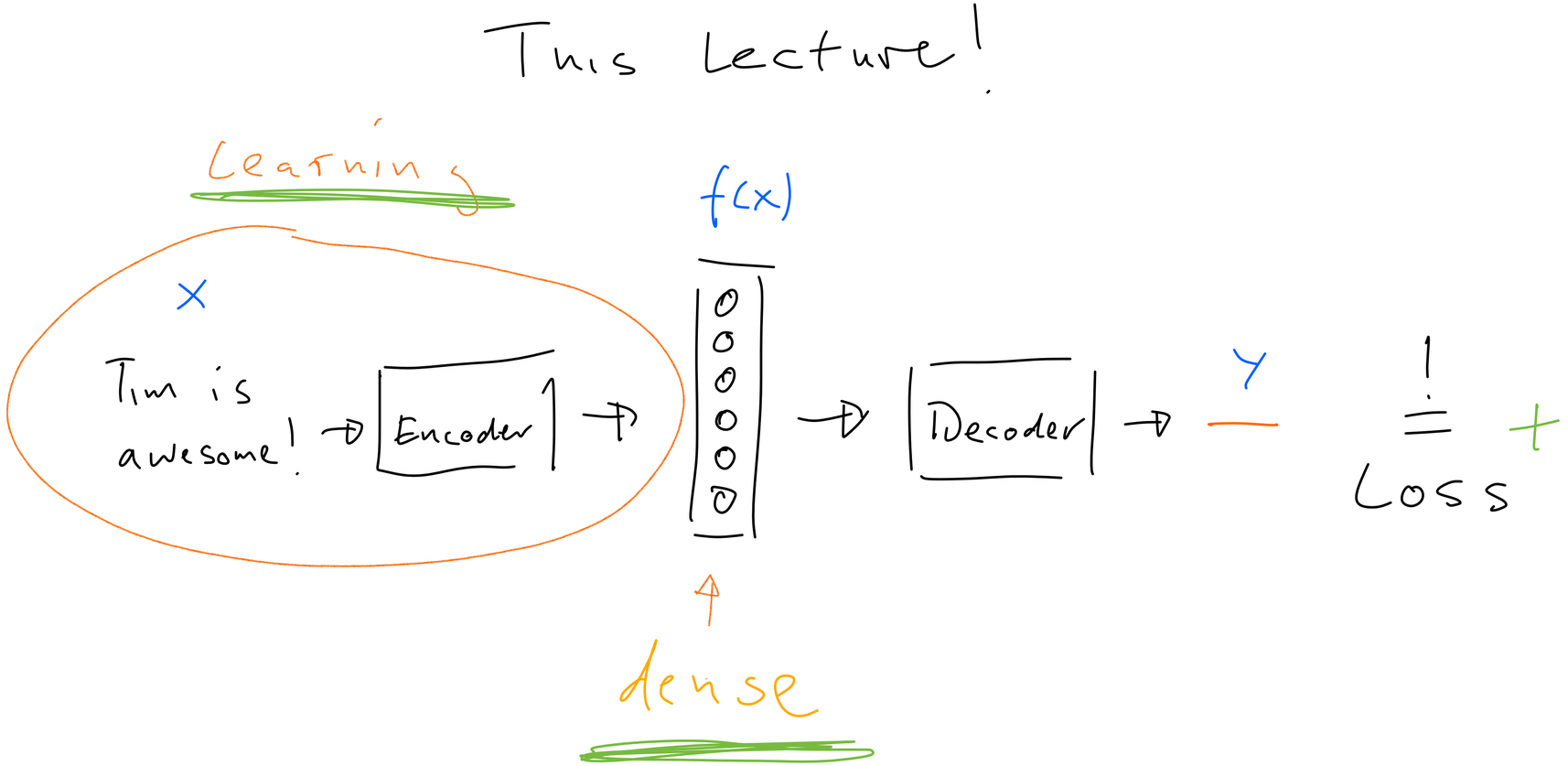
Introduction to Neural Networks and Backpropagation

Tim Rocktäschel & **Sebastian Riedel**

COMP0087 Natural Language Processing

Overview


This Lecture!
Learning
f(x)
X
Tim is awesome!
Encoder
Decoder
Y
!
=
Loss
+
dense

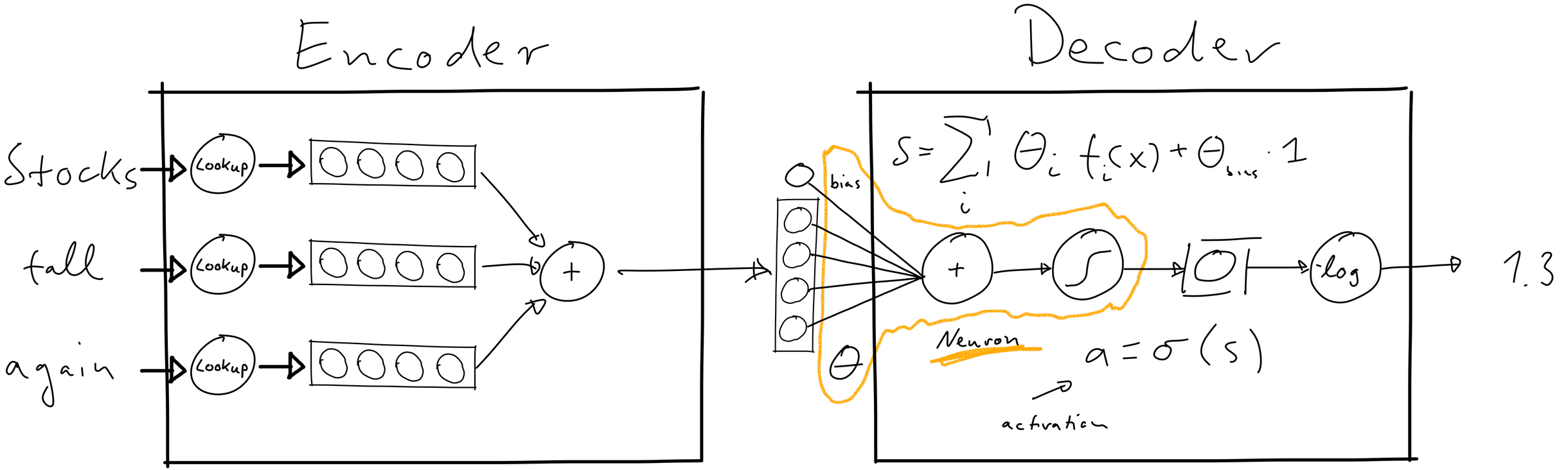
Task: Text Classification

St Pauli: the club that stands for all the right things ... except winning → **Sports**

