The challenging neural decoding with general-purpose networks and its improvement via probabilistic embeddings

Xiaolin Wang[1], Joseph Jean Boutros[2], and Olivier Rioul[1]

[1] Télécom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France.
firstname.lastname@telecom-paris.fr
[2] Texas A&M University, 23874 Doha, Qatar. boutros@tamu.edu

Abstract. As deep learning is becoming more and more popular in a variety of fields, it is natural to ask whether it can be used for decoding error-correcting codes. In this work, we show that the answer is yes, but with a caveat: Naive application of general-purpose neural networks is not well suited for bitwise decoding. We thoroughly identify the challenges that prevent general-purpose neural networks from decoding successfully, including the curse of dimensionality and the requirement of extremely high accuracy. We then propose a probabilistic embedding method in the preprocessing stage that facilitate the learning. We also show that this new method allows general-purpose neural networks to decode with a performance that is close to the theoretical optimality while saving time compared to traditional decoding methods such as maximum-likelihood decoding or the BCJR algorithm.

Keywords: Channel Coding, Deep Learning, Communication

1 Motivation

Digital communication is a vital infrastructure of our modern life. It can be abstractively described by a mechanism of transferring encoded messages over a noisy channel and recovering the original messages with arbitrarily high reliability. Traditional algebraic decoding methods such as maximum-likelihood decoding that reach the optimal performance are known to be NP-hard. The thriving deep learning method recently has provided promising solutions in various areas, including neural decoding. This is especially favorable as the inference time is kept nearly constant once the neural network has been trained. However, as seen in the next section, trivially applying a general-purpose neural network to decoding does not perform well, due to the curse of dimensionality and the extremely high accuracy requirement.

Currently, the most popular approaches to neural decoding are based on learning Tanner Graphs [5, 6]. These approaches provide an effective way to avoid the problem of exponential growth of the number of parameters. However, they are constrained by the pattern of a known decoding algorithm and the underlying assumptions it represents, which limits the model's ability to generalize to different decoding scenarios. Some other studies consider non-domain-specific neural networks such as Autoencoder [7, 4], Transformer [2], etc. While delivering promising results and showing flexibility, there are remaining problems that prevent them from becoming practical, including relatively big model sizes for mobile devices as well as the requirement of a redesigned non-linear encoding whereas all practical codes are linear.

Our works present a probabilistic embedding method which is installed before the input layer as a preprocessing. Such probabilistic embedding overcomes the challenges of decoding with arbitrary general-purpose neural networks by lowering the barrier of deployment compared to existing methods. To demonstrate that, we construct a Feedforward Neural Network (FNN) (although our method equally applies to other neural network structures) and compare the decoding performance with and without probabilistic embedding. Our results show that the proposed method significantly improves the decoding performance, and can even approach the theoretical optimality while requiring much less training data.

2 Challenges of decoding with general-purpose neural networks

Although the universal approximation theorem guarantees the existence of a neural network that can approximate any function under some assumptions, it does not provide relationships between hyperparameters such as the number of neurons, the sample size, etc., and the targeting accuracy. In reality, the general-purpose neural networks are observed to be not well suited for decoding in previous studies [3]. We investigate the reasons and give detailed explanations.

Consider an $[n, k]$ binary code $\mathcal{C}$ of length n, therefore $\mathcal{C}$ is a subspace of $\mathbb{F}_2^n$ of dimension k. For simplicity, consider an FNN that only decodes the first bit b_1, in other words, it plays the role of a binary classifier that finds a boundary separating the space $\mathbb{R}^n$ into decision regions for $b_1 = 0$ and $b_1 = 1$. We can prove that the separating boundary scales at least as $2^{k-2}A_{d_{\min}}$ piecewise affine models, where $A_{d_{\min}}$ is the number of non-zero codewords of minimum Hamming weight in $\mathcal{C}$. This already gives 8704 affine models for the small $[15, 11]$ Hamming code (a BCH code with an error-correction capacity $t = 1$) and no less than 1.3×10^9 affine pieces for the binary BCH$[31, 26]$ code. This exponential growth illustrates the curse of dimensionality in neural decoding. As a result, naively applying general-purpose neural networks to decoding is very likely to fail, as the minimum number of neurons required increases exponentially with the code length and the sample size becomes prohibitive.

