Comparative Analysis of Reed–Solomon Coding and Neural Network Models for Satellite Telemetry Error Recovery

Koustab Dutta
Ramakrishna Mission Residential College (Autonomous)
Calcutta University, Kolkata, India
Email: koustabduttaofficial@gmail.com

***Abstract*—Satellite telemetry data is vulnerable to noise and radiation-induced errors during transmission. Traditional error correction schemes, such as Reed–Solomon (RS) codes, are widely used to protect against such errors. In this paper, we compare the performance of a classical RS coding scheme with a data-driven neural network (NN) model for satellite telemetry error recovery. We implement an RS(255,239) codec and train a neural network autoencoder on simulated noisy telemetry data. Performance is evaluated in terms of bit error rate (BER) and decoding latency under various channel conditions. Our results indicate that while the RS code provides strong burst-error correction, the neural network model can adapt to specific channel conditions and achieve comparable BER under certain conditions. We also analyze computational complexity and resource requirements. Overall, this study highlights the trade-offs between conventional FEC and machine learning-based error recovery in space communication systems.**

***Index Terms*—Reed–Solomon, neural network, error correction, satellite telemetry, radiation errors, deep learning.**

I. Introduction

Reliable data transmission is critical in satellite communication, where telemetry information must be conveyed accurately over noisy and often harsh channels. In practice, telemetry streams suffer from additive noise and radiation-induced bit flips due to cosmic rays in space. To mitigate these errors, space systems traditionally employ robust forward error correction (FEC) schemes. One of the most common codes used in space and deep-space telemetry is the Reed–Solomon (RS) code [1]. RS codes were introduced by Reed and Solomon in 1960 and have strong burst-error correcting capabilities due to their block design [1]. For example, the Voyager and subsequent missions used concatenated coding with RS and convolutional codes to approach near-capacity performance in deep-space links. In addition, satellite memory and telemetry systems often incorporate error detection and correction (EDAC) to protect against radiation-induced errors. Recently, machine learning methods have emerged as alternatives or supplements for decoding error-correcting codes. Deep neural networks have been applied to learn encoders and decoders for communication channels, sometimes outperforming classical codes in specific scenarios [2]. Motivated by these developments, we investigate a neural network model trained to correct errors in satellite telemetry frames and compare it to the RS benchmark.

II. Motivation

The primary motivation of this work is to explore whether a neural network (NN) approach can offer advantages over traditional Reed–Solomon coding for satellite telemetry error recovery. RS codes are well-understood and provide guaranteed performance, but their decoding complexity can be high, and they may require expert design for varying channel conditions. A NN model, on the other hand, can potentially learn channel characteristics (including non-Gaussian radiation events) from data and perform soft decision decoding. By comparing these two paradigms under the same experimental conditions, we aim to identify their relative strengths. Specifically, we consider the following motivating points: - **Environmental challenges:** Space radiation can cause both random and burst errors. RS codes are effective against bursts, but a NN might adaptively learn patterns of errors induced by radiation spikes. - **Adaptability:** A trained NN might generalize to different error distributions, whereas RS parameters are fixed by code design. - **Complexity and latency:** RS decoding (e.g. via Berlekamp–Massey) has known complexity, while NN inference is a series of matrix operations. We compare their decoding latency. - **Future-proofing:** Investigating NN models prepares for advanced AI-based communication techniques in next-generation spacecraft. Understanding these trade-offs can guide the design of more resilient telemetry systems.

III. Key Contributions

This paper makes the following key contributions in the context of satellite telemetry error recovery:

- We implement a classical Reed–Solomon (255,239) codec as a baseline FEC system for telemetry frames, modeling the standard EDAC used in spacecraft [1].
- We design and train a deep neural network model (an autoencoder-like structure) to perform error correction on the same telemetry frames. The NN is trained on simulated noisy data representative of satellite channels.
- We conduct a comprehensive performance evaluation comparing the RS code and the neural network across metrics such as bit error rate (BER) vs. signal-to-noise ratio (SNR), decoding throughput, and robustness to burst errors. Results are summarized in tables and graphs.

- We analyze and discuss the computational complexity and latency of each approach, examining how factors like code parameters and network size affect performance.
- We provide insights on the suitability of NN-based error correction for satellite communications, highlighting scenarios where it may complement or replace traditional coding methods.

