Machine Learning Autoencoder Applied to Communication Channels

E. Dadalto Camara Gomes[1] M. Benammar[2]

[1]ISAE-SUPAERO
Université de Toulouse
31055, Toulouse, France
Email: eduardo.dadalto-camara-gomes@student.isae-supaero.fr

[2]Department of Electronics, Optronics, and Signal processing
ISAE-SUPAERO
31055, Toulouse, France
Email: meryem.benammar@isae-supaero.fr

ISAE-SUPAERO, 2019

Outline

Context

Communication system context in general - what field will I be treating

- My first point.
- My second point.

Machine Learning applications - what could we do in communication system

- My first point.
- My second point.

Relevance & Challenges

Explain why the work is relevant and explain what are the challenges

- My first point.
- My second point.

Problem Statement

What exactly I will solve in this work

- My first point.
- My second point.

Second Slide Title

- First item.

Second Slide Title

- First item.
- Second item.

Second Slide Title

- First item.
- Second item.
- Third item.

Second Slide Title

- First item.
- Second item.
- Third item.
- Fourth item.

Second Slide Title

- First item.
- Second item.
- Third item.
- Fourth item.
- Fifth item.

Second Slide Title

- First item.
- Second item.
- Third item.
- Fourth item.
- Fifth item. Extra text in the fifth item.

Maximum a Posterior (MAP) Rule

Implementation of a MAP decoder for a linear block code through a BSC.

Algorithm 1 MAP rule for BSC and linear block code.

Input: received block $\mathbf{y}^n \in \{0,1\}^n$, code word set $\mathcal{X}$ and generator matrix $G_{k\times n}$.

Output: message estimation $\hat{\mathbf{u}}^k \in \{0,1\}^k$.

procedure MAP DECODER($y, \mathcal{X}, G$)

$p \leftarrow$ channel crossover probability

for i in $range(2^k)$ **do**

distances$[i] \leftarrow d_H(\mathbf{y}, word[i] \in \mathcal{X})$

$\hat{\mathbf{x}} \leftarrow argmin(\text{distances})$

$\hat{\mathbf{u}} \leftarrow \hat{\mathbf{x}}G^{-1}$ **return** $\hat{\mathbf{u}}$

Neural Network's Design and Architecture I

Show the architecture used for each case and remarks some important parameters

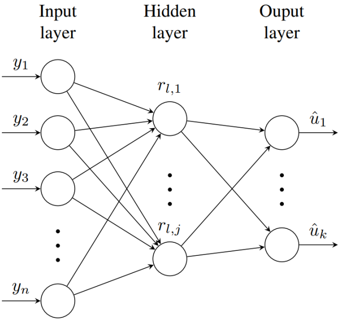


Figure: MLNN representative diagram, where $\mathbf{y}^n$ is the input vector, $\mathbf{r}_l^j$ is a hidden layer vector and $\hat{\mathbf{u}}^k$ is the output vector.

Neural Network's Design and Architecture II

Show the architecture used for each case and remarks some important parameters

Table: DNN array decoder architecture and parameters.

Decoder	Dense: 128, activation: ReLU, input size: n
	Dense: 64, activation: ReLU
	Dense: 32, activation: ReLU
	Dense: k, activation: Sigmoid
Total parameters: 12776	

Loss func.	Optimizer	N. Epochs	Batch Size
Binary cross-entropy	Adam	2^{16}	256

Neural Network's Design and Architecture III

Show the architecture used for each case and remarks some important parameters

Table: DNN one-hot decoder architecture and parameters.

