ELEC-C5220
Lecture 2:
Tensors for data representation

Machine learning in information technology

Aalto University
School of Electrical
Engineering

Lauri Juvela

18.1.2024

About programming exercises

- Exercise sessions on Mondays 14-16
- First session had plenty of room available
- Deadline for exercises is Monday evening on the following week
- You can still get help for the Exercise 01 in next week's session
- Who has already returned the Exercise for Week 1?
- How much time did it take?

Question from the exercise

- What are Tensors and why do we need them?
- In the first week exercise, everything was vectors and matrices

Lecture overview

- What are Tensors?
- Tensors for representing images
- Multi-class classifiers and one-hot encoding
- Tensors for representing audio
- Audio as 1D vector (waveform)
- Audio as 2D matrix (spectrogram)

What are tensors?

- **Tensors are n-dimensional rectangular arrays**
- **Topic for this lecture: how to use tensors to represent structure in data?**
- **Examples with**
  - Images
  - Audio
  - Classes (categorical)
  - Text is also categorical, more on this later

Tensor notation

- **Scalar – 0D Tensor** $x \in \mathbb{R}$ $()$
- **Vector – 1D Tensor** $\mathbf{x} \in \mathbb{R}^D$ (D)
- **Matrix – 2D Tensor** $\mathbf{X} \in \mathbb{R}^{N \times M}$ (N, M)

Tensor notation

- **3D Tensor, e.g., multi-channel audio**

$$x \in \mathbb{R}^{B \times C \times T} \qquad (B, C, T)$$

- **4D Tensor, e.g., color images**

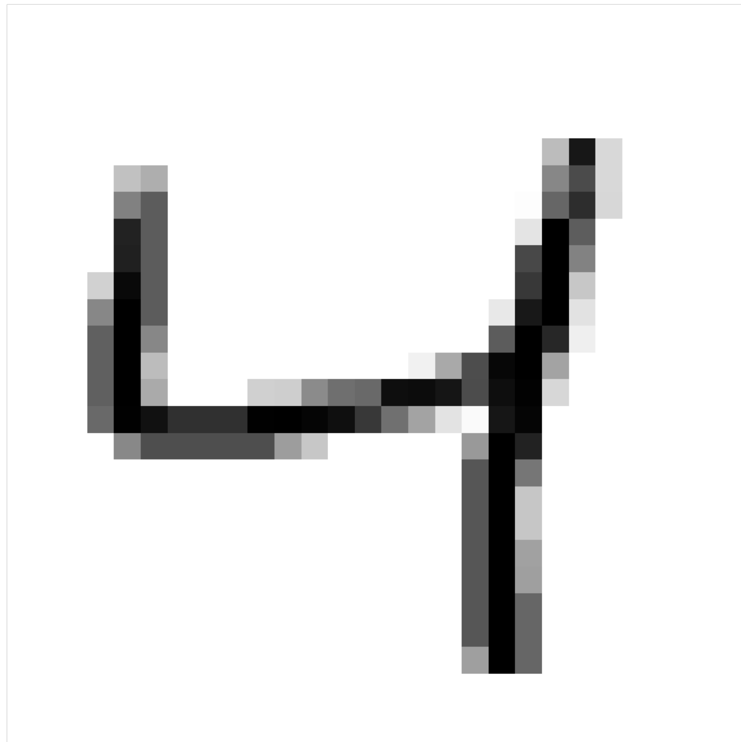
$$x \in \mathbb{R}^{B \times C \times H \times W} \qquad (B, C, H, W)$$

- **5D Tensor, e.g, video**

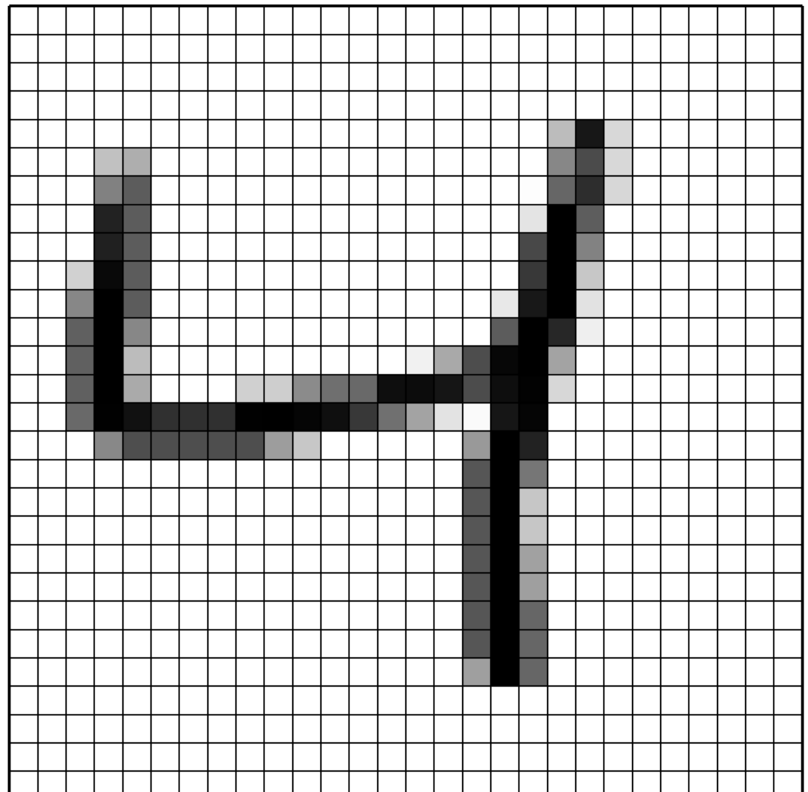
$$x \in \mathbb{R}^{B \times C \times H \times W \times T} \qquad (B, C, H, W, T)$$

Images - monochrome

MNIST hand-written digits

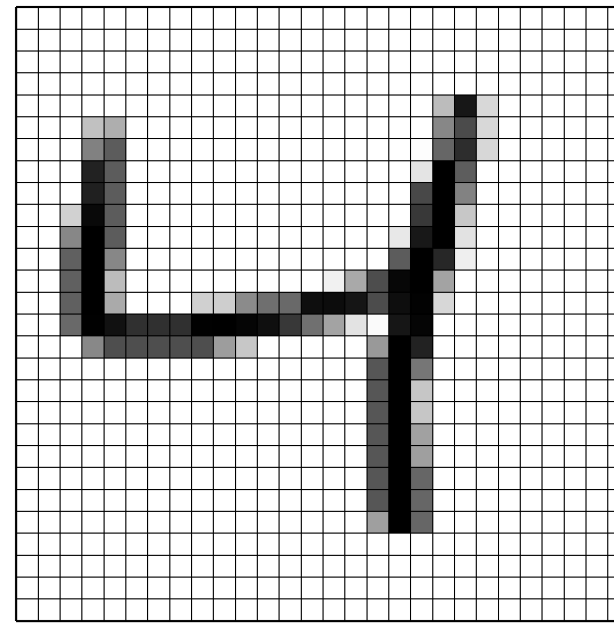


MNIST hand-written digits



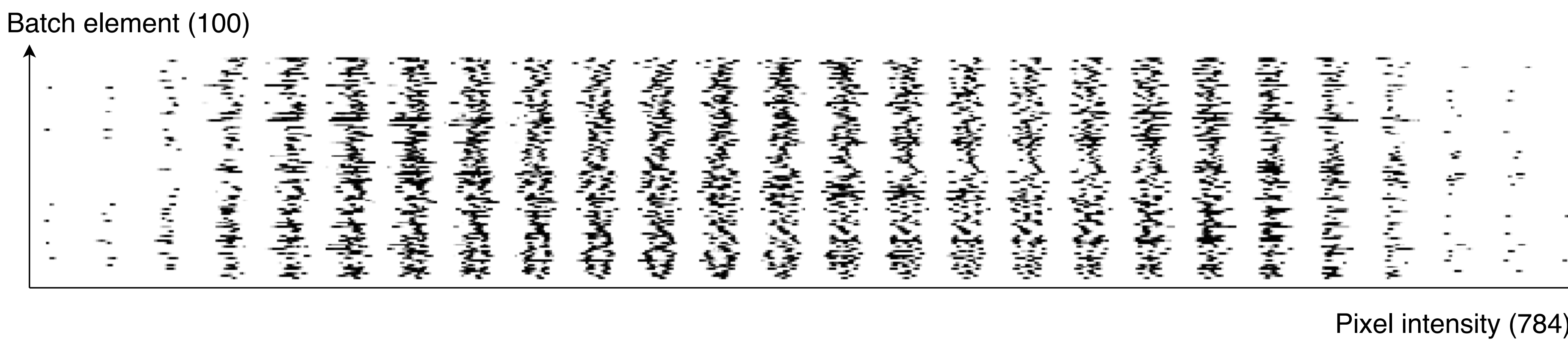
MNIST hand-written digits

- **28 x 28 pixel grid**
- **What if our model can only handle flat vector inputs, how to vectorise?**
- **Idea 1:**
  - Take every pixel value as a dimension
  - Rasterise the image to a 28 x 28 = 784 dimensional vector

