Generalization Bounds for Neural Belief Propagation Decoders

Sudarshan Adiga, Xin Xiao, Ravi Tandon, Bane Vasić, Tamal Bose
Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ, USA.
E-mail: {adiga, 7xinxiao7, tandonr, vasic, tbose}@arizona.edu

2023 IEEE International Symposium on Information Theory (ISIT) | 978-1-6654-7554-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISIT54713.2023.10206901

***Abstract*—Machine learning based approaches are being increasingly used for designing decoders for next generation communication systems. One widely used framework is neural belief propagation (NBP), which unfolds the belief propagation (BP) iterations into a deep neural network and the parameters are trained in a data-driven manner. NBP decoders have been shown to improve upon classical decoding algorithms. In this paper, we investigate the generalization capabilities of NBP decoders. Specifically, the generalization gap of a decoder is the difference between empirical and expected bit-error-rate(s). We present new theoretical results which bound this gap and show the dependence on the *decoder complexity*, in terms of code parameters (blocklength, message length, variable/check node degrees), decoding iterations, and the training dataset size. Results are presented for both regular and irregular parity-check matrices. To the best of our knowledge, this is the first set of theoretical results on generalization performance of neural network based decoders. We present experimental results to show the dependence of generalization gap on the training dataset size, and decoding iterations for different codes.**

Full version of this paper can be found in [1].

I. Introduction

Deep neural networks have emerged as an important tool in 5G and beyond for hybrid beamforming [2]–[4], channel encoding, decoding, and estimation [5]–[20], modulation classification [21]–[23], and physical layer algorithms [24]–[26]. Within the context of channel decoding, prior works have demonstrated that deep neural network based decoders achieve lower bit/frame error rates than conventional iterative decoding algorithms such as belief propagation in several signal-to-noise ratio (SNR) regimes [5], [6], [9], [14]–[16]. In another line of works [27]–[29], deep neural networks have been used to jointly design *both* encoder and decoder. Given the expansive applicability of deep neural networks for channel encoding and decoding, we note here that determining neural network architectures that generalize well to large block length codewords is an active area of research.

Iterative decoding algorithms (such as belief propagation (BP)) are commonly deployed for decoding linear codes; and are known to be equivalent to maximum aposteriori (MAP) decoding when the Tanner graph does not contain short cycles [30]. However, if the Tanner graph contains short cycles, then BP can be sub-optimal i.e., the messages passed between the variable nodes and parity check nodes cannot correctly recover the transmitted codeword [5], [31], [32]. One approach to mitigate the effect of short cycles is by generalizing the BP algorithm by means of a deep learning based approach [5]–[13]. It is shown that the weights learnt by optimizing over the training data ensure that any message repetition between the variable nodes and parity check nodes do not adversely impact the performance of BP based decoders [5], [6]. We refer to this class of belief propagation decoders as *Neural Belief Propagation* (NBP) decoders. The salient aspect of NBP decoders is that its structure is determined from the corresponding Tanner graph, and therefore its architecture is a function of the code parameters itself. Several variants of NBP decoders have been a subject of recent study [7]–[13]. Post-training, it is important that the NBP decoder achieves low bit-error-rate (BER) on unseen noisy codewords. Prior works on NBP decoders [7]–[13] are empirical; to the best of our knowledge there are no theoretical guarantees on the performance of NBP decoders on unseen data. To this end, given a NBP decoder, our goal is to understand how its architecture impacts its generalization gap [33], defined as the difference between empirical and expected BER(s). Motivated by the above discussion, we ask the following fundamental question: *Given a NBP decoder, what is the expected performance on unseen noisy codewords? And how is the generalization gap related to code parameters, neural decoder architecture and training dataset size?*

Main contributions: In this paper, we first upper bound the generalization gap of a generic deep learning decoder as a function of the Rademacher complexity of the individual bits of the decoder output (which we denote as the bit-wise Rademacher complexity). We next consider NBP decoders which belong to the class of belief propagation decoders whose architecture is a function of the code parameters. We upper bound the bit-wise Rademacher complexity as a function of the *covering number* of the NBP decoder, which is the cardinality of the set of all decoders that can closely approximate the NBP decoder. The covering number analysis provides an upper bound with a linear dependence of the generalization gap on spectral norm of the weight matrices and polynomial dependence on the decoding iterations. The

This work was supported by NSF grants CAREER 1651492, CCF-2100013, CNS-2209951, CNS-1822071,CIF- 1855879, CCF-2100013, CCSS-2027844, CCSS-2052751, and NSF-ERC 1941583. Bane Vasić was also supported by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and funded through JPL's Strategic University Research Partnerships (SURP) Program. Bane Vasić has disclosed an outside interest in Codelucida to the University of Arizona. Conflicts of interest resulting from this interest are being managed by The University of Arizona in accordance with its policies.

