國立清華大學 電機工程學系

實作專題成果摘要

Short-Length Linear Block Code Decoders for 5G URLLC

針對5G URLLC 的短線性區塊碼解碼器

專題領域： 通訊領域

組　　別： B351

指導教授： 翁詠祿 教授

組員姓名： 李政翰(Li, Cheng-Han)

潘玠旻(Pan, Chieh-Min)

研究期間：2023 年 2 月至 2023 年 11 月止，共 10 個月

摘要

5G NR(New Radio)提出的超可靠低延遲通信(Ultra-Reliable Low-Latency Communication, URLLC)，對低延遲和可靠性要求極高。在數位通訊編碼領域，為了滿足超可靠低延遲通信的需求，短長度編碼成為一個可能的實作選項。短長度編碼具有更快的解碼速度，得以低延遲，但其較短的碼長度會降低可靠性。因此，需要錯誤率表現更好，解碼複雜度更低的解碼器來提升可靠性。近年來，隨著電腦運算能力的提升，機器學習蓬勃發展。在數位通訊編碼領域，解碼演算法結合機器學習也成為熱門的研究方向。本篇著重討論超可靠低延遲通信解碼器的實現，並以機器學習的方式對現有的短碼長區塊碼解碼器進行討論與優化。

第一部分旨在熟悉機器學習解碼器原理與實作。以低密度奇偶檢查碼的最小和演算法與正規化最小和演算法作為解碼器演算法之基礎，並使用機器學習訓練模型獲得最佳化權重以優化原解碼器，來達成更佳的錯誤率表現。

第二部分主要重現兩個短碼的解碼演算法，分別為「由位元可靠度排序的猜測隨機可加性雜訊解碼演算法」(Ordered Reliability Bits Guessing Random Additive Noise Decoding, ORBGRAND 演算法）和「乘積碼的渦輪解碼演算法」(Turbo Decoding for Product Codes)。在重現中，我們藉由兩個解碼器的特性，提出在渦輪解碼器的架構下使用 ORBGRAND 演算法的想法，這個新的解碼器有望降低解碼複雜度。

在第二部分的最後，我們也為乘積碼的渦輪解碼器和提出的新解碼器找到可能的機器學習優化方案。在乘積碼渦輪解碼器中，我們發現其中錯誤率表現對於傳遞訊息的權重相當敏感。而翻閱參考文獻，也僅提到該權重由實驗得出。由此，結合第一部份的經驗我們希望以機器學習的方式來找到最佳的權重。

目錄

1. 研究背景

在3GPP 為5G 所開發的5G NR(New Radio)中，對於超可靠低延遲通信(Ultra-Reliable Low-Latency Communication, URLLC)的實現提出了相當嚴格的要求，使得超可靠低延遲通信成為新一代通訊標準中不可或缺的技術。特別是在數位通訊編碼領域，為滿足超可靠低延遲通信**低延遲**和**高可靠性**的要求，短長度編碼的使用受到了重視。短長度編碼雖然擁有較快的解碼速度以實現低延遲，但同時因其碼長較短而導致可靠性降低，因此需要錯誤率表現更好的解碼器來提升可靠性。

2. 研究目的

近年來，隨著電腦運算能力的提升，機器學習蓬勃發展。在數位通訊編碼領域，解碼演算法結合機器學習也成為熱門的研究方向。對於超可靠低延遲通信解碼器的研究，我們希望以機器學習的方式對現有的短區塊碼解碼器進行討論與優化。因此，我們分成兩個部分實作：

第一部分旨在熟悉機器學習解碼器原理與實作。這部分以低密度奇偶檢查碼的最小和演算法與正規化最小和演算法作為解碼器演算法之基礎，並使用機器學習訓練模型獲得最佳化權重以優化原解碼器，來達成更佳的錯誤率表現。此機器學習訓練方法參考 [1]，而建立模型並加以訓練與測試數據和結果皆是自己實作出來並實驗得到的。

第二部分主要重現兩個短長度編碼的解碼演算法，最後提出結合兩個演算法的想法。而這個解碼器兩個演算法分別為「由位元可靠度排序的猜測隨機可加性雜訊解碼演算法」[2]（Ordered Reliability Bits Guessing Random Additive Noise Decoding，本文中簡稱 ORBGRAND 演算法）和「乘積碼的渦輪解碼演算法」[3][4]（Turbo Decoding for Product Codes）。ORBGRAND 演算法適合短長度編碼的解碼，也因其高度平行化運算的超大型積體電路方案已被提出，可實現低延遲，是具有低延遲與高可靠特性的解碼演算法。ORBGRAND 演算法是其中一種猜測隨機可加性雜訊解碼演算法。所謂猜測隨機可加性雜訊解碼演算法會將收到的碼字經過硬判決(Hard Decision)之後與最有可能的錯誤碼型(Error Pattern) 二進制相加，並檢查結果是否在碼冊(Code Book)中。ORBGRAND 演算法的作者依照碼字中各位元的可靠度對碼字重排序，並提出了「滑坡」演算法(Landslide Algorithm)用以產生錯誤碼型進行解碼。ORBGRAND 演算法的解碼器是一個通用的解碼器(Universal Decoder)。在解碼表現上，對於任意長度、中等冗餘度的碼，區塊錯誤率(Block Error Rate, BLER)有接近最大似然性(Maximum Likelihood)的解碼能力[2]。在乘積碼的渦輪解碼演算法的模擬中利用延伸 BCH 碼(Extended Bose-Chaudhuri-Hocquenghem 碼，簡稱 eBCH 碼)進行乘積碼的編碼。解碼時乘積碼渦輪解碼器有兩個子解碼器，包含行解碼器和列解碼器。解碼仰賴兩個子解碼器之間的訊息交換，故經多次迭代可提升錯誤率表現。在行(列)解碼器中含有一個 Chase 解碼器[5]，用於選出候選碼字。而行(列)解碼器會對這些候選碼字的各個位元計算出對數概似性，並作為列(行)解碼器的外部訊息。對於乘積碼，在高斯通道僅需4次

迭代，即可達到大於98%的傳輸通道容量[4]。在第二部分的最後，我們提出在渦輪解碼器的架構下使用 ORBGRAND 演算法的想法。在乘積碼渦輪解碼器中，我們發現錯誤率表現對於傳遞訊息的權重相當敏感。而在參考文獻中，也僅提到該權重由實驗得出。由此，結合第一部份的經驗我們希望以機器學習的方式來找到最佳的權重。