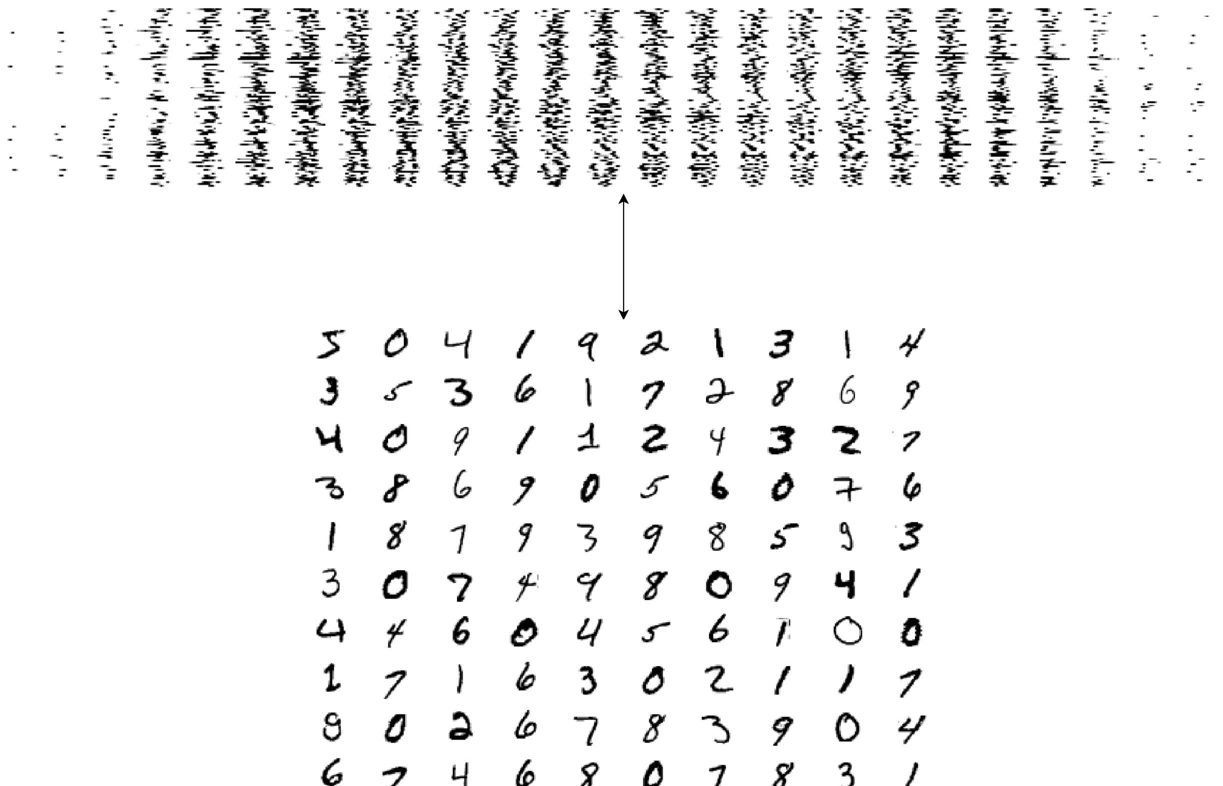


Flattened handwritten digits

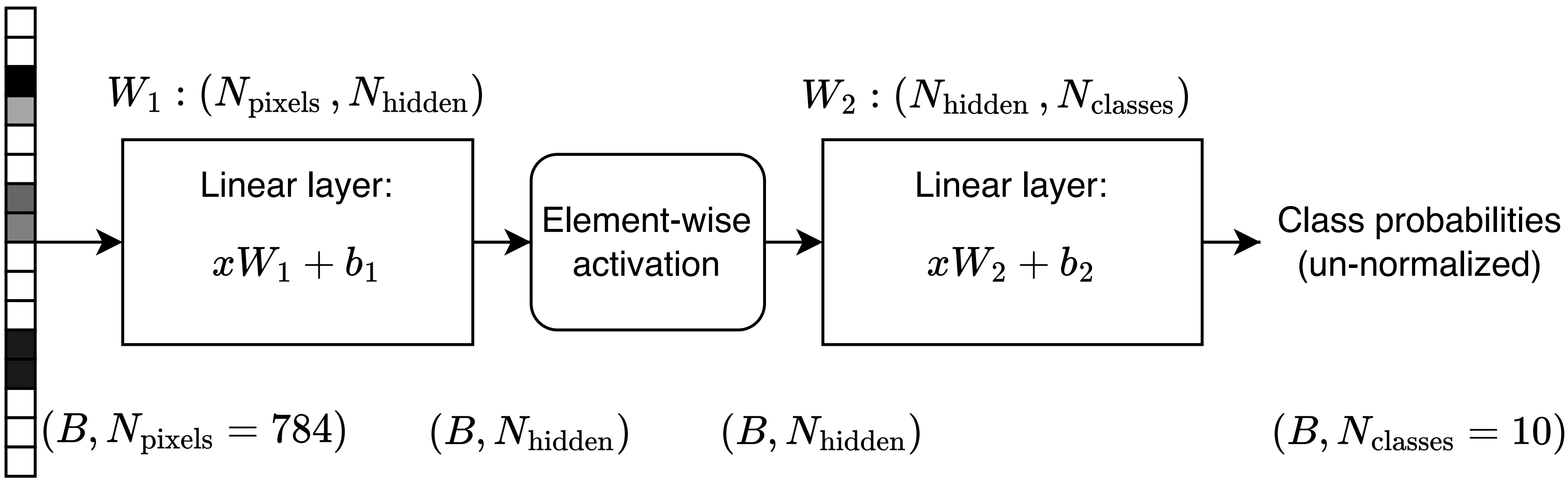
- **Pros: easy to do matrix multiplication to apply linear or DNN classifiers**
- **Cons: structure was lost, no notion of neighbouring pixels in vertical and horzontal directions**



Same data, different representations



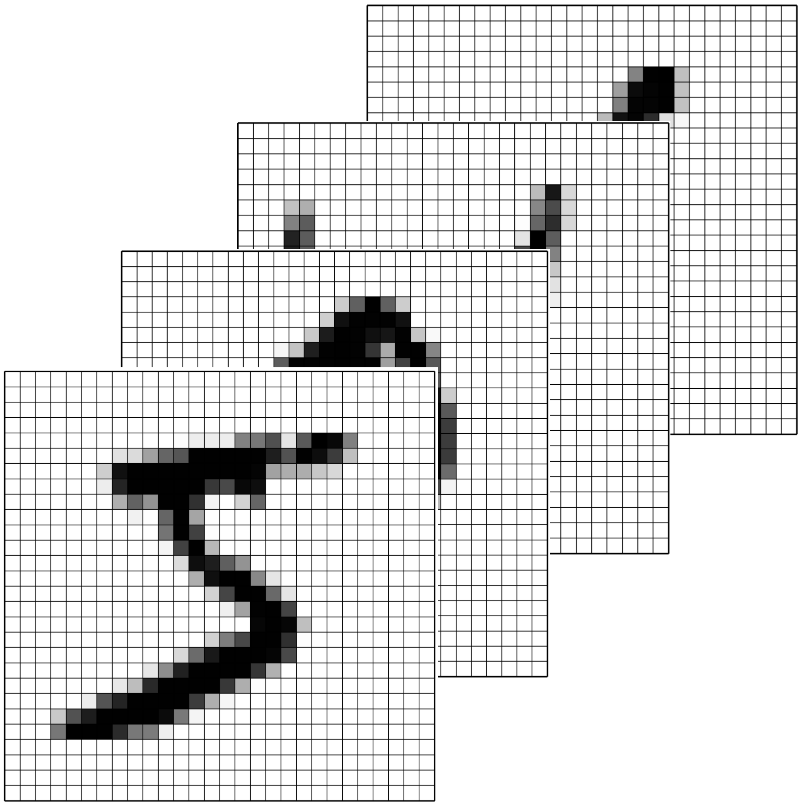
DNN Classifier for MNIST digits

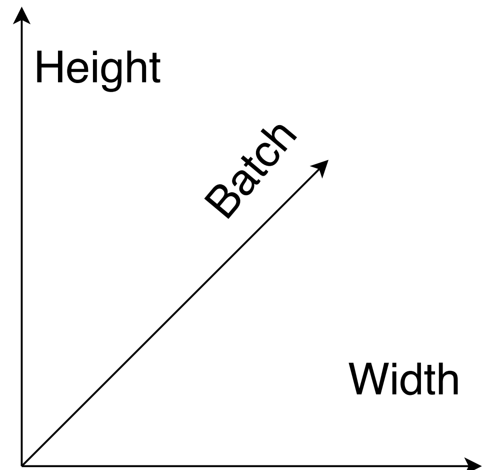


DNN Classifier for MNIST digits

- **Works fine for MNIST but,**
    - What if we want to work on other image sizes?
    - What about color
    - Very annoying for humans to inspect learned representations for debugging
    - Fragile and overparameterised
- **Try it yourself in Exercise 2!**

