End-to-End Learning of Optical Communication Systems: A Beginner's Guide

Christian Häger

Department of Electrical Engineering, Chalmers University of Technology, Sweden

ECOC 2022, Basel, Switzerland
September 21, 2022

FIBER-OPTIC COMMUNICATIONS RESEARCH CENTER

CHALMERS

Machine Learning in 2022

"teddy bears mixing sparkling chemicals as mad scientists in a steampunk style"

"Shiba Inu dog wearing a beret and black turtleneck"

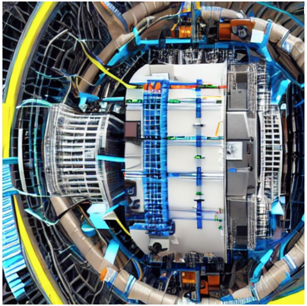
?

"Next-generation optical transmission system close to the Shannon limit"

Machine Learning in 2022

"teddy bears mixing sparkling chemicals as mad scientists in a steampunk style"

"Shiba Inu dog wearing a beret and black turtleneck"



"Next-generation optical transmission system close to the Shannon limit"

Acknowledgements

- This tutorial is based on work primarily done by our students:

Shen Li
Chalmers

Kadir Gümüș
TU/e

Jinxiang Song
Chalmers

Acknowledgements

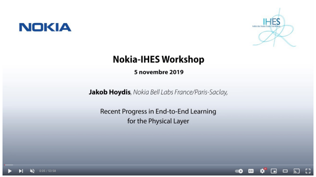
- This tutorial is based on work primarily done by our students:

Shen Li
Chalmers

Kadir Gümüş
TU/e

Jinxiang Song
Chalmers

- This tutorial is inspired by (and meant to be complementary to) two excellent existing tutorials:



youtube.com/watch?v=EPLJzsxReH4

youtube.com/watch?v=CnSqnlkKdJs

Agenda

Learning objectives

1. Introduction to basic topics:
   - What is the main idea behind end-to-end learning with simple examples
   - Main design elements: model selection, choice of loss function, and training paradigms
2. Overview of some more advanced topics:
   - How to estimate channel capacity with end-to-end learning
   - How to do end-to-end learning with multiple users

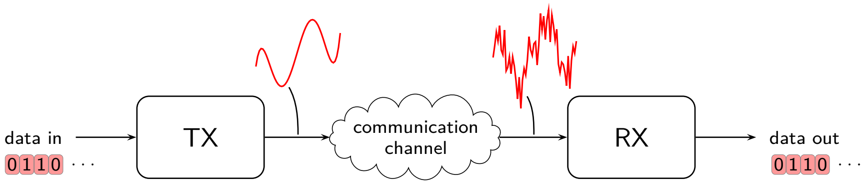
Outline

1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

Outline

1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

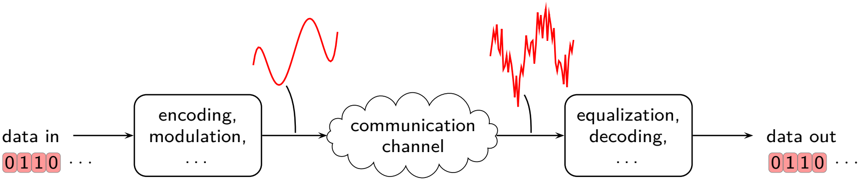
Introduction to End-to-End Learning



"*The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point*"

— Shannon, 1948

Introduction to End-to-End Learning

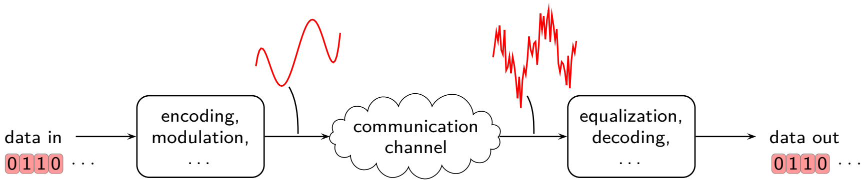


"*The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point*"

— Shannon, 1948

- Conventional design: handcrafted algorithms based on mathematical modeling

Introduction to End-to-End Learning

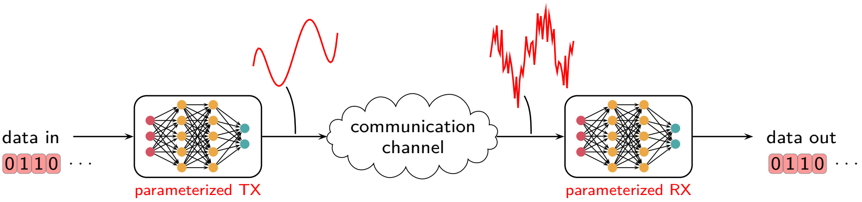


"*The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point*"

— Shannon, 1948

- Conventional design: handcrafted algorithms based on mathematical modeling
- Can we learn entire communication systems from scratch?

Introduction to End-to-End Learning



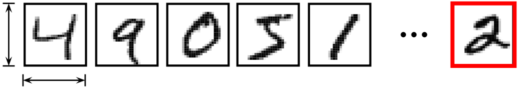
> "*The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point*"
>
> — Shannon, 1948

- Conventional design: handcrafted algorithms based on mathematical modeling
- Can we learn entire communication systems from scratch?
- Use function approximators (e.g., neural nets) and learn good parameter configurations from data
- This is similar to (denoising) autoencoders in machine learning

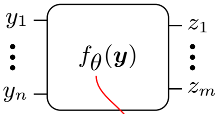
[O'Shea and Hoydis, 2017], An introduction to deep learning for the physical layer, (*IEEE Trans. Cogn. Commun. Netw.*)

Learning with Neural Nets (Quick Recap)

handwritten digit recognition (MNIST: 70,000 images)



28×28 pixels $\implies n = 784$



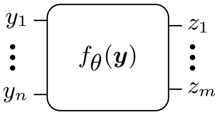
$\boldsymbol{z}$

0.01
0.92
0.01
0.00
0.00
0.01
0.00
0.04
0.01
0.01

parameters
to be optimized/learned

Learning with Neural Nets (Quick Recap)

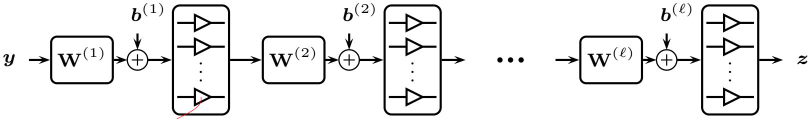
handwritten digit recognition (MNIST: 70,000 images)



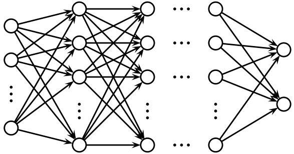
$\boldsymbol{z}$

0.01
0.92
0.01
0.00
0.00
0.01
0.00
0.04
0.01
0.01

How to choose $f_\theta(\boldsymbol{y})$? Deep feed-forward neural networks: universal function approximators



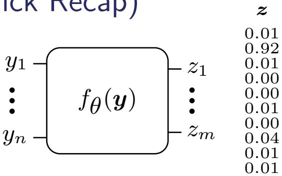
activation function



graph representation

Learning with Neural Nets (Quick Recap)

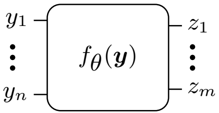
handwritten digit recognition (MNIST: 70,000 images)



How to optimize $\theta = \{\boldsymbol{W}^{(1)}, \dots, \boldsymbol{W}^{(\ell)}, \boldsymbol{b}^{(1)}, \dots, \boldsymbol{b}^{(\ell)}\}$?

Learning with Neural Nets (Quick Recap)

handwritten digit recognition (MNIST: 70,000 images)



$\boldsymbol{z}$	$\boldsymbol{x}$
0.01	0
0.92	1
0.01	0
0.00	0
0.00	0
0.01	0
0.00	0
0.04	0
0.01	0
0.01	0

How to optimize $\theta = \{\boldsymbol{W}^{(1)}, \dots, \boldsymbol{W}^{(\ell)}, \boldsymbol{b}^{(1)}, \dots, \boldsymbol{b}^{(\ell)}\}$?

Given a data set $\mathcal{D} = \{(\boldsymbol{y}^{(i)}, \boldsymbol{x}^{(i)})\}_{i=1}^{N}$, where $\boldsymbol{y}^{(i)}$ are model inputs and $\boldsymbol{x}^{(i)}$ are labels, we iteratively minimize

$$\frac{1}{|\mathcal{B}_k|} \sum_{(\boldsymbol{y},\boldsymbol{x}) \in \mathcal{B}_k} L(f_\theta(\boldsymbol{y}), \boldsymbol{x}) \triangleq \mathcal{L}(\theta) \qquad \text{using} \quad \theta_{k+1} = \theta_k - \lambda \nabla_\theta \mathcal{L}(\theta_k)$$

stochastic gradient descent

- $\mathcal{B}_k \subset \mathcal{D}$ and $|\mathcal{B}_k| = B$ is called the batch (or minibatch) size
- λ is called the step size or learning rate
- How to compute the gradients? TensorFlow, PyTorch, etc.

When To Use Machine Learning?

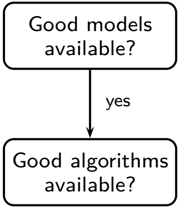
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?

Good models available?

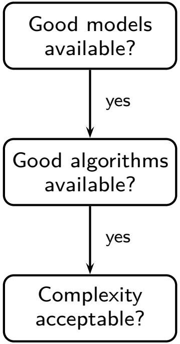
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



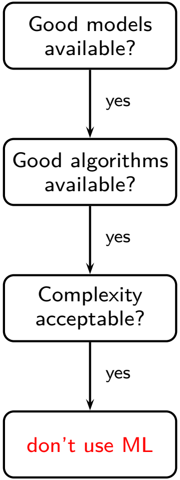
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



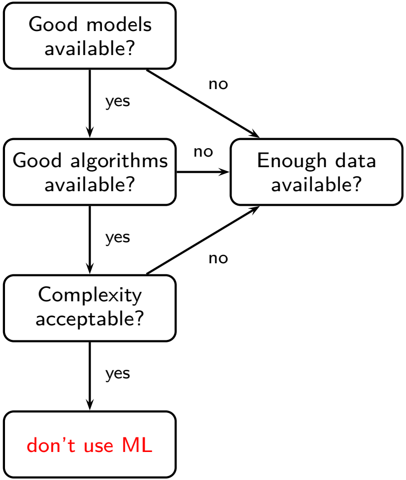
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



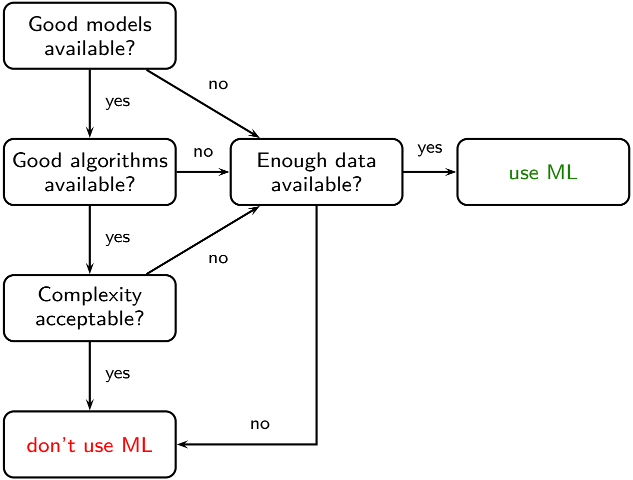
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



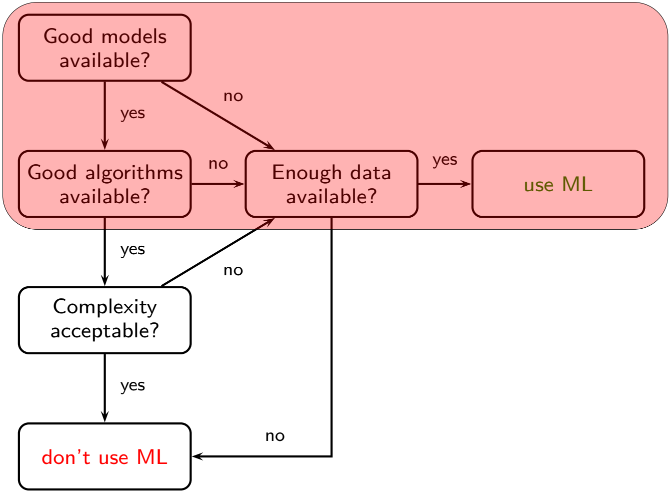
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