3. 研究方法與實驗結果

3.1 第一部分：最小和演算法與正規化最小和演算法的機器學習方案

3.1.1 研究方法與過程

此部分以最小和演算法(Min-Sum Algorithm)作為解碼器演算法來進行短低密度奇偶檢查碼(LDPC code)[6]之解碼，並加入神經網路來訓練模型以得到最佳權重參數，進而達到更好錯誤率表現的解碼器。

對於(n, k)低密度奇偶檢查碼，其奇偶校驗矩陣$\mathbf{H} = [h_{i,j}]_{0\le i\le m, 0\le j\le n}$， $h_{i,j} \in$ GF(2) 其中 $m = n\text{-}k$ 為檢查節點(check nodes, CN)的個數，n 為位元節點(variable nodes, VN)的個數。碼長為 k 的訊息 **u** 會藉由生成矩陣 **G** 進行編碼 $\mathbf{c} = \mathbf{uG}$而得到碼字(codeword) **c**，此碼字在經過二進制相位偏移調變(BPSK)後加入通道可加性白高斯雜訊(Additive White Gaussian Noise, AWGN)為 $\boldsymbol{y}$，最後進入解碼器，而此篇所使用的調變與雜訊分別皆為二進制相位偏移調變與可加性白高斯雜訊。**H** 可以等效表示成坦納圖(Tanner graph)，其中位元節點與檢查節點的連線稱為邊際(edge)，而低密度奇偶檢查碼常用的解碼方法最小和演算法及和積演算法(Sum-Product Algorithm)等等皆是在邊際上進行訊息傳輸來完成解碼。

最小和演算法[7]為和積演算法[8]之簡化版，雖然錯誤率表現比和積演算法低，但其降低了計算複雜度。本篇將以使用最小和(Min-Sum, MS)演算法[7]與正規化最小和(Normalized MS, NMS)演算法[9]兩解碼器為基礎，比較神經正規化最小和 (Neural NMS, NNMS) [1]及共參數神經正規化最小和 (Shared-parameter NNMS, SNNMS) [1]這兩種使用神經網絡訓練獲得權重參數的演算法解碼器模型。而上述這四種演算法結構皆與 MS 演算法相同，差異在由位元節點與檢查節點間訊息傳輸的權重不同與後兩者演算法解碼器其權重來源為神經網絡訓練。

表1. MS, NMS, NNMS, SNNMS 四種解碼器之權重

Decoder	VN-to-CN	CN-to-VN
MS	1	1
NMS	0.8 [9]	1
NNMS	$\alpha_{ij}^{(l)}$	$\beta_{ij}^{(l)}$
SNNMS	$\alpha^{(l)}$	$\beta^{(l)}$

表1. 呈現了四種演算法解碼器所使用的權重參數，其中 NNMS 與 SNNMS 演算法權重中的 l 代表的是迭代次數。而 NNMS 所使用參數$\alpha_{ij}^{(l)}$與$\beta_{ij}^{(l)}$的i與j則代表其維度，

等同於所使用奇偶校驗矩陣 $\mathbf{H} = [h_{i,j}]_{0\le i\le m, 0\le j\le n}$，$h_{i,j} \in$ GF(2)的維度相同。而本篇使用的短碼皆為(192,96)LDPC code，碼率(code rate)為0.5，對應的$\mathbf{H}_{96\times 192}$，位元節點度(degree)為6、檢查節點度為7，此奇偶校驗矩陣來源為[10]。

NNMS 演算法模型是針對每一次迭代時，每條 CN 與 VN 間的訊息傳輸去訓練得到特定權重參數；而 SNNMS 則是在同一次迭代中，所有 VN-to-CN 的權重皆為$\alpha^{(l)}$，同理，在同一次迭代中，CN-to-VN 的權重皆為$\beta^{(l)}$。並且 NNMS 與 SNNMS 模型在訓練時，輸出層會加入 sigmoid 函數使梯度存在，讓兩模型可以藉由損失函數與梯度下降法進行權重更新。

接下來將說明 NNMS 與 SNNMS 兩演算法模型在使用神經網絡訓練時的訓練參數選擇，實驗方法參考[1]。本實驗使用 Python TensorFlow 進行模型的建立與訓練，訓練結果與訓練參數選擇的實驗過程如表2.所示。

表2. 神經網絡模型訓練資料與參數

項目	資料或參數詳情	參考來源或實驗結果圖
Training data	0dB~5dB, 0.25dB/Step, Totally 21 kinds of data / Epoch	[8]
Loss function	Binary Cross-Entropy	[8]
Epoch & learning rate (lr)	Epoch = 10 & lr = 0.001	圖 1
Iteration	$l_{max} = 10$	圖 2
Activate function	LeakyReLU (r = 100)	圖 3

