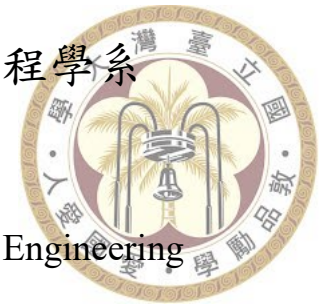
國立臺灣大學電機資訊學院資訊工程學系



碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

針對有限精度單峰映射影像加密演算法之圖神經網路參數迴歸攻擊

A GNN Based Parameter Regression Attack on Image Encryption Algorithm Using Logistic Map under Finite Precision Constraints

鄒汶橙

Wen-Cheng Tsou

指導教授: 吳家麟 博士

Advisor: Ja-Ling Wu Ph.D.

中華民國 111 年 7 月

July, 2022

doi:10.6342/NTU202201736

國立臺灣大學碩士學位論文
口試委員會審定書

針對有限精度單峰映射影像加密演算法之圖神經網路參數迴歸攻擊

A GNN Based Parameters Regression Attack on Image Encryption Algorithm Using Logistic Map under Fintie Precision Constraints

本論文係鄒汶橙君（學號 R09922131）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 111 年 7 月 6 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

吳家麟

（指導教授）

陳文進　　陳駿丞

許超雲

系主任　洪士灝

致謝

首先非常感謝我的指導教授吳家麟老師，老師對於大學就讀應用數學系的我給予相對應的教材和研究方向，在我還沒入學時就給了我很多資源和指導，讓我能很快地就進入狀況，也很早就確定自己要往混沌映射的研究前進。在研究所的這兩年，老師時常分享最新的研究趨勢，讓我在修課的同時能夠和研究進度並行。最重要的是，老師給了我們很高的自由度，並不會限制我們研究的主題，鼓勵我們有更多想法可以提出並且實現，讓我們在不斷嘗試和失敗中學習到每個領域的精髓所在。很幸運自己能在老師的麾下學習成長，也希望自己能永遠記住老師在研究上的精神，帶著這種精神在未來的路上前進。

當然也非常感謝實驗室的資源，不管是硬體設備上，助理的協助，以及實驗室夥伴們的討論和經驗分享，都讓我在學習的路上走得更順利。謝謝 Sylvia 和 Jimmy 和我討論研究方面的各種問題，每當我遇到某個困境的時候都能引導我走出死胡同，找到另一個觀點上的解法。謝謝哲寬，日翔，和元均在我修課期間提供的協助，非本科系的我在修課上遇到很多問題，謝謝他們不管時候都願意停下來花時間陪我一起解決問題。

最後要謝謝我的家人們，無條件的支持我往更高的學術殿堂前進，在這疫情肆虐的兩年我們總是聚少離多，但我相信我們的心是在一塊的。謝謝他們在金錢和精神上的援助，未來的我必定對社會有所貢獻，並將所學所得回饋給自己最愛的家人們。

doi:10.6342/NTU202201736

摘要

混沌映射性近幾年被廣泛應用在密碼學上，主要仰賴其確定性、非週期性、及對初始值的敏感性。而在影像加密的領域中，混沌映射更是被套用為在空間域中的加密元件，使其成為一種偽隨機亂數生成器 (PRNG)。跟傳統的影像加密方法相比，這種加密方式提供的最大好處是不會讓密文檔案大小過度膨脹，並且能提供極高速的加密速度，在現今要求即時多媒體應用的需求下，混沌映射是非常好的解藥。在此論文中，我們提出了一種以圖神經網路為基底的攻擊手法，透過針對特定混沌映射，在有限精度下設計的先備知識訓練集，預訓練出一個能夠迴歸參數的圖神經網路模型。攻擊時我們假設攻擊者能夠竊聽到一部分的混沌序列，並將該序列輸入模型進行迴歸，進而推導出出該混沌序列所對應到的參數，此參數在影像加密演算法中即相當於金鑰。換言之，若此法順利迴歸出混沌序列之參數，則表示攻擊 (或破密) 成功

關鍵字：混沌映射、有限精度、圖神經網路、參數迴歸

doi:10.6342/NTU202201736

Abstract

Owing to their deterministic, nonperiodic, and sensitive to initial state characteristics, chaotic maps are widely applied to cryptography in recent years. In chaos-based image encryption schemes, chaotic maps behave like a pseudo random number generator for encrypting digital images in the spatial domain. Contrary to traditional methods, the major advantage of chaos-based image encryption algorithm is its high encrypt speed and limited size expansion of the cipherimage. Thanks to this advantages, chaos-based image encryption algorithm has become a great remedy in real-time secure multimedia application areas nowadays. In this work, by designing a flexible and novel algorithm to generate appropriate training set based on the domain knowledge of specific chaotic maps in finite precision and pre-training a Graph Neural Network model for conducting parameters regressions, a GNN based parameter regression attack is proposed. To launch the attack, we assume that attacker has ability to eavesdrop a portion of the chaotic sequence, and the captured sequence segment is then input to the pre-tarined model we mentioned above for

doi:10.6342/NTU202201736

getting the corresponding parameter. The obtained parameter is equivalent to the key in a chaos-based image encryption algorithm. If the parameter can be found by the network, we say the attack is successful.

Keywords: Chaotic maps, Finite precision, Graph neural network, Parameter regression

doi:10.6342/NTU202201736

目錄

Page

口試委員審定書 i

致謝 iii

摘要 v

Abstract vii

目錄 ix

圖目錄 xiii

表目錄 xv

第一章 簡介 1

第二章 背景介紹和攻擊者設定 3

2.1 混沌映射影像加密演算法 3

2.2 單峰映射 (logistic map) 4

2.3 有限精度下的混沌映射 5

2.4 攻擊者模型 (Adversary model) 6

第三章 訓練集生成演算法 7

3.1 可接受區間 7

3.2 訓練資料集生成 8

3.2.1 GenerateSMN 函數 10

doi:10.6342/NTU202201736

3.2.2 CalculateMPL 函數 10
3.2.3 GetMeanofMPL 函數 10
3.2.4 GetMaxofMPL 函數 11
3.2.5 CalCleanPro 函數 11
3.2.6 VisitfromCand 函數 11
3.2.7 分配參數值 11
3.3 測試資料集生成 12
第四章 模型與實驗設計 **13**
4.1 圖神經網路模型 14
4.1.1 版本一 (Model 1) 14
4.1.2 版本二 (Model 2) 15
第五章 實驗結果 **17**
5.1 資料集設定 17
5.2 模型一實驗結果 19
5.2.1 不同精度下的學習曲線 (learning curve) 19
5.2.2 訓練趨於穩定後損失函數之比較 21
5.3 模型二實驗結果 23
5.3.1 不同精度下的學習曲線 (learning curve) 23
5.3.2 訓練趨於穩定後損失函數之比較 26
5.4 模型二在資訊受限之表現 27
5.5 gv 和 cp 在模型二對實驗各項指標的影響 30
5.5.1 固定 cp 31
5.5.2 固定 gv 33

doi:10.6342/NTU202201736

5.6 實際攻擊案例及分析 34
5.6.1 簡易影像加密演算法 34
5.6.1.1 GenerateCS 函數 35
5.6.1.2 Confusion 函數 35
5.6.1.3 Diffusion 函數 35
5.6.2 攻擊流程 35
5.6.3 金鑰空間縮減分析 36
第六章 結論 **39**
參考文獻 **41**

doi:10.6342/NTU202201736

doi:10.6342/NTU202201736

圖目錄

2.1 混沌映射影像加密演算法基本架構 3

2.2 單峰映射之分岔圖 5

2.3 單峰映射在 $r = \frac{122}{2^5}$ 之狀態映射網路 5

4.1 我們提出的參數迴歸攻擊模型之訓練架構 13

4.2 模型版本一 14

4.3 模型版本二 16

5.1 訓練資料集生成所需時間 (單位：秒) 18

5.2 模型一當 n = 5bits 時之學習曲線 19

5.3 模型一當 n = 6bits 時之學習曲線 19

5.4 模型一當 n = 7bits 時之學習曲線 20

5.5 模型一當 n = 8bits 時之學習曲線 20

5.6 模型一當 n = 9bits 時之學習曲線 20

5.7 模型一之驗證集損失函數 21

5.8 模型一之測試集損失函數 22

5.9 模型一測試集攻擊成功率 22

5.10 模型二當 n = 5bits 時之學習曲線 23

5.11 模型二當 n = 6bits 時之學習曲線 23

5.12 模型二當 n = 7bits 時之學習曲線 24

5.13 模型二當 n = 8bits 時之學習曲線 24

5.14 模型二當 n = 9bits 時之學習曲線 24

5.15 模型二當 n = 10bits 時之學習曲線 25

5.16 模型二當 n = 11bits 時之學習曲線 25

doi:10.6342/NTU202201736

5.17 模型二之驗證集損失函數 26
5.18 模型二之測試集損失函數 26
5.19 模型二之測試集攻擊成功率 27
5.20 Test image 36
5.21 Cipherimage 36
5.22 Decrypted image 37

doi:10.6342/NTU202201736

表目錄

5.1 隨機採樣資料集 (Random-set) 18

5.2 採樣演算法生成資料集 (Algorithm-set) 18

5.3 Cut_edge(n = 5) 28

5.4 Cut_edge(n = 6) 28

5.5 Cut_edge(n = 7) 29

5.6 Cut_edge(n = 8) 29

5.7 Cut_edge(n = 9) 29

5.8 Cut_edge(n = 10) 30

5.9 Cut_edge(n = 11) 30

5.10 Fixed $cp = 0.15$ 31

5.11 Fixed $cp = 0.1$ 32

5.12 Fixed $gv = 0.5$ 33

5.13 暴力破解時間複雜度分析 37

5.14 迴歸參數攻擊時間複雜度分析 38

doi:10.6342/NTU202201736

doi:10.6342/NTU202201736

第一章　簡介

傳統的影像加密方法常將影像這類的多媒體資料當作一般資料直接進行加密，這樣的做法忽略了影像資料在每個像素之間相似的特性。此外，因為影像的檔案大小通常較大，若經過加密會導致檔案大小過度膨脹。而影像通常在傳輸過程中都是保持加密狀態的，這不僅使儲存負擔增加，也使得傳輸所需的頻帶寬度大幅增加。傳統影像加密方法的另一缺點在於執行速度太慢；許多方法都會將影像先轉成位元 (bit-level) 再進行加密，這一來一往的轉換再加上執行傳統加密方法都會使執行時間急遽增加，而這樣的速度在現今社會及時串流的需求下很明顯是不合時宜的。混沌映射影像加密演算法改善了上述傳統影像加密演算法的問題。Fridrich 在 1998 年提出了第一個以混沌映射為基底的影像加密演算法 [1]，一直到現在都陸陸續續還有更多新的混沌映射影像加密演算法被提出。不過大部分的相關文獻都依循著 Fridrich 所提出的 confusion-diffusion 架構。最近幾年，有越來越多針對混沌映射影像加密演算法的安全性分析被提出，其中 [2] 針對 Fridrich 提出的架構進行分析，Solak 在 [2] 中利用矩陣理論，設計出了一種選擇密文攻擊法，其中使用的技巧是去找出經過混沌映射轉換後像素排列的規律，而非破解其使用的參數 (金鑰)。[3] 則是基於 [2] 做了更近一步的分析，同時也將其之中一些缺點進行修正。當然針對特定混沌映射影像加密演算法做分析的文獻也不勝枚舉，其中以 [4] 最為經典。[4] 針對有限精度如何影響混沌映射做了圖結構的分析，同時詳細描述 [5] 在在設計上的瑕疵和現今許多測量加密圖之標準的不合理處。

doi:10.6342/NTU202201736

本篇論文主要以攻擊者角度出發針對混沌映射做分析，利用對特定混沌映射在有限精度下的先備知識，進而能夠對混沌映射在某個參數值時所對應到的狀態圖 (state mapping network) 做初始狀態的採樣。我們設計了一個快速有效且具彈性的演算法，能夠對每個參數區間對應到的狀態圖做最大效益的採樣。假設攻擊者使用上述的技巧生成訓練集，即可訓練一個圖神經網路進行圖層面的迴歸任務，迴歸目標即為該筆資料的參數值。由於本篇論文的討論是在有限精度前提下實施的，因此，即使迴歸後得到的值和原始資料之參數值有些許差異，在有限狀態機中仍有可能被視為對應到同個值，這也就是連續域混沌映射在理論上和有限精度實務面上的主要差異，其混亂性 (即對初始值些微更動的敏感性) 會因為實作在有限狀態機上而消失。

本篇論文的貢獻可以總結為以下三點:

(1) 針對混沌映射，我們設計了一個快速且具彈性的訓練集生成演算法

(2) 就我們的認知內，從未有人使用這樣的方法對混沌映射進行參數分析和攻擊

(3) 跟其他密碼學分析相比，本篇論文的攻擊手法可以針對任何使用特定混沌映射的影像加密演算法實施，也是本篇論文潛在的最大貢獻

第二章會介紹有關混沌映射影像加密演算法的背景知識和攻擊者能力設定，第三章說明我們設計的訓練集生成演算法，第四章介紹實作細節和模型，第五章為實驗結果，第六章做整體論文的總結

doi:10.6342/NTU202201736

第二章　背景介紹和攻擊者設定

2.1　混沌映射影像加密演算法

混沌映射影像加密演算法的基本架構如圖 2.1 所示，此方法直接對影像在空間域 (spatial-domain) 進行操作。其中 Confusion stage 主要的工作是將圖中像素的位置重新排列，Diffusion stage 則是將像素的值做有序更動。然而，上述兩個動作都必須依據混沌映射所迭代出的混沌序列執行。換言之，混沌映射的功能在此處等價於一個偽隨機亂數生成器 (PRNG)。直觀上來看混沌映射影像加密演算法的架構其實相對簡單，也就是因為架構簡單，在執行上能得到很快的速度。視覺上看加密後的圖片也完全無法得到任何原始輸入相關的資訊，但大部分的文獻都還是會將 confusion and diffusion 多做幾次遞迴，雖然會犧牲一點時間，但實驗結果顯