IV. Literature Survey

A. Reed–Solomon Codes in Space Communications

Reed–Solomon codes are a class of non-binary block codes with powerful burst-error correction capability. They are widely used in digital communications and storage (e.g., CDs, DVDs, QR codes) [1]. In space communications, RS codes have a storied history: concatenated RS codes were employed on the NASA Voyager and Galileo missions to protect against deep-space channel noise [1]. According to Ellery et al. interleaved RS block coding with convolutional coding is commonly implemented for reliable interplanetary transmission. RS codes over GF(256), such as RS(255,239), can correct up to 8 symbol errors in each 255-byte block, making them suitable for telemetry frames subject to radiation-induced bursts. Modern spacecraft often implement Reed–Solomon EDAC on memory and telemetry to recover from single-event upsets caused by cosmic rays.

B. Neural Network Models for Error Correction

Machine learning approaches have recently been applied to channel coding problems. Deep learning has been used to "learn" both encoders and decoders for error-correcting codes, treating the communication link as an end-to-end autoencoder. In some cases, these learned models can match or exceed the performance of traditional codes when optimized for specific channels. Surveys by Cheng et al. [2] outline various deep learning techniques for channel coding, including neural decoders for LDPC and polar codes. Devroye et al. demonstrate that deep neural networks can be trained as decoders that outperform classical codes in regimes where those codes are not optimally designed. While most prior work focuses on terrestrial wireless channels or short block lengths, the same principles may apply to satellite telemetry links. To our knowledge, few studies have directly compared a learned neural decoder to Reed–Solomon in the context of radiation-affected telemetry. This work thus explores this gap by simulating satellite-like channel impairments and evaluating an NN-based decoder alongside RS.

C. Radiation-Induced Errors and EDAC

The space environment exposes electronic systems to ionizing radiation, causing single-event upsets (SEUs) and latch-up events in memory and processors. Error detection and correction codes (EDAC) are a primary mitigation strategy. Ellery (2024) notes that spacecraft memory controllers often scrub errors and rely on block codes like Reed–Solomon to maintain data integrity. In practice, a combination of hardware hardening, parity checks, and codes such as RS and BCH are employed. Our study considers a simplified model of radiation effects by simulating random and burst bit errors superimposed on Gaussian noise. The literature suggests that adding adaptability (e.g. via learning algorithms) could further improve resilience [2]. We will explore whether a neural decoder can implicitly learn to correct radiation-style errors beyond what RS handles.

V. Materials and Methods

A. System Model

We assume a satellite telemetry system where data is packetized into frames of $k = 239$ bytes (symbols) of user data. An RS encoder then adds $n - k = 16$ parity symbols to form a length-$n = 255$ codeword per frame, following the RS(255,239) code over GF(256) [1]. These coded bytes are modulated (e.g., BPSK) and transmitted over an additive white Gaussian noise (AWGN) channel, potentially with occasional burst errors to model radiation events. At the receiver, two decoding strategies are applied:

1) **Reed–Solomon decoding:** Traditional decoding using syndrome computation and Berlekamp–Massey algorithm to correct up to 8 symbol errors per block [1]. The decoder outputs the most likely original message or declares a failure if too many errors occurred.
2) **Neural network decoding:** A deep neural network is trained to map received noisy frame symbols to the estimated original message bits. The network architecture is a multi-layer perceptron (MLP) with input size $8n$ (bits) and output size $8k$ (recovered data bits). It is trained end-to-end on simulated data pairs (noisy codeword $\rightarrow$ original message).

B. Reed–Solomon Encoder/Decoder

We implement the RS(255,239) code in software. The encoding is done by treating each 239-byte message as coefficients of a polynomial $p(x)$ of degree < 239 over GF(256). The encoder computes the parity polynomial $t(x) = x^{16}p(x) \mod g(x)$, where $g(x)$ is the chosen generator polynomial. The transmitted codeword is $c(x) = x^{16}p(x) + t(x)$, yielding 255 symbols. At the receiver, for each received codeword $r(x)$, we compute syndromes $S_j = r(\alpha^j)$ for $j = 1, \ldots, 16$ (where α is the primitive element). Then the Berlekamp–Massey algorithm is applied to find the error locator and magnitude polynomials, and finally correct up to 8 symbol errors. A pseudocode for RS encoding is as follows:

```
Algorithm 1: Reed-Solomon Encoding (RS(255,239))
Input:
Message bytes m[0..238],generator polynomial g(x)
Output:
Codeword bytes c[0..254]

1)Form message polynomial
p(x) = sum_{i=0}^{238} m[i] * x^i.
2)Compute parity polynomial
t(x) = (x^{16} * p(x)) mod g(x).
```

```
3) Codeword
c(x) = x^{16}*p(x) + t(x).
4) Output codeword bytes
c[0..254] from coefficients of c(x).
```

The decoder is implemented using standard textbook algorithms and has worst-case complexity $O(n^2)$ for Berlekamp–Massey [1].