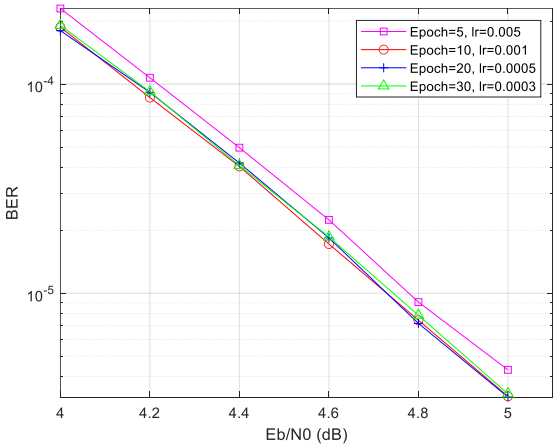

圖 1. 神經正規化最小和解碼器使用四種訓練時期與學習率組合

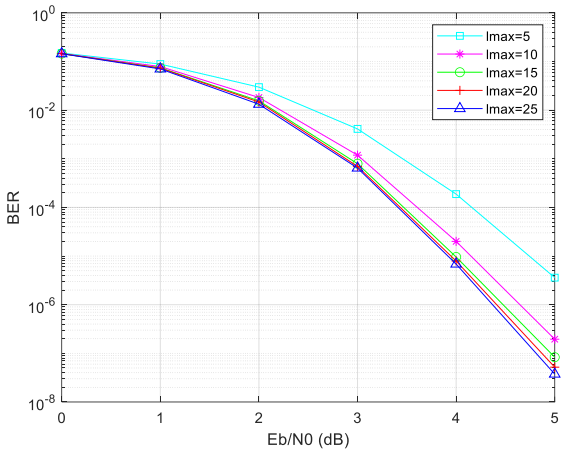

圖 2. 神經正規化最小和解碼器在不同l_{max}情形下之錯誤率表現

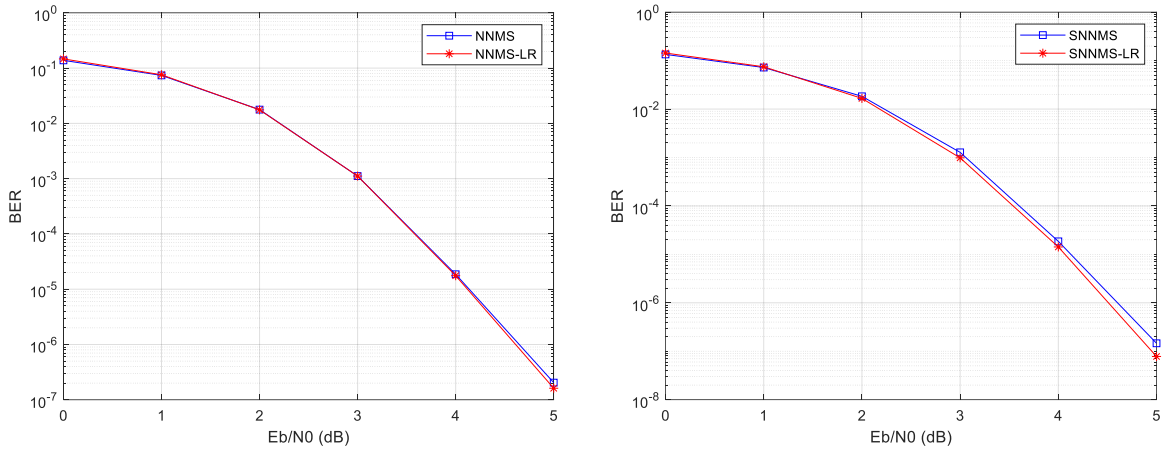

圖 3. NNMS 與 SNNMS Decoder 有無使用 LeakyReLU 函數之比較

3.1.2 研究結果

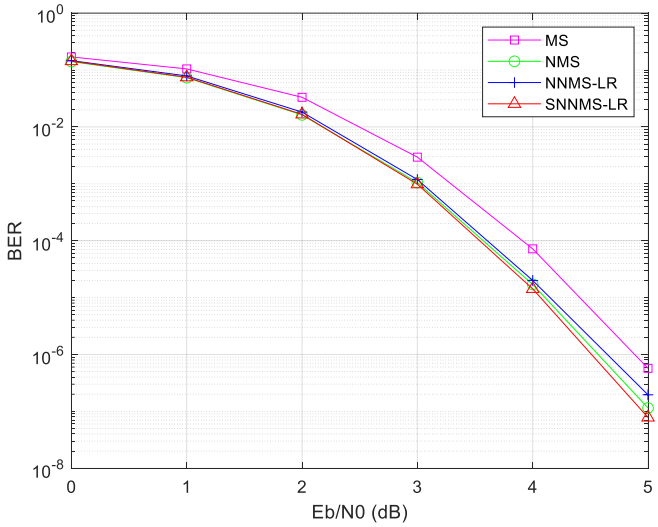

圖4. (192,96) 低密度奇偶檢查碼四種解碼器模型錯誤率之比較

由圖4 可見，具有龐大的權重數量且使用神經網絡訓練的 NNMS 解碼器，其錯誤率表現反而比單一權重 NMS 的錯誤率還要差。而在每一次迭代共用權重的 SNNMS 解碼器，經由神經網絡訓練所得到的權重，不僅參數數量較 NNMS 解碼器少很多，其錯誤率表現亦最佳，成功降低了解碼器的錯誤率。

3.2 第二部分：短長度編碼解碼演算法的重現

3.2.1 由位元可靠度排序的猜測隨機可加性雜訊解碼演算法(ORBGRAND)

A. ORBGRAND 原理

對一個n位元二進制碼字 $\boldsymbol{c^n} = (\mathrm{c}_1, \mathrm{c}_2, \dots, \mathrm{c}_\mathrm{n})$，$(\mathrm{c_i} \in \{0,1\}, \mathrm{i} = 1\sim\mathrm{n})$以二進制相位偏移調變(Binary Phase-Shift Keying modulation)得到傳送的碼字$mod(\boldsymbol{c^n}) \in \{\pm 1\}^n$。$mod(\boldsymbol{c^n})$經可加性白高斯雜訊通道(Additive White Gaussian Noise Channel)受雜訊$\boldsymbol{N^n}$干擾。收到的碼字即 $\boldsymbol{Y^n} = mod(\boldsymbol{c^n}) + \boldsymbol{N^n}$。後對$\boldsymbol{Y^n}$進行硬判決，得到$\boldsymbol{y^n} = (\mathrm{y}_1, \mathrm{y}_2, \dots, \mathrm{y}_\mathrm{n})$，$(\mathrm{y^i} \in \{0,1\}, \mathrm{i} = 1\sim\mathrm{n})$。而其中錯誤碼型 $\boldsymbol{Z^n} = \boldsymbol{c^n} \oplus \boldsymbol{y^n}$。在 ORBGRAND 演算法中，透過產生錯誤碼型 $\boldsymbol{Z^n}$，嘗試解出原始碼字$\boldsymbol{c^n}$。接下來我們推出錯誤碼型 $\boldsymbol{z^n}$的事後機率。對於收到的碼字 $\boldsymbol{Y^n}$中的元素Y_i，他的對數似然率(Log-likelihood Ratio, LLR):

$$LLR(Y_i) = log\frac{f_{Y|C}(Y_i|c=1)}{f_{Y|C}(Y_i|c=0)} \tag{1}$$

由$LLR(Y_i)$可推出序列 $\boldsymbol{B^n} = (\mathrm{B}_1, \mathrm{B}_2, \dots, \mathrm{B}_\mathrm{n})$，其中$\mathrm{B_i}$為$\mathrm{y_i}$錯誤的事後機率，寫作:

$$\mathrm{B}_i = \frac{e^{-|LLR(Y_i)|}}{1+e^{-|LLR(Y_i)|}} \tag{2}$$

從(2)可推出錯誤碼型 $\boldsymbol{z^n}$的事後機率 $P(\boldsymbol{Z^n} = \boldsymbol{z^n})$:

$$P(\boldsymbol{Z^n} = \boldsymbol{z^n}) = \prod_{i:z_i=0}(1-B_i)\prod_{i:z_i=1}B_i = \prod_{i=1}^{n}(1-B_i)\prod_{i:z_i=1}\frac{B_i}{(1-B_i)}$$

$$\propto \prod_{i:z_i=1}\frac{B_i}{(1-B_i)} = \exp\left(-\sum_{i=1}^{n}|LLR(Y_i)\,|z_i\right) \tag{3}$$

