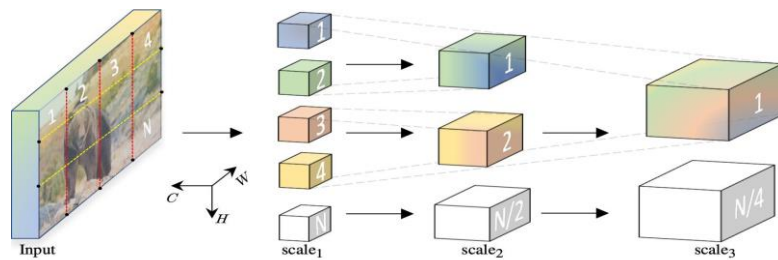


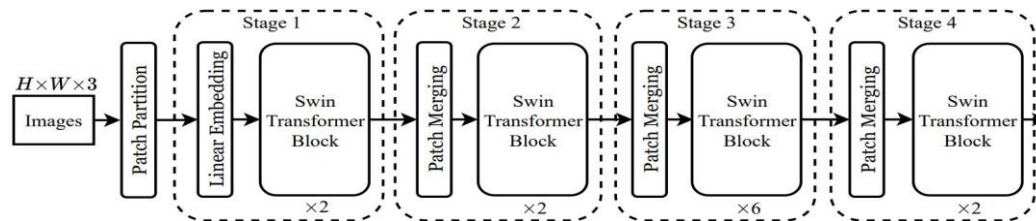
Ralf Möller, Sylvia Melzer

Backpropagation

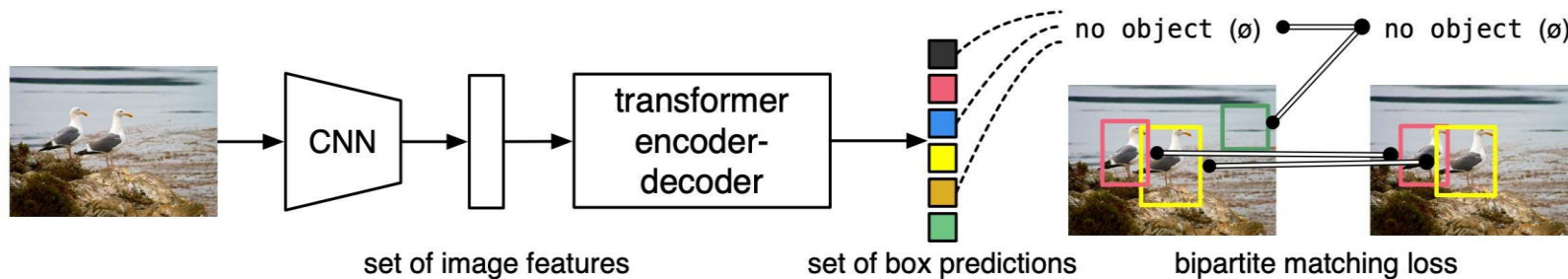
Vision Transformers: How to Train?



