



Objektdetektion

I. Grundlagen

2. Single-Stage-Detection

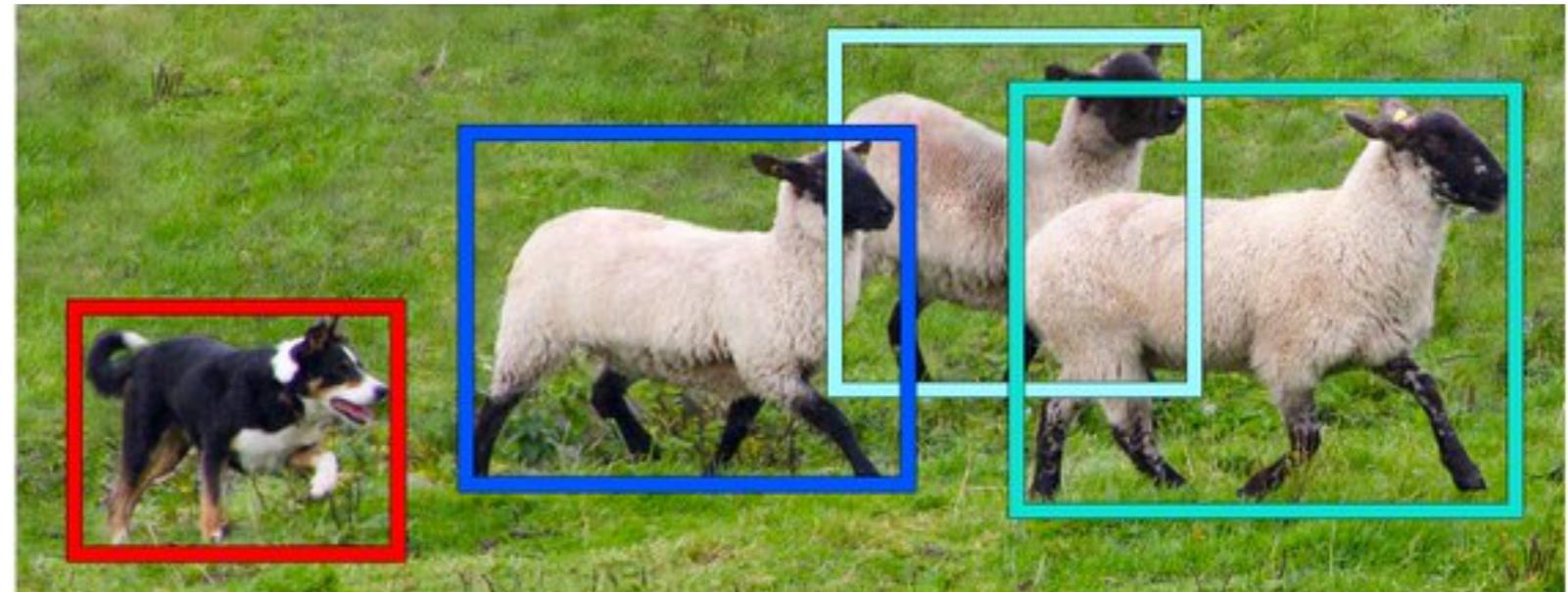
Objektdetektion

Input

Bild

Output

Menge von Objekten



Objekt

man braucht nur 4 Werte

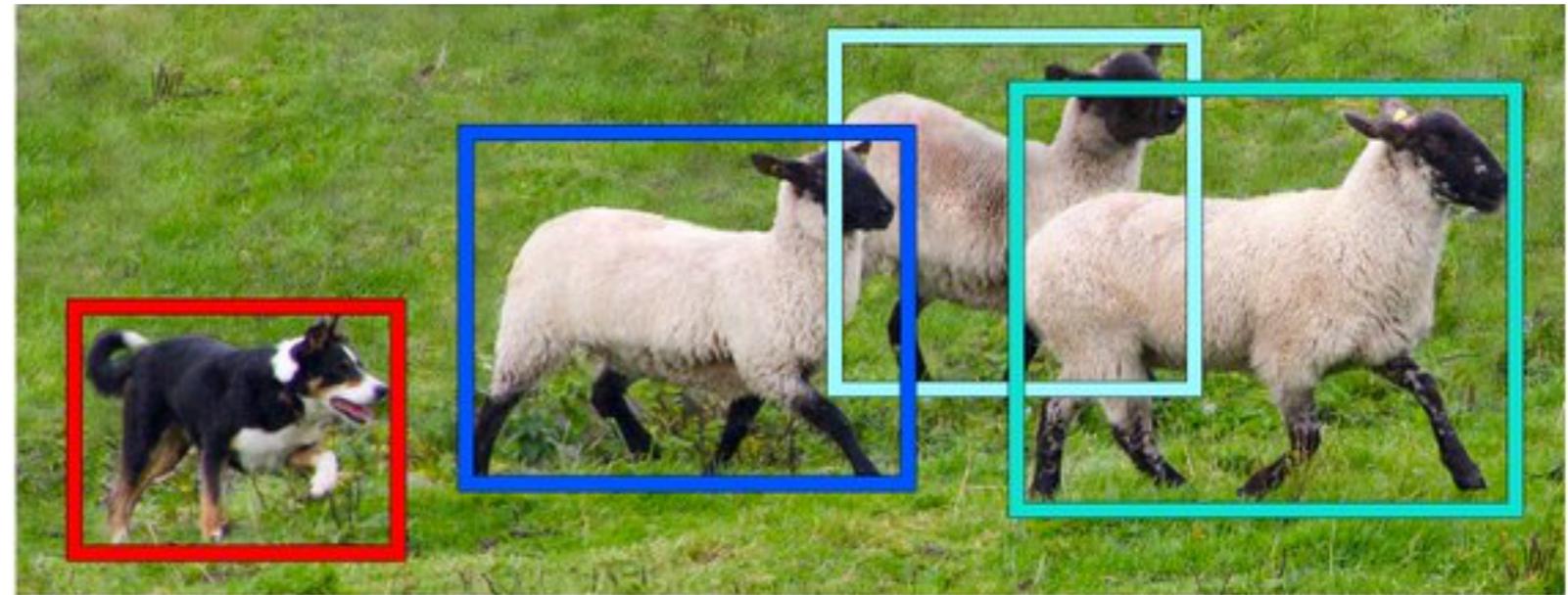
↳ Bounding Box: Höhe, Breite, Mittelpunkt (x,y)

↳ Klassen Label

Objektdetektion

Herausforderungen

Menge von Objekten



© ai-pool.com

Output ist Mischung aus verschiedenen Informationen

Classification

Klasse: Was?

Regression

Bb: Wo?

X₀

Y₀

H₀

... ,

w₀

→ Mischung aus Regression und Klassifikation

Objektdetektion

Annahme

genau ein Objekt erkennen



© C. Hiller, ReMIC, OTH Regensburg

Faltungshetz
mit Output knoten



H Klassen + 4



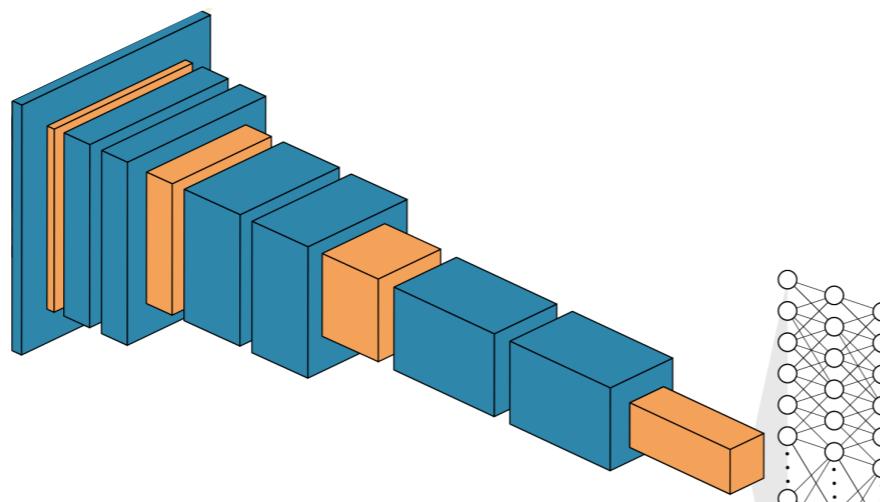
Objektdetektion

Annahme

genau ein Objekt erkennen



© C. Hiller, ReMIC, OTH Regensburg



Label	
Katze	
Hund	
Kaninchen	
...	

0 } Class, filtering: Cross Entropy
0 }
0 }
0 }
0 }
0 }

0 } Regression,
0 }
0 } Mean Squared Error Loss

=> generic loss functions

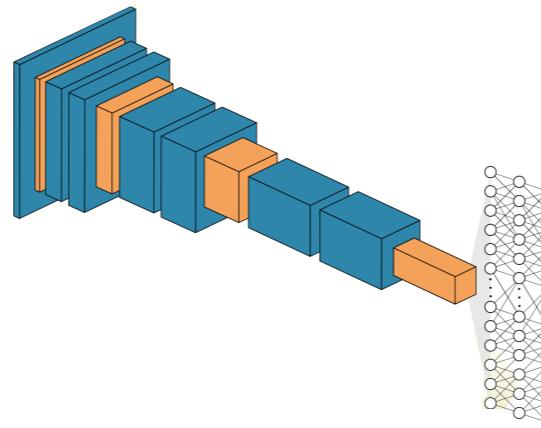
e. B. additive

Objektdetektion

mehrere Objekte erkennen



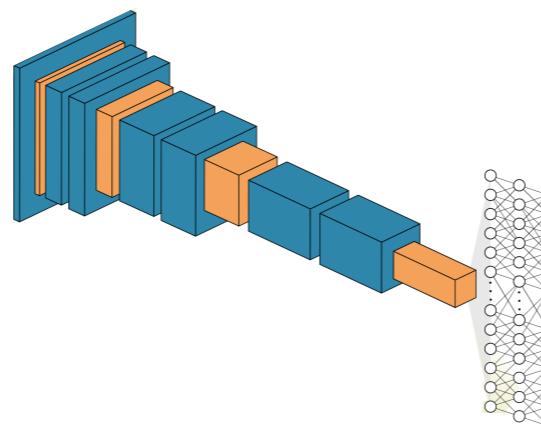
© C. Hiller, ReMIC, OTH Regensburg



4 Werte + Label
Katze (x, y, w, h)



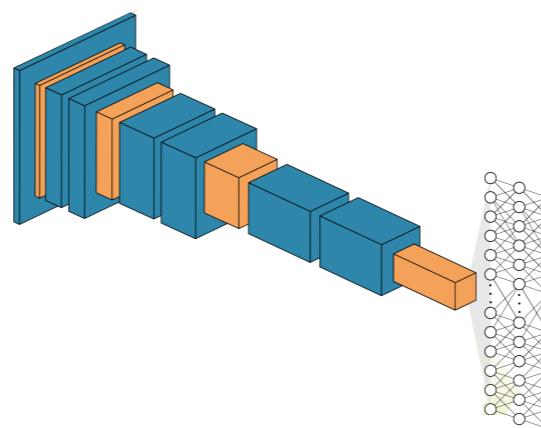
© Eric Isselee / Shutterstock



Katze (x, y, w, h)
Kätzchen (x, y, w, h)
Hund (x, y, w, h)
Hase (x, y, w, h)