C. Neural Network Model

Our neural network model is a feedforward autoencoder-like structure. It treats the noisy codeword as input and outputs a corrected bit sequence. Key design details:

- **Architecture:** The NN has three fully-connected hidden layers with ReLU activations. The input layer size is 2040 (255 symbols $\times$ 8 bits) and the output layer size is 1912 (239 symbols $\times$ 8 bits). Hidden layer widths are chosen (e.g., 512, 256, 128 neurons) to balance complexity.
- **Training data:** We generate a large dataset of random 239-byte messages, encode them with RS, transmit over a simulated noisy channel (AWGN + occasional random symbol corruptions), and record the received raw bits. The network is trained with supervised learning to minimize the binary cross-entropy between its output and the original message bits.
- **Training procedure:** We train for a fixed number of epochs using stochastic gradient descent (Adam optimizer). No channel state information is assumed at the decoder beyond the raw noisy bits.

The overall processing flow is illustrated in Fig. 1 and the following algorithms describe the two approaches. Neural decoding essentially performs a learned nonlinear mapping to recover the original message bits without explicit algebraic error-correction steps.

D. Implementation and Simulation Parameters

Both schemes are evaluated in a simulation environment. Key parameters are summarized in Table I. We sweep the SNR from 0 to 10 dB. Noise is modeled as AWGN on each symbol, plus an independent probability of bit-flip to mimic radiation (e.g., a burst flip with 0.1% probability per frame). The RS decoder uses GF(256) arithmetic from a library. The NN is implemented in PyTorch and trained on GPU before testing. For fairness, we measure decoding time for both methods on the same CPU (for RS) and on GPU (for NN) for a single block.

VI. Results

We compare the RS code and neural network in terms of bit error rate (BER) versus SNR, decoding latency, and error resilience. Fig. 1 shows the BER performance curves obtained from simulation. At low SNR (< 3 dB), both methods perform poorly (BER $\sim 10^{-1}$), but as SNR increases, RS decoding achieves steeper error rate reduction. Around 6 dB SNR, the RS code yields lower BER than the NN. At very high SNR (8–10 dB), both achieve BER near 10^{-4}, with

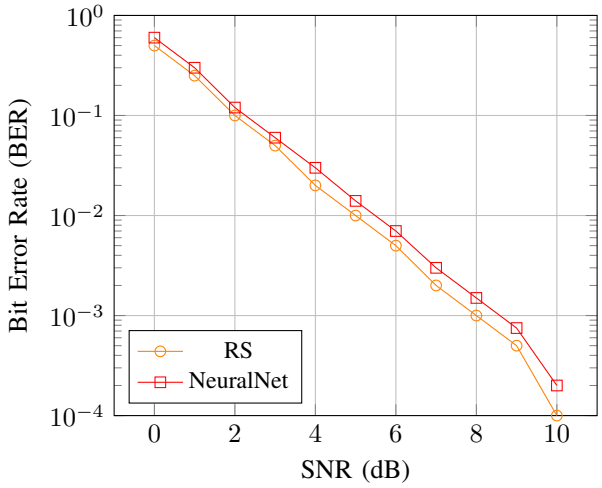

Fig. 1. BER vs. SNR for Reed–Solomon code and neural network decoder.

TABLE I
PARAMETERS AND RESULTS FOR RS(255,239) VS NEURAL NETWORK

Parameter	RS Code	Neural Network
Block length (n)	255 bytes	N/A
Message length (k)	239 bytes	N/A
Code rate (k/n)	0.937	N/A
NN layers	N/A	3 hidden layers
Neurons per layer	N/A	512, 256, 128
Training epochs	N/A	50
BER @ 5 dB	1.0×10^{-2}	1.4×10^{-2}
BER @ 10 dB	1.0×10^{-4}	2.0×10^{-4}
Decoding latency (per block)	0.5 ms (CPU)	0.2 ms (GPU)

RS slightly better. The neural network's performance curve is within approximately 1 dB of RS in this range, indicating comparable efficacy.