其中，定義可靠度 $Rel(\boldsymbol{z^n}) = \sum_{i=1}^{n}|LLR(Y_i)|\,z_i$。

由(3)知，$Rel(\boldsymbol{z^n})$愈小，錯誤碼型 $\boldsymbol{z^n}$的事後機率愈大。因此，擁有最小可靠度$Rel(\boldsymbol{z^n})$的$\boldsymbol{z^n}$即為發生率最高的錯誤碼型。由此，我們可以找到一群可能的錯誤碼型$\boldsymbol{z^n}$並將這些錯誤碼型根據他們可靠度$Rel(\boldsymbol{z^n})$的大小從小排到大。解碼時依序將這些錯誤碼型與收到的碼字 $\boldsymbol{y^n}$二進制相加，最終$\boldsymbol{c^n} = \boldsymbol{y^n} \oplus \boldsymbol{z^n}$ 解出。剩下的問題是如何找出可靠度$Rel(\boldsymbol{z^n})$。

B. 基本版 ORBGRAND

在基本版 ORBGRAND 中，我們將$|LLR(Y_i)|$由小到大排序，並藉由簡單的線性模型對排序的$|LLR(Y_i)|$進行線性近似，以此找到可靠度$Rel(\boldsymbol{z^n})$。在本專題中使用可加性白高斯雜訊通道進行模擬，在可加性白高斯雜訊通道中$|LLR(Y_i)| \propto |Y_i|$，因此我們可以直接利用$|Y_i|$進行排序即可。

考慮線性模型$\boldsymbol{\lambda^n} = (\lambda_1, \lambda_2, \dots, \lambda_n)$，其中$\lambda_i = \beta i, i = 1, 2, \dots, n$，而$\beta > 0$是斜率。用$\boldsymbol{\lambda^n}$對可靠度$Rel(\boldsymbol{z^n})$進行線性近似：

$$Rel(\boldsymbol{z^n}) \approx \sum_{i:z_i=1} \lambda_i = \beta \sum_{i=1}^{n} i z_i = \beta * w_L(\boldsymbol{z^n}) \qquad (4)$$

在(4)中，我們定義計算權重(Logistic Weight) $w_L(\boldsymbol{z^n}) = \sum_{i=1}^{n} i z_i$，代表錯誤碼型 $\boldsymbol{z^n}$上錯誤位置的索引值總和。考慮$\boldsymbol{w_L(z^n)}$及對應的$\boldsymbol{z^n}$，如表3：

表 3. $w_L(\boldsymbol{z^n})$與$\boldsymbol{z^n}$對應表

$w_L(\boldsymbol{z^n})$	$\boldsymbol{z^n}$
1	$(0,\ 0,\ 0,\ \dots,\ 0)$
2	$(1,\ 0,\ 0,\ \dots,\ 0)$
3	$(0,\ 1,\ 0,\ \dots,\ 0)$
4	$(0,\ 0,\ 1, 0,\ \dots,\ 0)\ or\ (1,\ 1,\ 0,\ \dots,\ 0)$
:	:
$n(n+1)/2$	$(1,\ 1,\ 1,\ \dots,\ 1)$

由表3可以發現，$\boldsymbol{z^n}$序列依照$w_L(\boldsymbol{z^n})$遞增排序，由此我們無需估計β值。此外，錯誤碼型 $\boldsymbol{z^n}$的計算權重 $W \in \{0, 1, \dots, n(n+1)/2\}$。

考慮所有$\boldsymbol{z^n}$的集合S_W：

$$S_W = \{\boldsymbol{z^n} \in \{0,1\}^n :\ w_L(\boldsymbol{z^n}) =\ W\} \qquad (5)$$

考慮$\boldsymbol{z^n}$的漢明權重 $w_H(\boldsymbol{z^n}) = w$。一個漢明權重 w 的錯誤碼型 $\boldsymbol{z^n}$，當$\boldsymbol{z^n}$的前w 位元都是1時有最小的計算權重。由此可知漢明權重與計算權重的關係: $w(w+1)/2 \le W$。

我們改寫(5)：

$$S_W = \bigcup_{w=1}^{\lfloor\sqrt{1+8W}-1/2\rfloor} \{\boldsymbol{z^n} \in \{0,1\}^n, w_H(\boldsymbol{z^n}) = w,\ w_L(\boldsymbol{z^n}) =\ W\} \qquad (6)$$

其中聯集的上界$\sqrt{1+8W} - 1/2$由不等式$w(w+1)/2 \le W$得。

考慮(6)中的一個子集合$\{\boldsymbol{z^n} \in \{0,1\}^n, w_H(\boldsymbol{z^n}) = w,\ w_L(\boldsymbol{z^n}) =\ W\}$，將$\boldsymbol{z^n}$改以$\boldsymbol{v^w}$表示，記錄$\boldsymbol{z^n}$中為1的$w$個位置：

$$\left\{\boldsymbol{v^w} \in \mathbb{N}^w : 1 \le v_1 < \cdots < v_w \le n, \sum_{i=1}^{w} v_i = W\right\} \qquad (7)$$

為後續演算法方便，我們改寫(7)，使得$u_i = v_i - i, (i = 1, 2, \ldots, w)$。

$$\left\{ \boldsymbol{u^w} \in \mathbb{Z}_+^w : 0 \le u_1 \le \cdots \le u_w \le n', \sum_{i=1}^{w} u_i = W' \right\} \tag{8}$$

在(8)中，由於原先的v_i都減去了自身位置的值，n和W分別被改寫為$n' = n - w$和$W' = W - w(w+1)/2$。舉例來說，$\boldsymbol{z^5} = (1,0,0,0,1)$時，有$\boldsymbol{v^2} = (1,5)$，$\boldsymbol{u^2} = (0,3)$。