Tensors for representing images

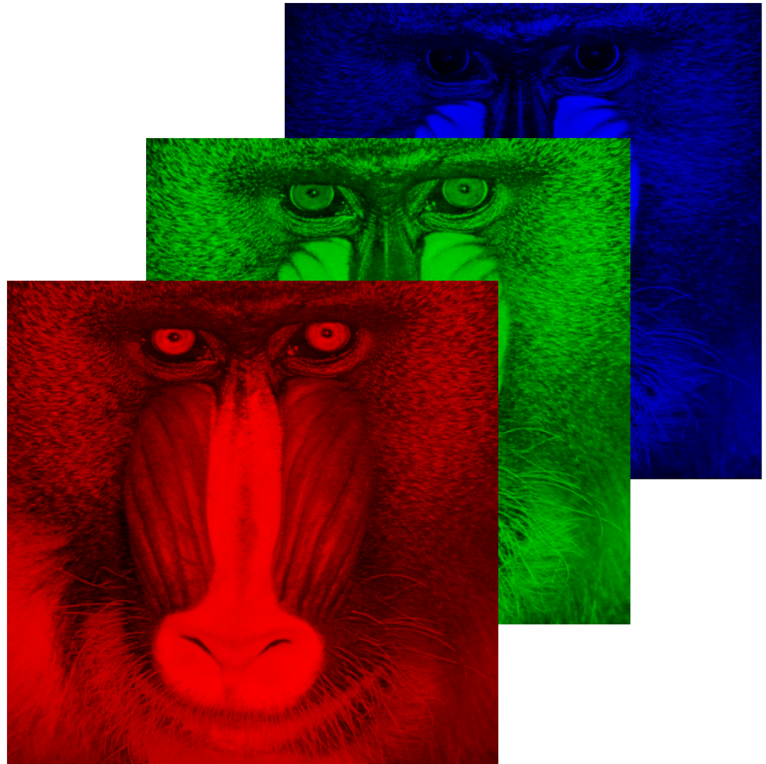


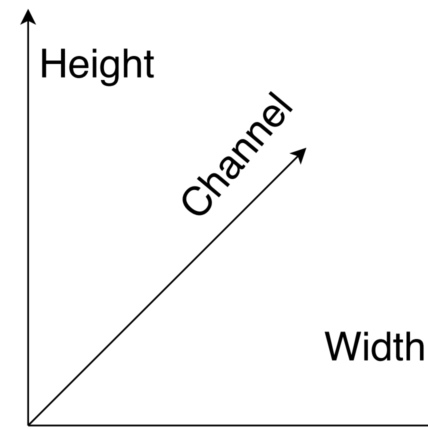


3D: (Batch, Height, Width)

4D: (Batch, Channels=1, Height, Width)

Tensors for RGB color images

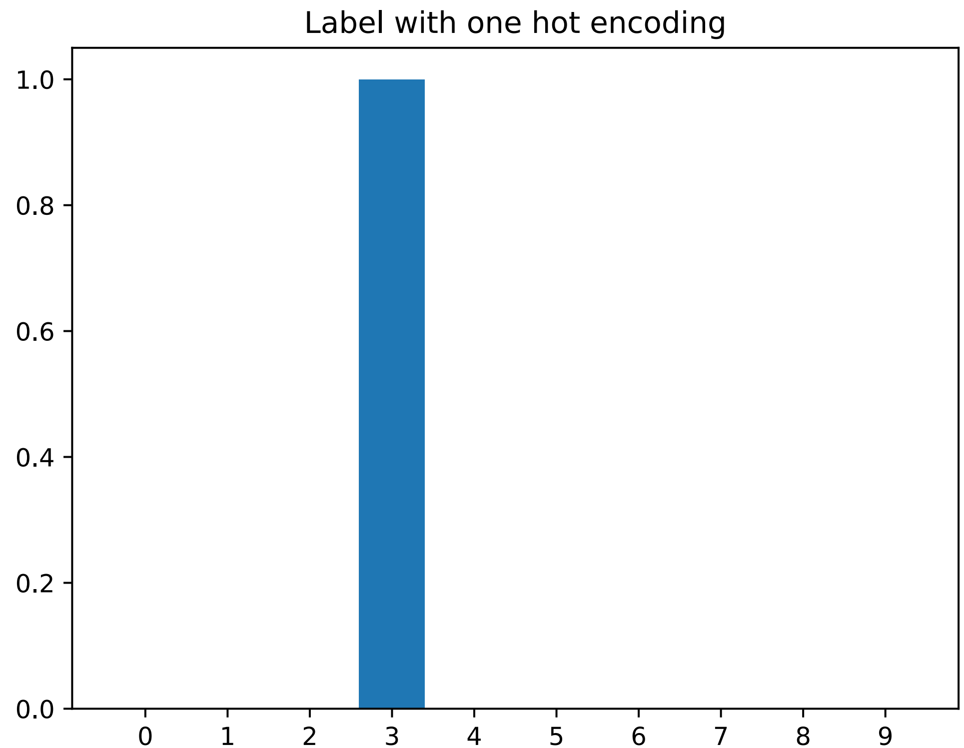




(Channel, Height, Width)

Categorigal distribution for classifiers

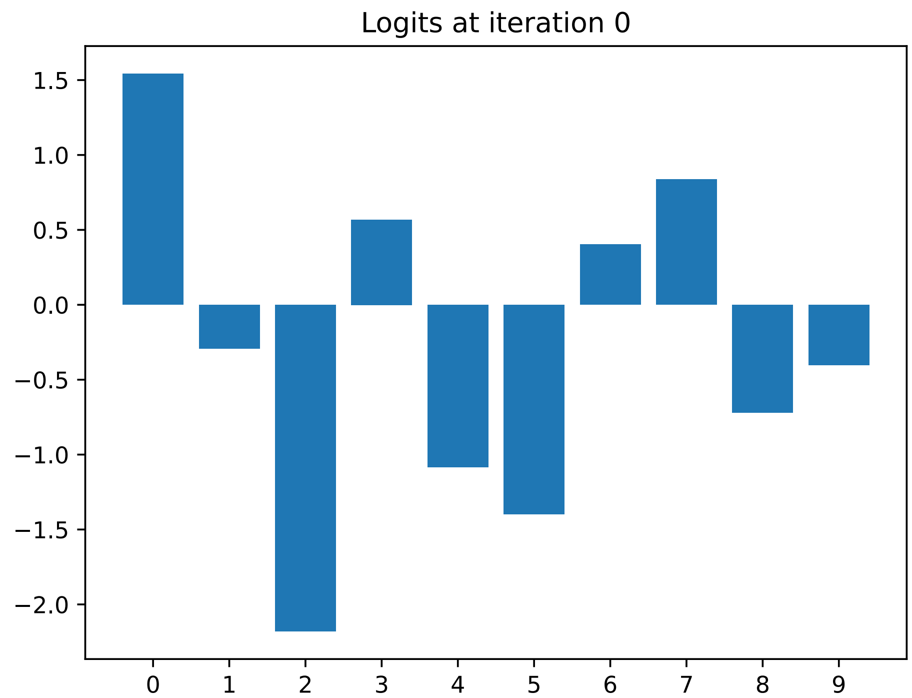
- **Let's look at one example of digit "3"**
- **Probability 1 for the correct class**
- **Probability 0 for the other classes**
- **This is also called a *one-hot embedding***



Logits and Softmax

- **Classifier initially outputs a vector of random numbers**
- **Normalize these to probability distribution with the Softmax function**

$$\text{softmax}(\mathbf{z}) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$



Cross-entropy loss for categorical distribution

- **Binary cross-entropy (Lecture 1)**

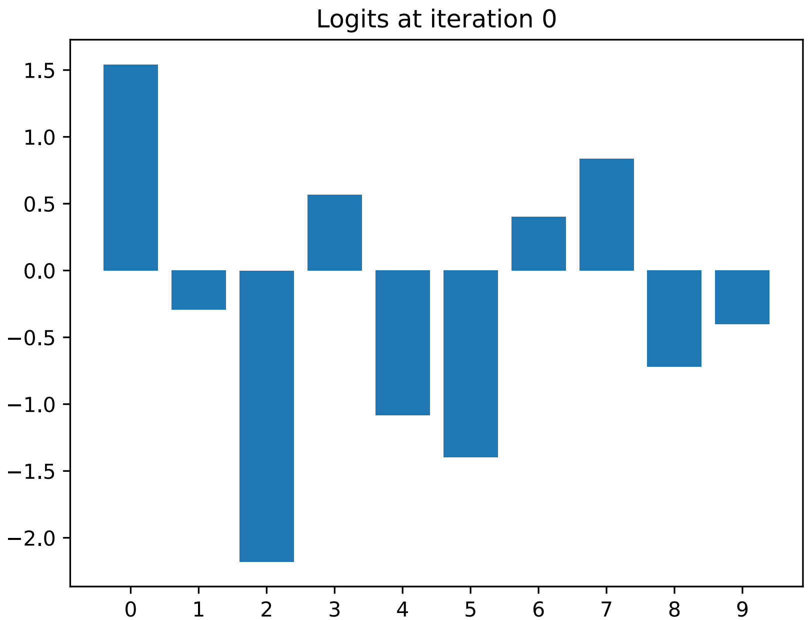
$$L_{\text{BCE}} = -\mathbb{E}[\mathbf{y} \log \hat{\mathbf{y}} + (1 - \mathbf{y}) \log(1 - \hat{\mathbf{y}})]$$

