УДК 007.3
DOI: 10.26102/2310-6018/2025.48.1.026

Neural network to optimize the adaptive exponential min sum decoding algorithm

Zhang Weijia✉, Ibrahem Mouhamad, V.M. Saklakov, Dushantha Nalin K. Jayakody

National Research Tomsk Polytechnic University, Tomsk, the Russian Federation

Abstract. Currently, deep learning, as a hot research direction, has yielded fruitful research results in natural language processing and graph recognition and generation, such as ChatGPT and Sora. Combining deep learning with decoding algorithms for channel coding has also gradually become a research hotspot in the field of communication. In this paper, we use deep learning to improve the adaptive exponential min sum (AEMS) algorithm for LDPC codes. Initially, we extend the iterative decoding procedure between check nodes (CNs) and variable nodes (VNs) in the AEMS decoding algorithm into a feedforward propagation network based on the Tanner graph derived from the H matrix of LDPC codes. Second, in order to improve the model training efficiency and reduce the computational complexity, we assign the same weight factor to all the edge information in each iteration of the AEMS decoding network, which reduces the computational complexity while guaranteeing the decoding performance, and we call it the shared neural AEMS (SNAEMS) decoding network. The simulation results show that the decoding performance of the proposed SNAEMS decoding network outperforms that of the conventional AEMS decoder, and its coding gain is gradually enhanced as the code length increases.

Keywords: LDPC, deep learning, neural network, exponential algorithm, min sum.

For citation: Zhang Weijia, Ibrahem Mouhamad, Saklakov V.M., Dushantha Nalin K. Jayakody. Neural network to optimize the adaptive exponential min sum decoding algorithm. *Modeling, Optimization and Information Technology*. 2025;13(1). URL: https://moitvivt.ru/ru/journal/pdf?id=1807 DOI: 10.26102/2310-6018/2025.48.2.026

Нейронная сеть для оптимизации адаптивного экспоненциального алгоритма декодирования минимальной суммы

Чжан Вэйцзя✉, Ибрагим Мухамад, В.М. Саклаков, Душанта Налин К. Джаякоди

Национальный исследовательский Томский политехнический университет, Томск, Российская Федерация

Резюме. В настоящее время глубокое обучение как актуальное направление исследований дало плодотворные результаты в области обработки естественного языка, распознавания и генерации графов, например, ChatGPT и Sora. Объединение глубокого обучения с алгоритмами декодирования для канального кодирования также постепенно становится актуальным направлением исследований в области связи. В этой статье мы используем глубокое обучение для улучшения адаптивного алгоритма экспоненциальной минимальной суммы (AEMS) для LDPC-кодов. Во-первых, мы расширяем итеративную процедуру декодирования между контрольными узлами (CNs) и переменными узлами (VNs) в алгоритме декодирования AEMS в сеть распространения с прямой передачей, основанную на графе Таннера, полученном из H-матрицы LDPC-кодов. Во-вторых, для повышения эффективности обучения модели и снижения вычислительной сложности мы присваиваем одинаковый весовой коэффициент всей краевой информации в каждой итерации сети декодирования AEMS, что снижает вычислительную сложность и гарантирует эффективность декодирования, и мы называем ее общей нейронной сетью декодирования AEMS (SNAEMS). Результаты моделирования показывают, что производительность декодирования предложенной сети декодирования SNAEMS превосходит производительность обычного декодера AEMS, а коэффициент усиления кодирования постепенно увеличивается по мере увеличения длины кода.

Ключевые слова: LDPC, глубокое обучение, нейронная сеть, экспоненциальный алгоритм, минимальная сумма.

Для цитирования: Чжан Вэйцзя, Ибрагим Мухамад, Саклаков В.М., Душанта Налин К. Джаякоди. Нейронная сеть для оптимизации адаптивного экспоненциального алгоритма декодирования минимальной суммы. *Моделирование, оптимизация и информационные технологии.* 2025;13(1). (На англ.). URL: https://moitvivt.ru/ru/journal/pdf?id=1807 DOI: 10.26102/2310-6018/2025.48.2.026

Introduction

Deep learning simulates the working mode of the human brain by constructing a multi-layer neural network [1]. It has been widely used in face recognition, natural language processing and autonomous driving [1, 2]. Low Density Parity Check (LDPC) code is a linear block code firstly introduced by Gallager in 1962 [3]. Its characteristic is that the decoding capability increases with the increase of code length. It has become the main channel coding for 5G due to its high decoding throughput and excellent decoding performance. Deep learning can also be well combined with the decoding algorithm of LDPC code to improve its decoding performance [4, 5].