© Fritz Hertel, UBA



Pinguin (x, y, w, h)
nicht praktikabel, zu viele Werte

Objektdetektion

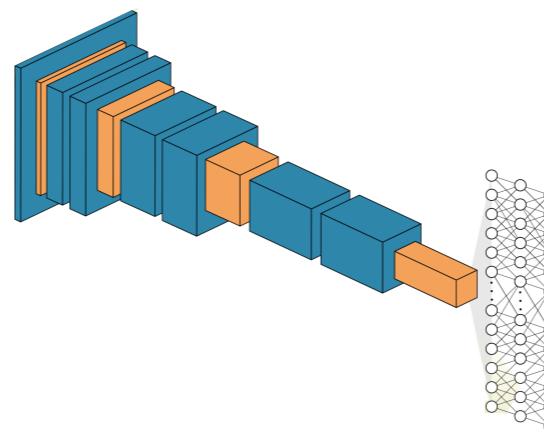
mehrere Objekte erkennen

Lösung 1

Sliding Window



© Eric Isselee / Shutterstock



Label	
Katze	
Hund	
Kaninchen	
Hintergrund	

Betrachtung von einem Fenster für Bild der Größe w, h

$$(w-w+1)(h-h+1)$$

Variation der Fenstergroße: $\sum_{h=1}^H \sum_{w=1}^W (w-w+1)(h-h+1)$

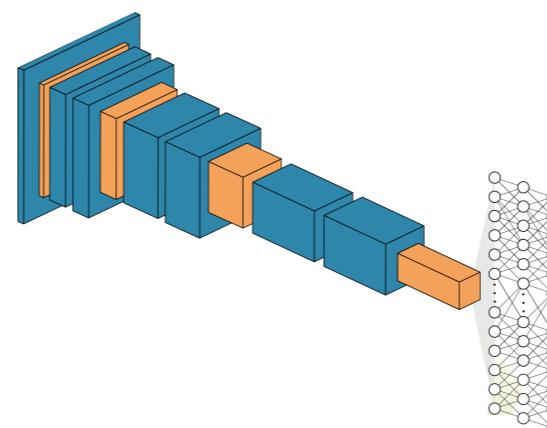
Objektdetektion

mehrere Objekte erkennen

Lösung 1



© Eric Isselee / Shutterstock



Label	
Katze	
Hund	
Kaninchen	
Hintergrund	

$$\sum_{h=1}^H \sum_{w=1}^W (V - v_{h,w}) (H - h + 1) = \frac{W(W+1)}{2} \cdot \frac{H(H+1)}{2}$$

Ans: 800x600 Pixel

→ 58 Billionen mögliche Fehler

Objektdetektion

mehrere Objekte erkennen

Lösung 1

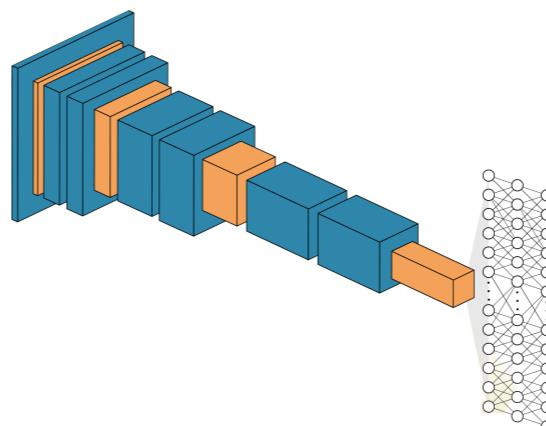


© Eric Isselee / Shutterstock

Probleme:

- ↳ nur Teile in Fenster
- ↳ überschappende Fenster mit gleichen Objekt

→ Vorverarbeitung!



Label	
Katze	
Hund	
Kaninchen	
Hintergrund	

Objektdetektion

mehrere Objekte erkennen

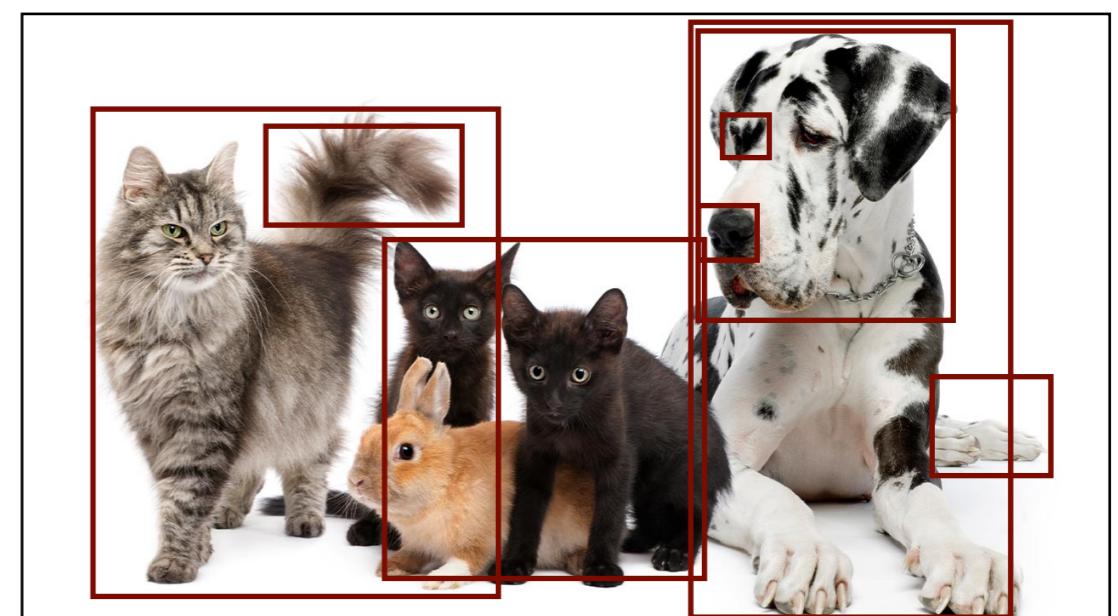
Lösung 2

finde Merkmale, die Aussagen über Objekt/nicht-Objekt zulassen

finde eine kleine Zahl von Bounding Boxes, die wahrscheinlich Objekte enthalten



© Eric Isselee / Shutterstock



↳ Objectness messen

↳ Merkmale: z.B. Helligkeit, Kanten

Objektdetektion

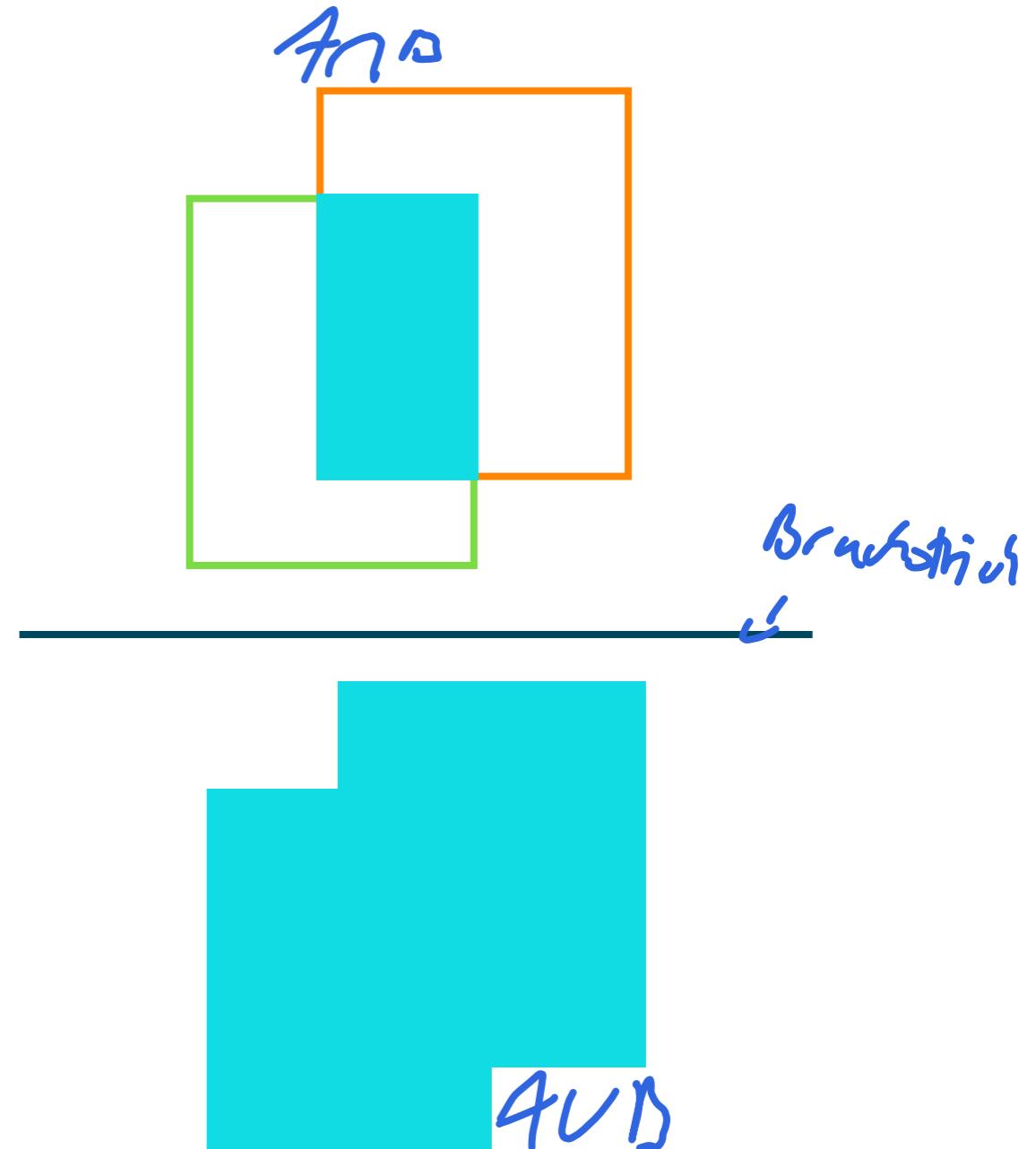
Intersection over Union (IoU)