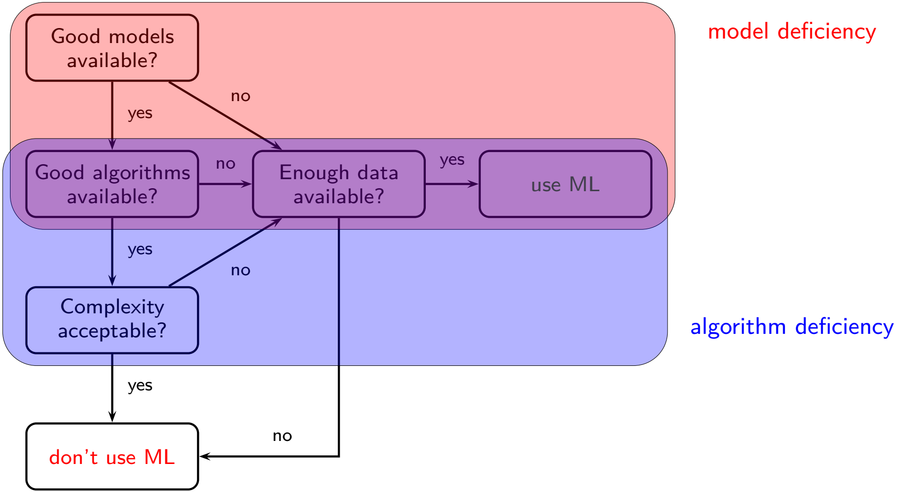
When To Use Machine Learning?



model deficiency

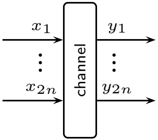
adapted from https://www.youtube.com/watch?v=EPLJzsxReH4

When To Use Machine Learning?



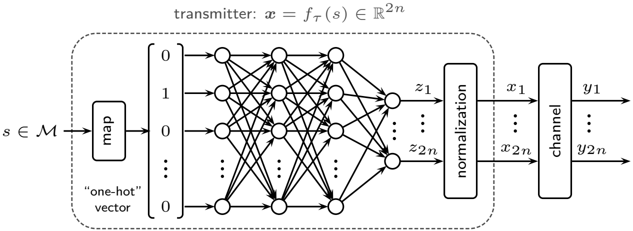
adapted from `https://www.youtube.com/watch?v=EPLJzsxReH4`

End-to-End Learning Example



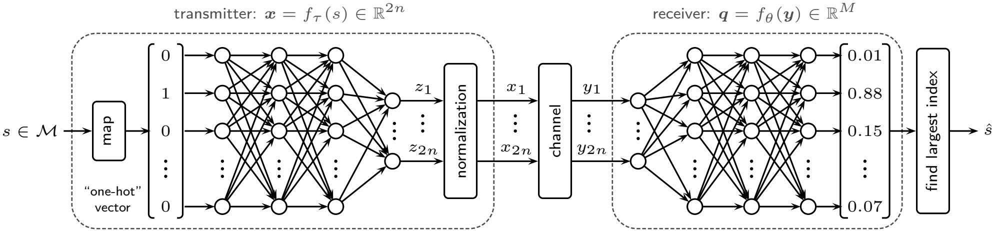
- Blockwise memoryless channel: input $\boldsymbol{x} = (x_1, \dots, x_{2n})^\intercal$, output $\boldsymbol{y} = (y_1, \dots, y_{2n})^\intercal$

End-to-End Learning Example



- Blockwise memoryless channel: input $\boldsymbol{x} = (x_1, \dots, x_{2n})^\intercal$, output $\boldsymbol{y} = (y_1, \dots, y_{2n})^\intercal$
- AE(n, k): map $M = 2^k$ messages to n complex-valued channel uses ($R = k/n$)
- Normalization (over the batch) such that $\frac{1}{B}\sum_i \|\boldsymbol{x}^{(i)}\|^2 = n$, where B is the batch size

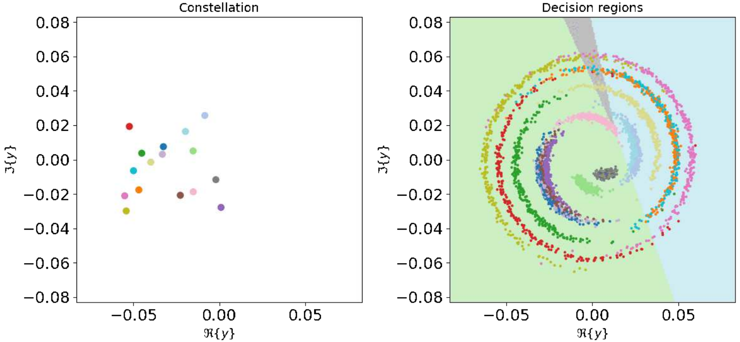
End-to-End Learning Example



- Blockwise memoryless channel: input $\boldsymbol{x} = (x_1, \dots, x_{2n})^\intercal$, output $\boldsymbol{y} = (y_1, \dots, y_{2n})^\intercal$
- AE(n, k): map $M = 2^k$ messages to n complex-valued channel uses ($R = k/n$)
- Normalization (over the batch) such that $\frac{1}{B}\sum_i \|\boldsymbol{x}^{(i)}\|^2 = n$, where B is the batch size
- Softmax layer $\implies$ final output can be interpreted as a probability distribution over the messages
- Receiver tries to learn the posterior $q_\theta(s|\boldsymbol{y}) \triangleq [\boldsymbol{q}]_s \approx f_{S|\boldsymbol{Y}}(s|\boldsymbol{y})$ using

$$\mathcal{L}(\tau, \theta) = -\frac{1}{B}\sum_{i=1}^{B} \log q_\theta(s^{(i)}|\boldsymbol{y}^{(i)}) \qquad \text{(cross-entropy loss)}$$

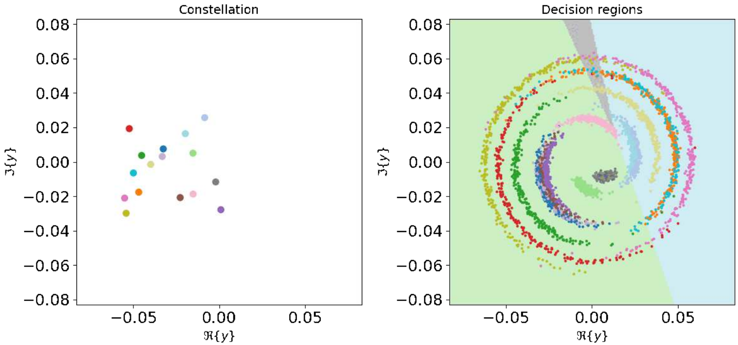
Optimization Results AE$(n = 1, k = 4)$



Nonlinear phase-noise channel (split-step method w/o dispersion):

$$X_{t+1} = X_t e^{\jmath \gamma L |X_t|^2 / K} + N_{t+1}, \quad 0 \leq t \leq K$$

Optimization Results AE$(n = 1, k = 4)$



Nonlinear phase-noise channel (split-step method w/o dispersion):

$$X_{t+1} = X_t e^{\jmath \gamma L |X_t|^2 / K} + N_{t+1}, \quad 0 \le t \le K$$

animation source code: `https://github.com/kit-cel/HHI_SummerSchool_2021` (see also `https://github.com/henkwymeersch/AutoencoderFiber`)

Summary

- The main idea behind end-to-end learning is to reinterpret the communication problem as a data-driven reconstruction task using fully parameterized transmitter–receiver pairs
- In principle allows us to learn optimal communication systems (neural net universality) from scratch (without domain knowledge) for any channel
- There are three main design components: (i) the (neural network) model, (ii) the loss function, and (iii) training method

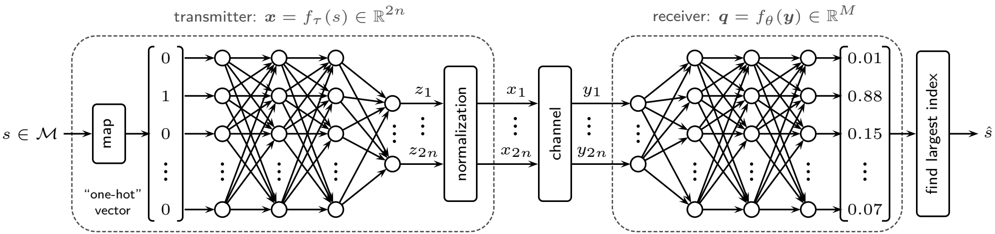
Further reading:

- [O'Shea and Hoydis, 2017], "An introduction to deep learning for the physical layer"
- [Dörner et al., 2018], "Deep Learning-Based Communication Over the Air"
- [Li et al., 2018], "Achievable information rates for nonlinear fiber communication via end-to-end autoencoder learning"
- [Karanov et al., 2018], "End-to-end deep learning of optical fiber communications"
- [Jones et al., 2018], "Deep learning of geometric constellation shaping including fiber nonlinearities"
- [Karanov et al., 2019], "End-to-End Optimized Transmission over Dispersive Intensity-Modulated Channels Using Bidirectional Recurrent Neural Networks"
- [Uhlemann et al., 2020], "Deep-learning autoencoder for coherent and nonlinear optical communication"
- [Jovanovic et al., 2021], "End-to-end Learning of a Constellation Shape Robust to Variations in SNR and Laser Linewidth"
- ...

Outline

1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

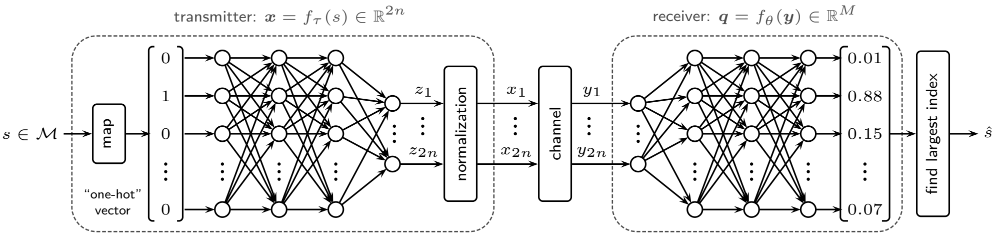
Loss Function Revisited



- Minimizing cross-entropy loss maximizes a lower bound on mutual information

$$\mathcal{L}(\tau, \theta) = -\frac{1}{B} \sum_{i=1}^{B} \log q_\theta(s^{(i)} | \boldsymbol{y}^{(i)})$$

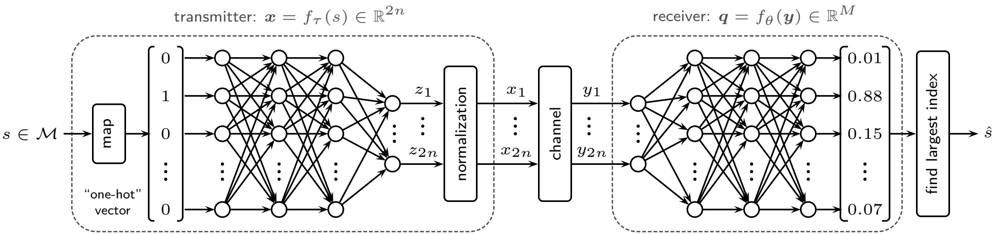
Loss Function Revisited



- Minimizing cross-entropy loss maximizes a lower bound on mutual information

$$\mathcal{L}(\tau,\theta) = -\frac{1}{B}\sum_{i=1}^{B} \log q_\theta(s^{(i)}|\boldsymbol{y}^{(i)}) + C \geq -\mathsf{MI} \quad (B \to \infty)$$

