Deep learning-based channel coding of short packets : an IoT opportunity

Meryem Benammar[†]

[†] Département d'Electronique, Optronique et Signal (DEOS)
ISAE-Supaéro

[†]

Neural-based communication algorithms

Intended learning outcomes

- Implement a communication chain with a Maximum A Posteriori (MAP) detector
- Implement basic classifiers using neural network
- Implement a neural based decoder which approaches the MAP
- Implement an auto-encoder to implement a point to point communication chain

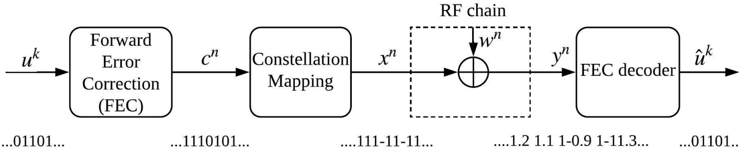
About ...

- Format : short project
- Working teams : in pairs or triplets
- Evaluation : competition at the end of the course

The channel coding problem

Deep channel decoding

The channel coding problem

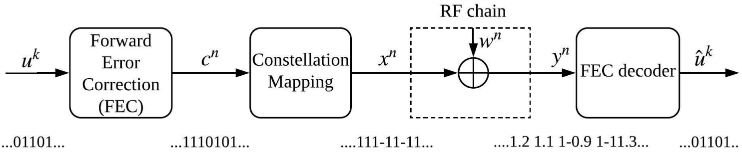


- An information source : bitstream u^k (k bits)
- An error correction code (FEC) : codewords c^n (n bits)
- A constellation mapping, BPSK for instance : channel input x^n (n symbols)
- Adjunction of a memoryless additive noise

$$y^n = x^n + w^n$$

  with associated density $P(y^n|x^n)$
- A FEC decoder : an estimate $\hat{u}^k$ (k bits) of u^k

The channel coding problem (cont.)



Construct a FEC encoder and decoder

- Reliability

$$P_e = \mathbb{P}\left(\hat{U}^k \neq U^k\right) \to 0$$

- Spectral efficiency

$$\frac{k}{n} \text{ as high as possible}$$

- Decoding complexity/latency as low as possible

What is the best channel encoder/decoder to achieve the tradeoff ?

A bit of History

Shannon enunciated the capacity formula in 1948

$$P_e = \mathbb{P}\left(\hat{U}^k \neq U^k\right) \to 0 \quad \text{i.i.f} \quad \frac{k}{n} \leq \max_{P_X} I(X;Y) = f\left(P_{Y|X}\right)$$

A bit of History

Shannon enunciated the capacity formula in 1948

$$P_e = \mathbb{P}\left(\hat{U}^k \neq U^k\right) \to 0 \quad \text{i.i.f} \quad \frac{k}{n} \leq \max_{P_X} I(X;Y) = f\left(P_{Y|X}\right)$$

- Multi-level codes (Ungerbock 1976, Imai & Hirakawa 1977) : Successive cancellation decoding
- LDPC codes (Gallager 1960s, McKay 2000) : Belief propagation
- Turbo-codes (Glavieux & Bérou 1990) : BCJR, Viterbi
- Polar codes (Arikan 2008) : List successive cancellation

A bit of History

Shannon enunciated the capacity formula in 1948

$$P_e = \mathbb{P}\left(\hat{U}^k \neq U^k\right) \to 0 \quad \text{i.i.f} \quad \frac{k}{n} \leq \max_{P_X} I(X;Y) = f\left(P_{Y|X}\right)$$

- Multi-level codes (Ungerbock 1976, Imai & Hirakawa 1977) : Successive cancellation decoding
- LDPC codes (Gallager 1960s, McKay 2000) : Belief propagation
- Turbo-codes (Glavieux & Bérou 1990) : BCJR, Viterbi
- Polar codes (Arikan 2008) : List successive cancellation

Linear block codes

Encoding $\underbrace{c^n}_{\text{codeword}} = \underbrace{G}_{\text{generator}} \times \underbrace{u^k}_{\text{information}} \mod[2]$

Decoding $\underbrace{\hat{u}^k}_{\text{information}} = \underbrace{f_{\text{dec}}}_{\text{decoder}} \underbrace{(y^n)}_{\text{received}}$

Yet, in IoT ...

Communications are quite costy

- Power consumption
- Memory available
- Computational complexity

https ://www.cio.com/article/2983713/internet

At short and medium blocklength n

- Encoding complexity is reasonable (linear block codes)
- Sub-optimal decoding performances (BP and cycles, Polar and list size)
- Often, high latency/complexity

The need for low complexity, quasi optimal decoders!