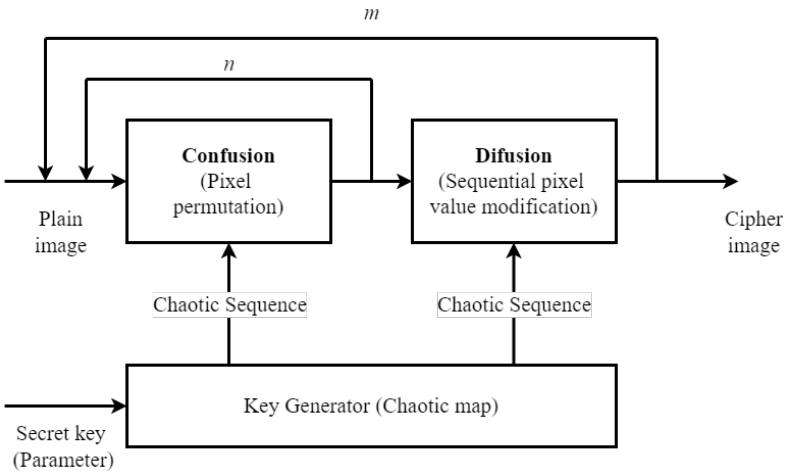


Figure 2.1: 混沌映射影像加密演算法基本架構

doi:10.6342/NTU202201736

示這樣會讓加密後的圖片得到更高的安全性。也有許多文獻 [6, 7] 漸漸轉為使用更高維度的混沌映射，希望可以讓產生出來的混沌序列有更高的不確定性。不過這個動作也使每次迭代需要的運算提升不少，如果在演算法設計上沒有仔細考量，很容易造成時間上的浪費。例如，[8] 在混沌序列上只使用了一個維度，但為了生成整段序列需要把另外兩個維度也一併納入計算，卻沒有投入使用，使其執行時間拖延非常之久，失去了混沌映射影像加密演算法的最大優勢。同時，因安全性越來越受重視，越來越多針對單一混沌映射影像加密演算法的攻擊如雨後春筍般出現，主要以選擇明文攻擊 (chosen-plaintext attack) 和差分密碼分析 (Differential attack) 為最大宗。上述種種跡象都表明 Fridrich 所提出的混沌映射影像加密演算法架構 (confusion-diffusion based) 可能存在安全性上的問題，也有一批學者是針對演算法所使用的混沌映射分析，使得近幾年大部分的混沌映射影像加密演算法都會再進一步對其使用的映射加以特別的設計；例如，串聯不同的映射，對混沌映射的參數進行浮動調整，以增加其系統的不確定性。

2.2 單峰映射 (logistic map)

本篇論文主要針對單峰映射做分析，同時單峰映射也是混沌映射中最為經典也最多人使用的混沌映射。單峰映射的數學式為 $x_{n+1} = rx_n(1 - x_n)$，此處 r 為參數值，x_n 為初始值，通常在加密系統中會將參數值和初始值組成金鑰對 (key pairs)，任何加密演算法一但公開，整體的安全性保障就全依賴於金鑰空間 (key space) 大小。而就單峰映射而言，其金鑰空間大小取決於機器使用的精度，且單峰映射的參數值在某個特定區間才有混沌的特性，如圖 2.2 所示，r 在 $3.6 \sim 4$ 時才有混沌的性質，在 $r < 3.6$ 時單峰映射的跌代都會收斂至某些固定值。

doi:10.6342/NTU202201736

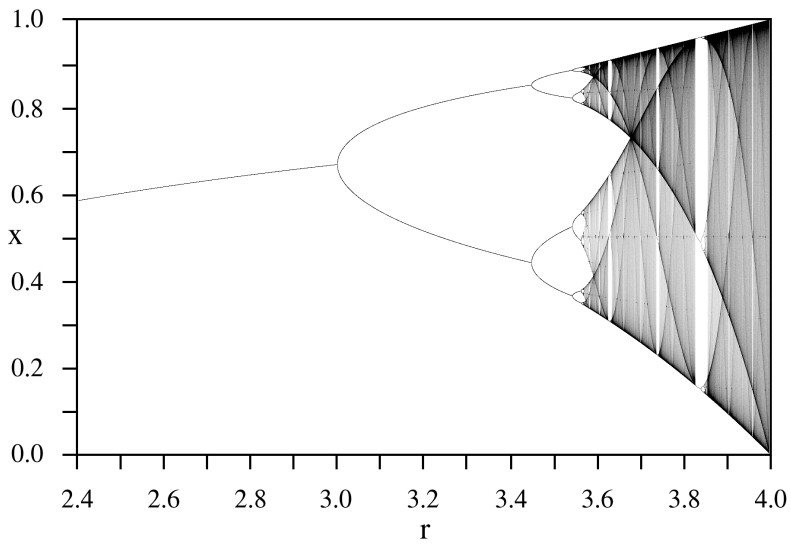


Figure 2.2: 單峰映射之分岔圖

2.3 有限精度下的混沌映射

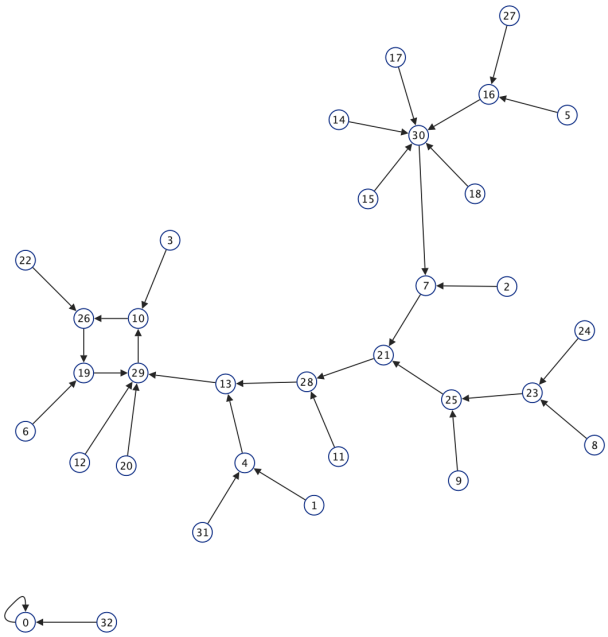


Figure 2.3: 單峰映射在 $r = \frac{122}{2^5}$ 之狀態映射網路

混沌映射的理論背景是建立在數學的連續性上的。以單峰映射為例，單峰映射所迭代的值會在 $[0, 1]$ 之間，若想把混沌映射實作在有限狀態機 (Finite-state machine) 上，就勢必得承受有限精度 (finite precision) 所帶來的動態衰退

doi:10.6342/NTU202201736

(dynamical degradation)[9–11]，而這點也正好是本篇論文主要的攻擊面向的立論基礎：利用混沌映射在有限精度的限制下，能夠將其在各個參數值所對應到的有限狀態機行為描繪出來。[12–14] 分析了混沌映射在有限精度下的有限狀態傳遞行為，透過理論證明了混沌映射在固定精度時，參數值會等價對應到一個狀態映射網路 (state-mapping network)。

圖 2.3 為單峰映射在精度為 5 時之定點數運算 (fixed-point) 的狀態映射網路，每個節點 (node) 代表的是 $[0, 1]$ 之間的初始狀態，有限精度的條件會把 $[0, 1]$ 的連續區間硬生生轉換成離散的集合 $\{x \mid x = \frac{i}{2^n}\}_{i=0}^{2^n}$。而每條邊 (edge) 代表該狀態代入單峰映射後的下一個值 (狀態)。由圖 2.3 可以看到單峰映射在多次迭代後會進入一個長度為 4 的迴圈 (cycle) 中，或者自循環 (self-loop) 中，這就是動態退化最直覺的呈現，也是我們本篇論文能夠執行攻擊的最大依據。

2.4 攻擊者模型 (Adversary model)

本篇論文以攻擊者角度出發，我們替攻擊者的能力做了以下定性的假設：

(1) 攻擊者有辦法得知使用者加密時所使用機器之精度

(2) 攻擊者有一定的機器學習背景知識

(3) 攻擊者有竊聽使用者傳送加密資料之部分混沌序列的能力

doi:10.6342/NTU202201736

第三章　訓練集生成演算法

這個章節將會討論我們如何人為生成訓練集，不需要在現實世界中蒐集資料，因此，攻擊者是有辦法自行預訓練一個圖神經網路的攻擊模型。我們認為既然每個參數可以等價對應到一張狀態映射圖，當然有機會利用機器學習的方法，讓機器去學習每個參數之狀態映射圖的拓樸結構。當攻擊者成功竊聽到一部份的混沌序列 (即狀態映射圖的子圖) 時，機器有辦法透過訓練集中的歷史資訊，成功猜出該混沌序列所對應到的參數。這種攻擊手法最大的優勢在於攻擊對象不僅限於某個特定的混沌映射影像加密演算法，而是對所有使用單峰映射的影像加密演算法都可以進行攻擊，也是促使我們進行相關研究的最大動機。

3.1　可接受區間

我們都知道機器學習所訓練的模型在做迴歸時一定會存在誤差，而誤差在混沌映射中又會導致迭代結果完全不同。但上述的性質是混沌映射在理論上 (即連續域中) 存在，實際上，當我們需要去模擬混沌映射時，必然要使用有限狀態機去進行模擬，而這時就產生了理論上和實際操作上的差異。也就是有了這個前提，使我們做完迴歸得到的參數值若落在一定區間內即可以被視為是同一個參數值 (此現象與信號經有限精度量化時的狀況類似)，而本篇論文中把這個區間稱作可接受區間 (tolerable region)。在介紹可接受區間前，先講解我們所假設使用

doi:10.6342/NTU202201736

的有限狀態機是按照定點數運算格式進行的；而捨入誤差的部分是採用四捨五入 (rounding) 方式。簡單來說當 $n = 5$，即該有限狀態機有 $\{x \mid x = \frac{i}{2^5}\}_{i=0}^{2^5} = 33$ 個狀態 (state)，則 $[0, 1]$ 的實數區間可以被劃分成 33 個點，第一個點為 $\frac{0}{2^5} = 0$，第二個點為 $\frac{1}{2^5} = 0.03125$，第三個點為 $\frac{2}{2^5} = 0.0625$。可接受區間的定義即為 $[\frac{(\frac{x-1}{2^n}+\frac{x}{2^n})}{2}, \frac{(\frac{x}{2^n}+\frac{x+1}{2^n})}{2}]$，落在此區間內的值在同個有限狀態機下都被視為是 x。因此先前提到單峰映射只有當其參數 r 在 $[3.6, 4]$ 範圍擁有混沌的性質。因此我們在 $n = 5$ 時，將參數分成 $\frac{114}{2^5} \sim \frac{128}{2^5}$ 的 14 個參數區間。

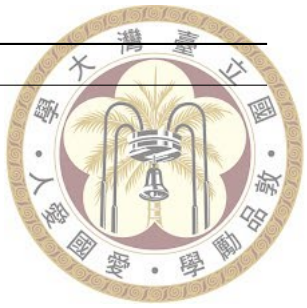
3.2 訓練資料集生成

前一個段落中，我們談到混沌映射的參數可以被劃分成一個個的可接受區間，而每個參數區間又等價對應到一張狀態映射圖，現在必須決定如何在每個狀態映射圖中去選擇初始狀態以組成金鑰對當作訓練集。我們的目的在於模仿使用者加密影像時的動作，然而如何有效的對初始狀態進行採樣即是本章所介紹之演算法要解決的問題。我們希望提出的演算法能達到以下三個目標：

1. 希望模擬使用者在設定金鑰對時會有偏好的選擇
2. 希望所有採樣所形成的混沌序列能夠盡可能覆蓋整個原圖
3. 希望採樣所形成的混沌序列可以盡可能不要產生太多重複

我們所設計的初始狀態採樣程序如演算法 1 所示。

doi:10.6342/NTU202201736



Algorithm 1 Initial states sampling algorithm

Initialize a empty list SP
for parameters in $[3.6, 4]$ **do**
  Initialize an empty list C
  GenerateSMN()
  CalculateMPL()
  Average_len = GetMeanofMPL()
  Initialize a boolean array V
  while $\frac{sum(V)}{len(V)} < gv$ **do**
    candidate, value = GetMaxofMPL()
    if value == 0 **then**
      break
    end if
    if candidate == 0 or 2^n **then**
      MPL[candidate] = 0
    end if
    if CalCleanPro() < cp or value < Average_len **then**
      MPL[candidate] = 0
    else
      C.append(candidate)
      VisitfromCand()
      MPL[candidate] = 0
    end if
  end while
  SP.append(C)
end for

演算法 1 中，SP 代表所有採樣點集合，C 為某個參數區間所屬的採樣點集合，V 紀錄下圖中的點是否有被拜訪過。gv 和 cp 是可以人為彈性調整的參數，gv 具體意義為圖中有多少比例的點已經被拜訪過，本演算法的其中一個核心思維

doi:10.6342/NTU202201736

是想讓機器能夠在不接收太多原狀態映射圖資訊的前提下，透過其骨架 (skeleton) 學習到該圖的大致拓樸結構，因此我們在演算法中選擇將 gv 設定為一個參數以便做彈性的調整，希望能跟暴力破解 (brute-force) 做個明顯的區分。cp 則是檢查候選點是否對目前的狀態映射圖有所貢獻，我們希望可以利用最少的採樣點去囊括最大面積的狀態映射圖。

3.2.1 GenerateSMN 函數

此函數會依照精度和給定的參數值將有限狀態機中每個狀態輸入單峰映射，再依照輸入和輸出狀態建立狀態映射圖 (SMN)，另外把 indegree 為零的點記錄為葉節點 (leaf-node)。

3.2.2 CalculateMPL 函數

此步驟會根據上一步所產出的狀態映射圖，去計算每個葉節點的最大路徑長度 (Max period length：MPL) 並記錄在 array MPL 中，非葉節點的 MPL 則設為 0。此部分的靈感來自 [15]。[15] 將狀態映射圖中的節點進行分類形成一個樹狀結構，並設計了一個快速計算路徑長度的演算法以分析不同精度下狀態映射圖的安全性。

3.2.3 GetMeanofMPL 函數

此函數回傳 array MPL 之平均值，用於限制候選點的條件，若候選點的 MPL 小於平均值，則被視為沒有貢獻的點。

doi:10.6342/NTU202201736

3.2.4 GetMaxofMPL 函數

此函數回傳 array MPL 之最大值和指標 (index)，若有多個節點值為最大值，則優先選取節點指標最小者，此為候選點，接著要檢查候選點是否滿足條件，才能被當作採樣點。