Loss Function Revisited

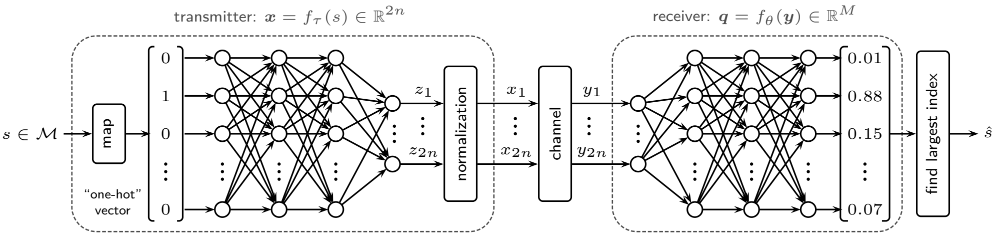


- Minimizing cross-entropy loss maximizes a lower bound on mutual information

$$\mathcal{L}(\tau,\theta) = -\frac{1}{B}\sum_{i=1}^{B} \log q_\theta(s^{(i)}|\boldsymbol{y}^{(i)}) + C \geq -\mathsf{MI} \quad (B \to \infty)$$

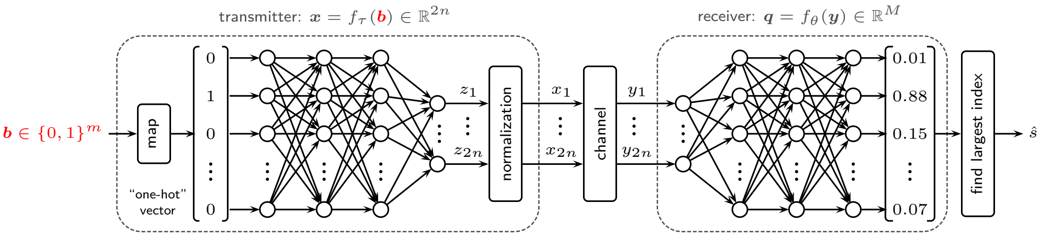
- However, we are typically more interested in bit-wise metrics, e.g., bit-error rate rather than symbol/message-error rate
- Binary interface desirable for compatibility with potential binary forward error correction (FEC)

Loss Function Revisited



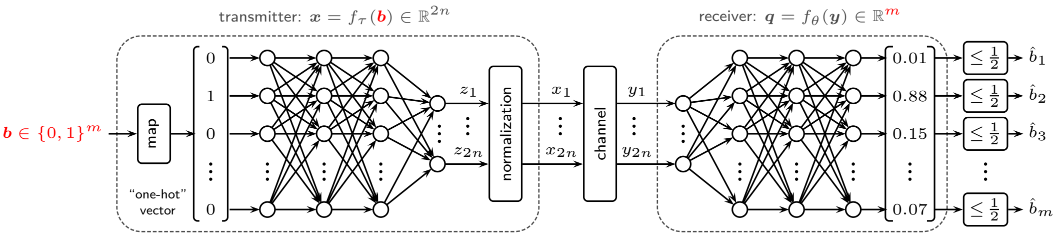
- Only two changes required:

Loss Function Revisited



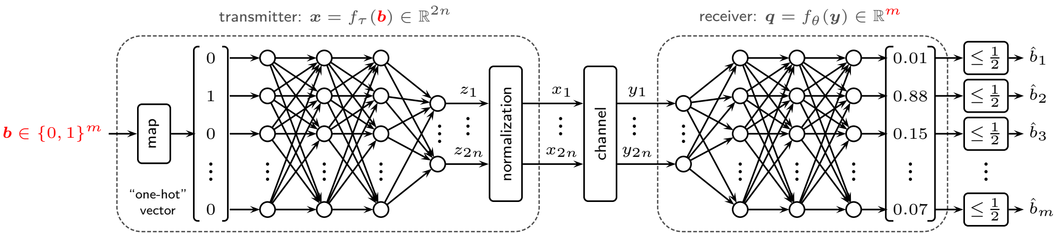
- Only two changes required:
  - Associate each message with a fixed binary label $\boldsymbol{b}$ of length $m = \log_2 M$

Loss Function Revisited



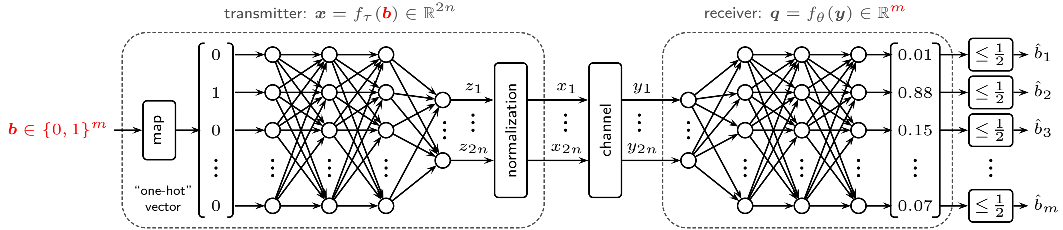
- Only two changes required:
  - Associate each message with a fixed binary label $\boldsymbol{b}$ of length $m = \log_2 M$
  - m output neurons with activation that maps to $[0, 1]$

Loss Function Revisited



- Only two changes required:
  - Associate each message with a fixed binary label $\boldsymbol{b}$ of length $m = \log_2 M$
  - m output neurons with activation that maps to $[0, 1]$
- Now the output can be interpreted as m probability distributions over the bits

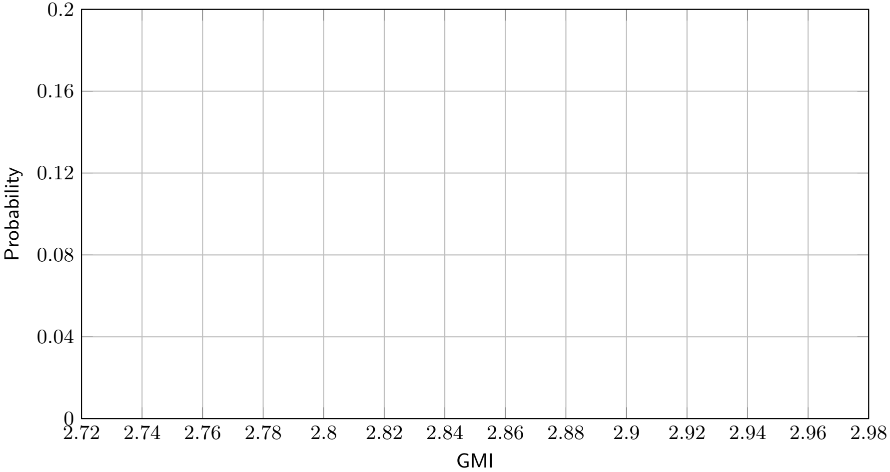
Loss Function Revisited



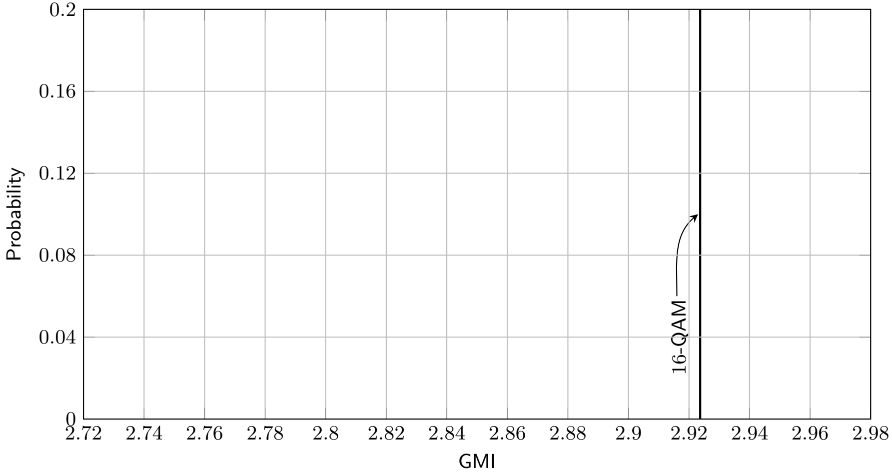
- Only two changes required:
  - Associate each message with a fixed binary label $\boldsymbol{b}$ of length $m = \log_2 M$
  - m output neurons with activation that maps to $[0,1]$
- Now the output can be interpreted as m probability distributions over the bits
- New loss based on the generalized (or bit-wise) mutual information (GMI):

$$\mathcal{L}(\tau,\theta) = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{m} \log q_\theta(b_j^{(i)}|\boldsymbol{y}^{(i)}) + C \geq -\text{GMI} \quad (B \to \infty)$$

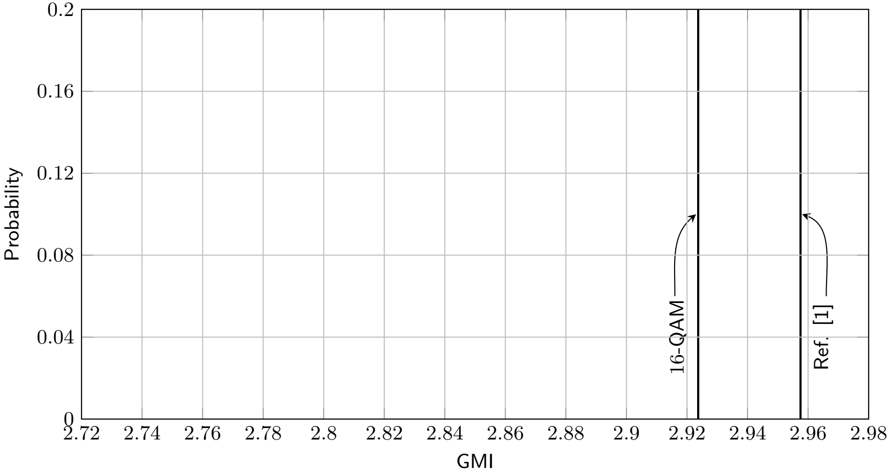
200 Training Runs for AE(1,4) on AWGN channel



200 Training Runs for AE(1,4) on AWGN channel

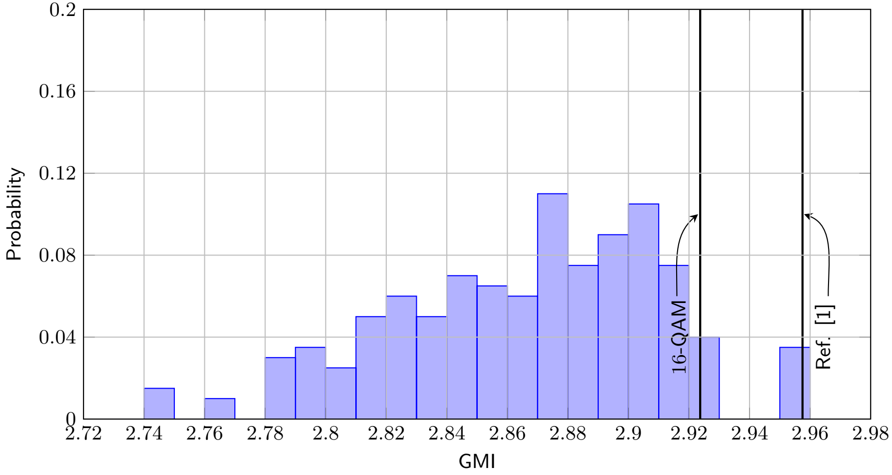


200 Training Runs for AE(1,4) on AWGN channel



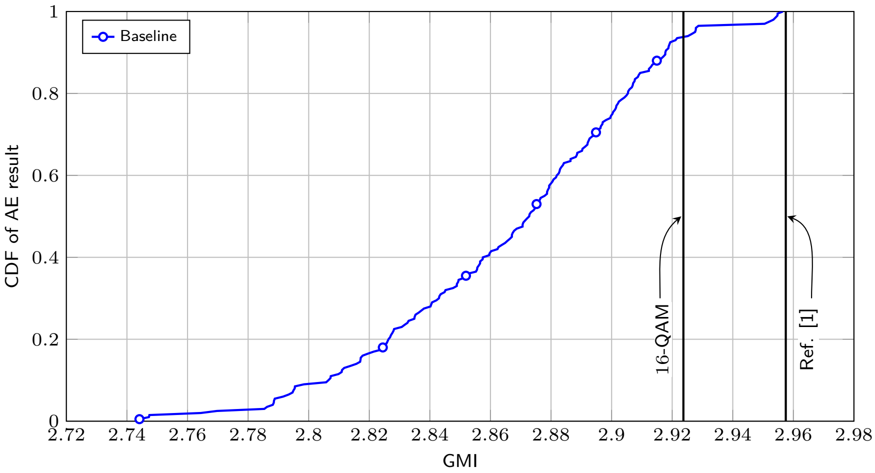
[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

