EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Department of Computer Science,
Cluster of Excellence "Machine Learning for Science",
Tübingen AI Center

# Data Compression With and Without Deep Probabilistic Models

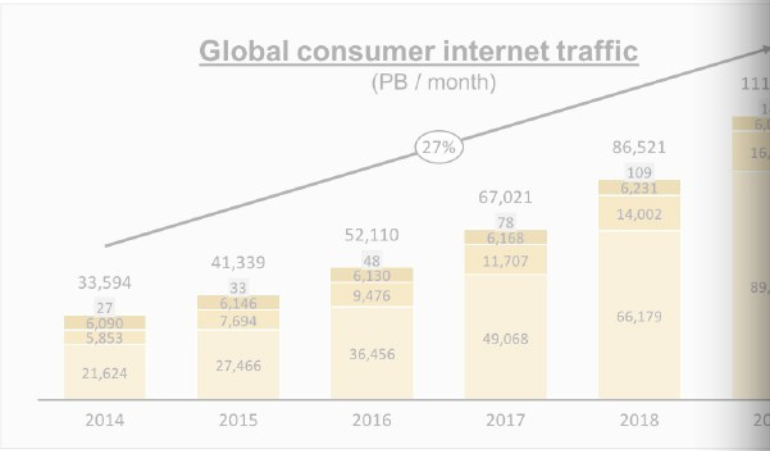
Prof. Robert Bamler

Lecture #1 of course "Data Compression With Deep Probabilistic Models"

University of Tuebingen • 21 April 2022

# Why Are We Here?

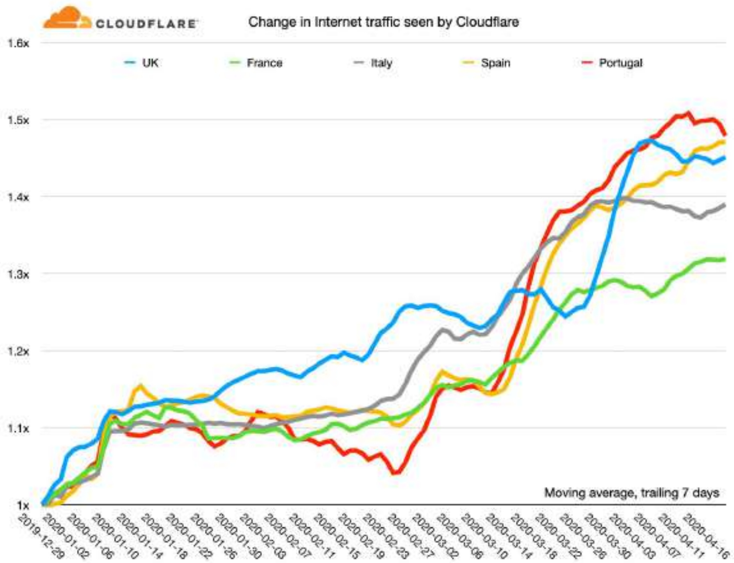
## Projection by Cisco from 2015:
27% growth per year




Source: Cisco VNI, 2015

[https://www.aploris.com/blog/charts/stack-bar-chart-examining-the-growth-of-global-consumer-internet-traffic/]

## Observation by Cloudflare in 2020




40% growth in 3 months

January 2020

April 2020

[https://blog.cloudflare.com/recent-trends-in-internet-traffic/]

