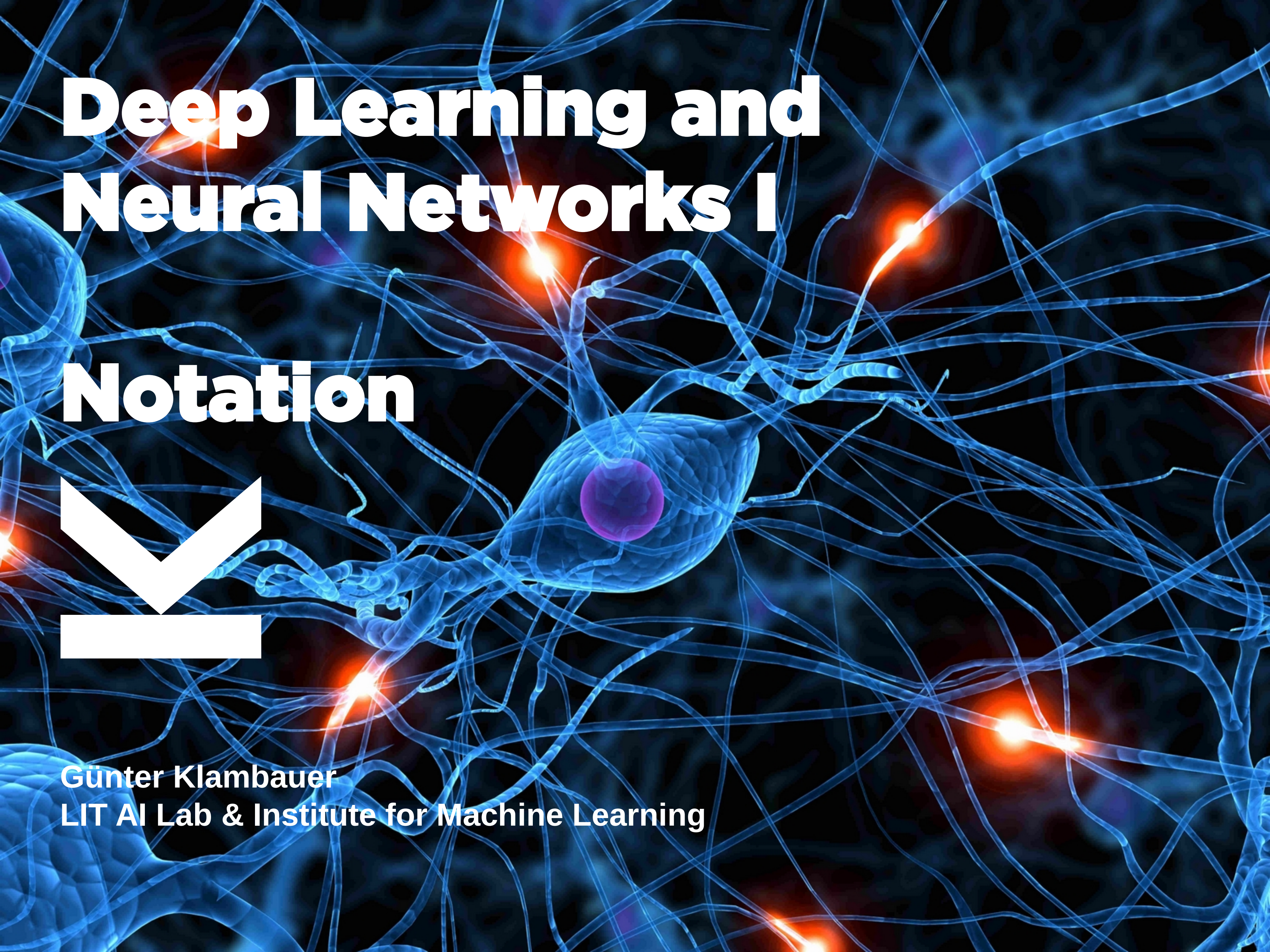
JOHANNES KEPLER
UNIVERSITY LINZ

JOHANNES KEPLER
UNIVERSITY LINZ

Deep Learning and Neural Networks I

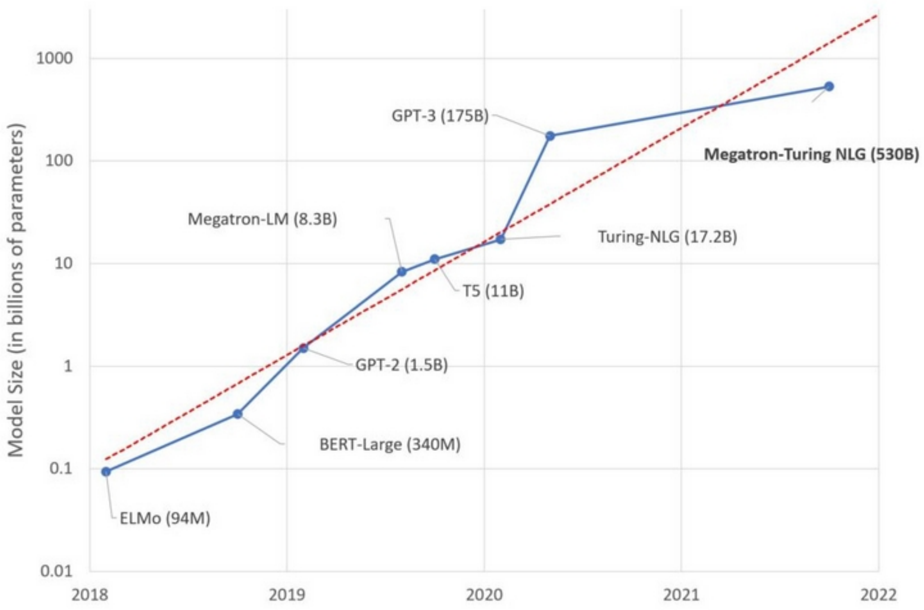
Notation



Günter Klambauer
LIT AI Lab & Institute for Machine Learning

Current topics

- Microsoft and NIVIDIA have just announced a deep language model with 530B parameters!
  - Parallelized across thousands of GPUs
- Kunihiko Fukushima just won the 2021 Bower Award for Achievement in Science
  - ML community discusses: https://www.reddit.com/r/MachineLearning/comments/q76js4/schmidhuber_pays_tribute_to_kunihiko_fukushima/
- We are looking for a technician for a Deep Learning cluster
  - Please contact: Doris Kaiserrainer kaiserreiner@ai-lab.jku.at if you are interested!
    - Email subject “Technician”
  - Details on Moodle



Source: https://www.microsoft.com/en-us/research/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/

Notation example: Supervised learning

$\boldsymbol{X}$			$\boldsymbol{y}$
$\boldsymbol{x}^1$	0.13	0.03	0.35
$\boldsymbol{x}^2$	*0.15*	0.14	0.57
	0.79	0.19	0.87
	0.48	0.28	1.21
	0.93	0.43	0.48
$\boldsymbol{x}^n$	0.43	0.41	0.70
	0.62	0.63	-0.44
	0.69	0.69	-1.05
	0.19	0.79	-1.29
	0.23	0.85	-1.11
$\boldsymbol{x}^N$	0.11	0.99	0.32

- Our supervised data set is:
  - Samples are rows in the data matrix $\boldsymbol{X}$:

    $$\boldsymbol{X} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N)$$

  - Features are columns of $\boldsymbol{X}$:
    - Single features,e.g.: $x_{21} = 0.15$
    - Feature vector, e.g.: $\boldsymbol{x}_{.1}$

  - Scalar label for each data point:

    $$\boldsymbol{y} = (y^1, \ldots, y^N)^T$$

Data objects