200 Training Runs for AE(1,4) on AWGN channel



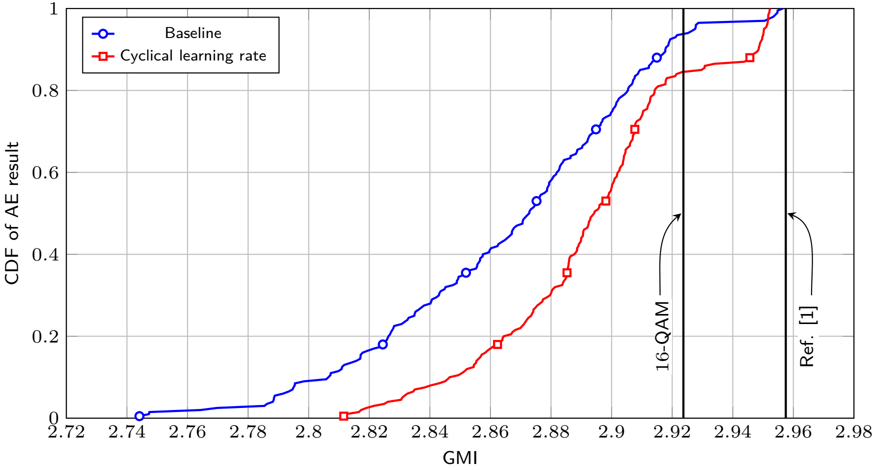
[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

200 Training Runs for AE(1,4) on AWGN channel



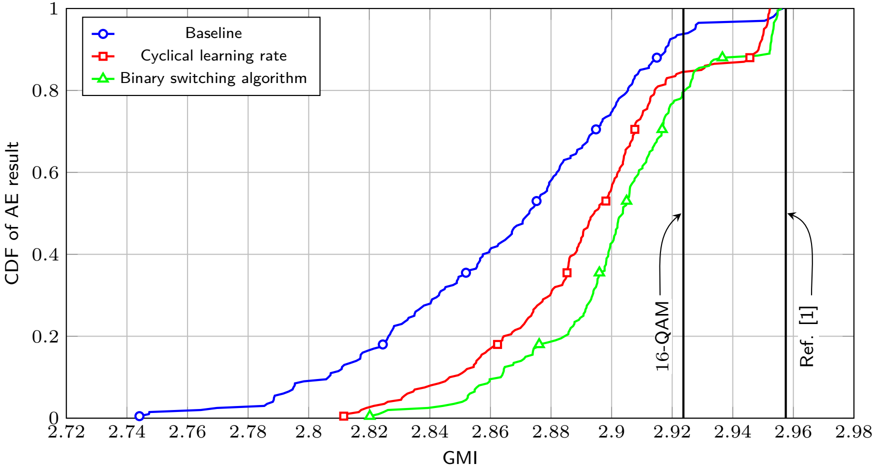
[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

200 Training Runs for AE(1,4) on AWGN channel



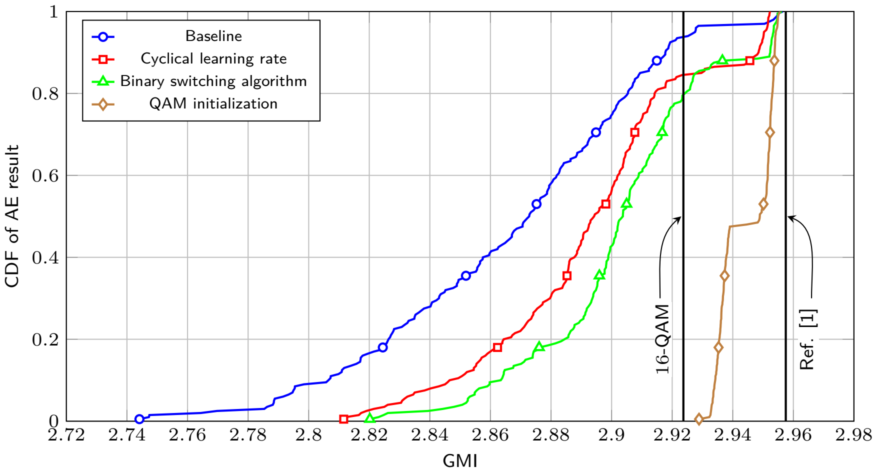
[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

200 Training Runs for AE(1,4) on AWGN channel



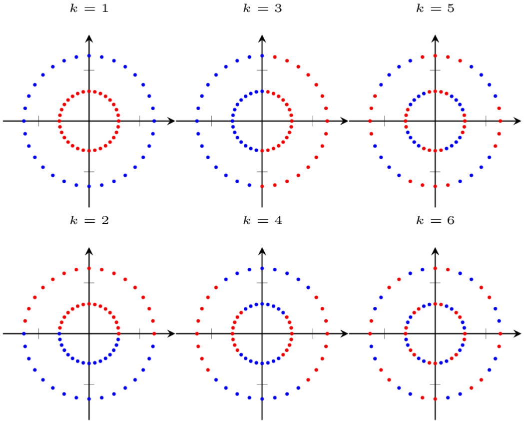
[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

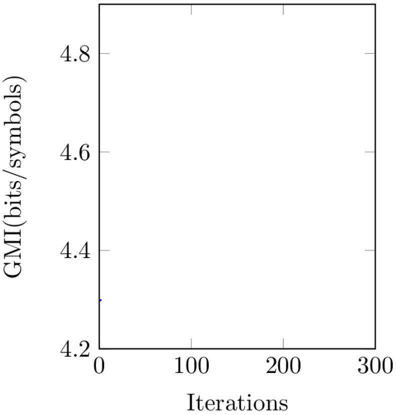
200 Training Runs for AE(1,4) on AWGN channel



[1] Bin Chen et al. "Increasing Achievable Information Rates via Geometric Shaping"

Optimization Behavior for AE(1,6)

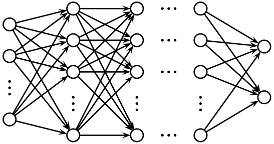




Neural Network Model Revisited

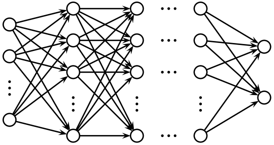
- Model initialization can be critical

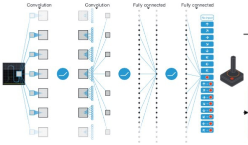
Neural Network Model Revisited

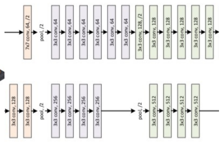


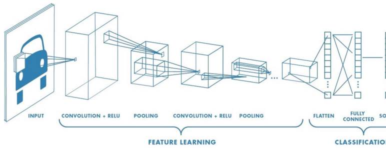
- Model initialization can be critical
- But how to choose the network architecture itself?
  - number of layers
  - number of neurons per layer
  - which activation function
  - ...

Neural Network Model Revisited





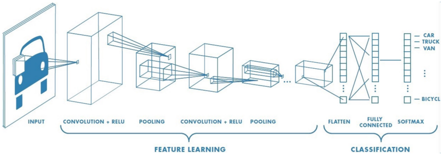


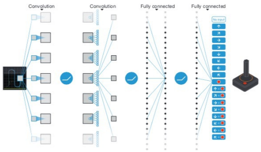


- Model initialization can be critical
- But how to choose the network architecture itself?
  - number of layers
  - number of neurons per layer
  - which activation function
  - ...
- Neural network zoo: feed-forward neural nets, recurrent neural nets (RNNs), convolutional neural nets (CNNs), long short-term memory (LSTM), residual networks (ResNets), Highway Nets, transformers, ...

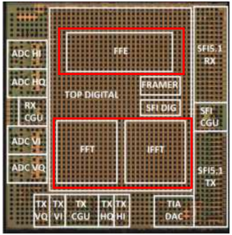
[Freire et al., 2021], Performance versus complexity study of neural network equalizers in coherent optical systems, (*JLT*)

How Feasible Are Large Neural Networks?

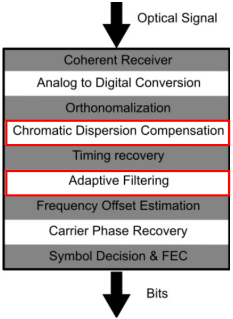




vs.

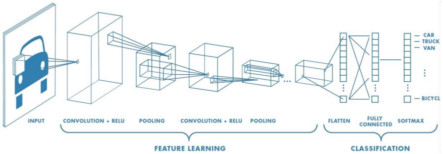


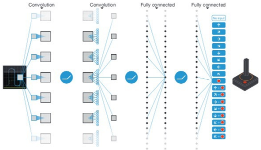
[Crivelli et al., 2014]



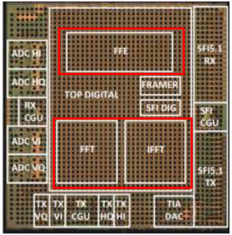
- Large neural networks can have millions of trainable parameters

How Feasible Are Large Neural Networks?

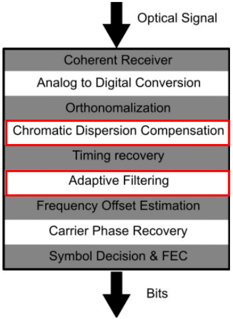




vs.

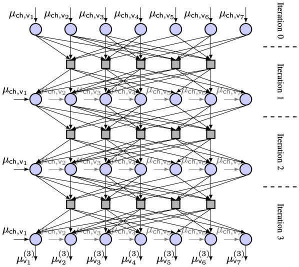


[Crivelli et al., 2014]



- Large neural networks can have millions of trainable parameters
- Significant challenge for real-time implementation at realistic throughputs
- Example (right figure): gradient-descent-based adaptive (linear) LMS equalizer (effectively a single-layer 1D-CNN with no activations) with only 256 trainable parameters

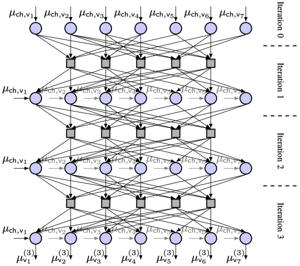
Exploiting Domain Knowledge

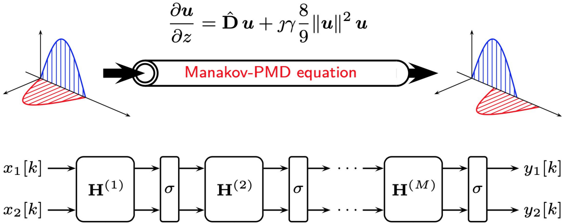


- Algorithm-inspired: Unrolling iterations from iterative algorithms, e.g, neural belief propagation

[Nachmani *et al.*, 2016], Learning to Decode Linear Codes Using Deep Learning, (*Proc. Allerton*)
[Lian *et al.*, 2018], Learned Belief-Propagation Decoding with Simple Scaling and SNR Adaptation, (*Proc. ESSCIRC*)
[Buchberger *et al.*, 2021], Pruning and Quantizing Neural Belief Propagation Decoders , (*IEEE JSAC*)