Soft decision approaches to LDPC codes, including the belief propagation (BP) algorithm and the min sum (MS) algorithm, are iterative algorithms based on Tanner graphs and can be relatively easily expanded into a neural network structure [6]. When the Tanner graph is expanded, each iterative decoding process is considered separately and each side information is assigned a weight. The resulting decoder is a neural network structure. The performance of this "neural network decoder" is better than that of traditional soft decision decoding algorithms because they use appropriate weights to enhance the performance of the algorithm during the iteration process through training.

In this paper, we use neural networks to optimize the adaptive exponential min sum (AEMS) decoding algorithm. Since the BP algorithm requires extensive logarithmic and multiplication operations for calculating check node (CN) messages. In contrast, the MS decoding algorithm substantially decreases computational demands and device complexity, but at the expense of decoding performance [7]. The AEMS decoding algorithm employs an enhanced MS algorithm featuring an adaptive exponential correction factor, resulting in superior decoding performance compared to the BP algorithm, while preserving lower complexity [8]. We extend the iterative process between the CNs and the variable nodes (VNs) in the AEMS algorithm into a feedforward propagation network to enhance decoding performance. In order to improve the training efficiency of the neural network, we use the same weight factor for the edge information of each layer of the AEMS decoding network, which greatly reduces the number of multipliers in the AEMS decoding network. We call it the shared neural adaptive exponential min sum (SNAEMS) decoding network. The (SNAEMS) decoding network is fast to train and has better decoding performance than the traditional AEMS decoder.

Tanner graph and AEMS decoding algorithm

Every LDPC code can be characterized by a sparse check matrix H, and every H check matrix corresponding to a Tanner graph. The check matrix H contains a limited number of "1" elements, and its associated Tanner graph is a bidirectional structure with three components: check nodes (CNs), variable nodes (VNs), and edges linking CNs and VNs [9]. VNs are denoted by squares, while CNs are represented by circles. If $\mathrm{H(i,j)}$ equals "1", then VN_i is connected to CN_j, otherwise it is not connected. A 3 row 6 column H check matrix and its corresponding Tanner graph are shown in Figure 1. The soft decision decoding algorithm is an iterative decoding method based on Tanner graph, in which messages are exchanged back and forth between VNs and CNs during the iteration process. After several iterations, the obtained message values are stabilized so that the best decision can be made accordingly. In the decoding process, the information produced by the CNs and VNs updates is termed external information, whereas the initial information obtained from the channel at the start of decoding is referred to as a posteriori probability.

$$\boldsymbol{H} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \longrightarrow$$

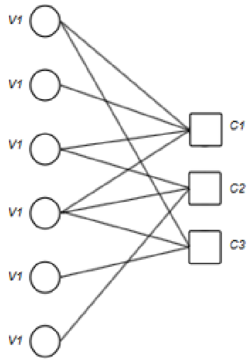

Figure 1 – LDPC (6, 3) H check matrix and its corresponding Tanner graph
Рисунок 1 – LDPC (6, 3) контрольной матрицы H и соответствующий ей граф Таннера

The AEMS decoding algorithm is a soft decision algorithm modified on the basis of the MS decoding algorithm. The MS algorithm simplifies the complex calculation of the CNs update formula of the BP algorithm by replacing all external information with the minimum external information. On this basis, the AEMS algorithm additionally introduces secondary (secondary minimum) external information values to optimize the CNs update operation of the MS algorithm. The decoding performance of the MS algorithm is improved by mathematically transforming the secondary minimum external information value and using it as an exponential correction factor. Compared with existing MS algorithm optimization methods, the AEMS algorithm maintains a low computational complexity and has very strong decoding performance [8]. The specific decoding process is as follows and the detailed meaning of the algorithm variables is shown in Table 1. This section briefly introduces the AEMS decoding algorithm. For the detailed reasoning and proof process of the AEMS algorithm, please refer to [8].