Fan et al, “Multiscale Vision Transformers”, ICCV 2021



Liu et al, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, CVPR 2021

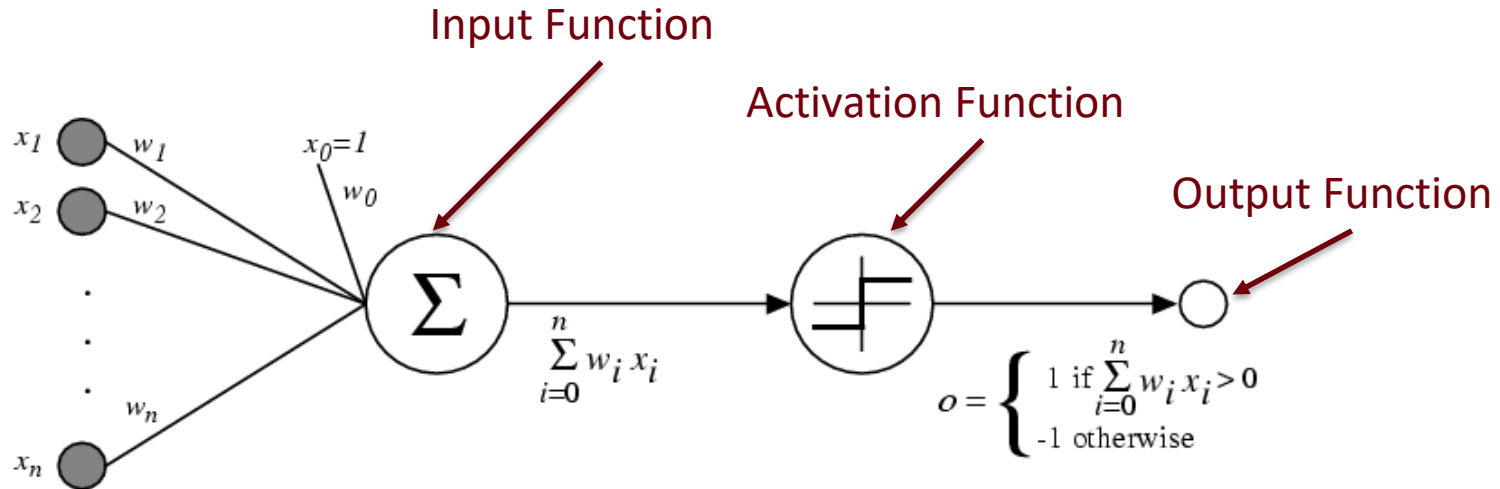


Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

<http://people.cs.umass.edu/~miyyer/cs685/>

Perceptron

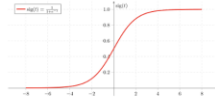
Frank Rosenblatt, The Perceptron--a perceiving and recognizing automaton. Report 85-460-1, Cornell Aeronautical Laboratory, 1957



Examples for Activation Functions

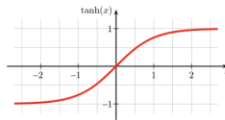
- Linear activation: $a_k = f_{act}(net_k) = c_k \cdot net_k$
- Threshold activation: $a_k = f_{act}(net_k) = \begin{cases} 1, & \text{falls } net_k \geq \theta_k, \\ 0, & \text{sonst.} \end{cases}$
- Logistic activation: $a_k = f_{act}(net_k) = \frac{1}{1 + e^{-\frac{net_k}{T}}}$

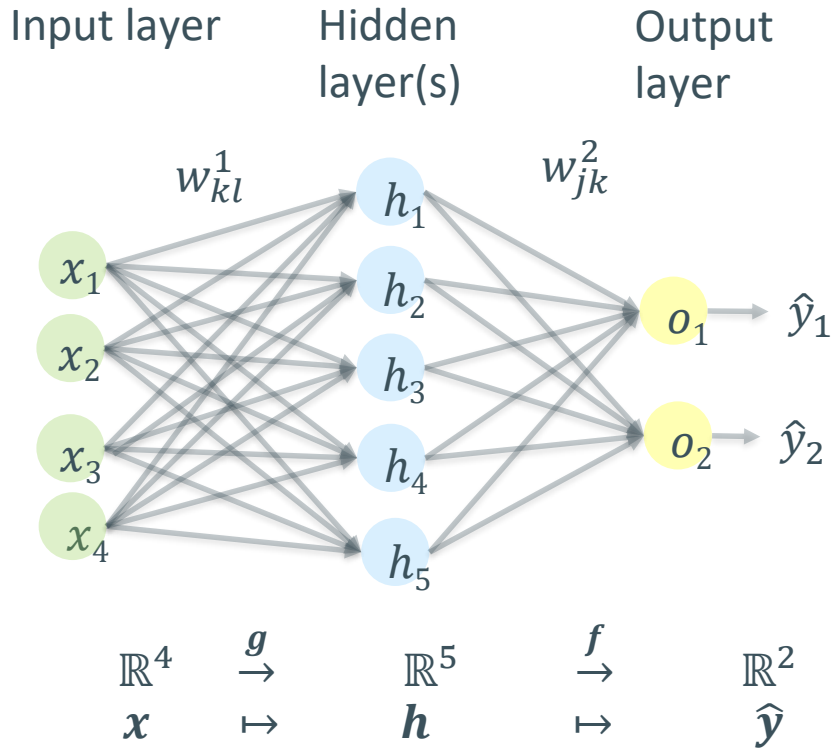
Schwellenwert,
häufig 0



Special case Sigmoid: $T=1$

- Hyperbolic tangent: $a_k = f_{act}(net_k) = \frac{e^{net_k} - e^{-net_k}}{e^{net_k} + e^{-net_k}} = \frac{1 + \tanh(net_k)}{2}$





i : Layer i

\mathbf{b}^i : Bias for i

\mathbf{W}^i : Weight matrix for i

σ^i : Activation function for i

\mathbf{out}^i : $\sigma^i(\mathbf{W}^i \mathbf{out}^{i-1} + \mathbf{b}^i)$
Activation in i