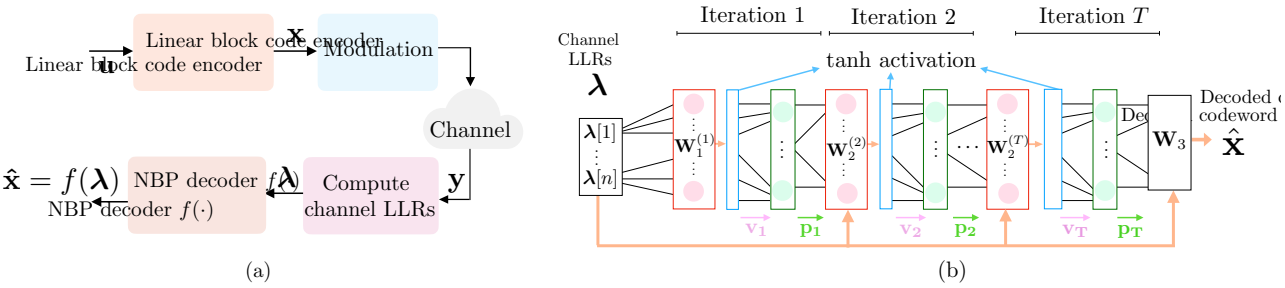

Fig. 1: (a) End-to-End block diagram for communication using neural belief propagation (NBP) decoders for linear block codes; (b) Architecture of the NBP decoder for T decoding iterations where each decoding iteration corresponds to 2 hidden layers: (1) variable node layer, (2) parity check node layer.

bound we obtain is tighter than the other approaches such as VC-dimension and PAC-Bayes approaches in which the upper bound exponentially depends on the decoding iterations. From our results, we show that the generalization gap scales with the inverse of the square root of the dataset size, linearly with the variable node degree and the decoding iterations, and the square-root of the blocklength. To the best of our knowledge, this is the first result that determines upper bounds on the generalization gap as a function of the code-parameters. We also present experimental results to show the dependence of the generalization gap of the NBP decoders on the training dataset size, and the decoding iterations for different codes.

II. Preliminaries and Problem Statement

In Fig. 1, we consider a linear block code denoted by $\mathcal{C}$ of blocklength n and message length k. Let the code $\mathcal{C}$ be characterized by a *regular* parity check matrix $\mathbf{H} \in \{0,1\}^{(n-k)\times n}$, and we denote the Tanner graph as $\mathcal{G} = (\mathcal{V}, \mathcal{P}, \mathcal{E})$; where $\mathcal{V} = \{v_1, \cdots, v_n\}$ is the set of variable nodes, $\mathcal{P} = \{p_1, \cdots, p_{n-k}\}$ is the set of parity check nodes, and $\mathcal{E} = \{e_1, \cdots, e_{nd_v}\}$ is the set of edges. Here, d_v represents the variable node degree, i.e., the number of parity checks a variable node participates in. Let $\{v_i, p_j\}$ denote the edge in the Tanner graph $\mathcal{G}$ connecting variable node v_i to parity check node p_j. $\mathcal{V}(v_j) = \{p_i | \mathbf{H}[i,j] = 1\}$ denote the set of parity check nodes adjacent to the variable node v_j in the Tanner graph $\mathcal{G}$. Similarly, $\mathcal{P}(p_i) = \{v_j | \mathbf{H}[i,j] = 1\}$ denote the set of variable nodes adjacent to the parity check node p_i in $\mathcal{G}$.

Let $\mathcal{Y} \subseteq \mathbb{R}^n$ be the space of n dimensional channel outputs, $\mathcal{X} \subseteq \{0,1\}^n$ be the space of n dimensional codewords, $\mathcal{U} \subseteq \{0,1\}^k$ be the space of k dimensional messages, and $\mathcal{Z} \subseteq \mathbb{R}^n$ be the space of n dimensional channel noise. The message $\mathbf{u} = [\mathbf{u}[1], \cdots, \mathbf{u}[k]]^\top \in \mathcal{U}$ is encoded to the codeword $\mathbf{x} = [\mathbf{x}[1], \cdots, \mathbf{x}[n]]^\top \in \mathcal{X}$. The channel is assumed to be memoryless, described by $\Pr(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^n \Pr(\mathbf{y}[i]|\mathbf{x}[i])$. The receiver receives the channel output $\mathbf{y} = [\mathbf{y}[1], \cdots, \mathbf{y}[n]]^\top \in \mathcal{Y}$; which is the codeword $\mathbf{x}$ modulated, and corrupted with additive noise $\mathbf{z} = [\mathbf{z}[1], \cdots, \mathbf{z}[n]]^\top \in \mathcal{Z}$. The goal of the decoder is to recover the message $\mathbf{u}$ from the channel output $\mathbf{y}$. The input to the decoder is the log-likelihood ratio (LLR) of the posterior probabilities denoted by $\boldsymbol{\lambda} \in \mathbb{R}^{n\times 1}$ and is given as $\boldsymbol{\lambda}[i] = \log \frac{\Pr(\mathbf{x}[i]=0|\mathbf{y}[i])}{\Pr(\mathbf{x}[i]=1|\mathbf{y}[i])}$, for $1 \le i \le n$. Denote the output of the NBP decoder with T decoding iterations as $\hat{\mathbf{x}} = f(\boldsymbol{\lambda})$, where $f(\cdot)$ denotes the decoding function.

