Abstract—In the decoding of linear block codes, it was shown that noticeable gains in terms of bit error rate can be achieved by introducing learnable parameters to the Belief Propagation (BP) decoder. Despite the success of these methods, there are two key open problems. The first is the lack of analysis for channels other than AWGN. The second is the interpretation of the weights learned and their effect on the reliability of the BP decoder. In this work, we aim to bridge this gap by looking at non-AWGN channels such as Extended Typical Urban (ETU) channel. We study the effect of entangling the weights and how the performance holds across different channel settings for the min-sum version of BP decoder. We show that while entanglement has little degradation in the AWGN channel, a significant loss is observed in more complex channels. We also provide insights into the weights learned and their connection to the structure of the underlying code. Finally, we evaluate our algorithm on the over-the-air channels using Software Defined Radios.

I. INTRODUCTION

Short length block codes are an attractive choice for use cases with stringent latency requirements [1]. However, they suffer from poor error correction performance. One key reason for this is the presence of short cycles in the Tanner graph of the code, making their decoding sub-optimal. Unfortunately at such short lengths, codes designed without cycles have worse error correction performance [2]. Additionally, when fading channels are considered, the Log Likelihood Ratio (LLR) computation suffers from errors because of imperfect Channel State Information (CSI) and equalization, which makes the Belief Propagation (BP) decoder sub-optimal [3]. Hence modifications to the traditional BP decoder are necessary to improve the performance.

In [4], the authors demonstrate the advantage of using neural networks for improving the BP decoder by placing multiplicative weights along the edges of the Tanner graph structure and unrolling the iterations, resulting in a neural BP decoder. Since then, there have been many works that explored the possibility of reducing the complexity of the neural BP decoders. In [5], the authors show that instead of using multiplicative weights, comparable gains can be achieved by using offset factors combined with min-sum approximation, which reduces the implementation cost. Later in [6], the authors explore the possibility of reducing the number of weights needed by entangling them across iterations in a recurrent fashion and show that the performance is still comparable to the fully parameterized model. More recently, the authors in [7] propose entangling the weights not only across iterations but also across the edges. To compensate for the performance loss, a shallow neural network referred to as Parameter Adapter Network (PAN), is used to select different weights for different SNRs regions resulting in a performance very close to [6].

Despite the success of these works, there are two key open problems that we answer in this work. The first is the trade-offs associated with entanglement, which is limited to only AWGN channels in [4]–[7]. We consider non-AWGN channels such as ETU and show that the trade-offs are quite different from AWGN. The second is the interpretation of the learned weights. While it has been conjectured that the use of normalization or offsets to modify the beliefs helps mitigate the detrimental effects of short cycles [4], [5], there is no supporting evidence. We present empirical evidence that the weights learned are directly related to the short cycles present in the code and investigate how they help in improving the reliability of the posterior LLRs.

Our main contributions can be summarized are as follows:

- **Complexity Trade-off**: We explore the complexity vs performance trade-off for neural augmented min-sum decoders across various channels. We show that we benefit significantly by having more number of weights for a complex channel while fewer weights are sufficient for a simple channel such as AWGN and thus the choice of parameters can be made based on the channel conditions.

- **Robustness and adaptivity**: We show that the learned weights are fairly robust to channel variations i.e, the decoder trained on one channel still outperforms the classical decoders on other channels. We further show that these neural decoders can be quickly adapted to new channels with minimal additional training, to achieve additional gains.

- **Over-the-air channels**: We evaluate neural min-sum decoders for over-the-air channels using USRPs. We show that that the fully parameterized min-sum decoder outperforms the entangled variant.

- **Interpretation**: We provide insights into the learned decoder by establishing a connection between the number of cycles in the Tanner graph and the weights learned. We also show that the learned weights reduce the correlation between the beliefs propagated, which is a desirable property since more independent beliefs result in more reliable estimation of the posterior LLRs [3].

