Designing Communication Receivers Using Machine Learning Techniques

Brian M. Kurkoski
Japan Advanced Institute of Science and Technology

November 20, 2017
Oregon State University
Corvallis, Oregon, USA
Smartphone Communications Receivers

Your smartphone has numerous wireless receivers:
- cellular radio (LTE, CDMA, UMTS, GSM)
- WiFi (802.11n, ac, etc.)
- Bluetooth

The data storage also uses a “receiver”
- flash memories

Many other applications:
- digital video broadcast, wired ethernet, IoT-like protocols, SSDs and hard drives
Smartphone Communications Receivers

- “Communication receiver” includes equalization, detection and error-correction
- VLSI is more efficient than software
- VLSI approximates floating point numbers using fixed point
- tradeoff between number of bits and performance
- “ad hoc” implementation by engineers

Can we give a theoretical foundation to implementations?
Outline

1 Background on Communications and LDPC Codes
   - Just enough information theory
   - LDPC codes and their decoding

2 Quantization and Classification
   - Channel quantization = classification in machine learning
   - Optimal quantization for binary inputs
   - “KL-means” algorithm and information bottleneck method

3 Hardware-Aware Information Theory
   - Max-LUT method
   - LDPC decoding: 4 bits/message “performs like floating point”
Just Enough Information Theory

Good channel (few errors) — high code rate $R$ (few parity bits)
Just Enough Information Theory

information \[ \rightarrow \text{ENCODER} \rightarrow \text{NOISY CHANNEL} \rightarrow \text{DECODER} \rightarrow \text{received} \ y \rightarrow \text{estimated codeword} \ \hat{x} \]

- Good channel (few errors) — high code rate \( R \) (few parity bits)
- Bad channel (many errors) — low code rate \( R \) (many parity bits)
Just Enough Information Theory

What is the best we can do?

Code rate < Channel Capacity

\[ R < C = \max_{p_X(x)} I(X; Y) \]

Claude Shannon: mutual information is the highest achievable rate
The Reasons for Success of LDPC Codes

Many error-correcting codes: Hamming, Reed-Solomon, BCH, convolutional codes

Low-density parity-check (LDPC) codes are widely used. In communications standards:

- cellular data, WiFi 802.11, video broadcasting
- Ethernet over twisted pair

In data storage:

- flash memories, SSD drives, hard drives

Reasons for success of LDPC code:

- LDPC codes are good codes — as block length increases, can approach Shannon limit
- LDPC decoding complexity is linear in the block length
Low-Density Parity-Check (LDPC) Codes

LDPC code is defined by a low-density parity-check matrix $H$

A codeword $x$ satisfies $Hx = 0 \mod 2$

$H = \begin{bmatrix}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0
\end{bmatrix}$

Parity-Check Matrix

Recall decoder attempts to solve: $\max_{x \in \mathcal{C}} \Pr(y|x)$
Decoding LDPC Codes

Input from channel

\[
\log \frac{\Pr(x_0 = 1 | y_0)}{\Pr(x_0 = 0 | y_0)} = -0.4512
\]
Decoding LDPC Codes

Variable-to-check messages

Iteration 1 (first half):
pass channel messages to check nodes
Decoding LDPC Codes
Check-to-Variable Messages

Iteration 1 (second half):
check nodes perform processing,
pass results to variable nodes
Decoding LDPC Codes

Variable-to-check messages

Iteration 2 (first half):
variable nodes perform processing,
pass results to check nodes
In practice, perform 5 to 50 iterations. Stop when:

- Codeword is detected $H\mathbf{x} = 0$
- maximum number of iterations reached
Nodes are Functions
Edges are Messages

check node

$-2 \tanh^{-1} \left( \prod_{i \in \mathcal{M}(c) \setminus c} \tanh \left( - \frac{L_i}{2} \right) \right)$

variable node

$Y_v + \sum_{i \in \mathcal{N}(v) \setminus v} V_i$

final decisions:

$\hat{L}_v = Y_v + \sum_{i \in \mathcal{N}(v)} V_i$

“sum-product rule”:
for edge $e$, do not use incoming message $e$
Quantization of Factor Graph

- Fixed point messages/functions replace floating point
- 3 to 8 bits/message is typically used

Efficient quantization schemes are desired
VLSI quantization schemes are often chosen ad hoc way by engineers
Quantization and Classification

- Key problem is channel quantization to maximize mutual information
- Identical to classification in machine learning
- Optimal, polynomial-time algorithm for binary input
- “KL-means” algorithm is suboptimal but efficient
Given a discrete memoryless channel and input distribution $p_{XY}(x, y)$,
Given a discrete memoryless channel and input distribution $p_{XY}(x, y)$, find the quantizer $Q$ which maximizes mutual information:

$$Q^* = \arg \max_Q I(X; Z)$$

with $|Z| < |Y|$. 