Table 1 – Symbol explanation in the calculation process
Таблица 1 – Интерпретация символов при расчетах

Symbols	Meaning
v_i	The *i* variable node
c_j	The *j* check node
$r_{ji}^l(b)$	The external information passed from check node *j* to variable node *i* in the *l*th iteration, b = 0, 1
$q_{ij}^l(b)$	The external information passed from variable node *i* to check node *j* in the *l*th iterations, b = 0, 1
$C(i)$	The set of all check nodes connected to the *i*th variable node
$V(j)$	The set of all variable nodes connected to the *j*th check node
$C(i)$	The set of check nodes connected to the *i*th variable node except the *j*th check node
$V(j) \setminus i$	The set of variable nodes connected to the *j*th check node except the *i*th variable node
$P_i(b)$	The posterior probability of receiving y_i at the receiving end, corresponding to the code word $c_i = b$ at the sending end, b = 0, 1
$q_i^l(b)$	The posterior probability information of the *i*th variable node of the *l*th iteration, b = 0, 1

Step 1: Calculate the received channel message of VNs for initialization as shown in (1).

$$L^0\left(q_{ij}\right) = L(P_i) = ln\frac{P_i(0)}{P_i(1)}. \quad (1)$$

Step 2: Calculate the CN-to-VN messages as described in (2). The AEMS algorithm not only introduces the smallest external information value $\left|L^{l-1}\left(q_{i'j}\right)_{min1}\right|$, but also additionally introduces the second smallest external information value $\left|L^{l-1}\left(q_{i'j}\right)_{min2}\right|$, using EI_{m1} and EI_{m2} instead of them to simplify the expression. In order to use EI_{m1} and EI_{m2} to improve the decoding performance, they are mathematically deformed to $\lambda = 2 - (EI_{m2} - EI_{m1})$ as an exponential correction factor. Whether λ is substituted as an exponential correction factor depends on whether $EI_{m2} \leq 1$ or not.

$$\begin{cases} L^l(r_{ji})_{AEMS} = \prod \quad sgn(EI)EI_{m1}{}^{\lambda}, \ EI_{m2} \leq 1 \\ L^l(r_{ji})_{AEMS} = \prod \quad sgn(EI)EI_{m1}, \ EI_{m2} > 1 \end{cases}. \quad (2)$$

Step 3: Calculate the CN-to-VN messages as shown in (3).

$$L^l\left(q_{ij}\right) = ln\frac{P_i(0)\prod_{j'\in C(i)\setminus j} r^l_{j'i}(0)}{P_i(1)\prod_{j'\in C(i)\setminus j} r^l_{j'i}(1)} = L(P_i) + \sum_{j'\in C(i)\setminus j} \ L^l\left(r_{j'i}\right). \quad (3)$$

Step 4: Calculate all the messages obtained by the VNs as shown in (4).

$$L^l(q_i) = ln\frac{P_i(0)\prod_{j\in C(i)} r^l_{ji}(0)}{P_i(1)\prod_{j\in C(i)} r^l_{ji}(1)} = L(P_i) + \sum_{j\in C(i)} \ L^l\left(r_{ji}\right). \quad (4)$$

After obtaining the calculation result of $L^l(q_i)$, if the value of $L^l(q_i)$ is larger than 0, its corresponding estimate output codeword $\tilde{v}_i$ is determined to be 0, otherwise $\tilde{v}_i$ is determined to be 1 as shown in (5). In order to decide $\tilde{v}_i$, then the whole codeword $\tilde{v}$ obtained.

$$\tilde{v}_i = \begin{cases} 1, & if \quad L^l(q_i) \leq 0 \\ 0, & if \quad L^l(q_i) > 0 \end{cases}. \quad (5)$$

Step 5: Decoding decision: If $\tilde{v}H^T = 0$ or the iteration number reaches its limit, the decoding ceases; otherwise, the algorithm reverts to step 2.

Methods for organizing shared adaptive exponential min sum decoding neural network

In this section, we propose a shared neural adaptive exponential min sum (SNAEMS) decoding network, which is founded on the AEMS decoding method for LDPC codes and the principles of neural networks. We will explain how to unfold the AEMS decoding algorithm into a neural network and how to design the decoding network and select network parameters.