Brexit: Are we running out of parliamentary time? → **Politics**

Stocks fall after Morgan Stanley earnings miss → **Business**

Computation Graph

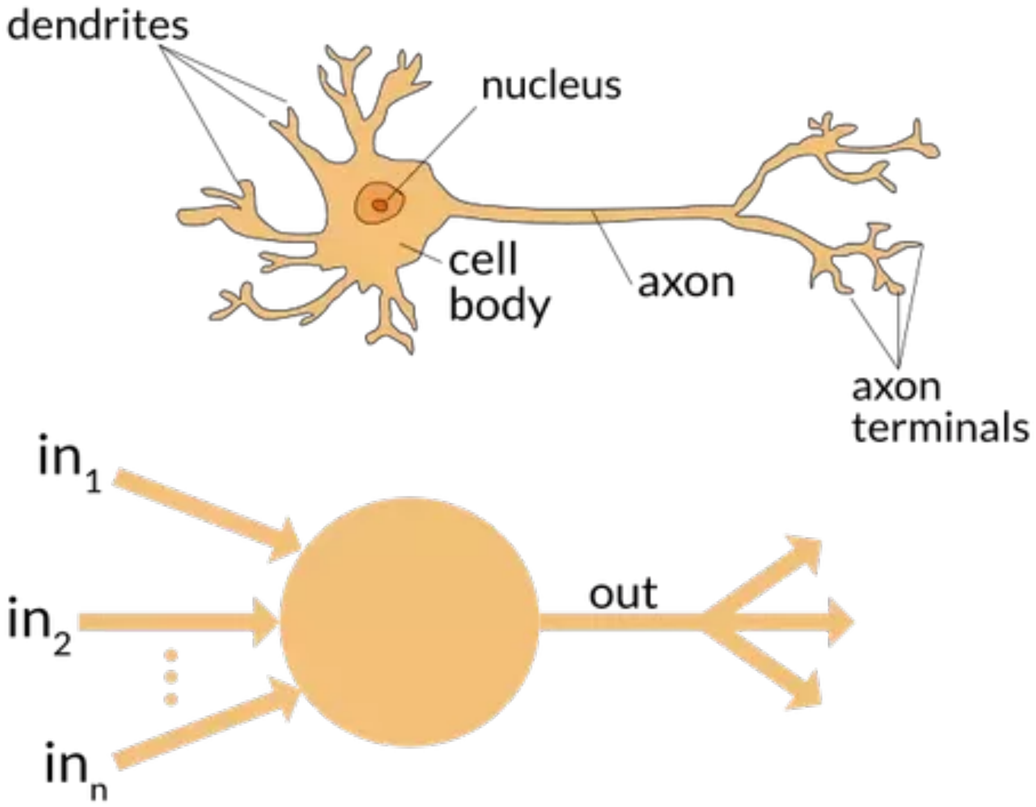


x

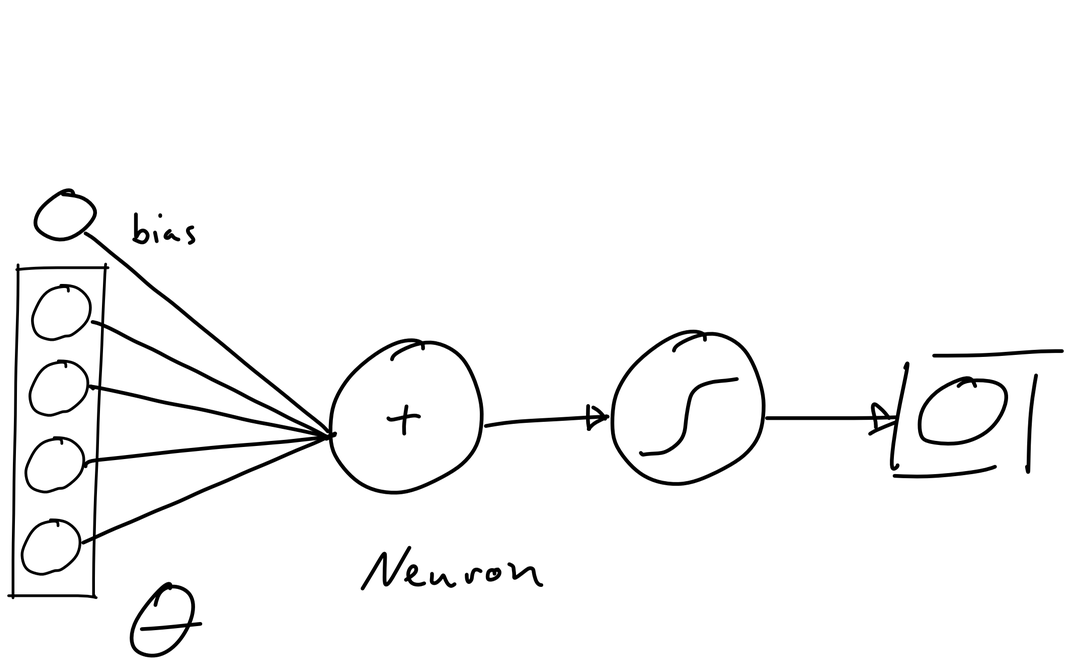
$f(x)$

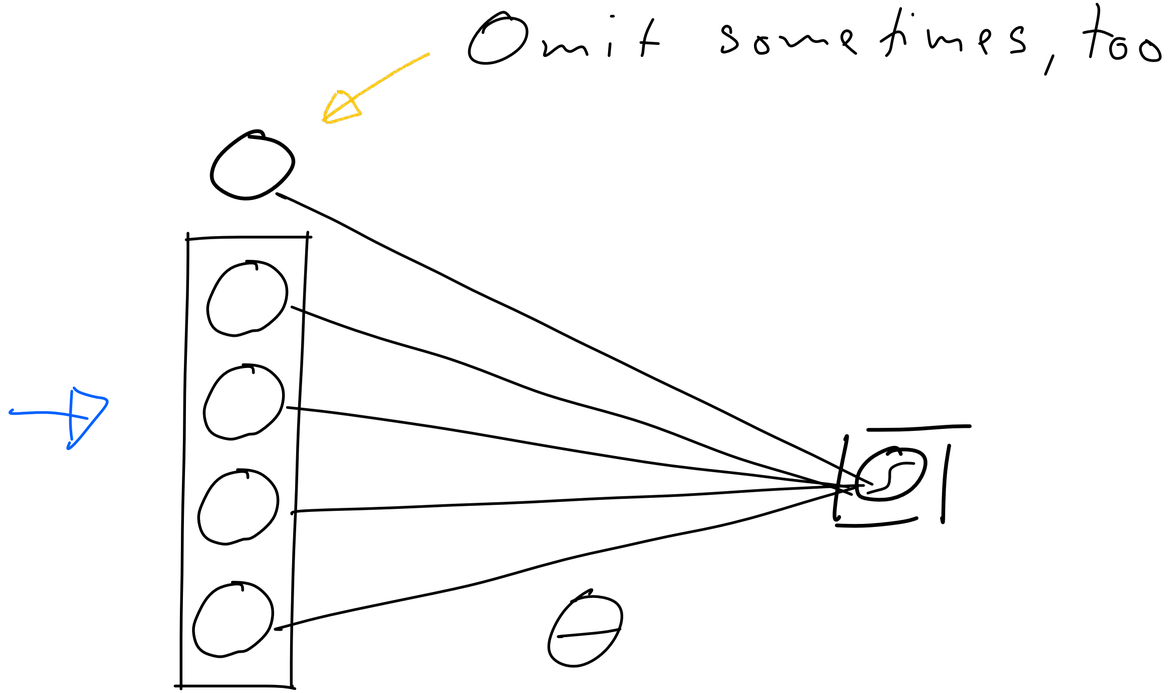
$L(x, y=\text{Biz}; \theta)$

Artificial and Biological Neurons



Simplified Notation

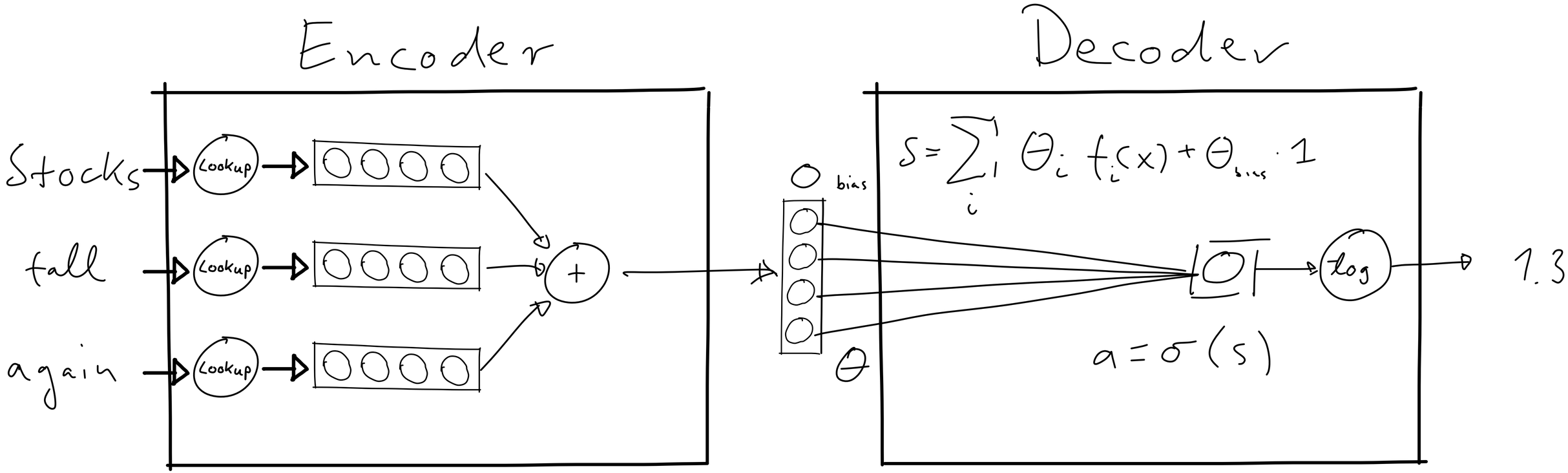




$$a = \sigma(\theta^T x)$$

$$a = \sigma(\theta^T x)$$

Computation Graph

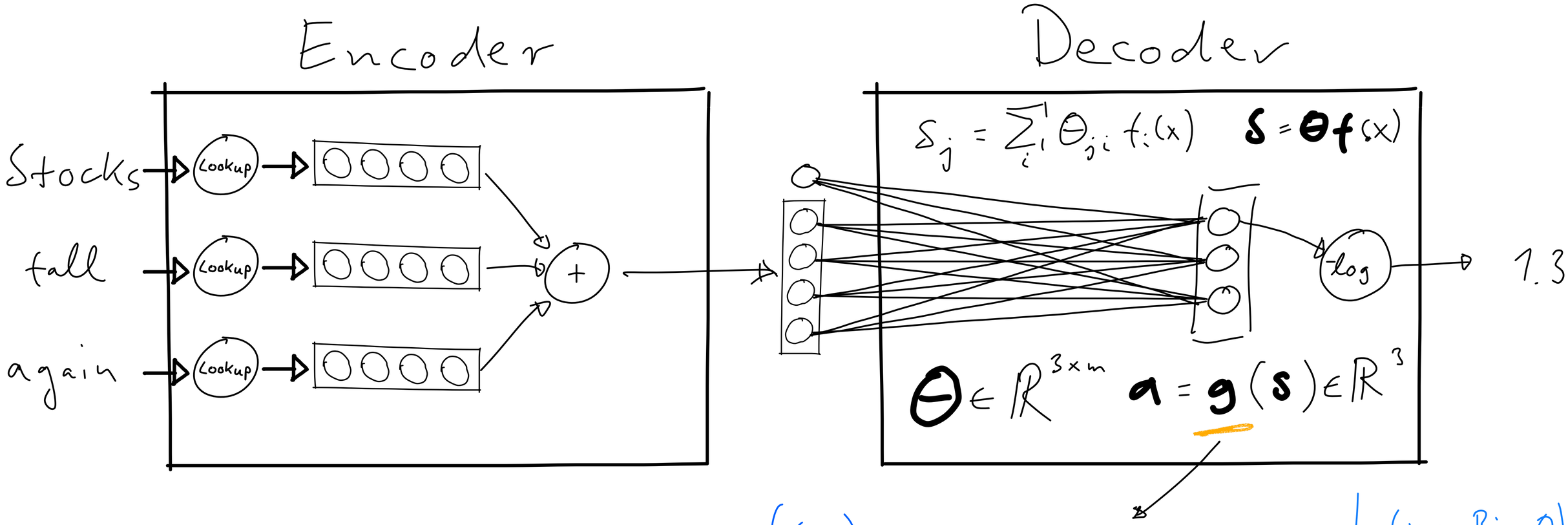


x

$f(x)$

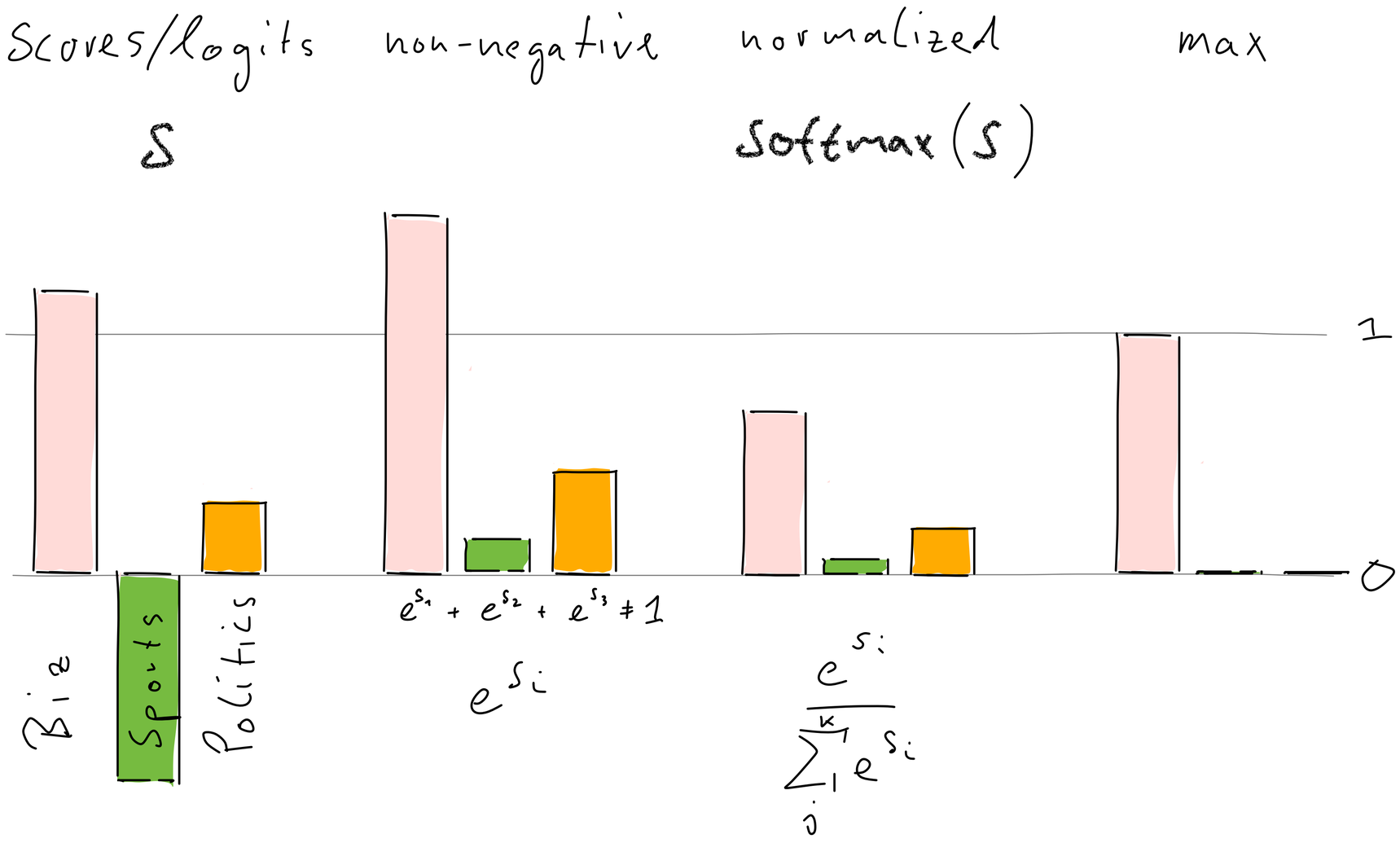
$L(x, y=\text{Biz}; \theta)$

Multiclass

