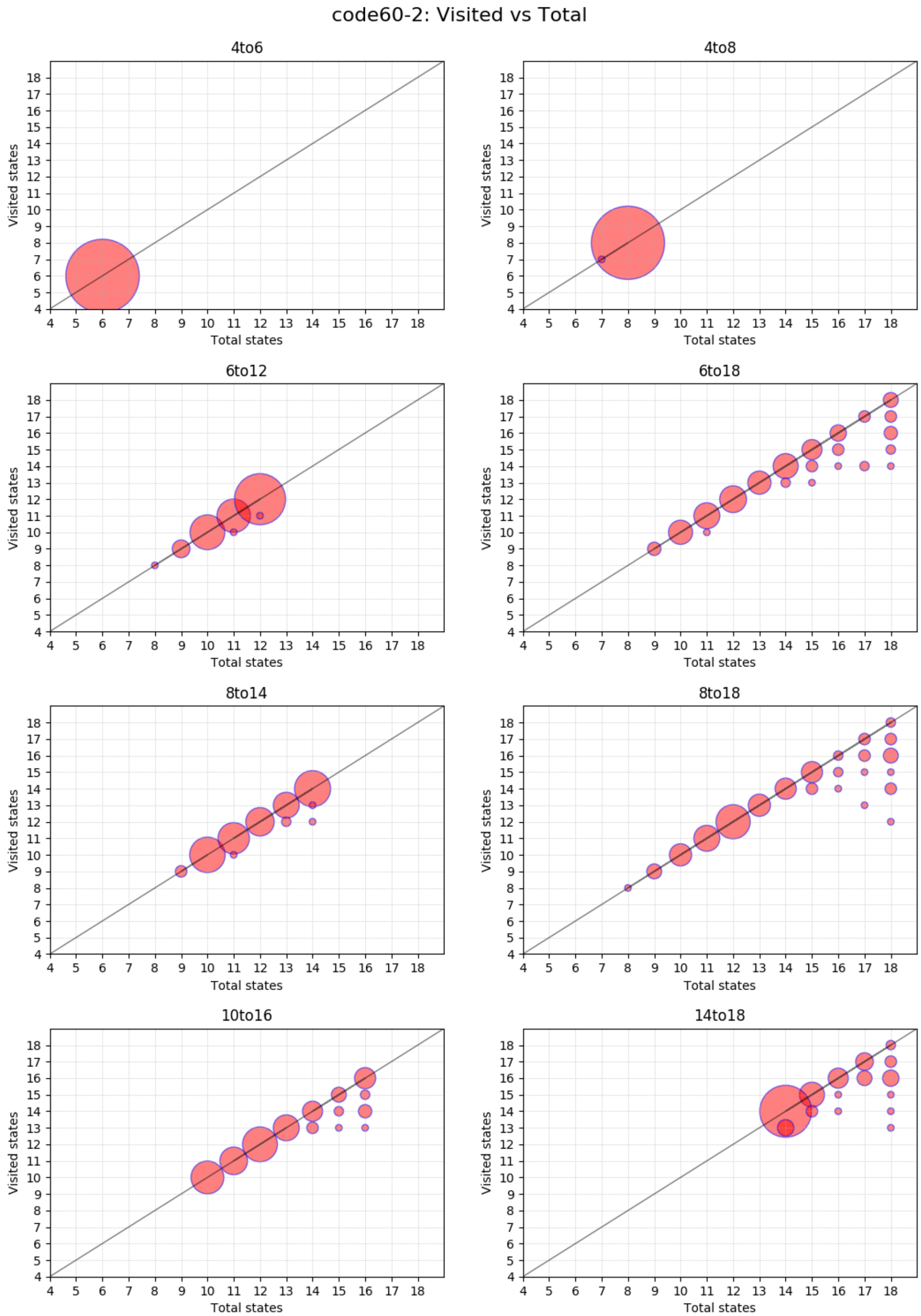

Figure B.42: The final SEMs machine size (total states) against the number of visited states across all experiments for Code60-2