# Data Compression With Machine Learning

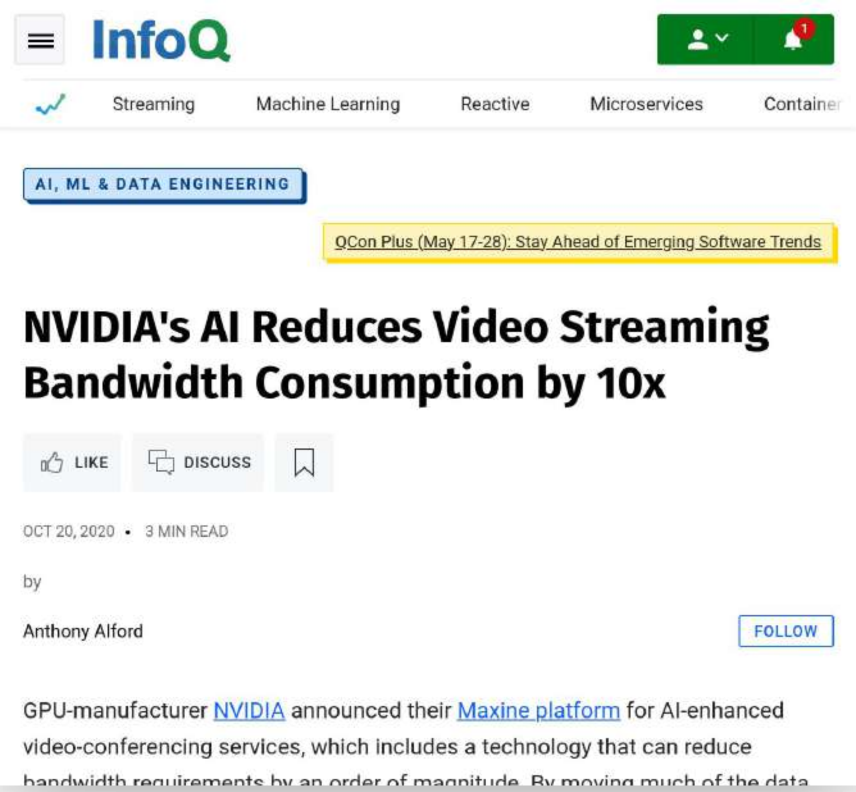
## a) common data types

NETFLIX

YouTube

Disney+

zoom

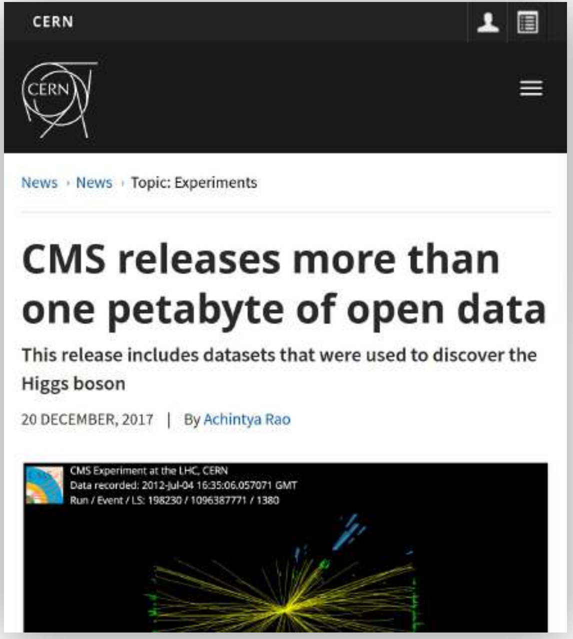



[https://www.infoq.com/news/2020/10/nvidia-video-bandwidth/]

# Data Compression With Machine Learning

## a) common data types

## b) specialized data types


CERN

News › News › Topic: Experiments

CMS releases more than one petabyte of open data

This release includes datasets that were used to discover the Higgs boson

20 DECEMBER, 2017 | By Achintya Rao



[home.cern]

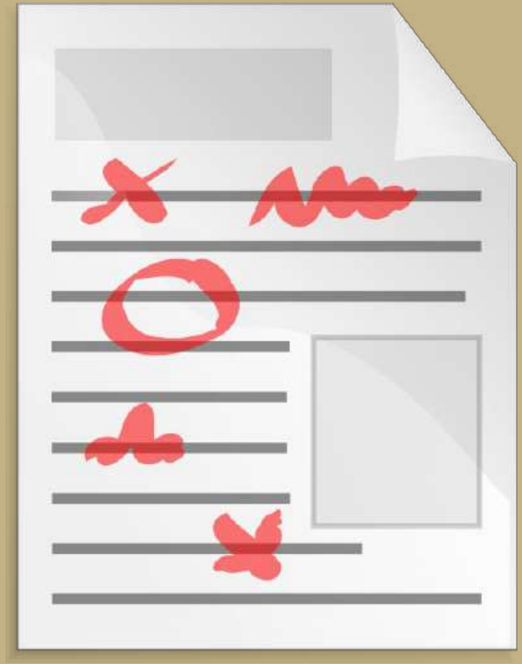
[openneuro.org]