Neural networks work by using a multi-layer cascade model with nonlinear processing units that mimic the way neurons in the human brain work to make decisions, or recognize phenomena, weigh pros and cons and draw conclusions [10, 11]. A neural network comprises several layers of nodes, including an input layer, one or more hidden layers, and an output layer, each containing either the same or varying quantities of neurons. Each neuron in a different node layer is connected to a neuron in a neighboring node layer with associated weights. During training, the neural network adjusts these weights to improve its performance on a given task. Once the training is complete, the neural network model with suitable weights can be used to make predictions or recognize patterns. The check matrix H of an LDPC code specifies the quantity of CNs and VNs, along with the edges linking them. The message iterations between CNs and VNs correspond to the Tanner graph produced by the H matrix. The AEMS decoding algorithm can be unfolded based on the Tanner graph and the iteration process, and the resulting decoder is a neural network [12]. The computation process of CNs to VNs messages and VNs to CNs messages can be realized by hidden layers. In order to better distinguish the functions performed by the hidden layer, the hidden layer can be divided into two categories: the CN layer and the VN layer. Neurons in the CN layer compute messages from CNs to VNs and neurons in the VN layer compute messages from VNs to CNs. The number of operator units in each hidden layer of a neural network is proportional to the size of the Tanner graph as well [13]. Thus, T iterations of the AEMS decoding algorithm could be articulated as a deep neural network including $2l_{max+2}$ layers, which includes l_{max} CN layers, l_{max}VN layers, 1 input layer and 1 output layer. Based on the above principle, combined with the Tanner graph generated by the H matrix in Figure 1, the corresponding LDPC code decoding process can be unfolded into a neural network decoding structure. Figure 2 shows the neural network decoding structure with 3 complete iterations which has the same H matrix structure as Figure 1. Each additional iteration means that 1 CN layer and 1 VN layer are added to the hidden layer. The deep neural decoding network architecture for LDPC has 6 hidden layers, corresponding to 3 full iterations.

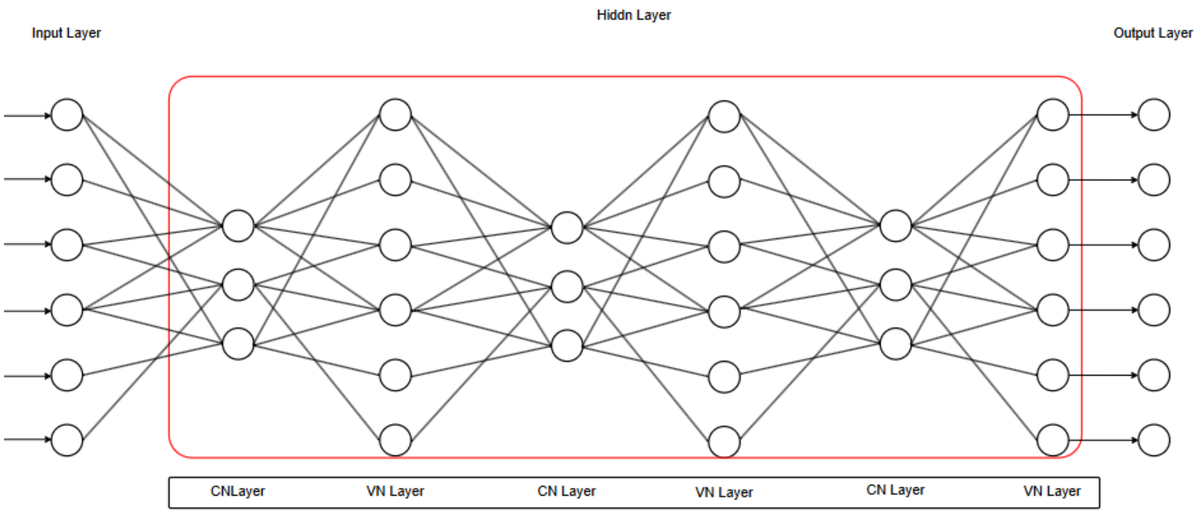

Figure 2 – The deep neural decoding network architecture for LDPC (6, 3)
Рисунок 2 – Архитектура сети глубокого нейронного декодирования для LDPC (6, 3)

According to the characteristics of the above neural network decoder, we expand the traditional AEMS decoder into an AEMS neural network decoder. We give the same weight factor to all layers except the input layer, so that the weight factors of all connecting edges are the same in each iteration. The same weight factor in each layer will not change the structure of the H matrix, which is more conducive to the AEMS neural network decoder to capture the topological relationship between the nodes of the entire LDPC code during forward propagation [4], and can greatly improve the training efficiency of the neural network decoder. The AEMS neural network decoder with shared weights in each of the above layers is referred to as the SNAEMS decoding network. Thus, the computation of CN to VN messages of the SNAEMS decoding network in the CN layer is shown in (6).

