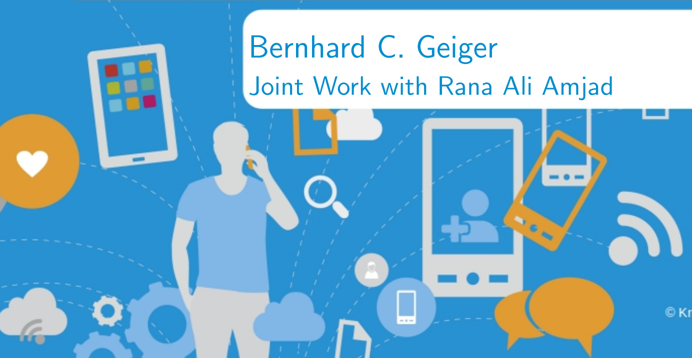
AUSTRIA'S LEADING RESEARCH CENTER
FOR DATA-DRIVEN BUSINESS AND BIG DATA ANALYTICS

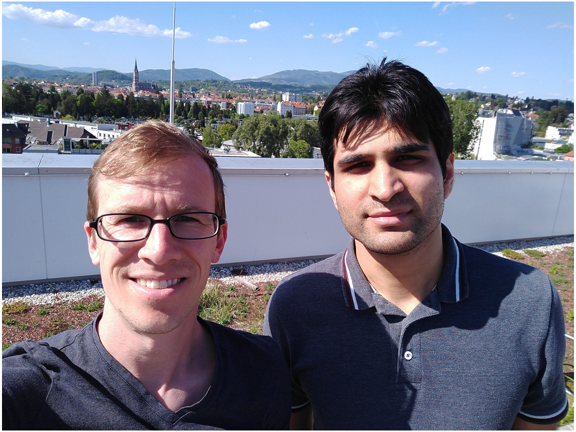
# How (Not) To Train Your DNN Using The Information Bottleneck Functional


Bernhard C. Geiger

Joint Work with Rana Ali Amjad

# The Authors and Funders

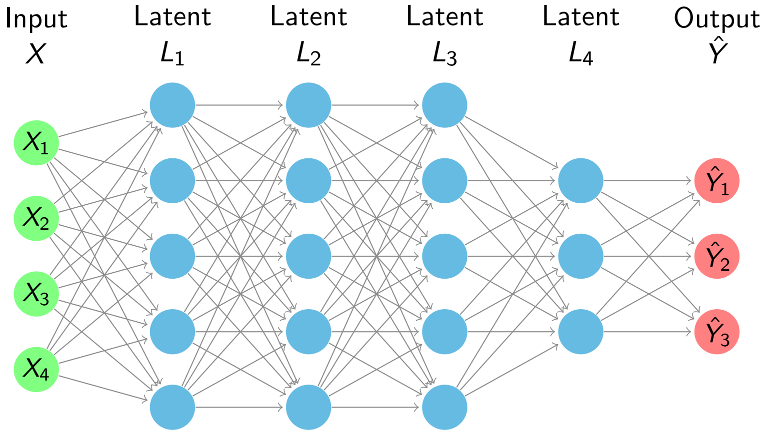


Der Wissenschaftsfonds.

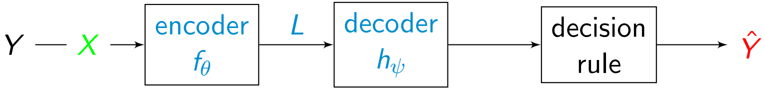
Unterstützt von / Supported by

Alexander von Humboldt
Stiftung / Foundation

# Neural Network for Classification



# Setup and Notation




- $Y \in \mathcal{Y}$, $\mathcal{Y}$ finite set
- $X \in \mathbb{R}^N$
- Joint distribution of $X, Y$ is known
- Encoder and decoder are **deterministic**, e.g.,

$$L_{i+1} = \sigma\left(\mathbb{W}_i^T L_i + b_{i+1}\right)$$

and $\theta = \{\mathbb{W}_0, \ldots, \mathbb{W}_{i-1}, b_1, \ldots, b_i\}$

# Learning Representations for Classification

Intermediate representation $L$ should

P1 Contain sufficient info for classification (DPI!)