Decoder	Dense: 256, activation: Softmax, input size: n
Total parameters: 4352	

Loss func.	Optimizer	N. Epochs	Batch Size
Binary cross-entropy	Adam	2^{14}	256

Neural Network's Design and Architecture IV

Show the architecture used for each case and remarks some important parameters

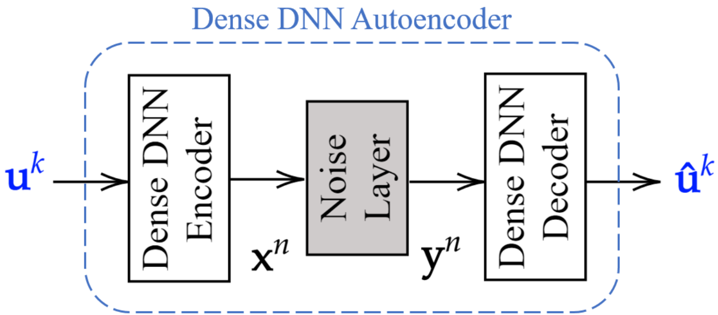


Figure: Representation of a DNN autoencoder composed of dense layers.

Neural Network's Design and Architecture V

Show the architecture used for each case and remarks some important parameters

Table: DNN array autoencoder architecture.

Encoder	Dense: 512, activation: ReLU, BN[1], input size: 8 Dense: 256, activation: ReLU, BN Dense: 16, activation: Sigmoid
Channel	Lambda: $Round(\mathbf{x})$, input size: 16 Lambda: $\mathbf{x} \oplus$ noise
Decoder	Dense: 128, BN, input size: 16 Dense: 64, activation: ReLU, BN Dense: 8, activation: Sigmoid
Total parameters: 154072	

Loss func.	Optimizer	N. Epochs	Batch Size
MSE	Adam	2^{17}	256

Neural Network's Design and Architecture VI

Show the architecture used for each case and remarks some important parameters

Table: DNN one-hot autoencoder architecture.

Encoder	Dense: 196, activation: ReLU, BN, input size: 256
	Dense: 128, activation: ReLU, BN
	Dense: 96, activation: ReLU, BN
	Dense: 64, activation: ReLU, BN
	Dense: 32, activation: ReLU, BN
	Dense: 16, activation: Sigmoid
Channel	Lambda: $Round(\mathbf{x})$, input size: 16
Decoder	Dense: 128, activation: ReLU, BN, input size: 16
	Dense: 256, activation: Softmax
Total parameters: 134052	

Loss func.	Optimizer	N. Epochs	Batch Size
MSE	Adam	2^{16}	256

[1] Batch Normalization (BN)

Error Correction and Monte Carlo Simulations

Explain how we could use NN to predict the results with certain confidence.

- My first point.
- My second point.

Blocks

Block Title

You can also highlight sections of your presentation in a block, with it's own title

Theorem

There are separate environments for theorems, examples, definitions and proofs.

Example

Here is an example of an example block.

DNN Array Decoder

Show the results for the array decoder in terms of train p, Mep, Parameters, etc

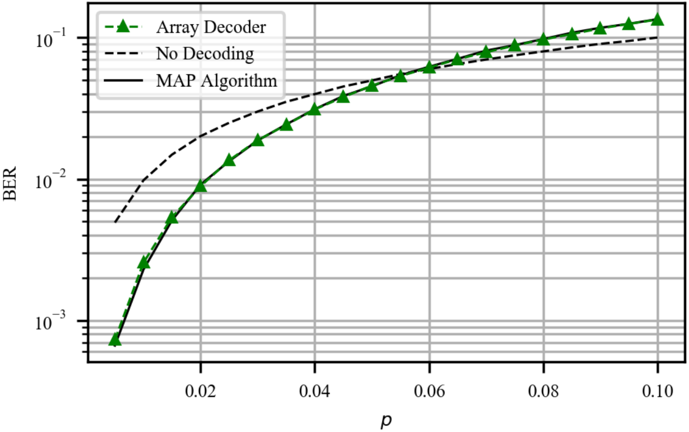


Figure: Array decoding BER performance. NN trained with a channel crossover probability error of $p_t = 0.07$.

