If Turing and Rejewski Took CS109

Syed Ashal Ali

March 2024

1 Introduction

1.1 Background

In the 1920s, the Germans started encrypting their movements using a technology that they had developed called the **Enigma**. As World War II started, the German Enigma-encoded messages were considered undecipherable: this rendered the intelligence of the Allied Powers useless. Through the key work of Polish mathematician Marian Rejewski and British mathematician Alan Turing, the Enigma code was cracked through the invention of the **Bombe**. It is believed that cracking Enigma was the most pivotal part of World War II and it was able to reduce the duration of the war by several years.

1.2 Structure of the Engima

Before delving into the approach, it is important to understand the structure of the Enigma (Figure 1). The Enigma had the following three main components:

- five rotors from which three were chosen and then arranged,
- each rotor had to be set to a starting position which could be one of the 26 letters of the alphabet,
- a plugboard that connected two letters with a wire that would swap the letters (eg. if A and T were connected, typing A would output T). There were 10 such wires.

A sample flow of encrypting a message with the enigma would start with the operator selecting and arranging the three rotors from the five rotors, and then setting up the 10 wire pairs on the plugboard. As the operator would type out their message, the letters would be swapped by the plugboard and the swapped letters would be further encrypted as they passed through each rotor. The overall output would be the enigma-encrypted code.

1.3 My Passion and the Approach

As a huge fan of history and logic, I enjoyed learning about the World Wars and particularly, the Enigma Machine. I was, and still am, stunned by the ingenuity of individuals to be able to create something like the Enigma, and more so, with

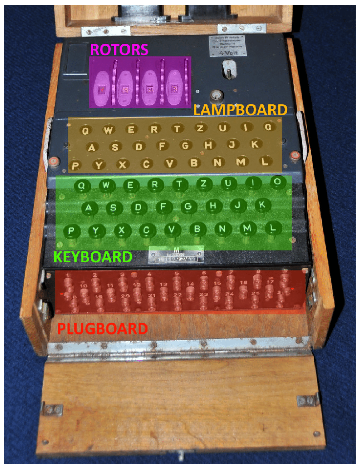


Figure 1: Diagram of the Enigma Machine

the creativity to be able to decode it. After watching ”The Imitation Game” (I highly recommend this movie!), I knew that I wanted to explore the Enigma further through this project.

While Rejewski and Turing used advanced cryptography, math, and logic to crack the Enigma, I will be using concepts that I have learned in CS 109 to devise a strategy that outputs the probability of reaching the correct settings of the Enigma for a given day. A successful implementation of this strategy would be if the probability of reaching the correct settings through my approach is greater than the probability arrived via a brute force solution.

2 Assumptions

For this paper, the Enigma can be considered to be ”cracked” if we are able to initialize it to the correct settings as this would mean that we have replicated the encryption process: decryption would involve following the steps in reverse order. Since a large part of Turing's original approach involved intuitive predictions based on real-time context, I will be illustrating a scenario that will inform our statistical strategy. For the purpose of this paper, let's assume that there is an impending Blitzkrieg (an offensive strategy responsible for Germany's successes in World War II) that the CS109 intelligence agency has predicted. Additionally, we will be assuming that the encryption and decryption is from English.

3 A Statistical Approach to Cracking the Enigma

3.1 Brute Force Solution

The brute force probability of predicting the right settings is the control with which we will compare our approach against. It can be calculated as follows:

$$P = \frac{|E|}{|S|}$$

where $|E|$ = one possible correct setting for the day and $|S|$ = (number of ways to arrange three rotors and set starting position) x (number of ways to connect 26 letters with 10 wires on plugboard). Hence, the complete calculation is:

$$P = \frac{|E|}{|S|} = \frac{1}{(5 \cdot 4 \cdot 3) \cdot (26^3) \cdot \sum_{k=0}^{13} \binom{26}{2 \cdot 10} \cdot \frac{(2 \cdot 10)!}{10! \cdot 2^{10}}}$$
$$= \frac{1}{1,054,560 \cdot 150,738,274,937,250} = \frac{1}{158,962,555,217,826,360,000}$$

This shows just how complex the Enigma encryption is and how unlikely it would be for a random brute force attempt to achieve the correct setting for a given day!