\mathbf{net}^i : $\mathbf{W}^i \mathbf{out}^{i-1} + \mathbf{b}^i$
Linear output in layer 1

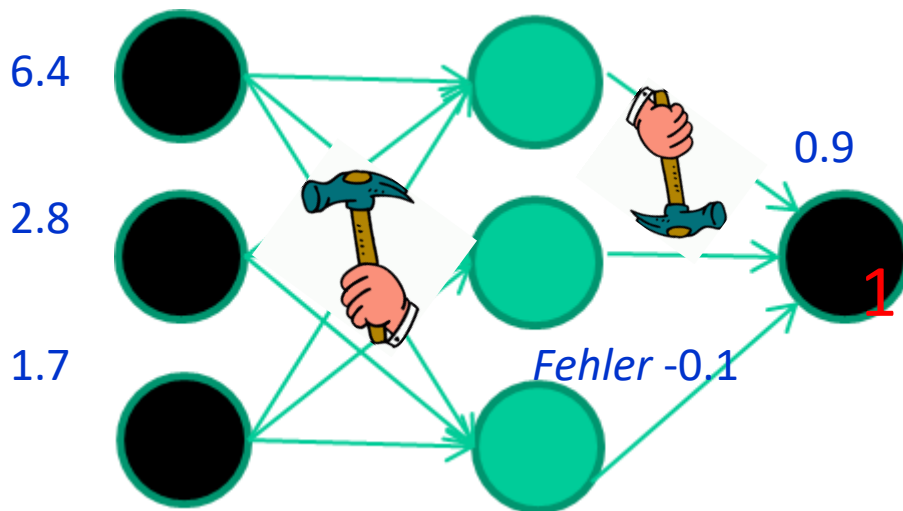
$$\hat{\mathbf{y}} = \mathbf{f}(\mathbf{g}(\mathbf{x}; \mathbf{W}^1, \mathbf{b}^1); \mathbf{W}^2, \mathbf{b}^2)$$

$$= \sigma^2(\mathbf{W}^2 \sigma^1(\mathbf{W}^1 \mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2)$$

Adaptation of mapping parameters (“weights”)

<i>Fields</i>	<i>class</i>
1.4 2.7 1.9	0
3.8 3.4 3.2	0
6.4 2.8 1.7	1
4.1 0.1 0.2	0
etc ...	

Simple case $o = Wx$



Delta Learning Rule

D. Hebb: The organization of behavior. A neuropsychological theory.
Erlbaum Books, Mahwah, N.J., 1949

$$w_i \leftarrow w_i + \Delta w_i$$

where

$$\Delta w_i \leftarrow \eta(t - o)x_i$$

and

- $t = c(\vec{x})$ is the goal value
- o is the perceptron output
- η is a small constant

Adaptation with averaging over whole dataset D

Justification of Delta Rule

- Idea: minimize quadratic error
 - D training datasets
 - t_d value for $d \in D$
 - o_d output for d

$$E[\vec{w}] \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

- Determine minimum with 1st derivative

$$E[\vec{w}] = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

Gradient Descent

- Gradient

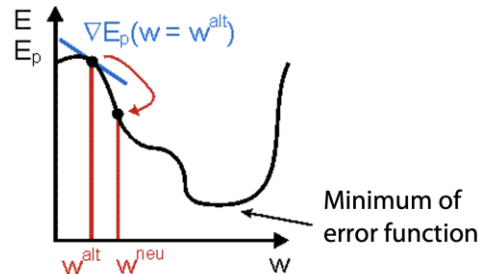
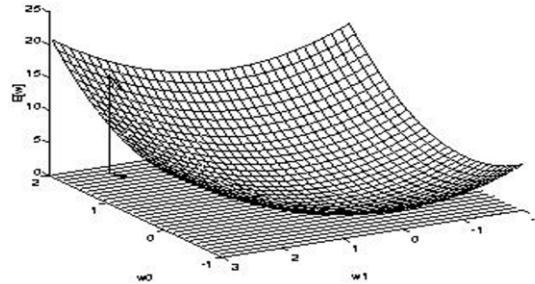
$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$$

- Learning rule

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

- i.e.

