Li-Vitanyi Approach to Noisy Channel Coding

Aman Chawla

September 25, 2024

Abstract

In this report, the authors have an LLM work out a version of the noisy channel coding theorem on the basis of Kolmogorov complexity and similar non-Shannon measures.

Consider the statement by Li and Vitanyi:

> Thus, we have discovered an interesting phenomenon: the description of some strings can be compressed considerably. In fact, there is no limit to the amount to which strings can be compressed, provided they exhibit enough regularity. This observation, of course, is the basis of all systems to express very large numbers, and was exploited early on by Archimedes in "The Sand Reckoner". However, if regularity is lacking, it becomes more cumbersome to express large numbers.

from the "Handbook of Theoretical Computer Science A", page 190. In the noisy channel coding theorem of Shannon, a random codebook consisting of M rows (messages), each of length n is generated from a pmf in an i.i.d. fashion. The above quote suggests that some of these strings don't require a length n representation, but can be captured by fewer entries. Let there be r such rows and let them be captured by $k < n$ entries. Collect these length k messages in one matrix and the remaining in another matrix or codebook. When the codebook(s) is to be used, get the message index, map it to the correct codebook and message within that codebook and transmit that message. For this modified procedure, in this paper we step through the proof of the noisy channel coding theorem, highlighting points where our new setup diverges from the standard proof.

In this modified setup of the noisy channel coding theorem, the central idea is to separate strings (messages) into two groups based on their compressibility. To examine the divergence from the standard proof, we proceed step by step and note where the differences arise.

Standard Proof of the Noisy Channel Coding Theorem

The standard proof of the noisy channel coding theorem can be summarized in these key steps:

Random Codebook Generation:

Generate a random codebook consisting of $M = 2^{nR}$ codewords, each of length n, drawn i.i.d. from a probability mass function (pmf).

Encoding and Transmission:

For each message, select a corresponding codeword from the codebook and transmit it over a noisy channel.

Decoding:

The receiver decodes the received noisy version of the codeword by searching for the codeword in the codebook that is "typical" relative to the received sequence, within a certain noise tolerance.

Error Probability Analysis:

The probability of decoding error is made arbitrarily small by ensuring the rate R is less than the channel capacity C, i.e., $R < C$.

Modified Steps in the Proof

1. Random Codebook Generation

Standard Proof:

A random codebook of $M = 2^{nR}$ codewords is generated, with each codeword of length n, drawn i.i.d. from the same pmf.

Modified Setup:

There is no divergence from the standard proof at this stage. All codewords are still generated as length-n strings in the same way as in the standard proof. The key difference arises after the codebook is generated, when the compressible and non-compressible messages are identified.

2. Partition of Codebook into Compressible and Non-compressible Messages

Standard Proof:

No such partitioning occurs; all codewords are treated equally.

Modified Setup:

After generating the full codebook, some messages are identified as compressible and can be represented with fewer entries, say $k < n$ symbols. This creates two sets of messages:

Compressible: r rows can be compressed to length k.

Non-compressible: The remaining $M - r$ rows retain their full length n.

Divergence: This step introduces the partitioning of the original codebook into two parts based on compressibility, leading to variable-length codewords.

3. Rate of the Code

Standard Proof:

The rate is $R = \frac{1}{n}\log(M)$, uniform across the entire codebook.

Modified Setup:

The overall rate now becomes a weighted sum of the rates for compressible and non-compressible messages. For compressible messages, the rate is effectively $R_k = \frac{1}{k}\log(r)$, and for non-compressible messages, the rate remains $R_n = \frac{1}{n}\log(M - r)$.

Divergence:

The rate analysis changes, as now there are effectively two different rates for compressible and non-compressible messages. The overall rate becomes a combination of these two: $R_{overall} = \frac{r}{M}R_k + \frac{M-r}{M}R_n$

This can affect the achievable rates relative to the channel capacity.

4. Encoding and Transmission

Standard Proof:

Each message is encoded into a length-n codeword and transmitted over the channel.

Modified Setup:

After determining whether a message is compressible or non-compressible, the encoding follows two paths: For compressible messages, the shorter length-k sequence is transmitted.

For non-compressible messages, the full length-n sequence is transmitted.

Divergence:

The transmission now involves codewords of variable lengths, adding complexity in terms of both encoding and channel utilization. The channel input distribution changes, as shorter messages (length k) are now mixed with longer messages (length n).

5. Decoding

Standard Proof:

The decoder looks for the codeword in the codebook that is "typical" relative to the received sequence, treating all codewords uniformly.