C. 錯誤碼型的產生器

錯誤碼型產生器是將總體權重W'整數值分成w份整數，而每一份的數值不超過n'。錯誤碼型產生器產生的整數分割$\boldsymbol{u^w}$可以透過關係式$v_i = u_i + i$得到$\boldsymbol{v^w}$，$\boldsymbol{v^w}$紀錄錯誤碼型$\boldsymbol{z^n}$為1的位置，因此很容易就能轉換成$\boldsymbol{z^n}$。

關於錯誤碼型產生器，我們先定義函式：

1. 下降函式(Drop Function) — $d(i)$

$$d(i) = \begin{cases} 0 & if\ i = w \\ u_{i+1} - u_i & if\ i \in \{1, \ldots, w-1\} \end{cases} \tag{9}$$

2. 累積下降函式(Accumulated Drop Function) — $D(i)$

$$D(i) = \sum_{j=i}^{w} d(j) \tag{10}$$

3. 造山函式(Build Mountain Function) — Build(u_k)

對Build(u_k)，若u_k之值為g，設定$u_{k+1} = u_{k+2} = \cdots = u_w = g$。總體權重$W'$分給$w-k+1$個位置，每個位置分得$g$。對剩下$W_{Remain} = W' - g(w-k+1)$，從$u_w$開始依序向$u_{w-1}, u_{w-2}, \ldots$分配，直到某個$u_j$，每個位置分配到上限$n'$，直到$W_{Remain}$分完為止。

「滑坡」演算法(Landslide Algorithm)是錯誤碼型產生器的演算法，輸入W', w, n'，輸出則為改寫過的錯誤碼型$\boldsymbol{u^{w,j}}$。虛擬碼(Pseudo Code)如下：

「滑坡」演算法

輸入: W', w, n'
輸出: $\{\boldsymbol{u^{w,j}}, j = 1,2, \ldots\}$ //漢明權重為w的所有錯誤碼型

```
Build(u_1)                          //初始化
j ← 1
u^{w,j} ← u^w
針對 i = 1~w 更新 D(i)
迴圈開始：當 D(1) ≥ 2 執行
    在所有 D(k) ≥ 2 的 k 中找最大的 k
    u_k ← u_k + 1
    Build(u_k)
    針對 i = 1~w 更新 D(i)
    j ← j + 1
    u^{w,j} ← u^w
迴圈結束
回傳 {u^{w,j}, j = 1,2, ...}
```

在「滑坡」演算法中，雖然只能產生漢明權重為w的錯誤碼型，但若平行的運算不同的w，可以快速的產生錯誤碼型。

D. 模擬方式與結果

將n位元二進制碼字u以(n,k)BCH 碼進行編碼。由二進制相位偏移調變，經可加性白高斯雜訊通道得到碼字r。將r以實作的基本版 ORBGRAND 解碼，並取錯誤碼型的數量上限為2^{n-k}。圖5繪出模擬的區塊錯誤率(BLER)並與論文[11]比對，驗證了實作的正確性。對 BCH(127, 106)的區塊錯誤率，與論文基本版 ORBGRAND 的解碼性能相符。同時，也畫出 BCH(255, 231)的區塊錯誤率，以比較不同碼長的表現。

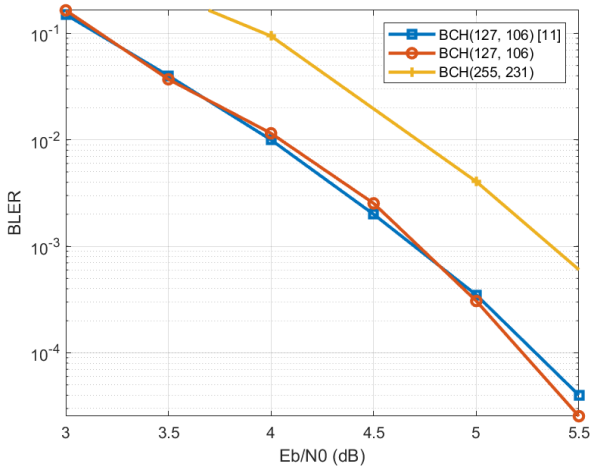

圖 5. BCH(127, 106)和 BCH(255, 231) ORBGRAND 解碼的區塊錯誤率

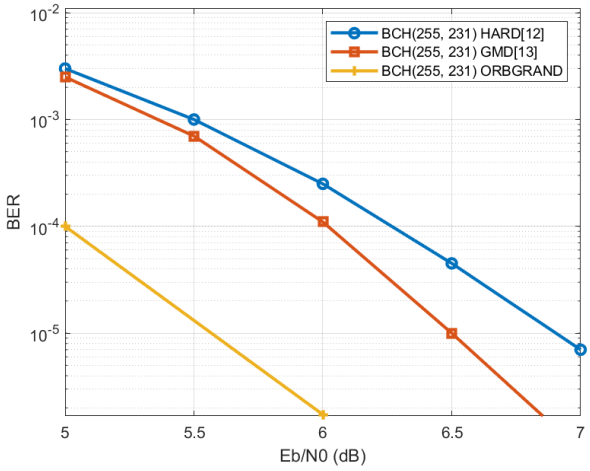

圖 6. BCH(255, 231) ORBGRAND 解碼與 HARD 解碼、GMD 解碼的位元錯誤率

在圖6中，比較不同解碼器與基本版 ORBGRAND 的位元錯誤率(BER)解碼表現，與硬解碼器(HARD)[12]，和廣義最小距離解碼器(GMD)[13]比較，以10^{-5}分貝為基準，對廣義最小距離解碼器約有1分貝的增益。

3.2.2 乘積碼的解碼：區塊渦輪碼

A. 區塊渦輪碼

區塊渦輪碼(Block Turbo Code, BTC) $C(n_1n_2, k_1k_2, d_1d_2)$由兩個線性區塊碼 $C_1(n_1, k_1, d_1)$、$C_2(n_2, k_2, d_2)$做克羅內克積(Kronecker Product)得來。實作上由$C_1(n_1, k_1, d_1)$對$k_1 \times k_1$的訊息矩陣做行編碼和再由$C_2(n_2, k_2, d_2)$對C_1編碼完的$k_1 \times n_1$矩陣做列編碼，得到$n_2 \times n_1$的碼字矩陣。本專題選擇用 eBCH 碼來編區塊渦輪碼主要有兩個目的：其一是 eBCH 碼只比 BCH 碼多一個冗餘位元，但在編成區塊渦輪碼後可以大幅提升最小距離(Minimum Distance)增加解碼性能；其二是 eBCH 碼是5G 超可靠低延遲通信的候選碼之一。