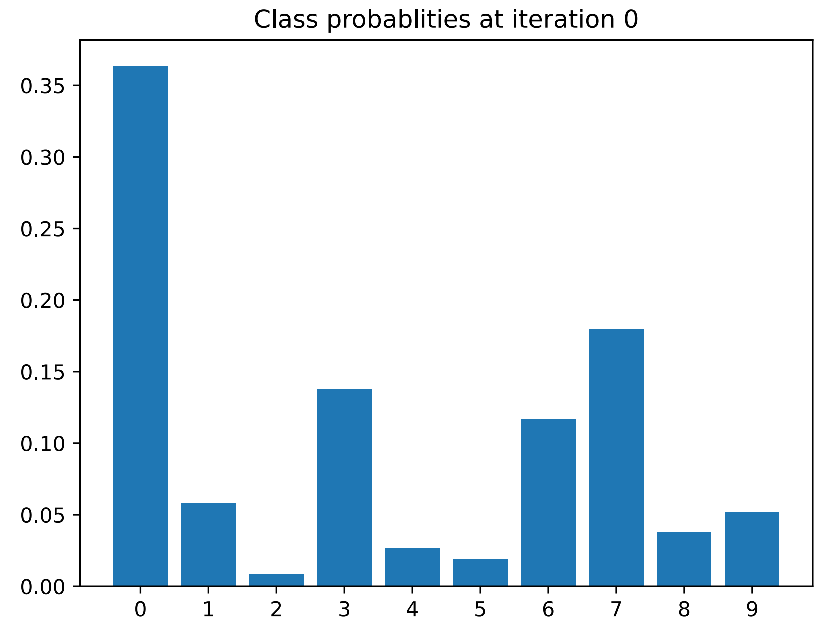
- **Categorical cross-entropy**

$$L_{\text{CCE}} = -\mathbb{E}\left[\sum_{i=1}^{K} \mathbf{y}_i \log \hat{\mathbf{y}}_i\right]$$

- **Probability of other classes is zero!**

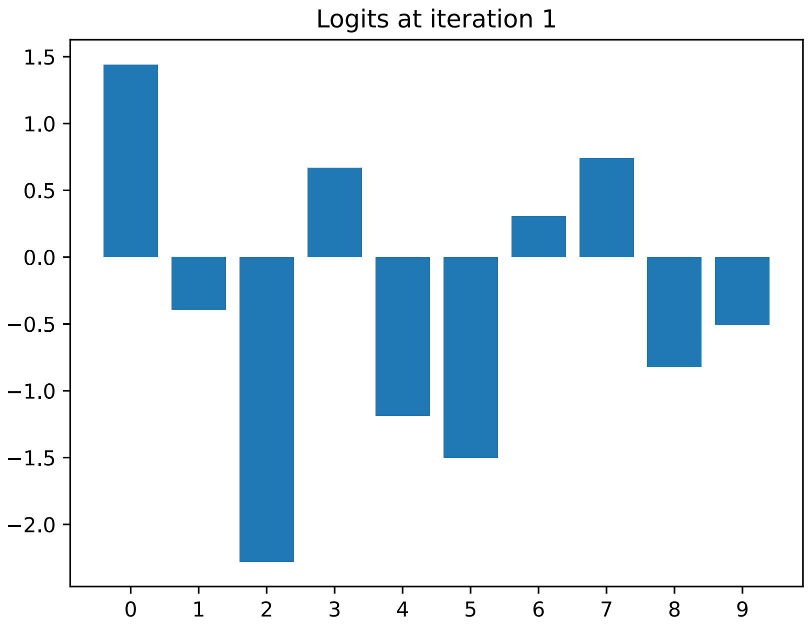
Fitting softmax for one example

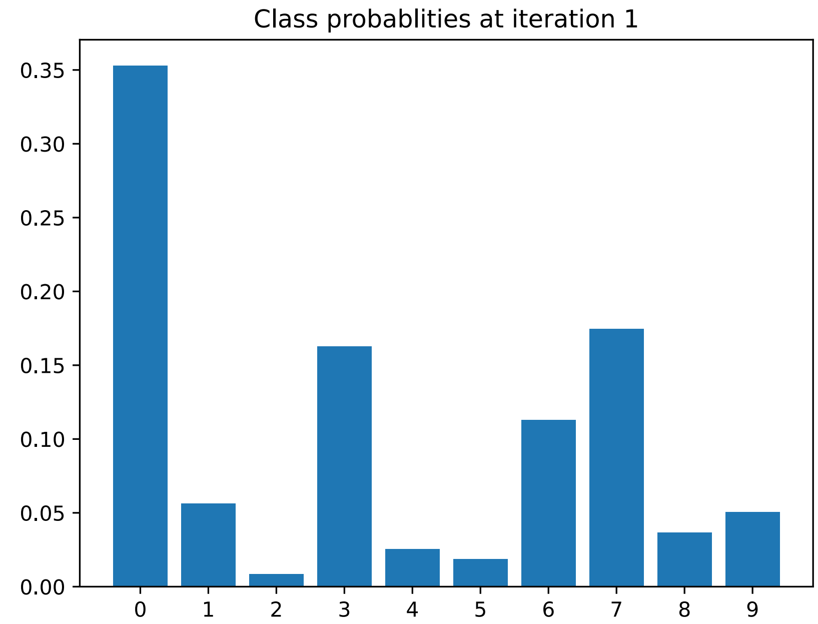




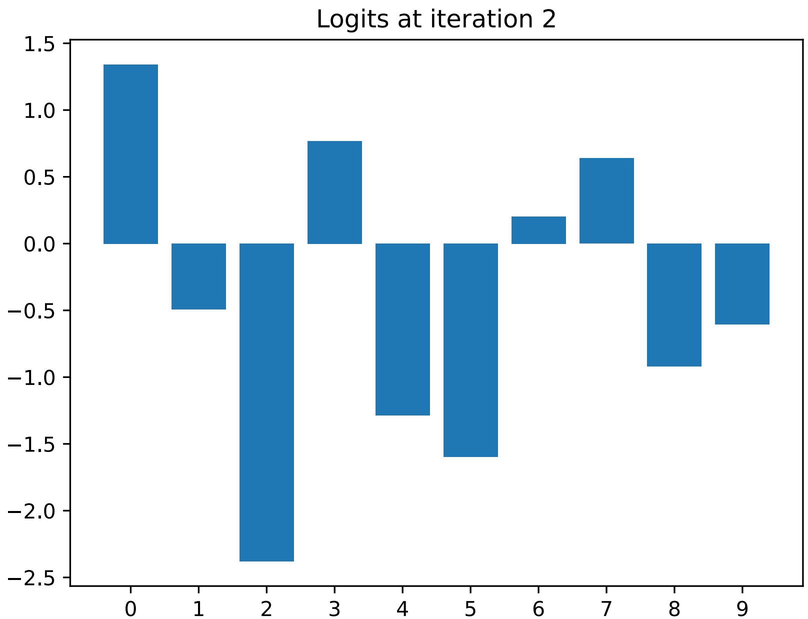
Aalto University
School of Electrical
Engineering