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.6	12	12	1	11.3	12	12	2
6to12	Visited	11.6	12	12	1	11.3	12	12	2
6to18	Total	14.8	18	15	4	13.7	18	14	5
6to18	Visited	14.1	18	14	3	13.4	18	14	4
8to14	Total	12.6	14	13	3	12	14	12	3
8to14	Visited	12.5	14	13	3	11.9	14	12	4
8to18	Total	14.4	18	14.5	5	13.4	18	13	5
8to18	Visited	13.3	17	13	3	13.1	18	13	4
10to16	Total	12.3	16	12	3	12.2	16	12	4
10to16	Visited	12.1	16	12	2	12.1	16	12	3
14to18	Total	15.2	18	14	3	15.2	18	14	3
14to18	Visited	14.7	17	14	2	14.8	18	14	2

Table B.83: Code60-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	10.9	12	11	2	10.7	12	11	2
6to12	Visited	10.9	12	11	2	10.7	12	11	2
6to18	Total	12.3	18	11	3	13.3	18	13	3
6to18	Visited	12.1	18	11	3	13.2	17	13	3
8to14	Total	11.8	14	11.5	4	11.4	14	11	3
8to14	Visited	11.7	14	11.5	3	11.4	14	11	3
8to18	Total	12.8	18	12	4	12.8	18	12	4
8to18	Visited	12.7	18	12	3	12.7	16	12	4
10to16	Total	12.8	16	12.5	3	12.9	16	12	3
10to16	Visited	12.6	16	12.5	3	12.7	16	12	2
14to18	Total	15.1	18	14.5	2	14.9	18	14	2
14to18	Visited	14.7	18	14	1	14.7	17	14	2