Key Problem — Quantization

Quantizer $Q$: mapping from channel outputs to quantizer outputs
Given a discrete memoryless channel and input distribution $p_{XY}(x, y)$, find the quantizer $Q$ which maximizes mutual information:

$$Q^* = \arg \max_Q I(X; Z)$$

with $|Z| < |Y|$. $|Z| \geq |Y|$ is trivial.

Claude Shannon: mutual information is the highest achievable rate.
Machine Learning: Classification

Minimum-entropy optimal classifier $Q^*$:

$$Q^* = \arg \min_Q H(Z|X)$$

Connection with machine learning:

$$\max_Q I(X; Z) = H(X_3) - \min_Q H(Z|X)$$

---

**Pr($Y|X$)**

$X \xrightarrow{\Pr(Y|X)} Y \xrightarrow{Q} Z$

strawberry

banana

kiwi

---

$X$

$Y$

$Z$

---

Camera
Backward Channel $\Pr(X \mid Y)$ as a Vector

$$\mathbf{u}_y = \begin{bmatrix} \Pr(X = 1 \mid Y = y), \Pr(X = 2 \mid Y = y), \ldots, \Pr(X = J \mid Y = y) \end{bmatrix}$$

$J = 3$ input DMC

$J = 3$ dimensional probability simplex
Quantization in Backwards Channel
Quantization in Backwards Channel

quantizer outputs Z

cluster
Optimal Quantizer Design Algorithm for Binary Inputs

- Cluster (preimage of optimal quantizer) is convex [Burshtein et al, 1992]
- Dynamic programming: Search over all convex quantizers
- Provably optimal — max mutual information
- Complexity is $M^3$

$$u_y = [1 \ 0]$$

K-Means Algorithm (machine learning)
Lloyd-Max Algorithm (information theory)

1. given \( n \)-dimensional data set, randomly choose \( K \) means (centroids)

2. nearest neighbor \( K \) clusters consists of data points closest to its mean in Euclidean distance

3. centroid step move the mean to the center of the cluster

Not optimal, but works well in practice. Hugely successful in machine learning.
Clustering with Bregman Divergences

Motivated by the success of the K-Means clustering algorithm, we develop a Bregman divergence measure for vector space models. Our approach allows for the use of a wide variety of distance measures, including the cosine similarity and Itakura-Saito divergences. The Bregman divergence provides a good measure of similarity for text documents.

Enhanced Word Clustering for Hierarchical Text Classification

ABSTRACT

In this paper we propose a new algorithm for word clustering. In previous work, an objective function has been used to cluster documents into semantic groups. In this paper, we propose using a new objective function that considers the hierarchical structure of the documents.
KL-Means Algorithm: KL Divergence

“KL-Means algorithm” replace Euclidean distance with Kullback-Leiber Divergence
KL-Means Algorithm

“KL-Means algorithm” replace Euclidean distance with KL distance

Min KL divergence = max. mutual information

\[ Q^* = \arg \max_Q I(X; Z) = \arg \min_Q E(D(U||V)) \]

Numerical results show tradeoff:
- increasing number of quantizer outputs
- decreases the loss of mutual information

KL-Means = Information Bottleneck Method


The information bottleneck method

N Tishby, FC Pereira, W Bialek - arXiv preprint ph
Abstract: We define the relevant information in a set of images, and show that this signal provides about another signal $Y$ that face images provide about the names of the people.

Cited by 1257 Related articles All 41 versions

$$\min_{p_{Z|Y}(z|y)} I(Y; Z) - \beta I(X; Z)$$

The information bottleneck and KL-means algorithms both try to solve:

$$\max_Q I(X; Z)$$

When $\beta \to \infty$ the two algorithms are equivalent.

Hardware-Aware Information Theory

- Max-LUT method — Mutual-information maximizing lookup tables
- Application of Max-LUT to LDPC decoding
- Numerical results: 4 bits/message “performs like floating point”
Max-LUT Method

Max-LUT is a method for implementing the node decoder functions for graph-based decoders, using lookup tables that maximize mutual information.
Characteristics of the Max-LUT Method

- We need a factor graph
- We need an input distribution
- Factor graph messages are discrete
- Decoding functions are look up tables (LUT)
- Lookup tables are designed to maximize mutual information

- goal is investigating the resolution/performance tradeoff
Lookup Table (LUT) Implementation

Assume that LUTs are easy to implement in VLSI hardware
Max-LUT Method: Central Idea

Encoder Side. Code symbols $X$

- Check node $f : X_3 = X_1 + X_2$
- Var node $f : X_1 = X_2 = X_3$
- etc.