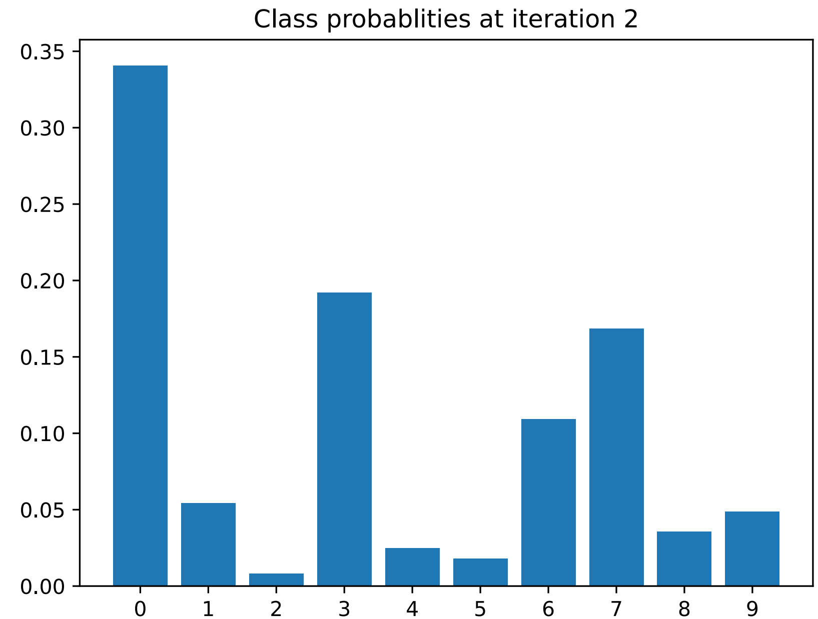
Fitting softmax for one example



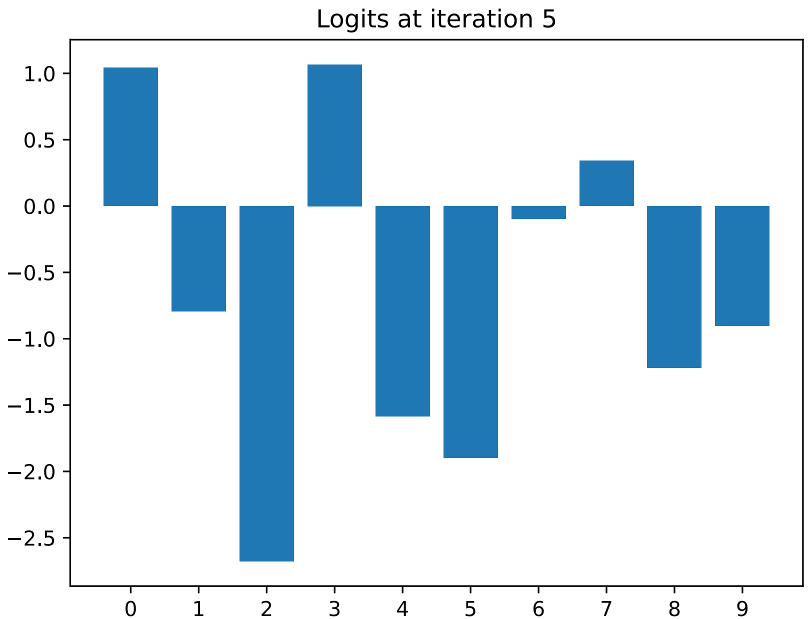


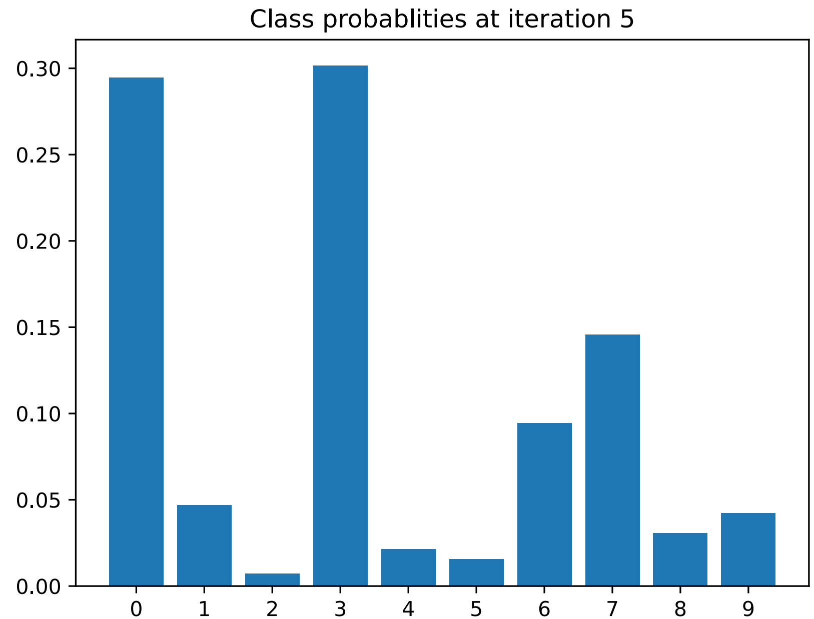
Fitting softmax for one example



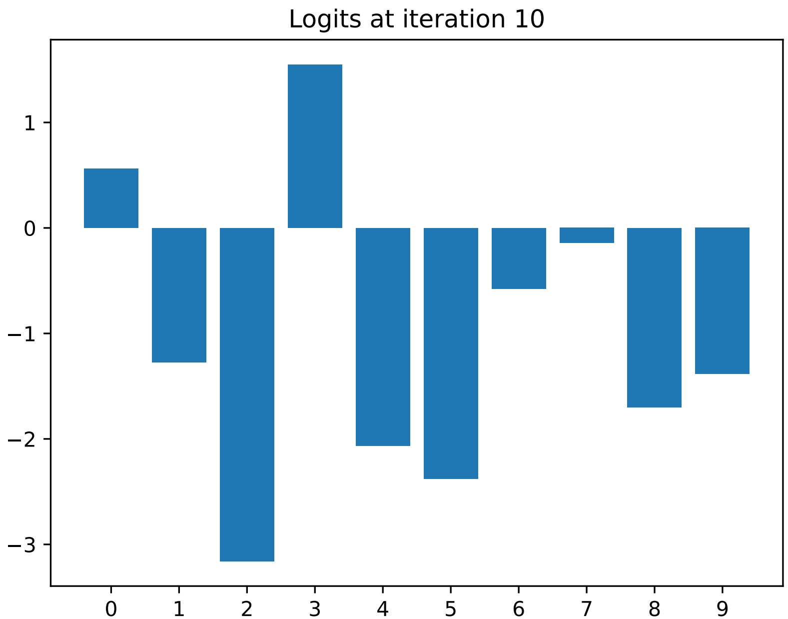


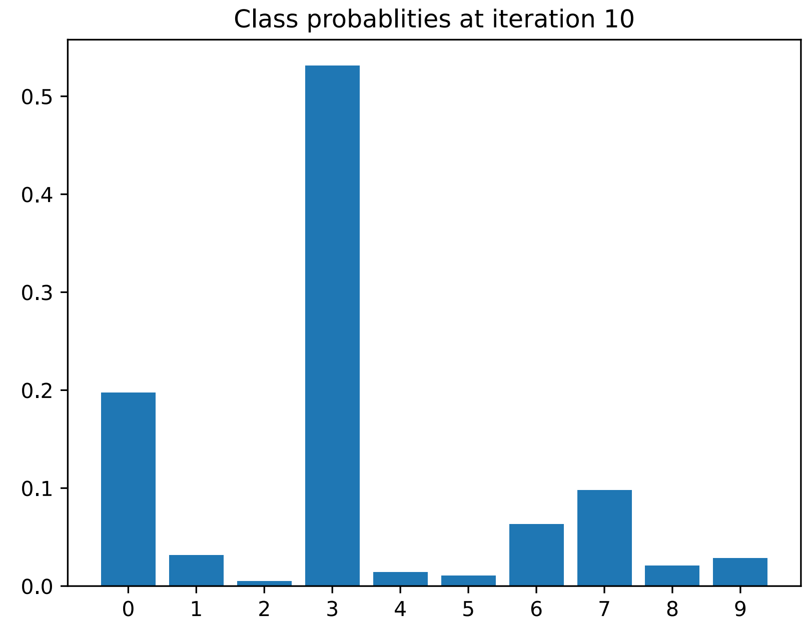
Fitting softmax for one example



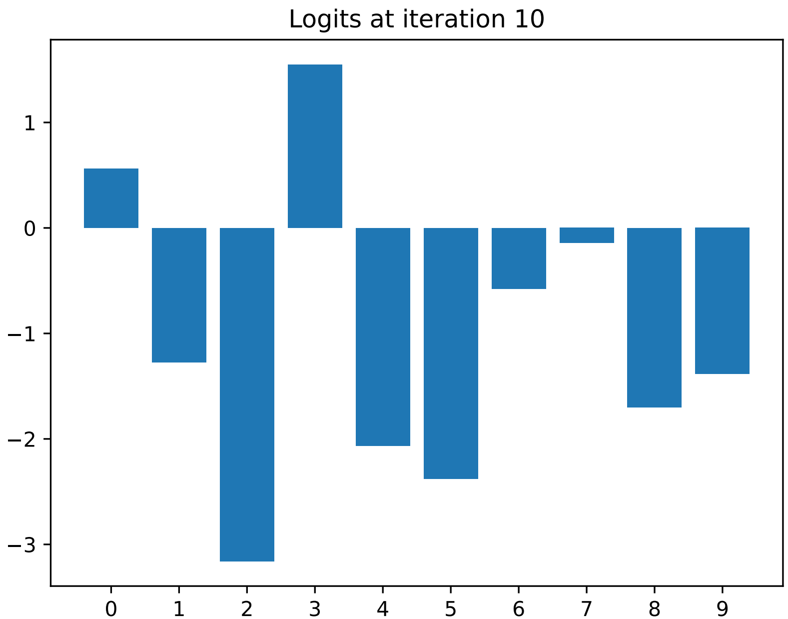


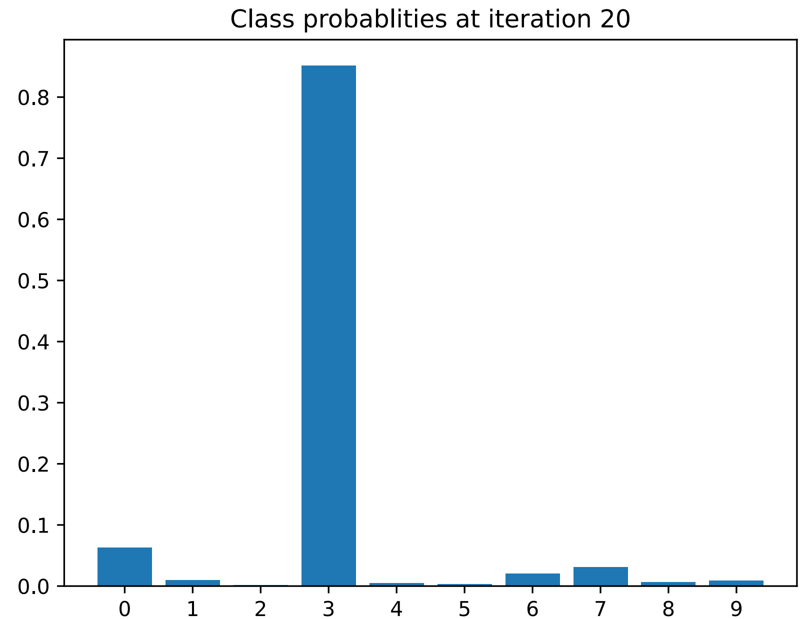
Fitting softmax for one example



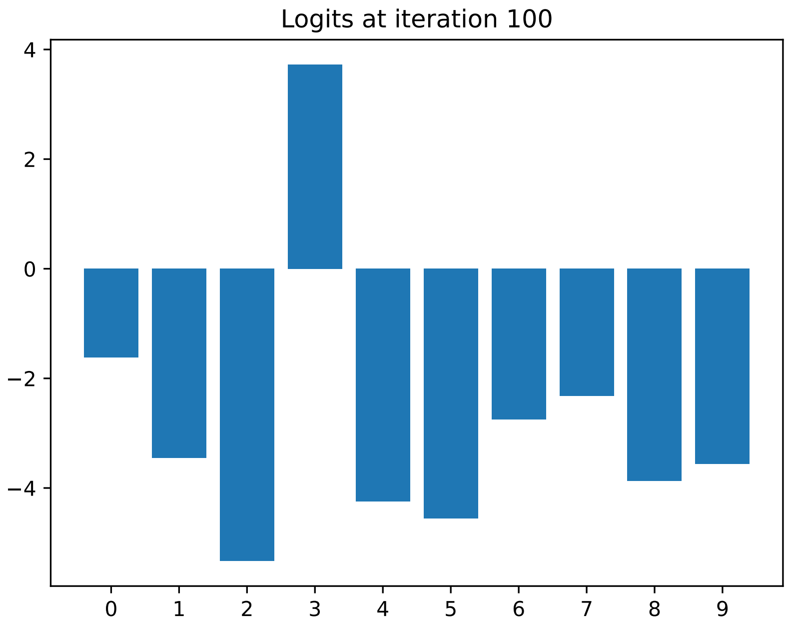


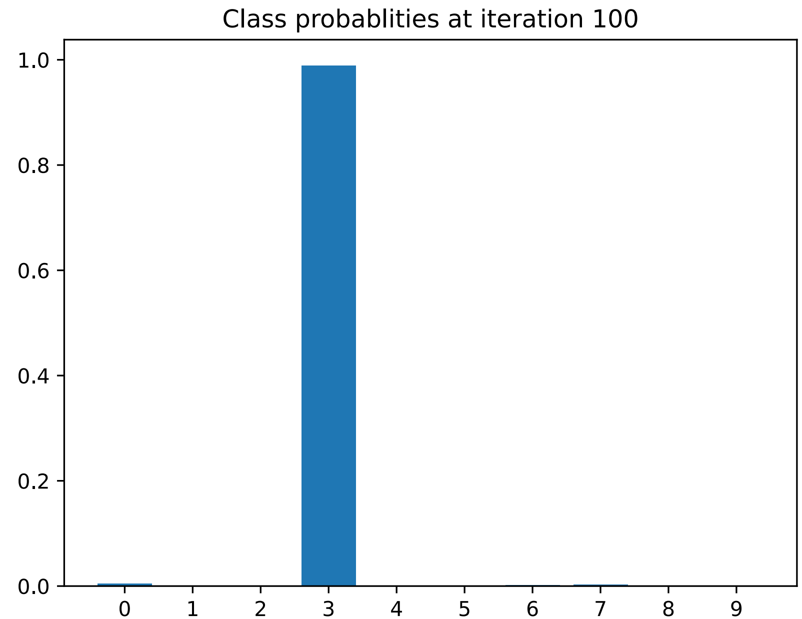
Fitting softmax for one example