This project was sponsored by the Office of Naval Research.
II. SYSTEM MODEL

In this work, we consider linear block codes. A \((N, K)\) block code maps a message of length \(K\) to a codeword of length \(N\) and is uniquely described by its parity check matrix \(H\) of dimensions \((N - K) \times N\), where the rate of the code is \(R = K/N\). The linear block code can be also represented using a bipartite graph, known as Tanner graph, which can be constructed using its parity check matrix \(H\). The Tanner graph consists of two types of nodes. We refer to them as Check Nodes (CN) that represent the parity check equations and Variable Nodes (VN) that represent the symbols in codeword. There is an edge present between a check node \(c\) and a variable node \(v\) if \(H(c, v) = 1\).

We consider a system with Binary Phase Shift Keying (BPSK) modulation that transmits a random vector \(X \in \{-1, 1\}^N\), where \(N\) is the code length. The generated vector is then passed through a channel to receive \(Y\) as follows

\[
Y = \text{Channel}(X) + W,
\]

where \(\text{Channel}(X)\) applies the effect of channel on the transmit vector \(X\) and \(W\) is the noise at the receiver. As a special case, when the channel is AWGN the LLR at the receiver for \(v\)th symbol can be calculated as

\[
l_v = \frac{p(Y_v = y_v | X_v = -1)}{p(Y_v = y_v | X_v = 1)} = \frac{2y_v}{\sigma^2},
\]

where \(\text{Var}(W) = \sigma^2\) is the noise variance.

Apart from the AWGN channel, we also consider multi-path fading propagation channels from the 3GPP specifications [9], [10]. These channels result in interference and distortion at the receiver because of the multi-path components and also the mobility of the users. Specifically, we consider multi-path fading ETU channel, which has high delay spread and hence hard to achieve perfect equalization. We use the MATLAB LTE Toolbox to simulate the transmission and receiving of data. Finally, we also test the decoder on the real-world data by transmitting and receiving over the air using a USRP N200 SDR setup in non line of sight (NLOS) multi-path environment. More comprehensive results for other channels and the code for the algorithm can be found at: https://github.com/sravan-ankireddy/nams

III. BACKGROUND

In this section, we briefly review the BP decoding and existing classical and data driven methods for its improvement. We then look at the min-sum variant of the BP decoder and the corresponding methods for its improvement. We also introduce and review the baseline algorithms that we will be comparing against.

A. Neural Belief Propagation decoder

The BP decoder is an iterative soft-in soft-out decoder that operates on the Tanner graph structure of the linear block codes to compute the posterior LLRs of the received vector. We also refer to these messages as beliefs. In each iteration, the check nodes and the variable nodes process the information to update the beliefs passed along the edge. Operating in such an iterative fashion allows for incremental improvement in the estimated LLRs, converging towards the right codeword that is obtained by binary quantization after the final iteration.

During the first half of the iteration, at the VN \(v\), the received channel LLR \(l_v\) is combined with the rest of the incoming beliefs \(\mu_{c',v}^{t-1}\) from check node to calculate a new updated belief, to be passed to the check nodes in next iteration. Hence, the message from VN \(v\) to CN \(c\) at iteration \(t\) can be computed as

\[
\mu_{v,c}^t = l_v + \sum_{c' \in N(v) \setminus c} \mu_{c',v}^{t-1},
\]

where \(N(v) \setminus c\) is the set of all check nodes connected to variable node \(v\) except \(c\).

During the latter half of the iteration, at the CN \(c\), the message from the CN to any VN is calculated based on the criterion that the incoming beliefs \(\mu_{v,c}^{t-1}\) at any check node should always satisfy the parity constraint. Specifically, the message from CN \(c\) to VN \(v\) at iteration \(t\) is given by

\[
\mu_{c,v}^t = 2 \tanh^{-1} \left( \prod_{v' \in M(c) \setminus v} \tanh \left( \frac{\mu_{v',c}^{t-1}}{2} \right) \right),
\]

where \(M(c) \setminus v\) is the set of all variable nodes connected to check node \(c\) except \(v\).