Table B.84: Code60-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

B.2.3 Codes of Length 14

Code201

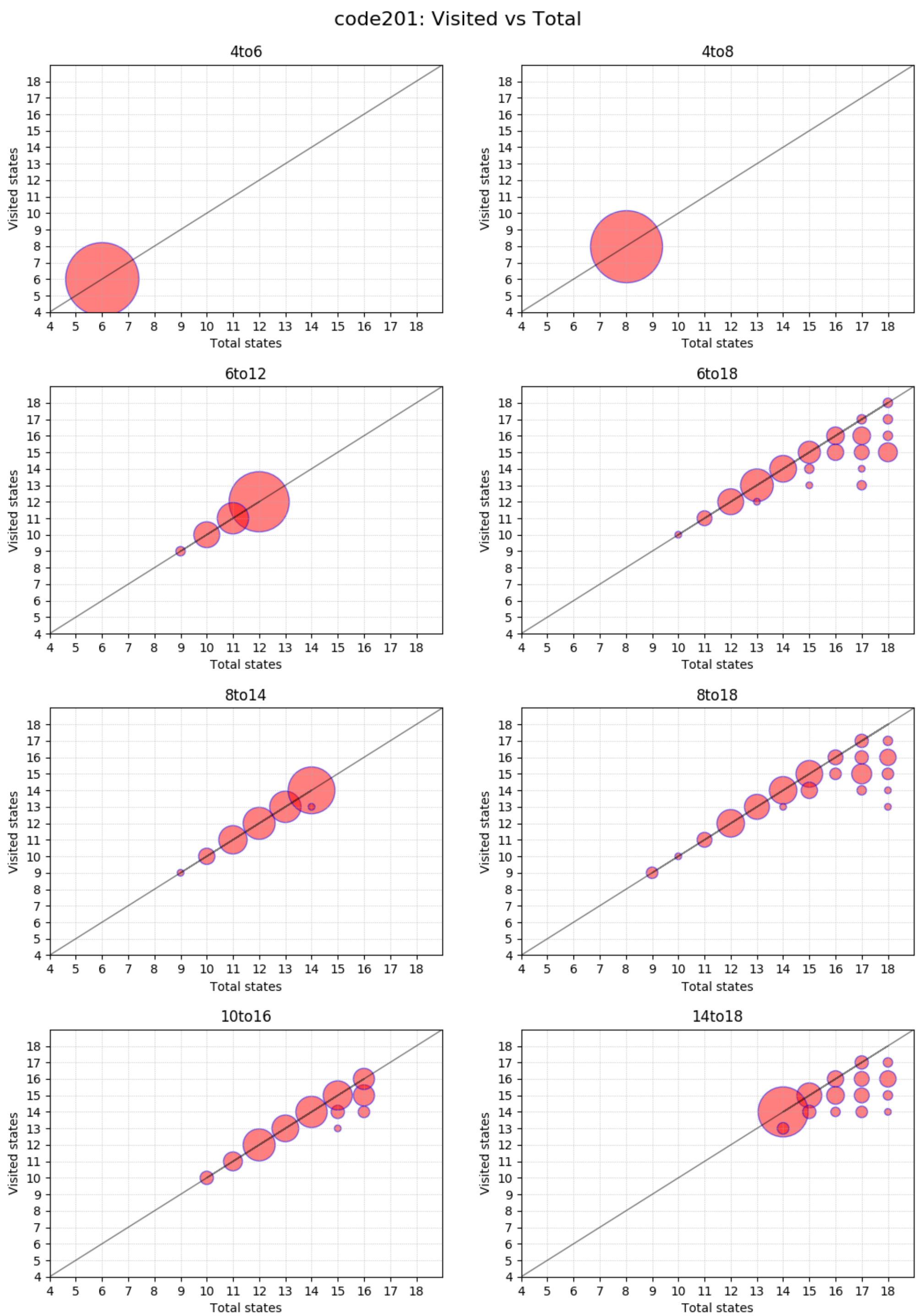

Figure B.43: The final SEMs machine size (total states) against the number of visited states across all experiments for Code201

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.8	12	12	0	11.6	12	12	0
6to12	Visited	11.8	12	12	0	11.6	12	12	0
6to18	Total	15	18	15	4	14.4	18	14	4
6to18	Visited	14	17	14	2	13.8	17	14	3
8to14	Total	13.1	14	14	2	12.9	14	13	2
8to14	Visited	13.1	14	14	2	12.9	14	13	2
8to18	Total	15.5	18	16	3	14.6	18	15	5
8to18	Visited	14.5	17	15	3	14.1	17	15	3
10to16	Total	14	16	14	2	14.1	16	14.5	3
10to16	Visited	13.7	16	14	2	13.9	16	14	2
14to18	Total	15.4	18	14.5	3	15.1	18	14.5	2
14to18	Visited	14.6	17	14	1	14.5	16	14	1

Table B.85: Code201, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8.9	16	8	0
4to8	Visited	8	8	8	0	8.9	16	8	0
6to12	Total	11.3	12	12	1	11.3	12	12	1
6to12	Visited	11.3	12	12	1	11.3	12	12	1
6to18	Total	14.5	18	14	3	14.6	18	15	3
6to18	Visited	14.1	17	14	2	14.2	18	15	3
8to14	Total	12.6	14	13	2	12.4	14	12.5	3
8to14	Visited	12.6	14	13	2	12.4	14	12.5	3
8to18	Total	14.4	18	14	1	13.6	18	13.5	3
8to18	Visited	13.9	16	14	2	13.4	17	13.5	3
10to16	Total	13.5	16	14	3	13.4	16	13	3
10to16	Visited	13.3	16	13.5	2	13.3	16	13	3
14to18	Total	14.9	17	14	1	15.3	18	15	3
14to18	Visited	14.6	17	14	1	14.9	17	14	2

Table B.86: Code201, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code205-1