B. 解碼

在解碼器部分，主要由兩個軟輸入軟輸出的 Chase 解碼器依序對行和列進行解碼，解碼過程中行(列)解碼器計算出外部訊息(Extrinsic Information)並傳給列(行)解碼器進行解碼。

對一個n位元的二進制碼字$\boldsymbol{m^n} = (\mathrm{m_1}, \mathrm{m_2}, \ldots, \mathrm{m_n})$，$(\mathrm{m_i} \in \{0,1\}, \mathrm{i} = 1\sim\mathrm{n})$以二進制相位偏移調變得到傳送碼字$\boldsymbol{c^n} \in \{\pm 1\}^n$。$\boldsymbol{c^n}$經可加性白高斯雜訊通道受雜訊$\boldsymbol{N^n}$干擾。收到的碼字即$\boldsymbol{Y^n} = \boldsymbol{c^n} + \boldsymbol{N^n}$。對$\boldsymbol{Y^n}$進行硬判決，得到$\boldsymbol{y^n} = (\mathrm{y_1}, \mathrm{y_2}, \ldots, \mathrm{y_n})$，$(\mathrm{y^i} \in \{0,1\}, \mathrm{i} = 1\sim\mathrm{n})$。對收到的$Y_i$定義對數似然率(Log-likelihood Ratio, LLR):

$$LLR(Y_i) = log\frac{f_{Y|C}(Y_i|c_i = +1)}{f_{Y|C}(Y_i|c_i = -1)} \tag{11}$$

$|LLR(Y_i)|$即Y_i的可靠度。在可加性白高斯雜訊通道中$|LLR(Y_i)| \propto |Y_i|$，比較$|Y_i|$可知可靠度大小。

C. Chase 解碼器

對於一個(n,k,d)碼，我們取$p = \lfloor d/2 \rfloor$個最低可靠度位置，來產生錯誤碼型。其中錯誤碼型為$\boldsymbol{Z^n}$，共有2^p個。產生方式由其中一個最低可靠度位置為1，其餘為0；其中兩個最低可靠度位置為1，其餘為0；如此產生下去。將硬判決得到的$\boldsymbol{y^n}$與$\boldsymbol{Z^n}$以二進制相加，得暫時的候選碼字$\boldsymbol{T^n} \in \{0,1\}^n$，此時$T^n$未必是合法的碼字。對$\boldsymbol{T^n}$進行代數解碼(Algebraic Decode)，得到候選碼字$\widehat{\boldsymbol{T}}^{\boldsymbol{n}} \in \{\pm 1\}^n$。

定義候選碼字集合Ω：

$$\Omega = \{\widehat{\boldsymbol{T}}^{\boldsymbol{n}}\text{: 硬判決}(\widehat{\boldsymbol{T}}^{\boldsymbol{n}}) \in (n,k,d)\text{碼冊}\} \tag{12}$$

Chase 解碼器即在Ω中找出與$\boldsymbol{Y^n}$的歐幾里得距離最小的$\widehat{\boldsymbol{T}}^{\boldsymbol{n}}$，作為輸出$\widehat{\boldsymbol{c}}^{\boldsymbol{n}}$：

$$\widehat{\boldsymbol{c}}^{\boldsymbol{n}} = arg\min_{\hat{T}^n \in \Omega} \left\| \boldsymbol{Y^n} - \widehat{\boldsymbol{T}}^{\boldsymbol{n}} \right\|^2 \tag{13}$$

D. 外部訊息(Extrinsic Information)

在一個$\widehat{\boldsymbol{T}}^{\boldsymbol{n}}$的第$i$個位元$(i = 1\sim n)$，寫作$\hat{T}_i$。有對數似然率：

$$LLR(\hat{T}_i) = log\frac{P_r(\hat{T}_i = +1|y)}{P_r(\hat{T}_i = -1|y)} \tag{14}$$

若假設在碼字的分布是均勻分布，由貝式定理可以寫成：

$$LLR(\hat{T}_i) = log\frac{\sum_{\hat{T}\in\Omega_i^+} p(Y|\hat{T})}{\sum_{\hat{T}\in\Omega_i^-} p(Y|\hat{T})} \tag{15}$$

其中$p\left(\boldsymbol{Y}\middle|\widehat{\boldsymbol{T}}\right) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{\left\|\boldsymbol{Y^n} - \widehat{\boldsymbol{T}}^{\boldsymbol{n}}\right\|^2}{2\sigma^2}\right)$；$\Omega_i^+(\Omega_i^-)$為在Ω中$\hat{T}_i = +1(\hat{T}_i = -1)$的所以有$\hat{T}$組成的子集合。對(15)進一步推導有：

$$LLR(\hat{T}_i) \approx \frac{1}{2\sigma^2}\left(\left\|\boldsymbol{Y^n} - \hat{T}_i^+\right\|^2 - \left\|\boldsymbol{Y^n} - \hat{T}_i^-\right\|^2\right) = \frac{\boldsymbol{Y^n}}{\sigma^2}\left(\hat{T}_i^+ - \hat{T}_i^-\right) \tag{16}$$

以$\hat{T}_i^+\left(\hat{T}_i^-\right)$表示在$\Omega_i^+(\Omega_i^-)$中的最靠近$\boldsymbol{Y^n}$的元素。$\hat{T}_i^+$和$\hat{T}_i^-$之中必定有一個是$\widehat{\boldsymbol{c}}^{\boldsymbol{n}}$，另一個

記作$\boldsymbol{c'^n}$，將(16)改寫成：

$$\tilde{L}(\hat{T}_i) \triangleq \frac{\boldsymbol{Y^n}}{\sigma^2}(\hat{\boldsymbol{c}}^{\boldsymbol{n}} - \boldsymbol{c'^n})\hat{T}_i = \frac{2}{\sigma^2}\left\{Y_i + \frac{\hat{T}_i}{2}\left(\sum_{s\neq k} Y_s\,\hat{\boldsymbol{c}}^{\boldsymbol{n}} - \sum_{s\neq k} Y_s\,\boldsymbol{c'^n}\right)\right\} \tag{17}$$

忽略(17)式中的常數我們有：

$$\tilde{L}(\hat{T}_i) = Y_i + \frac{\hat{T}_i}{2}\left(\sum_{s\neq k} Y_s\,\hat{\boldsymbol{c}}^{\boldsymbol{n}} - \sum_{s\neq k} Y_s\,\boldsymbol{c'^n}\right) = Y_i + L_i^e \tag{18}$$