3.2.5 CalCleanPro 函數

計算候選點在目前的狀態映射圖中所貢獻的路徑乾淨比例，也就是未拜訪的節點個數 / 總路徑長。

3.2.6 VisitfromCand 函數

從候選點開始進入狀態映射圖中迭代，並且將走過的節點在 V 中設為 1(即為已拜訪)。

3.2.7 分配參數值

在 Algorithm1 中，我們已經完成每個參數區間的初始狀態採樣，機器會依照給定的 gv 和 cp 決定每個參數區間需要多少採樣個數。下一個步驟要把每個參數區間的初始狀態採樣和參數值進行分配以組成金鑰對。這部分我們選擇將參數區間進行高斯分佈的採樣，雖說對於固定精度的有限狀態機而言這些參數值都屬於同個值，但就使用者角度而言是沒有這個資訊的。因此，我們希望模仿使用者的行為，也替資料集中金鑰對的標籤增加多樣性。到這邊訓練集資料就已經算是完成了。組好的金鑰對相當於標籤 (labels)，將此金鑰對輸入有限精度下的單峰映射得到的狀態映射子圖即為特徵 (features)。

doi:10.6342/NTU202201736

3.3 測試資料集生成

測試資料集生成就相對簡單，我們在生成訓練資料集後計算訓練集的個數。生成固定比例的測試資料集，在測試集中我們將每個參數區間的初始狀態採樣做隨機選點，但候選點之 MPL 必須滿足 $MPL[candidate] > max(MPL) * 0.9$。此條件使我們產生的測試集是相較使用者隨便選擇初始條件來說，安全性高上很多的。因此我們堅信在這個測試集上若表現得好，應該能夠一定程度上代表攻擊模型相當成功。

doi:10.6342/NTU202201736

第四章　模型與實驗設計

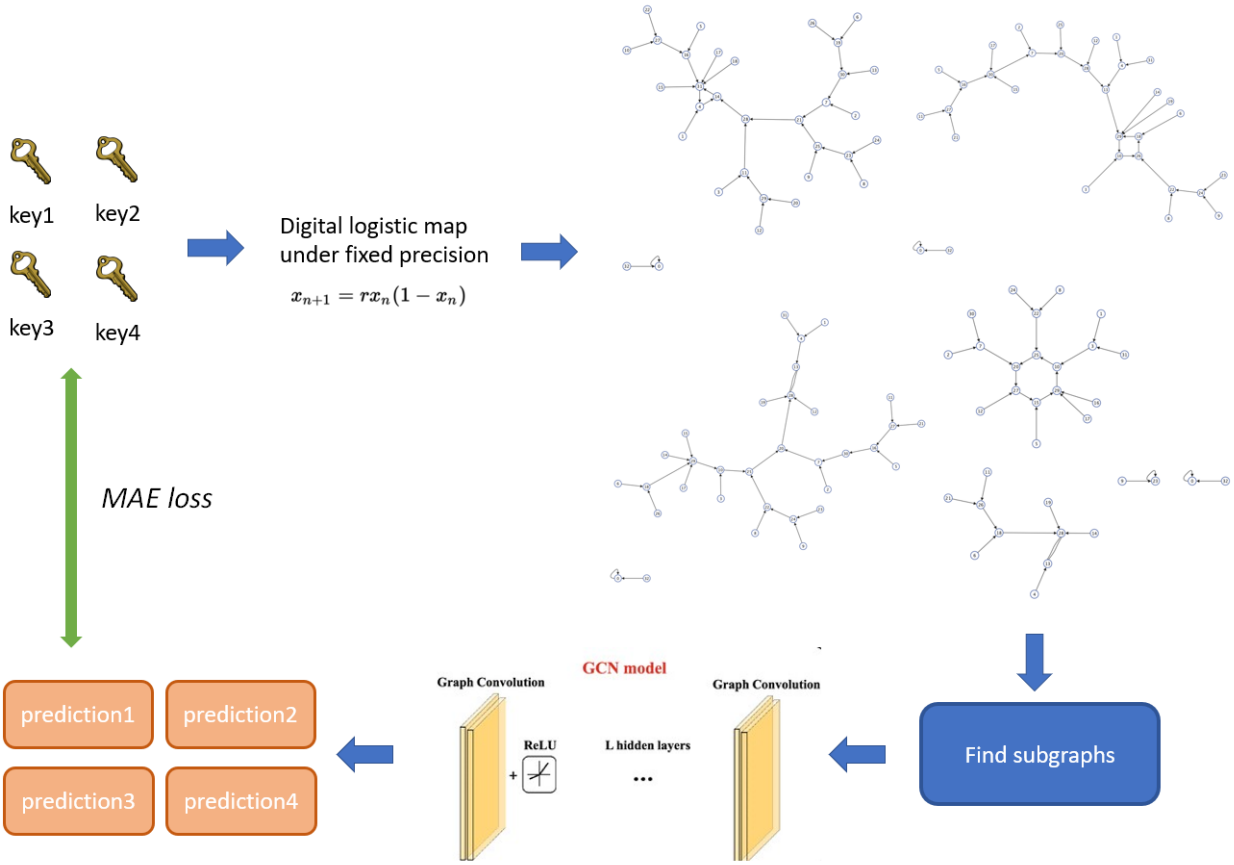


Figure 4.1: 我們提出的參數迴歸攻擊模型之訓練架構

此章節我們將會探討有關實驗的細節和我們如何設計模型以及背後的動機。如圖 4.1 所示，我們將利用多個金鑰對產生狀態映射子圖當作輸入，去訓練一個圖神經網路的模型，希望該模型可以學習到低維度的表示，再透過全連接層進行參數迴歸的學習。廣義的來說，可以將我們的攻擊架構視為一種選擇明文攻擊。我們透過自行實作加密器和使用多個金鑰試圖去找出統計上的規律，差別在於我

doi:10.6342/NTU202201736

們並不是從加密後的影像去分析，而是直接從加密產生的混沌序列進行分析。

4.1 圖神經網路模型

4.1.1 版本一 (Model 1)

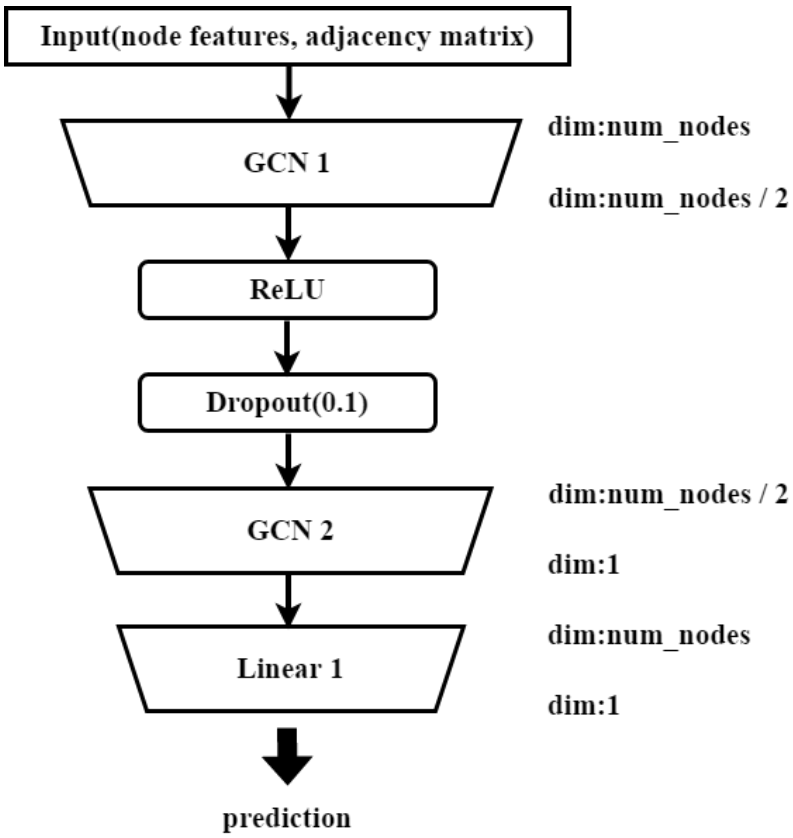


Figure 4.2: 模型版本一

在版本一中，因為我們是先從精度較小的資料集開始攻擊。因此，一開始的模型我們並沒有設定的很複雜，我們參考了 [16] 的架構，僅僅用兩層 GCN[17] 對我們的單峰映射圖進行嵌入，希望模型能從中學習到如何用低維度的型態表示。維度的部分在 GCN layer 1 我們選擇降至原維度的一半，GCN layer 2 再將維度降至 1，到這裡的特徵向量都還是以節點形式呈現 (node-level)。我們接著將學習到的潛在向量透過線性層希望將學習到每個節點的一維表示轉為圖層面 (graph-level) 的一維表示，即最終預測的回歸參數值，再將此回歸值與最初之金鑰對的參數值

doi:10.6342/NTU202201736

計算絕對平均值誤差 (MAE loss) 進行反向傳遞。

4.1.2 版本二 (Model 2)

在版本一中我們發現了幾個問題:

(1) 當精度往上提升時，節點數量和鄰居矩陣皆以指數方式成長。而 GCN 以矩陣相乘的計算方法會導致圖形處理器 (GPU) 記憶體不足和計算時間過長

(2) 維度選擇和模型的架構上似乎過度限制了機器的能力

為了解決記憶體不足的問題，我們參考了 GraphSAGE[18] 以取代 GCN layer。GraphSAGE 主要的優勢在於它對於整張圖的節點更新並不是一起做的，而是改用小批次 (mini-batch)，一次更新一部分的節點。且在更新節點的時候並不把所有鄰居都考慮進來，只考慮一部分的鄰居 (可彈性調整)，以減少計算量。在版本二中我們更改了直接進行迴歸的動作，取而代之的是，使用先分類後迴歸的方法，主要是參考 [19] 這篇論文。[19] 進行了人臉預估年齡的任務，該任務的目標其實與本篇論文相近，差別只在使用的資料屬性不同。其將年齡分成 100 個類別 (class)(在本篇論文中類別數相當於固定精度下的參數區間個數)，透過機器進行機率分布的計算後將各類別的機率乘上各類別的標籤去計算期望值以進行迴歸。最後在維度的選擇上，因為版本一的維度似乎過小，限制了機器的學習，我們將 GraphSAGE layer 1 的維度設定從原維度至目標參數區間的兩倍，並將 Dropout 的比例調整為 0.4，讓機器有更高的自由度去決定哪些神經元 (neuron) 應該留下。GraphSAGE layer 2 則保持維度不變的圖嵌入 (graph embedding) 學習。接著透過一個線性層將每個節點的向量維度降至參數區間的個數，到這個步驟每張圖都還是節點形式的呈現。我們再透過一個池化層 (max pooling) 將其轉換至圖形式的表

doi:10.6342/NTU202201736

示，再將每張圖經過歸一化指數函數 (softmax) 計算出機器認為該資料屬於每個區間的機率，最後乘上每個參數區間對應到的標籤去計算期望值。

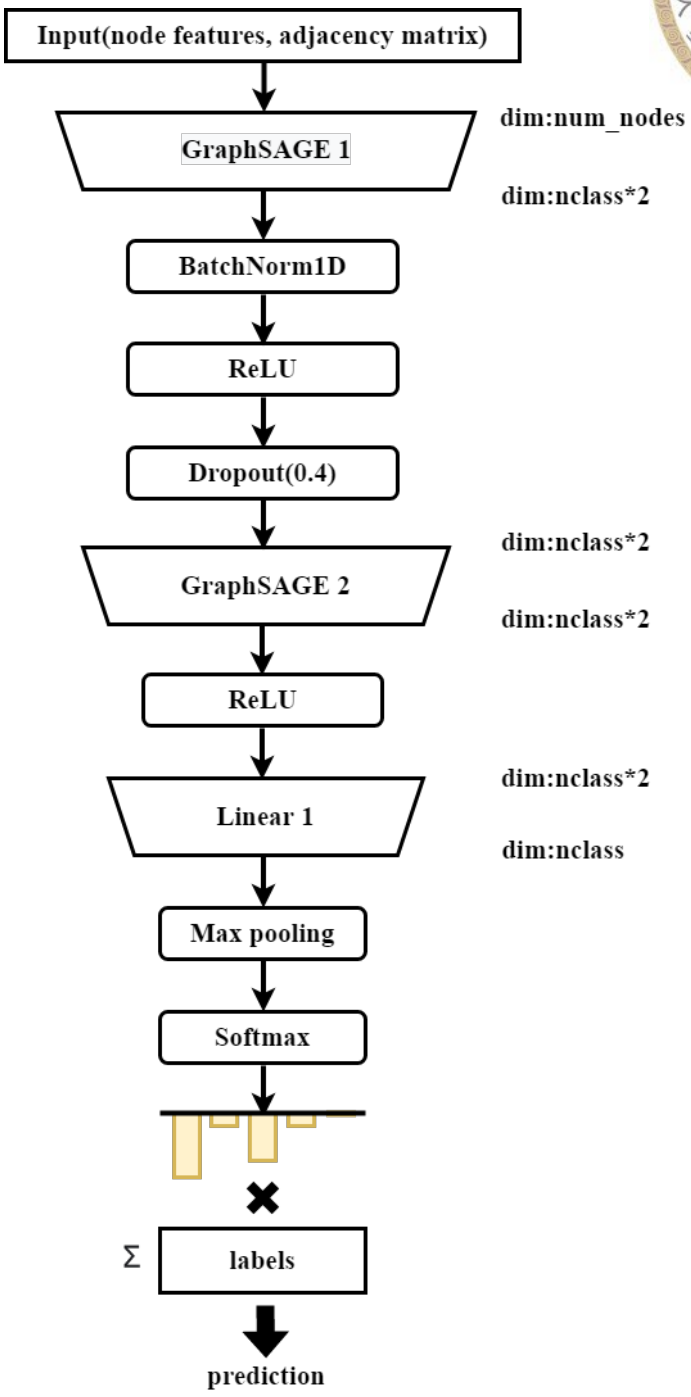


Figure 4.3: 模型版本二

doi:10.6342/NTU202201736

第五章　實驗結果

5.1　資料集設定

為了觀察我們所設計的資料集生成演算法是否有效，在這章中我們為每個精度生成一份隨機選擇初始狀態的資料集做為對照組，每個參數區間所對應的映射狀態圖不再自行決定要採樣多少初始狀態，而是預先設定每個參數區間都取固定個數的採樣點，而這個參數是由資料集大小決定，我們盡力去生成一個和我們提出的方法相同大小的對照資料集。這個段落會展示我們兩份資料集生成的設定還有相關的數據，精度分別由 5 ～ 11 位元，在表 5.1 中 “Sample number” 代表每個參數區間採樣幾個初始狀態。從圖 5.1 可以看到使用我們提出的採樣演算法生成所花費的時間隨著精度提高呈指數成長，但整體花費的時間仍在可以接受的範圍內。

doi:10.6342/NTU202201736

Table 5.1: 隨機採樣資料集 (Random-set)