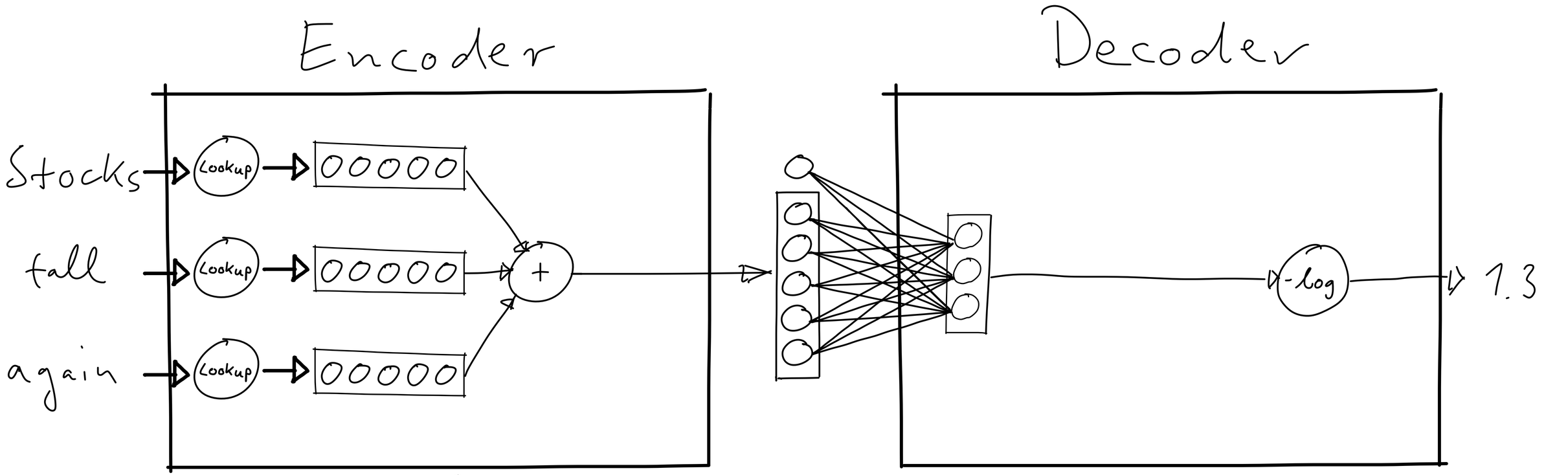

Encoder
Decoder
Stocks
fall
again
Lookup
Lookup
Lookup
+
$s_j = \sum_i \theta_{ji} f_i(x)$
$\mathbf{s} = \boldsymbol{\theta} \mathbf{f}(x)$
log
1.3
$\boldsymbol{\theta} \in \mathbb{R}^{3 \times m}$
$\mathbf{a} = \mathbf{g}(\mathbf{s}) \in \mathbb{R}^3$
$f(x)$
$\in \mathbb{R}^m$
How to define $\mathbf{g}$ such that $\mathbf{a}$ is a distribution?
$L(x, y=\text{Biz}; \theta)$

×

Softmax


Scores/logits
non-negative
normalized
max
S
Softmax(S)
1
0
Bird
Sports
Politics
$e^{s_1} + e^{s_2} + e^{s_3} \neq 1$
e^{s_i}
$\frac{e^{s_i}}{\sum_j^K e^{s_i}}$

Limits of Feature Engineering



engineer

"Boo! Can we just learn this?!"