# This Course

## From information theory to applications:

- theory of communication (*aka* information theory)
- theoretical bounds for lossless & lossy compression
- generic compression algorithms (“entropy coders”)
  - mathematical proofs of optimality & practical implementations
- probabilistic models & probabilistic inference
  - mathematical derivations & practical implementations
- interplay between (deep) probabilistic models & compression algorithms

# Administrative Issues



# I Need Your Feedback

**This is an advanced course. Please don’t be shy and ask questions.**

# Website

https://robamler.github.io/teaching/compress22/

**Contains:**

- dates & times
- slides & notes (including these slides)
- problem sets (including code) & solutions
- additional video material (occasionally)
- links to moodle (which has zoom link)

# Lecture Times & Tutorials

**Lectures:** Thursdays, 12:15 - 13:45, room TTR2 (Maria-von-Linden-Straße 6)

- mix of whiteboard, slides, and some brief programming demos

**Tutorials:** Fridays, 12:15 - 13:45, room A302 (Sand 1)

- Problem sets online after each Thursday lecture.
- Tutorials will discuss the problem set from 8 days ago (except tomorrow, where we'll discuss today's introductory problem set).
- Problem sets are not graded but there will be anonymous self-evaluation polls on moodle. **Please participate in your own interest!**

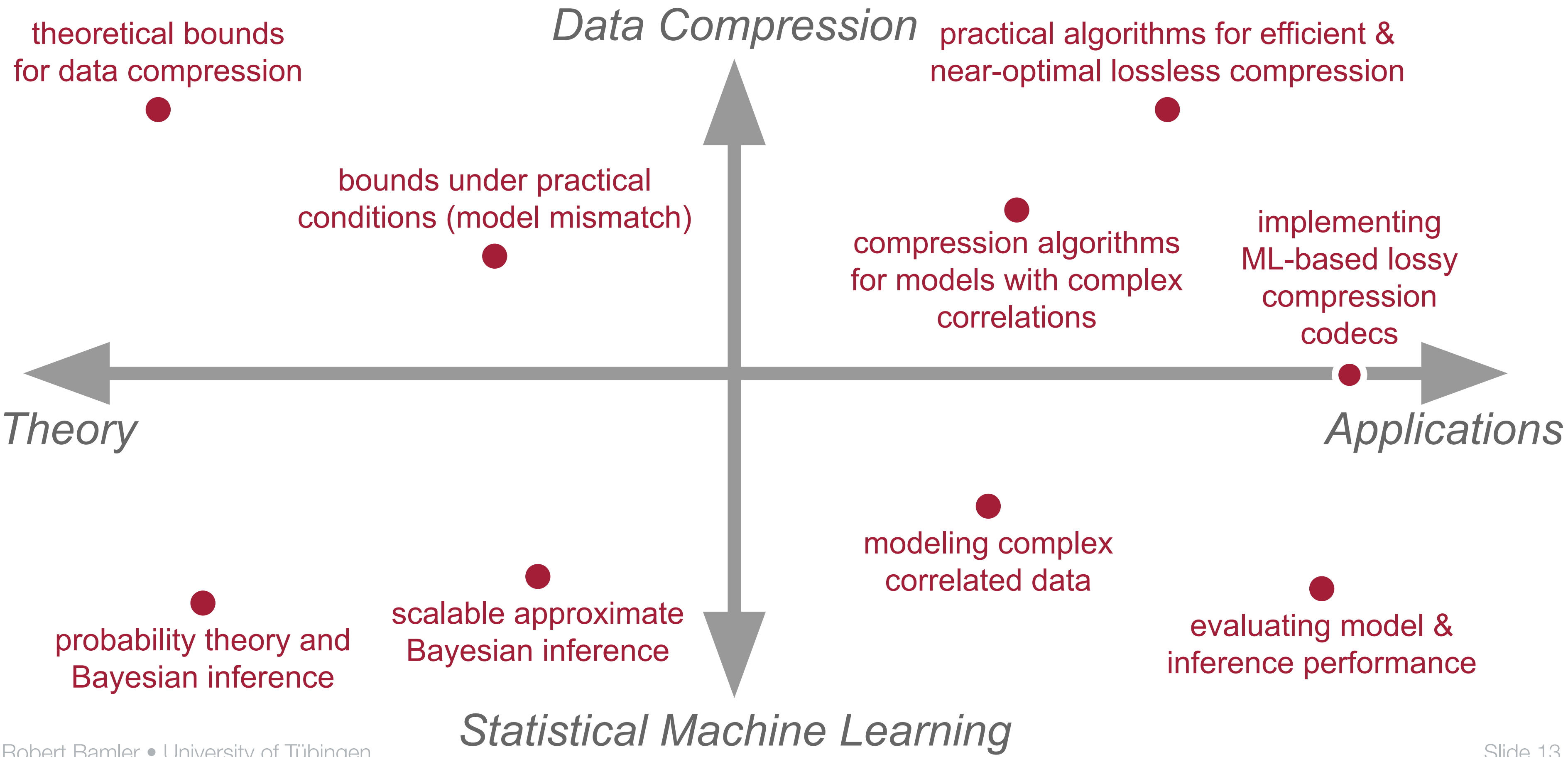
# Grading

- 6 ECTS
- exam date TBD (any conflicts?)

# Getting to Know You

# Course Overview

# Spectrum of Topics



# Prerequisites

*Data Compression*

multivariate calculus,
interest in proving
mathematical theorems

experience with Python
and with some compiled
language (C, C++, Rust)

*Theory*

*Applications*

some experience
with PyTorch
and/or Tensorflow

*Statistical Machine Learning*

# Agenda

https://robamler.github.io/teaching/compress22/

# Recommended Literature

## Books:

- **Information Theory:** MacKay, *Information theory, inference and learning algorithms*. Cambridge university press, 2003
  legal free PDF: http://www.inference.org.uk/mackay/itprnn/book.html
- **Probabilistic ML:** Murphy, Kevin P. *Machine learning: a probabilistic perspective*. MIT press, 2012.

## Videos:

- lectures by David MacKay:
  https://youtube.com/playlist?list=PLruBu5BI5n4aFpG32iMbdWoRVAA-Vcso6
- mathematicalmonk: https://youtube.com/playlist?list=PLE125425EC837021F
- videos from last year's course: see link on course website

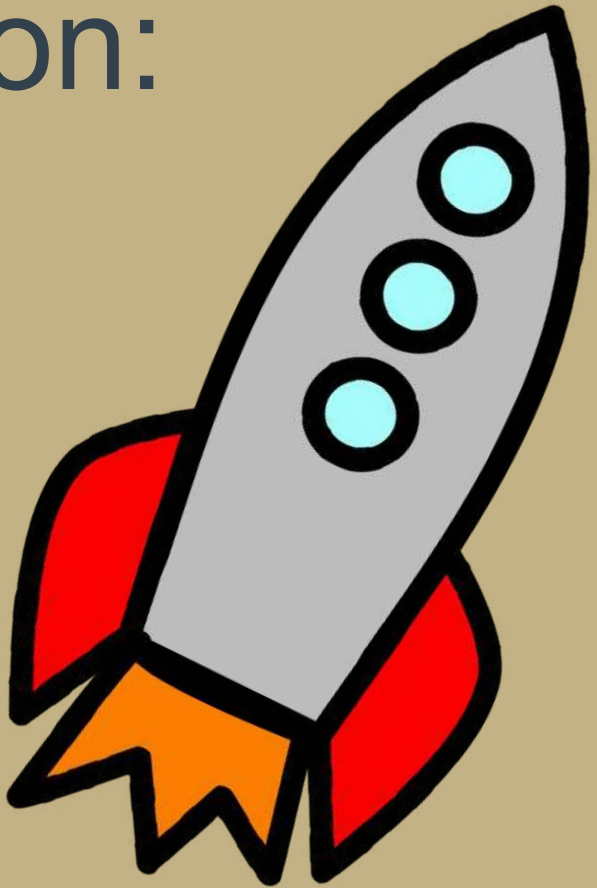
## Related Lecture:

- Probabilistic Machine Learning by Prof. Hennig (this term)

# USP: How is This Lecture Different?

- full spectrum from theory to applications
  - we'll *prove* theorems but we'll also *implement* real compression codecs
- full spectrum from information theory to statistical machine learning
  - "models and algorithms"
- highly active field of research
  → lots of recent developments not yet covered in secondary literature
  - both compression algorithms & deep probabilistic models
- interactivity
  - mini-problems in class so I can assess which topics I need to clarify

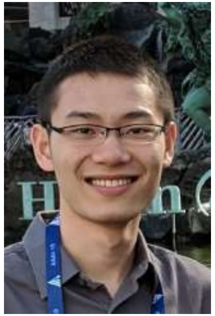
# Neural Compression:
## a highly active field of research.



# Active Research Topics (Selection)

- entropy models for important data types (especially video)
  - normalizing flows
  - implicit generative models
- capturing correlations in compression algorithms
  - compression algorithms for hierarchical models
  - compression algorithms for stochastic sampling
- quantization of real-valued neural activations
- model compression
- semantic compression
- learned distortion measures
- computational efficiency of neural compression meth

collaborators:

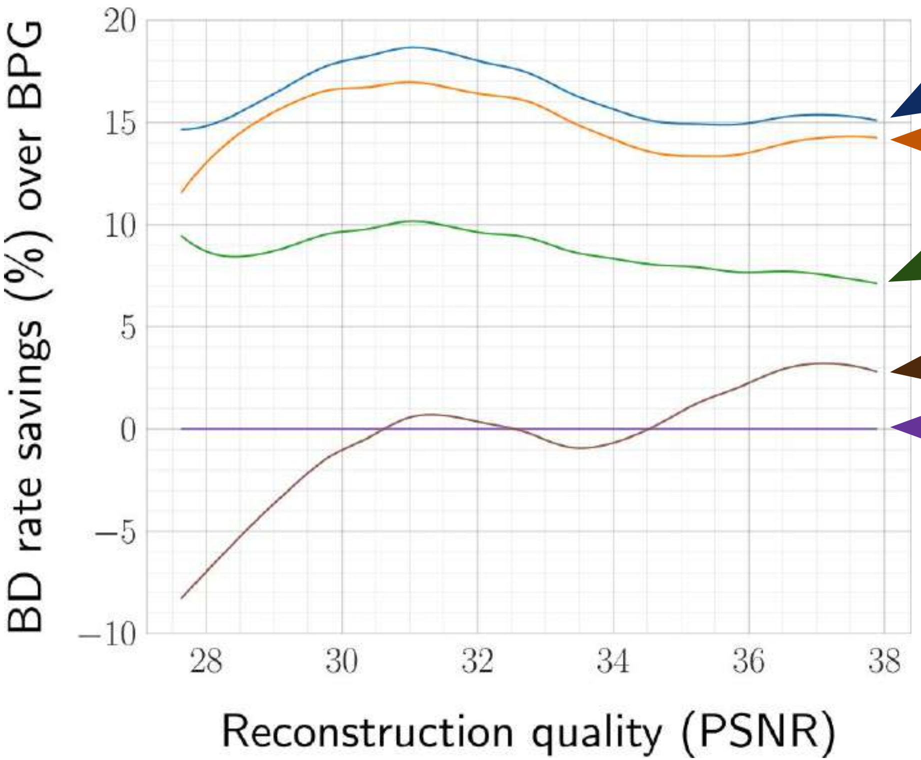


Yibo Yang (UC Irvine)