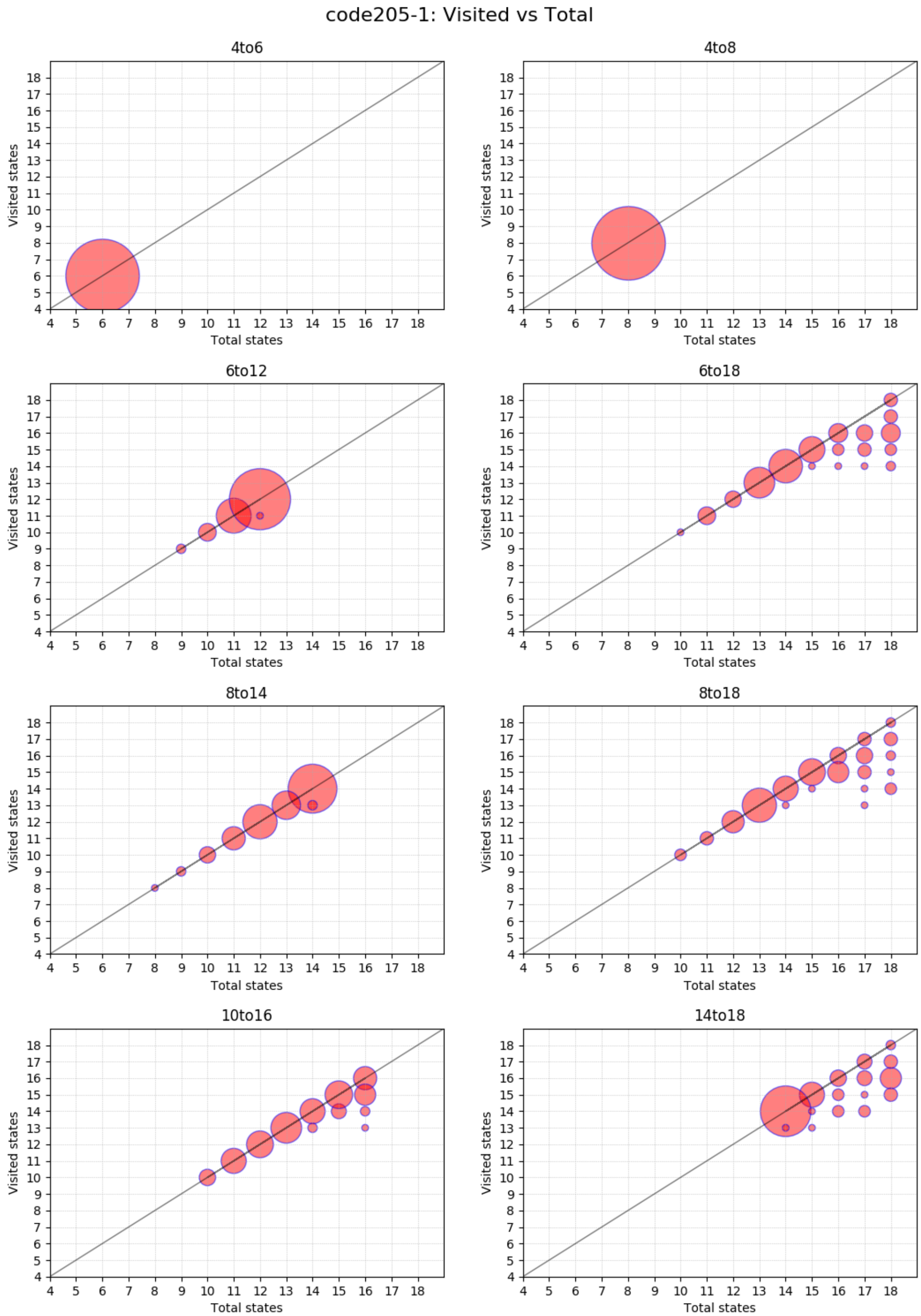

Figure B.44: The final SEMs machine size (total states) against the number of visited states across all experiments for Code205-1

