Scientific Research Publishing

Journal of Computer and Communications, 2020, 8, 90-99
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Improved Polar Decoder Utilizing Neural Network in Fast Simplified Successive-Cancellation Decoding

Jiaxin Fang, Chunwu Liu

College of Artificial Intelligence, National University of Defense Technology, Changsha, China
Email: 865560626@qq.com

How to cite this paper: Fang, J.X. and Liu, C.W. (2020) Improved Polar Decoder Utilizing Neural Network in Fast Simplified Successive-Cancellation Decoding. *Journal of Computer and Communications*, **8**, 90-99.
https://doi.org/10.4236/jcc.2020.87008

Received: June 5, 2020
Accepted: July 28, 2020
Published: July 31, 2020

Abstract

Polar codes using successive-cancellation decoding always suffer from high latency for its serial nature. Fast simplified successive-cancellation decoding algorithm improves the situation in theoretically but not performs well as expected in practical for the workload of nodes identification and the existence of many short blocks. Meanwhile, Neural network (NN) based decoders have appeared as potential candidates to replace conventional decoders for polar codes. But the exponentially increasing training complexity with information bits is unacceptable which means it is only suitable for short codes. In this paper, we present an improvement that increases decoding efficiency without degrading the error-correction performance. The long polar codes are divided into several sub-blocks, some of which can be decoded adopting fast maximum likelihood decoding method and the remained parts are replaced by several short codes NN decoders. The result shows that time steps the proposed algorithm need only equal to 79.8% of fast simplified successive-cancellation decoders require. Moreover, it has up to 21.2 times faster than successive-cancellation decoding algorithm. More importantly, the proposed algorithm decreases the hardness when applying in some degree.

Keywords

Polar Codes, Decoding Latency, Fast Simplified Successive-Cancellation Decoding (Fast-SSC), Neural Network (NN)

1. Introduction

Polar code was proved to be first class capacity-achieving codes of symmetric binary-input memoryless channels using SC decoding algorithm when the code

length goes to infinity by Arıkan [1]. However, two problems hinder the adoption of polar codes: the error-correction performance is not reason-able for finite code length; and they suffer from high latency as well as limited throughput due to the serial nature of the SC decoding algorithm. New algorithms are proposed to improve the error-correction performance of SC decoding algorithm such as successive cancellation list (SCL) decoding algorithm [2] [3] [4] [5] [6], successive cancellation stack (SCS) decoding algorithm. But these algorithms realize better performance at the cost of higher computational complexity and lower throughput. In this work, we focus on the latency issue, which is exacerbated by the requirement of long codes since the parallel algorithms such as BP decoding achieve lower latency while introducing an unacceptable computational complexity [7].

To reduce the decoding latency without having a bad influence on the error-correction performance, simplified successive-cancellation (SSC) decoding was proposed to take advantage of rate-zero and rate-one nodes. The latency of SC decoding varied almost between two and twenty times that of SSC decoding depending on code length [8]. Further, Gabi Sarkis present a fast simplified successive-cancellation decoder (fast-SSC) that improves the decoding speed by utilizing single-parity-check (SPC) node, repetition (REP) node [9]. However, the theoretical advantages of fast simplified successive-cancellation decoder can only turn into reality when the corresponding sub-blocks are worth to be solved this way. In practical, heavy burden faced by nodes identification and presence of massive short blocks for long codes restrict the use of fast simplified successive-cancellation decoder. As a result of this, our main work is trying to solve these difficulties to enhance the decoding efficiency. Later, a new approach using NN (neural network) has been proposed to realize to decode polar codes with MAP performance for small block lengths in [10], but the NN decoder's training complexity increases exponentially with information bits in the code words limits its use. Inspired by this, NN decoding may be helpful to improve fast-SSC decoder, because NN decoder estimates the bits by passing each layer only once which promises low-latency implementations and it can avoid too short blocks according to our design.