DNN One-hot Decoder

Show the results for the one-hot decoder in terms of train p, Mep, Parameters, etc

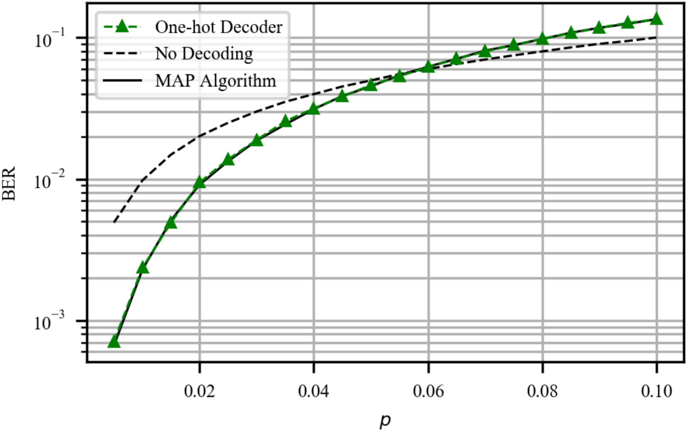
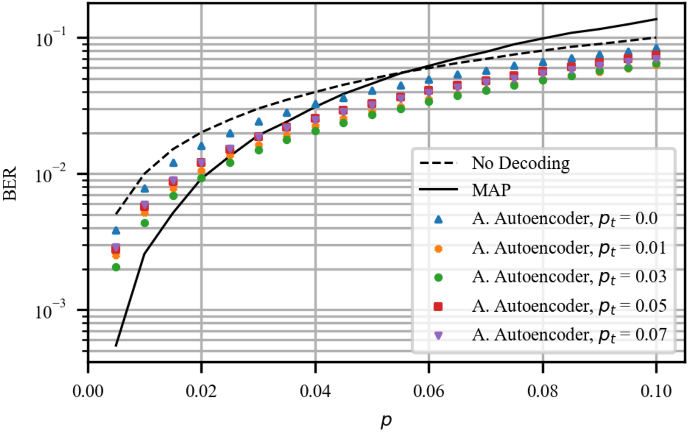


Figure: One hot decoding BER performance. NN decoder trained with a channel crossover probability error of $p_t = 0$.

DNN Array Autoencoder I

Show the results for the autoencoder in terms of train p, Mep, Parameters, etc



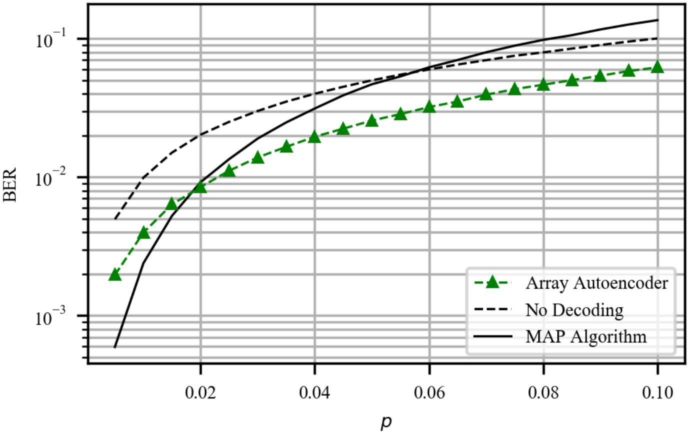
DNN Array Autoencoder II

Show the results for the autoencoder in terms of train p, Mep, Parameters, etc

Figure: Training crossover probability simulation for the array autoencoder. $P_t = 0.3$ demonstrated to have best performance to this particular architecture.

DNN Array Autoencoder III

Show the results for the autoencoder in terms of train p, Mep, Parameters, etc



Show the results for the autoencoder in terms of train p, Mep, Parameters, etc

Figure: Array autoencoder BER performance. DNN array autoencoder trained with a channel crossover probability error of $p_t = 0.03$.