$\times$

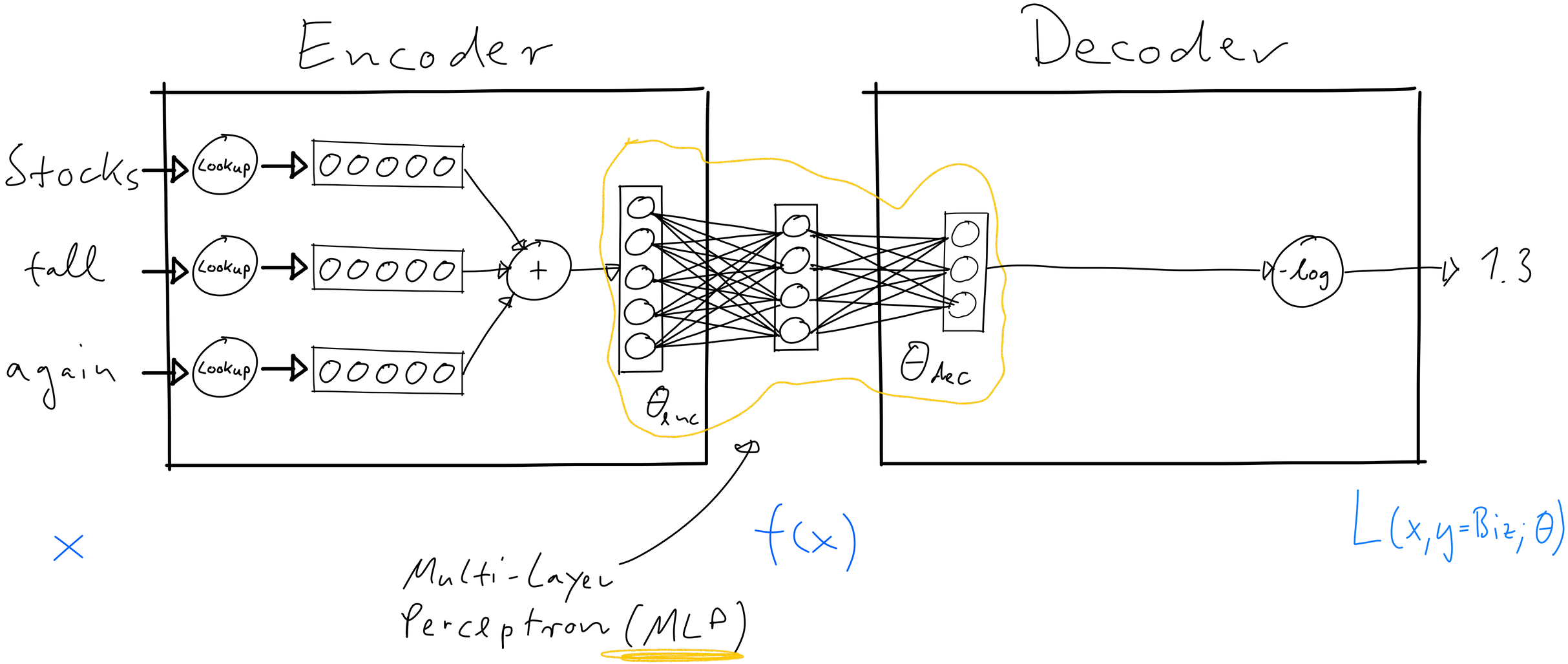
$f(x)$

① tedious

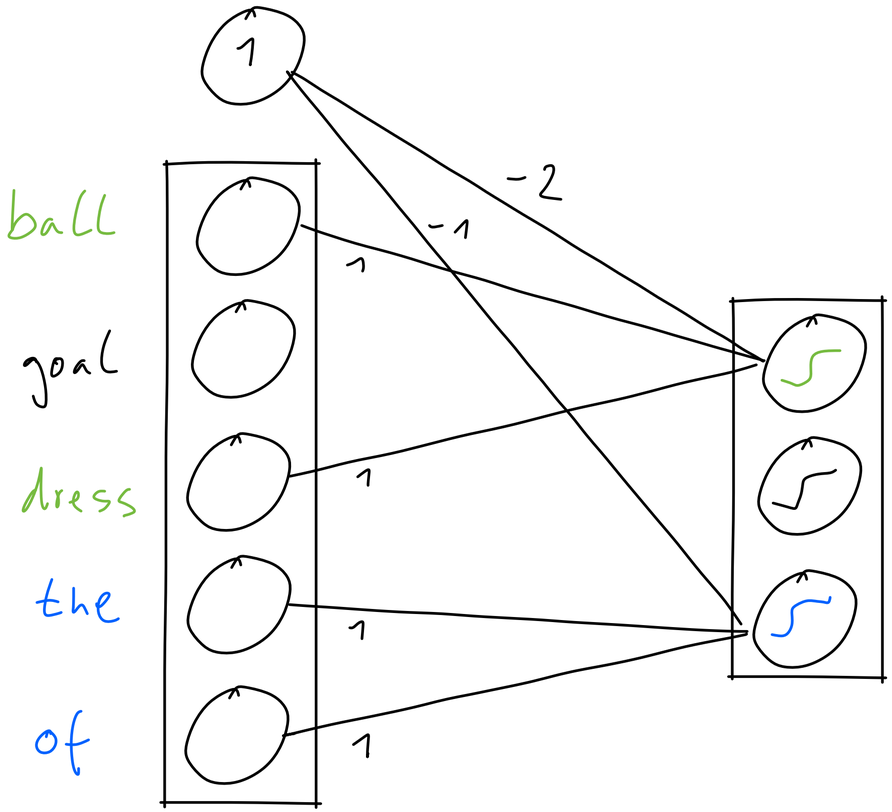
② suboptimal

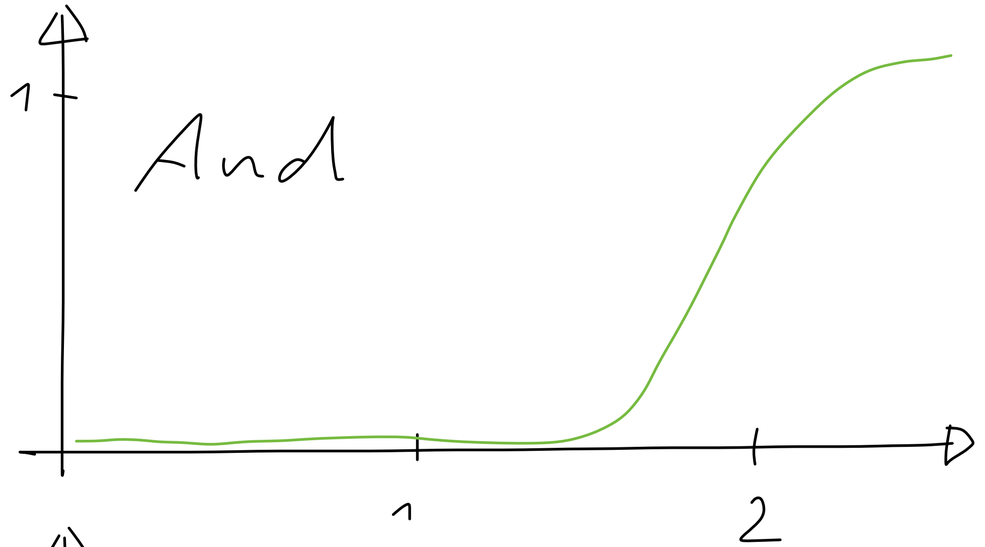
$L(x, y=\text{Biz}; \theta)$

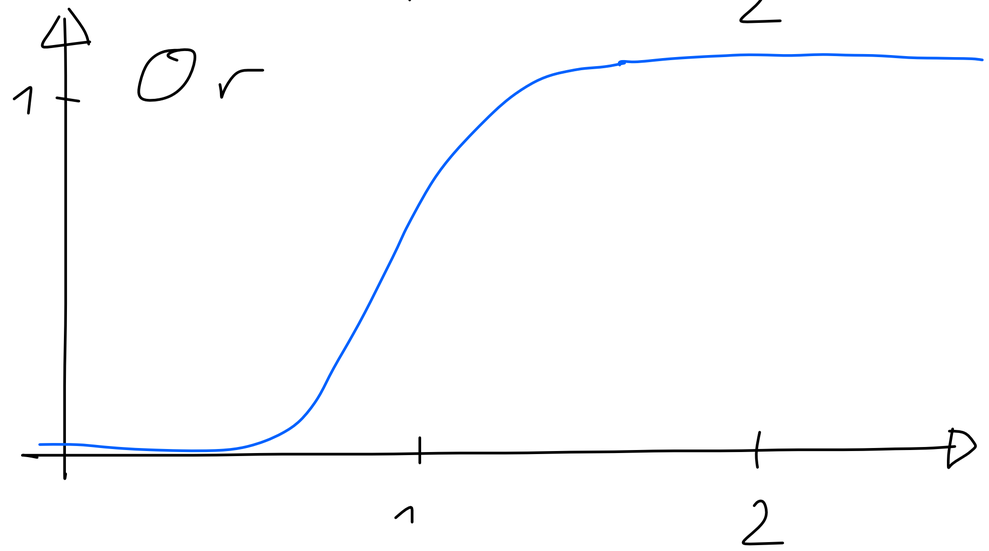
Deep Learning



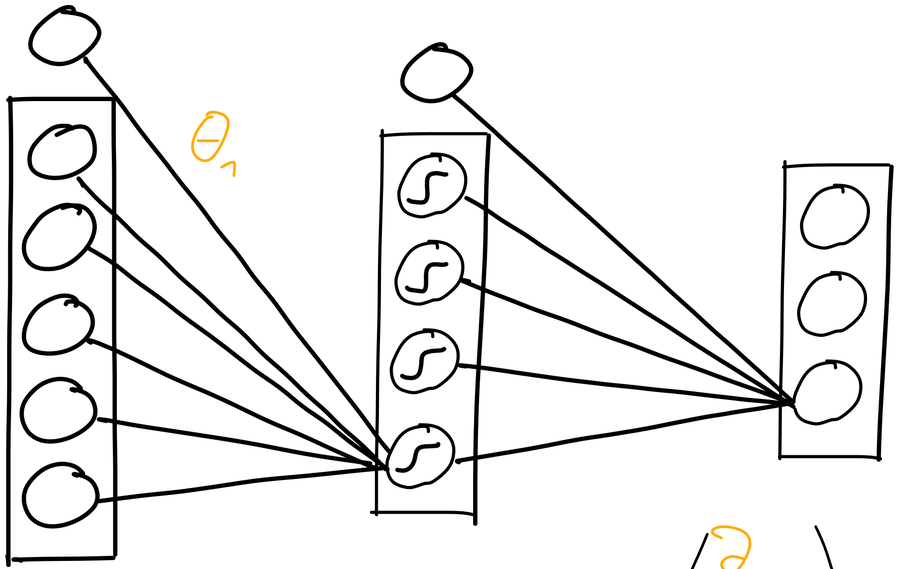
What can the layer learn?







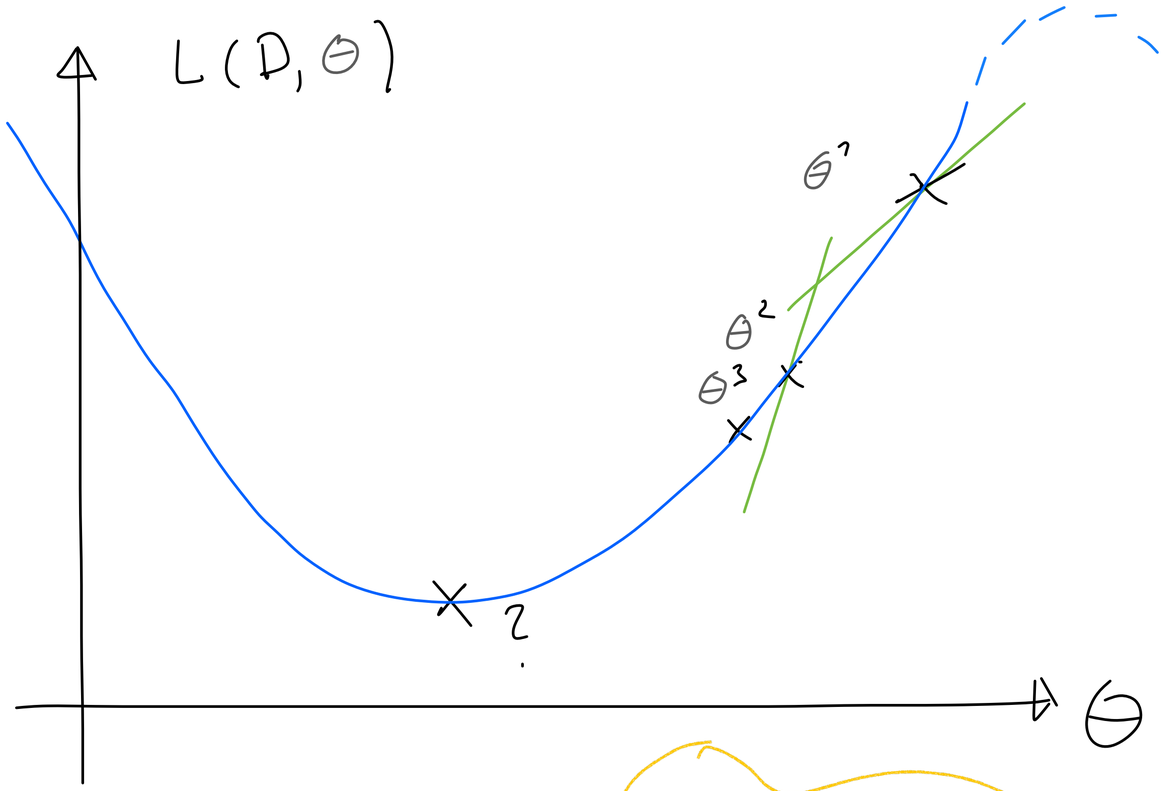
Training an MLP



$$\theta = \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_m \end{pmatrix}$$

$$\nabla_\theta = \begin{pmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_m} \end{pmatrix}$$

can be calculated automatically



$$\theta^{i+1} = \theta^i - \alpha \nabla_\theta \ell(x_j, y_j, \theta^i)$$

need this!

Warning: It will get a little technical

But: This is (almost) all there is to Deep Learning

Leibniz Notation

$x \in \mathbb{R}$

$y = f(x) \in \mathbb{R}$

derivative of y with respect to x

$\frac{\partial y}{\partial x}(x) = f'(x)$

often omitted

if I wiggle x a tiny tiny bit at x, how does y change?