Precision(bits)	Sample number	Trainset size(numbers)	Cost time(secs)
5	4	56	0.1433
6	6	162	0.0805
7	7	371	0.6368
8	9	945	1.8687
9	11	2299	2.8445
10	11	4587	10.9528
11	16	13328	41.8899

Table 5.2: 採樣演算法生成資料集 (Algorithm-set)

Precision(bits)	gv	cp	Trainset size(numbers)	Cost time(secs)
5	0.5	0.15	51	0.1766
6	0.5	0.15	157	0.6968
7	0.5	0.15	387	3.9900
8	0.5	0.15	951	14.8796
9	0.4	0.15	2344	82.1048
10	0.2	0.15	4635	370.3282
11	0.2	0.15	13647	2005.2871

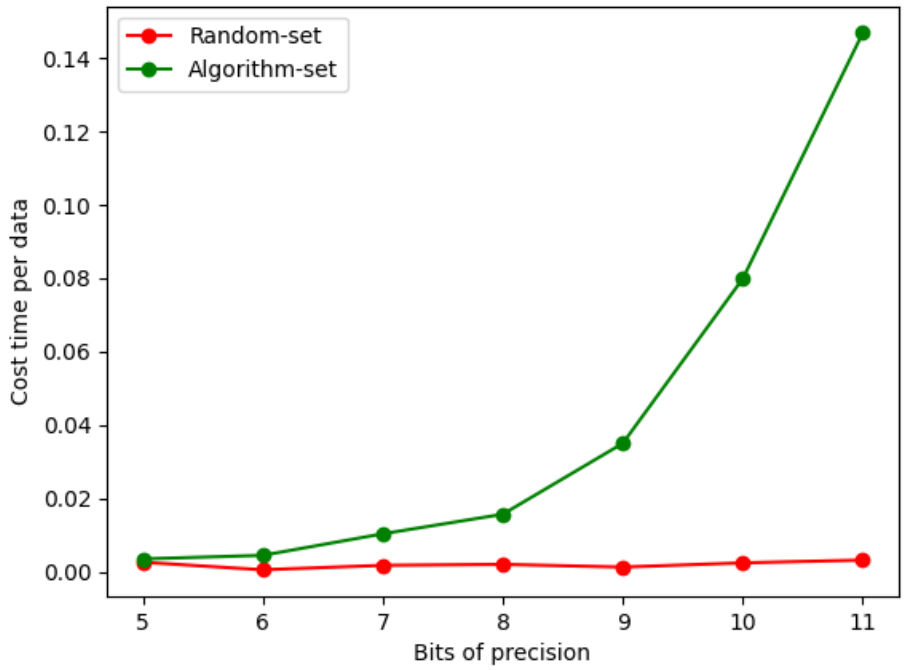


Figure 5.1: 訓練資料集生成所需時間 (單位：秒)

doi:10.6342/NTU202201736

5.2 模型一實驗結果

5.2.1 不同精度下的學習曲線 (learning curve)

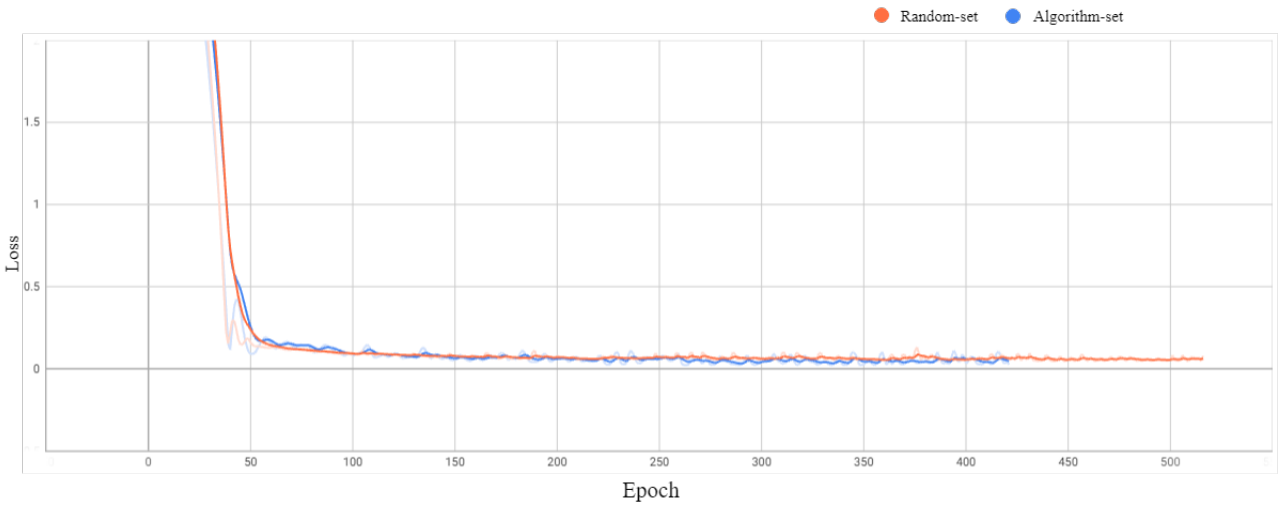


Figure 5.2: 模型一當 n = 5bits 時之學習曲線

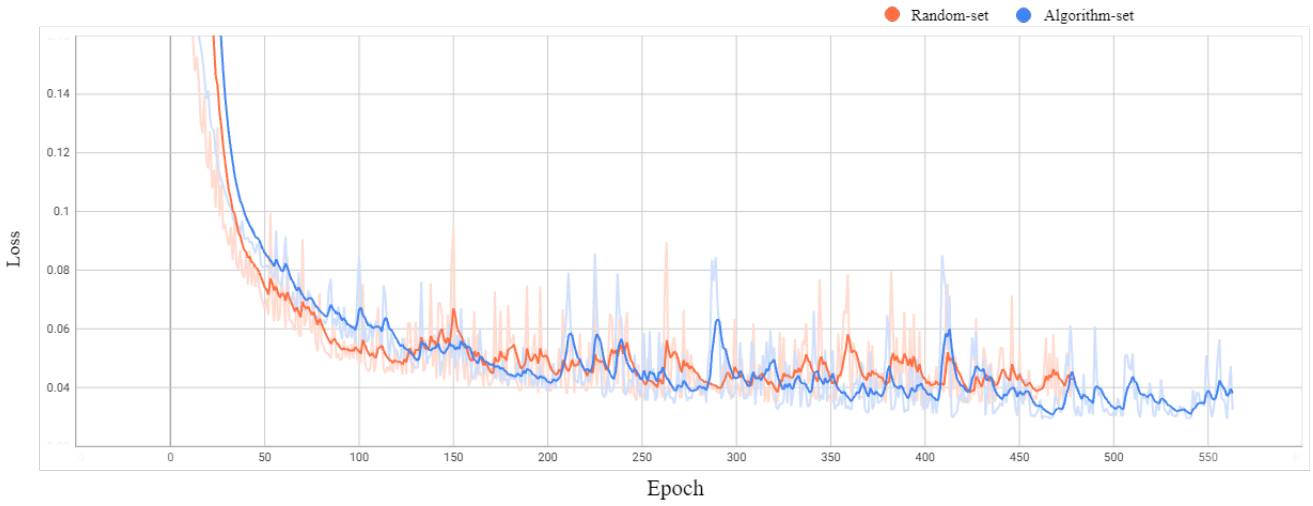


Figure 5.3: 模型一當 n = 6bits 時之學習曲線

doi:10.6342/NTU202201736

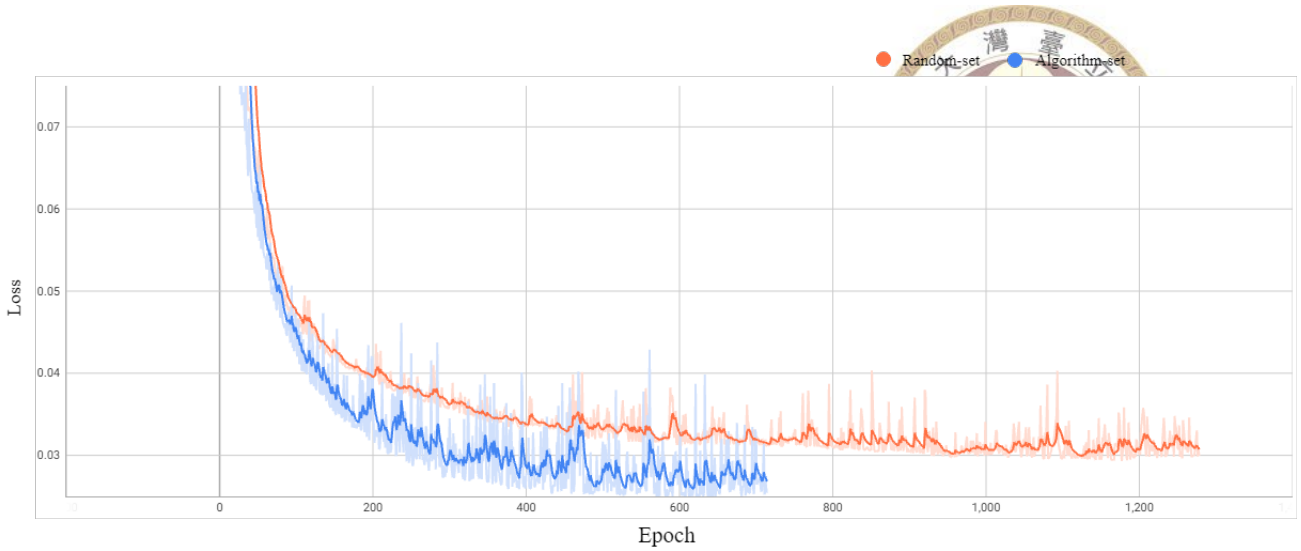


Figure 5.4: 模型一當 n = 7bits 時之學習曲線

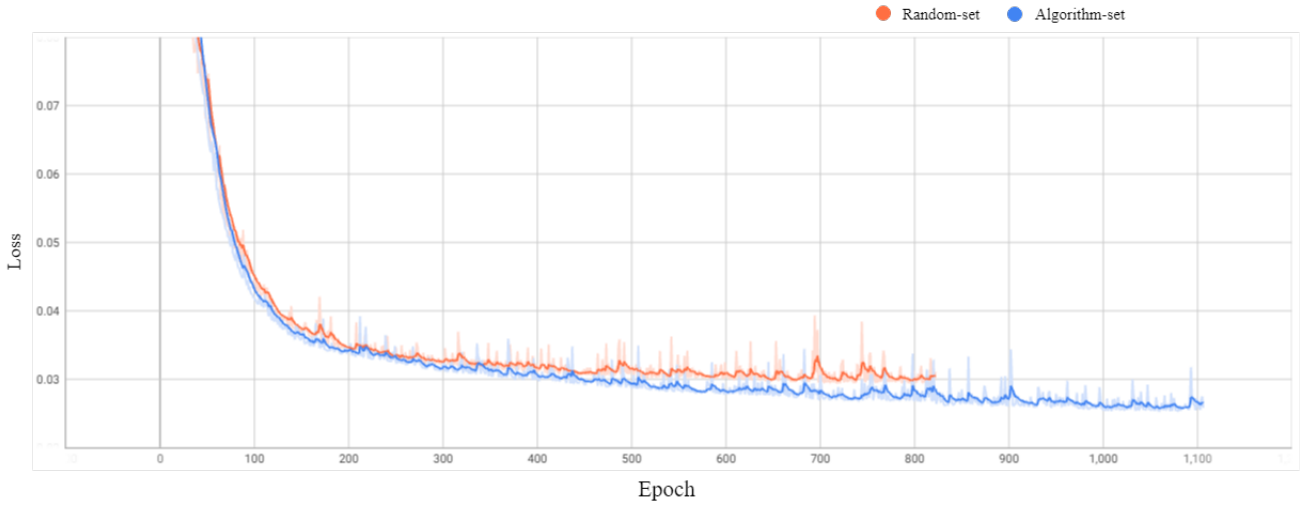


Figure 5.5: 模型一當 n = 8bits 時之學習曲線

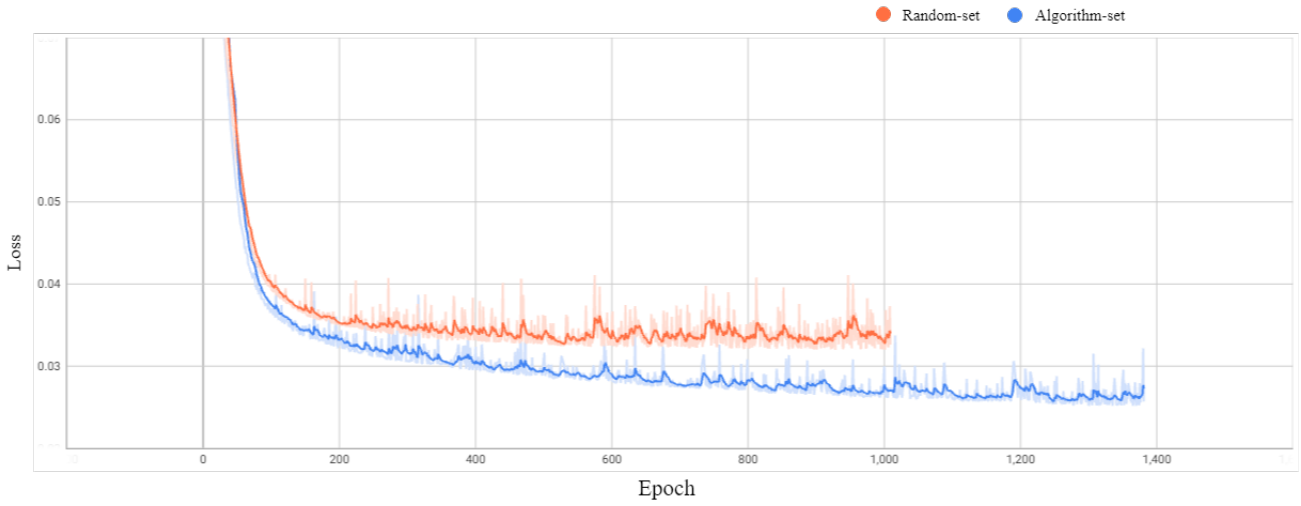


Figure 5.6: 模型一當 n = 9bits 時之學習曲線

doi:10.6342/NTU202201736

從 5.2.1 可以看到我們所建立的資料集與隨機採樣的資料集相比都能得到更低的 Loss，這與我們預期達到的表現相符，但受限於模型一的能力，Loss 表現仍然卡在 0.02 下不去，且模型一的 GCN layer 是採用矩陣相乘的作法，無法對 n = 10; 11bits 資料集中的高維矩陣進行訓練。