Fitting softmax for one example





Aalto University
School of Electrical
Engineering

Categorical distribution for text

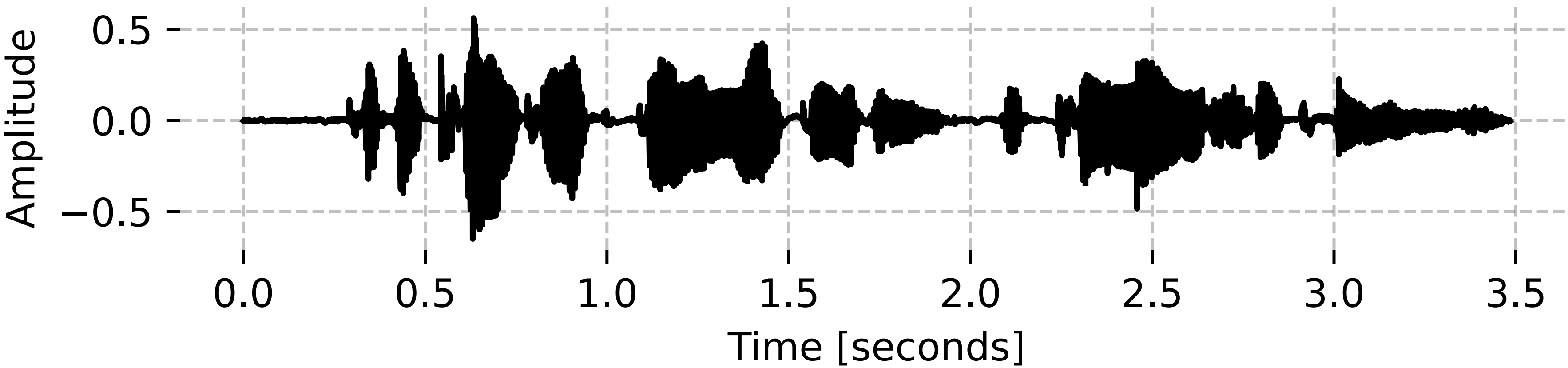
- You can use similar one-hot embeddings to encode text or other symbols
- More about this later on the language modeling Lecture 6

Break

- Questions?
- Next: audio representations

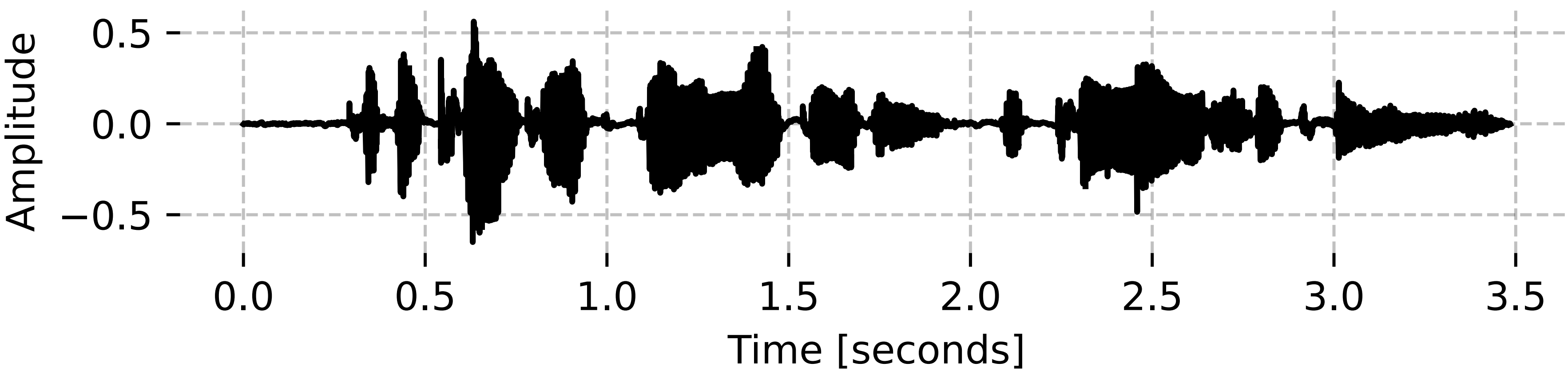
Audio - waveform

- **Sequence of amplitude values sampled at regular intervals (e.g., 48 kHz or 16 kHz)**
- **Monophonic (single channel) audio is a vector, where time corresponds to dimension**