$\in \mathbb{R}$

$\mathbf{x} \in \mathbb{R}^n$

$y = f(\mathbf{x}) \in \mathbb{R}$

$$\frac{\partial y}{\partial \mathbf{x}}(\mathbf{x}) = \begin{pmatrix} \frac{\partial y}{\partial x_1} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{pmatrix}$$

$\| $

$\nabla_{\mathbf{x}} f(\mathbf{x})$

$\in \mathbb{R}^n$

$\mathbf{x} \in \mathbb{R}^n$

$\mathbf{y} = \mathbf{f}(\mathbf{x}) \in \mathbb{R}^m$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}(\mathbf{x}) = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \vdots & & \vdots \\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{pmatrix}$$

Jacobian

$\in \mathbb{R}^{n \times m}$

Chain Rule

$$a = f(x) \qquad wx$$

$$b = g(a) \qquad \log(a)$$

$$\frac{\partial b}{\partial x}(x) = \boxed{\frac{\partial a}{\partial x}(x)}^{\text{Local}} \cdot \boxed{\frac{\partial b}{\partial a}(a)}^{\text{upstream}}$$

$$\frac{\partial \log(wx)}{\partial x} \stackrel{?}{=}$$

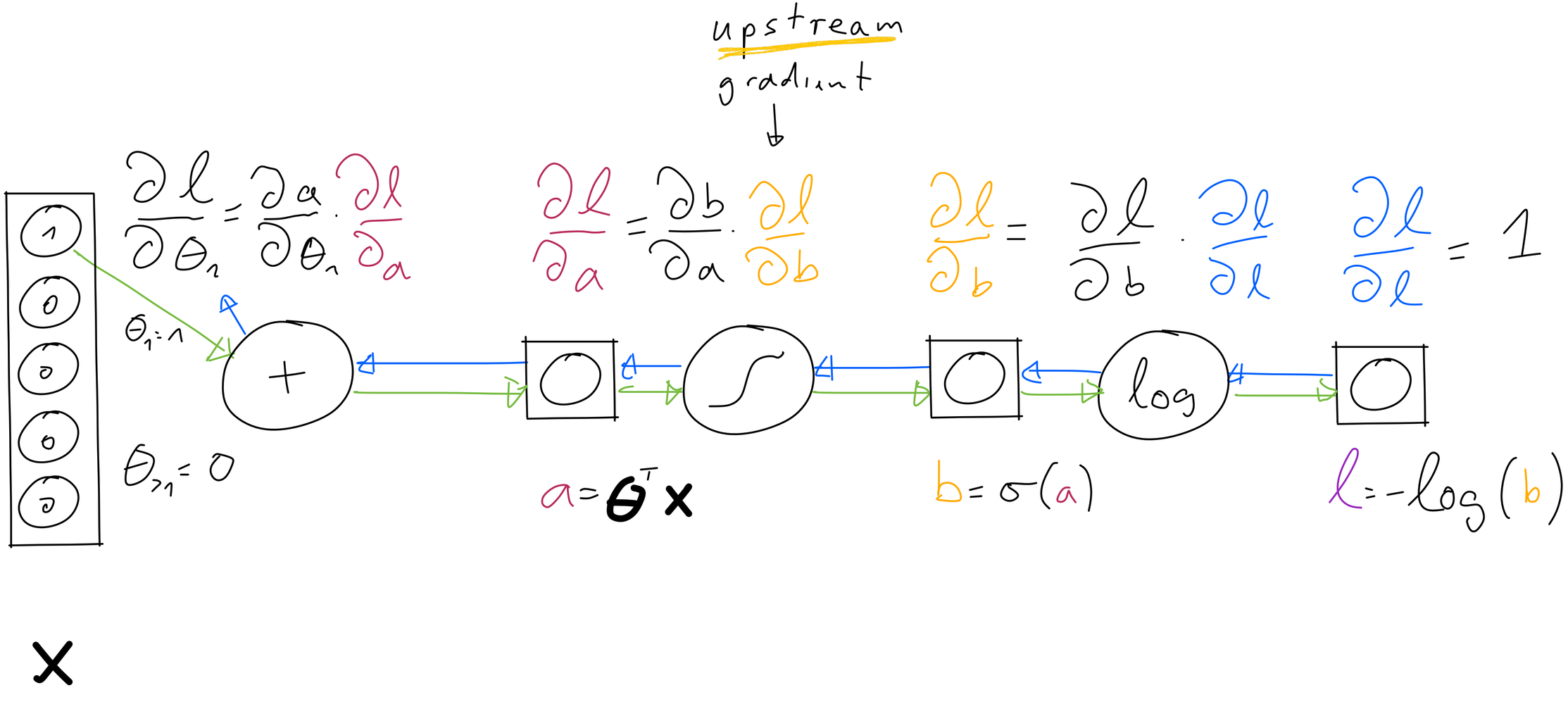
Multidimensional Chain Rule

$$x \in \mathbb{R}^{\ell} \qquad a = f(x) \in \mathbb{R}^{n} \qquad b = g(a) \in \mathbb{R}^{m}$$

$$\frac{\partial b}{\partial x}(x) = \underbrace{\frac{\partial a}{\partial x}(x)}_{\text{Local}} \cdot \underbrace{\frac{\partial b}{\partial a}(a)}_{\text{upstream}}$$

$$\mathbb{R}^{\ell \times m} \qquad \mathbb{R}^{\ell \times n} \qquad \mathbb{R}^{n \times m}$$