↳ Schnittmenge getilt durch Vereinigung



© pixabay.com

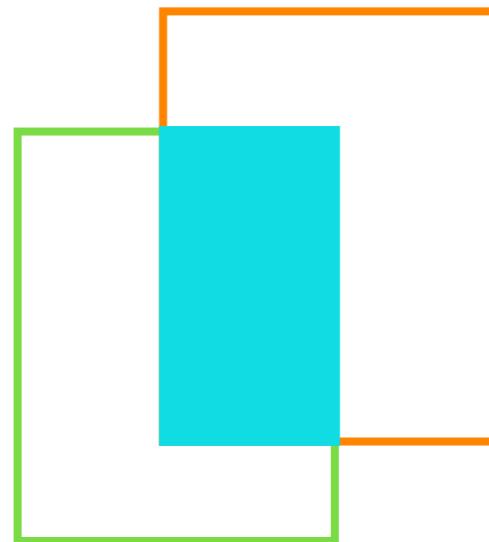
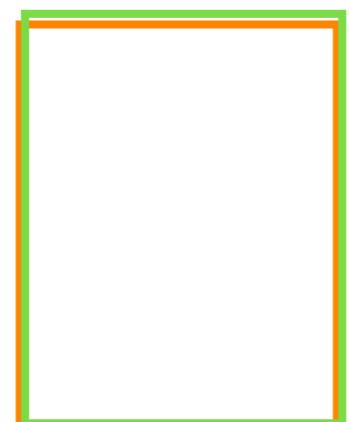
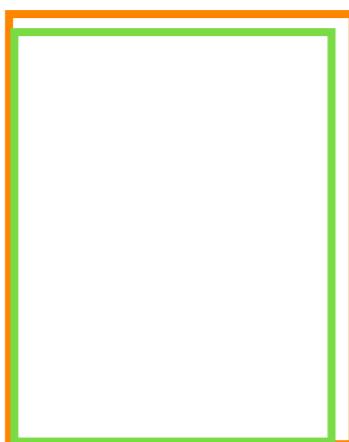
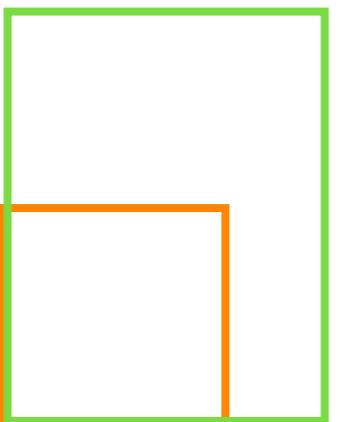
ground truth (GT)
Dargest. Voraussage



Objektdetektion

Intersection over Union (IoU)

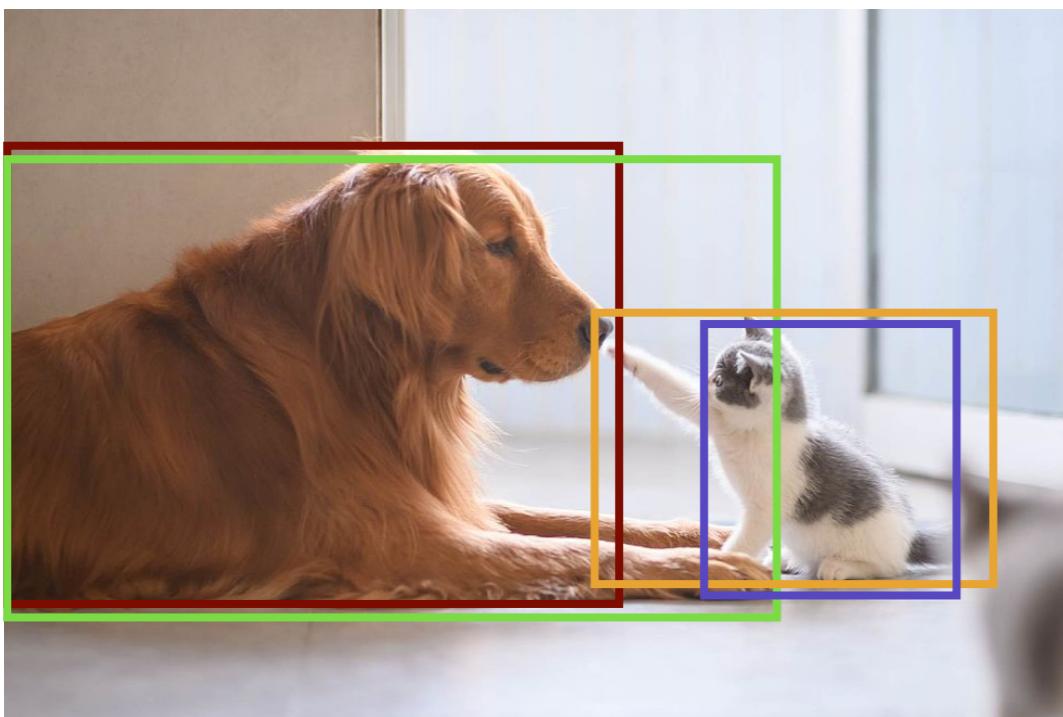
IoU erfasst Position und Größe der Detektion zu betrachten



Objektdetektion

Non-Maximum-Suppression

- finde BB mit höchstem Score
- Berechne Intersection of Union (IoU) zu allen anderen BBs des gleichen Labels
- Lösche alle BBs mit $\text{IoU} < \text{Schwellenwert}$ *0.5 verstreut*
- Wenn noch nicht alle Boxen betrachtet, beginne von vorne



© Rinti

rot: $P(\text{Hund}) = 0,4 \times$
grün: $P(\text{Hund}) = 0,75$ - Löschen
blau: $P(\text{Katze}) = 0,97 \times$ - *1st Grechen*
orange: $P(\text{Katze}) = 0,84$ - *Wing*
Score $\leq \text{Schw für eig Label}$

Objektdetektion

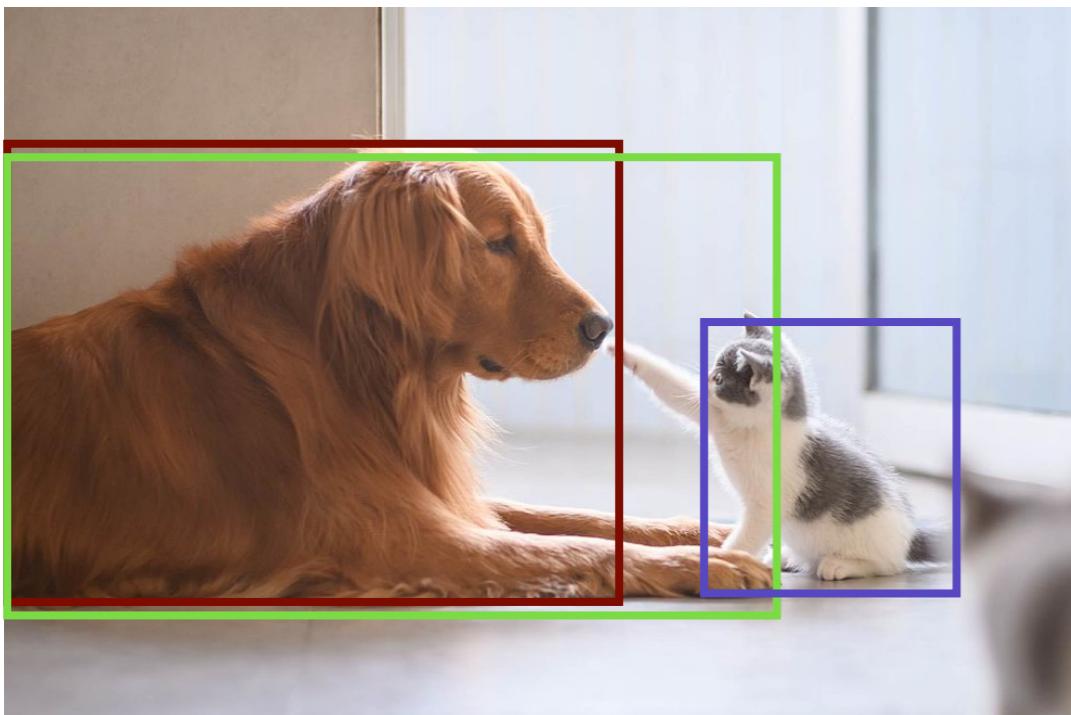
Non-Maximum-Suppression

finde BB mit höchstem Score

Berechne Intersection of Union (IoU) zu allen anderen BBs

Lösche alle BBs mit IoU > Schwellenwert

Wenn noch nicht alle Boxen betrachtet, beginne von vorne



© Rinti

Objektdetektion

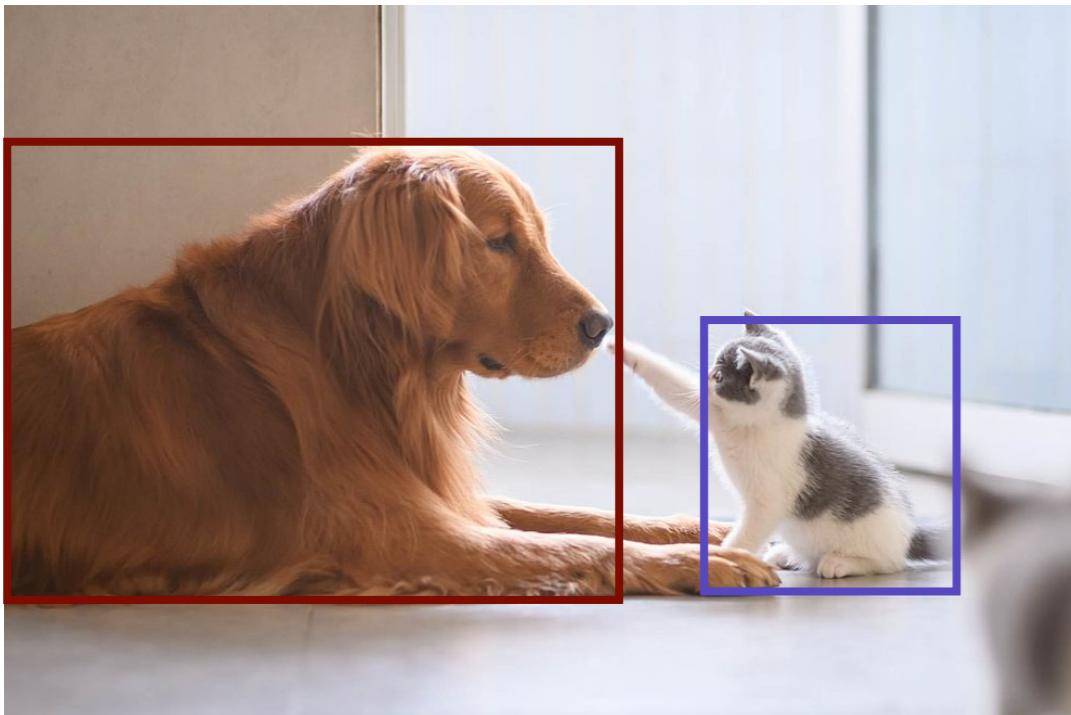
Non-Maximum-Suppression

finde BB mit höchstem Score

Berechne Intersection of Union (IoU) zu allen anderen BBs

Lösche alle BBs mit IoU > Schwellenwert

Wenn noch nicht alle Boxen betrachtet, beginne von vorne



© Rinti

→ adaptige Schwellenwert je nach Anwendung / Kabel



Objektdetektion

↳ Notiz: mAP

Fehlermaß zur Evaluation: Mean Average Precision (mAP)

für alle Testbilder: Objektdetektion inkl. Non-Maximum-Suppression

für jedes Label: berechnet die mittlere Genauigkeit (Average Precision, AP)