# Learning Representations for Classification

Intermediate representation $L$ should

- P1 Contain sufficient info for classification (DPI!)
- P2 ...but not more info than necessary (compression)

# Learning Representations for Classification

Intermediate representation $L$ should

- P1 Contain sufficient info for classification (DPI!)
- P2 ...but not more info than necessary (compression)
- P3 Allow extracting this info easily (w.r.t. decoder)

# Learning Representations for Classification

Intermediate representation $L$ should

- P1 Contain sufficient info for classification (DPI!)
- P2 ...but not more info than necessary (compression)
- P3 Allow extracting this info easily (w.r.t. decoder)
- P4 Be robust to small noise and deformations (generalization)

# Learning Representations for Classification

Intermediate representation $L$ should

- P1 Contain sufficient info for classification (DPI!)
- P2 ...but not more info than necessary (compression)
- P3 Allow extracting this info easily (w.r.t. decoder)
- P4 Be robust to small noise and deformations (generalization)

$$\text{P1} \Leftrightarrow \text{large } I(Y;L)$$
$$\text{P2} \Leftrightarrow \text{small } I(X;L)$$

# IB Principle for Training DNN Classifier

IB principle for training DNNs[1]

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

[1]Tishby and Zaslavsky, "Deep learning and the information bottleneck principle", 2015

[2]Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[3]Alemi et al., "Deep Variational Information Bottleneck", 2017

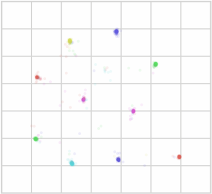
# IB Principle for Training DNN Classifier

IB principle for training DNNs[1]

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

Approximations yield[2,3]

- simple latent representation
- improved generalization
- adversarial robustness



taken from [2]

[1] Tishby and Zaslavsky, "Deep learning and the information bottleneck principle", 2015

[2] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[3] Alemi et al., "Deep Variational Information Bottleneck", 2017

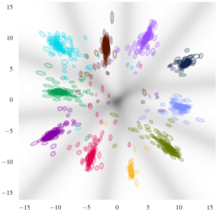
# IB Principle for Training DNN Classifier

IB principle for training DNNs[1]

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

Approximations yield[2,3]

- simple latent representation
- improved generalization
- adversarial robustness




taken from [3]

[1] Tishby and Zaslavsky, "Deep learning and the information bottleneck principle", 2015
[2] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018
[3] Alemi et al., "Deep Variational Information Bottleneck", 2017

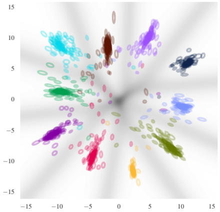
# IB Principle for Training DNN Classifier

IB principle for training DNNs[1]

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

Approximations yield[2,3]

- simple latent representation
- improved generalization
- adversarial robustness




taken from [3]

Do we have $(P1 \wedge P2) \implies (P3 \wedge P4)$?

[1] Tishby and Zaslavsky, "Deep learning and the information bottleneck principle", 2015

[2] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[3] Alemi et al., "Deep Variational Information Bottleneck", 2017

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X; L) - \beta I(Y; L)$$

- Focus on P1 and P2, defined via mutual information

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

- Focus on P1 and P2, defined via mutual information
  - Computable?

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

- Focus on P1 and P2, defined via mutual information
  - Computable?
  - Optimizable?

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

- Focus on P1 and P2, defined via mutual information
  - Computable?
  - Optimizable?
  - Invariant under bijections

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

- Focus on P1 and P2, defined via mutual information
  - Computable?
  - Optimizable?
  - Invariant under bijections
- Focus on the encoder $f_\theta$, decoder (P3!) not considered

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

- Focus on P1 and P2, defined via mutual information
  - Computable?
  - Optimizable?
  - Invariant under bijections
- Focus on the encoder $f_\theta$, decoder (P3!) not considered
- (Focus on $L$, architectural simplicity not considered)

# Computability

**Theorem**

Let $X$ have a PDF $f_X$ that is continuous on $\mathcal{X} \subset \mathbb{R}^N$.

# Computability

**Theorem**

Let $X$ have a PDF $f_X$ that is continuous on $\mathcal{X} \subset \mathbb{R}^N$. Let $\sigma$ be either bi-Lipschitz or continuously differentiable with strictly positive derivative.