(18)式中L_i^e為外部訊息。

若在$\Omega_i^+(\Omega_i^-)$中找不到$\boldsymbol{c'^n}$，使

$$\tilde{L}(\hat{T}_i) = \beta_{iter}\hat{c}_i \tag{19}$$

(19)式中$\beta_{iter} \in [0,1]$與迭代次數正相關。

E. 渦輪解碼器

渦輪解碼器的架構如圖7 [3]，包含兩個以 Chase 解碼器為基礎的解碼器，行解碼器和列解碼器，還有區塊交織器(Block Interleaver) π和π^{-1}。

兩個解碼器根據收到的Y_i和$\hat{T}_i$分別計算出似然率 $L_{row}(\hat{T}_i)$和$L_{col}(\hat{T}_i)$：

$$L_{row}(\hat{T}_i) = Y_i + \alpha_{iter}L_{rc,i}^e + \alpha_{iter}L_{cr,i}^e \tag{20}$$

$$L_{col}(\hat{T}_i) = Y_i + \alpha_{iter}L_{cr,i}^e + \alpha_{iter}L_{rc,i}^e \tag{21}$$

解碼經多次迭代，對最終的$L_{row}(\hat{T}_i)$或$L_{col}(\hat{T}_i)$進行硬判決完成，外部訊息權重 $\alpha_{iter} \in [0,1]$與迭代次數正相關，並由實驗得出。

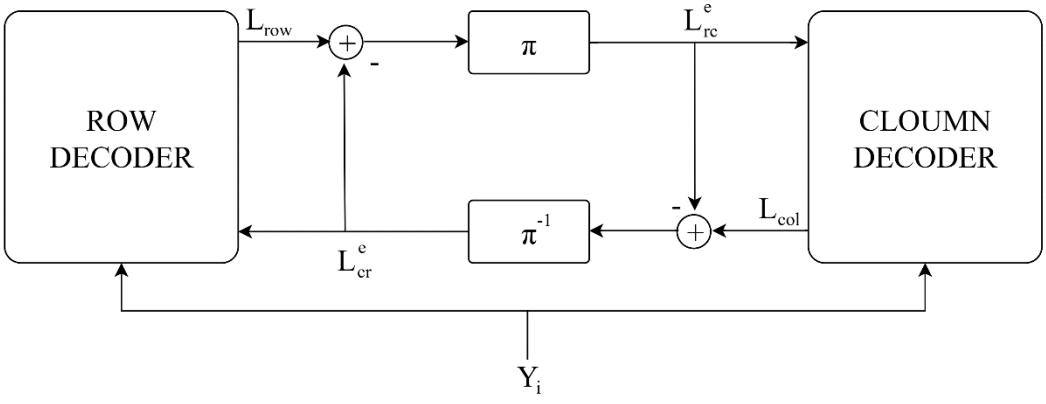

圖7. 渦輪解碼器的架構圖[3]

F. 模擬方式與結果

將n位元二進制碼字u以兩個相同(n,k,d)eBCH 碼進行區塊渦輪碼編碼。由二進制相位偏移調變，經可加性白高斯雜訊通道得到碼字r。將r以實作的乘積渦輪解碼器解碼，取 Chase 解碼器的不可靠位元數為$\lfloor d/2 \rfloor$，d為乘積碼的最小距離。在四次迭代下有圖8結果，其中藍色線結果取自論文[4]。由結果發現乘積渦輪解碼器的解碼表現對和迭代次數相關的參數$\alpha_{iter}, \beta_{iter}$非常敏感，經過實驗調整參數後，可與論文表現相符。

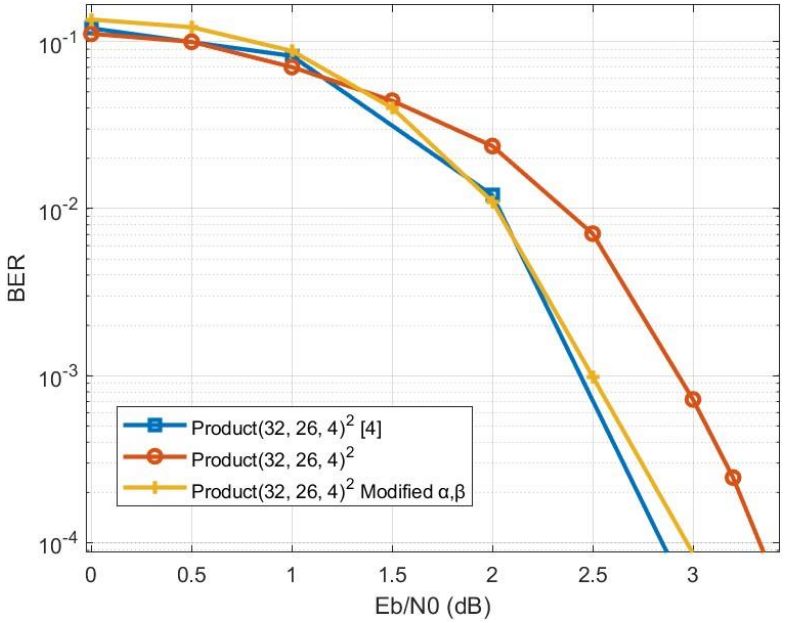

圖 8. 乘積渦輪解碼器對區塊渦輪碼$(32,26,4)^2$的解碼錯誤率表現

3.2.3 未來研究

在重現了上述兩篇論文之後，希望將實作的乘積渦輪解碼器中的兩個行列解碼器利用 ORBGRAND 解碼器進行替換。

在重現中我們了解到，ORBGRAND 解碼器和乘積渦輪解碼器的共通點在於，兩個解碼器都要由某種方式產生出錯誤碼型。其中 ORBGRAND 使用的是滑坡演算法，而乘積渦輪解碼器採用的是 Chase 解碼器。分析兩個解碼器：ORBGRAND 的優點在於滑坡演算法無須在解碼過程中生成錯誤碼型，可預先製成表，以查表方式逐一檢查錯誤碼型。而缺點在於若雜訊很大時要成功解碼，往往要成千上萬的比對次數；關於乘積渦輪解碼器的優點在於 Chase 解碼器過程中無須像 ORBGRAND 一樣產生非常多錯誤碼型，而是生成一小部份候選碼字再透過迭代解碼。其缺點在於要在解碼的當下產生錯誤碼型，這造成複雜度來源。