$$\begin{cases} L^l(r_{ji})_{SNAEMS} = \alpha^{l-1} \prod \; sgn(EI) EI_{m1}{}^{\lambda}, \; EI_{m2} \le 1 \\ L^l(r_{ji})_{SNAEMS} = \alpha^{l-1} \prod \; sgn(EI) EI_{m1}, \; EI_{m2} > 1 \end{cases}. \tag{6}$$

The computation of VN to CN messages of the SNAEMS decoding network in the VN layer is shown in (7).

$$L^l(q_{ij}) = L(P_i) + \beta^l \sum_{j' \in C(i) \setminus j} L^l(r_{j'i}). \tag{7}$$

The activation function can introduce nonlinearities to the neurons, allowing the neural network to approximate any nonlinear function, otherwise each layer of the neural network's output can only take up a linear transformation of the input function of the previous layer. For the SNAEMS decoding network, since the output v_i is 0 or 1 is a binary classification problem, we use sigmoid as the activation function, the *sigmoid* function shown in (8) is a powerful tool for dealing with binary classification problems, which will map any real value to the (0, 1) interval, indicating the probability of belonging to a certain category.

$$\sigma(x) = (1 + e^{-x})^{-1}. \quad (8)$$

Therefore, for the output layer, each output v_i of the SNAEMS decoding network is presented in (9). In (6), (7) and (9), where l represents the number of iterations, while l_{max} denotes the maximum number of iterations.

$$v_i = \sigma(L(P_i) + \sum_{j \in C(i)} \ (\beta^{l_{max}} L^{l_{max}} \left(r_{ji}\right))). \quad (9)$$

We employ the cross-entropy loss function for training the SNAEMS decoding network. The cross-entropy loss quantifies the discrepancy between the neural network output $\tilde{v}_{\ i}$ and the transmitted codeword $v_{\ i}$, and its definition is shown in (10), where $v_{\ i}$ and $\tilde{v}_{\ i}$ are the $i_{\ th}$transmitted information bit and SNAEMS decoding network estimate information bit. The loss function represents the difference between the predicted value and the ideal value. The smaller the difference, the closer the predicted value is to the ideal value, which indirectly indicates that the weight parameters of the neural network model are more reasonable.

$$L(v, \tilde{v}\ ) = -\frac{1}{N} \sum_{i=1}^{N} \ v_i log\ (v_i) + (1 - v_i) log\ (1 - \tilde{v}_i). \quad (10)$$

Simulation and comparison with others decoding algorithms

In this section, we compare the decoding performance of the SNAEMS decoding network introduced above with AEMS, MS, and BP algorithms for different LDPC codes. We use three short block channel codes recommended by Consultative Committee on Space Data Systems (CCSDS) 231.1-O-1 for TC synchronization, which are suitable for telecontrol commands and have good error correction capabilities and low decoding complexity [14]. In the simulation, we utilize an additive white Gaussian noise (AWGN) channel and transmit the coded bits using binary phase shift keying (BPSK) modulation. The signal-to-noise ratio (SNR) is established between 1.0 dB and 5.0 dB, with the number of frames fixed at 10,000 and the maximum iterations set at 10, after which the bit error ratio (BER) for various algorithms is computed. We use (64, 128), (128, 256) and (256, 512) CCSDS LDPC codes and build the SNAEMS decoding network with Python, because Python is currently the most powerful neural network programming language and has rich library and tool support, we use TensorFlow version 2.6.2.

Figures 3, 4 and 5 illustrate the BER performance of three CCSDS LDPC codes (64, 128), (128, 256) and (256, 512) under different decoding algorithms and different SNR conditions. It can be clearly seen from Figures 3, 4 and 5 that the SNAEMS decoding network has the best decoding performance among the three short block CCSDS LDPC codes, which is stronger than the traditional AEMS decoding algorithm. At the same time, the decoding performance of the traditional AEMS algorithm is better than the BP algorithm and the MS algorithm. From Figure 3 alone, it can be seen when the SNR exceeds 3.0 dB, the BER of the SNAEMS decoding network becomes inferior to that of the AEMS algorithm, and the SNAEMS decoding network has a coding gain of 0.1dB compared with the AEMS algorithm.