The channel coding problem

Deep channel decoding

From Biology to Computer Science

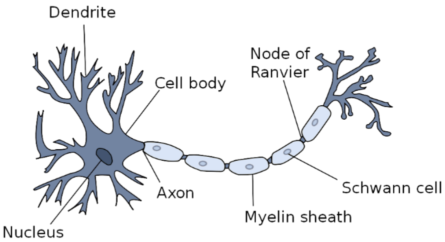
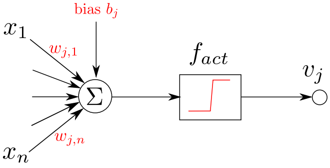
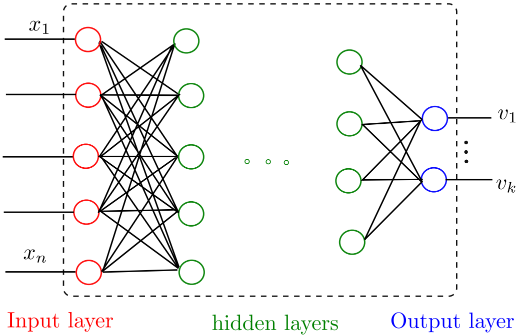


FIGURE – D. Kriesel – A Brief Introduction to Neural Networks



$$v_j = f_{\text{act}}\left(\sum_i w_{i,j} x_i + b_j\right)$$

Artificial neural networks

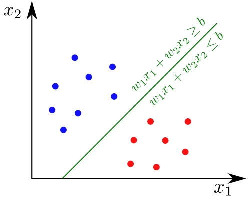


- Set of input vectors $x^n = (x_1, ..., x_n)$
- Set of expected outputs $v^k = (v_1, ..., v_k)$
- Training : optimization problem (backpropagation)

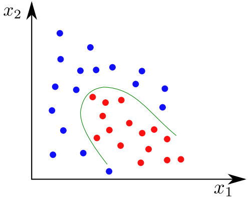
$$W^\star = \text{argmin}_W \ \ \text{Loss}\left(v^k - \hat{v}^k\right)$$

Function approximation

With one layer of perceptrons : all linearly separable functions

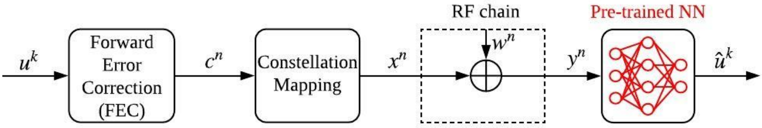


With two layers : all functions
With three layers : all functions and their derivatives

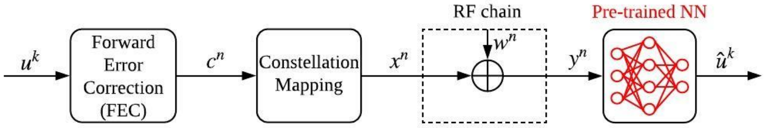


Neural networks are universal function approximators

Neural network-based decoders



Neural network-based decoders



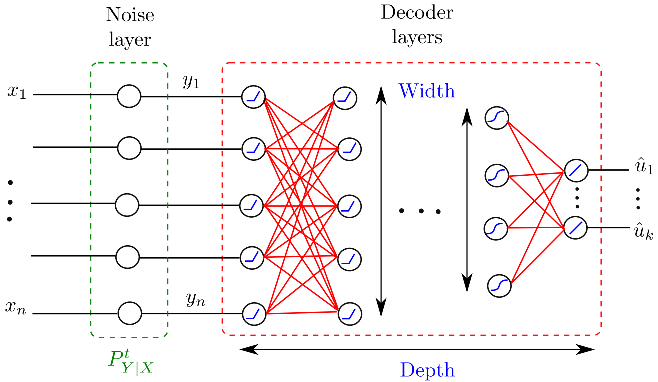
- Fix a FEC encoder $f_e(\cdot)$

$$c^n = f_e\left(u^k\right)$$

- Fix a constellation mapping
- Generate the training and validation data set
- Train a neural network offline to recover the sequence

> Trade online complexity at the receiver with off-line complexity at the transmitter

Design meta-parameters : Training



How to choose the optimal design metaparameters ?

What's around in literature

Early contributions

- Linear block code decoder [BruckBlaum'89]
- Hamming codes [TalliniCull'95]
- Viterbi decoders [WangWicker'96]

More recent ones

- Improved message passing alogrithm [NachmaniBe'eryBurshtein'16]
- Polar decoding [GruberCammererHoydisTenBrink'17]
- LDPC codes [LewandowskiBauch'17]
- Syndrome decoding of block codes [BennatanChoukrounKissilev'18]

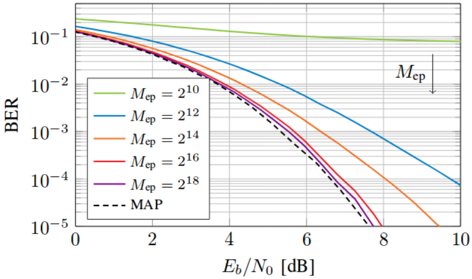
Design is often heuristic or mimics existing algorithms

What's around in literature

[Gruber-Cammerer-Hoydis-TenBrink'17] investigated

- FEC encoder : Polar code ($k = 8, n = 16$)
- BPSK modulation
- Channel $P_{Y|X}$ Gaussian with variance σ^2
- Network size $[16, 128, 64, 32, 8]$
- Various channel conditions

$$Eb/N0 = -10 \log_{10} \left(\sigma^2\right)$$



A neural network can approach the Maximum A Posteriori