Exploiting Domain Knowledge

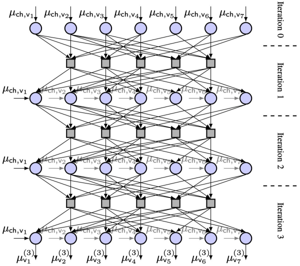


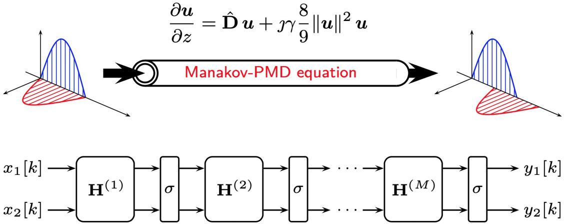


- Algorithm-inspired: Unrolling iterations from iterative algorithms, e.g, neural belief propagation
- Physics-based: parameterize numerical "split-step" methods to solve differential equations

[Häger & Pfister, 2018], Nonlinear Interference Mitigation via Deep Neural Networks, *(OFC)*
[Häger & Pfister, 2021], Physics-Based Deep Learning for Fiber-Optic Communication Systems, *IEEE J. Sel. Areas Commun.*

Exploiting Domain Knowledge



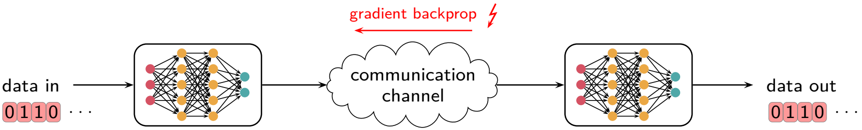


- Algorithm-inspired: Unrolling iterations from iterative algorithms, e.g, neural belief propagation
- Physics-based: parameterize numerical "split-step" methods to solve differential equations
- Such approaches are application-tailored (less universal), but can have significantly fewer parameters, need less training data, and be more amenable to hardware implementation

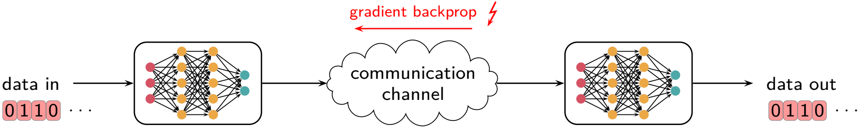
[Häger & Pfister, 2018], Nonlinear Interference Mitigation via Deep Neural Networks, *(OFC)*
[Häger & Pfister, 2021], Physics-Based Deep Learning for Fiber-Optic Communication Systems, *IEEE J. Sel. Areas Commun.*

Training Revisited

Training Revisited



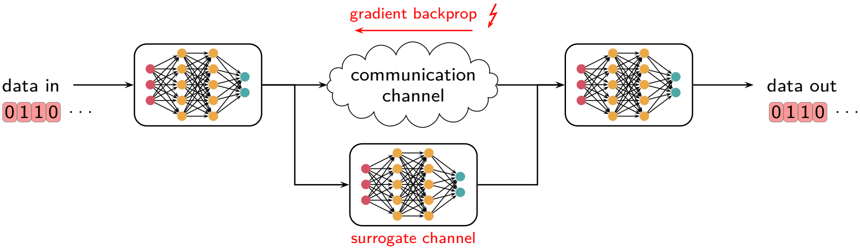
Training Revisited



- Solution 1: Pretrain with simple models and finetune the receiver

[Dörner et al., 2018], Deep Learning-Based Communication Over the Air, (*JSAC*)

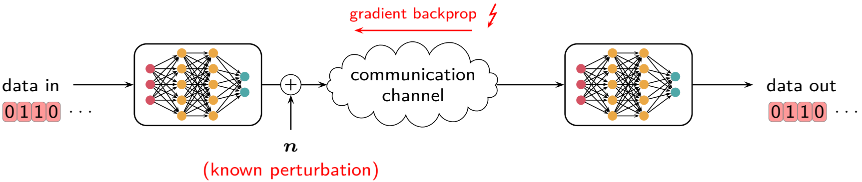
Training Revisited



- Solution 1: Pretrain with simple models and finetune the receiver
- Solution 2: Train surrogate channels

[O'Shea et al., 2018], Approximating the void: Learning stochastic channel models from observation with variational GANs, (*arXiv*)
[Ye et al., 2018], Channel agnostic end-to-end learning based communication systems with conditional GAN, (*arXiv*)
[Wang et al., 2020], Data-driven optical fiber channel modeling: A deep learning approach , (*arXiv*)
. . .
[Srinivasan et al., 2022], Learning Optimal PAM Levels for VCSEL-Based Optical Interconnects , (*ECOC 2022*)

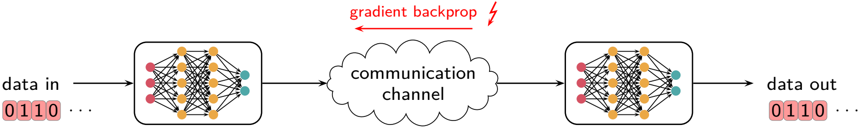
Training Revisited



- Solution 1: Pretrain with simple models and finetune the receiver
- Solution 2: Train surrogate channels
- Solution 3: Stochastic transmitters + reinforcement learning = surrogate gradients

[Aoudia & Hoydis, 2019], Model-Free Training of End-to-End Communication Systems, (*JSAC*)
[Song et al., 2020], Learning Physical-Layer Communication with Quantized Feedback, (*TCOM*)

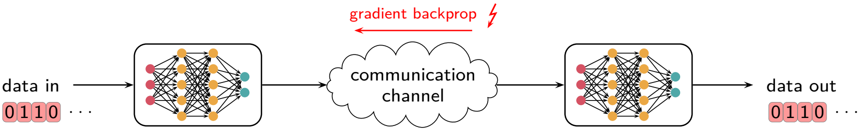
Training Revisited



- Solution 1: Pretrain with simple models and finetune the receiver
- Solution 2: Train surrogate channels
- Solution 3: Stochastic transmitters + reinforcement learning = surrogate gradients
- Solution 4: Kalman filtering approach

[Jovanovic et al., 2021], Gradient-Free Training of Autoencoders for Non-Differentiable Communication Channels, (*JLT*)

Training Revisited



- Solution 1: Pretrain with simple models and finetune the receiver
- Solution 2: Train surrogate channels
- Solution 3: Stochastic transmitters + reinforcement learning = surrogate gradients
- Solution 4: Kalman filtering approach
- Solution 5: MINE (mutual information neural estimation)

[Fritschek et al., 2019], Deep Learning for Channel Coding via Neural Mutual Information Estimation, (*SPAWC*)

Summary

- The loss function determines the optimization landscape which can heavily affect the convergence behavior
- The neural network can be pre-trained using domain knowledge; custom neural networks based on algorithm knowledge and/or physics
- Training in the absence of a differentiable channel model is possible using a variety of methods: surrogate models, reinforcement learning, . . .

Further Reading:

- [Jones et al., 2019], "End-to-end Learning for GMI Optimized Geometric Constellation Shape"
- [Gümüs et al., 2020], "End-to-End Learning of Geometrical Shaping Maximizing Generalized Mutual Information"
- [Cammerer et al., 2020], "Trainable Communication Systems: Concepts and Prototype"
- [Song et al., 2022a], "Model-Based End-to-End Learning for WDM Systems With Transceiver Hardware Impairments"
- [Aoudia and Hoydis, 2019], "Model-Free Training of End-to-End Communication Systems"
- [Song et al., 2020], "Learning Physical-Layer Communication with Quantized Feedback"
- [Jovanovic et al., 2021], "Gradient-Free Training of Autoencoders for Non-Differentiable Communication Channels"
- . . .

Outline

1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

Motivation

- The capacity of a (memoryless) channel with input $X \in \mathcal{X}$ and output $Y \in \mathcal{Y}$ is

$$C = \max_{f_X} I(X;Y)$$

- Analytical expressions for C are rare
- Numerical methods (e.g., Blahut–Arimoto) require knowledge about the channel law $f_{Y|X=x}(y)$

Motivation

- The capacity of a (memoryless) channel with input $X \in \mathcal{X}$ and output $Y \in \mathcal{Y}$ is

$$C = \max_{f_X} I(X;Y)$$

- Analytical expressions for C are rare
- Numerical methods (e.g., Blahut–Arimoto) require knowledge about the channel law $f_{Y|X=x}(y)$

Can we estimate capacity even if we don't know the precise channel law?

Assume we have access only to channel input–output samples $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots$

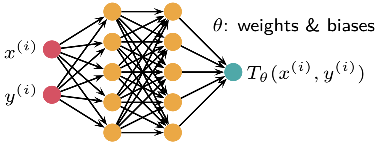
- Estimate mutual information (MI): notoriously difficult problem
- Find optimal input distribution: how to represent? how to optimize efficiently?

MINE: Mutual Information Neural Estimation

- Estimate $I(X;Y) = D(f_{X,Y}||f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$

[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

MINE: Mutual Information Neural Estimation

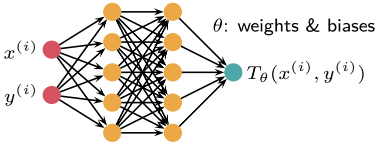


(param. function $T_\theta : \mathbb{R}^2 \to \mathbb{R}$)

- Estimate $I(X;Y) = D(f_{X,Y} || f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$

[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

MINE: Mutual Information Neural Estimation



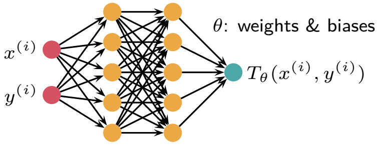
(param. function $T_\theta : \mathbb{R}^2 \to \mathbb{R}$)

- Estimate $I(X;Y) = D(f_{X,Y} || f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$:

$$\hat{I}_\theta = \frac{1}{B} \sum_{i=1}^{B} T_\theta(x^{(i)}, y^{(i)}) - \log \left(\frac{1}{B} \sum_{i=1}^{B} e^{T_\theta(\tilde{x}^{(i)}, \tilde{y}^{(i)})} \right)$$

[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

MINE: Mutual Information Neural Estimation



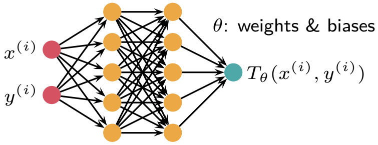
(param. function $T_\theta : \mathbb{R}^2 \to \mathbb{R}$)

- Estimate $I(X;Y) = D(f_{X,Y} || f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$:

$$\hat{I}_\theta = \frac{1}{B}\sum_{i=1}^{B} T_\theta(x^{(i)}, y^{(i)}) - \log\left(\frac{1}{B}\sum_{i=1}^{B} e^{T_\theta(\tilde{x}^{(i)}, \tilde{y}^{(i)})}\right) \lessapprox I(X;Y)$$

[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

MINE: Mutual Information Neural Estimation



(param. function $T_\theta : \mathbb{R}^2 \to \mathbb{R}$)

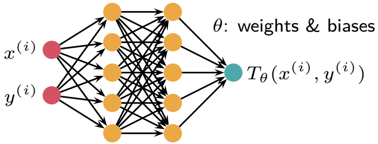
- Estimate $I(X;Y) = D(f_{X,Y} || f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$:

$$\hat{I}_\theta = \frac{1}{B} \sum_{i=1}^{B} T_\theta(x^{(i)}, y^{(i)}) - \log \left(\frac{1}{B} \sum_{i=1}^{B} e^{T_\theta(\tilde{x}^{(i)}, \tilde{y}^{(i)})} \right) \lessapprox I(X;Y)$$

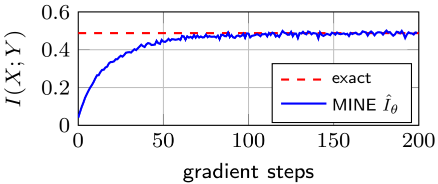
- Why? Donsker–Varadhan representation: $D(P||Q) = \sup_{T \in \mathcal{T}} \mathbb{E}_P[T] - \log \left(\mathbb{E}_Q \left[e^T \right] \right)$

[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

MINE: Mutual Information Neural Estimation



(param. function $T_\theta : \mathbb{R}^2 \to \mathbb{R}$)