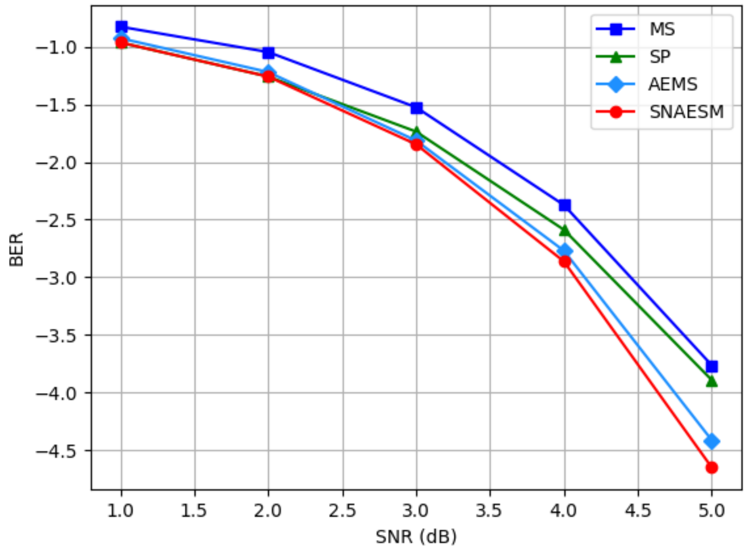

Figure 3 – BER of (64, 128) CCSDS with 10 iterations
Рисунок 3 – BER (64, 128) CCSDS с 10 итерациями

As can be seen from Figure 4, when the SNR exceeds 2.0 dB, the decoding performance of the SNAEMS decoding network starts to be stronger than the AEMS algorithm, and the SNAEMS decoding network has a coding gain of 0.1 dB over the AEMS algorithm. In Figure 5, the decoding performance advantage of the SNAEMS decoding network is more obvious, when the SNR exceeds 2.0 dB, the BER of the SNAEMS decoding network starts to be significantly lower than that of the AEMS algorithm, and the SNAEMS decoding network and the AEMS decoding algorithm finish decoding first when the SNR exceeds 3.0dB. At this time, the other decoding algorithms have not finished decoding, and the SNAEMS decoding network has a decoding gain of 0.15 dB over the AEMS algorithm. Combining Figures 3, 4 and 5, it can be seen that with the increase of the CCSDS LDPC code length, the decoding ability of the SNAEMS decoding network is gradually enhanced, and the decoding gain is gradually increased compared to that of the AEMS algorithm. The above results prove that the AEMS algorithm optimized by neural network has stronger decoding performance and its decoding performance increases with the increase of code length.

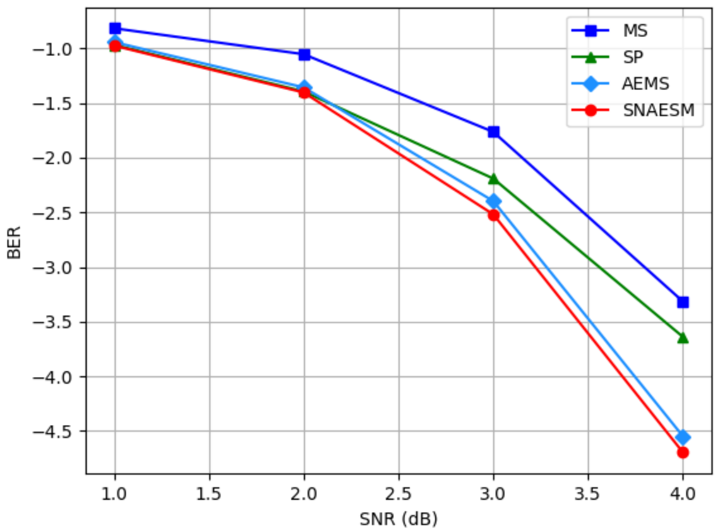

Figure 4 – BER of (128, 256) CCSDS with 10 iterations
Рисунок 4 – BER в (128, 256) CCSDS с 10 итерациями