若將實作的乘積渦輪解碼器中的兩個行列解碼器利用 ORBGRAND 解碼器進行替換，則結合了兩個解碼器的優點。如此可以利用滑坡演算法產生二到數個碼字，而將這些碼字以乘積渦輪解碼器的迭代方式解碼。將降低解碼複雜度。

4. 結論

本專題嘗試在超可靠低延遲通信(Ultra-Reliable Low-Latency Communication, URLLC)的解碼器中尋找可能的機器學習改善方案。

在第一部分，我們旨在熟悉機器學習解碼器的原理與實作。針對最小和演算法、正規化最小和演算法開發以機器學習優化的短碼長低密度奇偶檢查碼解碼器。結果顯示，共參數神經正規化最小和演算法，在錯誤率表現上較神經正規化最小和演算法好，可能因神經正規化最小和演算法含有大量權重，而導致過擬合。相對共參數神經正規化最小和演算法因其共參數的特性防止過擬合的發生，同時較少的參數也讓訓練上的時程較短。

第二部分，我們重現了兩個短碼長解碼演算法：ORBGRAND 和乘積渦輪解碼。透過在實作中的發現，利用 ORBGRAND 的預先處理錯誤碼型的優點，並用乘積渦輪解碼器迭代解碼的方式彌補 ORBGRAND 雜訊很大時要相當多比對次數的缺點。提出將乘積渦輪解碼器中的兩個行列解碼器利用 ORBGRAND 解碼器進行替換的構想。其中，乘積渦輪解碼器的錯誤率表現對於傳遞訊息的權重相當敏感。查閱相關文獻，並未提及該權重的得出方式。由此，結合第一部份的經驗我們認為以機器學習的方式來找到最佳的權重，是最佳化乘積渦輪解碼器的方法。

由於時程關係，在本專題並未實作我們所提出的新解碼器，以及其機器學習優化方案。但是我們的分析和文獻回顧仍然有助於理解相關技術及其在解碼過程中的潛在影響。這些洞察為未來的研究提供了一個有價值的起點。

5. 參考資料

[1] Qing Wang, Qing Liu, Shunfu Wang, Leian Chen, Haoyu Fang, Luyong Chen, Yuzhang Guo, and Zhiqiang Wu, “Normalized Min-Sum Neural Network for LDPC Decoding”, *IEEE Trans. On Cognitive Commun.* vol. 9, no.1, pp.70-81, Feb 2023.

[2] K. R. Duffy, W. An and M. Médard, "Ordered Reliability Bits Guessing Random Additive Noise Decoding," *IEEE Trans. on Signal Processing*, vol. 70, pp. 4528-4542, 2022, doi: 10.1109/TSP.2022.3203251.

[3] Ryan, W., & Lin, S. (2009). Turbo Codes. In *Channel Codes: Classical and Modern* , pp.298-338. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511803253.008

[4] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. on Commun.*, vol. 46, no. 8, pp. 1003-1010, Aug. 1998, doi: 10.1109/26.705396.

[5] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 170-182, January 1972, doi: 10.1109/TIT.1972.1054746.

[6] R.Gallager, “Low-density parity-check codes,” *IRE Trans. Inform. Theory,* vol. 8, no. 1, pp. 21-28, 1962.

[7] M. P. C. Fossorier, M. Mihaljevic, and H.lmai, “Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation,” *IEEE Trans. On Commun.* vol. 47, no.5, pp.673-680, May 1999.

[8] D. J. C. Mackay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans. Inform. Theory*, vol. 45, pp.399-431, Mar 1999.

[9] Jinghu Chen and Marc P. C. Fossorier, “Density Evolution for Two Improved BP-Based Decoding Algorithms of LDPC Codes,” *IEEE Trans. On Commun.* vol. 6, no.5, pp.208-210, May 2002.

[10] M. Helmling et al. “Database of channel codes and ML simulation results.” University Koblenz Landau. 2016. [Online]. Available: www.uni-kl.de/channel-codes

[11] K. R. Duffy, "Ordered Reliability Bits Guessing Random Additive Noise Decoding," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, 2021, pp. 8268-8272, doi: 10.1109/ICASSP39728.2021.9414615.

[12] Lin, Yi-Min, Lee, Chen-Yi, & Chang, Hsie-Chia (2010). Area-Efficient Soft BCH and RS Decoders. http://hdl.handle.net/11536/40709.

[13] G. Forney, "Generalized minimum distance decoding," *IEEE Trans. Inform. Theory*, vol. 12, no. 2, pp. 125-131, April 1966, doi: 10.1109/TIT.1966.1053873.

6.心得

錯誤更正碼是一個在大學部較少接觸的領域，在大學部基礎課程只接觸過最簡單的漢明碼和重複碼。在十個月的研究過程中，我們從最基礎的漢明碼複習起，並開始學習一些較為現代的編碼，包括 BCH 碼(Bose – Chaudhuri – Hocquenghem code)，和5G標準中的低密度奇偶檢查碼（Low-density parity-check code，LDPC code）、極化碼(Polar Codes)。在學習過程中也實作不同碼的編碼方式及對應的經典解碼器，包括低密度奇偶檢查碼的和積演算法(Sum-Product Algorithm)，極化碼的列表連續消除演算法(Successive Cancellation List)等。此外，我們也練習了機器學習解碼器的實作。

在對不同種類的編碼有初步的認識後，我們分工做不同方向的研究：玠旻主要研究短碼長的低密度奇偶檢查碼的最小和演算法及正規化最小和演算法；政翰主要研究 ORBGRAND 演算法及區塊渦輪碼的渦輪解碼演算法。感謝彼此在遇到困難時的相互協助，雖然我們的新解碼器和機器學習優化方案沒有來得及實現，但在十個月內，對於一個幾乎陌生的領域，從基礎知識到對錯誤更正碼領域有較為深入的了解，並能對於演算法的缺點提出改善方案，我們相當有成就感。在實作中，發現問題，並解決問題的研究的過程也讓我們獲益良多。

最後，我們特別感謝指導教授翁詠祿老師及博士班學長古軒。感謝教授在每週的討論中提供指導與建議，指引可能的研究方向。也感謝學長對於錯誤更正碼基礎知識的教學，並在我們遇到困難時抽出時間和我們討論問題。