The architecture of the NBP decoder is derived from the trellis representation of $\mathcal{G}$ and illustrated in Fig. 1(b). Each decoding iteration t (where, $1 \le t \le T$) corresponds to two hidden layers each of width $|\mathcal{E}| = nd_v$, namely: (1) variable layer $\mathbf{v_t}$, (2) parity check layer $\mathbf{p_t}$. The hidden nodes in layers $\mathbf{v_t}$ and $\mathbf{p_t}$ correspond to the messages passed along the edges of the Tanner graph $\mathcal{G}$. For instance, the output of the node $\mathbf{v_t}[\{l,m\}]$ in the NBP decoder corresponds to the message passed from variable node v_l to parity check node p_m in the t-th iteration, and is given as,

$$\mathbf{v_t}[\{l,m\}] = \tanh\left(\frac{1}{2}\left(\mathbf{W}_1^{(t)}[\{l,m\},l]\boldsymbol{\lambda}[l] + \sum_{m' \in \mathcal{V}(l)\setminus m} \mathbf{W}_2^{(t)}[\{l,m\},\{l,m'\}]\mathbf{p_{t-1}}[\{l,m'\}]\right)\right), \quad (1)$$

where, $\mathbf{p_{t-1}}[\{l,m'\}]$ corresponds to the message passed from the parity check node $p_{m'}$ to the variable node v_l in the $(t-1)$-th iteration. For $t = 1$, we have $\mathbf{p_0} = [0, \cdots, 0]^\top$. $\mathbf{W}_1^{(t)} \in \mathbb{R}^{nd_v \times n}$, and $\mathbf{W}_2^{(t)} \in \mathbb{R}^{nd_v \times nd_v}$ are sparse weight matrices trained using backpropagation in the t-th decoding iteration. $\mathbf{W}_1^{(t)}$ is strictly a lower triangular matrix with exactly d_v non-zero entries in every column, and one non-zero entry in every row. $\mathbf{W}_2^{(t)}$ has exactly $d_v - 1$ non-zero entries in every row, and $d_v - 1$ non-zero entries in every column. We consider that the t-th decoding iteration is characterized by weight matrices $\mathbf{W}_1^{(t)}$, and $\mathbf{W}_2^{(t)}$, where t can take integer values $t \in \{1, \cdots, T\}$. The output of the parity check hidden layer in the t-th decoding iteration for the NBP decoder is,

$$\mathbf{p_t}[\{l,m\}] = \prod_{l' \in \mathcal{P}(m)\setminus l} sign(\mathbf{v_t}[\{l',m\}]) \min_{l' \in \mathcal{P}(m)\setminus l} |\mathbf{v_t}[\{l',m\}]|. \quad (2)$$

The estimated codeword after T decoding iterations in the NBP decoder is given as,

$$\hat{\mathbf{x}}[l] = s(\mathbf{W}_4^{(T)}[l,l]\boldsymbol{\lambda}[l] + \sum_{m' \in \mathcal{V}(l)} \mathbf{W}_3^{(T)}[l,\{l,m'\}]\mathbf{p_T}[\{l,m'\}]) \quad (3)$$

where, $\mathbf{W}_3 \in \mathbb{R}^{n\times nd_v}$, $\mathbf{W}_4 \in \mathbb{R}^{n\times n}$, and $s(\cdot)$ is the sigmoid activation. $\mathbf{W}_3$ is strictly an upper triangular matrix with exactly d_v non-zero entries in every row, while $\mathbf{W}_4$ is a diagonal matrix. The NBP decoder (denoted by $f(\cdot)$) is characterized by the following four sparse weight matrices: (a)

$\mathbf{W}_1^{(t)}$, where $t = 1, \cdots, T$, (b) $\mathbf{W}_2^{(t)}$, where $t = 1, \cdots, T$, (c) $\mathbf{W}_3$, and (d) $\mathbf{W}_4$. The weight matrices are learnt by training the NBP decoder to minimize the bit error rate (BER) loss that is defined as,

$$l_{\text{BER}}(f(\boldsymbol{\lambda}), \mathbf{x}) = \frac{d_H(f(\boldsymbol{\lambda}), \mathbf{x})}{n} = \frac{\sum_{j=1}^{n} \mathbb{1}(f(\boldsymbol{\lambda})[j] \neq \mathbf{x}[j])}{n} \quad (4)$$

Here, $d_H(\cdot, \cdot)$ denotes the Hamming distance, and $\mathbb{1}(\cdot)$ denotes the indicator function. In practice, we train the NBP decoder to minimize the BER loss over the dataset $S = \{(\boldsymbol{\lambda}_j, \mathbf{x}_j)\}_{j=1}^m$ comprising of pairs of log-likelihood ratio and its corresponding codeword. Then, we define the empirical risk of f as $\hat{\mathcal{R}}_{\text{BER}}(f) = \frac{1}{m} \sum_{j=1}^{m} l_{\text{BER}}(f(\boldsymbol{\lambda}_j), \mathbf{x}_j)$. The true risk of f is defined as $\mathcal{R}_{\text{BER}}(f) = \mathbb{E}_{\boldsymbol{\lambda}, \mathbf{x}}[l_{\text{BER}}(f(\boldsymbol{\lambda}), \mathbf{x})]$.

Problem Statement. The generalization gap is defined as the difference $\mathcal{R}_{\text{BER}}(f) - \hat{\mathcal{R}}_{\text{BER}}(f)$. The main goal of this paper is to understand the behavior of the generalization gap (specifically upper bounds) as a function of a) training dataset size, m, b) the *complexity* of the NBP decoder, in terms of the number of decoding iterations T and c) code parameters, such as message length k, blocklength n, variable node degree d_v, parity check node degree d_c.