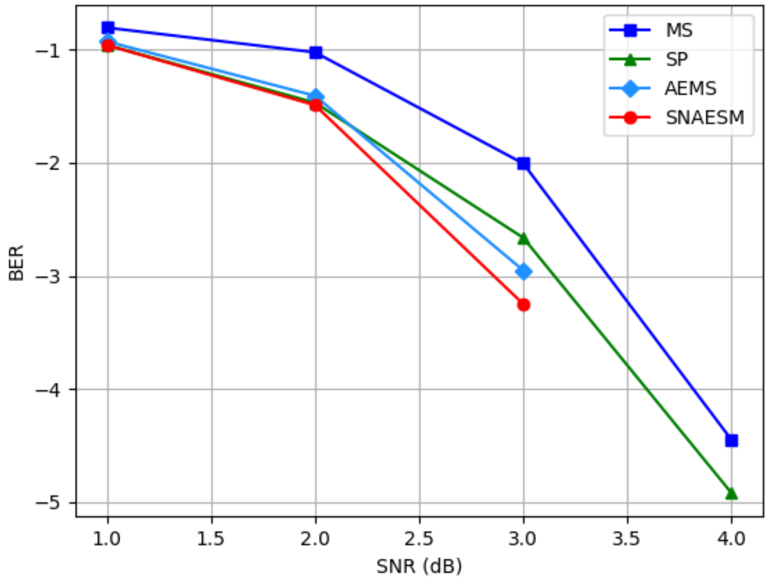

Figure 5 – BER of (256, 512) CCSDS with 10 iterations
Рисунок 5 – BER в (256, 512) CCSDS с 10 итерациями

In SNAEMS decoding network, the messages exchanged during each iteration between CNs and VNs are multiplied by distinct correction factors α and β, thereby incorporating weight parameters into the edges of the Tanner graph. These weights are obtained through deep learning neural network training, and they partially compensate for the deleterious effects of small cycles in the Tanner graph. Such "neural decoders" outperform traditional decoders because they learn to use weights during decoding iterations to enhance performance. The application of SNAEMS decoding algorithm is divided into two phases, in the training phase, the SNAEMS decoding network will consume a lot of computational resources and time to obtain suitable weights α and β. In the usage phase, compared to having a traditional AEMS decoder, the SNAEMS decoding network, requires more memory to store the corresponding weights α and β, and with the increase of code length and number of iterations, the stored values will increase proportionally with the code length and the number of iterations, and the SNAEMS decoding network performs one more multiplication operation for each iteration operation. Therefore, the SNAEMS decoding network is suitable for communication scenarios that require extremely high decoding accuracy and can provide better hardware resources.

Conclusion

This paper presents the SNAEMS LDPC code decoding network, which transforms the AEMS decoding algorithm into a neural network architecture by adhering to the Tanner graph of the LDPC code and the algorithm's iterative sequence. We reduce the computational complexity of the SNAEMS decoding network by sharing the weight factors of each layer. Simulation results show that the decoding performance of the SNAEMS decoding network with appropriate weight factors obtained through training is superior to that of the conventional AEMS decoder compared to the traditional AEMS decoder, and its decoding advantage increases with the increase of code length. Neural network is a powerful tool to optimize the LDPC decoding algorithm, but it inevitably increases the consumption of computational and storage resources, while using neural network to improve the performance of the decoding algorithm, attention should be paid to the balance between the performance enhancement and the additional consumption of hardware resources.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Mathew A., Amudha P., Sivakumari S. Deep Learning Techniques: An Overview. In: *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020, 13–15 February 2020, Jaipur, India.* Singapore Springer; 2021. pp. 599–608. https://doi.org/10.1007/978-981-15-3383-9_54
2. Li L., Mu X., Li S., Peng H. A Review of Face Recognition Technology. *IEEE Access.* 2020;8:139110–139120. https://doi.org/10.1109/ACCESS.2020.3011028
3. Gallager R. Low-density parity-check codes. *IRE Transactions on Information Theory.* 1962;8(1):21–28. https://doi.org/10.1109/TIT.1962.1057683
4. Wang Qi., Wang S., Fang H., Chen L., Chen L., Guo Yu. A Model-Driven Deep Learning Method for Normalized Min-Sum LDPC Decoding. In: *2020 IEEE International Conference on Communications Workshops (ICC Workshops), 07–11 June 2020, Dublin, Ireland.* IEEE; 2020. pp. 1–6. https://doi.org/10.1109/ICCWorkshops49005.2020.9145237
5. Nachmani E., Marciano E., Lugosch L., Gross W.J., Burshtein D., Be'ery Ya. Deep Learning Methods for Improved Decoding of Linear Codes. *IEEE Journal of Selected Topics in Signal Processing.* 2018;12(1):119–131. https://doi.org/10.1109/JSTSP.2017.2788405
6. Lugosch L., Gross W.J. Neural offset min-sum decoding. In: *2017 IEEE International Symposium on Information Theory (ISIT), 25–30 June 2017, Aachen, Germany.* IEEE; 2017. pp. 1361–1365. https://doi.org/10.1109/ISIT.2017.8006751
7. Fossorier M.P.C., Mihaljevic M., Imai H. Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. *IEEE Transactions on Communications.* 1999;47(5):673–680. https://doi.org/10.1109/26.768759
8. Чжан В., Мухамад И., Саклаков В.М. Разработка адаптивного экспоненциального алгоритма декодирования минимальной суммы. *Моделирование, оптимизация и информационные технологии.* 2024;12(4). (На англ.). https://doi.org/10.26102/2310-6018/2024.47.4.019