Stephan Mandt (UC Irvine)

# Example: Lossy Image Compression

[Yang, RB, Mandt, NeurIPS 2020]




ours (with **lossy bits-back coding**)

ours

previously best neural method [Minnen et al., 2018]

our starting point [Ballé et al., 2018]

reference: best non-neural codec (BPG 4:4:4) [Bellard, 2014]

# What’s the Population of Rome?

- ► On 30 April 2018: 2,879,728
- ► In the year 500 AD: 100,000


original

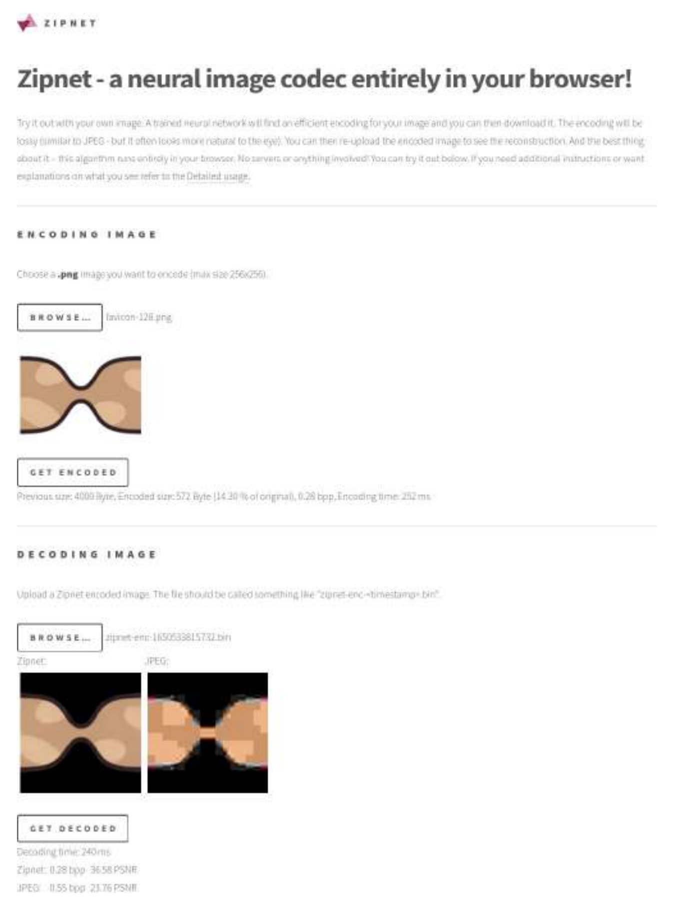
# JPEG @ 0.24 bits per pixel


Ours @ 0.24 bits per pixel

# Case Study: Student Project

Neural compression method running *in a web browser*

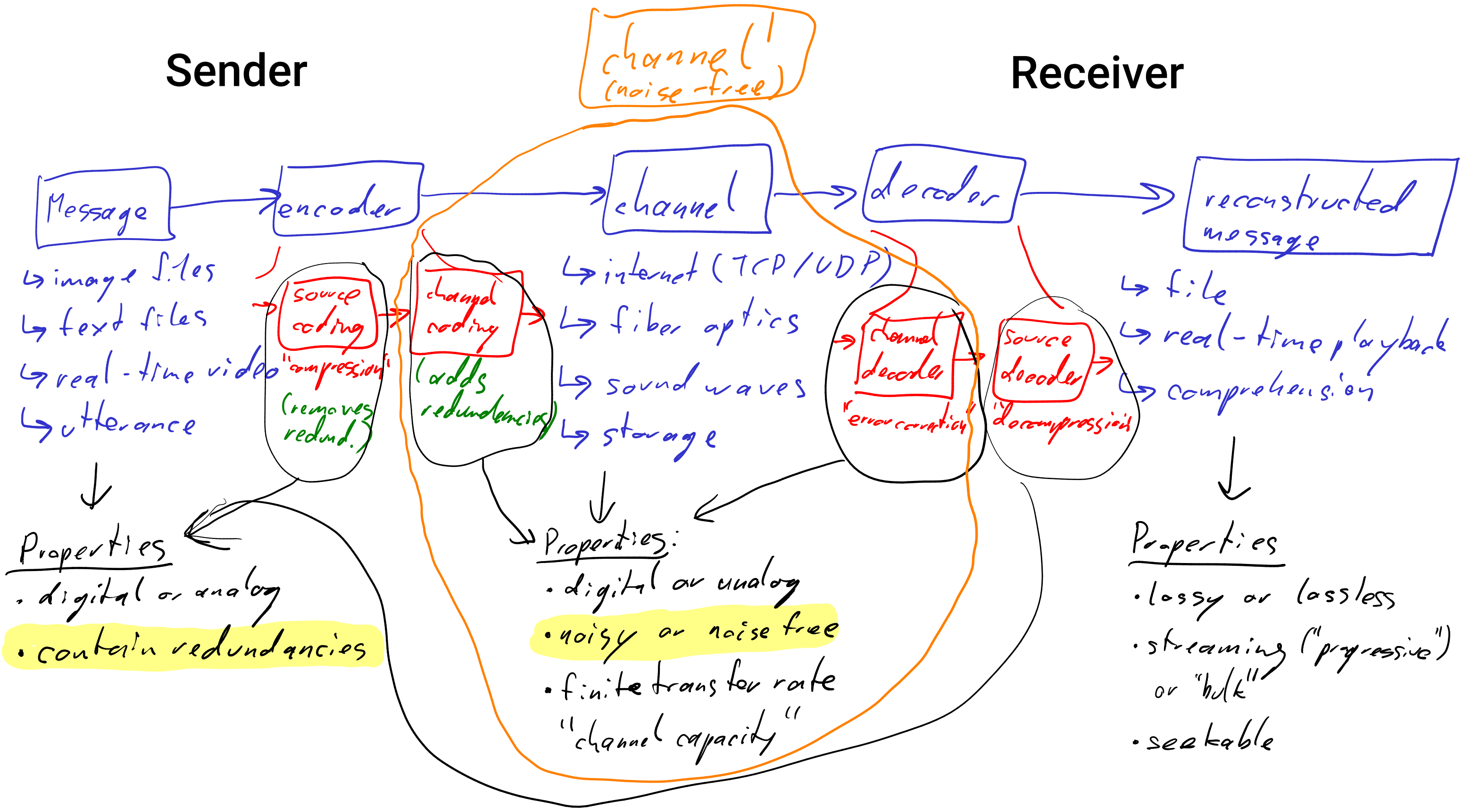
© Alexander Conzelmann and Shrisha Bharadwaj



# Data Compression With and Without Deep Probabilistic Models

Lecture 1 (21 April 2022)

## Problem Setting: communication over a channel




"Source - Channel separation theorem" → we'll cover this in more detail when we talk about lossy compression

Goal: transmit message from Sender to Receiver:
- fast, i.e., using the channel as little as possible
- reliably, i.e., without errors or with as little (relevant) distortion as possible

**Question 1:** Assume you want to transmit some message over some channel. The message is given as a string of N bits, and the channel transmits one bit at a time but it occasionally introduces a random bit flip. How many bits do you have to transmit over the channel if you want to be certain beyond reasonable doubt that the receiver of the message can decode it without errors?

A) N bits

B) more than N bits

C) fewer than N bits

D) it depends

→ • If we have any prior knowledge then we can compress the message (e.g., if we know that the message is an ASCII representation of English text then we know that the letter "e" is most frequent and that the letter "q" is almost always followed by a "u").

• Error correction requires to add some bits

→ see Questions 2&3 below for further thoughts

**Question 2:** Assume you have a noise-free channel and a message that contains some redundancies (e.g., English text). What should the encoder and decoder do with these redundancies if our goal is efficient communication (i.e., using the channel as little as possible)?

→ encoder should remove redundancies
→ decoder should reconstruct the redundancies
} compression

E.g., construct a code book, on which encoder & decoder have to agree beforehand

**Question 3:** Now let's assume that the channel introduces noise (e.g., occasional bit flips). What additional task do encoder and decoder have to do? Think again about redundancies.