Finally, at the end of iteration \(t\), we combine all the incoming beliefs to estimate the posterior belief as

\[
o_v^t = l_v + \sum_{c' \in N(v)} \mu_{c',v}^t.
\]

In [4], it is shown that the decoding performance of BP can be improved by introducing trainable weights along the edges of the Tanner graph, resulting in a Weighted Belief Propagation (WBP), at the VN as

\[
\mu_{v,c}^t = w_v^t l_v + \sum_{c' \in N(v) \setminus c} w_{c',v}^t \mu_{c',v}^t.
\]

B. Neural Min-Sum decoding

To reduce the computational complexity of BP decoding, a hardware friendly variant known as min-sum approximation is used in practice. This simplifies (2) to

\[
\mu_{c,v}^t = \min_{v' \in M(c) \setminus v} \left( (\mu_{v',c}^t) \prod_{v'' \in M(c) \setminus v} \text{sign}(\mu_{v'',c}^t) \right).
\]

While the min-sum approximation simplifies the computation, it also comes with a loss in performance. It is readily shown in [11] that min-sum approximation is always greater than the true LLR from BP. To correct for this, two popular techniques have been introduced [11–13]. The first is to use a normalization factor \(\alpha\) to scale the beliefs appropriately as

\[
\mu_{c,v}^t = \alpha \left( \min_{v' \in M(c) \setminus v} (\mu_{v',c}^t) \right) \prod_{v'' \in M(c) \setminus v} \text{sign}(\mu_{v'',c}^t).
\]
Another approach is to use an offset factor which reduces the beliefs at the CN by $\beta$, while maintaining the sign, as

$$\mu_{c,v}^t = \max \left( \min_{v' \in M(c) \setminus v} (|\mu_{v',c}^t|) - \beta, 0 \right) \prod_{v' \in M(c) \setminus v} \text{sign} \left( \mu_{v',c}^t \right). \quad (7)$$

While these methods have been shown to improve the error correction performance of min-sum algorithm empirically, there has been no analytical solution to calculate the optimal normalization or offset parameters. One of the key reasons for this is the difficulty in modeling the posterior probabilities. The authors in [11] use Monte Carlo simulations to select the best normalization factors and also remark the difficulty in the theoretical analysis beyond first iteration.

Similar to the neural BP decoder, the neural min-sum decoder can be formulated by using different normalization or offset factors along different edges [5], [6]. The equations (6) and (7) can be modified respectively to produce Neural Normalized Min-Sum (NNMS) and Neural Offset Min-Sum (NOMS) algorithms, given by

$$\mu_{c,v}^t = \alpha_{v',c}^t \left( \min_{v' \in M(c) \setminus v} (|\mu_{v',c}^t|) \prod_{v' \in M(c) \setminus v} \text{sign} \left( \mu_{v',c}^t \right) \right) \quad (8)$$

$$\mu_{c,v}^t = \max \left( \min_{v' \in M(c) \setminus v} (|\mu_{v',c}^t|) - \beta_{v',c}, 0 \right) \prod_{v' \in M(c) \setminus v} \text{sign} \left( \mu_{v',c}^t \right) \quad (9)$$

respectively, where the coefficients $\alpha_{v',c}^t$ and $\beta_{v',c}$ denote trainable weights.

The drawback of using neural network based decoders is the increase in storage and computational complexity. Hence, it is important to study the gains achieved and explore possible simplifications to the neural decoder with little or no trade-off in performance.

C. Reducing the complexity via entangling the weights

To reduce the storage complexity of WBP, the authors in [6] propose entangling the weights across iterations. Further, in [2], the authors propose entangling weights not only across iterations but across edges as well. However, this results in a loss of performance, which is compensated using a parameter adapter network that selects different weights for different SNR regions, known as BP-PAN. Similarly, for the neural min-sum decoders, the authors in [6], [7] conclude that entangling the weights across iterations or/and edges only comes with a negligible loss for AWGN channels.