   Zhang W., Mouhamad I., Saklakov V.M. Development of adaptive exponential min sum decoding algorithm. *Modeling, Optimization and Information Technology.* 2024;12(4). https://doi.org/10.26102/2310-6018/2024.47.4.019
9. Tanner R. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory.* 1981;27(5):533–547. https://doi.org/10.1109/TIT.1981.1056404
10. Lyu W., Zhang Z., Jiao Ch., Qin K., Zhang H. Performance Evaluation of Channel Decoding with Deep Neural Networks. In: *2018 IEEE International Conference on Communications (ICC), 20–24 May 2018, Kansas City, USA.* IEEE; 2018. pp. 1–6. https://doi.org/10.1109/ICC.2018.8422289
11. Goodfellow Ia., Bengio Yo., Courville A. *Deep Learning.* Cambridge: MIT Press; 2016. 800 p.
12. Nachmani E., Be'ery Ya., Burshtein D. Learning to decode linear codes using deep learning. In: *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 27–30 September 2016, Monticello, USA.* IEEE; 2017. pp. 341–346. https://doi.org/10.1109/ALLERTON.2016.7852251
13. Wu X., Jiang M., Zhao Ch. Decoding Optimization for 5G LDPC Codes by Machine Learning. *IEEE Access.* 2018;6:50179–50186. https://doi.org/10.1109/ACCESS.2018.2869374
14. CCSDS Historical Document "Short Block Length LDPC Codes for TC Synchronization and Channel Coding" CCSDS 231.1-O-1 (2015). URL: https://public.ccsds.org/Pubs/231x1o1s.pdf [Accessed 12th December 2024].

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Чжан Вэйцзя, аспирант, ассистент, Национальный исследовательский Томский политехнический университет, Томск, Российская Федерация.
e-mail: victoryzh@tpu.ru
ORCID: 0000-0003-2252-2750

Zhang Weijia, Postgraduate Student, Assistant Lecturer, National Research Tomsk Polytechnic University, Tomsk, the Russian Federation.

Ибрагим Мухамад, аспирант, ассистент, Национальный исследовательский Томский политехнический университет, Томск, Российская Федерация.
e-mail: ibragim1@tpu.ru
ORCID: 0000-0003-1569-5493

Ibrahem Mouhamad, Postgraduate Student, Assistant Lecturer, National Research Tomsk Polytechnic University, Tomsk, the Russian Federation.

Саклаков Василий Михайлович, старший преподаватель, Отделение информационных технологий, Национальный исследовательский Томский политехнический университет, Томск, Российская Федерация.
e-mail: saklavas@tpu.ru
ORCID: 0000-0003-1716-4581

Vasiliy M. Saklakov, Senior Lecturer in Division for Information Technology, National Research Tomsk Polytechnic University, Tomsk, the Russian Federation.

Душанта Налин К. Джаякоди, Ph.D., профессор, Национальный исследовательский Томский политехнический университет, Томск, Российская Федерация.
e-mail: nalin@tpu.ru
ORCID: 0000-0002-7004-2930

Dushantha Nalin K. Jayakody, Ph.D., Professor, National Research Tomsk Polytechnic University, Tomsk, the Russian Federation.

Статья поступила в редакцию 24.01.2025; одобрена после рецензирования 18.02.2025; принята к публикации 27.02.2025.

The article was submitted 24.01.2025; approved after reviewing 18.02.2025; accepted for publication 27.02.2025.