DNN One-hot Autoencoder

Show the results for the autoencoder in terms of train p, Mep, Parameters, etc

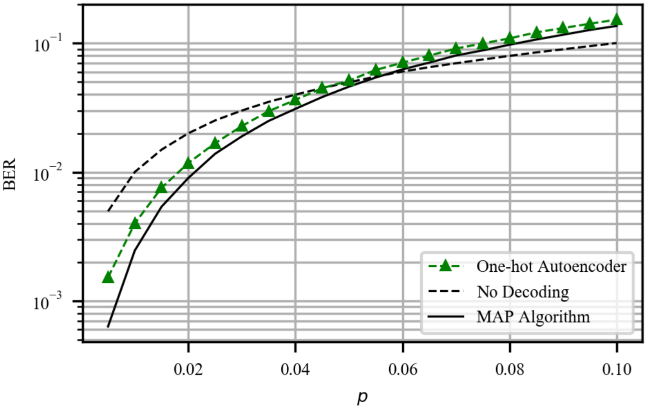


Figure: One-hot autoencdoer BER performance. Trained without a noise channel.

Delay Time Analysis

Comparison of encoding, transmission and decoding time for each method.

Table: Decoding time comparison between the MAP algorithm and the DNN decoders and autoencoders. The data is normalized to the average MAP algorithm decoding time.

MAP	Array Decoder	One-hot Decoder
1.00 ± 0.02	0.74 ± 0.03	0.76 ± 0.02
Array Autoencoder		One-hot Autoencoder
1.33 ± 0.05		3.02 ± 0.06

Conclusions

- My first point.
- My second point.

Future Work

- My first point.
- My second point.

Acknowledgment

I would like to thank ...

- The guidance and advice from my supervisor Meryem Benammar, Ph.D;
- My colleague Rémy Zawislak who worked in the same theme and collaborated indirectly to the research;
- Marjorie Grzeskowiak Lucas, Ph.D, who through the Institut Supérieur de l'Aéronautique et de l'Espace (ISAE-SUPAERO) enabled the confection of the project; and
- The supervisors, English teachers and audience present today.

Bibliography I

C. E. Shannon, "A mathematical theory of communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, pp. 3–55, Jan. 2001.

F. D. Calabrese, L. Wang, E. Ghadimi, G. Peters, and P. Soldati, "Learning radio resource management in 5g networks: Framework, opportunities and challenges," *CoRR*, vol. abs/1611.10253, 2016.

T. J. O'Shea and J. Hoydis, "An introduction to machine learning communications systems," *CoRR*, vol. abs/1702.00832, 2017.

T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to Communicate: Channel Auto-encoders, Domain Specific Regularizers, and Attention," *arXiv e-prints*, Aug. 2016.

Bibliography II

D. Goldin and D. Burshtein, "Performance Bounds of Concatenated Polar Coding Schemes," *arXiv e-prints*, Oct. 2017.

E. Worm, S. Member, P. Hoeher, S. Member, and N. Wehn, "Turbo-decoding without snr estimation," *IEEE Communications Letters*, pp. 193–195, 2000.

A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260–269, April 1967.

P. Robertson, P. A. Hoeher, and E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding.," *European Transactions on Telecommunications*, vol. 8, no. 2, pp. 119–125, 1997.

Bibliography III

- H. J., R. P., and P. L., "iterative turbo decoding of systematic convolutional codes with the map and sova algorithms," pp. 21 – 29, 10 1994.
- M. Jordan and R. Nichols, "The effects of channel characteristics on turbo code performance," pp. 17 – 21 vol.1, 11 1996.
- M. Ibnkahla, "Applications of neural networks to digital communications-survey," *Signal Processing*, vol. 80, pp. 1185–1215, 07 2000.
- M. A. Nielsen, "Neural networks and deep learning," 2018.
- K. P. Murphy, *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013.

Bibliography IV

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016.

F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
PMID: 16764513.

Bibliography V

M. Benammar and P. Piantanida, "Optimal training channel statistics for neural-based decoders," in *52nd Asilomar Conference on Signals, Systems, and Computers, ACSSC 2018, Pacific Grove, CA, USA, October 28-31, 2018* (M. B. Matthews, ed.), pp. 2157–2161, IEEE, 2018.

N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *CoRR*, vol. abs/1609.04836, 2016.

S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.