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$



$$E[\vec{w}] = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

Gradient

$$\begin{aligned} \frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\ &= \frac{1}{2} \sum_{d \in D} \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\ &= \frac{1}{2} \sum_{d \in D} 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\ &= \sum_{d \in D} (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d) \\ &= \sum_{d \in D} (t_d - o_d) (-x_{i,d}) = - \sum_{d \in D} (t_d - o_d) x_{i,d} \end{aligned}$$

Gradient Descent (cntd.)

D. Hebb: The organization of behavior. A neuropsychological theory.
Erlbaum Books, Mahwah, N.J., 1949

- Gradient

$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$$

$$\frac{\partial E}{\partial w_i} = - \sum_{d \in D} (t_d - o_d) x_{i,d}$$

- Learning rule

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

- i.e.

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

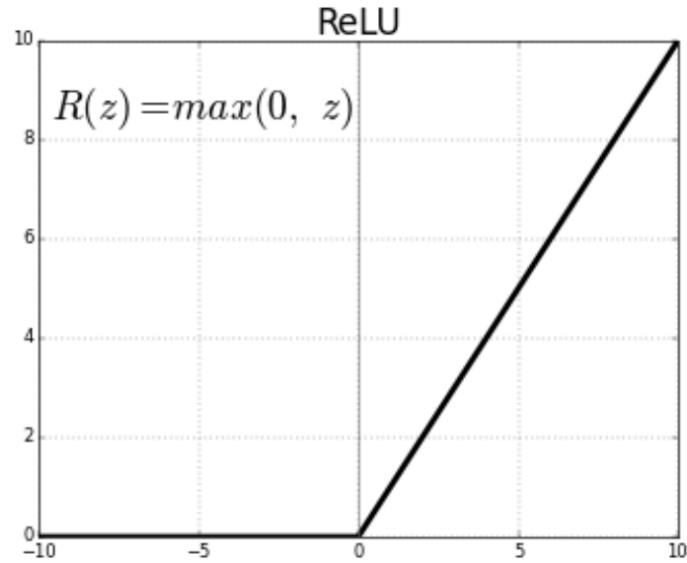
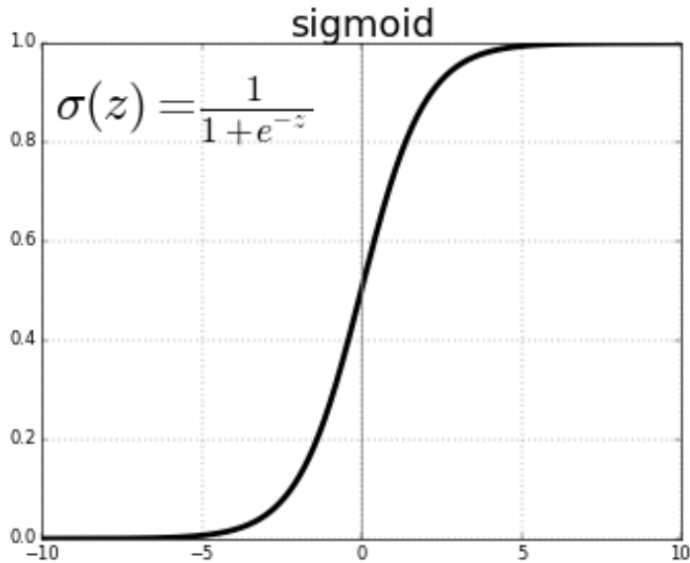
Iterate over data point d

$$\Delta w_i = \eta (t - o) x_i$$

$$\Delta w_i = \eta \sum_{d \in D} (t_d - o_d) x_{i,d}$$

What if the
gradients
become very
small?