→ introduce new redundancies that are designed for the channel

# Lossless Compression I: Symbol Codes

## Problem Setting

- goal: efficient lossless communication over a noise free channel
- sender has some message $\underset{\sim}{x}$, wants to transmit it losslessly to a receiver in as few bits as possible

We assume that the message $\underset{\sim}{x}$ is a sequence of symbols from a discrete alphabet: ← $\mathfrak{X}$ is finite or countably infinite

$$\underline{x} = (x_1, x_2, x_3, \dots, x_k) \equiv (x_i)_{i=1}^{k} \quad \text{where} \quad x_i \in \mathfrak{X} \quad \forall i$$

↑ alphabet

Thus, our goal is to find a mapping:

$$\underline{x} \mapsto \text{bit string} \in \{0, 1\}^* \quad \leftarrow \text{Kleene star}$$

such that:

- injective (i.e. invertible)
- bit strings are short

**Symbol Codes:** Map each symbol in the message to a bit string, then simply concatenate these bit strings

← or $\{0, 1, \dots, B\}$ (with $B \geq 2$)

- code book $C: \mathfrak{X} \to \{0, 1\}^*$
- to encode a message: concatenate

concatenation ↓

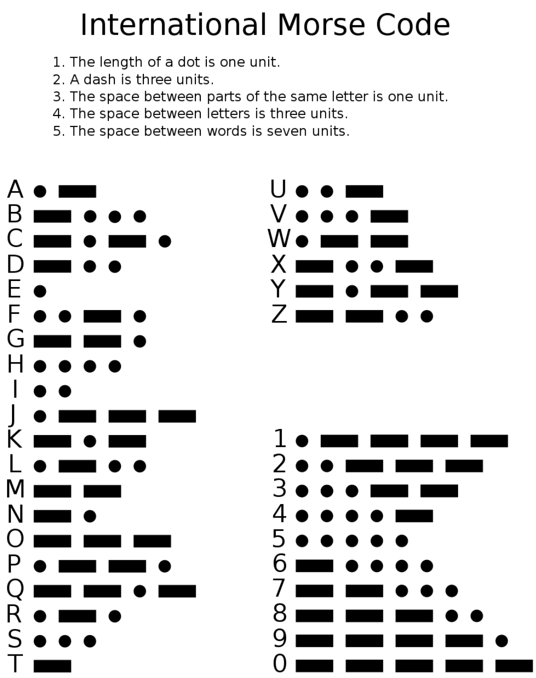
$$C^*(\underline{x}) = C^*((x_1, x_2, \dots, x_k)) := C(x_1) \,\|\, C(x_2) \,\|\, \dots \,\|\, C(x_k)$$

**Some nomenclature:**

- $C$: "Code Book"
- $C^*$: "code"
- $C(x)$: "Code word of symbol $x$"
- Def: $\ell(x) :=$ the length of $C(x)$ (in bits)

## Examples of Symbol Codes

1) Morse Code $B = 3$ (dot, dash, pause)




2) UTF-8 $B = 256$

3) "Simplified Game of Monopoly":

- throw a pair of dice and record the sum of their results as a symbol
- repeat this process several times
- for simplicity, let's use (fair) 3-sided dice:

$\mathcal{X} = \{2, 3, 4, 5, 6\}$

| $x$ | $p(x)$ | $c^{(1)}(x)$ | $c^{(2)}(x)$ | | $c^{(5)}(x)$ |
|---|---|---|---|---|---|
| 2 | 1/9 | 10 | 010 | | 010 |
| 3 | 2/9 | 11 | 011 | | 01 |
| 4 | 1/3 | 100 | 100 | | 00 |
| 5 | 2/9 | 101 | 101 | | 11 |
| 6 | 1/9 | 110 | 110 | | 110 |

problem: $c^{(1)*}$ is not injective

Claim: • $c^{(5)*}$ is injective
• an optimal code (in some sense)