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.9	12	12	0	11.7	12	12	0
6to12	Visited	11.9	12	12	0	11.7	12	12	0
6to18	Total	15.2	18	15	5	14.4	18	14	2
6to18	Visited	14.5	18	14.5	3	14	17	14	2
8to14	Total	13.3	14	14	1	12.7	14	13	2
8to14	Visited	13.3	14	14	1	12.7	14	13	2
8to18	Total	15.2	18	15	3	14.9	18	14	4
8to18	Visited	14.4	18	15	2	14.4	18	14	3
10to16	Total	14	16	14	2	13.6	16	13.5	4
10to16	Visited	13.8	16	14	2	13.4	16	13	3
14to18	Total	15.6	18	16	3	15.5	18	14.5	4
14to18	Visited	14.8	17	14	2	15	18	14	2

Table B.87: Code205-1, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.4	12	12	1	11.4	12	11	1
6to12	Visited	11.4	12	12	1	11.4	12	11	1
6to18	Total	15.2	18	15	2	14.4	18	14	3
6to18	Visited	14.7	18	14	2	14.1	18	14	2
8to14	Total	12.4	14	12.5	2	12.7	14	13	2
8to14	Visited	12.4	14	12.5	2	12.7	14	13	2
8to18	Total	14.5	18	14.5	3	13.9	18	13	4
8to18	Visited	14.2	17	14	2	13.6	17	13	3
10to16	Total	13.7	16	14	3	13.1	16	13	2
10to16	Visited	13.6	16	14	3	12.9	15	13	2
14to18	Total	14.9	17	15	2	15.3	18	14.5	3
14to18	Visited	14.7	17	14	1	14.7	17	14	1

Table B.88: Code205-1, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)