ReLU: Rectified Linear Unit

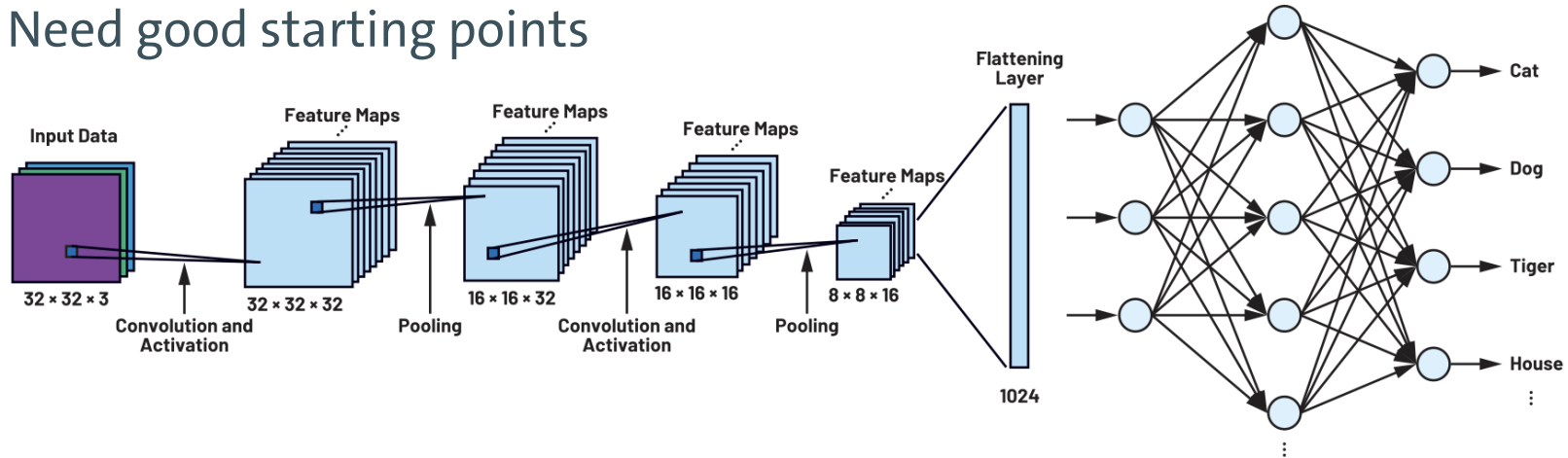


Learning Parameters (“Hyper parameters”)

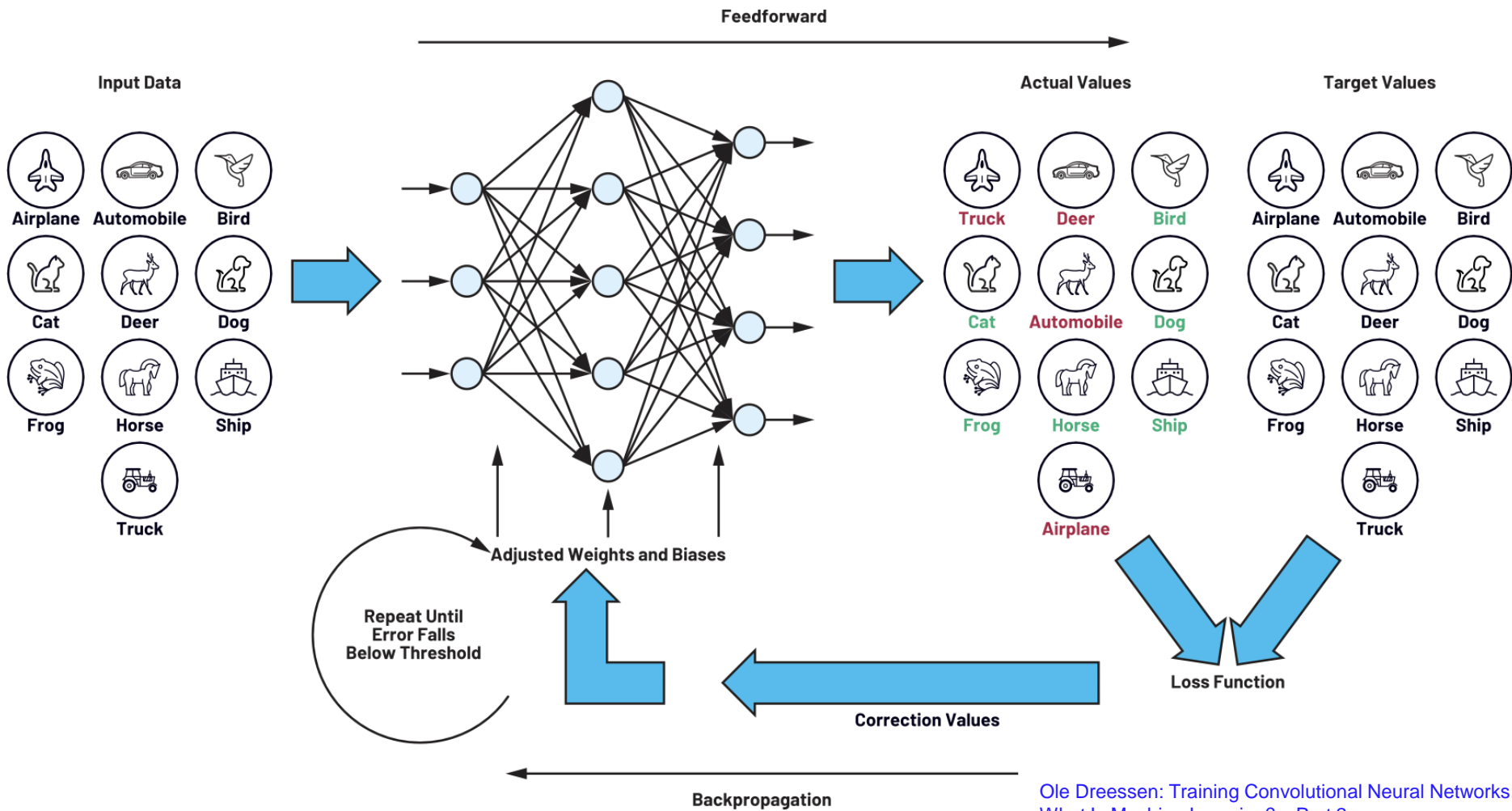
- Batch size
 - Number of samples processed before model is updated
- Epochs
 - Number of complete passes through the training dataset