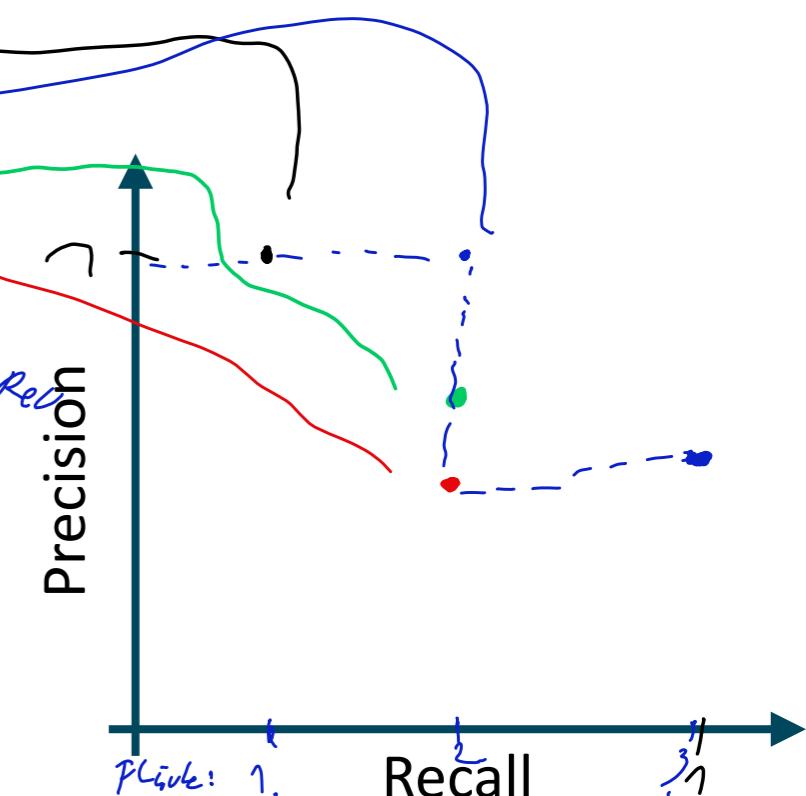
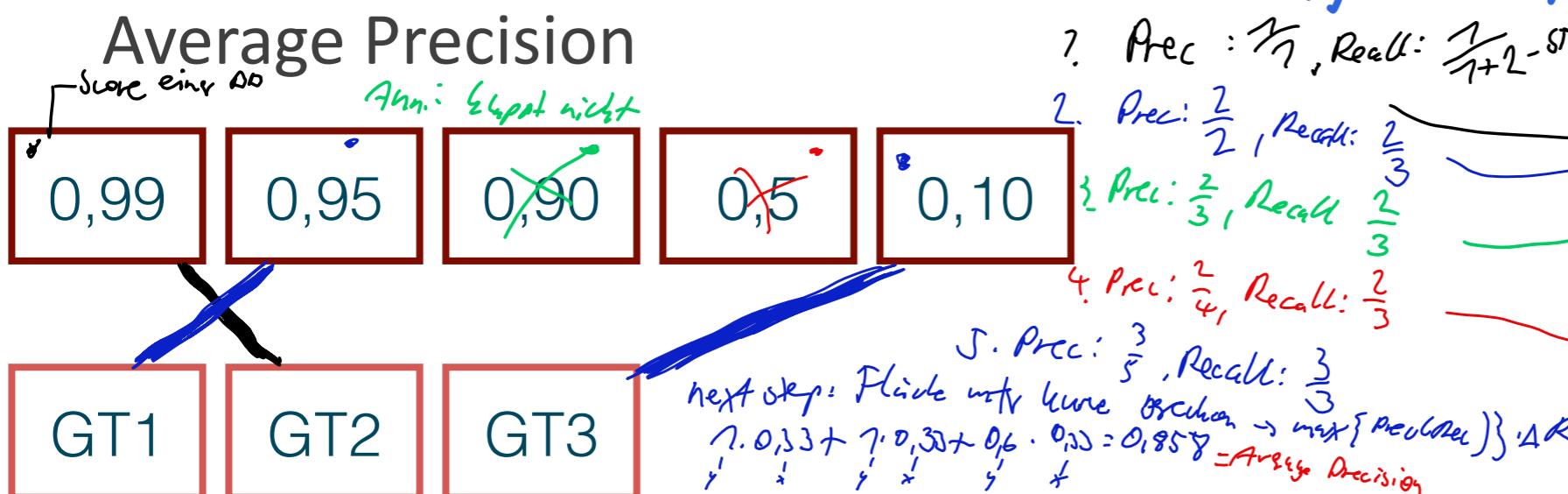
berechne Mittelwert der APs aller Label

$$\text{Recall} : \frac{TP}{TP + FN}$$

$$\text{Precision} : \frac{TP}{TP + FP}$$

1) sortiere BBs eines Labels nach dem Score

2) für alle Ground Truths berechne, was mind 50% sein



FP: keine GT-Box getroffen, obwohl DB vorhergesagt

FN: keine DB vorhergesagt \Rightarrow GT nicht getroffen

Objektdetektion

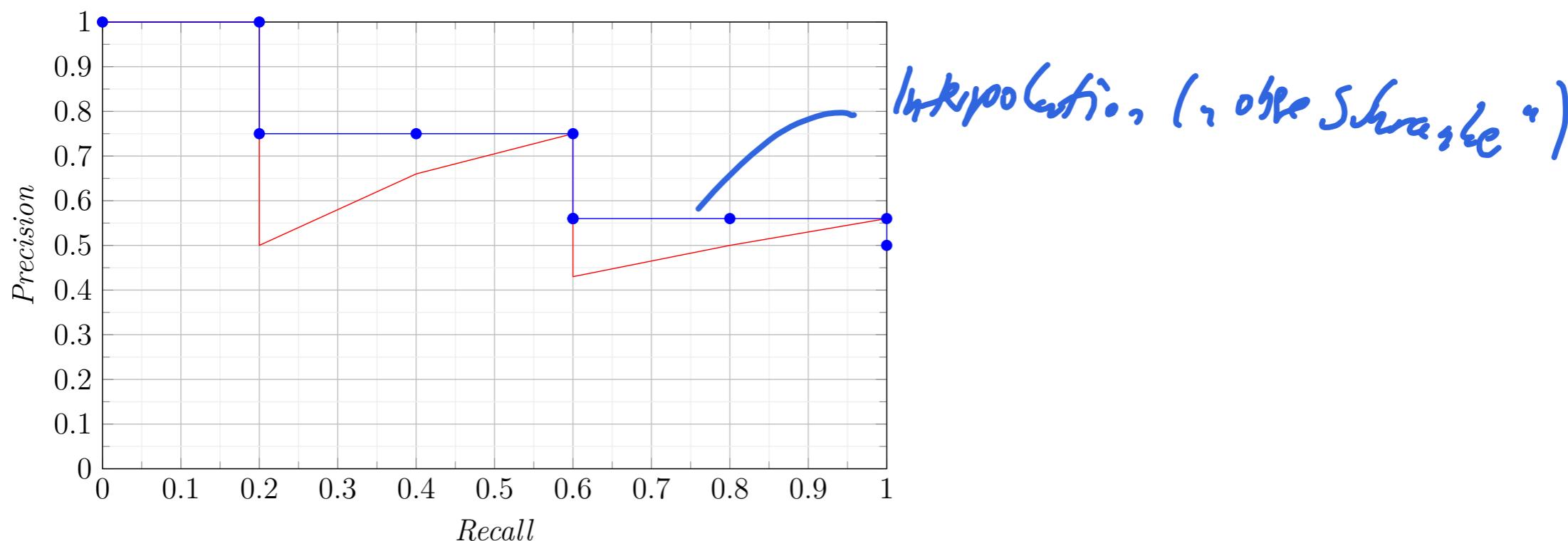
Fehlermaß zur Evaluation: Mean Average Precision (mAP)

für alle Testbilder: Objektdetektion inkl. Non-Maximum-Suppression

für jedes Label: berechnet die mittlere Genauigkeit (Average Precision, AP)