2. Fast Simplified Successive-Cancellation Decoding

2.1. The Fast-SSC Decoder

A fast-SSC decoder graph is built by transforming a non-bit reversed polar code graph into a binary tree of five node types: rate-0 nodes, rate-1nodes, REP nodes, SPC nodes and rate-R nodes, denoted $\mathcal{N}^0, \mathcal{N}^1, \mathcal{N}^{REP}, \mathcal{N}^{SPC}$ and $\mathcal{N}^R$ respectively. A polar code consist of frozen bits are $\mathcal{N}^0$ nodes, non-frozen bits are $\mathcal{N}^1$ nodes. Only $\boldsymbol{u}_N$ is information bits are $\mathcal{N}^{REP}$ nodes and the $\mathcal{N}^{SPC}$ nodes referring to only $\boldsymbol{u}_1$ is frozen bits. Finally, a node whose descendants contain different types mentioned before is an $\mathcal{N}^R$ node.

A polar code can be represented by a binary code tree in which every node

represents a codeword. **Figure 1** depicts a length-32 polar code using binary-tree expression. The white leaf nodes represent the frozen, black ones represent the information bits and grey ones represent nodes containing both frozen bits and information bits.

It illustrates the SC decoding binary tree of polar code $\mathcal{P}(32,16)$. For a node of length N_v, the estimated hard values pass $\beta=\{\beta_0,\beta_1,...,\beta_{N_V-1}\}$ from its child nodes to the parent node, while the channel log-likelihood ratios (LLR) values $\alpha=\{\alpha_0,\alpha_1,\ldots,\alpha_{N_V-1}\}$ pass through the opponent direction. Once α is available from the parent, the left child node $\alpha^l=\{\alpha_0^l,\alpha_1^l,\ldots,\alpha_{N_v/2-1}^l\}$ and the right child node $\alpha^r=\{\alpha_0^r,\alpha_1^r,\ldots,\alpha_{N_v/2-1}^r\}$ are calculated using the min-sum (MS) approximation [11] to simplify calculation as:

$$\alpha_i^l=sgn(\alpha_i)sgn(\alpha_{i+N_v/2})min(|\alpha_i|,|\alpha_{i+N_v/2}|). \tag{1}$$

$$\alpha_i^r=\alpha_{i+N_v/2}+(1-2\beta_i^l)\alpha_i. \tag{2}$$

in which the β^l are got through α^l. Then after the β^r have been calculated, the β values can be passed as follows:

$$\beta_i=\begin{cases}\beta_i^l\oplus\beta_i^r, & \text{if } i<N_v/2\\ \beta_{i-N_v/2}^r, & \text{otherwise}\end{cases} \tag{3}$$

where the $\oplus$ is xor operation. Finally, the estimated message bits $\hat{u}$ are determined as

$$\hat{u}_i=\begin{cases}0, & \text{if } i\in\mathcal{F} \text{ or } \alpha_i\geq 0\\ 1, & \text{otherwise}\end{cases} \tag{4}$$

In SC decoding process exists quite a large number of redundant operations which results in a high latency of 2*N*-2. Besides, the estimated hard values of some bits do not depend on LLR, so it is unnecessary to calculate all the LLR.

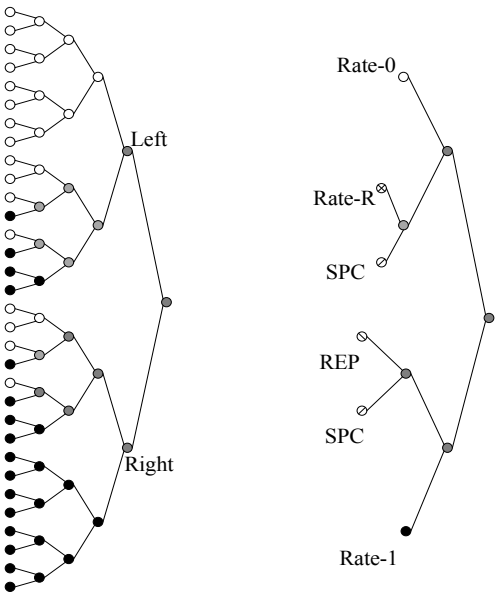

Figure 1. Decoder trees corresponding to the decoding algorithms.

SSC decoding algorithms are proposed to avoid the redundant operations and high latency in SC decoding but the effect is limited. Further, fast-SSC decoding is proposed and there is a fast maximum likelihood decoding method for the $\mathcal{N}^0, \mathcal{N}^1, \mathcal{N}^{REP}, \mathcal{N}^{SPC}$ mentioned before in the binary tree so that the corresponding decoding can be completed in one step for each node.