Autoencoder

- Training of convolutional networks
- Need good starting points



Ole Dreessen: Training Convolutional Neural Networks:
What Is Machine Learning?—Part 2



ViLBERT (Vision and Language BERT)

ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 2, 13–23. 2019

Presented by - **Sidharth Singla**, 20774908



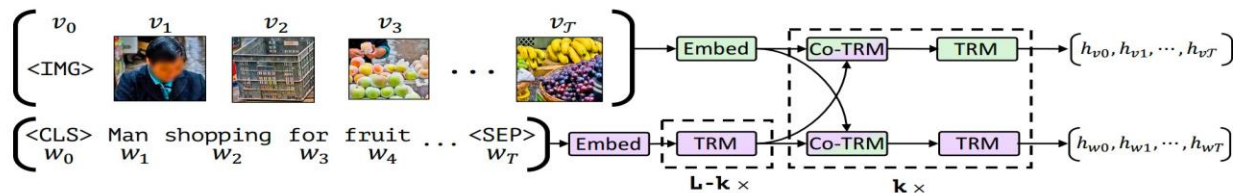
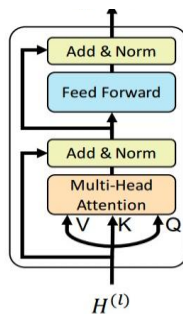
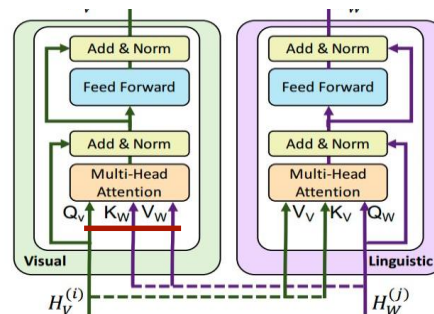


Figure 1: Our ViLBERT model consists of two parallel streams for visual (green) and linguistic (purple) processing that interact through novel co-attentional transformer layers. This structure allows for variable depths for each modality and enables sparse interaction through co-attention. Dashed boxes with multiplier subscripts denote repeated blocks of layers.



(a) Standard encoder transformer block



(b) Our co-attention transformer layer

Figure 2: We introduce a novel co-attention mechanism based on the transformer architecture. By exchanging key-value pairs in multi-headed attention, this structure enables vision-attended language features to be incorporated into visual representations (and vice versa).

Method: Contrastive Pre-training