III. Main Results

In this section, we present our main results on the generalization gap for NBP decoders. Let $S = \{(\boldsymbol{\lambda}_j, \mathbf{x}_j)\}_{j=1}^m$ be the training dataset, and $\mathcal{F}_T$ be a class of NBP decoders with T decoding iterations. For the scope of this paper, we focus on the family of NBP decoders whose non-zero weight entries are bounded by a constant w. Specifically, we assume that for every (i, j) and $1 \leq t \leq T$, $|\mathbf{W}_1^{(t)}[i, j]| \leq w$, $|\mathbf{W}_2^{(t)}[i, j]| \leq w$, $|\mathbf{W}_3[i, j]| \leq w$ and $|\mathbf{W}_4[i, j]| \leq w$, i.e., the maximum absolute value of the (i, j) coordinates for all the weight matrices are bounded by a non-negative constant w. In addition, we also assume that input log-likelihood ratio $|\boldsymbol{\lambda}[i]| \leq b_\lambda$ for all $i = 1, \ldots, n$.

We define the hypothesis class $\mathcal{F}_{L,T}$, derived from the class $\mathcal{F}_T$ of NBP decoders as follows:

$$\mathcal{F}_{L,T} = \{(\boldsymbol{\lambda}, \mathbf{x}) \mapsto l_{\text{BER}}(f(\boldsymbol{\lambda}), \mathbf{x}) : f \in \mathcal{F}_T\}. \quad (5)$$

Intuitively, for each $f \in \mathcal{F}_T$, the output of the corresponding function in $\mathcal{F}_{L,T}$ is the BER loss of the decoder f. We next define the empirical Rademacher complexity of $\mathcal{F}_{L,T}$.

Definition 1. *(Rademacher complexity of $\mathcal{F}_{L,T}$) The empirical Rademacher complexity of $\mathcal{F}_{L,T}$ is defined as*

$$R_m(\mathcal{F}_{L,T}) \triangleq \mathop{\mathbb{E}}_{\sigma} \left[\sup_{f \in \mathcal{F}_T} \frac{1}{m} \sum_{i=1}^{m} \sigma_i l_{BER}(f(\boldsymbol{\lambda}_i), \mathbf{x}_i) \right], \quad (6)$$

where σ_i's are i.i.d. Rademacher random variables, i.e., $\Pr(\sigma_i = 1) = \Pr(\sigma_i = -1) = \frac{1}{2}$.

We note that the loss function l_{BER} takes the values between $[0, 1]$; and consequently using a standard result from PAC learning literature (Theorem 3.3 in [33]), one can bound the generalization gap in terms of $R_m(\mathcal{F}_{L,T})$. Specifically, for any $\delta \in (0, 1)$, with probability atleast $1-\delta$, the generalization gap for any $f \in F_T$ is bounded as follows:

$$\mathcal{R}_{\text{BER}}(f) - \hat{\mathcal{R}}_{\text{BER}}(f) \leq 2R_m(\mathcal{F}_{L,T}) + \sqrt{\frac{\log(1/\delta)}{2m}}. \quad (7)$$

To proceed further, we introduce bit-wise Rademacher complexity of $\mathcal{F}_T$; which is a new notion and captures the correlation between j-th channel output of the NBP decoder and a random decision (Rademacher random variable).

Definition 2. *(Bit-wise Rademacher complexity of $\mathcal{F}_T$) For a NBP decoder class $\mathcal{F}_T$, the empirical bit-wise Rademacher complexity corresponding to its j-th output bit is defined as:*

$$R_m(\mathcal{F}_T[j]) \triangleq \mathop{\mathbb{E}}_{\sigma} \left[\sup_{f \in \mathcal{F}_T} \frac{1}{m} \sum_{i=1}^{m} \sigma_i \cdot f(\boldsymbol{\lambda}_i)[j] \right]. \quad (8)$$

We next present Proposition 1 in which we upper bound the generalization gap as a function of the empirical bit-wise Rademacher complexity $R_m(\mathcal{F}_T[j])$.

Proposition 1. *For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the generalization gap for any NBP decoder $f \in \mathcal{F}_T$ can be upper bounded as follows,*

$$\mathcal{R}_{BER}(f) - \hat{\mathcal{R}}_{BER}(f) \leq \frac{1}{n} \sum_{j=1}^{n} R_m(\mathcal{F}_T[j]) + \sqrt{\frac{\log(1/\delta)}{2m}}, \quad (9)$$

where $R_m(\mathcal{F}_T[j])$ denotes the bit-wise Rademacher complexity for the jth output bit.

The proof of Proposition 1 is presented in Appendix A in [1]. We now present Theorem 1 which is the main result of this paper. The main technical challenge is to bound the bit-wise Rademacher complexity $R_m(\mathcal{F}_T[j])$ as a function of the number of decoding iterations T, training dataset size m and code parameters (blocklength n and variable node degree d_v).