# Computability

**Theorem**

Let $X$ have a PDF $f_X$ that is continuous on $\mathcal{X} \subset \mathbb{R}^N$. Let $\sigma$ be either bi-Lipschitz or continuously differentiable with strictly positive derivative. Then, for almost every choice of $\theta$, we have

$$I(X; L) = \infty.$$

# Optimizability

Let $X$ have a discrete distribution $\implies$ IB functional is finite

# Optimizability

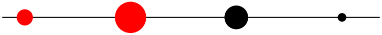
Let $X$ have a discrete distribution $\Longrightarrow$ IB functional is finite

- IB functional is a piecewise constant function of $\theta$
- Cannot use gradient-based optimization techniques

# Optimizability

Let $X$ have a discrete distribution $\implies$ IB functional is finite

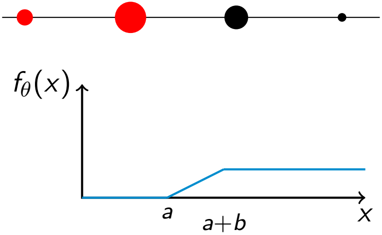
- IB functional is a piecewise constant function of $\theta$
- Cannot use gradient-based optimization techniques



# Optimizability

Let $X$ have a discrete distribution $\implies$ IB functional is finite

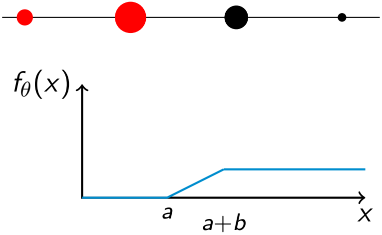
- IB functional is a piecewise constant function of $\theta$
- Cannot use gradient-based optimization techniques

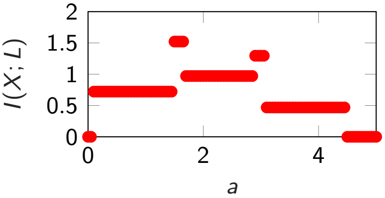


# Optimizability

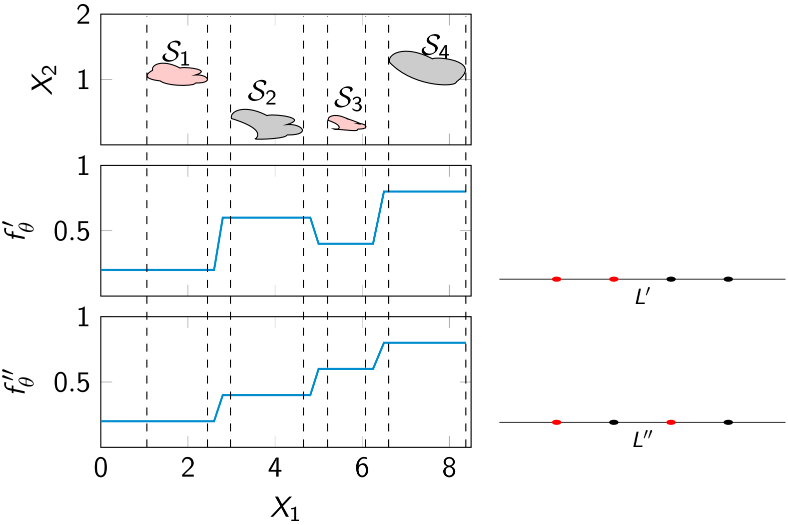
Let $X$ have a discrete distribution $\implies$ IB functional is finite

- IB functional is a piecewise constant function of $\theta$
- Cannot use gradient-based optimization techniques