- $\boldsymbol{x}^n \in \mathbb{R}^D$ or $\boldsymbol{x} \in \mathbb{R}^D$: the n-th input data point or a general data point, respectively. A column vector. Sometimes also used for the inputs of a particular neural network layer, then the dimensions could be $\boldsymbol{x} \in \mathbb{R}^J$.

- $y^n \in \mathbb{R}$ or $y \in \mathbb{R}$: a scalar label for the n-th data point or a general label y, respectively.

- $\boldsymbol{y}^n \in \mathbb{R}^K$ or $\boldsymbol{y} \in \mathbb{R}^K$: For multi-class of multi-task problems, this is a *label vector* for the n-th data point or a general label vector, respectively. A column vector.

- $\hat{y} \in \mathbb{R}$ or $\hat{\boldsymbol{y}} \in \mathbb{R}^K$: the predicted scalar label or – for a multi-class problem – the predicted label vector, respectively.

- $p \in \mathbb{R}$ or $\boldsymbol{p} \in \mathbb{R}^K$: same as above if outputs can be interpreted probabilistically, the predicted scalar label or – for a multi-class problem – the predicted label vector.

- $\boldsymbol{a} \in \mathbb{R}^I$: *activations* of a neural network.

- $\boldsymbol{s} \in \mathbb{R}^I$: *pre-activations*, also called *netI*, of a neural network.

Sizes and dimensions

- N: number of training examples, i.e. objects or samples, in the *training data set*. Running index: n.
- D: number of input units which is also the number of features that a sample has. Running index: d
- M: number of test or validation examples, i.e. objects or samples, in the *test data set*. Running index: m with $N + 1 \leq m \leq N + M$.
- K: number of output units. Running index: k.
- L: number of layers in a network without counting the input layer. Running index: l.
- J: input dimension of a general neural network layer. Running index: j.
- I: output dimension of a general neural network layer. Running index: i.