2.2. The Study of Fast-SSC Decoder

Although the four-types nodes can simplify decoding stage to realize lower latency, their expectation can only be fully realized when the corresponding codeword length reaches a certain length. In fact, it cannot perform quite better than SC decoding method when the sub-blocks are short while increase the workload of node type recognition in practical. So, in our work, the distribution of information bits and frozen bits of polar codes is studied to select the appropriate sub-block size.

The basic idea of polar codes is to construct a coding system in which the information bits are transmitted in higher reliability channels, while other channels with lower reliability are used to carry frozen bits (usually set to be 0). Because the Bhattacharyya parameter cannot accurately describe the actual channel situation. Many methods such as density evolution (DE), Gaussian approximation (GA) and polarization weigh (PW) have been proposed to assess channel reliability accurately. The GA-based channel reliability evaluation is related to the channel, while the reliability order relationship of these channels can be determined in a fixed way regardless of channel conditions. In fact, the code length corresponding to $\mathcal{N}^0, \mathcal{N}^1, \mathcal{N}^{REP}, \mathcal{N}^{SPC}$ is not small in a long polar code.

Here we count the number of four types nodes and the corresponding length of a 16,384-length polar code with code rate 0.2, 0.5, 0.8 which represents different level of code rate using GA to construct the code. The parameter noise variance σ^2 is set to 0.81 and the result is shown in **Table 1**.

Similar situation appears when count other code length. In fact, for a long polar code, there are many nodes belonging to the four types nodes. When the code rate is high or low, there exists few but very long $\mathcal{N}^0, \mathcal{N}^1$ constituent codes. Besides, the long polar codes also have a large number of $\mathcal{N}^{REP}, \mathcal{N}^{SPC}$ constituent codes especially when the code rate is moderate. Therefore, it can greatly simplify the decoding process and reduce the decoding latency adopting the fast-SSC. However, we can also see many nodes belonging to these types only with code length of 2 or 4. For these parts, fast-SSC has limited simplification capabilities and heavy workload of identifying nodes. Finally, the codewords are only divide into a large amount of parts at least 8-length. After we take block length into account, finding a method to process these rate-R nodes become our main work. Here we only analyzed GA algorithm because it's relatively easy to describe, but analysis about other algorithms for polar codes construction leads to the same or extremely approximate conclusion.

Table 1. Number of four type nodes of different sizes in three polar codes of length 16384 and rates 0.8 0.5, and 0.2.

Code	Code Rate	rate-0, $N_v \in$			
		(0, 4]	(4, 64]	(64, 4096]	(4096, 16,384]
(16,384, 13,107)	0.8	173	0	0	0
(16,384, 8192)	0.5	0	0	0	0
(16,384, 3277)	0.2	0	0	0	1
		rate-1, $N_v \in$			
		(0, 4]	(4, 64]	(64, 2048]	(2048, 16,384]
(16,384, 13,107)	0.8	56	132	4	1
(16,384, 8192)	0.5	0	3	0	0
(16,384, 3277)	0.2	0	0	0	0
		SPC, $N_v \in$			
		(0, 4]	(4, 64]	(64, 1024]	(1024, 16,384]
(16,384, 13,107)	0.8	610	203	0	0
(16,384, 8192)	0.5	1231	275	0	0
(16,384, 3277)	0.2	464	54	0	0
		REP, $N_v \in$			
		(0, 4]	(4, 64]	(64, 1024]	(1024, 16,384]
(16,384, 13,107)	0.8	885	70	0	0
(16,384, 8192)	0.5	2088	286	0	0
(16,384, 3277)	0.2	1008	273	0	0

3. NN-fSSC Decoder for Codes

It shows that neural networks can be used for decoding and approximate MAP performance with enough training times in [5]. Compared with other decoding algorithms, neural network decoding does not require iteration and achieve low latency for its high parallelizable structure, but the training complexity limits neural networks only suitable for short codes. Different from fast-SSC decoders that are designed to decode special constituent codes, the NN decoder here is trained to decode any node without considering the locations of frozen bit and information bit. Obviously, neural network just rightly be used to as an alternative to decode rate-R nodes, which can not only reduce the workload of node recognition, but also lower the decoding delay. In this paper we use N to denote code length, R to denote code rate, K to denote information length, $\mathcal{A}$ to denote free set, and $\overline{\mathcal{A}}$ to denote frozen set. The information bit is 0 or 1 and the frozen bit is 0.