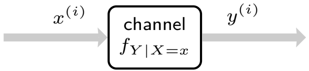
- Estimate $I(X;Y) = D(f_{X,Y} || f_X f_Y)$ from samples $(x^{(i)}, y^{(i)}) \sim f_{X,Y}$, $(\tilde{x}^{(i)}, \tilde{y}^{(i)}) \sim f_X f_Y$:

$$\hat{I}_\theta = \frac{1}{B} \sum_{i=1}^{B} T_\theta(x^{(i)}, y^{(i)}) - \log \left(\frac{1}{B} \sum_{i=1}^{B} e^{T_\theta(\tilde{x}^{(i)}, \tilde{y}^{(i)})} \right) \lessapprox I(X;Y)$$

- Why? Donsker–Varadhan representation: $D(P||Q) = \sup_{T \in \mathcal{T}} \mathbb{E}_P[T] - \log \left(\mathbb{E}_Q \left[e^T \right] \right)$
- Find best lower bound by optimizing θ using gradient ascent: $\theta \leftarrow \theta + \alpha \nabla_\theta \hat{I}_\theta$

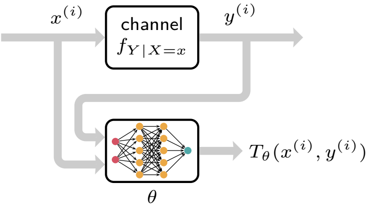
[Belghazi et al., 2018], Mutual information neural estimation, (*ICML*)

Estimating Capacity using MINE



$$C = \max_{f_X} I(X;Y)$$

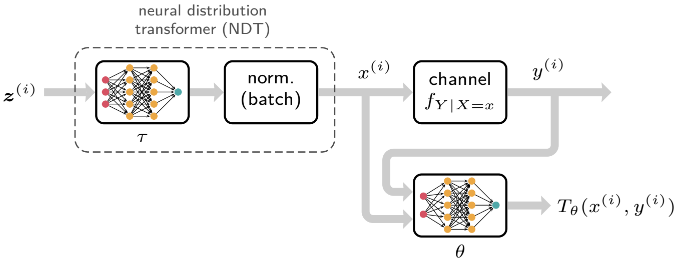
Estimating Capacity using MINE



$$C = \max_{f_X} I(X;Y)$$

- Assume we have a good MINE $\hat{I}_\theta$ based on T_θ. How to optimize f_X?

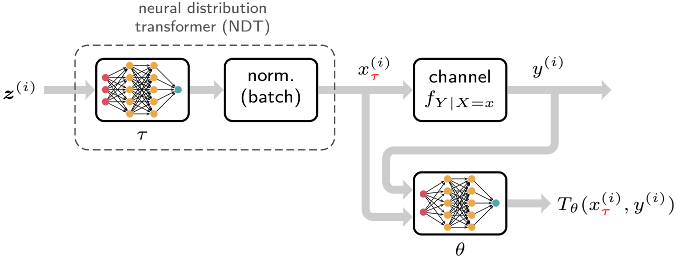
Estimating Capacity using MINE



$$C = \max_{f_X} I(X;Y)$$

- Assume we have a good MINE $\hat{I}_\theta$ based on T_θ. How to optimize f_X?
- Neural distribution transformer (NDT) with parameters τ
  - Transforms samples $z^{(i)} \in \mathbb{R}^l$ from a known (e.g., multivariate Gaussian) distribution
  - Normalization block over $i \in \{1, \dots, B\}$ enforces potential input (e.g., average-power) constraint

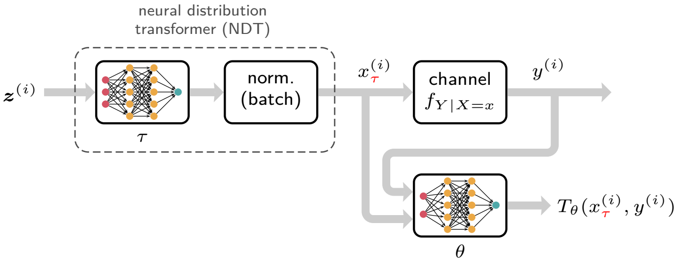
Estimating Capacity using MINE



$$C = \max_{f_X} I(X;Y)$$
$$\approx \max_{\theta,\tau} \hat{I}_{\theta,\tau}$$

- Assume we have a good MINE $\hat{I}_\theta$ based on T_θ. How to optimize f_X?
- Neural distribution transformer (NDT) with parameters τ
  - Transforms samples $z^{(i)} \in \mathbb{R}^l$ from a known (e.g., multivariate Gaussian) distribution
  - Normalization block over $i \in \{1, \ldots, B\}$ enforces potential input (e.g., average-power) constraint
- Input symbols are differentiable functions of τ. So is MINE $\hat{I}_{\theta,\tau}$!
- Can be used to train autoencoder transmitters as well

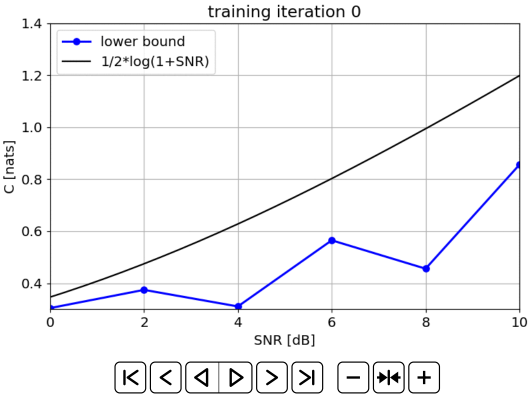
Estimating Capacity using MINE



$$C = \max_{f_X} I(X;Y)$$
$$\approx \max_{\theta,\tau} \hat{I}_{\theta,\tau}$$

- Assume we have a good MINE $\hat{I}_\theta$ based on T_θ. How to optimize f_X?
- Neural distribution transformer (NDT) with parameters τ
  - Transforms samples $z^{(i)} \in \mathbb{R}^l$ from a known (e.g., multivariate Gaussian) distribution
  - Normalization block over $i \in \{1, \dots, B\}$ enforces potential input (e.g., average-power) constraint
- Input symbols are differentiable functions of τ. So is MINE $\hat{I}_{\theta,\tau}$!
- Can be used to train autoencoder transmitters as well
- Capacity estimation: alternate between (gradient-based) training of τ (NDT) and θ (MINE)

Example: AWGN Channel



- Very simple network architectures, $B = 20000$ samples, Adam optimizer with step size 0.001

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

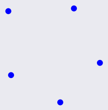
where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$



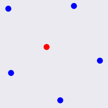
where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$



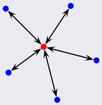
where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

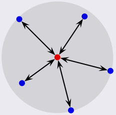


where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is



$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

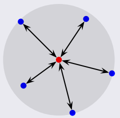
where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$



where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.

- Any fixed choice for q_Y gives an upper bound on capacity

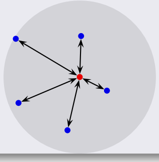
Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.



- Any fixed choice for q_Y gives an upper bound on capacity

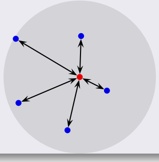
Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.



- Any fixed choice for q_Y gives an upper bound on capacity
- Ingredient 1: Train "MINE" to estimate the divergence terms $D(f_{Y|X=x}||q_Y) \approx \hat{D}_\theta$
  - Challenge: we would need a different estimator (and neural network) for each channel input x
  - Idea: provide x as an input to the network T_θ; this gives a parameterized estimator $\hat{D}_\theta(x)$

[Häger & Agrell, 2022], Data-Driven Estimation of Capacity Upper Bounds, (*Comm. Lett., to appear*)

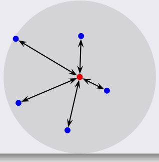
Upper Bounds via Duality

Duality formula [Csiszár and Körner, 1981, p. 142]

The capacity of a memoryless channel is

$$C = \min_{q_Y} \max_{x \in \mathcal{X}} D(f_{Y|X=x} || q_Y),$$

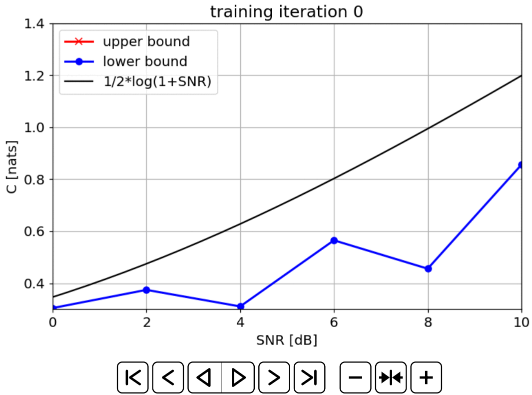
where q_Y ranges over distributions on the output alphabet $\mathcal{Y}$.



- Any fixed choice for q_Y gives an upper bound on capacity
- Ingredient 1: Train "MINE" to estimate the divergence terms $D(f_{Y|X=x}||q_Y) \approx \hat{D}_\theta$
  - Challenge: we would need a different estimator (and neural network) for each channel input x
  - Idea: provide x as an input to the network T_θ; this gives a parameterized estimator $\hat{D}_\theta(x)$
- Ingredient 2: Represent q_Y using an NDT and train by minimizing loss function $\max_{x \in \mathcal{X}} \hat{D}_\theta(x)$
  - Challenge: maximization over x for continuous-input channels
  - Solution: we resort to input-space discretization

[Häger & Agrell, 2022], Data-Driven Estimation of Capacity Upper Bounds, (*Comm. Lett., to appear*)

AWGN Channel



- Very simple network architectures, $B = 20000$ samples, Adam optimizer with step size 0.001

Summary

- End-to-end learning can be extended to provide data-driven estimates of lower and upper bounds on channel capacity
- Could potentially be interesting in optical scenarios where capacity is still unknown

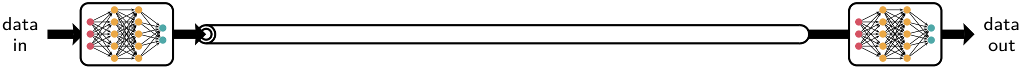
Further reading:

- [Fritschek et al., 2019], "Deep Learning for Channel Coding via Neural Mutual Information Estimation": use MINE to optimize AE transmitters
- [Aharoni et al., 2020], "Capacity of Continuous Channels with Memory via Directed Information Neural Estimator": propose DINE (directed information neural estimator) for channels with memory
- [Letizia and Tonello, 2021], "Capacity-Driven Autoencoders for Communications": MINE-regularized AE training
- [Mirkarimi and Farsad, 2021], "Neural Computation of Capacity Region of Memoryless Multiple Access Channels": consider memoryless multiple-access channels
- [Fritschek et al., 2020], "Neural Mutual Information Estimation for Channel Coding: State-of-the-Art Estimators, Analysis, and Performance Comparison"
- [Mirkarimi et al., 2021], "Neural Capacity Estimators: How reliable Are They?"
- [Häger and Agrell, 2022], "Data-Driven Estimation of Capacity Upper Bounds", IEEE Commun. Lett. (to appear), see `https://arxiv.org/abs/2205.06471` (source code: `https://github.com/chaeger/upper_capacity_bounds`)
- [Mirkarimi and Rini, 2022], "A Perspective on Neural Capacity Estimation: Viability and Reliability"
- [Tsur et al., 2022], "Neural Estimation and Optimization of Directed Information over Continuous Spaces"

Outline

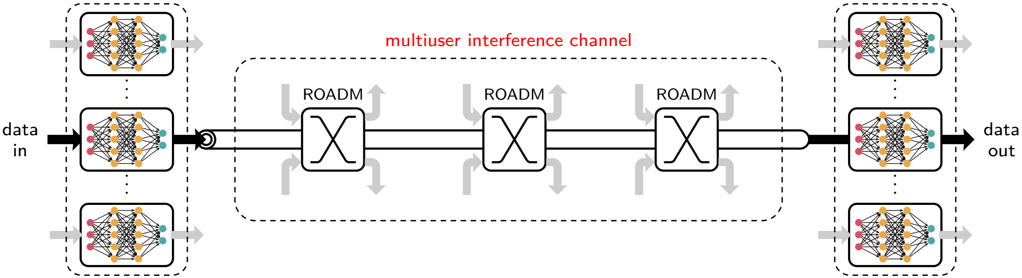
1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