Another difficulty of neural decoding is the requirement of extremely high accuracy. For instance, the optimal BCH$[15, 11]$ decoder nowadays can decode with a bit error rate (BER) of about 10^{-4} at a signal-to-noise ratio (SNR) of 6 dB, which translates to a classifier of an accuracy of about 99.99%. Reaching such a high level of accuracy is not conventional in common deep learning tasks, especially with noisy input data, as typical fine-tuning/optimizing techniques would, in this extreme case, easily lead to overfitting.

3 Proposed method of probabilistic embedding

Word embedding is a popular technique in natural language processing in order to represent words in real-valued vectors in such a way that words with similar semantics are closer to each other in the embedding vector space. Here, we develop a novel preprocessing procedure that embeds the input vector into a larger vector space, whereby some apriori knowledge of the code structure is combined with the channel likelihood. This is done by transforming the received channel likelihoods (the observation) into vectors of posterior probabilities with the help of a parity check matrix of the considered code, i.e., the proposed embedding does a partial probabilistic decoding.

Specifically, in the context of an $[n, k]$ binary code, a parity check matrix $\mathbf{H}$ is a $(n-k) \times n$ matrix such that $\mathbf{H}\mathbf{c}^\mathsf{T} = 0$, where $\mathbf{c} = (c_1, \ldots, c_n) \in \mathcal{C}$ is a codeword. The binary digits of a codeword are transmitted over a memoryless channel defined by the conditional density $p(y_j|c_j)$ of the noisy channel output $y_j \in \mathbb{R}$ given the input c_j. The channel output $\mathbf{y} = (y_1, \ldots, y_n)$ is transformed into a $(n-k) \times n$ matrix of posterior probabilities Extr_{ij} (known as the extrinsics) by the following equation:

$$\mathrm{Extr}_{ij} \;=\; \mathbb{P}\big(c_j = 1 \mid \mathbf{y}, c_j \;=\; \sum_{j' \neq j, \mathbf{H}_{i,j} \neq 0} c_{j'}\big) \;=\; \frac{1 - \prod_{j' \neq j, \mathbf{H}_{i,j} \neq 0} 1 - 2\,\mathrm{obs}(c_{j'})}{2}, \tag{1}$$

where the probability $\mathrm{obs}(c_j)$ is a normalized version of $p(y_j|c_j)$ assuming that c_j's are Bernoulli(1/2). The matrix Extr_{ij} and the line vector $\mathrm{obs}(c_j)$ are then stacked together vertically, resulting in an input matrix of dimension $(n - k + 1) \times n$ for the neural network. After the transformation, the received signals that originate from the same codeword form better-separated clusters in the probability space (concerning the digit to be decoded), which is a more suitable and learnable input for neural networks.

4 Experiments and results

In our experiment, we built and trained FNNs of four hidden layers with 256, 64, 32, and 8 hyperbolic tangent activated neurons and an output layer with sigmoid activation on randomly generated messages coded using the BCH$[15, 11]$ code and transmitted through additive white Gaussian noise (AWGN) channel. For the neural network, the input is the (probabilistically embedded) received signals, and the output is the soft approximation of codewords. The training is done with the Adam optimizer with a learning rate of 10^{-3}, and the loss function is the binary cross entropy (BCE). For each trial of training, 200 epochs with a batch size of 10000 are carried out. The training database is generated at a fixed SNR of 2 dB, and the test is done on a range of SNRs from 0 to 7 dB. The results are shown in Fig. 1. For reference, we compare the results to the performance of BCJR decoder [1] which represents the theoretical optimality. From the figure, we can see that the naive neural decoder is far from the optimal performance. However, after probabilistic embedding, the performance is much closer to the optimality. Moreover, the neural decoder with embedding outperforms the naive neural decoder with much smaller sample sizes. As far as we know, this is the first approach that demonstrates decoding performance on par with the theoretical optimality of BCH$[15, 11]$ with an FFN.