The source sequence $u_1^N = (u_1, u_2, \ldots, u_N)$ are encoded to codeword $x_1^N = (x_1, x_2, \ldots, x_N)$ with $\mathcal{A}$ to place the K information bits and $\overline{\mathcal{A}}$ to place

the frozen bits. We use y_1^N to denote the output of noisy channel after modulation and the input of polar decoder. The output of the decoder is denoted by m_1^K. Then we should get the decoding results m_1^K to approximate to the original K information bits using our NN decoder, then y and m become training data in this machine learning problem.

As shown above, the NN decoding problem is a machine learning problem. To make the trained neural network be fit for the test data when decoding, a large amount of data must be necessary. Here, we can generate enough training data labeled. Under this condition, we can train on unlimited training set by simply increasing the number of epochs M_{ep}. The process of collecting training data for each NN decoder is shown in **Figure 2**. As shown in this figure, we need to add two additional layers without trainable parameters into the NN before decoding layers, one for modulation, the other for adding noise randomly. In this paper, binary phase shift keying (BPSK) modulation and an additive white Gaussian noise (AWGN) channel are used for simplifying problem. It shows the system's work process. We use the whole NN when training, but use only decoding layers when decoding.

A fully connected NN decoder contains M hidden layers with the size of $\{L_1, L_2, \ldots, L_t\}$, where $L_t > 0$ and $1 \le t \le M$. The size of the input of the network is equal to the sub-block length, here is 8. The size of output layer is set to the number of output bits.

The full network structure can be expressed as

$$\{L_0, L_1, \cdots, L_M, L_{M+1}\} = \{8, 16K, 8K, K\},$$

where L_0 and L_{M+1} denote the size of the input and output layers, respectively. In order to make the training process more efficient and avoid neurons invalidity during training process, we set the activation function to be Leaky ReLU for $0 < m \le M$, and the sigmoid function for $m = M + 1$. The sigmoid function restricts the range of the decoder's output values to [0, 1].

$$Leaky\ ReLU = \max(0.01x, x) \tag{5}$$

$$Sigmoid = \frac{1}{1+e^{-x}} \tag{6}$$

To evaluate the decoding performance, the binary-cross-entropy (BCE) functions

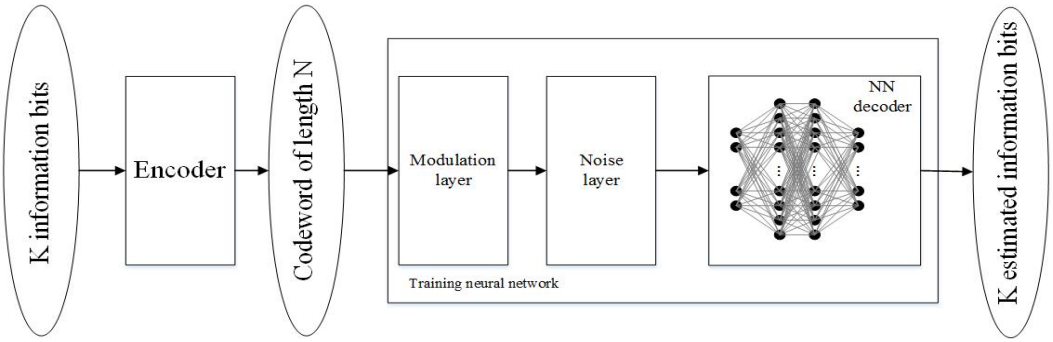

Figure 2. Deep learning setup for channel coding.

is adopted to express the expected loss of the neural network as

$$\mathcal{L}_{BCE} = -\frac{1}{k}\sum_i \left[a_i \ln\left(\hat{a}_i\right) + \left(1-a_i\right)\ln\left(1-\hat{a}_i\right)\right] \tag{7}$$

where $a_i \in \{0,1\}$ is the ith information bit and $\hat{a}_i \in \{0,1\}$ is the estimation of NN.

Simulations result in **Figure 3** shows that NN decoder trained has similar BER performance with these conventional decoders as long as the epochs is large.