Motivation



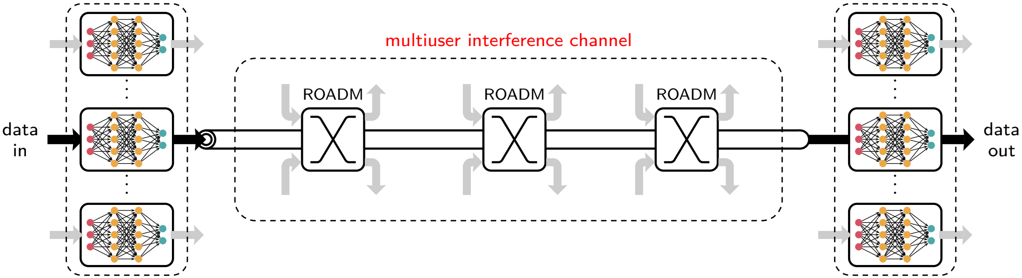
- Up till now: design and train a single transmitter–receiver pair

Motivation



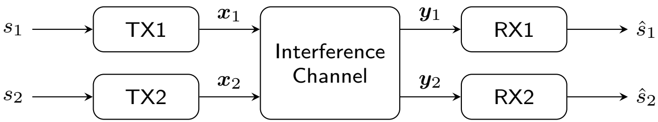
- Up till now: design and train a single transmitter–receiver pair
- Optical fiber is inherently a shared medium with potentially multiple WDM users

Motivation



- Up till now: design and train a single transmitter–receiver pair
- Optical fiber is inherently a shared medium with potentially multiple WDM users
- So far, relatively little work on multiuser end-to-end learning
- Probably not a coincidence: quite challenging to optimize and interpret the solutions (as we will see), but could potentially provide novel ways for dealing with nonlinear interference

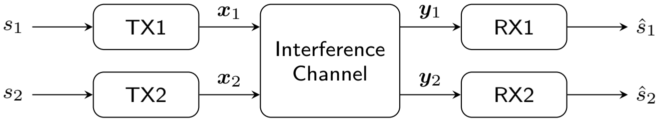
End-to-End Learning for Interference Channels



- To gain some insight, we consider a simple 2-user Gaussian interference channel similar to [O'Shea & Hoydis (2017)]:

$$\begin{aligned} \boldsymbol{y}_1 &= \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{n}_1, \\ \boldsymbol{y}_2 &= \boldsymbol{x}_2 + \boldsymbol{x}_1 + \boldsymbol{n}_2, \qquad \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathbb{C}^n \end{aligned}$$

End-to-End Learning for Interference Channels

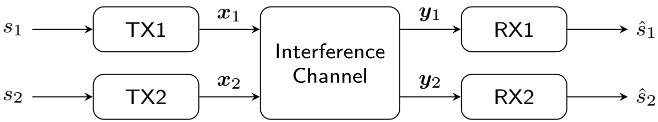


- To gain some insight, we consider a simple 2-user Gaussian interference channel similar to [O'Shea & Hoydis (2017)]:

$$\boldsymbol{y}_1 = \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{n}_1,$$
$$\boldsymbol{y}_2 = \boldsymbol{x}_2 + \boldsymbol{x}_1 + \boldsymbol{n}_2, \qquad \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathbb{C}^n$$

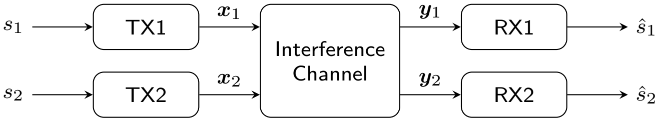
- AE(n, k): both users want to transmit 2^k messages over n complex-valued channel uses
- TX1, TX2, RX1, RX2 represented by fully-connected neural networks as before (including one-hot mapping, normalization, and softmax layers)

End-to-End Learning for Interference Channels



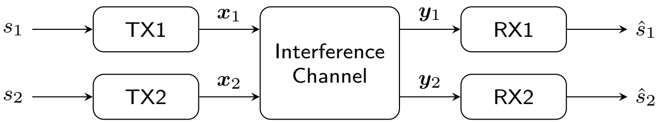
- Each user has a separate (cross-entropy) loss function defined by L_1 and L_2

End-to-End Learning for Interference Channels



- Each user has a separate (cross-entropy) loss function defined by L_1 and L_2
- Problem: Simply optimizing $L = L_1 + L_2$ does not work
- Unstable optimization dynamics based on initial conditions: one of the users tends to "dominate" the overall loss

End-to-End Learning for Interference Channels



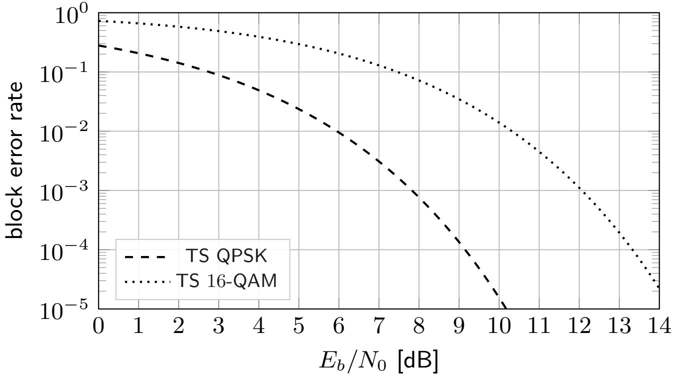
- Each user has a separate (cross-entropy) loss function defined by L_1 and L_2
- Problem: Simply optimizing $L = L_1 + L_2$ does not work
- Unstable optimization dynamics based on initial conditions: one of the users tends to "dominate" the overall loss
- Dynamic reweighting trick: loss function in iteration t is $L = \alpha_t L_1 + (1 - \alpha_t) L_2$, where

$$\alpha_t = \frac{L_1(\theta_{t-1})}{L_1(\theta_{t-1}) + L_2(\theta_{t-1})}$$

Optimization Details

- We consider AE(4,4) and AE(4,8), i.e., each user transmits 16 or 256 messages over 4 complex-valued channel uses
- Baseline: uncoded QAM $+$ time-sharing
- All neural networks have one hidden layer with $M = 2^k$ neurons and ReLU activation
- Fixed training SNR $E_b/N_0 = 7\,\text{dB}$ for AE(4,4) and $E_b/N_0 = 10\,\text{dB}$ for AE(4,8)
- Adam optimizer with learning rate $\gamma = 0.001$ and batch size $B = 10000$
- Number of training iterations: 20000

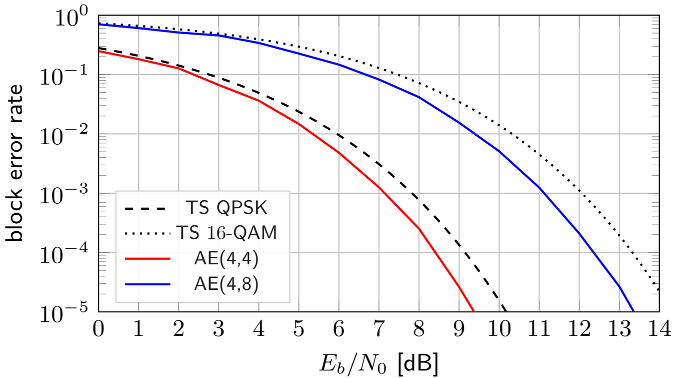
Optimization Results



user 1:	QPSK	0	QPSK	0	$\cdots$		16-QAM	0	16-QAM	0	$\cdots$
user 2:	0	QPSK	0	QPSK	$\cdots$		0	16-QAM	0	16-QAM	$\cdots$

$(R = 1)$ $(R = 2)$

Optimization Results



user 1:	QPSK	0	QPSK	0	$\cdots$
user 2:	0	QPSK	0	QPSK	$\cdots$

$(R = 1)$

16-QAM	0	16-QAM	0	$\cdots$
0	16-QAM	0	16-QAM	$\cdots$

$(R = 2)$

What Does The Autoencoder Learn?

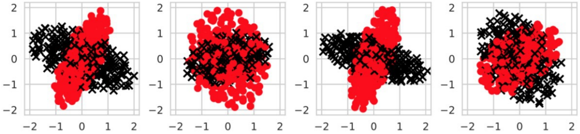


Fig.: Learned AE(4,8) constellation with 256 points (black: user 1, red: user 2)

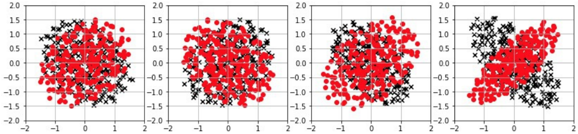
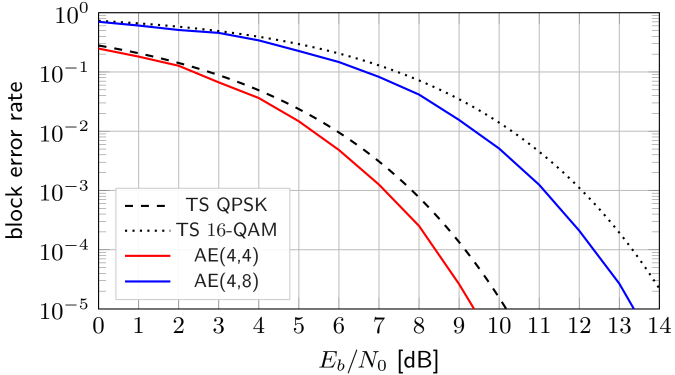


Fig.: Our results

“For (4, 4) and (4, 8), the constellations are more difficult to interpret, but we can see that the constellations of both transmitters resemble ellipses with orthogonal major axes and varying focal distances.” [O’Shea & Hoydis (2017)]

Optimization Results: Improved Baseline



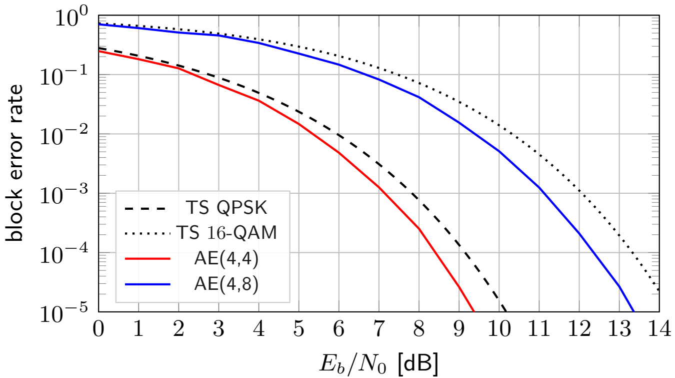
user 1:	QPSK	0	QPSK	0	$\cdots$
user 2:	0	QPSK	0	QPSK	$\cdots$

$(R = 1)$

16-QAM	0	16-QAM	0	$\cdots$
0	16-QAM	0	16-QAM	$\cdots$

$(R = 2)$

Optimization Results: Improved Baseline



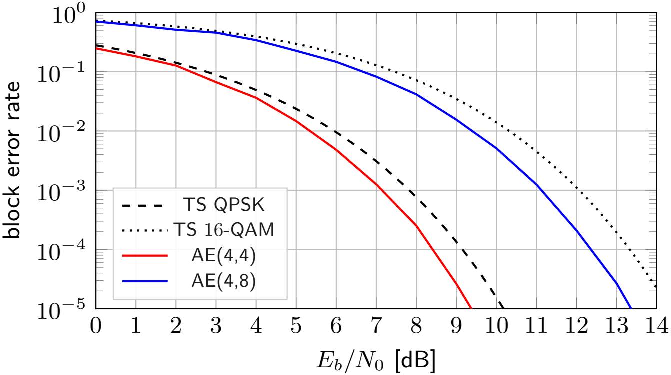
user 1:	QPSK	QPSK	0	0	$\cdots$
user 2:	0	0	QPSK	QPSK	$\cdots$