5.2.2 訓練趨於穩定後損失函數之比較

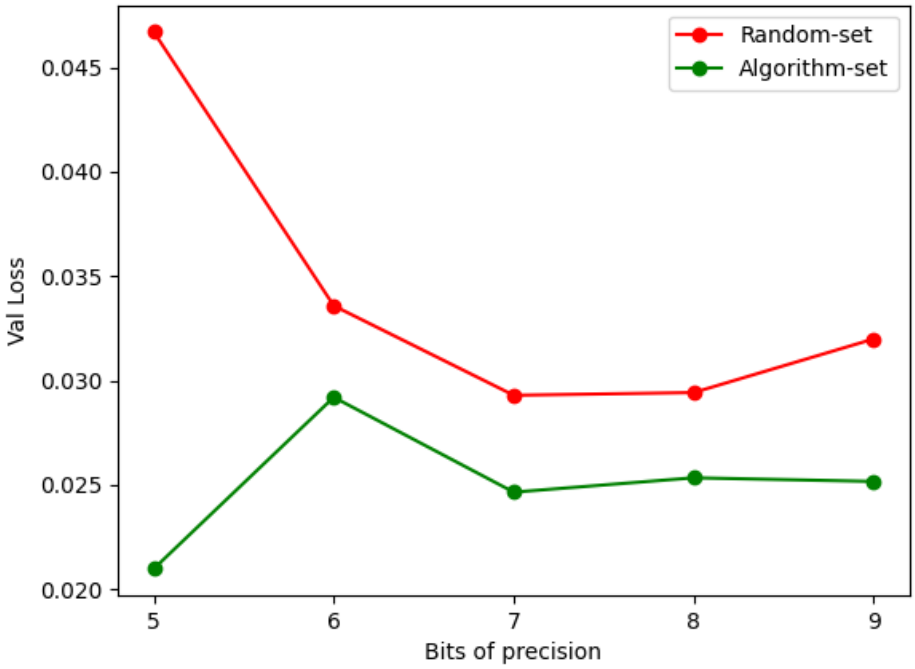


Figure 5.7: 模型一之驗證集損失函數

doi:10.6342/NTU202201736

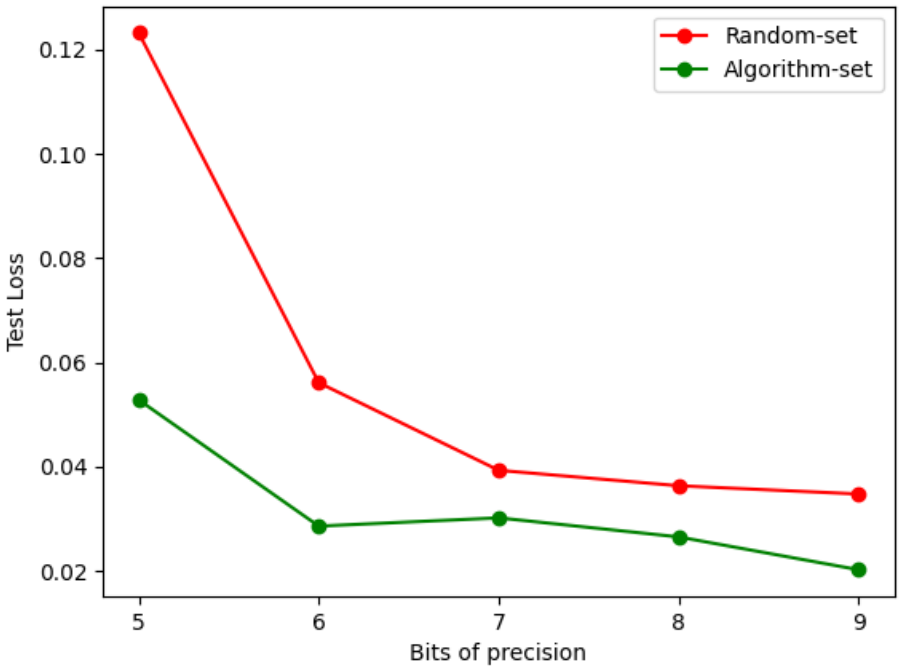


Figure 5.8: 模型一之測試集損失函數

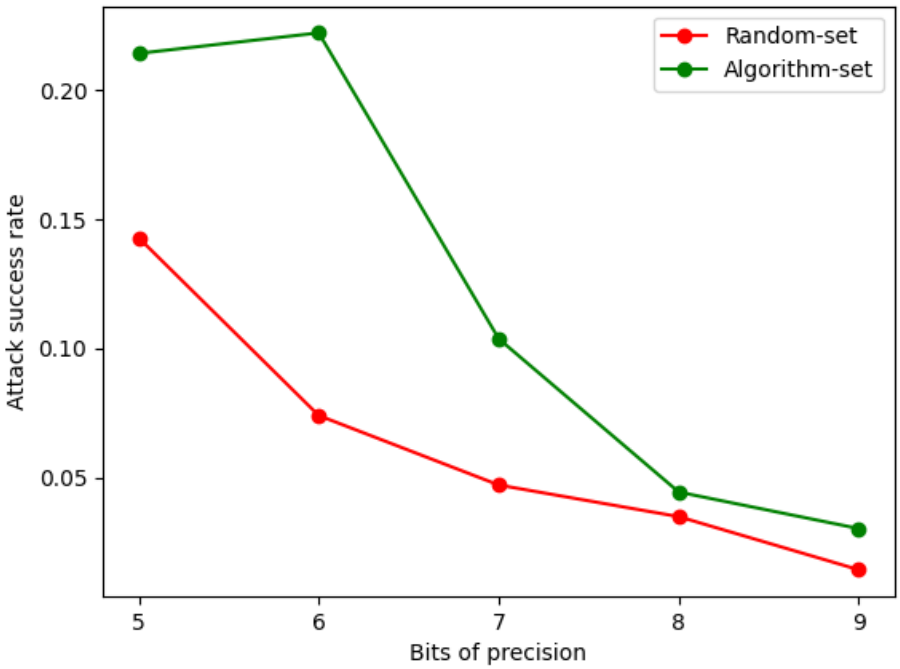


Figure 5.9: 模型一測試集攻擊成功率

在圖 5.8 中我們更能很明顯看出雖然我們所設計的資料集表現比隨機採樣好，

doi:10.6342/NTU202201736

但 Loss 幾乎都卡在 0.02 下不去。圖 5.9 也可以明顯看出攻擊成功率跟著精度上升呈指數下降。下個段落我們將呈現模型二的實驗結果，期望可以突破攻擊成功率不足和損失函數過高的窘境。

5.3 模型二實驗結果

5.3.1 不同精度下的學習曲線 (learning curve)