Code205-2

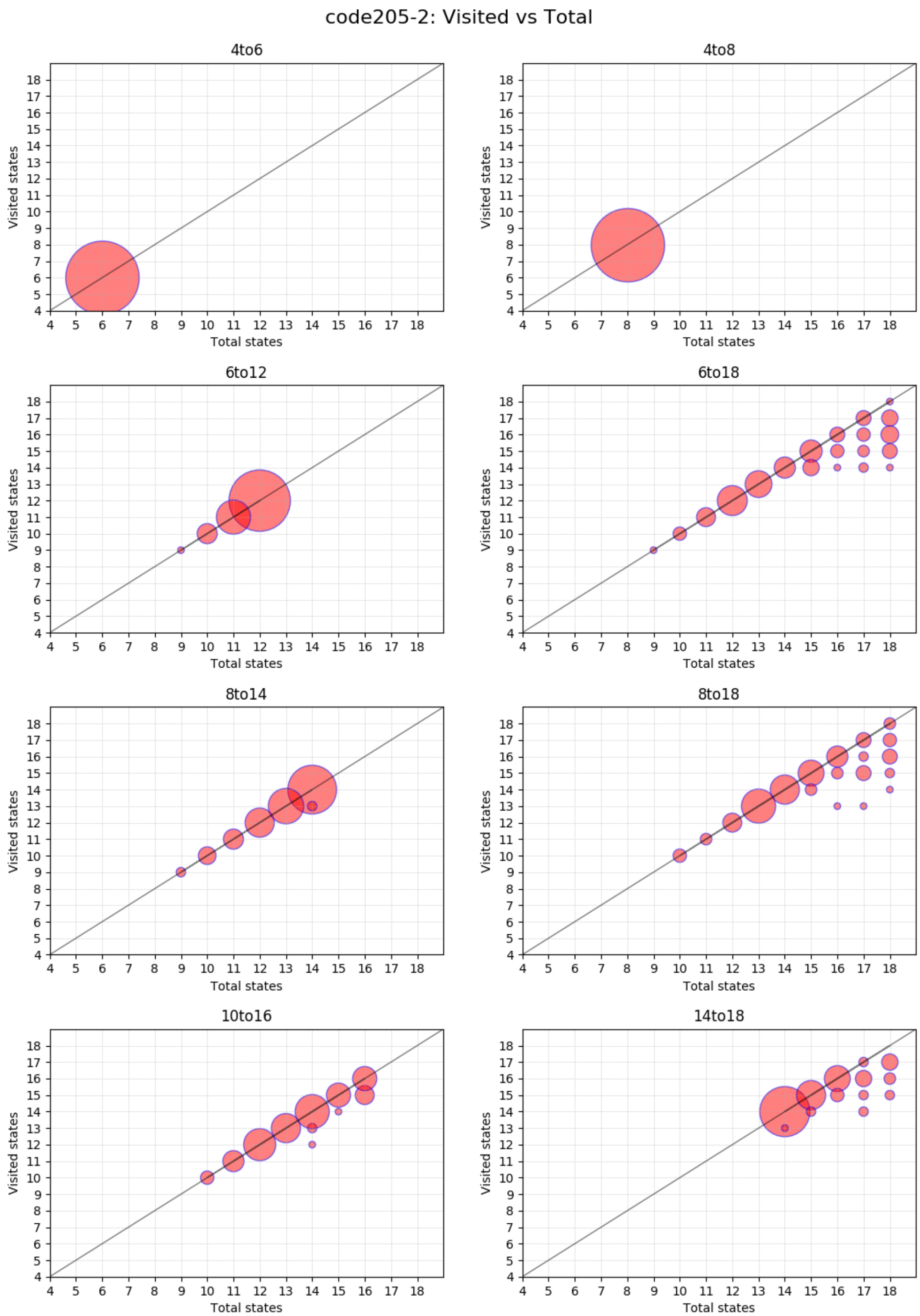

Figure B.45: The final SEMs machine size (total states) against the number of visited states across all experiments for Code205-2

State Range	Final Machine State Size	E1 Mean	E1 Max	E1 Median	E1 IQR	E2 Mean	E2 Max	E2 Median	E2 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.7	12	12	0	11.7	12	12	0
6to12	Visited	11.7	12	12	0	11.7	12	12	0
6to18	Total	16.3	18	17	3	14.2	18	14	4
6to18	Visited	15	17	15	2	13.8	18	14	3
8to14	Total	13.2	14	13.5	2	13.2	14	14	2
8to14	Visited	13.1	14	13	2	13.2	14	14	2
8to18	Total	14.8	18	14.5	4	14.6	18	14.5	3
8to18	Visited	14.1	17	14	2	14	18	14	3
10to16	Total	13.9	16	14	3	12.8	16	12	2
10to16	Visited	13.7	16	14	2	12.7	15	12	2
14to18	Total	15.5	18	15	3	15.2	18	15	3
14to18	Visited	14.9	17	14.5	2	14.9	17	14.5	1

Table B.89: Code205-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 1(E1) and For Experiment 2(E2)

State Range	Final Machine State Size	E3 Mean	E3 Max	E3 Median	E3 IQR	E4 Mean	E4 Max	E4 Median	E4 IQR
4to6	Total	6	6	6	0	6	6	6	0
4to6	Visited	6	6	6	0	6	6	6	0
4to8	Total	8	8	8	0	8	8	8	0
4to8	Visited	8	8	8	0	8	8	8	0
6to12	Total	11.4	12	11.5	1	11.6	12	12	1
6to12	Visited	11.4	12	11.5	1	11.6	12	12	1
6to18	Total	13.6	18	13	5	13.8	18	13.5	3
6to18	Visited	13.3	17	13	4	13.5	17	13.5	3
8to14	Total	12.5	14	13	3	12.8	14	13	2
8to14	Visited	12.5	14	13	3	12.8	14	13	2
8to18	Total	14.7	18	14.5	3	14.6	18	14.5	3
8to18	Visited	14.5	17	14.5	3	14.4	18	14.5	2
10to16	Total	14.1	16	14	1	13.4	16	13	2
10to16	Visited	14	16	14	2	13.4	16	13	2
14to18	Total	15.2	18	15	2	14.7	17	14	1
14to18	Visited	15	17	15	2	14.5	16	14	1

Table B.90: Code205-2, Difference Between Total and Visited Number of States of Final Machine For Experiment 3(E3) and For Experiment 4(E4)