Theorem 1. *For any $\delta \in (0, 1)$, with probability at least $1-\delta$, the generalization gap for any NBP decoder $f \in \mathcal{F}_T$ can be upper bounded as follows,*

$$\mathcal{R}_{BER}(f) - \hat{\mathcal{R}}_{BER}(f) \leq \frac{4}{m} + \sqrt{\frac{\log(1/\delta)}{2m}} + 12\sqrt{\frac{(nd_v^2T + 1)(T + 1)}{m} \log\big(8\sqrt{mn} w d_v b_\lambda\big)}, \quad (10)$$

where n denotes the blocklength, d_v is the variable node degree, T is the number of decoding iterations (number of layers in NBP), m is the training dataset size; w and b_λ are upper bounds on the weights in the NBP decoder and input log-likelihood ratio, respectively.

Proof-sketch of Theorem 1: The detailed proof of Theorem 1 is presented in Appendix in [1] and here we briefly describe the main ideas. We first upper bound the bit-wise Rademacher complexity in terms of Dudley entropy integral (specifically, leveraging Massart's Lemma in [34] and adapting it to our problem). The resulting bound is expressed in terms of the

covering number of the NBP decoder class, i.e., the *smallest* cardinality of the set of functions in $\mathcal{F}_T$ that can closely approximate the NBP decoding function f. To further bound the covering number, we first show that the NBP decoder is Lipschitz in its weight matrices which is proved in Lemma 1 (see Appendix B in [1]). In other words, for a given input, the output of the NBP decoder remains invariant to small perturbations in its weight matrices. Using this fact, we obtain a bound on the covering number of the NBP decoder class in terms of a product of covering numbers (each corresponding to a weight matrix). We then observe that the weight matrices for the NBP decoder are sparse, where the structure and number of non-zero entries is determined by the parity check matrix and the code parameters (such as blocklength n, variable node degree d_v etc.). We then use the fact that the covering number of a sparse weight matrix is always smaller than that of a non-sparse vector (of the same size as the total non-zero entries in the original sparse matrix). Using our result in Lemma 3, we can finally upper bound the bit-wise Rademacher complexity as a function of the code parameters to deduce the result in Theorem 1.

Remark 1 (**Representation in Terms of Code-rate and Parity Check Node Degree**). *The result in Theorem 1 can also be expressed as follows:*

$$\mathcal{R}_{BER}(f) - \hat{\mathcal{R}}_{BER}(f) \leq \frac{4}{m} + \sqrt{\frac{\log(1/\delta)}{2m}} + 12\sqrt{\frac{(nd_c^2(1-\kappa)^2T+1)(T+1)}{m}\log\big(8\sqrt{mn}wd_vb_\lambda\big)}. \quad (11)$$

We use the fact that the blocklength, message length, variable node degree, and parity check node degree are related as $nd_v = (n-k)d_c$. Using this relation in Theorem 1 we obtain (11)*. From the result in* (11) *we note that the generalization gap reduces for codes with a high code-rate κ.*

Remark 2 (**Impact of the Code-parameters**). *We plot the generalization gap bound obtained in Theorem 1 in Fig. 2 for blocklength $n = 100$, variable node degree $d_v = 10$, decoding iterations $T = 10$, and dataset size $m = 10^6$. To understand the dependence of the generalization gap on a parameter, we vary that parameter while keeping the values of the remaining parameters fixed. Smaller training dataset size results in overfitting, and therefore corresponds to a larger generalization gap. We observe this in Fig. 2(a), wherein the generalization gap decays as $\mathcal{O}(\frac{1}{\sqrt{m}})$. While more decoding iterations (i.e., more hidden layers) are expected to improve decoding performance, it can also overfit the training data. Therefore, we expect the generalization gap to increase with the number of decoding iterations. As seen from Fig. 2(b), we note that the generalization gap of the NBP decoder scales linearly as $\mathcal{O}(T)$. Our theoretical result in Theorem 1 tells us that the generalization gap scales with the blocklength as $\mathcal{O}(\sqrt{n})$ as shown in Fig. 2(c). However, the generalization gap scales linearly with the variable node degree as $\mathcal{O}(d_v)$ as shown in Fig. 2(d).*

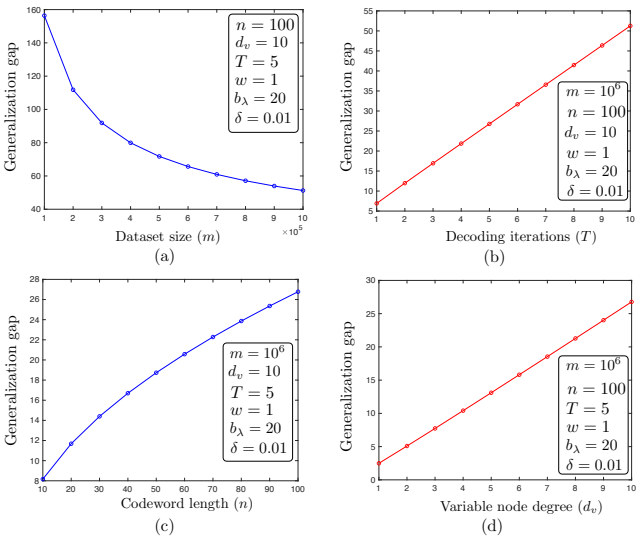

Fig. 2: (a) RHS in Theorem 1 vs Dataset size (m), (b) RHS in Theorem 1 vs Decoding iterations (T), (c) RHS in Theorem 1 vs Blocklength (n), (d) RHS in Theorem 1 vs Variable node degree (d_v).