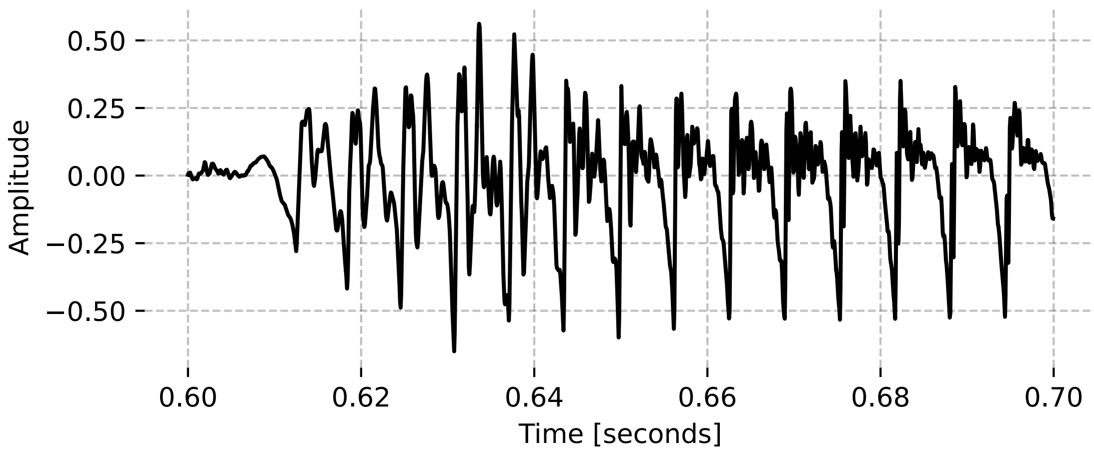
Audio - waveform

- For example, one second of audio at 16 kHz would be a 16 000 dimensional vector (remember the curse of dimensionality?)
- How to work on continous streams of audio?
- Hard to see anything on this zoom-level



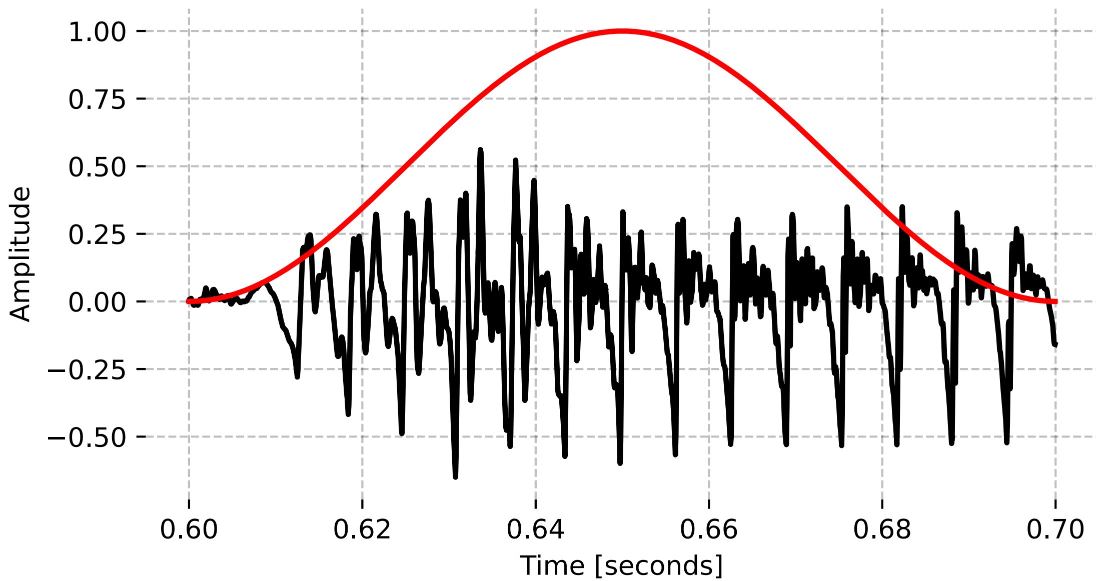
Audio and short-time frames

- **Zoom in on 100ms of audio**
- **In this example, voiced speech is (quasi) periodic, we can do Fourier analysis!**



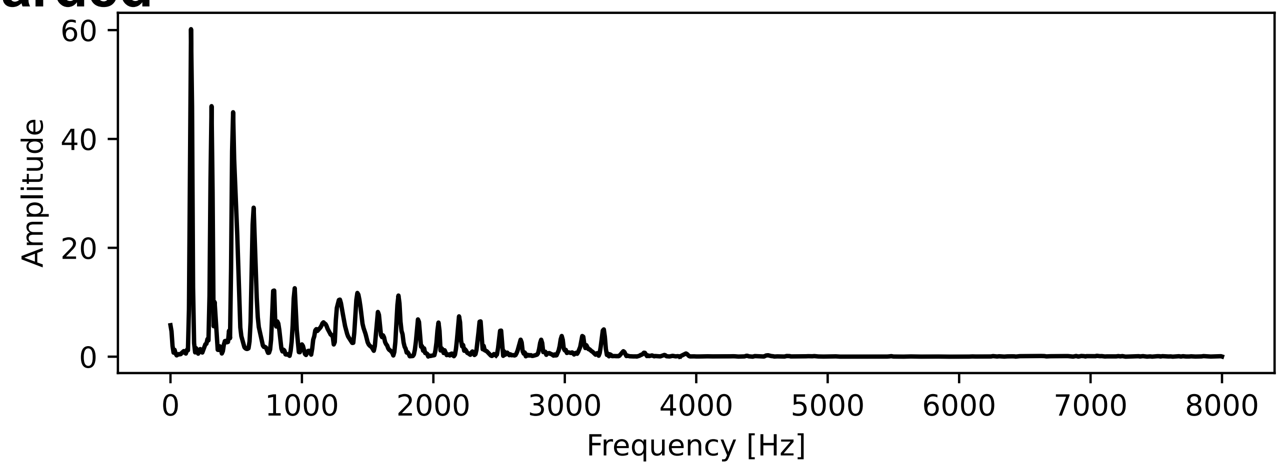
Audio and short-time frames

- Multiply with a tapered window (half cosine, Hann, etc.)
- Apply Discrete Fourier Transform (DFT)
- In practice, Fast Fourier Transform (FFT)



Audio and short-time spectrum

- **Plot frequency magnitudes on linear scale - harmonic overtone series shows up!**
- **Remember: FFTs are complex-valued, phase information was discarded**



Fourier transform and complex numbers

- **Fourier transform outputs sinusoids (i.e., complex numbers on the unit circle)**
- **Remember phase when processing signals (synthesis, coding, enhancement)**
- **Ignore phase for detection tasks (speech recognition, etc.)**