D. Non-AWGN channels

In channels with multi-path and Doppler components, the received LLRs are more unreliable compared to AWGN channel, which makes the decoding harder. Hence, whether the observation in [6], [7] that entangling the weights lead to negligible reliability remains the same for non-AWGN channels remains an open problem. In the following, we study the effect of weight entanglements for various channels, such as ETU and the over-the-air channels, beyond AWGN channels.

IV. NEURAL AUGMENTED MIN-SUM DECODING

In this section, we propose combining the augmentation of normalization and offsets to form a more generalized neural min-sum decoder. We refer to this as Neural Augmented Min-Sum (NAMS) decoder. The modified CN update rule at iteration $t$ is given by

$$\mu_{c,v}^t = \alpha_{v',c}^t \left( \max \left( \min_{v' \in M(c) \setminus v} (|\mu_{v',c}^t|) - \beta_{v',c}, 0 \right) \right) \prod_{v' \in M(c) \setminus v} \text{sign} \left( \mu_{v',c}^t \right). \quad (10)$$

We consider the min-sum approximation and limit the modifications to only CN updates to keep the complexity low. Entanglement. We consider three variants of NAMS with different levels of entanglement.

1) NAMS-full: Different weights for all edges across all iterations as seen in [11].
2) NAMS-relaxed: Entangled weights across the iterations.
3) NAMS-simple: Entangled weights across the iterations and edges.

Loss function and Training. We use the binary cross entropy with logits loss function to measure the difference between the soft output of the each layer and the transmitted codeword and take the sum of losses across each layer to calculate the multiloss term [6] as

$$L(o,x) = -\frac{1}{N} \sum_{t=1}^{T} \sum_{v=1}^{N} x_v \log(o_v^t) + (1 - x_v) \log(1 - o_v^t),$$

where $o_v^t$ is the output for symbol $v$ at iteration $t$ of NAMS and $x$ is the transmitted codeword.

To keep the comparison fair, we use the same amount of training data as [5]. Each mini-batch consists of 10 codewords per SNR point in the training region and Stochastic Gradient Descent (SGD) method is used to learn the weights. We use the Adam optimizer with a initial learning rate of 0.005 and use cosine annealing learning rate scheduler [14].
V. Results

We evaluate the error correction performance of different neural decoders and study the trade-off associated with entanglement for various channels. We also look at the robustness of the learned decoders by testing across channels. Additionally, we also study the adaptability of the learned decoders via additional training on the new channel. Finally, we demonstrate the performance of our NAMS decoder on practical data using USRP N210 SDR transmitter and receiver setup.

A. Entanglement vs channels

In Fig. 1 we plot the the Bit Error Rate (BER) performance of three variants of the NAMS decoder with varying degrees of entanglement. We choose BCH(63,36) specifically to enable direct comparison against the existing neural decoders [4]–[7]. We can see from Fig. 1(a) that the performance is almost indistinguishable across the variants for AWGN channel. In Fig. 1(b), we perform the same experiment for ETU channel. For this simulation, we estimate the channel at the receiver using the pilots of OFDM symbols and use the MMSE equalizer from LTE MATLAB toolbox to perform equalization. We can see from Fig. 1(b) that even after equalization, the choice of entanglement has significant impact on the error correction performance. At a moderate BER of $10^{-3}$, the NAMS-full outperforms NAMS-simple by more than 2 dB.