Remark 3 (**Other Approaches for Bounding the Generalization Gap**). *Vapnik-Chervonenkis (VC) dimension bounds [35], [36], PAC-Bayes analysis [37], [38] are other techniques to upper bound the generalization gap. While VC-dimension approach yields a bound independent of the data distribution, it is found that these bounds are vacuous [37], [39] and scales exponentially with the number of parameters of the neural network. To obtain tighter and non vacuous generalization bounds, prior works [37], [40], [41] have proposed the use of PAC-Bayes analysis. For any $\delta \in (0,1)$, with probability at least $1-\delta$, the generalization gap using PAC-Bayes analysis is upper bounded as, $\mathcal{R}_{BER}(f) - \hat{\mathcal{R}}_{BER}(f) \leq \sqrt{\frac{KL(\zeta||\Gamma)+\log\sqrt{m}+\log(2/\delta)}{2m}}$. The PAC-Bayes prior on the space of neural network decoders ζ is chosen independent of the training data [40], [42]. The KL divergence term between the PAC-Bayes prior ζ and posterior Γ is typically the dominant term in the bound for the generalization gap. While the posterior Γ achieves minimal empirical risk, and is data-dependent; the KL divergence term can be large as the data-independent priors are chosen arbitrarily causing the bound to be vacuous [42]. Furthermore, it is difficult to obtain explicit dependence of the generalization gap on the code parameters (such as codelength, and variable node degree) using the PAC-Bayes analysis. PAC-Learning approach used in this paper leads to a cleaner analysis (inspired by recent results on generalization bounds for graph neural networks and recurrent neural networks [43], [44]), and the bound obtained has a closed-form expression with explicit dependence on code parameters, decoding iterations, and the training dataset size.*

We next show that Theorem 1 can be readily generalized to irregular parity check matrices. Specifically, consider an irregular parity check matrix $\mathbf{H} \in \{0,1\}^{(n-k)\times n}$ where d_{v_i} is the variable node degree of the i-th bit in the codeword,

and d_{c_j} is the parity check node degree of the j-th parity check equation. The NBP decoder corresponding to such this irregular parity check matrix is characterized by the weight matrices $\{\mathbf{W}_1^{(t)}|1 \leq t \leq T\}$, $\{\mathbf{W}_2^{(t)}|1 \leq t \leq T\}$, $\mathbf{W}_3$, $\mathbf{W}_4$. Here, for every $1 \leq t \leq T$, and $\theta = \sum_{i=1}^{n} d_{v_i}$, we have that $\mathbf{W}_1^{(t)} \in \mathbb{R}^{\theta \times n}$, $\mathbf{W}_2^{(t)} \in \mathbb{R}^{\theta \times \theta}$, $\mathbf{W}_3 \in \mathbb{R}^{n \times \theta}$, and $\mathbf{W}_4 \in \mathbb{R}^{n \times n}$. For any value of t, the weight matrix $\mathbf{W}_1^{(t)}$ has one non-zero entry in every row, and d_{v_i} non-zero entries in the i-th column. In the weight matrix $\mathbf{W}_2^{(t)}$, the i-th bit in the codeword with variable node degree d_{v_i} corresponds to d_{v_i} rows and d_{v_i} columns, and these rows and columns each have exactly $d_{v_i} - 1$ non-zero entries. Using similar steps to prove Theorem 1, we derive a bound on the generalization gap for a NBP decoder corresponding to irregular parity check matrix.

Corollary 1. *For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the generalization gap for any NBP decoder $f \in \mathcal{F}_T$ corresponding to irregular parity check matrix can be upper bounded as follows,*

$$\mathcal{R}_{BER}(f) - \hat{\mathcal{R}}_{BER}(f) \leq \frac{4}{m} + \sqrt{\frac{\log(1/\delta)}{2m}} + 12\sqrt{\frac{\sum_{j=1}^{n} d_{v_j}^2 (T+1)^2}{m} \log\left(8\sqrt{mn}w \max_i d_{v_i} b_\lambda\right)}. \quad (12)$$

IV. Experimental Results

In this section, we present some numerical results to complement our theoretical bounds. We consider binary phase shift keying (BPSK) modulation and AWGN channel, and the received channel output for $1 \leq i \leq n$ is given as $\mathbf{y}[i] = (-1)^{\mathbf{x}[i]} + \mathbf{z}[i]$. We focus on Tanner codes with: (i) $n = 155$, $k = 64$, $d_v = 3$, $d_c = 5$; (ii) $n = 310$, $k = 128$, $d_v = 3$, $d_c = 5$ and study the empirical generalization performance of NBP decoders whose architecture was proposed in [5], and also described in Section II of this paper. We adopt the software provided with the papers [6], [7] for our experiments. We train the weights of the NBP decoder until convergence by minimizing the cross-entropy loss between the true and the predicted codeword. We use ADAM optimizer for training with a learning rate of 0.01. We evaluate the NBP decoder by measuring the generalization gap (difference between average BER attained on the test and training datasets). We perform each experiment for 10 trials, and the distribution of the generalization gap over these 10 randomized runs are plotted on a boxplot. We next discuss the impact of the dataset size (m), and the decoding iterations (T) on the generalization gap.