For a long polar code, all the node types in a binary tree can be divide into two parts, type-nodes denoted $\mathcal{N}^0, \mathcal{N}^1, \mathcal{N}^{REP}, \mathcal{N}^{SPC}$ and NN-nodes denoted $\mathcal{N}^R$. In NN-fSSC decoder, the type-nodes using fast maximum likelihood decoding method and the NN-nodes using NN decoding. After the training stage, NN decoder obtains its weight and bias matrices used to decode $\mathcal{N}^R$.

Since we care about the decoding latency here, we assume that hardware resources are met, which means the minimum time steps to approximately measure the efficiency of decoders. The number of time steps required to finish the decoding process in the NN-node is dependent on the number of hidden layers and can be calculated as $T_{NN} = M + 1$, here T_{NN} is 3. As type-node only require one time step, so in NN-fSSC decoder, each type-node or NN-node can reduce different time steps which is corresponding to their length. However, each NN-node need three steps to decode which means two extra steps introduced. On this basis, the decoding latency in terms of the number of time steps for the proposed NN-fSSC decoder can be approximately calculated as

$$T_{NN\text{-}fSSC} = \frac{3N}{2} - 2 - \left(\frac{3\sum N_{part}}{2} - 2n_{parts}\right) + 2n_{NN}. \tag{8}$$

where N_{part} is the length of type-node or NN-node, n_{NN} is the number of NN-nodes.

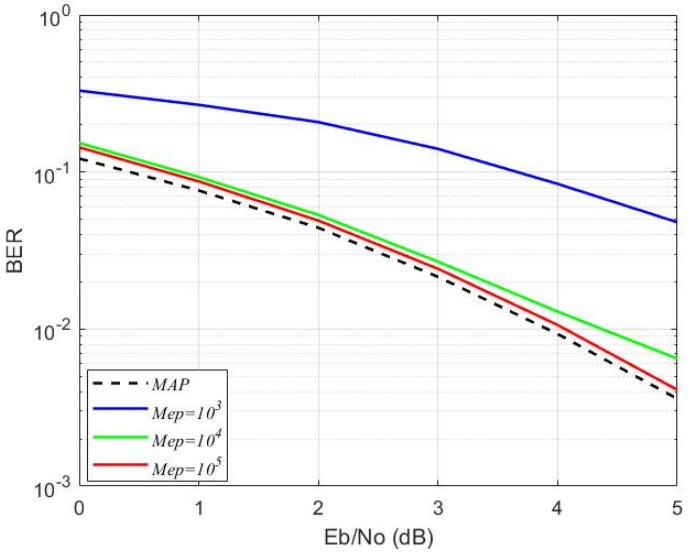

Figure 3. Performance of NN decoders.

5. Simulation Results and Analysis

5.1. Performance of NN-fSSC Decoder

Although the aim of NN-fSSC is to decrease decoding latency, this cannot at the cost of error-correction performance degeneration.

Figure 4 illustrates the Simulations of a (2048, 1024) and a (16,384, 13,107) polar codes when transmitting random codewords over the additive white Gaussian noise (AWGN) channel using binary phase shift keying (BPSK). In this figure, fast-SSC and NN-fSSC decoding has a negligible effect on error-correction performance which is affected slightly by code rate.

5.2. Efficiency of NN-fSSC Decoder

In order to study the extent of improvement which can be achieved in decoding delay, two significant factors (code rate R and codelength N) affecting the decoding speed of fast-SSC and NN-fSSC are observed. We compare the proposed algorithm with some current algorithms, like SC and fast-SSC.

Overall, NN-fSSC decoder perform better than SC and fast-SSC decoder, where it can be observed that NN-fSSC is 4.7 to 21.2 times faster than SC, and that the latency of NN-fSSC is approximately 79.8% of the fast-SSC. Besides, the change of time steps shows a clear relationship with code rate in the decoding process. **Figure 5** illustrates this trend for a polar code of length 2048 and 16,384, when the code rate is moderate, the introduction of NN can bring greater progress as there exists more short length nodes in moderate rate codes. High code rate decreases slightly more than low code rate.

Finally, to study the influence of code length N on the decoding time latency, the information throughput of nine polar codes of lengths varying from 2^{10} to 2^{14} is shown in **Figure 6** for code rates 0.2, 0.5, and 0.8.