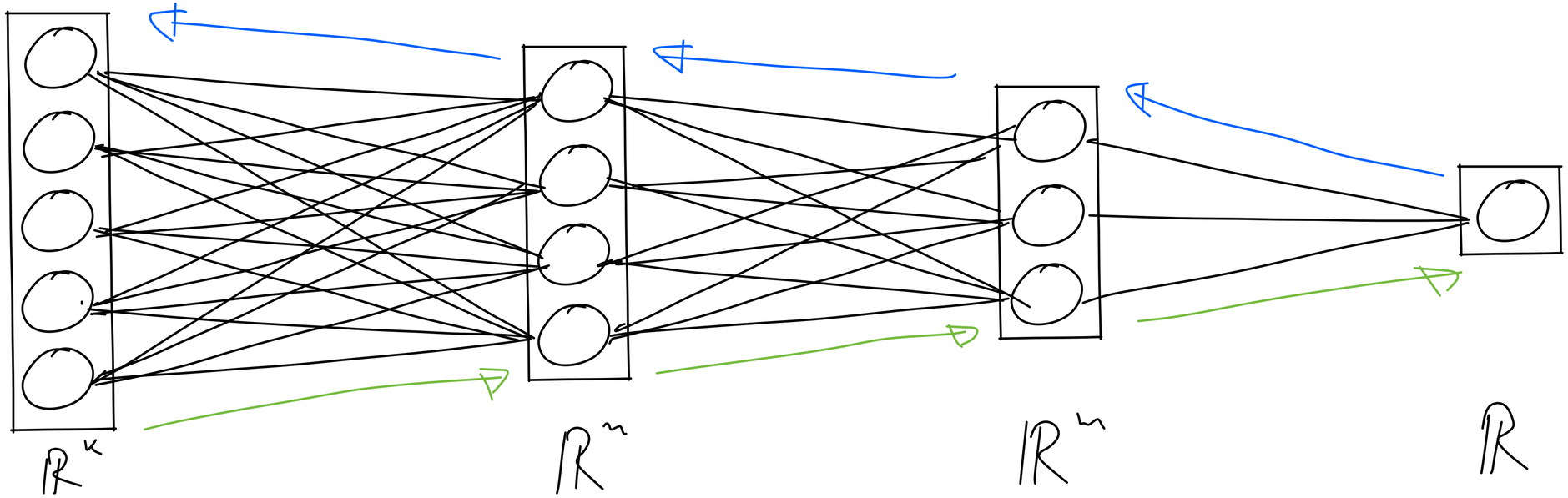
Forward and Backward Pass



Multi-dimensional

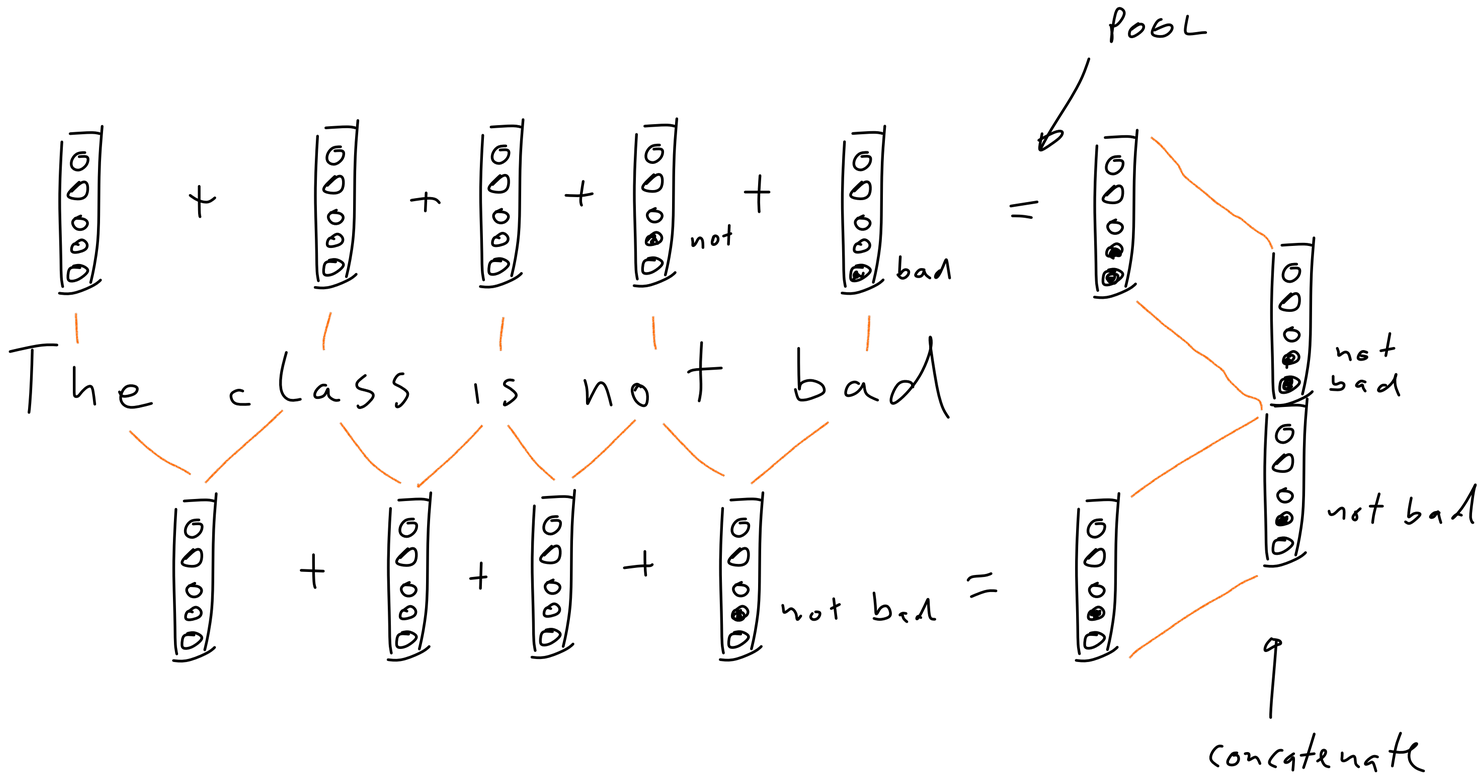
$$\frac{\partial L}{\partial x} = \frac{\partial a}{\partial x} \cdot \frac{\partial c}{\partial a} \qquad \frac{\partial L}{\partial a} = \frac{\partial b}{\partial a} \cdot \frac{\partial c}{\partial b} \qquad \frac{\partial L}{\partial b}$$

1



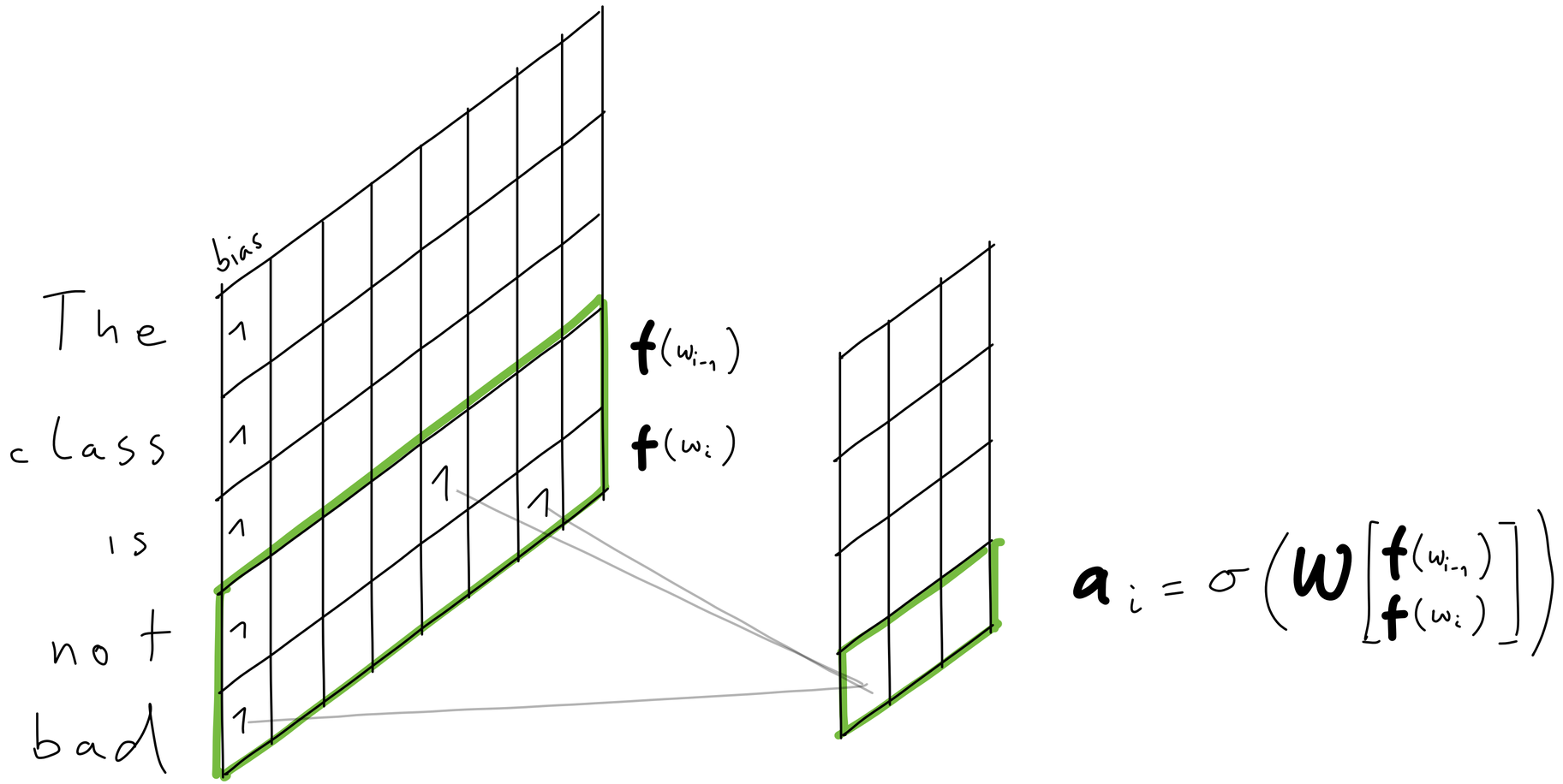
X $\quad a = \sigma(Wx)$ $\quad b = \sigma(Va)$ $\quad c = \sigma(\theta_b^T)$

Remember Bigrams?