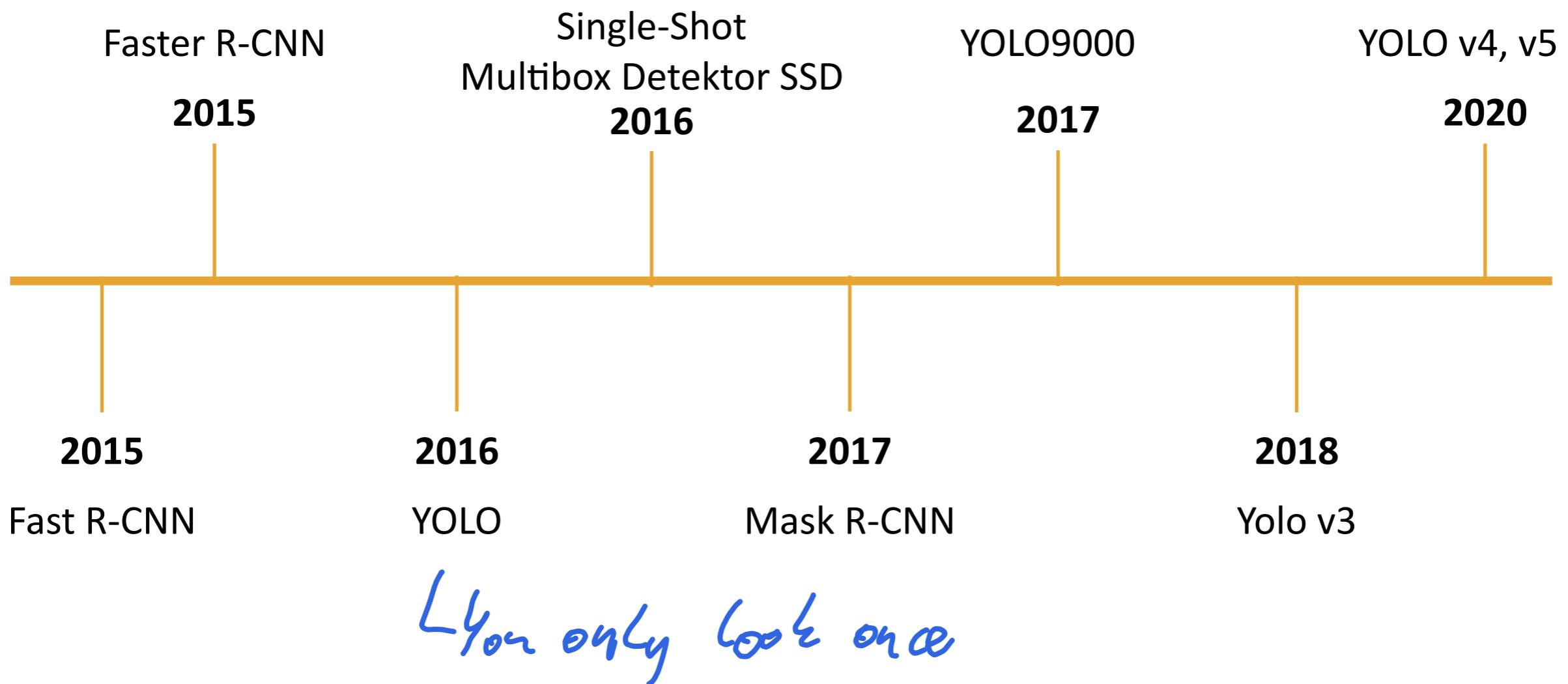
berechne Mittelwert der APs aller Label

Average Precision



Objektdetektion

Entwicklung





Objektdetektion

I. Grundlagen

2. Single-Stage-Detection

gibt auch
Multi-Stage

for our research purposes:

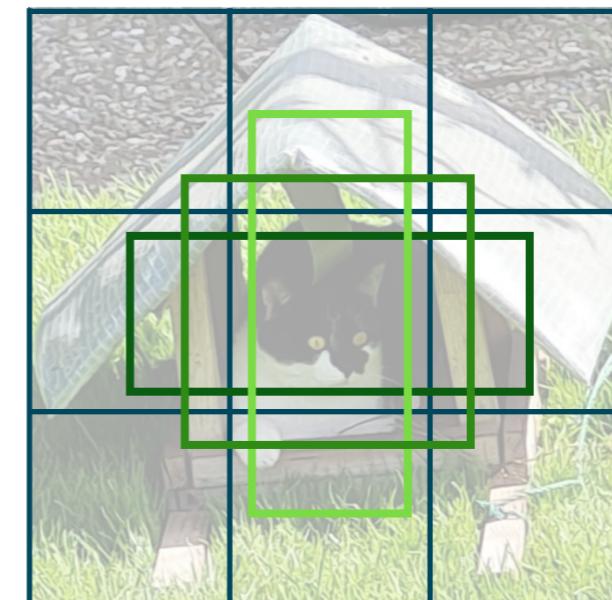
Single-Shot \neq Single-Stage

Objektdetektion

Single-Shot Objektdetektion

- Bild rastern
- Ankerboxen definieren → Mittelpunkt davon
= Rastermittelpunkt

→ haben verschiedene Seitenverhältnisse



© C. Hiller, ReMIC, OTH Regensburg

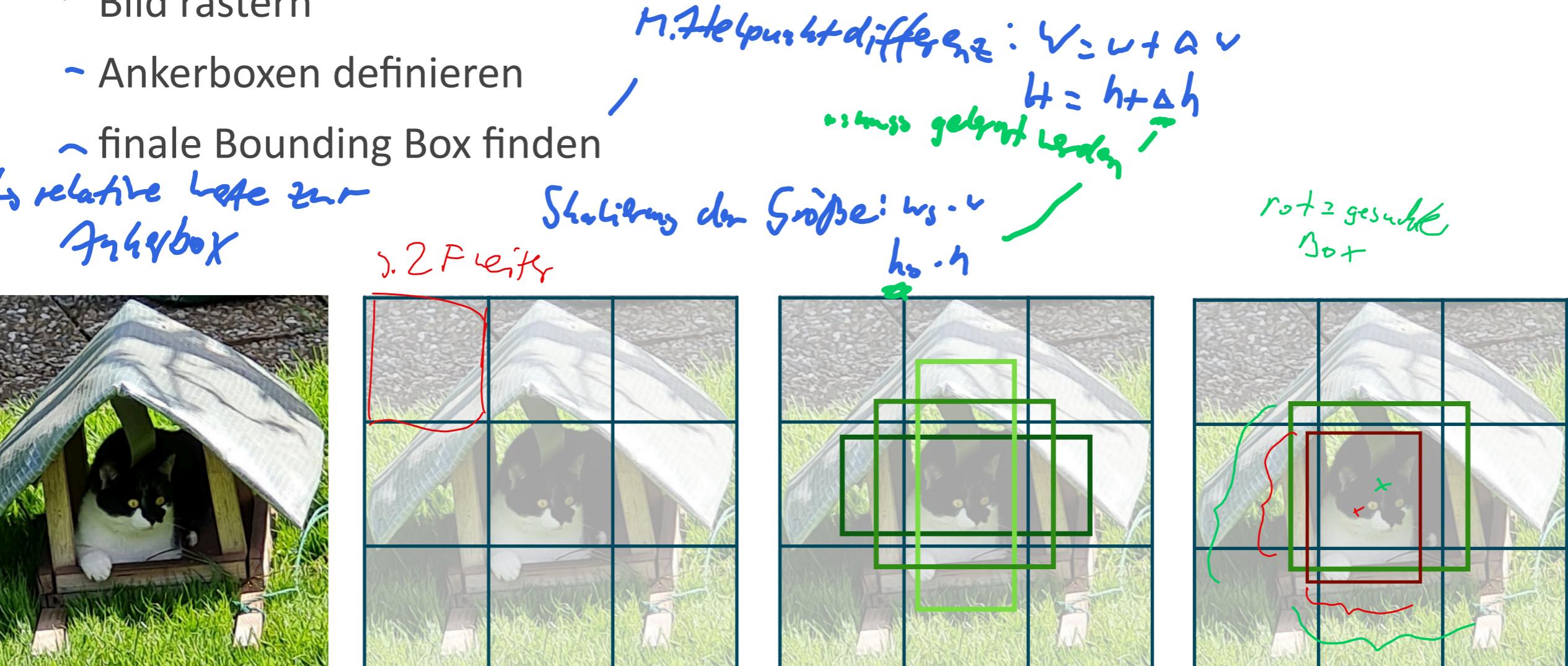
Objektdetektion

Single-Shot Objektdetektion

- ~ Bild rastern
- ~ Ankerboxen definieren
- ~ finale Bounding Box finden

↳ relative Lage zur
Ankerbox

© C. Hiller, ReMIC, OTH Regensburg



Objektdetektion

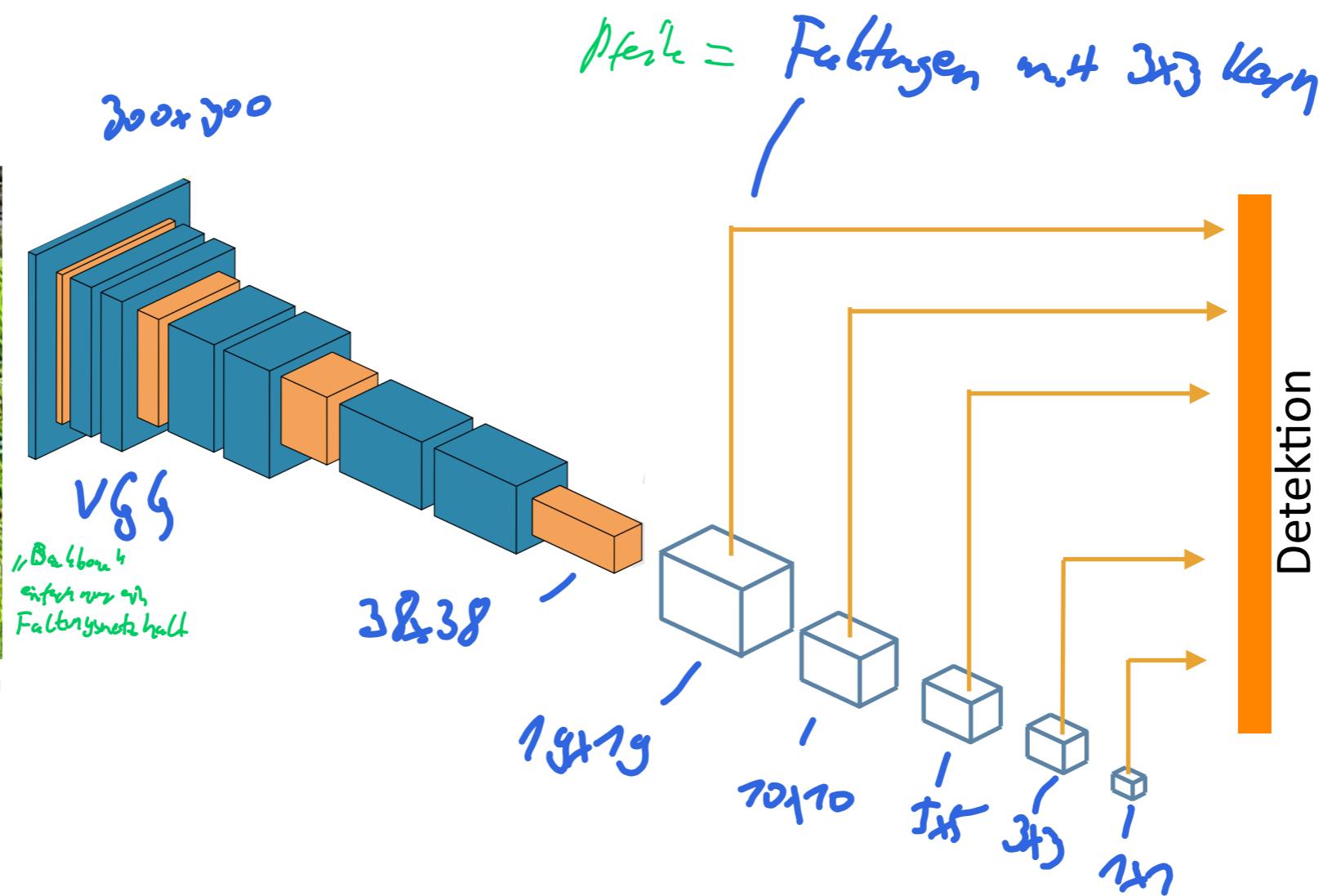
Single-Shot Objektdetektion

Faltungsnetz als Backbone

5 zusätzliche Faltungsschichten