Modified Setup:

The decoder must first determine whether the received sequence corresponds to a compressible (length k) or non-compressible (length n) codeword. Then, it decodes the message within the appropriate subset of the codebook.

Divergence:

The decoding complexity increases because the decoder must now handle variable-length inputs. Additionally, typicality is defined relative to sequences of different lengths (k for compressible and n for non-compressible).

6. Error Probability Analysis

Standard Proof:

The probability of decoding error is made arbitrarily small as long as the rate R is less than the channel capacity C.

Modified Setup:

The overall error probability is now split into two parts: The probability of error for compressible messages depends on their shorter length k and the corresponding effective rate. The probability of error for non-compressible messages remains governed by length n sequences.

The total error probability is a weighted combination of the error probabilities for compressible and non-compressible messages:

$P_{error} = \frac{r}{M} P_{error,compressible} + \frac{M-r}{M} P_{error,non-compressible}$

Divergence:

The error analysis is more involved because the decoder has to distinguish between compressible and non-compressible messages, and typicality arguments apply differently based on the length of the message being decoded. The rate R must now be adjusted to ensure the error probability stays within the acceptable bounds for both parts of the codebook.

Conclusion

With the split occurring after codebook generation, the divergence from the standard noisy channel coding theorem begins at the partitioning stage (step 2). This affects the rate, encoding, transmission, decoding, and error probability analysis due to the introduction of variable-length codewords and the need to handle compressible and non-compressible messages separately. However, the i.i.d. generation of codewords remains intact, maintaining consistency with the standard setup up until the partitioning.

An analytical comparison of $R_{overall}$ and R

In this section we analytically compare the overall rate in the modified setup, R_{overall}, with the rate R in the standard noisy channel coding theorem.

Standard Rate R

In the standard noisy channel coding theorem, the rate R is defined as:

$$R = \frac{1}{n} \log M$$

where:
- M is the total number of messages.
- n is the length of each codeword.

Modified Setup:

Overall Rate R_{overall}

In your modified setup, the messages are partitioned into two groups:

1. Compressible Messages: r messages can be represented using k symbols ($k < n$).

2. Non-compressible Messages: $M - r$ messages require the full n-length representation.

Thus, we can define two separate rates:

- For compressible messages:

$$R_k = \frac{1}{k} \log r$$

- For non-compressible messages:

$$R_n = \frac{1}{n} \log(M - r)$$

Now, the overall rate R_{overall} is a weighted combination of these two rates:

$$R_{\text{overall}} = \frac{r}{M} R_k + \frac{M - r}{M} R_n$$

Substituting R_k and R_n into this expression:

$$R_{\text{overall}} = \frac{r}{M} \cdot \frac{1}{k} \log r + \frac{M - r}{M} \cdot \frac{1}{n} \log(M - r)$$

Analytical Comparison: R_{overall} vs. R

To compare R_{overall} with R, we break down their components and analyze the key factors:

1. Relative Proportion of Compressible and Non-compressible Messages

- If r (the number of compressible messages) is large compared to M, the term $\frac{r}{M} \cdot \frac{1}{k} \log r$ will dominate R_{overall}, leading to a rate that largely depends on the shorter codewords of length k.

- Conversely, if r is small, the second term $\frac{M-r}{M} \cdot \frac{1}{n} \log(M - r)$ dominates, so R_{overall} behaves similarly to the standard R.

2. Impact of Compressibility ($k < n$)

- The compressible messages (with length k) reduce the overall encoding length, thereby potentially reducing the rate required to transmit these messages.

- When $k \ll n$, the contribution of the compressible messages can significantly reduce R_{overall} compared to R, because these messages are represented more compactly than in the standard case.

3. Rate Difference

Next we can calculate the difference between the two rates:

$$\Delta R = R_{\text{overall}} - R = \left(\frac{r}{M} \cdot \frac{1}{k} \log r + \frac{M - r}{M} \cdot \frac{1}{n} \log(M - r) \right) - \frac{1}{n} \log M$$

Expanding this:

$$\Delta R = \frac{r}{M} \cdot \frac{1}{k} \log r + \frac{M - r}{M} \cdot \frac{1}{n} \log(M - r) - \frac{1}{n} \log M$$

If the proportion of compressible messages r is small, then $\log r$ will have a smaller influence, and R_{overall} will approach R. However, if r is large, $\log r$ becomes significant, and the overall rate R_{overall} can be lower than R because compressible messages take advantage of the smaller codeword length k.

4. Key Observations

- If $k \ll n$ and r is substantial compared to M, R_{overall} will be significantly less than R. The more compressible messages there are, and the smaller the length k, the more compression is achieved, reducing the overall rate.