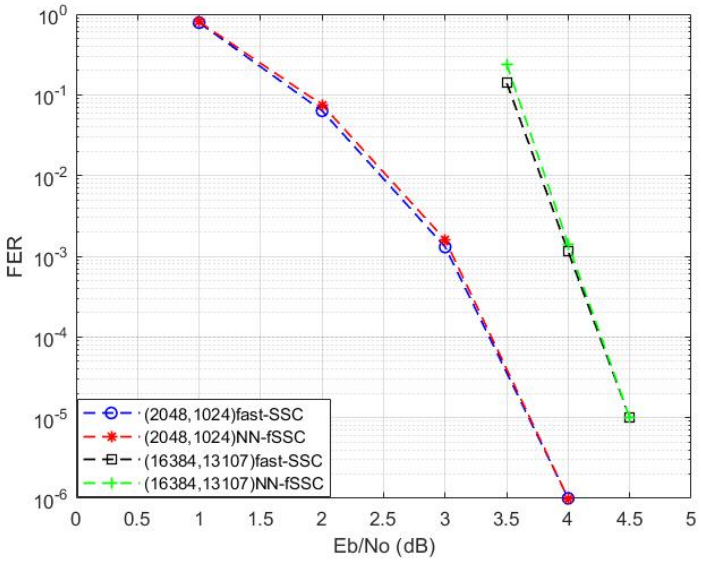

Figure 4. FER of fast-SSC and NN-fSSC.

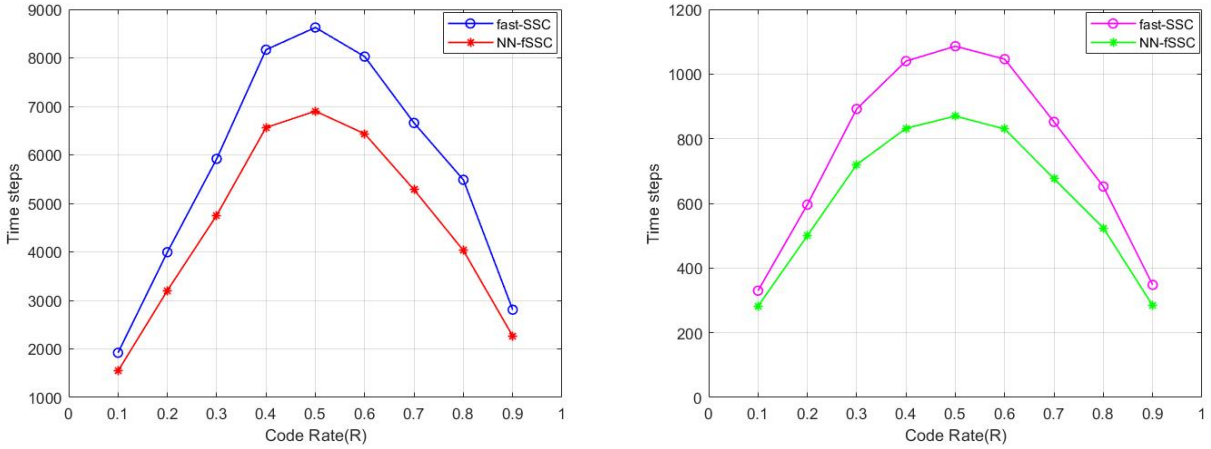

Figure 5. Time steps of the fast-SSC, NN-fSSC decoders for a code with different code length.

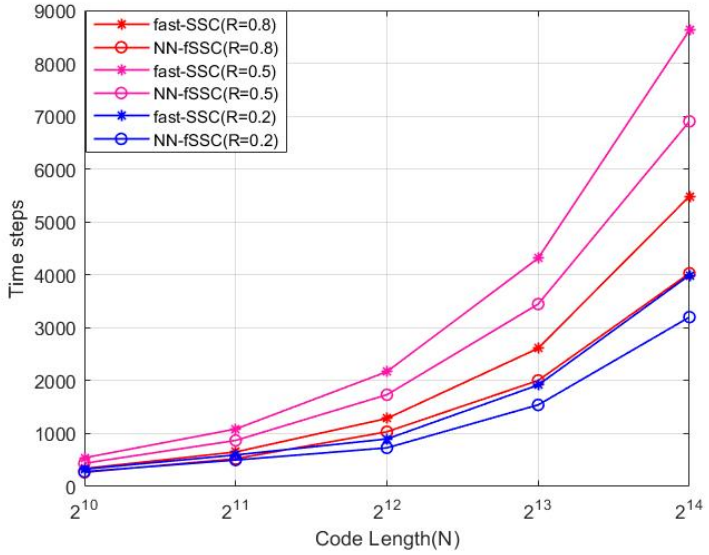

Figure 6. Time steps of the fast-SSC, NN-fSSC decoders for different codes of rates.