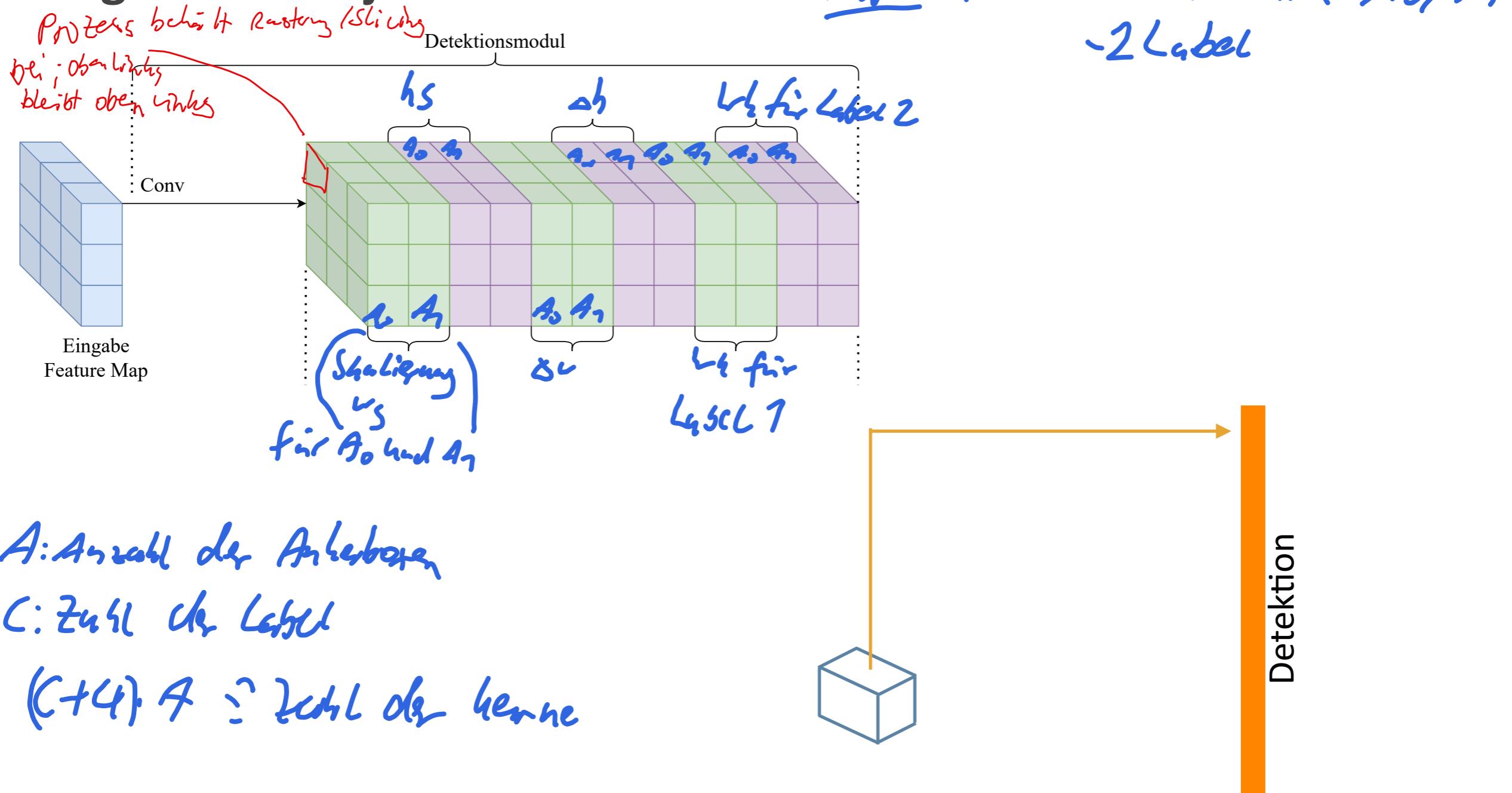


© C. Hiller, ReMIC, OTH Regensburg



Objektdetektion

Single-Shot Objektdetektion



Objektdetektion

Single-Shot Objektdetektion

„mit einem Netz hat man alles geschlagen“

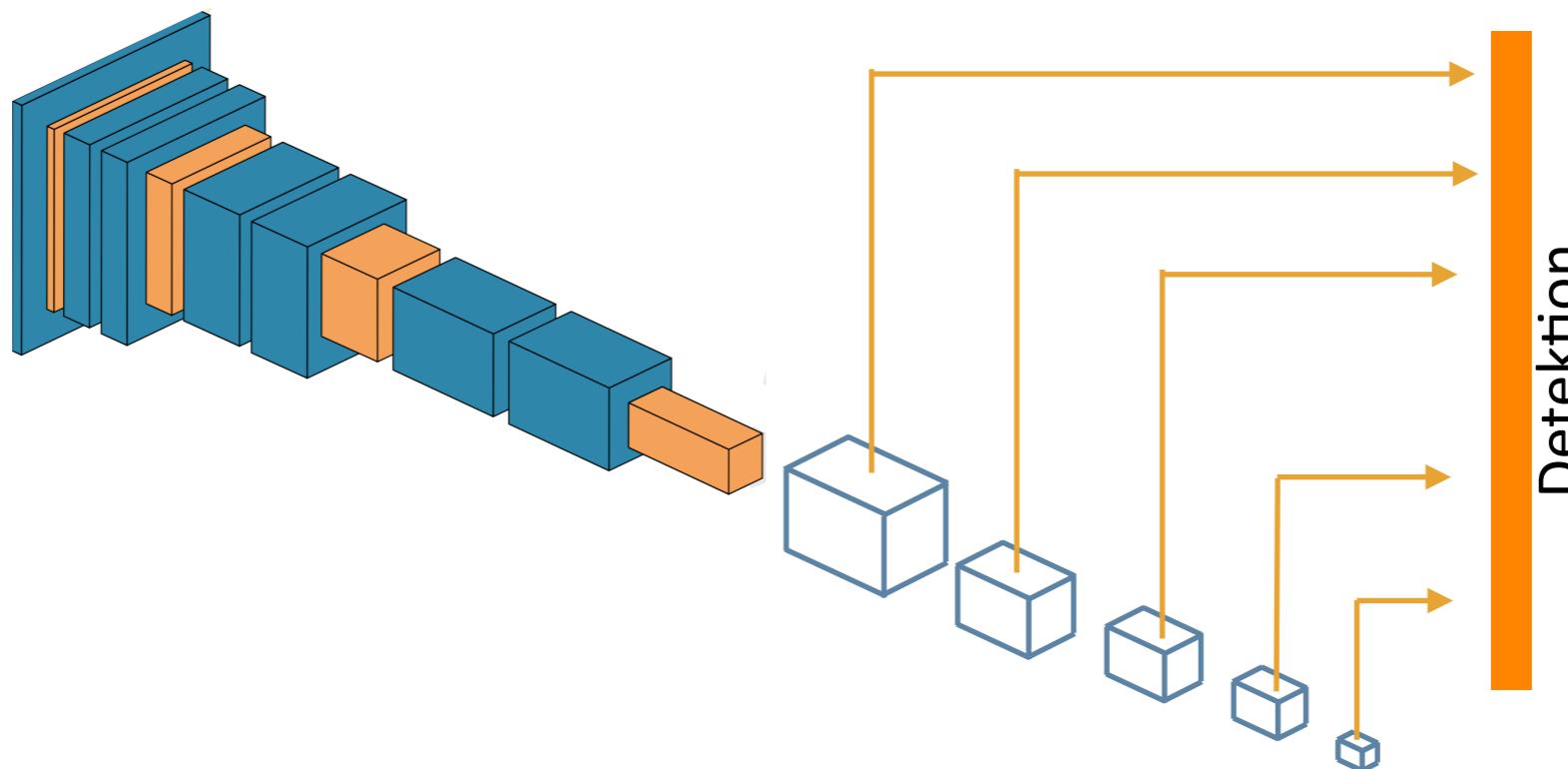
Faltungsnetz als Backbone

5 zusätzliche Faltungsschichten

Non-Maximum-Suppression



© C. Hiller, ReMIC, OTH Regensburg



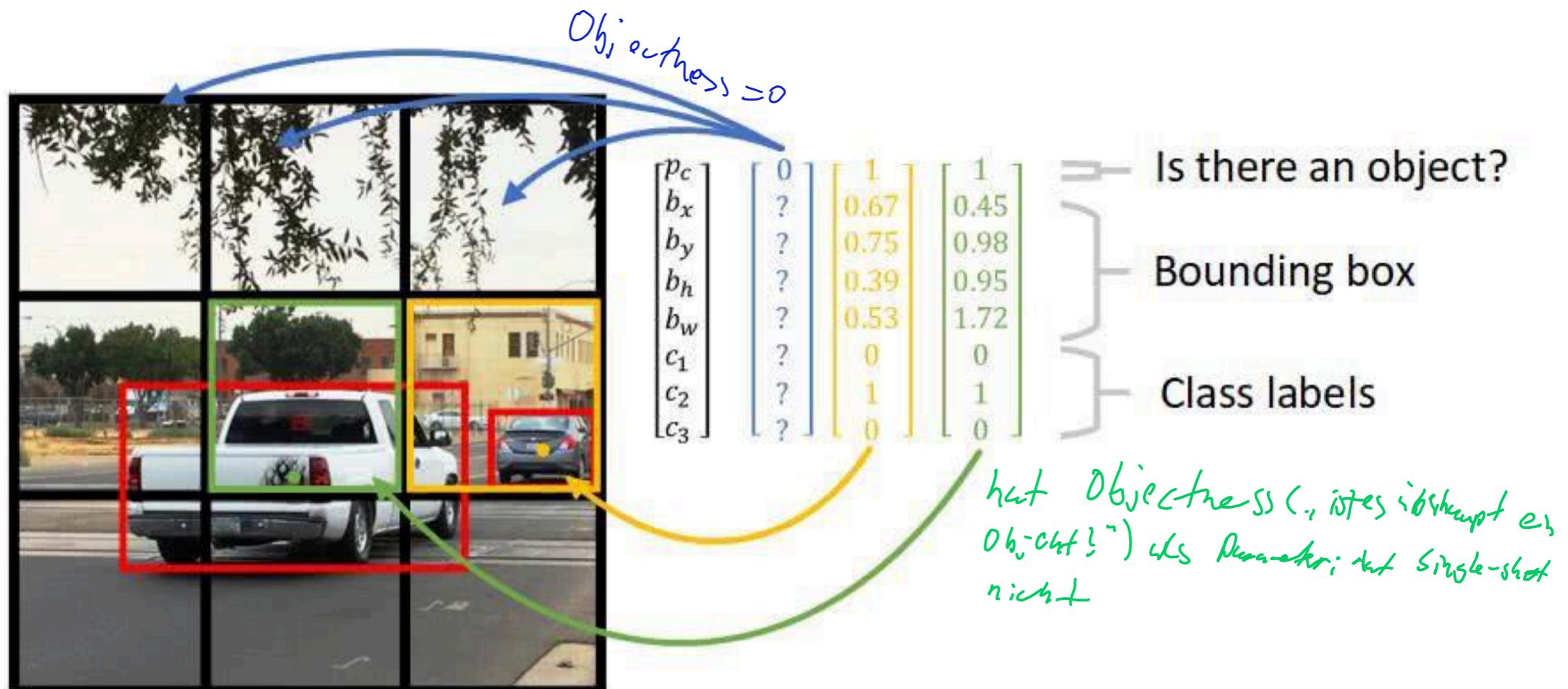
Objektdetektion

YOLO

GoogLeNet als Backbone

Konfidenz als zusätzlicher Parameter pro Bounding Box

Bsp: 3 Klassen



<https://heartbeat.fritz.ai/gentle-guide-on-how-yolo-object-localization-works-with-keras-part-2-65fe59ac12d>