AWGN vs. ETU. One of the key differences between the AWGN and ETU channels is that the multi-path components in the ETU channel create a correlation between adjacent received symbols. To understand the impact of this better, we compute the expected value of correlation coefficient between the consecutive received symbols post equalization i.e., $E[\rho_{l_v, l_{v+1}}]$ where $l_v$ is LLR of the $v^{th}$ received symbol. From this, we see that the correlation between adjacent symbols is much higher for ETU channel. For the setting in Fig. 1 we observed an expected correlation coefficient of 0.02 and 0.15 between adjacent symbols for AWGN and ETU channels respectively. We conjecture that through training, the decoder learns to compensate for this imperfect equalization and improve the error correction performance.

B. Comparison with existing neural decoders

For a complete comparison, we first compare NAMS-full against the neural BP decoders WBP [4] and BP-PAN [7]. Fig. 2 shows that with similar number of weights, the performance of NAMS-full is comparable to that of WBP decoder but with much lower complexity because of the min-sum approximation. Additionally, we also see that NAMS-full outperforms BP-PAN by more than 1 dB, by taking advantage of the large number of weights.

1 We note that the results in Fig. 2 are based on our PyTorch implementation. When the simulations are run using the publicly available code for NOMS and NNMS, the performance was poor and hence we use our own implementation of these algorithms with the hyper-parameters optimized for the ETU channels.
testing gives the best performance [15], we also desire the

C. Robustness and adaptivity to new channels

that the NAMS decoder has good adaptive capabilities. 2

matched training (i.e., training on the ETU channel), showing

ETU channel, and observe that we recover the reliability of

A WGN channel, on minimal (i.e.) training on the ETU channel. This is mainly because the weight selection by

degrades significantly when the decoder is directly reused on

ETU channel. This is mainly because the weight selection by

PAN is heavily dependent on the SNR which does not translate
directly when the channel is changed. This shows that while

BP-P AN [7] learns very well on a simple channel with very

few parameters, NAMS takes advantage of the larger number
of weights and generalizes well, making it more robust.

Adaptivity. Given the dynamic nature of channel condi-
tions experienced in practical wireless channels, we would like
our decoder to be able to adapt to new channels quickly. To
this end, we train our existing decoder, that was trained on
AWGN channel, on minimal (10 – 15%) amount of data from
ETU channel, and observe that we recover the reliability of
matched training (i.e., training on the ETU channel), showing
that the NAMS decoder has good adaptive capabilities. 2

C. Robustness and adaptivity to new channels

While using the same channel conditions for training and
testing gives the best performance [15], we also desire the
learned decoder to be (a) reasonably robust to any changes in
channel conditions and (b) quickly adaptable to new channels.

D. Over The Air channels

Apart from learning on the simulated data, we train and
test the NAMS decoder on a real multi-path fading envi-
ronment by using two USRP N200 series RF-transceivers to

10^-4

10^-5

10^-6

10^-7

10^-8

10^-9

10^-10

10^-11

10^-12

10^-13

Eb / N0

BER

BP

BP-PAN trained on AWGN

WBP trained on AWGN

NOMS-full trained on AWGN

NNMS-full trained on AWGN

NAMS-full trained on AWGN

Fig. 3: BER for BCH(63,36) ETU channel: The neural min-sum
family is more robust than BP-PAN. Among the neural min-sum
family, NAMS-full is up to 0.5 dB more robust compared to NOMS-
full and NNMS-full.

the ETU channels. Fig. 2 shows that NAMS-full outperforms
NOMS [5] and NNMS [6] by 0.2-0.3 dB.

Robustness. We test the robustness by taking a decoder
trained on AWGN channel and testing on ETU channel. Fig. 3
shows that when the channel changes from AWGN to ETU, the
NAMS-full decoder still outperforms the original BP decoder.
Additionally, the degradation in performance because of the
change of channel is lower for NAMS-full and is up to 0.4
dB more robust compared to other min-sum decoders. It is
also interesting to note that the performance of BP-PAN [7]
degrades significantly when the decoder is directly reused on
ETU channel. This is mainly because the weight selection by
PAN is heavily dependent on the SNR which does not translate
directly when the channel is changed. This shows that while
BP-PAN [7] learns very well on a simple channel with very
few parameters, NAMS takes advantage of the larger number
of weights and generalizes well, making it more robust.