Decoder Side

- $L_i$ is a noisy version of $X_i$
- $Z$ is a noisy version of $X_3$

Choose LUT to maximize mutual information

$$\max_{\text{LUT}} I(X_3; Z) = \max I(X_3; \text{LUT}(L_1, L_2))$$
Max-LUT Method: Three Steps

- Step 1: Find joint distribution from marginal distributions
- Step 2: Quantize joint distribution maximize mutual information
- Step 3: Find LUT from the quantizer

Example

- LDPC variable node, two inputs $L_1, L_2$ with $Pr(L_i|X_i)$
- local constraint: “$x_1 = x_2 = x_3$”
- Goal: find max-MI lookup table $Z = LUT(L_1, L_2)$
Max-LUT Step 1: Joint Distribution

- Step 1: Construct joint distribution

\[
\begin{align*}
\Pr(L_1 | X_1) & \quad \Pr(L_2 | X_2) \\
L_1 & \in \{1, 2, 3, 4, 5, 6\}
\end{align*}
\]

\[
\begin{array}{c c c}
X_1 & L_1 & X_2 & L_2 & X_3 \\
1 & 6 & 1 & 4 & 1 \\
0 & 5 & 0 & 3 & 1 \\
0 & 4 & 0 & 2 & 0 \\
0 & 3 & 0 & 1 & 0 \\
0 & 2 & 0 & 1 & 0
\end{array}
\]

\[
\Pr(L_1, L_2 | X_3)
\]

local constraint:

\[
f(x_1, x_2) = x_3
\]
Max-LUT Step 2: Quantize

- Too many levels! Reduce to $\mathbb{Z}$ with $K$ levels
- Quantizer is a mapping from $(L_1, L_2)$ to $\mathbb{Z}$

$$Q : \mathcal{L}_1 \times \mathcal{L}_2 \rightarrow \mathbb{Z}$$

$$\max_Q I(X_3; Z)$$

$$K = 5$$
Max-LUT Step 3: Lookup Table

Lookup table:

\[ Z = \text{LUT}(L_1, L_2) \]
Max-LUT Method in One Slide

Variable node

Conditional distribution

Joint distribution

Quantizer

Lookup table (mapping)
Application to LDPC Code Decoding

- How to obtain the densities needed by Max-LUT method?
  > Density evolution

- How to keep the lookup table reasonable size?
  > Node decomposition or “opening the node”

- How does it perform numerically?
  > Similar to BP with four bits/message
Density Evolution Unwraps the Graph

Apply Max-LUT:
- construct lookup table
- get next density $\Pr(L|X)$

iter 1 — iter 2 — iter ...

(from channel (discrete output))

$\Pr(R|X)$ $\Pr(L|X)$ $\Pr(R|X)$ $\Pr(L|X)$ $\Pr(R|X)$ $\Pr(L|X)$
Noise Thresholds with Quantization

Levels of Channel Quantization

Figure 8. BER and WER results for the LUT decoding algorithm. $d_v = 4$, $d_c = 9$, $R = 0.56$, $N = 4113$, Max. Iter. = 30, Array code [2].

BSC: Lower Error Floor than Sum-Product  
But not lower than FAIDs

\[ N = 2388, (d_v = 3, d_c = 12), R = 0.75 \text{ and Max. iter } = 60 \]

FAIDs are designed to avoid the effects of harmful subgraphs, lowering the error floor Planjery et al. (2013).

The proposed decoding mapping functions can be used in a variety of channels not only in the BSC.
The proposed decoding mapping functions

- using 10 iterations can achieve the same BER performance than full SPA using 30 iterations.
- using 30 iterations can surpass the BER performance of full SPA using 30 iterations.

This code is used in IEEE 802.3an 10GBase-T standard producing an operation of 10 Gb/s.
Conclusion: Hardware-Aware Information Theory

Machine learning

• Overlap between machine learning and information theory
• Variation of K-means clustering for quantization
• Belief-propagation algorithm is widely used for decoding LDPC codes

Hardware-Aware

• Main goal is investigating the resolution/performance tradeoff
• Assume lookup table can be easily implemented in VLSI

Communication Receivers

• Not just LDPC decoders, but applicable to equalization, detection, etc.
• Can deal with arbitrary noise models