Stacked data and labels

- $\boldsymbol{X} \in \mathbb{R}^{N \times D}$: input data matrix. The rows represent objects, i.e. samples. We assume that objects, i.e. samples, are represented or described by *feature vectors* $\boldsymbol{x}^n$.
- $\boldsymbol{y} \in \mathbb{R}^N$: the scalar labels of all samples stacked to a column vector.
- $\boldsymbol{Y} \in \mathbb{R}^{N \times K}$: for multi-class or multi-task problems, stacked labels yield a *label matrix*.

Parameter objects

- $\boldsymbol{w} \in \mathbb{R}^D$: a *weight* or *parameter* vector of a simple machine learning method, such as linear regression. A column vector.
- $\boldsymbol{W} \in \mathbb{R}^{I \times J}$: a *weight* or *parameter* matrix of a learning method mapping from an input space with dimension J to an output space with dimension I.
- $\boldsymbol{b} \in \mathbb{R}^I$: a bias vector.
- $\boldsymbol{\theta}$: a set of parameters of a probabilistic model.

Multiple layers

- $n_h^{[l]}$: number of hidden units of the l-th layer. Thus, $D = n_h^{[0]}$ and $K = n_h^{[L+1]}$.
- $\boldsymbol{a}^{[l]} \in \mathbb{R}^{n_h^{[l]}}$: *activations* of a neural network in the l-th layer.
- $\boldsymbol{s}^{[l]} \in \mathbb{R}^{n_h^{[l]}}$: *pre-activations*, also called *netI*, of a neural network in the l-th layer.
- $\boldsymbol{W}^{[l]} \in \mathbb{R}^{n_h^{[l-1]} \times n_h^{[l]}}$: the weight matrix connecting the $(l-1)$-th layer with the l-th layer.
- $\boldsymbol{b}^{[l]} \in \mathbb{R}^{n_h^{[l]}}$: the bias vector in the l-th layer.

Common transformations

$$\boldsymbol{w}^T\boldsymbol{x} + b = w_1 x_1 + \ldots w_D x_D + b. \tag{1}$$

$$\boldsymbol{s} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}. \tag{2}$$

$$\boldsymbol{a} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}). \tag{3}$$

$$\boldsymbol{w}^T\boldsymbol{w} = \sum_{i=1}^{I} w_i^2 = \|\boldsymbol{w}\|^2, \tag{4}$$

Functions

- $f : \mathbb{R} \to \mathbb{R}$: a non-linear *activation function* that is applied element-wise to a vector or matrix. Sometimes also denoted as ϕ.
- $g(\boldsymbol{x}; \boldsymbol{w})$: a machine learning model with input $\boldsymbol{x}$ and parameters $\boldsymbol{w}$.
- $p(\boldsymbol{x}; \boldsymbol{\theta})$: a probabilistic model with data point $\boldsymbol{x}$ and set of parameters $\boldsymbol{\theta}$.
- $L(y, g(\boldsymbol{x}; \boldsymbol{w}))$ or $L(y, \hat{y})$: a loss function L.
- $R_{\mathrm{emp}}(\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{w})$: an empirical error or risk function. Typically, the empirical error is an average of the loss function for a single training data point. For neural networks, this serves as a *cost function* that is minimized.
- $R_{\mathrm{emp}}(\boldsymbol{w})$: an empirical error or risk function. The same as above only that occasionally the dependency on $\boldsymbol{X}$ and $\boldsymbol{y}$ is dropped to keep the notation uncluttered.

Derivatives and matrix layout

We use the so-called *numerator layout* of matrix calculus:

- $\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}}$: a row vector of length W according to the definition of the Jacobian.
- $\frac{\partial \boldsymbol{a}}{\partial x}$: column vector
- $\frac{\partial a}{\partial \boldsymbol{x}}$: row vector
- $\frac{\partial \boldsymbol{a}}{\partial \boldsymbol{x}}$: matrix with as many rows as dimension of $\boldsymbol{a}$, as many columns as dimesion of $\boldsymbol{x}$.
- $\frac{\partial a}{\partial \boldsymbol{X}}$: dimension of transposed $\boldsymbol{X}$

With the *Nabla operator* ∇, we can switch to *denominator layout*:

- $\nabla R(\boldsymbol{w})$ or $\nabla_{\boldsymbol{w}} R(\boldsymbol{w})$: a column vector of length W. $\nabla_{\boldsymbol{w}} R(\boldsymbol{w}) = \left(\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}}\right)^T$

Distributions

- $\mathcal{U}(a, b)$: a uniform distribution in the interval $[a, b]$. This distribution has mean $\frac{1}{2}(a+b)$ and variance $\frac{1}{12}(b-a)^2$.
- $\mathcal{N}(\mu, \sigma^2)$: a normal distribution with mean μ and variance σ^2. Also commonly referred to as the Gaussian distribution.
- $\mathcal{B}(n, p)$: a binomial distribution with size parameter n and probability parameter p. Sometimes used as $\mathcal{B}(1, p)$ to denote Bernoulli distributions.