a. Impact of training dataset size (m): We consider the NBP decoder with $T = 3$ decoding iterations (equivalently, 6 layers) trained for channel SNR of 2 dB; we vary the training data set size from $m = 10^3$ to $m = 10^4$ in steps of 1000. From the results in Fig. 3(a), (b), we observe that the generalization gap is the largest for $m = 1000$, and generally decays with m. For a smaller dataset size, the overfitting on the training samples is severe. Therefore, the NBP decoder fails to generalize on unseen samples in the test data. We also repeated the above

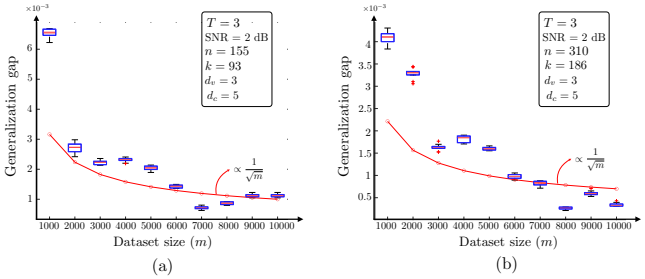

Fig. 3: Generalization gap as a function of the dataset size m at channel SNR = 2 dB for (a) Tanner code with $n = 155$, and $k = 93$, (b) Tanner code with $n = 310$, and $k = 186$.

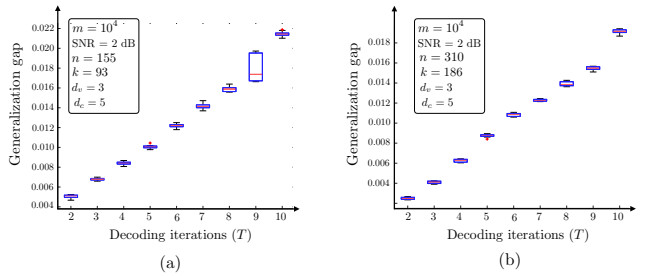

Fig. 4: Generalization gap as a function of the decoding iterations T ($\propto$ number of layers) at channel SNR = 2 dB for (a) Tanner code with $n = 155$, and $k = 93$, (b) Tanner code with $n = 310$, and $k = 186$.

experiment for various values of T as well as by changing SNR. We found the inverse monotonic dependence on m to be consistent across different values of T and SNR.

b. Impact of decoding iterations (T): In this experiment, we study the impact of decoding iterations (which is proportional to the number of hidden layers) in the NBP decoder on the generalization gap. Here, we fixed channel SNR of 2 dB, training dataset size $m = 10^4$ and varied T from $\{2, 3, \ldots, 10\}$. As seen in Fig. 4 the generalization gap grows linearly with T, which is consistent with Theorem 1 (which behaves as $\mathcal{O}(T)$). Increasing the number of parameters will cause overfitting of the NBP decoder resulting in a larger generalization gap. We note that this observation (i.e., linear dependence on T) was consistent for different dataset sizes, and channel SNR values.

V. Conclusions

In this work, we presented results on the generalization gap of NBP decoders as a function of training dataset size, decoding iterations and code parameters (such as blocklength, message length, variable node degree, and parity check node degree). To the best of our knowledge, our work is the first to provide theoretical guarantees for NBP decoders. Our bounds exhibit mild polynomial dependence on the blocklength n and the decoding iterations (layers), T. There are several interesting directions for future work, including a) comprehensive experimental verification of the behavior of generalization gap on code parameters (such as n, d_v); b) obtaining generalization bounds for ML based decoders with practical constraints (such as quantized weights); c) extending the ideas for other type of ML based codes/decoders (i.e., beyond BP type decoder architectures).

REFERENCES

[1] "Generalization bounds for neural belief propagation decoders." Full Version: https://drive.google.com/file/d/16yiGNmiD4PUz-4pyqpOzDD8I5j-A43LY/view?usp=share_link.

[2] X. Li and A. Alkhateeb, "Deep learning for direct hybrid precoding in millimeter wave massive mimo systems," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 800–805, IEEE, 2019.

[3] H. Huang, W. Xia, J. Xiong, J. Yang, G. Zheng, and X. Zhu, "Unsupervised learning-based fast beamforming design for downlink mimo," *IEEE Access*, vol. 7, pp. 7599–7605, 2018.

[4] T. Peken, S. Adiga, R. Tandon, and T. Bose, "Deep learning for svd and hybrid beamforming," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6621–6642, 2020.

[5] 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)*, pp. 341–346, IEEE, 2016.

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

[8] B. Vasić, X. Xiao, and S. Lin, "Learning to decode ldpc codes with finite-alphabet message passing," in *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–9, IEEE, 2018.

[9] E. Nachmani and L. Wolf, "Hyper-graph-network decoders for block codes," in *Advances in Neural Information Processing Systems*, pp. 2329–2339, 2019.

[10] N. Doan, S. A. Hashemi, E. N. Mambou, T. Tonnellier, and W. J. Gross, "Neural belief propagation decoding of crc-polar concatenated codes," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2019.

[11] A. Buchberger, C. Häger, H. D. Pfister, L. Schmalen, and A. G. i Amat, "Pruning and quantizing neural belief propagation decoders," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 1957–1966, 2020.