It shows that NN-fSSC decoder appears greater advantage with code length increasing when the code rate is same. So, the NN-fSSC decoder is more suitable for codeword which has a certain length.

6. Conclusions

In this paper, firstly we studied the principle of fast-SSC decoders and find that conventional fast-SSC decoders cannot perform well in practical, because lots of short sub-blocks require much work in nodes identifying. Then we divide fast-SSC decoders into groups with certain length to lower the difficult of identifying in some degree. Secondly, we train NN decoders for short polar codes. It rightly can be used to decode the sub-blocks which are not suitable for fast-SSC decoders. Finally, we propose NN-fSSC decoders combined by fast-SSC decoders and NN decoders for long polar codes. According to our simulation, NN-fSSC

decoders increase the decoding efficiency largely and make nodes identification easier with acceptable performance, which means this paper's main purpose accomplished.

In this paper, we mainly concern the decoding latency, but as for our future work, combination of NN and fast-SCL may be a good choice for improving performance.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Arikan, E. (2009) Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels. *IEEE Transactions on Information Theory*, **55**, 3051-3073. https://doi.org/10.1109/TIT.2009.2021379

[2] Balatsoukas-Stimming, A., Bastani Parizi, M. and Burg, A. (2015) LLR-Based Successive Cancellation List Decoding of Polar Codes. *IEEE Transactions on Signal Processing*, **63**, 5165-5179. https://doi.org/10.1109/TSP.2015.2439211

[3] Balatsoukas-Stimming, A., Raymond, A.J., Gross, W.J. and Burg, A. (2014) Hardware Architecture for List Successive Cancellation Decoding of Polar Codes. *IEEE Transactions on Circuits & Systems II Express Briefs,* **61**, 609-613. https://doi.org/10.1109/TCSII.2014.2327336

[4] Hashemi, S.A., Condo, C. and Gross, W.J. (2017) Fast Simplified Successive-Cancellation List Decoding of Polar Codes.

[5] Niu, K., Lin, J.R. and Chen, K. (2012) List Successive Cancellation Decoding of Polar Codes. *Electronics Letters*, **48**, 500-501. https://doi.org/10.1049/el.2011.3334

[6] Sarkis, G., Giard, P., Vardy, A., Thibeault, C. and Gross, W.J. (2015) Fast List Decoders for Polar Codes. *IEEE Journal on Selected Areas in Communications,* **34**, 318-328. https://doi.org/10.1109/JSAC.2015.2504299

[7] Arkan, E. (2008) A Performance Comparison of Polar Codes and Reed-Muller Codes. *Communications Letters IEEE,* **12**, 447-449. https://doi.org/10.1109/LCOMM.2008.080017

[8] Alamdar-Yazdi, A. and Kschischang, F.R. (2011) A Simplified Successive-Cancellation Decoder for Polar Codes. *IEEE Communications Letters,* **15**, 1378-1380. https://doi.org/10.1109/LCOMM.2011.101811.111480

[9] Sarkis, G., Giard, P., Vardy, A., Thibeault, C. and Gross, W.J. (2014) Fast Polar Decoders: Algorithm and Implementation. *IEEE Journal on Selected Areas in Communications,* **32**, 946-957. https://doi.org/10.1109/JSAC.2014.140514

[10] Gruber, T., Cammerer, S., Hoydis, J. and ten Brink, S. (2017) On Deep Learning-Based Channel Decoding (2017 51st Annual Conference on Information Sciences and Systems). IEEE, New York. (In English) https://doi.org/10.1109/CISS.2017.7926071

[11] Leroux, C., Raymond, A.J., Sarkis, G. and Gross, W.J. (2013) A Semi-Parallel Successive-Cancellation Decoder for Polar Codes. *IEEE Transactions on Signal Processing,* **61**, 289-299. https://doi.org/10.1109/TSP.2012.2223693