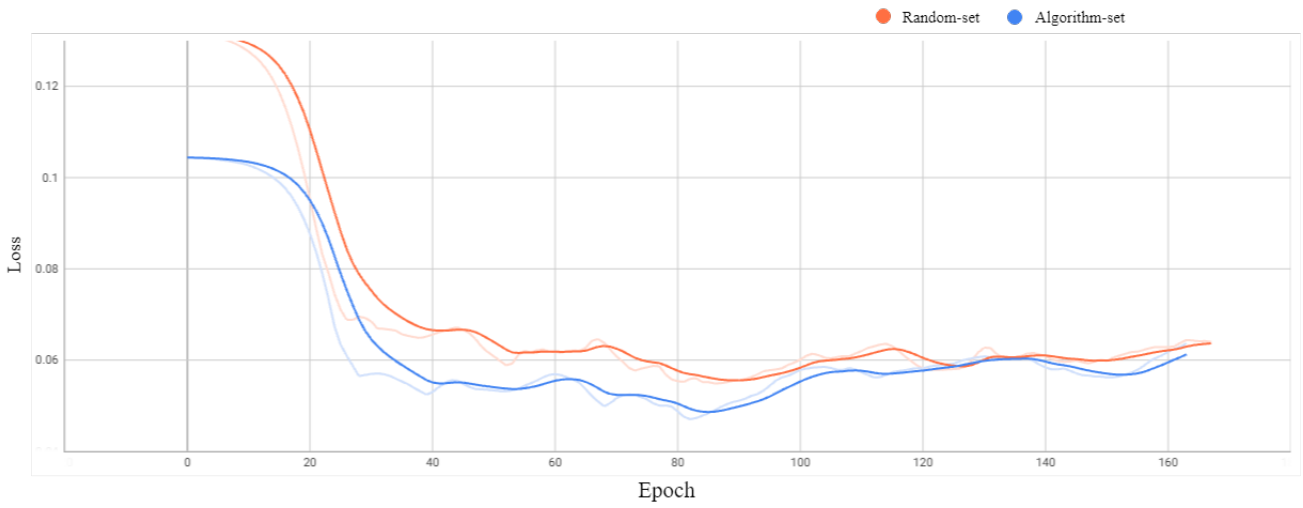


Figure 5.10: 模型二當 n = 5bits 時之學習曲線

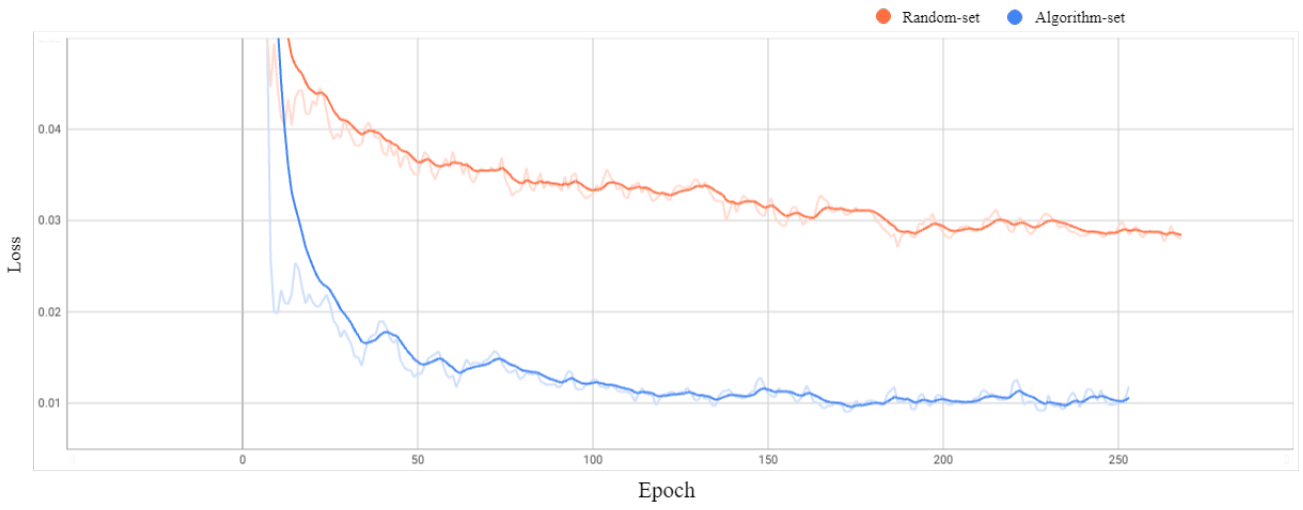


Figure 5.11: 模型二當 n = 6bits 時之學習曲線

doi:10.6342/NTU202201736

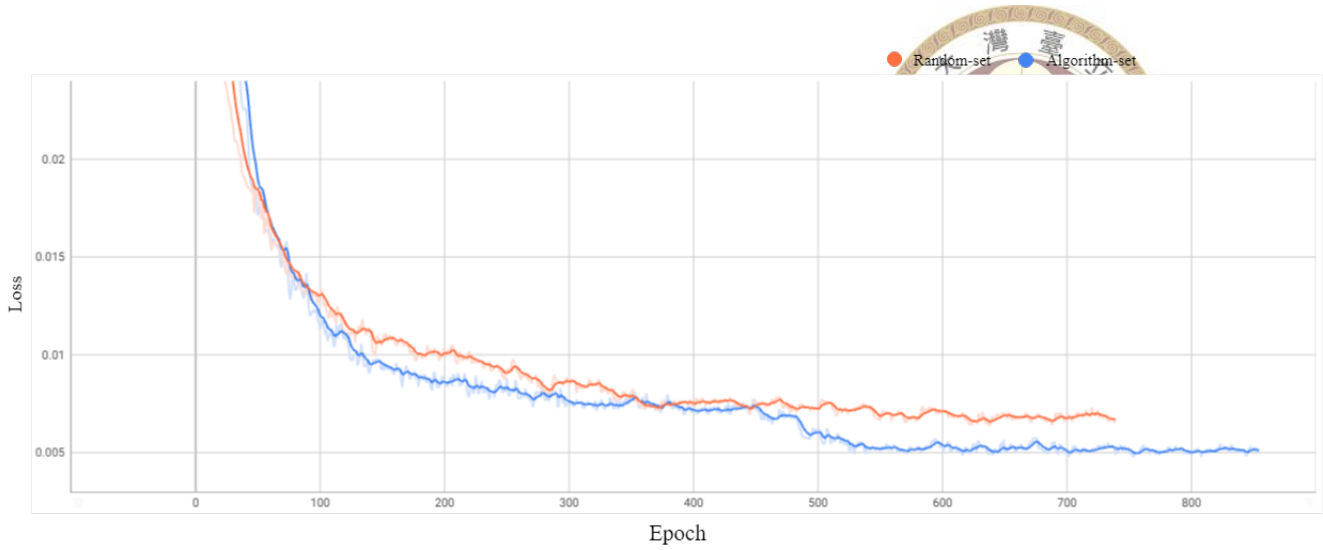


Figure 5.12: 模型二當 n = 7bits 時之學習曲線

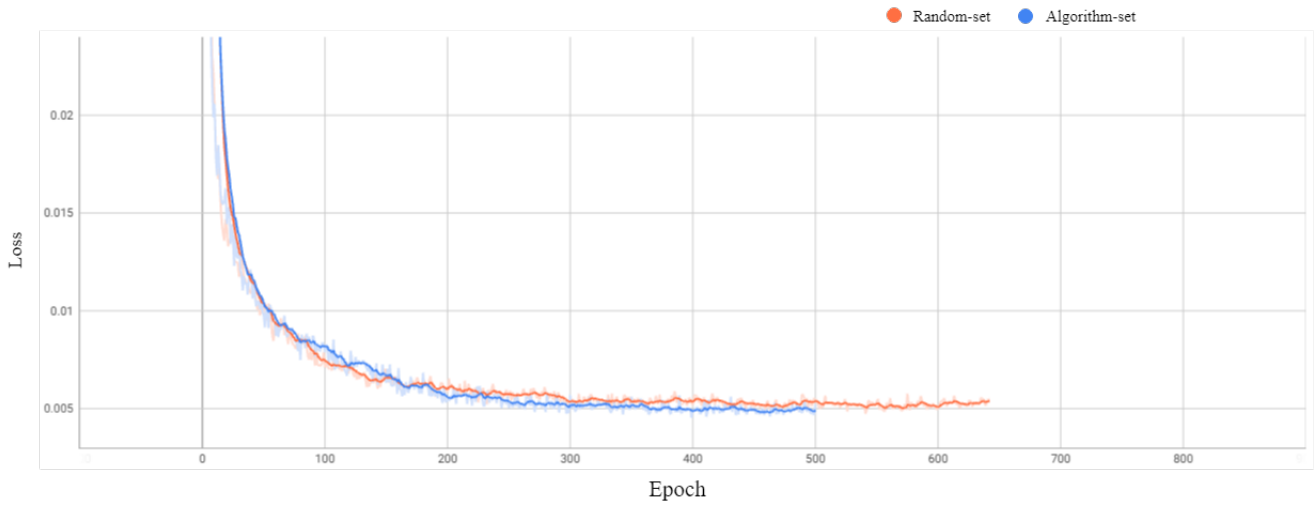


Figure 5.13: 模型二當 n = 8bits 時之學習曲線

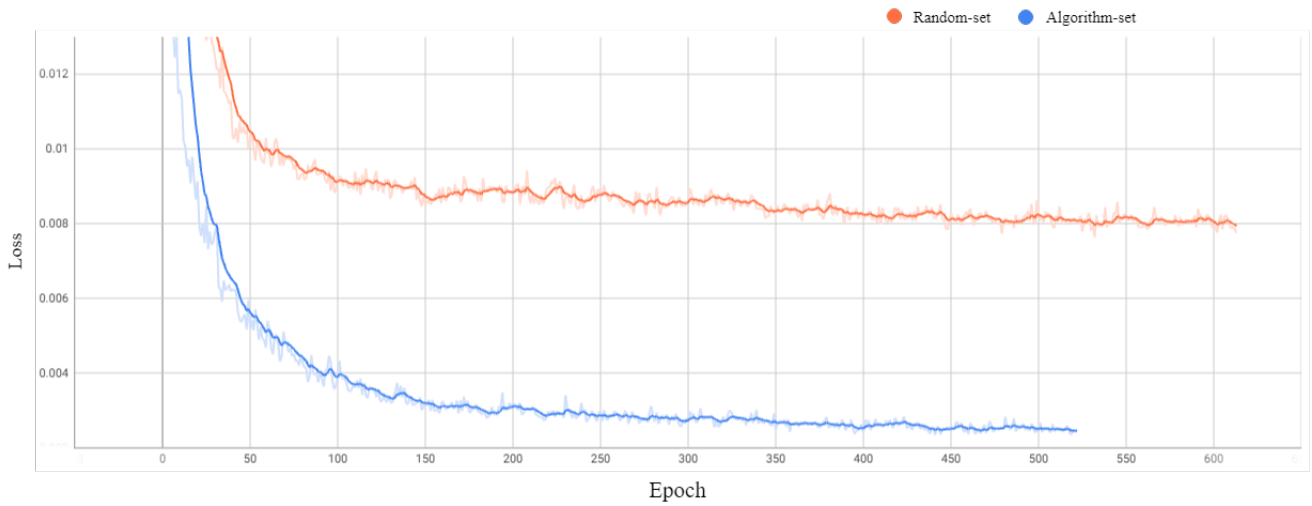


Figure 5.14: 模型二當 n = 9bits 時之學習曲線

doi:10.6342/NTU202201736

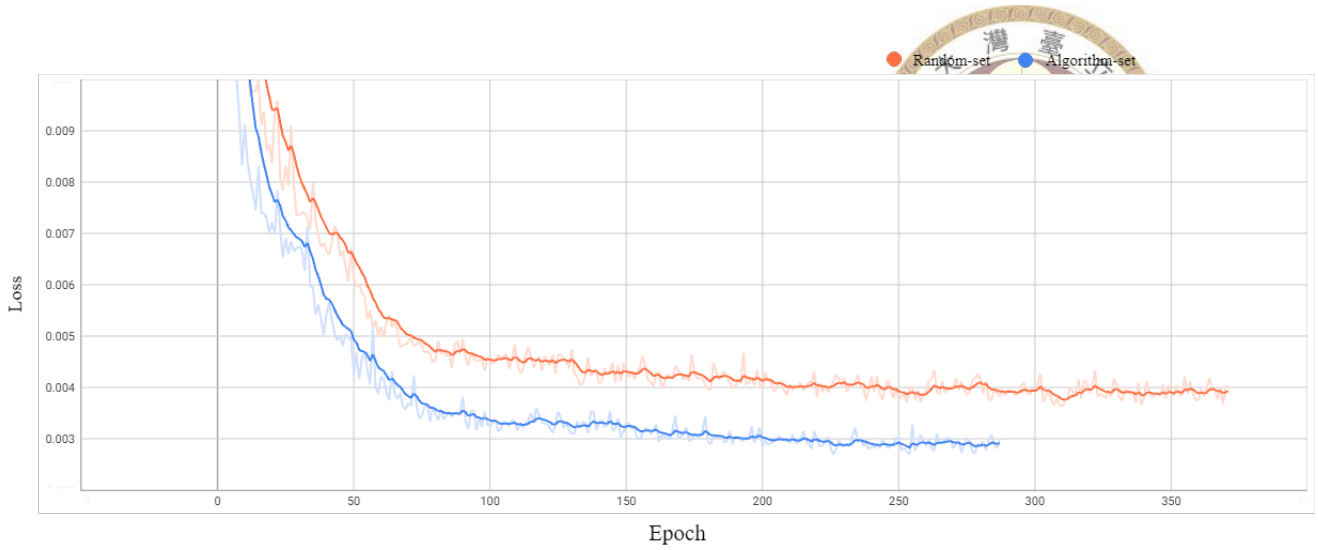


Figure 5.15: 模型二當 n = 10bits 時之學習曲線

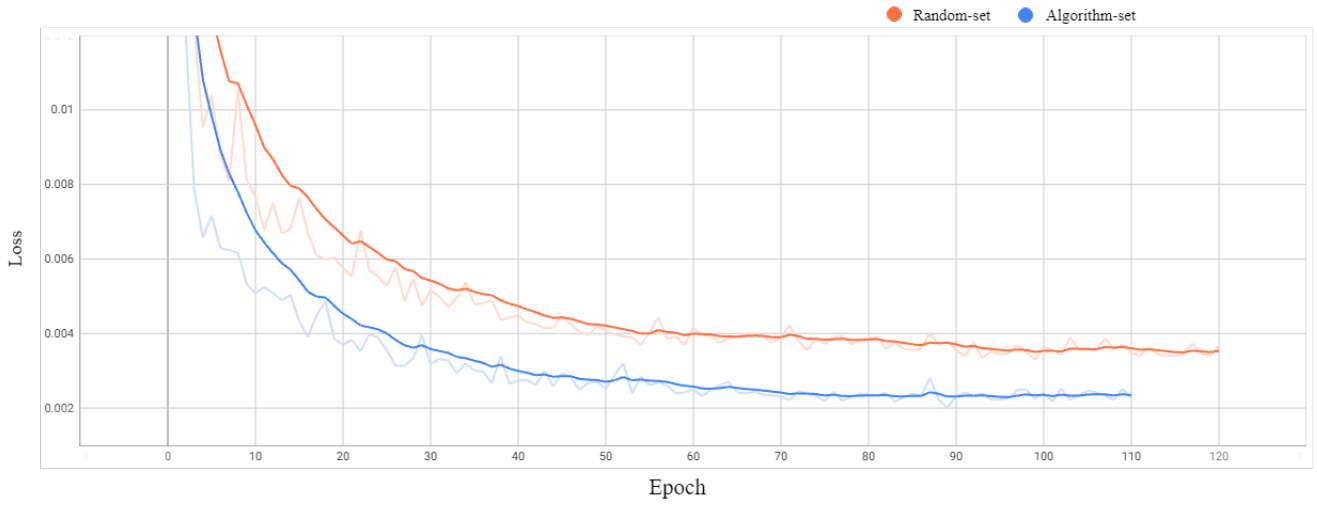


Figure 5.16: 模型二當 n = 11bits 時之學習曲線

5.3.1 節表明在不同精度底下，我們所設計的資料集表現仍然優於隨機採樣。且比起模型一，模型二表現更為突出，可以使 Loss 降到 0.002 左右，這樣的數據對於我們後面進行時間複雜度分析是一個很大的優勢。令人驚訝的是，隨機生成的資料集表現意外的好，我們認為這必須歸功於模型二良好的設計。此外，隨機生成資料集的採樣方法和測試集其實是一樣的，都是將每個參數區間進行平均分配採樣數。因此，視野拉遠來看其實隨機生成的資料集和測試集的參數標籤都會形成一個均勻分佈 (uniform distribution)，我們認為這可能是隨機生成資料集表現突出的原因。這個結果也相對說明了攻擊者不需要擁有針對特定混沌映射的背景知識，只要會訓練模型即可達到攻擊的目的，一方面降低了攻擊者的假設。