5 Conclusion

In this study, we have demonstrated the efficacy of our novel probabilistic embedding method in enhancing the decoding performance of general-purpose neural networks for error correction tasks. Our method has

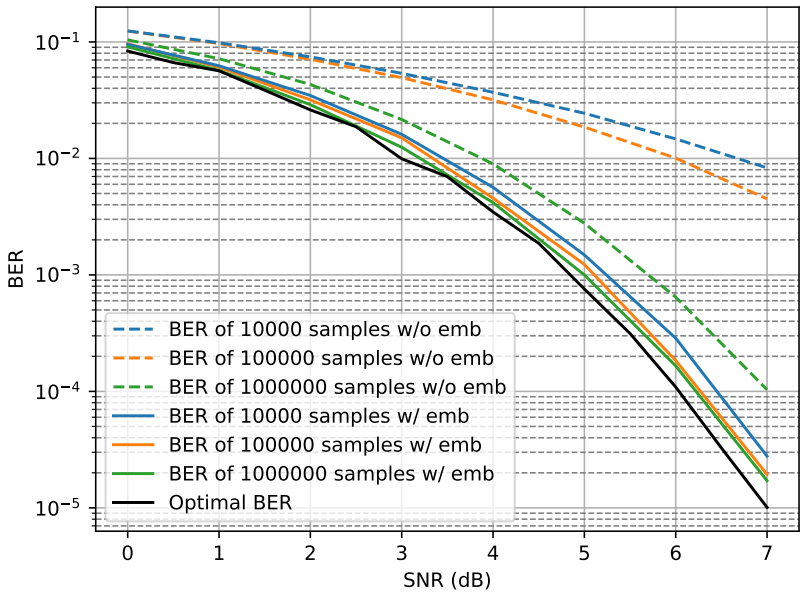

Fig. 1. Comparison of the performance of the FNN decoder with/without probabilistic embedding trained with different sample sizes and the optimal decoder [1]. Each curve is averaged from 10 trials of training.

shown notable success in improving the accuracy of decoding for the BCH[15, 11] code. As a next step, it is important to extend our investigation to more complex error-correcting codes with higher dimensions, longer block lengths, and potentially more intricate patterns.

Additionally, understanding the impact of various parameters on our preprocessing method is another avenue for future research, including sample size, the choice of the training SNR, the choice of the training algorithm, etc.

Moreover, judging from the sparsity and the locality of the input matrix in probability space, it would be possible to apply conventional image processing techniques such as convolutional filters to the input data, in order to achieve effective input compression. This is a promising direction for overcoming the curse of dimensionality.

The method's versatility, potential for adaptation, and performance improvements make it a strong candidate for driving advancements in communication systems. By addressing the critical challenge of decoding with general-purpose neural networks, we believe that our approach can serve as a foundational building block for the design and implementation of next-generation communication systems.

References

1. Bahl, L., Cocke, J., Jelinek, F., Raviv, J.: Optimal decoding of linear codes for minimizing symbol error rate (Corresp.). IEEE Transactions on Information Theory 20(2), 284–287 (Mar 1974)
2. Choukroun, Y., Wolf, L.: Error Correction Code Transformer. In: Advances in Neural Information Processing Systems. vol. 35, pp. 38695–38705. Curran Associates, Inc. (Dec 2022)
3. Jiang, Y.: Deep Learning for Channel Coding. Thesis, University of Washington, Seattle, Washington, United States (2021)
4. Jiang, Y., Kim, H., Asnani, H., Kannan, S., Oh, S., Viswanath, P.: Turbo Autoencoder: Deep learning based channel codes for point-to-point communication channels. In: Advances in Neural Information Processing Systems. vol. 32, pp. 2758–2768. Curran Associates, Inc. (Dec 2019)
5. Nachmani, E., Marciano, E., Lugosch, L., Gross, W.J., Burshtein, D., Be'ery, Y.: Deep Learning Methods for Improved Decoding of Linear Codes. IEEE Journal of Selected Topics in Signal Processing 12(1), 119–131 (Feb 2018)
6. Nachmani, E., Wolf, L.: Hyper-Graph-Network Decoders for Block Codes. In: Advances in Neural Information Processing Systems. vol. 32, pp. 2329–2339. Curran Associates, Inc. (Dec 2019)
7. O'Shea, T., Hoydis, J.: An Introduction to Deep Learning for the Physical Layer. IEEE Transactions on Cognitive Communications and Networking 3(4), 563–575 (Dec 2017)