[12] V. G. Satorras and M. Welling, "Neural enhanced belief propagation on factor graphs," in *International Conference on Artificial Intelligence and Statistics*, pp. 685–693, PMLR, 2021.

[13] E. Nachmani and Y. Be'ery, "Neural decoding with optimization of node activations," *IEEE Communications Letters*, 2022.

[14] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6, IEEE, 2017.

[15] N. Shlezinger, Y. C. Eldar, N. Farsad, and A. J. Goldsmith, "Viterbinet: Symbol detection using a deep learning based viterbi algorithm," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, IEEE, 2019.

[16] J. Seo, J. Lee, and K. Kim, "Decoding of polar code by using deep feed-forward neural networks," in *2018 International Conference on Computing, Networking and Communications (ICNC)*, pp. 238–242, IEEE, 2018.

[17] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Communications Letters*, vol. 23, no. 4, pp. 652–655, 2019.

[18] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: An emerging interdisciplinary paradigm," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 133–139, 2020.

[19] Y. C. Eldar, A. Goldsmith, D. Gündüz, and H. V. Poor, *Machine Learning and Wireless Communications*. Cambridge University Press, 2022.

[20] X. Xiao, B. Vasić, R. Tandon, and S. Lin, "Designing finite alphabet iterative decoders of ldpc codes via recurrent quantized neural networks," *IEEE Transactions on Communications*, vol. 68, no. 7, pp. 3963–3974, 2020.

[21] T. Erpek, T. J. O'Shea, Y. E. Sagduyu, Y. Shi, and T. C. Clancy, "Deep learning for wireless communications," in *Development and Analysis of Deep Learning Architectures*, pp. 223–266, Springer, 2020.

[22] S. Peng, H. Jiang, H. Wang, H. Alwageed, Y. Zhou, M. M. Sebdani, and Y.-D. Yao, "Modulation classification based on signal constellation diagrams and deep learning," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 3, pp. 718–727, 2018.

[23] X. Liu, D. Yang, and A. El Gamal, "Deep neural network architectures for modulation classification," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pp. 915–919, IEEE, 2017.

[24] R. Fritschek, R. F. Schaefer, and G. Wunder, "Deep learning for the gaussian wiretap channel," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2019.

[25] T. O'shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[26] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Communications*, vol. 14, no. 11, pp. 92–111, 2017.

[27] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "Deepcode: Feedback codes via deep learning," in *Advances in neural information processing systems*, pp. 9436–9446, 2018.

[28] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "Turbo autoencoder: Deep learning based channel codes for point-to-point communication channels," in *Advances in Neural Information Processing Systems*, pp. 2758–2768, 2019.

[29] K. Choi, K. Tatwawadi, A. Grover, T. Weissman, and S. Ermon, "Neural joint source-channel coding," in *International Conference on Machine Learning*, pp. 1182–1192, 2019.

[30] H. Tang, J. Xu, S. Lin, and K. A. Abdel-Ghaffar, "Codes on finite geometries," *IEEE Transactions on Information Theory*, vol. 51, no. 2, pp. 572–596, 2005.

[31] J. Liu and R. C. de Lamare, "Low-latency reweighted belief propagation decoding for ldpc codes," *IEEE Communications Letters*, vol. 16, no. 10, pp. 1660–1663, 2012.

[32] S. Zhang and C. Schlegel, "Causes and dynamics of ldpc error floors on awgn channels," in *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1025–1032, IEEE, 2011.

[33] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[34] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[35] P. L. Bartlett, N. Harvey, C. Liaw, and A. Mehrabian, "Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 2285–2301, 2019.

[36] E. D. Sontag *et al.*, "Vc dimension of neural networks," *NATO ASI Series F Computer and Systems Sciences*, vol. 168, pp. 69–96, 1998.

[37] G. K. Dziugaite and D. M. Roy, "Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data," *arXiv preprint arXiv:1703.11008*, 2017.

[38] R. Liao, R. Urtasun, and R. Zemel, "A pac-bayesian approach to generalization bounds for graph neural networks," *arXiv preprint arXiv:2012.07690*, 2020.

[39] G. K. Dziugaite, A. Drouin, B. Neal, N. Rajkumar, E. Caballero, L. Wang, I. Mitliagkas, and D. M. Roy, "In search of robust measures of generalization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11723–11733, 2020.

[40] F. Biggs and B. Guedj, "Non-vacuous generalisation bounds for shallow neural networks," in *International Conference on Machine Learning*, pp. 1963–1981, PMLR, 2022.

[41] P. Alquier, "User-friendly introduction to pac-bayes bounds," *arXiv preprint arXiv:2110.11216*, 2021.

[42] G. K. Dziugaite and D. M. Roy, "Data-dependent pac-bayes priors via differential privacy," *Advances in neural information processing systems*, vol. 31, 2018.

[43] V. Garg, S. Jegelka, and T. Jaakkola, "Generalization and representational limits of graph neural networks," in *International Conference on Machine Learning*, pp. 3419–3430, PMLR, 2020.

[44] M. Chen, X. Li, and T. Zhao, "On generalization bounds of a family of recurrent neural networks," *arXiv preprint arXiv:1910.12947*, 2019.