doi:10.6342/NTU202201736

5.3.2 訓練趨於穩定後損失函數之比較

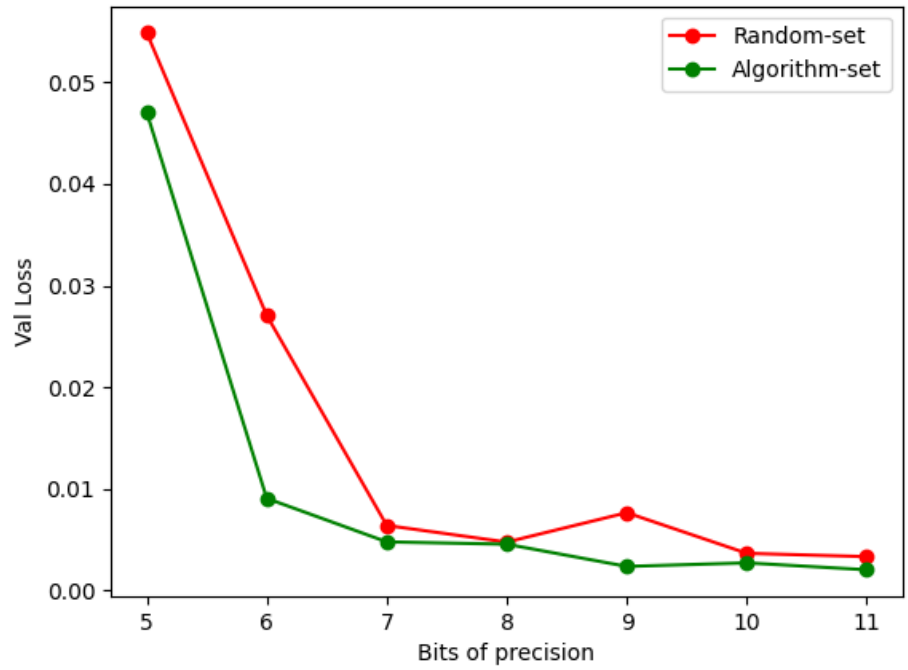


Figure 5.17: 模型二之驗證集損失函數

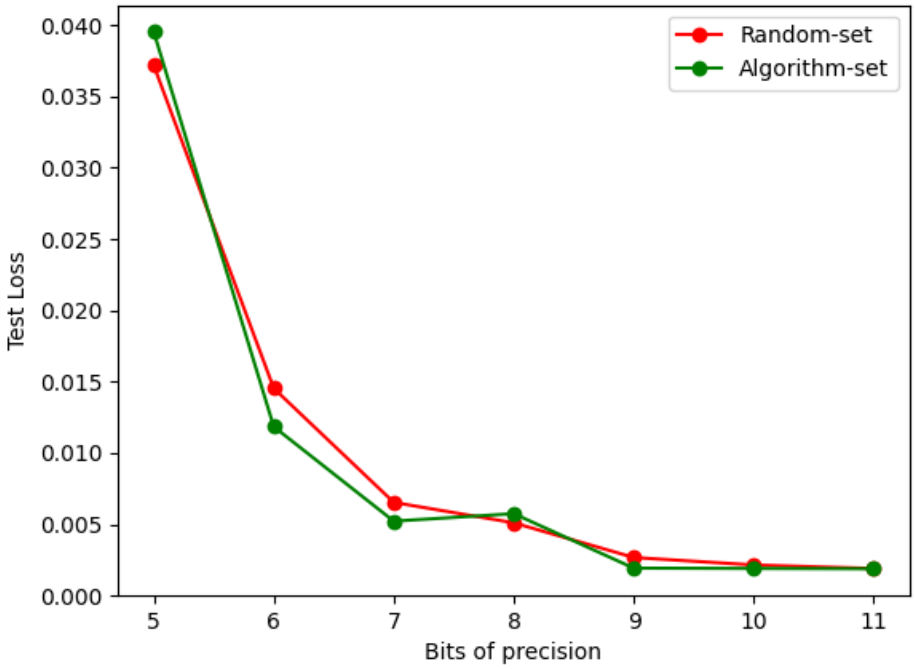


Figure 5.18: 模型二之測試集損失函數

doi:10.6342/NTU202201736

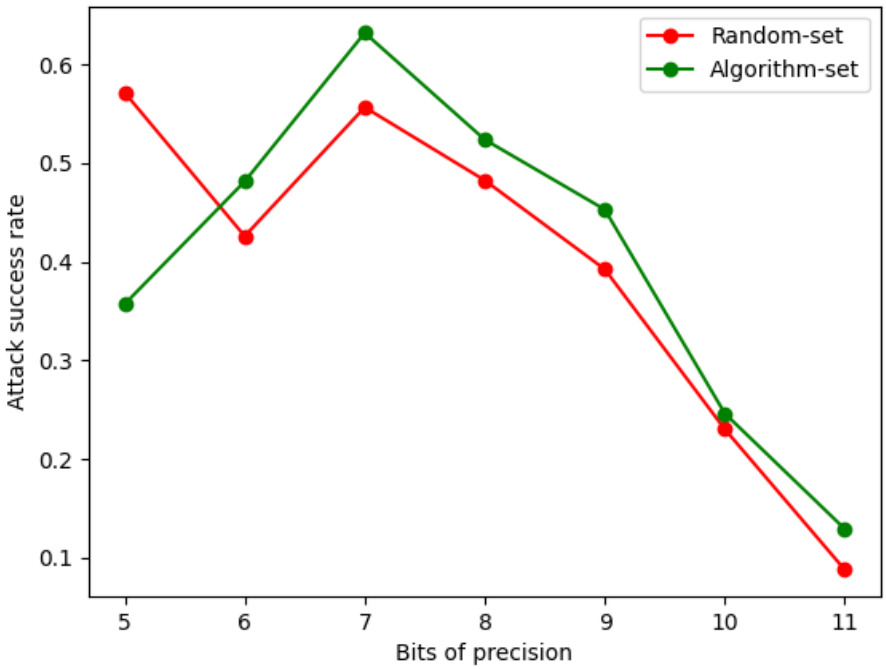


Figure 5.19: 模型二之測試集攻擊成功率

從圖 5.17 和圖 5.18 可以看到 n = 5 和 n = 6 時資料集還不夠多，當資料量變大時損失函數就降至 0.01 以下了。我們所提出的採樣演算法資料集仍在大部分時候表現優於隨機生成的資料集。從圖 5.19 可以發現模型二大大的提升了攻擊成功率，雖然隨著精度上升成功率還是下降的很快，但 Loss 部分仍然保持在一定的水平，這對我們在後續談到攻擊實作時的金鑰空間縮減會起到很大的作用。

5.4 模型二在資訊受限之表現

在這個段落我們想要討論若是測試集的迭代數 (邊的個數) 並不是全知道的情況下，會對成功率有怎樣的影響，這樣比較貼近我們一開始對攻擊者的假設。以下表中的 “Cut_Edge” 指的是砍到剩下多少比例的邊，0.9 代表剩下前 90% 的邊。從下面的圖表可以發現當邊切得越多時，成功率下降的越快，特別是在 0.8 之後迅速下降，我們認為是因為在 0.8 後大部分的資料都失去了迴圈 (cycle) 的資訊，

doi:10.6342/NTU202201736

因為每筆資料都會進入迴圈，在邊的資訊上機器有可能對於迴圈的權重較大，故當失去迴圈時，機器就相對失去一部份的能力。但在大部分的時候我們提出的資料集的表現仍優於隨機採樣資料集。

Table 5.3: Cut_edge(n = 5)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0371	**0.0395**	0.5714	**0.3571**
0.9	0.0475	**0.0434**	0.3571	**0.3571**
0.8	0.0514	**0.0434**	0.2857	**0.3571**
0.7	0.0503	**0.0463**	0.2857	**0.3571**
0.6	0.0556	**0.0690**	0.2857	**0.0714**
0.5	0.0671	**0.0720**	0.2142	**0.0714**

Table 5.4: Cut_edge(n = 6)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0270	**0.0118**	0.4259	**0.4814**
0.9	0.0173	**0.0140**	0.5185	**0.4444**
0.8	0.0183	**0.0160**	0.4444	**0.3888**
0.7	0.0199	**0.0157**	0.3518	**0.3148**
0.6	0.0275	**0.0282**	0.3148	**0.2777**
0.5	0.0476	**0.0419**	0.1666	**0.2222**

doi:10.6342/NTU202201736

Table 5.5: Cut_edge(n = 7)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0065	**0.0052**	0.5566	**0.6320**
0.9	0.0087	**0.0066**	0.4245	**0.5000**
0.8	0.0162	**0.0147**	0.2924	**0.4056**
0.7	0.0262	**0.0209**	0.1981	**0.2452**
0.6	0.0348	**0.0279**	0.1603	**0.2358**
0.5	0.0478	**0.0401**	0.1226	**0.1415**

Table 5.6: Cut_edge(n = 8)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0051	**0.0060**	0.4825	**0.5238**
0.9	0.0071	**0.0091**	0.3428	**0.3746**
0.8	0.0148	**0.0122**	0.2349	**0.2476**
0.7	0.0183	**0.0157**	0.1904	**0.1809**
0.6	0.0240	**0.0214**	0.1587	**0.1079**
0.5	0.0321	**0.0320**	0.1238	**0.0825**

Table 5.7: Cut_edge(n = 9)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0026	**0.0019**	0.3923	**0.4529**
0.9	0.0039	**0.0034**	0.2807	**0.3397**
0.8	0.0072	**0.0059**	0.1770	**0.2838**
0.7	0.0100	**0.0097**	0.1674	**0.2105**
0.6	0.0151	**0.0146**	0.1291	**0.1483**
0.5	0.0244	**0.0210**	0.0861	**0.1339**

doi:10.6342/NTU202201736

Table 5.8: Cut_edge(n = 10)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0021	**0.0018**	0.2302	**0.2462**
0.9	0.0047	**0.0037**	0.1814	**0.1910**
0.8	0.0071	**0.0057**	0.1430	**0.1574**
0.7	0.0107	**0.0102**	0.1143	**0.1366**
0.6	0.0144	**0.0126**	0.0807	**0.0991**
0.5	0.0189	**0.0192**	0.0583	**0.0631**

Table 5.9: Cut_edge(n = 11)

Cut_edge	loss(Random)	loss(Algo)	Attack rate(Random)	Attack rate(Algo)
1	0.0018	**0.0018**	0.0876	**0.1290**
0.9	0.0025	**0.0029**	0.1200	**0.1098**
0.8	0.0035	**0.0042**	0.0990	**0.1002**
0.7	0.0050	**0.0059**	0.0804	**0.0834**
0.6	0.0068	**0.0094**	0.0696	**0.0744**
0.5	0.0099	**0.0135**	0.0552	**0.0606**

5.5 *gv* 和 *cp* 在模型二對實驗各項指標的影響

這個段落將會討論 *gv* 和 *cp* 這兩個參數對於生成資料集所帶來的影響，其中包含生成所花費的時間，對機器學習上的效果等等。從表 5.2 就能看到我們一開始在生成資料集時，當精度上升我們將 *gv* 這個參數降低以減少生成時間，為了尋求模型二上更好的結果，接下來我們將會在精度 n=8 ~ 11bits 分別調整 *gv* 和 *cp* 兩個參數去觀察結果。

doi:10.6342/NTU202201736

5.5.1 固定 cp

Table 5.10: Fixed $cp = 0.15$

Precision	gv	Trainset size	Cost time	Loss	Attack rate
8	0.5	951	14.8796	0.0060	0.5238
8	0.6	951	15.9144	-	-
8	0.7	951	17.1409	-	-
9	0.4	2344	70.0148	0.0019	0.4529
9	0.5	2344	73.2149	-	-
9	0.6	2344	82.8059	-	-
10	0.2	4635	370.3282	0.0018	0.2462
10	0.3	5974	396.8634	0.0021	0.2765
11	0.2	13647	2005.2871	0.0018	0.1290
11	0.3	14859	2072.2073	0.0024	0.1050

從表 5.10 可以看到，當 cp 設定為 0.15 時，這個條件限制了採樣，導致 gv 不管調的再高，在 n = 8, 9bits 的時候都不影響採樣 (因 Algorithm1 中 line16 有一條件為若候選點的最大路徑長度 < 平均值，則放棄該點，認為該點是沒有貢獻的)。在 n = 10,11bits 時，因為更高的 gv 值不影響採樣就沒有展示在表上，到這裡可以發現其實 gv 值在我們的演算法中對時間上的影響其實不大，雖然執行時間有變長，但資料集大小也有一定程度上的增加。

doi:10.6342/NTU202201736

Table 5.11: Fixed $cp = 0.1$

Precision	gv	Trainset size	Cost time	Loss	Attack rate
8	0.5	2543	44.4987	0.0044	0.5523
8	0.6	3721	46.0312	0.0026	0.5968
8	0.7	4182	44.0672	0.0027	0.5333
9	0.4	5990	215.7035	0.0020	0.4545
9	0.5	9191	242.0484	0.0014	0.5326
9	0.6	11505	250.0089	0.0015	0.4162
10	0.2	6066	784.1016	0.0026	0.2262
10	0.3	12629	788.8826	0.0018	0.3101
10	0.4	21806	809.0084	0.0015	0.2565
10	0.5	29691	773.9325	0.0011	0.3077
11	0.2	21757	2031.0349	-	-
11	0.3	46245	2119.3409	-	-
11	0.4	73789	2091.2931	-	-
11	0.5	86890	2091.7836	-	-

表 5.11 我們將 cp 固定至 0.1，將條件放寬且將 Algorithm1 中 line 16 的第二個條件去掉，希望不會因此限制 gv 的影響。從表中我們可以看到資料集大小確實因為 gv 的增加而變大，且執行時間不太受 gv 影響，和表 5.10 相比反而是 cp 會直接決定了執行時間。cp 為 0.1 和 0.15 相比效果都好上不少，我們認為 cp 值應該得隨不同精度下的圖結構，設定不同的值才能達到最好的效果。觀察 n=8, 9bits 的表現可以發現當 gv 增加且資料數量增加時，Loss 有很明顯的下降，但對於攻擊成功率卻沒有單調遞增的趨勢。對此，我們給出的解釋是因為我們將 Algorithm1 中的條件去除，導致資料集中，但並不是所有資料都是重要的，可能一定程度上對機器造成了誤導。將條件去除的另一個缺點是在精度越大的情況下資料量增加的速度飛快，這對訓練時間造成一定的負擔，偏離了我們一開始設計此演算法的初

doi:10.6342/NTU202201736

衷，因此我們不建議將該條件去除，我們認為本篇論文提出方法之最大的優勢在於利用少量但貢獻度高的資料達到有效的攻擊，並減少攻擊所需要花費的時間。

5.5.2 固定 gv

Table 5.12: Fixed $gv = 0.5$

Precision	cp	Trainset size	Cost time	Loss	Attack rate
8	0.15	951	14.8796	0.0060	0.5238
8	0.125	1209	15.9683	0.0074	0.5238
8	0.1	1513	16.8358	0.0050	0.6476
8	0.075	2018	16.9521	0.0037	0.6571
9	0.15	2344	73.2149	0.0019	0.4529
9	0.125	2904	76.7273	0.0020	0.4194
9	0.1	3798	79.5670	0.0014	0.5183
9	0.075	5162	85.0566	0.0017	0.4609
10	0.15	5974	402.3244	0.0021	0.2765
10	0.125	7524	419.6067	0.0031	0.2414
10	0.1	9892	395.0996	0.0032	0.2629
10	0.075	13912	416.3629	0.0025	0.2925
11	0.15	14859	1997.15	0.0024	0.1050
11	0.125	19031	2050.3329	0.0018	0.1188
11	0.1	25357	2012.0805	0.0018	0.1452
11	0.075	36132	2033.2666	0.0034	0.0858

因為在表 5.11 和表 5.10 都可以看出 gv 在 0.5 時有比較穩定的表現，因此表 5.12 中我們固定 $gv = 0.5$，去觀察不同 cp 帶來的影響。我們可以看出隨著 cp 條件下降，伴隨而來的是機器認為有貢獻的初始狀態變多，資料集自然跟著增加。但值得注意的是在每個精度下，損失函數和攻擊成功率並不隨著資料集增加而跟

doi:10.6342/NTU202201736

著單調遞減和遞增，我們認為 cp 值應該要進行實驗才能找到每個精度下的最佳 cp 值。

5.6 實際攻擊案例及分析

這個段落我們將演示本篇論文在實際應用上的攻擊流程，並且針對我們所設計的攻擊方法和暴力破解作時間上的分析和比較。為了方便展示，我們參考 [8] 中的影像加密演算法架構，自行設計了一種基於單峰映射之影像加密演算法。此外，我們模擬使用者加密影像和攻擊者竊聽之後的攻擊操作，證明了我們所提出的方法確實能對任何金鑰對都能有效的減少金鑰空間，進而在短時間內攻擊成功。

5.6.1 簡易影像加密演算法

我們自行設計了一個簡易的影像加密演算法，主架構和圖 2.1 相同，我們設定 $n = 1, m = 2$，且使用單峰映射做為此演算法之偽隨機數生成器，至於 Confusion and Diffusion stage 則是採用 [8] 的方法。因為本篇論文主要攻擊的面向為混沌映射，而非加密演算法的架構，因此我們在架構設計上從簡，並且相信我們的攻擊方法在不同架構上都能成功。