- If r is small (i.e., most messages are non-compressible), then $R_{\text{overall}} \approx R$, since the majority of messages use full-length codewords.

- The balance between r and $M - r$, as well as the disparity between k and n, will determine how much R_{overall} deviates from R.

Conclusion

- When r is large and k is much smaller than n, the overall rate R_{overall} is likely to be significantly lower than R, reflecting the fact that compressible messages are using shorter representations.

- When r is small or $k \approx n$, the overall rate R_{overall} approaches R, as most messages require full-length encoding.

This analytical comparison shows that the modified setup can offer rate savings when a significant portion of the messages are compressible, especially if the compression factor k/n is substantial.

Benefits of the New Setup

Thus for a given M and n, we have managed to reduce the rate since $R_{overall} \leq R$. In this section we do the full analysis taking the channel to be a discrete memoryless channel, so that we can determine if we gain any advantage. For example, we may have improved the error exponent or the rate at which the probability of error decays with number of transmissions. Since the overall code is now variable-length (specifically, there are two lengths, k and n), one can perhaps treat the post-split analysis as two separate channels, perform the analysis for each, and then recombine the partial results to get the overall picture.

In what follows, to analyze the advantage gained in the modified setup for a discrete memoryless channel (DMC), we will step through how the error probability and error exponent (the rate at which the error probability decays with the number of transmissions) change based on this new structure.

Standard Noisy Channel Coding Theorem

For a standard DMC, let:

- X^n be the transmitted n-length codeword, with each symbol drawn i.i.d. according to some probability mass function (pmf).

- The channel is characterized by a conditional probability distribution $P_{Y|X}$, where Y^n is the received sequence.

- The codebook consists of M codewords, each of length n.

- The channel transition probabilities $P_{Y|X}$ are memoryless, meaning:

$$P(Y^n|X^n) = \prod_{i=1}^{n} P(Y_i|X_i)$$

For any rate $R = \frac{1}{n} \log M$ below the channel capacity C, the probability of decoding error P_e decreases exponentially with n:

$$P_e \leq 2^{-nE(R)}$$

where $E(R)$ is the error exponent (or reliability function) that depends on the rate R and the properties of the channel.

Modified Setup: Two Lengths k and n

In the modified setup, the messages are split into two sets:

1. Compressible messages: r messages, which are represented by codewords of length k.

2. Non-compressible messages: $M - r$ messages, which are represented by codewords of length n.

We will treat this as two separate communication channels with the same DMC:

- One channel for the compressible messages (length k).

- One channel for the non-compressible messages (length \(n \)).

Error Exponent for Compressible and Non-compressible Messages

1. Compressible Messages (length k)

For the r compressible messages, each represented by a codeword of length k, the rate is:

$$R_k = \frac{1}{k} \log r$$

The probability of error for these messages decays exponentially with k, so we have:

$$P_e^k \leq 2^{-kE(R_k)}$$

where $E(R_k)$ is the error exponent corresponding to the rate R_k.

2. Non-compressible Messages (length n)

For the $M - r$ non-compressible messages, each represented by a codeword of length n, the rate is:

$$R_n = \frac{1}{n} \log(M - r)$$

The probability of error for these messages decays exponentially with n, so we have:

$$P_e^n \leq 2^{-nE(R_n)}$$

where $E(R_n)$ is the error exponent corresponding to the rate R_n.

Overall Error Probability

To compute the overall error probability for the modified setup, we can combine the error probabilities of the compressible and non-compressible messages. Since these are two disjoint sets of messages, the overall error probability is the weighted sum of the error probabilities for the two groups.

The overall error probability P_e^{overall} is:

$$P_e^{\text{overall}} = \frac{r}{M} P_e^k + \frac{M-r}{M} P_e^n$$

Substituting the bounds for P_e^k and P_e^n:

$$P_e^{\text{overall}} \leq \frac{r}{M} 2^{-kE(R_k)} + \frac{M-r}{M} 2^{-nE(R_n)}$$

This gives us the overall error bound in the modified setup, accounting for the two types of messages with different lengths.

Error Exponent for the Overall System

Next, we analyze the overall error exponent E_{overall}, which controls how fast the error probability decays with the number of transmissions. In the standard setup, the error probability decays as $2^{-nE(R)}$, but now we have two rates and two error exponents.

