Inventing Communication Algorithms via Deep Learning

Pramod Viswanath

University of Illinois
Deep Learning is part of daily life.

Machine Translation (e.g. “Hello world!” in Chinese?)

natural language processing

Computer Vision
Model Complexity

- Models are complicated
  - for both NLP and CV

- Data is hard to model
  - Inference problems hard to model

- Deep learning learns efficient models
  - inherently empirical and human-validated
Algorithmic Complexity

- In many problems models are simple
  - unlimited training data
  - mathematical performance metrics

- Challenge: space of algorithms very large

- Success of Deep learning: AlphaZero
  - Chess, Go, Protein Folding
Communication Algorithms

- Simple models: AWGN channel
  - unlimited training data
  - precise performance metrics

- Challenge: space of algorithms very large

- Information Theory, Communication Theory, Coding Theory
AWGN Channel

- Sporadic Progress:
  - individual human ingenuity

- Huge practical impact
Vision

- Discovery of codes
  - human eureka moments

- Automate Progress:
  - use deep learning to search for codes
Two Goals

• New (deep learning) tools for classical problems
  • New state of the art
  • Inherent practical value

• Insight into deep learning methods
  • No overfitting
  • Interpretability
One Lens

• Scalability

• Train on small settings

• Test on much larger (100x) settings
Codes

• Encoders and Decoders
  • end-to-end training
  • gradients have to pass through decoder

• Structure is essential
  • traditional: linearity
  • neural networks are nonlinear
Two Deep Learning Components

- Recurrent Neural Networks
  - in-built recursion capability

- Gated recurrent units
  - GRU, LSTM, Attention
Inventing Codes

- **AWGN channel**
  - Very well studied; high bar
  - Already close to information theoretic limits

- **New codes**
  - IP protection
  - robust/adaptive data driven decoders
  - Scientific curiosity
Learning Approach

\[ \text{Neural network encoder} \xrightarrow{\text{b}} \text{AWGN channel} \xrightarrow{x} \text{Neural network decoder} \xrightarrow{\hat{b}} \text{b} \]
Code Structure

- **Linear codes**
  - Coding+modulation

- **Neural Networks**
  - Directly map bits to real valued outputs
  - nonlinear
  - Still need a structure
Reed Muller Codes

- Classical
  - Muller, 1954
  - Efficient decoder by Reed, 1954

- Recent Interest
  - Polar codes
  - RM codes are capacity achieving (proved for BEC)
RM Codes: Algebraic Construction

- RM \((m,r)\)

- Codeword is the evaluation of a polynomial of degree utmost \(r\) on the vertices of \(m\)-dimensional binary hypercube

- \(\text{RM}(m,0)\) is simply the repetition code

- \(2^m\) \(m\)-dimensional codeword
Plotkin construction

New Code

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

Code 1

\[v \in \{0, 1\}^n\]

Code 2

\[u \in \{0, 1\}^n\]
RM Codes via Plotkin construction

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

\[v \in \{0, 1\}^n \quad u \in \{0, 1\}^n\]

RM(2,1)  \quad RM(1,0)  \quad RM(1,1)
Neural Plotkin Construction

New Code

\[(u, g(u, v, u \oplus v)) \in \{0, 1\}^{2n}\]

\[g: \text{neural network}\]

\[v \in \{0, 1\}^n\]
\[u \in \{0, 1\}^n\]

Code 1

Code 2
Dumer Decoding

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

First Decode \(\mathcal{V}\)
LLR\(_u\), LLR\(_{u \oplus v}\)

Next Decode \(\mathcal{U}\)

\[v \in \{0, 1\}^n \quad u \in \{0, 1\}^n\]

Dumer, 2004-06
Neural Dumer Decoding

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

\[v \in \{0, 1\}^n \quad u \in \{0, 1\}^n\]

First Decode \(\mathcal{V}\)

LLR_u, LLR_{u\oplus v}

Neural network

Next Decode \(\mathcal{U}\)

Dumer, 2004-06
Neural Plotkin-Dumer Codes

- Generalize Plotkin construction via neural networks
- Generalize Dumer decoding via neural networks
Neural Plotkin-Dumer Codes

"Learn"

Neural Plotkin encoder

AWGN channel

Neural successive decoder

b → x → y → ̂b
RM Codes (4,1)

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

\[\forall v \in \{0, 1\}^n, u \in \{0, 1\}^n\]

RM (4,1)

Information Symbols = 5

Coded Symbols = 16
RM vs Neural Plotkin-Dumer Codes

![Graph showing BER vs SNR (dB) for RM + Dumer and Neural RM + Neural Dumer codes.](image-url)
RM Codes (8,1)

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

- \(v \in \{0, 1\}^n\)
- \(u \in \{0, 1\}^n\)

RM (8,0) \rightarrow RM (8,1)

Information Symbols = 9
Coded Symbols = 256
RM vs Neural Plotkin-Dumer Codes

![Graph showing BER vs SNR for RM + Dumer and Neural RM + Neural Dumer](image-url)
Pairwise Codeword Distances

- Neural Code: $d_{\text{min}}=11.00$
- RM Code: $d_{\text{min}}=22.63$
- Random Gaussian Code: $d_{\text{min}}=18.60$
Reed (ML) Decoding

\[(u, u \oplus v) \in \{0, 1\}^{2n}\]

\[v \in \{0, 1\}^n \quad u \in \{0, 1\}^n\]