3.2 CS109 Approach: Rotors

3.2.1 Selection and Arrangement

There is not much modelling we can do to choose and arrange the rotors as regardless of the scenario, the selection and arrangement of rotors is equally likely. Hence, the selection and arrangement of three from the five rotors can be given by:

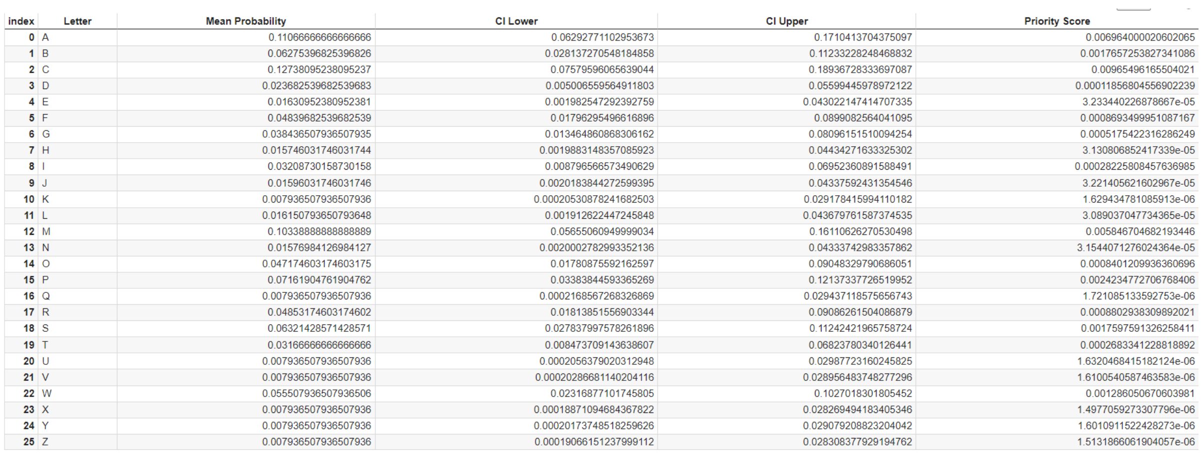
$$P = \frac{|E|}{|S|} = \frac{1}{(5 \cdot 4 \cdot 3)} = \frac{1}{60}$$

The correct selection and arrangement can be modelled by $X \sim \text{Ber}\left(\frac{1}{60}\right)$, which we will use later.

3.2.2 Rotor Starting Position

The goal of this part of our analysis is to maximize the probability of reaching the correct setting of rotor starting position. Our analysis is informed by the Blitzkrieg scenario explained earlier: it is best understood in conjunction with the code that I submitted on Gradescope.

- Step 1: Since no historical data was available, I simulated a dataset to determine the probabilities of each letter being a starting position for the rotor. This involved using Chat-GPT 4 to give me a list of 100 words that are most closely associated with Blitzkrieg.
- Step 2: I counted the frequency of all the first letters of the words in the dataset and applied Laplace Smoothing so that all letters had at least one occurrence.



index	Letter	Mean Probability	CI Lower	CI Upper	Priority Score
0	A	0.1106666666666666	0.06292771102953673	0.1710413704375097	0.006964000020602065
1	B	0.06275396825396826	0.028137270548184858	0.1123322824846832	0.0017657253827341086
2	C	0.12738095238095237	0.07579596065639044	0.18936728333697087	0.009654961655504021
3	D	0.023682539682539683	0.005006559564911803	0.05599445978972122	0.00011856804556902239
4	E	0.01630952380952381	0.001982547292392759	0.043022147414707335	3.233440226878667e-05
5	F	0.04839682539682539	0.01796295496616896	0.0899082564041095	0.0008693499951087167
6	G	0.038436507936507935	0.013464860868306162	0.08096151510094254	0.0005175422316286249
7	H	0.015746031746031744	0.0019883148357085923	0.04434271633325302	3.130806852417339e-05
8	I	0.03208730158730158	0.008796566573490629	0.06952360891588491	0.00028225808457636985
9	J	0.01596031746031746	0.0020183844272599395	0.04337592431354546	3.221405621602967e-05
10	K	0.007936507936507936	0.00020530878241682503	0.029178415994110182	1.62943478108591 3e-06
11	L	0.016150793650793648	0.001912622447245848	0.043679761587374535	3.089037047734365e-05
12	M	0.1033888888888889	0.05655060949999034	0.16110626270530498	0.005846704682193446
13	N	0.01576984126984127	0.0020002782993352136	0.04333742983357862	3.1544071276024364e-05
14	O	0.047174603174603175	0.01780875592162597	0.09048329790686051	0.0008401209936360696
15	P	0.07161904761904762	0.03383844593365269	0.12137337726519952	0.002423477270676840 6
16	Q	0.007936507936507936	0.0002168567268326869	0.029437118575656743	1.721085133592753e-06
17	R	0.04853174603174602	0.01813851556903344	0.09086261504086879	0.0008802938309892021
18	S	0.06321428571428571	0.027837997578261896	0.11242421965758724	0.0017597591326258411
19	T	0.0316666666666666	0.008473709143638607	0.06823780340126441	0.0002683341228818892
20	U	0.007936507936507936	0.0002056379020312948	0.02987723160245825	1.6320468415182124e-06
21	V	0.007936507936507936	0.00020286681140204116	0.028956483748277296	1.6100540587463583e-06
22	W	0.05550793650793650 6	0.0231687710174580 5	0.1027018301805452	0.001286050670603981
23	X	0.007936507936507936	0.0001887109468436782 2	0.028269494183405346	1.4977059273307796e-06
24	Y	0.007936507936507936	0.00020173748518259626	0.029079208823204042	1.6010911522428273e-06
25	Z	0.007936507936507936	0.0001906615123799911 2	0.028308377929194762	1.5131866061904057e-06