$(R = 1)$

user 1:	16-QAM	16-QAM	0	0	$\cdots$
user 2:	0	0	16-QAM	16-QAM	$\cdots$

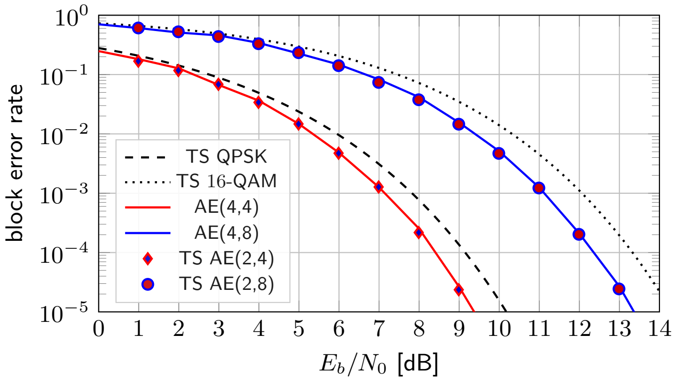
$(R = 2)$

Optimization Results: Improved Baseline



user 1:	AE(2,4)	AE(2,4)	0	0	$\cdots$	AE(2,8)	AE(2,8)	0	0	$\cdots$
user 2:	0	0	AE(2,4)	AE(2,4)	$\cdots$	0	0	AE(2,8)	AE(2,8)	$\cdots$
			$(R = 1)$					$(R = 2)$		

Optimization Results: Improved Baseline



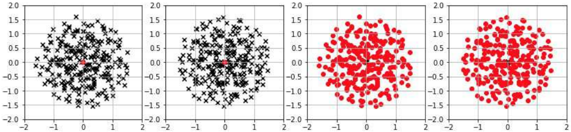
user 1:	AE(2,4)	AE(2,4)	0	0	$\cdots$
user 2:	0	0	AE(2,4)	AE(2,4)	$\cdots$

$(R = 1)$

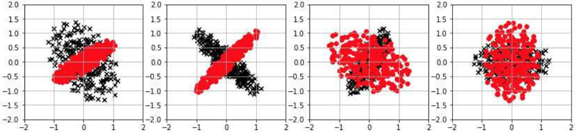
AE(2,8)	AE(2,8)	0	0	$\cdots$
0	0	AE(2,8)	AE(2,8)	$\cdots$

$(R = 2)$

What Does The Autoencoder Learn?

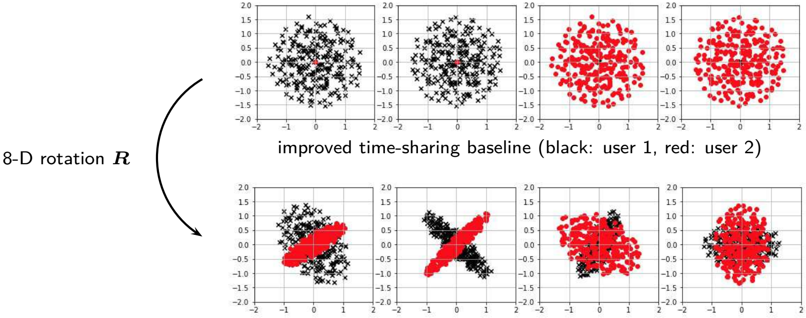


improved time-sharing baseline (black: user 1, red: user 2)



learned AE(4,8) constellation with 256 points

What Does The Autoencoder Learn?

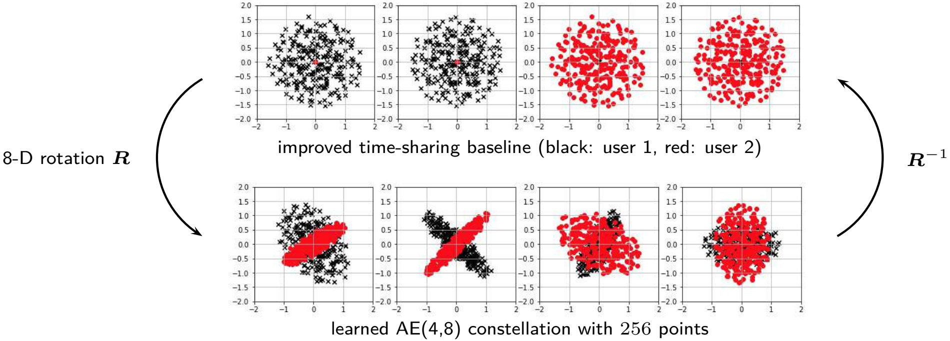


improved time-sharing baseline (black: user 1, red: user 2)

learned AE(4,8) constellation with 256 points

- Applying a random rotation matrix recovers elliptical shapes

What Does The Autoencoder Learn?



- Applying a random rotation matrix recovers elliptical shapes
- One can always find a "de-rotating" matrix through simple optimization, which (approximately) converts the learned AE solution to time-sharing

Summary

- End-to-end learning can be extended to multiple users. However, a key challenge is to properly benchmark and interpret the obtained solutions
- For a simple Gaussian interference scenario, the autoencoder learns to avoid interference, but in an arbitrarily rotated reference frame
- Could potentially be interesting for nonlinear optical WDM channels or multi-core/mode scenarios with independent processing

Further Reading:

- [O'Shea and Hoydis, 2017], "An introduction to deep learning for the physical layer": originally proposed multiuser learning
- [Song et al., 2022b], "Benchmarking and Interpreting End-to-end Learning of MIMO and Multi-User Communication": full interpretation of Gaussian interference channel results

  (source code: `github.com/JSChalmers/DeepLearningMIMO.git`)

Outline

1. Introduction to End-to-End Autoencoder Learning
2. Autoencoder Design Elements
3. Estimating Capacity Bounds
4. End-to-End Learning with Multiple Users
5. Conclusion

Conclusion

Learning objectives

1. Introduction to basic topics:
   - What is the main idea behind end-to-end learning with simple examples
   - Main design elements: model selection, choice of loss function, and training paradigms
2. Overview of some more advanced topics:
   - How to estimate channel capacity with end-to-end learning
   - How to do end-to-end learning with multiple users

Conclusion

Learning objectives

1. Introduction to basic topics:
   - What is the main idea behind end-to-end learning with simple examples
   - Main design elements: model selection, choice of loss function, and training paradigms
2. Overview of some more advanced topics:
   - How to estimate channel capacity with end-to-end learning
   - How to do end-to-end learning with multiple users

Thank you!

FORCE
FIBER-OPTIC COMMUNICATIONS RESEARCH CENTER

References I

Aharoni, Z., Tsur, D., Goldfeld, Z., and Permuter, H. H. (2020).
Capacity of continuous channels with memory via directed information neural estimator.
In *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Los Angeles, CA.

Aoudia, F. A. and Hoydis, J. (2019).
Model-free training of end-to-end communication systems.
IEEE J. Sel. Areas Commun., 37(11):2503–2516.

Cammerer, S., Aoudia, F. A., Dorner, S., Stark, M., Hoydis, J., and Ten Brink, S. (2020).
Trainable communication systems: Concepts and prototype.
IEEE Trans. Commun., 68(9):5489–5503.

Crivelli, D. E., Hueda, M. R., Carrer, H. S., Del Barco, M., López, R. R., Gianni, P., Finochietto, J., Swenson, N., Voois, P., and Agazzi, O. E. (2014).
Architecture of a single-chip 50 Gb/s DP-QPSK/BPSK transceiver with electronic dispersion compensation for coherent optical channels.
IEEE Trans. Circuits Syst. I: Reg. Papers, 61(4):1012–1025.

Csiszár, I. and Körner, J. (1981).
Information Theory: Coding Theorems for Discrete Memoryless Systems.
Academic Press.

Dörner, S., Cammerer, S., Hoydis, J., and ten Brink, S. (2018).
Deep learning-based communication over the air.
IEEE J. Sel. Topics Signal Proc., 12(1):132–143.

References II

Fritschek, R., Schaefer, R. F., and Wunder, G. (2019).
Deep learning for channel coding via neural mutual information estimation.
In *Proc. IEEE Int. Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Cannes, France.

Fritschek, R., Schaefer, R. F., and Wunder, G. (2020).
Neural mutual information estimation for channel coding: State-of-the-art estimators, analysis, and performance comparison.
In *Proc. IEEE Int. Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Atlanta, GA.

Gümüs, K., Alvarado, A., Chen, B., Häger, C., and Agrell, E. (2020).
End-to-end learning of geometrical shaping maximizing generalized mutual information.
In *Proc. Optical Fiber Communication Conf. (OFC)*, San Diego, CA.

Häger, C. and Agrell, E. (2022).
Data-driven estimation of capacity upper bounds.
IEEE Commun. Lett.

Jones, R. T., Eriksson, T. A., Yankov, M. P., and Zibar, D. (2018).
Deep learning of geometric constellation shaping including fiber nonlinearities.
In *Proc. European Conf. Optical Communication (ECOC)*, Rome, Italy.

Jones, R. T., Yankov, M. P., and Zibar, D. (2019).
End-to-end learning for GMI optimized geometric constellation shape.
In *Proc. European Conf. Optical Communication (ECOC)*, Dublin, Ireland.

Jovanovic, O., Yankov, M. P., Da Ros, F., and Zibar, D. (2021).
Gradient-free training of autoencoders for non-differentiable communication channels.
J. Lightw. Technol., 39(20):6381–6391.

References III

Karanov, B., Chagnon, M., Thouin, F., Eriksson, T. A., Bulow, H., Lavery, D., Bayvel, P., and Schmalen, L. (2018).
End-to-end deep learning of optical fiber communications.
J. Lightw. Technol., 36(20):4843–4855.

Karanov, B., Lavery, D., Bayvel, P., and Schmalen, L. (2019).
End-to-end optimized transmission over dispersive intensity-modulated channels using bidirectional recurrent neural networks.
Opt. Express, 27(14):19650–19663.

Letizia, N. A. and Tonello, A. M. (2021).
Capacity-driven autoencoders for communications.
IEEE Open J. Commun. Soc., 2:1366–1378.

Li, S., Häger, C., Garcia, N., and Wymeersch, H. (2018).
Achievable information rates for nonlinear fiber communication via end-to-end autoencoder learning.
In *Proc. European Conf. Optical Communication (ECOC)*, Rome, Italy.

Mirkarimi, F. and Farsad, N. (2021).
Neural computation of capacity region of memoryless multiple access channels.
In *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Melbourne, Australia.

Mirkarimi, F. and Rini, S. (2022).
A perspective on neural capacity estimation: Viability and reliability.
arXiv:2203.11793.

Mirkarimi, F., Rini, S., and Farsad, N. (2021).
Neural capacity estimators: How reliable are they?
arXiv:2111.07401 [cs.IT].

References IV

O'Shea, T. and Hoydis, J. (2017).
An introduction to deep learning for the physical layer.
IEEE Trans. Cogn. Commun. Netw., 3(4):563–575.

Song, J., Häger, C., Schröder, J., Amat, A. G. i., and Wymeersch, H. (2022a).
Model-based end-to-end learning for wdm systems with transceiver hardware impairments.
IEEE J. Sel. Topics. Quantum Electron., 28(4).

Song, J., Häger, C., Schröder, J., O'Shea, T. J., Agrell, E., and Wymeersch, H. (2022b).
Benchmarking and interpreting end-to-end learning of mimo and multi-user communication.
IEEE Trans. Wireless Commun., 21(9):7287–7298.

Song, J., Peng, B., Häger, C., Wymeersch, H., and Sahai, A. (2020).
Learning physical-layer communication with quantized feedback.
IEEE Trans. Commun., 68(1):645–653.

Tsur, D., Aharoni, Z., Goldfeld, Z., and Permuter, H. (2022).
Neural estimation and optimization of directed information over continuous spaces.
arXiv:2203.14743.

Uhlemann, T., Cammerer, S., Span, A., Dörner, S., and ten Brink, S. (2020).
Deep-learning autoencoder for coherent and nonlinear optical communication.
In *IEEE/ITG Symp. Photon. Netw.*, Leipzig, Germany.