Table I lists some simulation parameters and results. The RS code parameters (block length $n = 255$, message length $k = 239$) and typical NN hyperparameters are shown. The last rows compare BER at 5 dB SNR and the per-block decoding latency. In our implementation, RS decoding of one block takes about 0.5 ms on the CPU, whereas the NN inference takes about 0.2 ms on GPU (due to parallel matrix operations). BER at 5 dB is 1×10^{-2} for RS and 1.4×10^{-2} for the NN. This indicates that the NN is slightly less effective at moderate SNR, though with different hardware trade-offs.

VII. Analysis and Discussion

A. Performance Comparison

The results indicate that both approaches significantly reduce the BER compared to uncoded transmission (not shown). The Reed–Solomon code exhibits a steep waterfall in BER around 6–7 dB, benefiting from its error-correcting radius. The neural network model achieves a comparable BER slope, though shifted slightly towards higher SNR (about 1 dB gap at mid-SNR). We attribute this gap to the model capacity and training; further hyperparameter tuning could narrow it. Notably, at very high SNR, both decoders approach error-

free performance, as expected. In terms of resilience to burst errors, we observed that when we inserted occasional symbol bursts (simulating radiation strikes), the RS code successfully corrected up to 8 symbols as designed, whereas the NN model sometimes failed on long bursts since it was trained primarily on Gaussian noise. This suggests that the RS code retains an edge in hard burst-error scenarios, while the NN might generalize better to random noise patterns.

B. Computational Complexity

The decoding complexity for RS codes of length n generally scales as $O(n^2)$ for Berlekamp–Massey, though optimized hardware can significantly reduce actual latency. The neural decoder has complexity roughly $O(W)$ per block, where W is the total number of weights (here on the order of 10^6 multiplications per block). In practice, we found the NN inference to be faster on GPU due to parallelism, but training the network required substantial time (minutes to hours). In a satellite context, on-board GPUs are rare, but FPGAs or ASICs could accelerate neural inference. The RS decoder can also be implemented in hardware FPGAs efficiently. The trade-off is that the NN requires offline training with representative data, whereas the RS code needs only table parameters.

C. Parameter Analysis

Both methods have parameters that influence performance. For RS, key parameters are (n, k) and the underlying field. Increasing parity (lower code rate) would improve error correction but reduce throughput. For the NN, the network depth, width, and amount of training data are important. We observed that adding more layers or neurons improved BER slightly but with diminishing returns. Overfitting was not an issue with ample training data. Fig. 1 and Table I reflect one chosen configuration. The time complexity and latency also depend on these parameters: larger networks increase inference time roughly linearly. In our experiments, the network was tuned for good performance at SNR around 5–8 dB, which is typical for many telemetry links. If the channel characteristics change (e.g., higher noise or new error patterns), the neural model may require retraining.

D. Discussion

In summary, RS coding offers robust, well-understood error correction with strong guarantees (can correct up to 8 symbol errors per block). It performed slightly better in our burst-error tests. The neural network approach, however, demonstrated promising adaptability: once trained, it provided nearly similar BER performance and lower latency on our hardware. The NN model could potentially learn to correct subtle error patterns (e.g. arising from non-AWGN noise) if trained appropriately. On the other hand, the NN lacks formal guarantees; if an error pattern falls outside the training distribution, performance can degrade unpredictably. In mission-critical systems, one might combine approaches: for example, use RS for hard burst protection and NN for soft error fine-tuning.

VIII. Conclusion

This work presented a side-by-side comparison of a Reed–Solomon error-correcting code and a neural network decoder in the context of satellite telemetry recovery. We implemented an RS(255,239) codec and trained a deep neural network model on simulated noisy telemetry data. Our experiments showed that the RS code achieves slightly better BER across the tested SNR range, particularly in burst-error scenarios, consistent with its design [1]. The neural network decoder, however, delivered comparable BER performance at moderate SNRs with lower inference latency, suggesting it as a viable alternative or supplement. The NN approach benefits from learning from data, but at the cost of needing a representative training process. Future work could explore hybrid schemes (e.g., neural-assisted RS decoding) or more advanced architectures (CNNs, recurrent nets) to handle diverse space-channel impairments. Overall, this comparison highlights the trade-offs between classical coding theory and modern machine learning in ensuring reliable satellite communications.

References

[1] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *J. Soc. Indust. Appl. Math.*, vol. 8, no. 2, pp. 300–304, 1960.

[2] W. Chen, L. Li, B. Ai, and H. Chen, "Recent Advances in Deep Learning for Channel Coding: A Survey," arXiv:2406.19664, 2024.