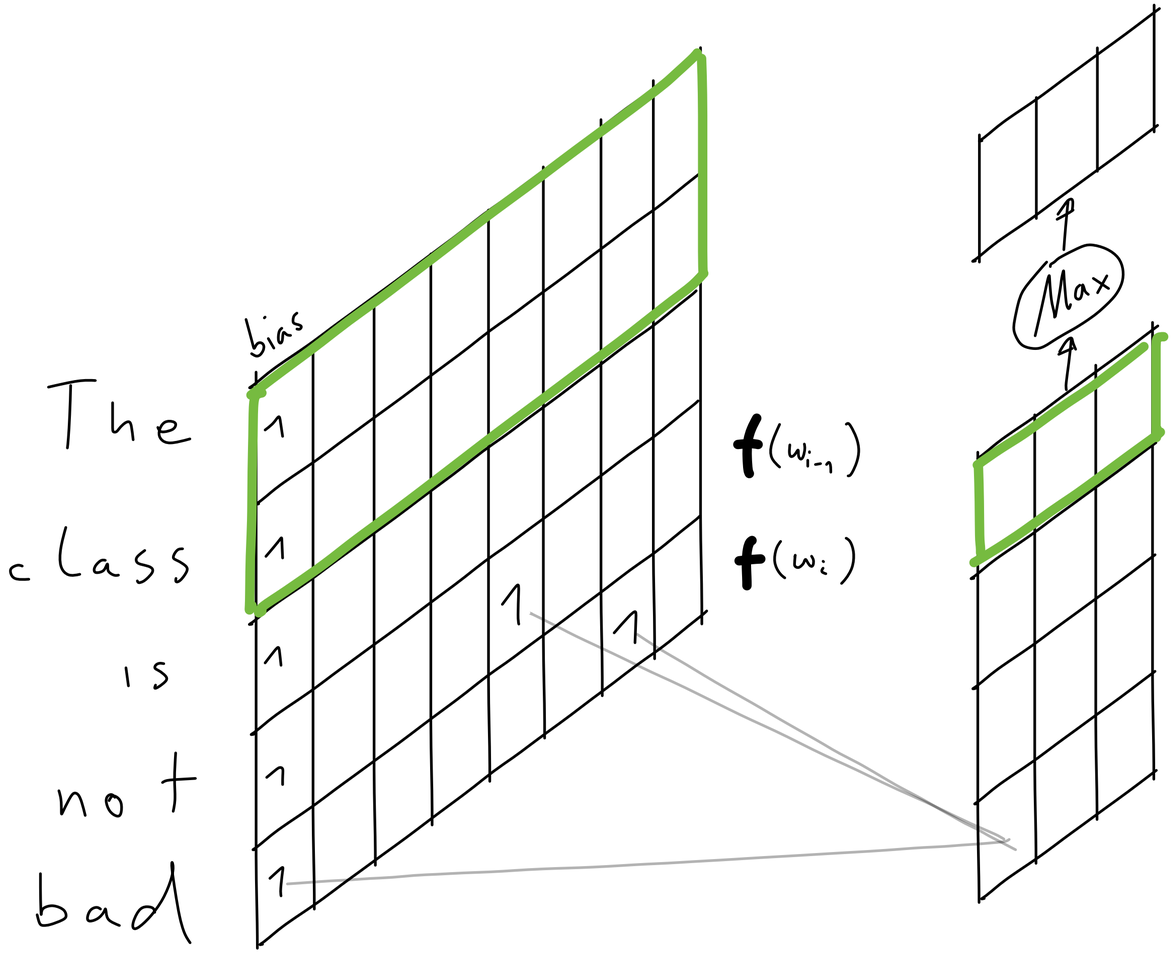


Can the MLP learn this ?

Convolutional Neural Networks



Convolutional Neural Networks

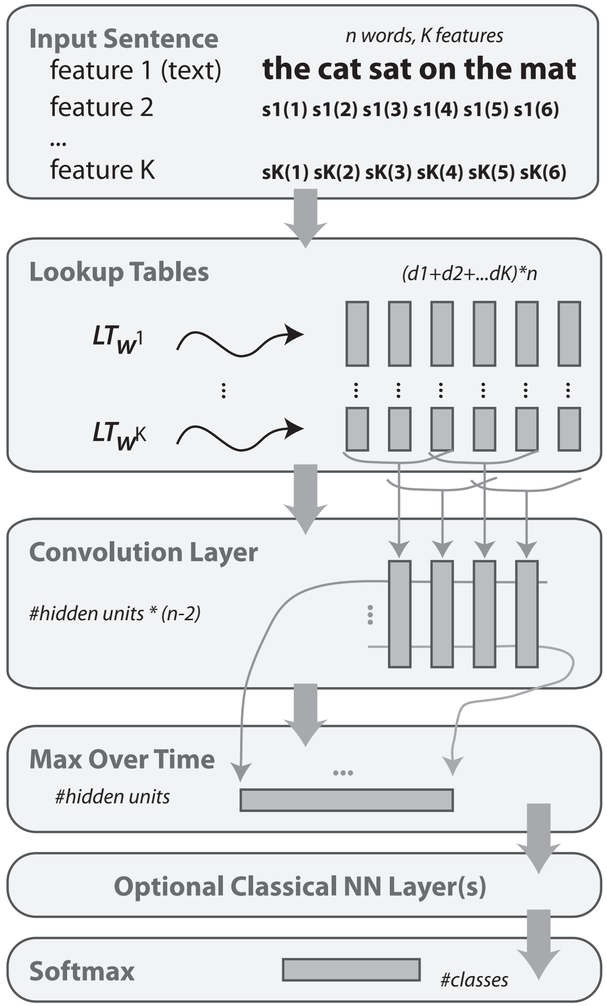


$$a = \max_i a_i$$

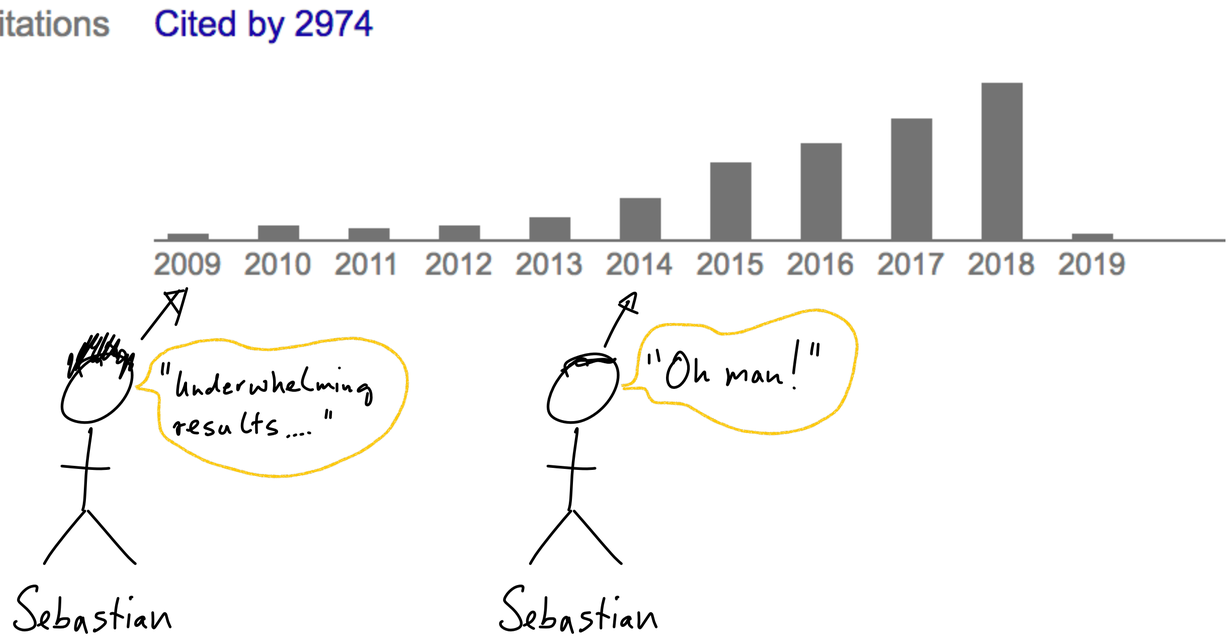
$$a_i = \sigma\left(W\begin{bmatrix} f(w_{i-1}) \\ f(w_i) \end{bmatrix}\right)$$

A Unified Architecture for Natural Language Processing

Collobert & Weston 2008



Encoder / Decoder ?



Final Words

On backpropagation:
"My view is throw it all away and start again."

"The future depends on some graduate student who is deeply suspicious of everything I have said."

Geoffrey Hinton
"Godfather of Deep Learning"

References

- A Unified Architecture for Natural Language Processing, Collobert & Weston, ICML 2008
- Natural Language Processing (Almost) from Scratch, Collobert et al., 2011, JMLR
- Goldberg, Chapter 4 & 6