Define the overall error exponent E_{overall} such that:

$$P_e^{\text{overall}} \leq 2^{-n_{\text{eff}} E_{\text{overall}}}$$

where n_{eff} is the effective length capturing the variable-length nature of the code. We can approximate n_{eff} as the expected length of a codeword:

$$n_{\text{eff}} = \frac{r}{M} k + \frac{M-r}{M} n$$

This is a weighted average of the lengths k and n.

Now, the overall error exponent E_{overall} can be approximated by combining the error exponents for the two groups:

$$E_{\text{overall}} = \frac{r}{M} E(R_k) + \frac{M-r}{M} E(R_n)$$

Thus, the overall error probability decays approximately as:

$$P_e^{\text{overall}} \leq 2^{-n_{\text{eff}} E_{\text{overall}}}$$

Comparison with Standard Setup

In the standard setup, the probability of error decays as $2^{-nE(R)}$, where n is the fixed length of the codewords and $E(R)$ is the error exponent for the given rate $R = \frac{1}{n} \log M$.

In the modified setup:

- The effective length n_{eff} is shorter than n, provided $k < n$ and r is non-zero.

- The overall error exponent E_{overall} is a weighted combination of the error exponents for the two groups, $E(R_k)$ and $E(R_n)$.

Thus, the modified setup may improve the error probability in two ways:

1. Faster error decay: The overall code may achieve a faster decay in error probability due to the lower effective length n_{eff}.

2. Improved error exponent: The overall error exponent E_{overall} is a combination of two different exponents. If $E(R_k)$ (for compressible messages) is significantly larger than $E(R_n)$, this can further improve the overall reliability.

Conclusion

In the modified setup:

- The overall rate R_{overall} is less than or equal to the standard rate R, indicating a potential gain in compression efficiency.

- The effective length n_{eff} is shorter than n, leading to a potentially faster decay of the error probability.

- The overall error exponent E_{overall} is a combination of the error exponents for the two groups, potentially improving the system's reliability.

Overall, this suggests that the modified codebook structure provides a possible advantage in terms of both rate and reliability, particularly if the compressible messages are a significant fraction of the total and the compression factor k/n is substantial.

Identifying Regularity

The remaining question is, how does one identify "regularity" in bit strings such that some of the messages can be sent on the length-k channel. Identifying "regularity" in bit strings that allows compression is a fundamental problem in both data compression and coding theory. For the modified setup, where some messages can be sent on a shorter channel (of length k), this "regularity" would imply that certain bit strings can be represented in a more compact form.

Defining Regularity in Bit Strings

In essence, regularity in bit strings refers to patterns, repetitions, or predictability that can be exploited to reduce the number of bits required for transmission. The more regularity a string has, the more it can be compressed. There are a few techniques that can be used to detect and quantify regularity in bit strings:

1. Kolmogorov Complexity - Kolmogorov complexity measures the shortest possible description of a bit string. It is the length of the shortest computer program that can generate the string. If a bit string has low Kolmogorov complexity, it is highly regular and can be described by a small program.

2. Pattern Recognition (Repetitive Patterns) - Bit strings with clear repetitive patterns, such as '01010101' or '111000111', can be compressed efficiently. This can be detected using:

- Run-length encoding (RLE), where consecutive bits of the same value are compressed.

- Lempel-Ziv compression, which identifies repeated substrings and replaces them with shorter references.

3. Markov Models (Low-order Markov Chains) - Markov chains model sequences where the probability of each symbol depends on a fixed number of preceding symbols (the order of the Markov chain). Bit strings with predictable transitions can be compressed based on this structure.

Post-split Codebook: Routing Messages to Channels

Once we have identified regular messages that can be compressed, we can proceed with the routing of messages:

1. Compressible Messages:

- After identifying messages that exhibit regularity and are compressible, store these in the compressed codebook with codewords of length k.

- Use a mapping function to efficiently encode these messages.

2. Non-compressible Messages:

- The remaining messages that show no regularity remain in the standard codebook and are transmitted on the full-length n-channel.

Final Considerations and Future Work

The challenge is balancing the complexity of detecting regularity (which can be computationally expensive) with the gains in compression and transmission efficiency. The identification of regularity must be done after codebook generation, meaning it is a post-processing step to reduce redundancy in certain messages.

In conclusion, to determine whether a message can be sent over the shorter k-length channel, we focus on pattern recognition, and approximate measures of Kolmogorov complexity using practical compression algorithms like run-length encoding. These methods allow us to flag "regular" messages and split them off for transmission on the more efficient channel.

In future work, we will do a full probability of error analysis for this split codebook procedure along the lines of Shannon's noisy channel coding theorem proof [1].

Acknowledgment

The author thanks ChatGPT for assisting with the analysis.

References

[1] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. 2005.