Real, Imaginary

$$z = x + iy$$

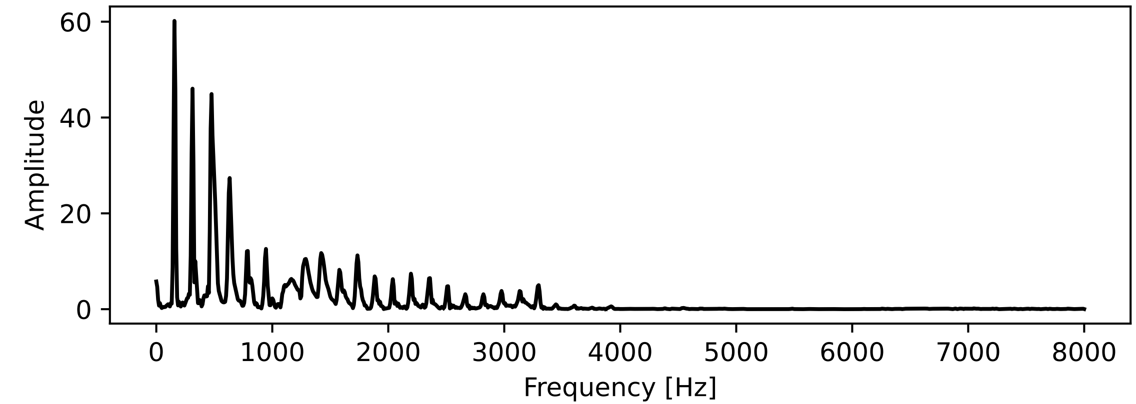
Amplitude, Phase

$$z = Ae^{i\theta}$$

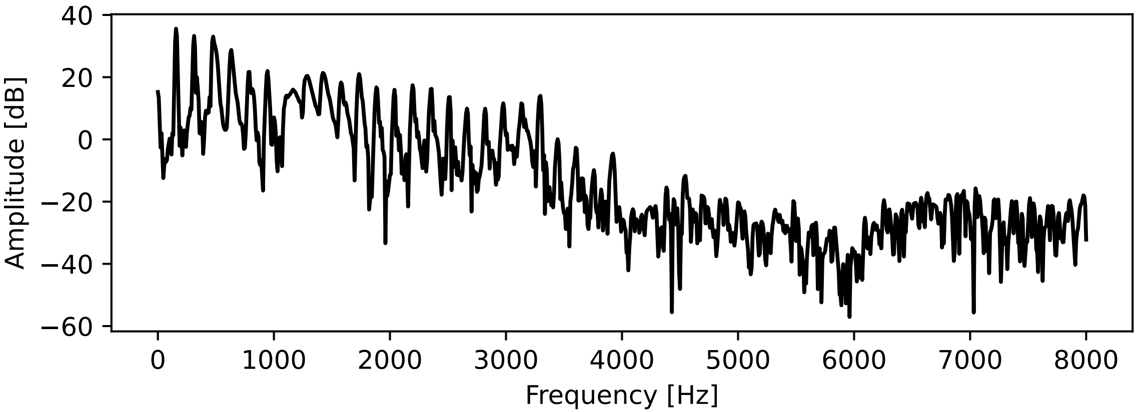
Frequency, Initial phase

$$z = Ae^{i(2\pi f + \theta_0)}$$

Decibels and log-scale spectrum



$$A = |z|$$

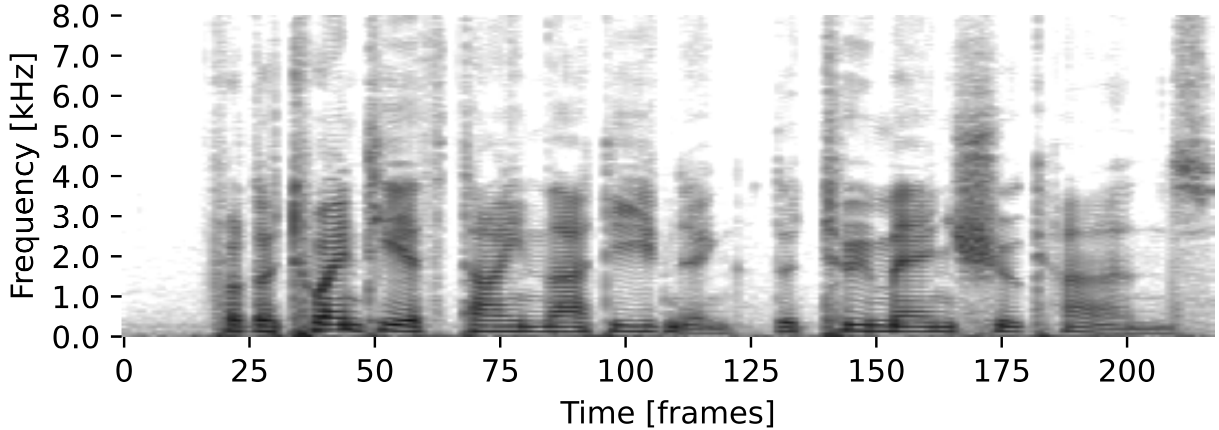


$$A_{\mathrm{dB}} = 20 \log_{10}(|z| + \varepsilon)$$

Aalto University
School of Electrical
Engineering

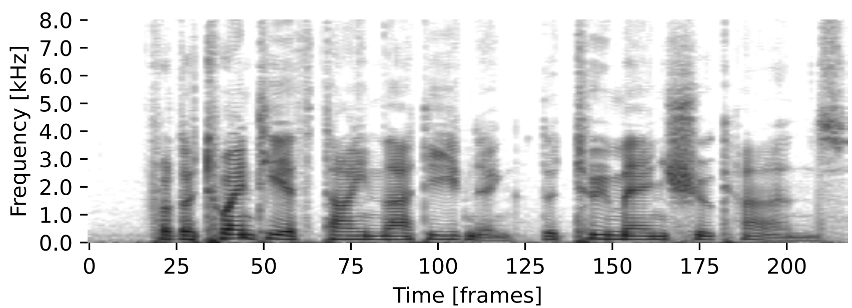
Audio spectrograms

- **Short-Time Fourier Transform (STFT)**
- **Sliding window, apply FFT to each frame**
- **Log-magnitudes (dB-scale) are usually best for visualisation**



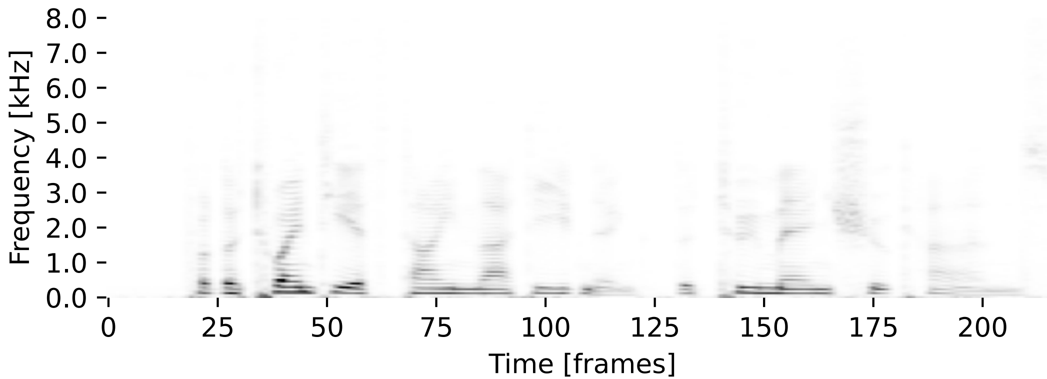
Audio spectrograms

- Spectrogram is a matrix, shape (T, F)
- T = Number of time frames
- F = Number of frequency bins
- Spectrogram is an image (you are looking at it)
- For machine learning models: 4D tensor $(B, C = 1, F, T)$



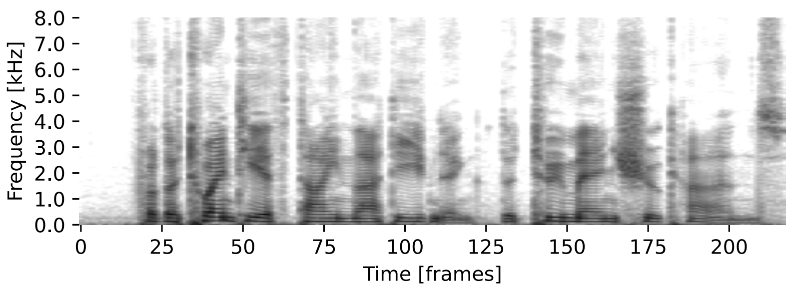
Audio spectrograms - pitfalls

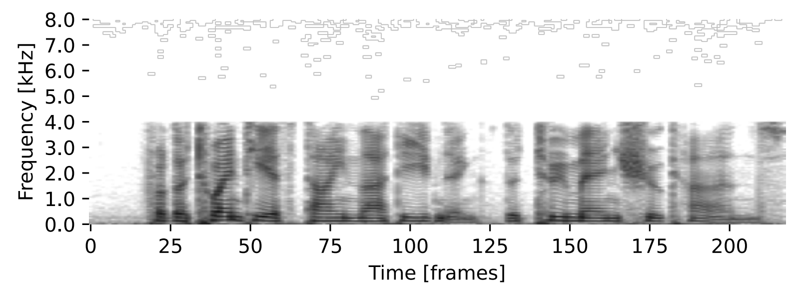
- **Linear amplitude spectrogram (below) are correct shape, but the dynamic range is too large to see much**
- **Logarithms of zeros are numerical trouble (remember to add a small value)**



Filtering in frequency domain

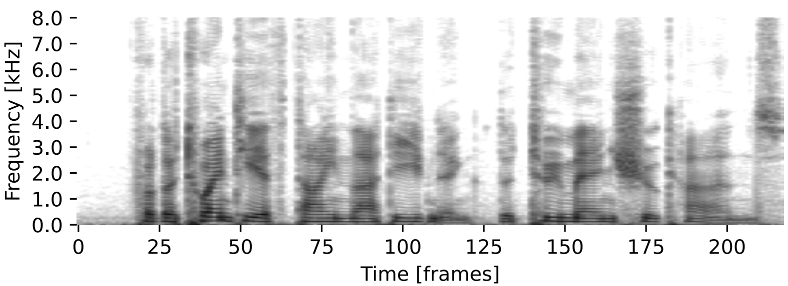
- **Filtering in the frequency domain corresponds to multiplying STFT amplitudes**
- **Exercise 2: implement a low-pass filter**

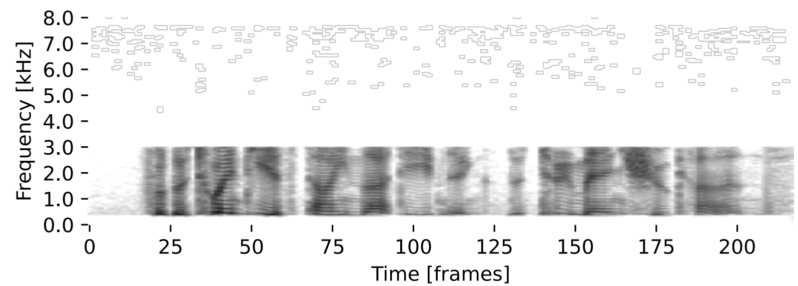




Filtering in frequency domain

- **Filtering in the frequency domain corresponds to multiplying STFT amplitudes**
- **Exercise 2: implement a band-pass filter**





Aalto University
School of Electrical
Engineering

End of Lecture 2

- Questions?