Fast Hadamard Transform

Efficient, first order RM codes

Reed, 1954
Neural Plotkin Codes

"Learn"

Neural Plotkin encoder

x

AWGN channel

y

Neural ML decoder

^b

b

"Learn"
RM (6,1) vs Neural Plotkin Codes
Pairwise Codeword Distances

- Random Gaussian Code: $d_{\text{min}}=8.60$
- Neural Code: $d_{\text{min}}=8.93$
- RM Code: $d_{\text{min}}=11.31$
Ongoing Work

- **Extension to longer block lengths**
- **Higher order RM codes**
  - Decoding gets complex
  - Long conjectured to be efficient
  - Abbe-Ye RPA decoding

- **Neural Polar codes**
  - Soft polarization
Long Block lengths: Learning to Decode

- Fix encoding
  - convolutional codes

- Deep Learning decoders
  - learn Viterbi and BCJR algorithms
  - dynamic programming

- Learning an Algorithm: strong generalization
  - across block lengths
  - across SNR
Sequential Encoding

- Fixed encoders
  - convolutional codes

- Optimal decoders
  - Viterbi (block error)
  - BCJR (bit error)
  - dynamic programming

- Formulaic and generalize readily
  - across block lengths
  - across SNR
Deep Sequential Decoding

- Neural network decoders
- Sequential decoders
  - Recurrent neural networks (RNN)
- Representation capability
  - can encode Viterbi/BCJR in principle
- Key question:
  - Can SGD learn the optimal rules?
Setting

• Supervised training:

• Rate 0.5 convolutional code

• Neural Network Architecture
  • Two layer Bi-GRU RNN; sigmoidal output
Training: Zoom in

- Training:
  - L2 loss function
  - Block length 100

- 10K training examples

- Choice of SNR:
  - training = test SNR?
  - a variety of SNRs during training?
Hardest Training Examples

SNR\_train vs Rate

Shannon limit
SGD Learns Viterbi and BCJR

Train: block length = 100
Test: block length = 100K

BER vs SNR graph
Decoding Turbo Codes

- Training:
  - L2 loss function
  - Block length 100
  - 10K training examples

- Retain iterative decoding structure

- Use neural convolutional decoders as modules
Decoding Turbo Codes

Train: block length = 1000
Test: block length = 100K

SNR vs BER graph

- Commpy 6 iteration
- LSTM 6 iter
Typical Error Analysis

- Standard Information Theoretic tool
  - nuanced understanding of decoders
  - Statistics of noise that cause most error

- Classical result for ML decoder:
  - dominant error due to large noise vector magnitude
  - not true for turbo decoder

- Finding: neural decoder similar to ML decoder
Robustness

Fixed Decoder; change noise to T-distribution
Adaptivity: Bursty Noise

Retrain decoder with bursty noise
Typical Error Analysis

- Feedback neural encoder/decoder:
  dominant error due to noise amplitude being large

- Robustness to non-Gaussian noise
Inventing Codes

- AWGN channel
  - very well studied; high bar

- **Network Information Theory**
  - AWGN channel with feedback
  - relay channel
  - interference channel
Communication with Feedback

- Joint encoding and decoding
- AWGN channel with feedback
  - noisy feedback
- Deep Learning methods
  - beat Schalkwijk-Kailath scheme
  - even with noiseless feedback
- Robustness to noisy feedback
  - generalization: block lengths; SNR
Key challenge: how to combine $b$ with feedback
Literature

• Noiseless feedback
  • Schalkwijk-Kailath, ’66
  • posterior matching
  • improved reliability

• Noisy feedback
  • Kim-Lapidoth-Weissman, ’07
  • Linear codes very bad
  • Negative result

• Opportunity to test deep learning approach
Sequential Neural Architecture

- Encoder and Decoder: RNN

- Several Innovations
  - systematic bits
  - parity bits use feedback
  - power allocation to bits
  - “correct” concatenations

- Training: end-to-end
Noiseless Feedback

- Rate 1/3, blocklength = 50
Noisy Feedback

BER

Feedback Noise
Generalization: Blocklength

BER vs SNR graph with two curves: one for Blocklength 50 and another for Blocklength 500.
Improved Error Exponents

BER

Blocklength

- Turbo 1/9
- Turbo 1/3 + N-Feedback 1/3
- Turbo 1/3 + N-Feedback 1/3 (10dB)
Properties of the Feedback Code

• Nonlinear convolutional code
  • Maps information bits directly to real numbers

• Dynamic memory
  • Feedback influences the memory

• Gated RNNs
  • Can capture long term and short term memory
Theoretical Agenda

- Gated Recurrent Neural Networks
  - Nonlinear dynamical systems
  - Switched linear systems

- Learning Theory meets Switched Dynamical Systems
  - Many open questions (AISTATS ‘19,’20, ICML ‘19)
  - Basic theoretical/mathematical value
Defense Against the Dark Arts

- deepcomm.github.io
  - Instructional material
  - Social networking
Collaborators

RM Codes: V. Jamali, X. Liu, A. Makkuva, H. Mahdavifar