Figure 2: Probability and Priority Distribution After Bootstrapping (please zoom in if not clear)

- Step 3: I simulated 10,000 instances of starting letters based on the smoothed probabilities to mimic the process of selecting rotor starting positions in an operational context.
- Step 4: Each letter was represented as a beta distribution based on the counts from my dataset (with smoothing applied) as a prior to account for uncertainty in the estimated probabilities.
- Step 5: Using bootstrapping, I analyzed the variability in the distribution of starting letters and calculated mean probabilities for each letter being a starting position across all samples as well as the respective confidence intervals.
- Step 6: I generated rotor "priority scores" by multiplying the lower bounds of the confidence intervals by the mean probabilities for a statistically rigorous maximization of what could be the most likely rotor starting position.
- Step 7: Finally, I selected the top three rotor starting positions based on these scores.

The three highest priority scores belonged to 'A', 'C', and 'M', which means that we could use them for the rotor starting positions by multiplying their respective mean probabilities together.

$$P_{start} = P_A \cdot P_M \cdot P_C$$

$$P_{start} = 0.110666 \cdot 0.103388 \cdot 0.127380 = \frac{1457}{1000000}$$

Hence, the rotor starting position being the correct setting for a given day can be expressed as $Y \sim \text{Ber}\left(\frac{1457}{1000000}\right)$

3.3 CS109 Approach: Plugboard

The plugboard was essential for implementing a letter swapping mechanism using 10 wires to connect pairs of letters, leaving 6 letters unconnected. In reality, Turing was able to establish the correct pairings using **crib analysis**: a cryptographic approach. Since the letter swapping was randomly generated and there is no historic dataset available, we can statistically model this probability through a combinatoric approach. The probability of obtaining the correct setting as a Bernoulli random variable as we did for the other components of the machine. This can be expressed as follows:

$$P = \frac{|E|}{|S|} = \frac{1}{\sum_{k=0}^{13} \binom{26}{2\cdot 10} \cdot \frac{(2\cdot 10)!}{10!\cdot 2^{10}}}$$

$$P = \frac{1}{150{,}738{,}274{,}937{,}250}$$

The probability of getting the correct plugboard setting can be modelled as $Z \sim \text{Ber}\left(\frac{1}{150{,}738{,}274{,}937{,}250}\right)$.

3.4 CS109 Approach: Putting it All Together

Overall, our statistical approach of obtaining the correct setting for the Enigma can yield a probability calculated by multiplying the probability of success of the three Bernoullis together: X, Y, and Z. Hence,

$$P_{correct} = \frac{1}{60} \cdot \frac{1457}{1000000} \cdot \frac{1}{150,738,274,937,250}$$

$$P_{correct} = \frac{1457}{9,044,296,496,235,000,000,000}$$

4 Conclusion

We can compare our statistical approach to the brute force solution by dividing the probabilities to determine the ratio of success between them.

$$Ratio = \frac{P_{cs109}}{P_{bruteforce}} = \frac{1457}{9,044,296,496,235,000,000,000} \cdot \frac{158962555217826360000}{1} = 25.608$$

Hence, we can see that the CS109 approach is 25.6 times more probable to be determine the correct settings of the Enigma in a day!

Reflecting on the probabilities, both are still very small. The actual cryptography and logical deductions applied by Rejewski and Turing were far more advanced, and simply using this approach would be very unlikely to break the Enigma code. When compared with the brute force solution, the ingenuity in our approach lies within being able to account for surrounding context to model each letter as a beta random variable and choosing the most likely starting positions for a particular scenario. This mechanism can be applied to new scenarios in a similar way by simulating a dataset and following the same approach based on starting letter frequency distribution. Through this, I would love to explore new real-life applications of using the Beta variable and bootstrapping as a method of introducing context into cryptographic analyses to crack puzzles all around the world.

5 References

1. Figure 1: Diagram of the Enigma Machine: Sommervoll, Åvald Nilsen, Leif. (2020). Genetic algorithm attack on Enigma's plugboard. Cryptologia. 45. 1-33. 10.1080/01611194.2020.1721617. Fig 1

2. The Imitation Game (2014)

3. Singh, Simon. The Code Book, 1999.