Adaptivity. Given the dynamic nature of channel condi-
tions experienced in practical wireless channels, we would like
our decoder to be able to adapt to new channels quickly. To
this end, we train our existing decoder, that was trained on
AWGN channel, on minimal (10 – 15%) amount of data from
ETU channel, and observe that we recover the reliability of
matched training (i.e., training on the ETU channel), showing
that the NAMS decoder has good adaptive capabilities. 2

C. Robustness and adaptivity to new channels

While using the same channel conditions for training and
testing gives the best performance [15], we also desire the
learned decoder to be (a) reasonably robust to any changes in
channel conditions and (b) quickly adaptable to new channels.

D. Over The Air channels

Apart from learning on the simulated data, we train and
test the NAMS decoder on a real multi-path fading envi-
ronment by using two USRP N200 series RF-transceivers to

10^-4

10^-5

10^-6

10^-7

10^-8

10^-9

10^-10

10^-11

10^-12

10^-13

Eb / N0

BER

BP

BP-PAN trained on AWGN

WBP trained on AWGN

NOMS-full trained on AWGN

NNMS-full trained on AWGN

NAMS-full trained on AWGN

Fig. 4: BER for BCH(63,36) OTA channel: at 3 dBm, NAMS-full
corrects 2.88 × more errors compared to NAMS-simple.

VI. INTERPRETATION

While neural decoders achieve some impressive gains, it is
natural to wonder about the reason for these gains. In [4],
[6], [7], it has been conjectured that the weights mitigate the
effect of cycles and thus improve the performance. While this
is an intuitively sound explanation, the evidence for the same is
lacking. We provide a first empirical evidence that the weights
are closely associated with the cycles in the graph.

For ease of analysis, we consider the NNMS-full version of
the neural min-sum decoder, which only has the normalization
factors. We first study the relation between the weights learned
and the cycles in the graph.

Connection to cycles. We look at three different rates of
length 63 BCH codes, which have different number of length-
4 cycles. For each code, we measure the average weight across
all edges and iterations. Given the degradation associated with
the cycles, we would expect the learned weights to attenuate
more in presence of higher number of cycles. From Table I,
we can see that this holds true, with BCH(63,30) having the
highest number of cycles and hence the least average weight,
while the default weight in traditional min-sum decoder would
be 1. This shows that the weights being learned are indeed
commensurate with the cycles present.

Since these cycles lead to correlation between the incoming
beliefs at the variable nodes, we proceed to empirically esti-
mate and compare these correlations to better understand the
effect of the weights.

Correlation. We consider the AWGN channel for this
analysis, where the correlation between incoming beliefs at
nodes is because of the cycles in Tanner graph. To find if the learned weights mitigate this effect, we measure and compare the expected pairwise correlation coefficient between the incoming messages at the variable nodes \( i.e., \mathbb{E}[\tilde{p}_{v,c_i}\tilde{p}_{v,c_j}], \) \( c_i, c_j \in M(v) \) and \( i \neq j \) and the expectation is over the message and the channel. We observe from Fig. 5 that the correlation is reduced considerably across all the bit positions, which supports our conjecture that the weights are improving the reliability of the posterior probabilities by reducing the correlation.