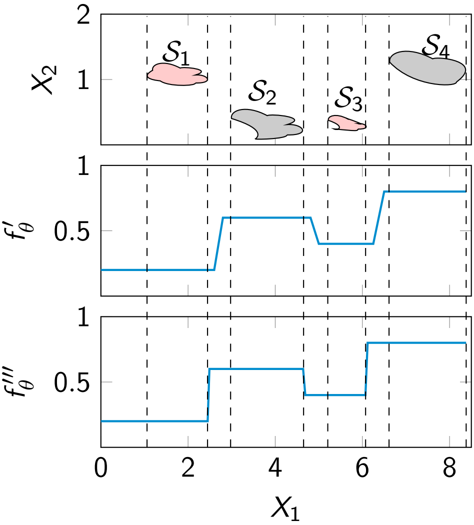


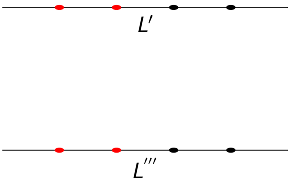



# Invariance under Bijections: No P3



# Invariance under Bijections: No P4



# IB for Learning Representations – Summary

The IB functional

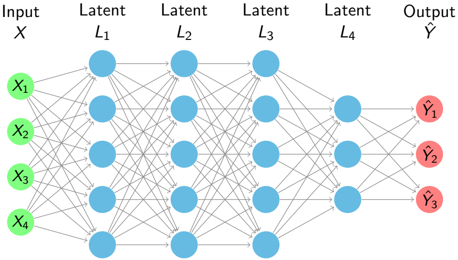
- is infinite for continuous input
- is piecewise constant in general
- does not encourage “simple” representations (P3)
- does not encourage robust representations (P4)

Why does it work?[4,5]

[4] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[5] Alemi et al., “Deep Variational Information Bottleneck”, 2017

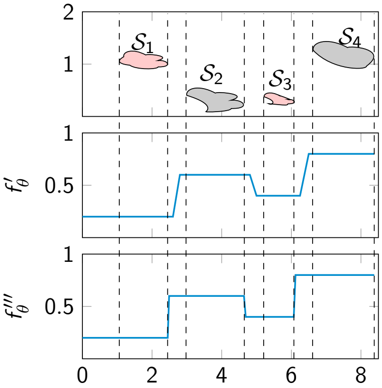
# How to Train your DNN (1)




$$\min_{\theta} I(X;\hat{Y}) - \beta I(Y;\hat{Y})$$

- Include decision rule (arg max, softmax, etc.) $\implies$ P3
- Compression term may become useless/harmful

# How to Train your DNN (2)




- Train a stochastic DNN (e.g., add noise)
- Leads to robustness (P4)
- Encourages geometric clustering[6] (P2)

[6] Goldfeld et al., *Estimating Information Flow in Neural Networks*, 2018

# How to Train your DNN (3)

From

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

to, e.g., cross-entropy and variational bounds.

- Replace IB functional by better-behaved cost function
- E.g., cross-entropy encourages P1 and P3
- Variational bounds may encourage geometric compression P2
- etc.

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

Implemented approximations yield[7,8,9,10]

- simple latent representation
- improved generalization
- adversarial robustness

[7] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[8] Alemi et al., "Deep Variational Information Bottleneck", 2017

[9] Banerjee and Montufar, *The Variational Deficiency Bottleneck*, 2018

[10] Alemi, Fischer, and Dillon, *Uncertainty in the Variational Information Bottleneck*, 2018

# IB Principle for Training DNN Classifier

$$\min_{\theta} I(X;L) - \beta I(Y;L)$$

Implemented approximations yield[7,8,9,10]

- simple latent representation
- improved generalization
- adversarial robustness

It's the approximations that make the IB principle work!

[7] Kolchinsky, Tracey, and Wolpert, *Nonlinear Information Bottleneck*, 2018

[8] Alemi et al., "Deep Variational Information Bottleneck", 2017

[9] Banerjee and Montufar, *The Variational Deficiency Bottleneck*, 2018

[10] Alemi, Fischer, and Dillon, *Uncertainty in the Variational Information Bottleneck*, 2018

# Conclusion

- IB principle is insufficient for training latent representations in deterministic DNNs
  - infinite
  - piecewise constant
  - invariant under bijections
- Remedies available and backed by evidence:
  - enforce geometric (not IT) compression (P2) $\implies$ P3
  - include the decoder $\implies$ P3
  - introduce stochasticity $\implies$ P4

# Conclusion

- IB principle is insufficient for training latent representations in deterministic DNNs
  - infinite
  - piecewise constant
  - invariant under bijections
- Remedies available and backed by evidence:
  - enforce geometric (not IT) compression (P2) $\implies$ P3
  - include the decoder $\implies$ P3
  - introduce stochasticity $\implies$ P4

Thanks!

# ReLU Activation Functions

IB functional is either

- infinite, or
- a piecewise constant function of $\theta$