Algorithm 2 simple CBIE algorithm

Input: Plainimage I; Key1 r; Key2 x_0

Output: Cipherimage C

1: GenerateCS(r, x_0)
2: **for** $m = 1$ to 2 **do**
3: Confusion()
4: Diffusion()
5: **end for**

doi:10.6342/NTU202201736

5.6.1.1 GenerateCS 函數

此函數會根據使用者輸入的金鑰對當作初始條件對單峰映射進行迭代生成相對應的混沌序列 CS，接著將 CS 的值進行調整以方便後續使用：$\lfloor CS \times 10^{-5} \rfloor \bmod 256$。

5.6.1.2 Confusion 函數

Confusion stage 會執行兩個動作：第一步先將整張影像在空間域做列的重新排列 (row rotation)，第二步則是行的重新排列 (column rotation)。分別按照 CS 的前 M 項 (影像長度) 和前 N 項 (影像寬度) 進行排序，即可按照排序前後索引 (index) 的位置進行重新排列以達到快速打亂影像資訊的目的。

5.6.1.3 Diffusion 函數

Diffusion stage 會提取 CS 的前 $M \times N$ 項 (影像總 pixel 數)，和經過 Confusion stage 的影像進行直接進行 element-wise XOR。

5.6.2 攻擊流程

Step 1: 使用者透過 Algorithm2 在有限精度的機器加密影像 (在本實測中設定為 11bits)，並將加密後的影像在公開頻道中傳送。下圖 5.20 和圖 5.21 分別是使用者原圖和加密後影像 ($r = 3.964156, x_0 = 0.4864156$)

Step 2: 攻擊者開始竊聽使用者加密時所使用的一部份混沌序列，在本次實測中，此比例設定為原混沌序列的 60%，以下稱此片段為 CS_{eav}。

Step 3: 攻擊者將 CS_{eav} 輸入至預訓練好的模型中，企圖從子圖中推敲出該狀態映

doi:10.6342/NTU202201736

Figure 5.20: Test image

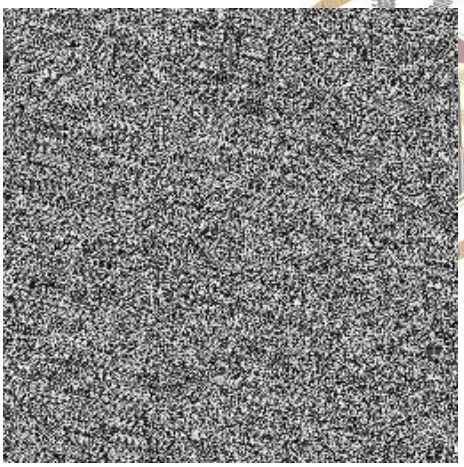


Figure 5.21: Cipherimage

射圖所對應到的參數 (即金鑰對中的第一把金鑰 r)，模型在此預測出的值為 $r_{pred} = 3.965381383$。

Step 4: 接著，攻擊者只需要從 r_{pred} 的左右兩側開始暴力搜尋，比對參數所生成的狀態映射圖中是否包含竊聽到的子圖 CS_{eav}，即可找出真正的 r。此處先將 $round(\frac{r_{pred}}{2^{-11}})$ 找出 r_{pred} 所屬的 state 為 8121，以 $\pm$ bias, for bias = 0,1,2,... 的方式左右搜尋以找到真實值 r。

Step 5: 攻擊者在找到 Key1 r，即可生成其相對應的狀態映射圖，並找出竊聽到的子圖迭代的最初狀態，從此位置反向做廣度優先搜尋 BFS，找出所有可能的 Key2。

Step 6: 使用確定的 Key1 和所有可能的 Key2 形成金鑰對送進解密演算法中嘗試，直到解密出原圖影像為止。

5.6.3 金鑰空間縮減分析

此段落我們要討論的是本篇論文和暴力搜尋相比在金鑰空間的縮減上比較，預訓練出來的迴歸模型雖然沒辦法精確的猜出使用者所使用的參數值，但我們利用在有限精度下，在一定誤差內對於機器來說會被視為相同值的特性，我們能在

doi:10.6342/NTU202201736

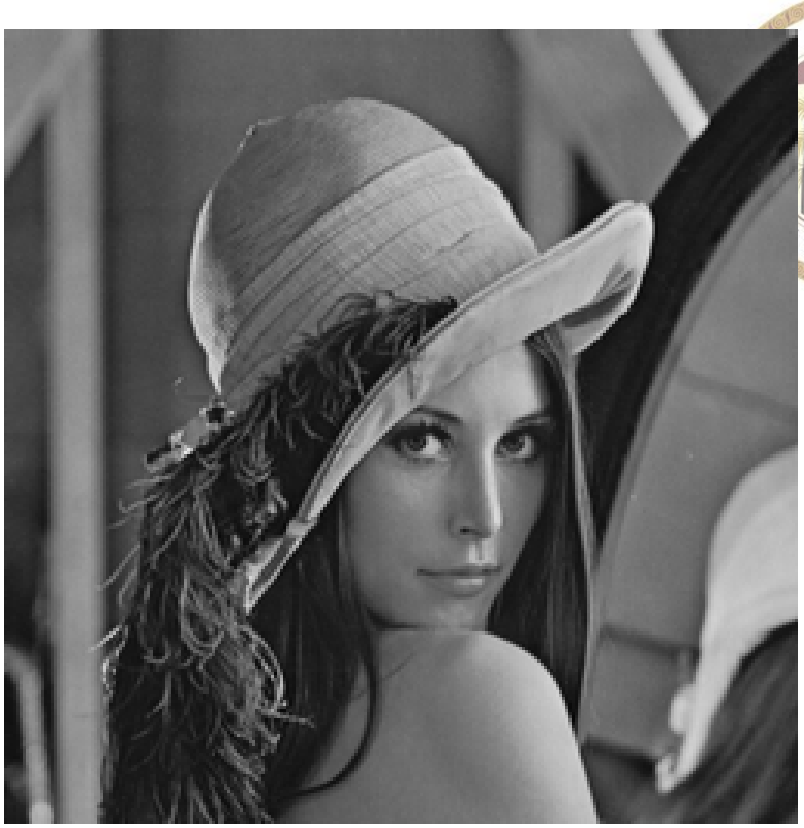


Figure 5.22: Decrypted image

預測出的值附近進行暴力搜尋。如此一來，我們所預訓練出來的模型就能被視為一種全面性的金鑰縮減工具，且此工具對於任何金鑰對都有效。

Table 5.13: 暴力破解時間複雜度分析

Method	Brute-force
Key1 space	$\frac{4-3.6}{2^{-n}}$
Compare CS_{eav}	$\mathcal{O}(len(CS_{eav}))$
Generate SMN	$\mathcal{O}(2^n)$
BFS from CS_{eav}(Key2 space)	$\mathcal{O}(\|V_{key2}\| + \|E_{key2}\|)$
Total	$\frac{4-3.6}{2^{-n}} \times \mathcal{O}(len(CS_{eav})) + \mathcal{O}(2^n) + \|V_{key2}\| \times decrpytion\_time$

doi:10.6342/NTU202201736

Table 5.14: 迴歸參數攻擊時間複雜度分析

Method	Paremeter regression attack
Model regression	Depends on different models
Key1 space	$\frac{2*Loss}{2^{-n}}$
Compare CS_{eav}	$\mathcal{O}(len(CS_{eav}))$
Generate SMN	$\mathcal{O}(2^n)$
BFS from CS_{eav}(Find Key2)	$\mathcal{O}(\|V_{key2}\| + \|E_{key2}\|)$
Total	$\frac{2*Loss}{2^{-n}} \times \mathcal{O}(len(CS_{eav})) + \mathcal{O}(2^n) + \|V_{key2}\| \times decrpytion\_time$

註：n 為機器使用精度 (單位：bits)，V_{key2} 和 E_{key2} 為從 CS_{eav} 之起點反向 BFS 之子圖點和邊個數。

表 5.14 針對我們所提出的迴歸參數攻擊做時間上的分析，可以看到 Key1 space 是根據我們最後在 Testing set 的 Loss 去決定我們最多需要幾次的搜尋，相較於暴力破解，節省了不少計算成本，且當精度 n 往上增加時，若能維持現有的 Loss 程度，能減少的時間複雜度更多。最終我們測試了 1000 筆不同的金鑰對，攻擊 Key1 平均需要 71.9 次的搜尋。

doi:10.6342/NTU202201736

第六章　結論

本篇論文提出了一種針對有限精度渾沌映射影像加密演算法的新形態參數迴歸攻擊方法。透過將對有限精度下渾沌映射離散的行為轉換為圖結構，設計一個人為生成的資料集。透過採樣演算法生成的資料集可以在少量資料的條件下達到不錯的效果和攻擊率。實驗也證實，儘管隨機生成的資料集在生成時有先天上的優勢，經過我們所設計的演算法生成的資料集在各個精度下都還是比隨機生成的資料集表現要好。雖然對於 gv 和 cp 目前只能透過實驗嘗試去找到最佳的組合，但實驗結果顯示在少量資料下機器能夠透過圖神經網路學習到不同精度下渾沌映射在各個參數區間的圖拓樸結構。

本篇論文另外一個重要的結論是有關現今大部分混沌映射影像加密演算法相關論文的質疑：在大多的文獻中，針對金鑰空間 (key space) 的討論都會直接假設機器精度去做計算，但本篇所提出的攻擊手法卻可以大幅縮減金鑰空間 (key space reduction)，使其更容易遭受暴力法的破解。並且，我們所預訓練好的模型是對所有金鑰對都有效，並不會因為不同金鑰對需要重新訓練，這對於實際應用上是很有幫助的，這也間接說明機器是真的有辦法學習到不同參數所生成之狀態映射圖的拓樸結構，並且進行分類後迴歸的任務。除此之外，攻擊者也不需要從任何地方得到訓練集資料，可以透過自行生成就能得到很好的效果。

未來我們期望實現更高的精度的模擬和訓練，我們相信若在更高精度仍然能維持目前的效果，對於金鑰空間的縮減會很有幫助。但是必須要考慮到如何設計節點

doi:10.6342/NTU202201736

特徵的問題，畢竟如果繼續使用和節點數量相同維度的單位矩陣，在高精度的情況下會造成嚴重的記憶體和計算量負擔，我們認為這是往高精度實驗必須首要先解決的問題。雖然本篇論文對於實際應用面要攻擊成功還有一段距離，但我們驗證了這樣新穎的攻擊手法在低精度上可以達到很好的效果，我們希望未來有更多志同道合的學者能夠往高精度的方向前進，達到實際應用面的破密，這樣就能真正證實混沌映射在有限精度下的安全性是需要被重新考量的。

doi:10.6342/NTU202201736

參考文獻

[1] J Fridrich. Symmetric ciphers based on two-dimensional chaotic maps. International Journal of Bifurcation and Chaos, 8:1259–1284, 1998.

[2] OLCAY TANER YILDIZ ERCAN SOLAK, CAHIT ÇOKAL and TÜRKER BIYIKOĞLU. Cryptanalysis of fridrich's chaotic image encryption. International Journal of Bifurcation and Chaos, 20(5):1405–1413, 2010.

[3] Eric Yong Xie, Chengqing Li, Simin Yu, and Jinhu Lü. On the cryptanalysis of fridrich's chaotic image encryption scheme. Signal Processing, 132:150–154, mar 2017.

[4] Chengqing Li, Dongdong Lin, Bingbing Feng, Jinhu Lu, and Feng Hao. Cryptanalysis of a chaotic image encryption algorithm based on information entropy. IEEE Access, 6:75834–75842, 2018.

[5] Guodong Ye, Chen Pan, Xiaoling Huang, Zhenyu Zhao, and Jianqing He. A Chaotic Image Encryption Algorithm Based on Information Entropy. International Journal of Bifurcation and Chaos, 28(1):1850010–750, January 2018.

[6] Sun Fu-Yan, Liu Shu-Tang, and Lü Zong-Wang. Image encryption using high-dimension chaotic system. Chinese Physics, 16:3616, 12 2007.

doi:10.6342/NTU202201736

[7] Yueping Li, Chunhua Wang, and Hua Chen. A hyper-chaos-based image encryption algorithm using pixel-level permutation and bit-level permutation. Optics and Lasers in Engineering, 90:238–246, 2017.

[8] Karim H. Moussa, Ahmed I. El Naggary, and Heba G. Mohamed. Non-linear hopped chaos parameters-based image encryption algorithm using histogram equalization. Entropy, 23(5), 2021.

[9] Philippe M. Binder and Roderick V. Jensen. Simulating chaotic behavior with finite-state machines. Phys. Rev. A, 34:4460–4463, Nov 1986.

[10] SHUJUN LI, GUANRONG CHEN, and XUANQIN MOU. On the dynamical degradation of digital piecewise linear chaotic maps. International Journal of Bifurcation and Chaos, 15(10):3119–3151, 2005.

[11] Zbigniew Galias. The dangers of rounding errors for simulations and analysis of nonlinear circuits and systems?and how to avoid them. IEEE Circuits and Systems Magazine, 13(3):35–52, 2013.

[12] Chengqing Li, Dongdong Lin, Bingbing Feng, Jinhu Lü, and Feng Hao. Cryptanalysis of a chaotic image encryption algorithm based on information entropy. IEEE Access, 6:75834–75842, 2018.

[13] Chengqing Li, Bingbing Feng, Shujun Li, Jüergen Kurths, and Guanrong Chen. Dynamic analysis of digital chaotic maps via state-mapping networks. IEEE Transactions on Circuits and Systems I: Regular Papers, 66(6):2322–2335, 2019.

[14] Zbigniew Galias. Periodic orbits of the logistic map in single and double precision implementations. IEEE Transactions on Circuits and Systems II: Express Briefs, 68(11):3471–3475, 2021.

doi:10.6342/NTU202201736

[15] Chunlei Fan and Qun Ding. Analysing the dynamics of digital chaotic maps via a new period search algorithm. Nonlinear Dynamics, 97, 07 2019.

[16] Martin Hanik, Mehmet Arif Demirtaş, Mohammed Amine Gharsallaoui, and Islem Rekik. Predicting cognitive scores with graph neural networks through sample selection learning. Brain Imaging and Behavior, 16(3):1123–1138, nov 2021.

[17] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[18] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2017.

[19] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), pages 252–257, 2015.

doi:10.6342/NTU202201736