![Fig. 5: BCH(63,36) AWGN channel at SNR 4dB. The expected pairwise correlation between incoming messages at variable nodes is reduced across all positions.](image)

### TABLE I: Number of short cycles present vs the weights learned.

| Code    | Length-4 cycles | Average weight |
|---------|-----------------|----------------|
|         |                 | AWGN ETU       |
| BCH (63,30) | 10122           | 0.2632 0.4079  |
| BCH (63,36) | 5909            | 0.2987 0.4318  |
| BCH (63,37) | 1800            | 0.3990 0.6583  |

VII. CONCLUSION AND REMARKS

In this work we provided a generalized framework for augmenting the min-sum variant of BP decoder for linear block codes. Through our simulations, we show that having more weights in the decoder is more advantageous in presence of complex channels. We also provide empirical evidence showing relation between the underlying code structure and the weights learned; we show that the weights learned are attenuating the effect of these cycles and improve the reliability of the posterior LLRs. As a future direction, we would like to understand more about the connections between the choice of hyper parameters and the structure of the code, that can help in faster learning of the weights. We also believe that a faster adaptation to newer channels is possible by using meta learning techniques. On the theoretical front, it would be interesting to analyze the performance of neural decoders using Gaussian approximation [16].

REFERENCES

[1] CCSDS and N.A.S.A, “Recommendation for space data system standards, ccsds 231.0-b-4,” https://public.ccsds.org/Pubs/231x0b4e0.pdf, 2021.

[2] W. Ryan and S. Lin, Channel codes: classical and modern. Cambridge university press, 2009.

[3] R. Yazdani and M. Ardakani, “Linear llr approximation for iterative decoding on wireless channels,” IEEE Transactions on Communications, vol. 57, no. 11, pp. 3278–3287, 2009.

[4] E. Nachmani, Y. Be’ery, and D. Burshtein, “Learning to decode linear codes using deep learning,” in 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2016, pp. 341–346.

[5] L. Lugosch and W. J. Gross, “Neural offset min-sum decoding,” in 2017 IEEE International Symposium on Information Theory (ISIT), 2017, pp. 1361–1365.

[6] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be’ery, “Deep learning methods for improved decoding of linear codes,” IEEE Journal of Selected Topics in Signal Processing, vol. 12, no. 1, pp. 119–131, 2018.

[7] M. Lian, F. Carpi, C. Häger, and H. D. Pfister, “Learned belief-propagation decoding with simple scaling and snr adaptation,” in 2019 IEEE International Symposium on Information Theory (ISIT), 2019, pp. 161–165.

[8] M. Yazdani, S. Hemati, and A. Banihashemi, “Improving belief propagation on graphs with cycles,” IEEE Communications Letters, vol. 8, no. 1, pp. 57–59, 2004.

[9] “3gpp ts 36.101. evolved universal terrestrial radio access (e-utra); user equipment (ue) radio transmission and reception,” in 3rd Generation Partnership Project; Technical Specification Group Radio Access Network.

[10] “3gpp ts 36.104. evolved universal terrestrial radio access (e-utra); base station (bs) radio transmission and reception," in 3rd Generation Partnership Project; Technical Specification Group Radio Access Network.

[11] J. Chen and M. Fossorier, “Near optimum universal belief propagation based decoding of low-density parity check codes,” IEEE Transactions on Communications, vol. 50, no. 3, pp. 406–414, 2002.

[12] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu, “Reduced-complexity decoding of ldpc codes,” IEEE Transactions on Communications, vol. 53, no. 8, pp. 1288–1299, 2005.

[13] J. Chen, R. Tanner, C. Jones, and Y. Li, “Improved min-sum decoding algorithms for irregular ldpc codes,” in Proceedings. International Symposium on Information Theory. 2005. ISIT 2005., 2005, pp. 449–453.

[14] I. Loshchilov and F. Hutter, “Sgrd: Stochastic gradient descent with warm restarts,” arXiv preprint arXiv:1608.03983, 2016.

[15] Y. He, J. Zhang, S. Jin, C.-K. Wen, and G. Y. Li, “Model-driven dnn decoder for turbo codes: Design, simulation, and experimental results,” IEEE Transactions on Communications, vol. 68, no. 10, pp. 6127–6140, 2020.

[16] H. El Gamal and A. Hammons, “Analyzing the turbo decoder using the gaussian approximation,” IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 671–686, 2001.