Objektdetektion

YOLOv2 - YOLO 9000

war das heiße Schiff

Better

Batch Normalisierung

Ankerboxen durch Clustering der Trainingsdaten

Multiskalen-Training

Faster

andere Backbone-Architektur: DarkNet

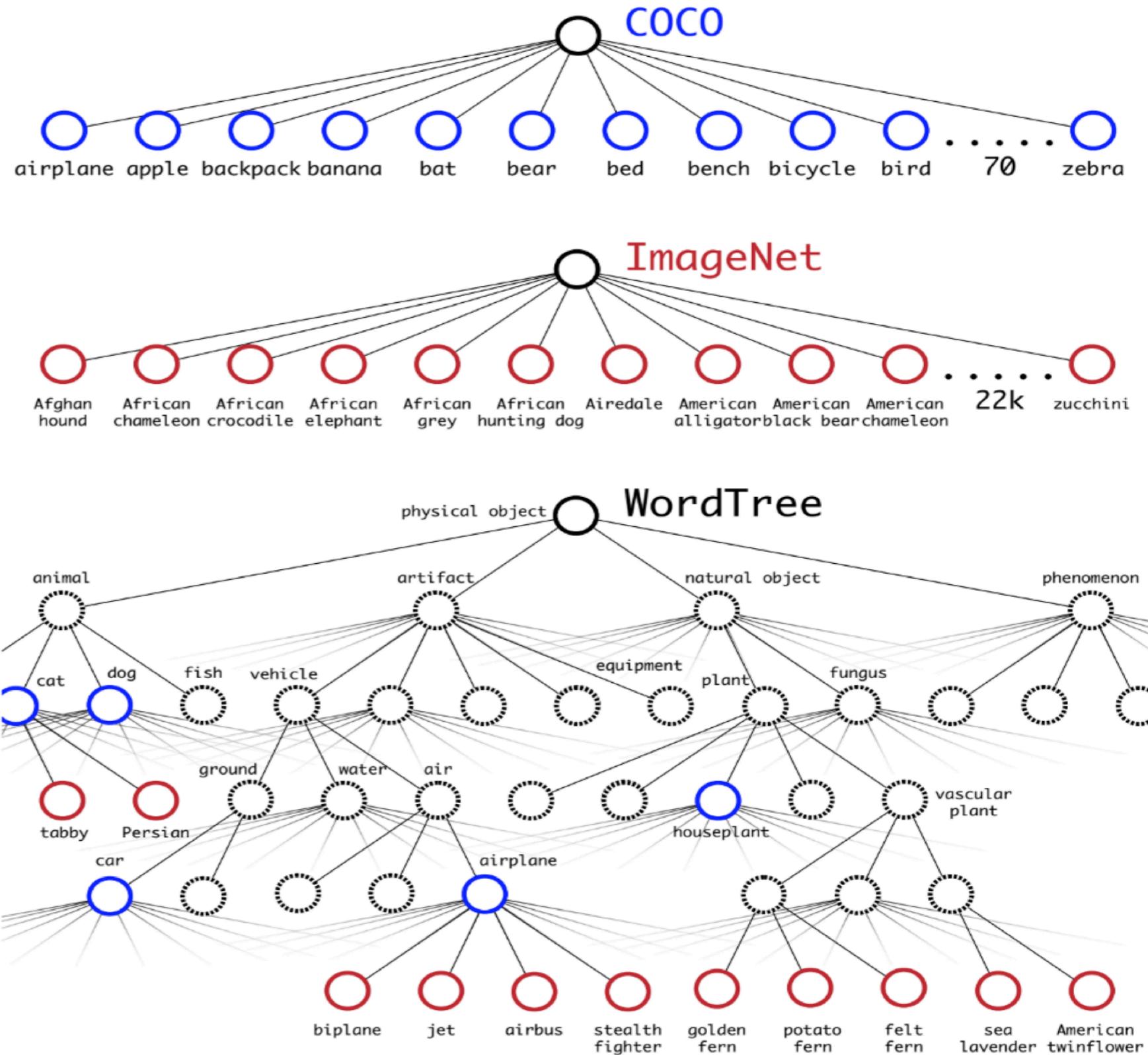
- hauptsächlich 3+3 Kone

Stronger

Hierarchische Klassifizierung

Objektdetektion

YOLOv2 - YOLO 9000

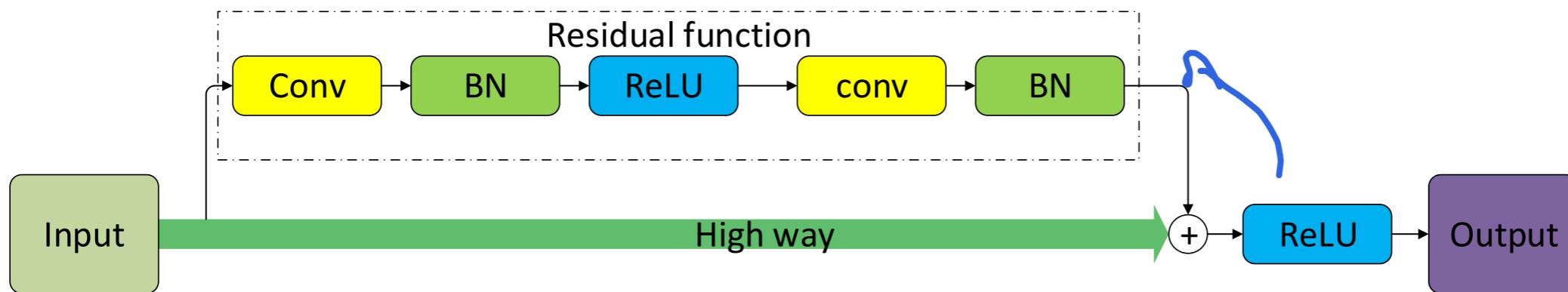


© K-T Lai 2021

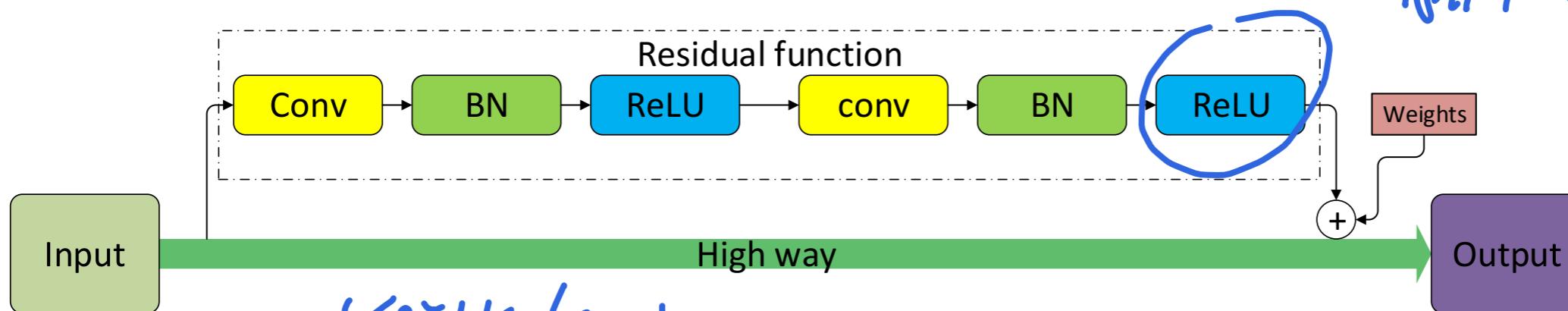
Objektdetektion

YOLOv4

Besonderheit = *Kennt man noch von den ResNets*
‘Weighted Residual Connections



Erst ist man durch Ausprobieren drauf gekommen *Weighted ResNet*



Weights lernbar: -1, 1 verkehrt

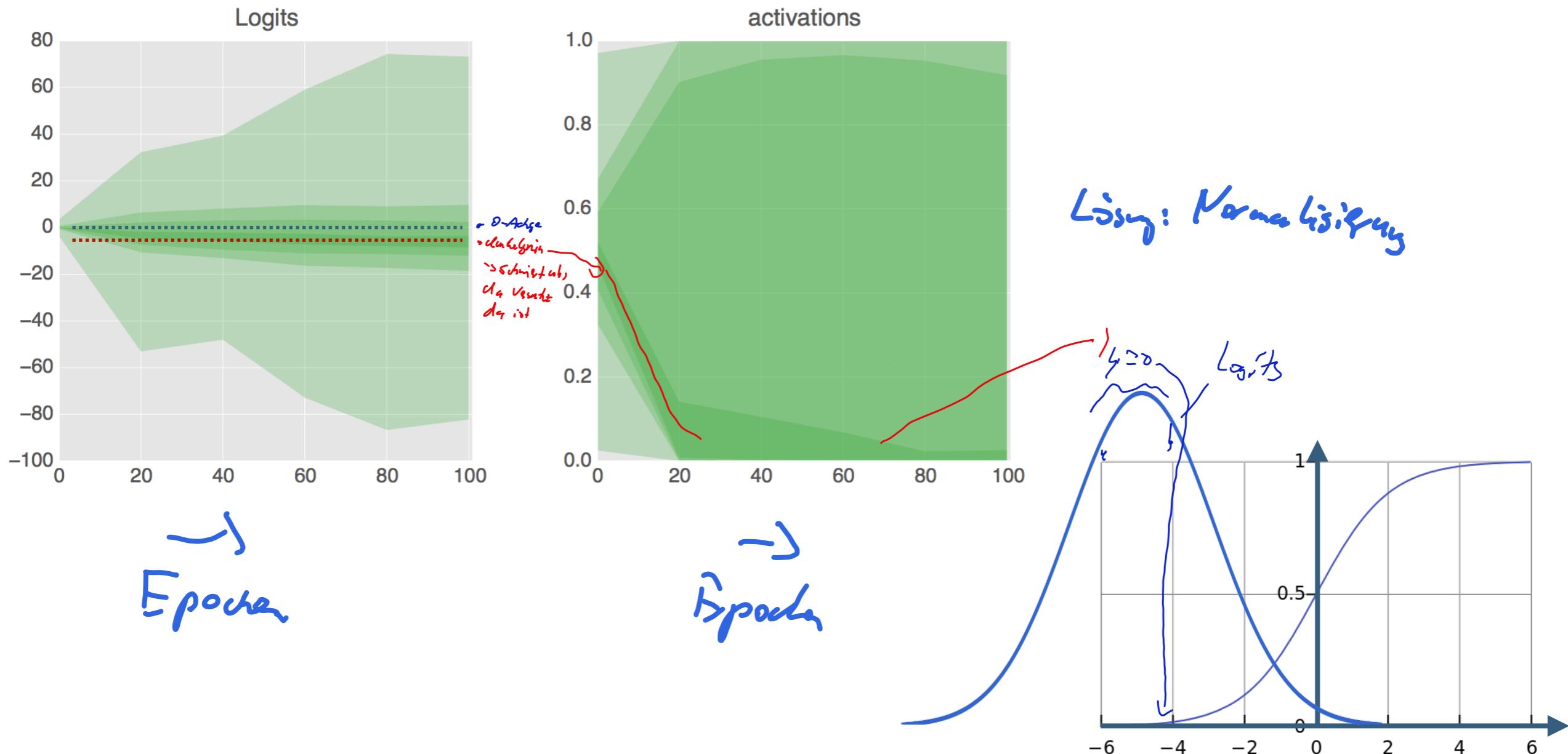
⇒ besonders bei sehr tiefen NN empfohlen!

© Shen et al., 2016

Objektdetektion

YOLOv4

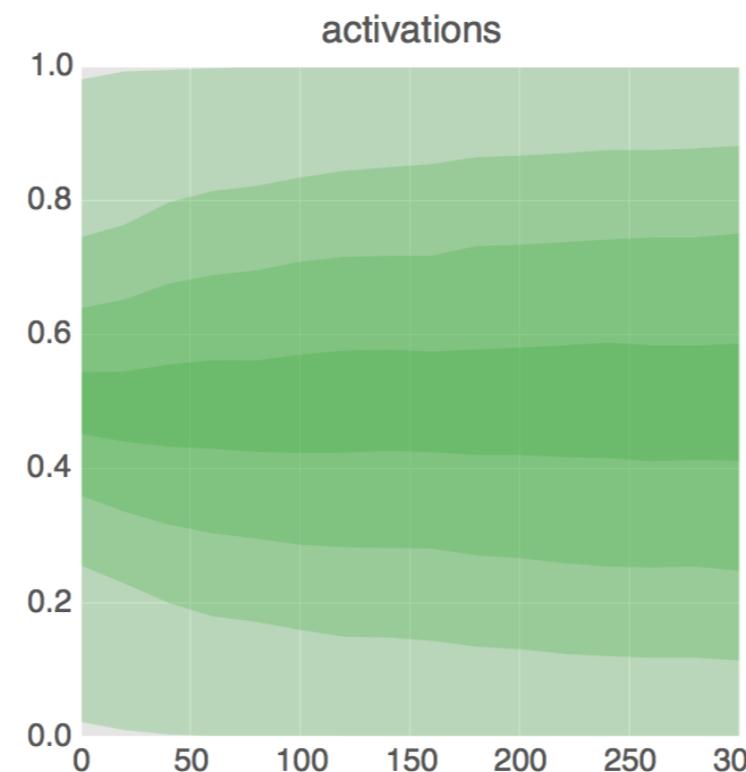
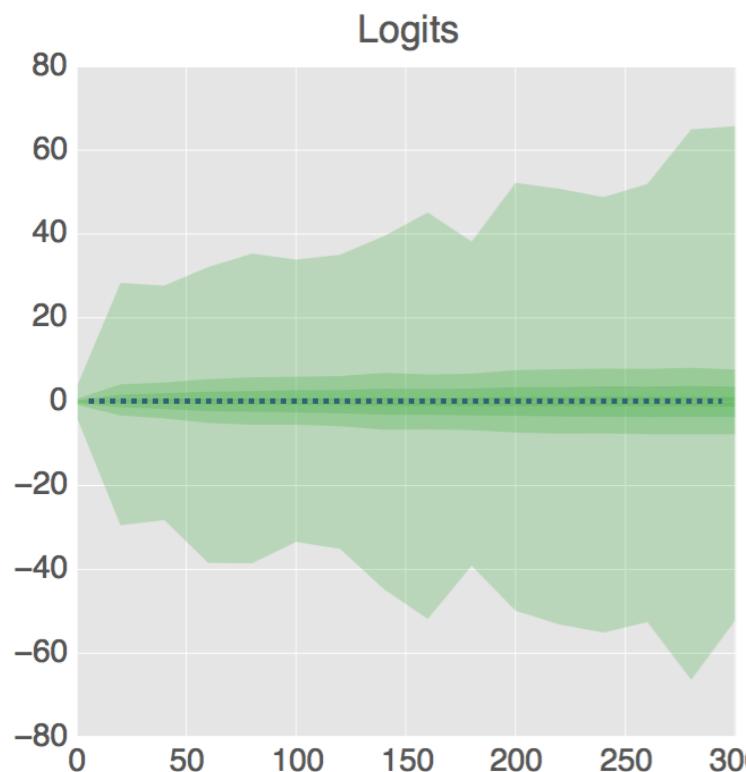
Batch Normalization



Objektdetektion

YOLOv4

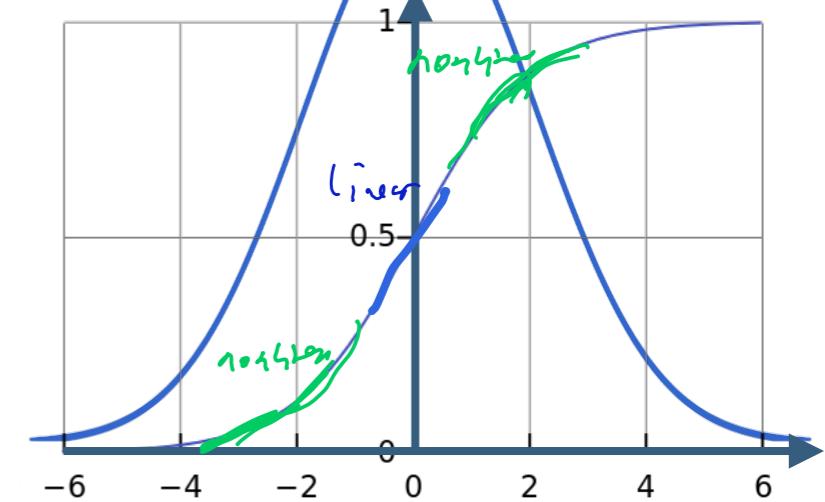
Batch Normalization



✓ Verschieben tun
Nullpunkt

Bsp: horizont. Test wäre interessant

ReLU wäre cooler, da Nichtlinearität
in Nullpunkt steckt



Objektdetektion

YOLOv4

$x_{t,i}$: ? iste Logit im t-ten Batch eines bestimmten Layers

Batch-Training: man nimmt kleine Teile von den Trainingsdaten und berechnet dadurch eine Gradientenschritt; wiederhole für alle Trainingsdaten; das ist dann eine Epoche

Batch Normalization

θ_t : Gewichte eines NN für den t-ten Batch die sich jedes Mal

$$\hat{x}_{t,i} = \frac{x_{t,i}(\theta_t) - \mu_t(\theta_t)}{\sigma_t(\theta_t) + \epsilon}$$

auf θ normieren = mittlerer abzüglich Standardabweichung (ϵ)

$$\mu_t(\theta_t) = \frac{1}{M} \sum_{i=1}^M x_{t,i}(\theta_t)$$

recht günstig; Bias wird weggelassen
→ Neue lernbare Parameter pro Schicht
= ∞

$$\sigma_t(\theta_t) = \sqrt{\frac{1}{M} \sum_{i=1}^M (x_{t,i}(\theta_t) - \mu_t(\theta_t))^2} = \sqrt{\nu_t(\theta_t) - \mu_t(\theta_t)^2}$$

$$\nu_t(\theta_t) = \frac{1}{M} \sum_{i=1}^M x_{t,i}(\theta_t)^2$$

α, β sind lernbar
↳ NN lernt, ob Normalisierung erforderlich ist

$$x_{t,i, BN}(\theta_t) = \alpha \hat{x}_{t,i}(\theta_t) + \beta$$

Objektdetektion

YOLOv4

Idee: Idee: Idee von YOLO mitberücksichtigen; Schiefeheit dazu:

zum aktuellen Zeit wissen, was wir den, um ich den vollen Satz nehmen würde?

→ Mittelwert der Logits des Batches berechnen, was dann genutzt für die Schiefe ist

Batch Normalization

$$\hat{x}_{t,i} = \frac{x_{t,i}(\theta_t) - \mu_t(\theta_t)}{\sigma_t(\theta_t) + \epsilon}$$

$$\mu_t(\theta_t) = \frac{1}{M} \sum_{i=1}^M x_{t,i}(\theta_t)$$

$$\sigma_t(\theta_t) = \sqrt{\nu_t(\theta_t) - \mu_t(\theta_t)^2}$$

$$\nu_t(\theta_t) = \frac{1}{M} \sum_{i=1}^M x_{t,i}(\theta_t)^2$$

$$x_{t,i,BN}(\theta_t) = \alpha \hat{x}_{t,i}(\theta_t) + \beta$$

⇒ Achtung: $\mu_{t,h}$ bereitet sich auf die Schiefe für h , nicht für $t-h$

$$\hat{x}_{t,i} = \frac{x_{t,i}(\theta_t) - \bar{\mu}_{t,K}(\theta_t)}{\bar{\sigma}_{t_K}(\theta_t) + \epsilon}$$

$$\bar{\mu}_{t,K}(\theta_t) = \frac{1}{K} \sum_{k=0}^{K-1} \mu_{t-k}(\theta_t)$$

⇒ Taylorreihenapproximation

$$\bar{\sigma}_{t,K}(\theta_t) = \sqrt{\bar{\nu}_{t,K}(\theta_t) - \bar{\mu}_{t,K}(\theta_t)^2}$$

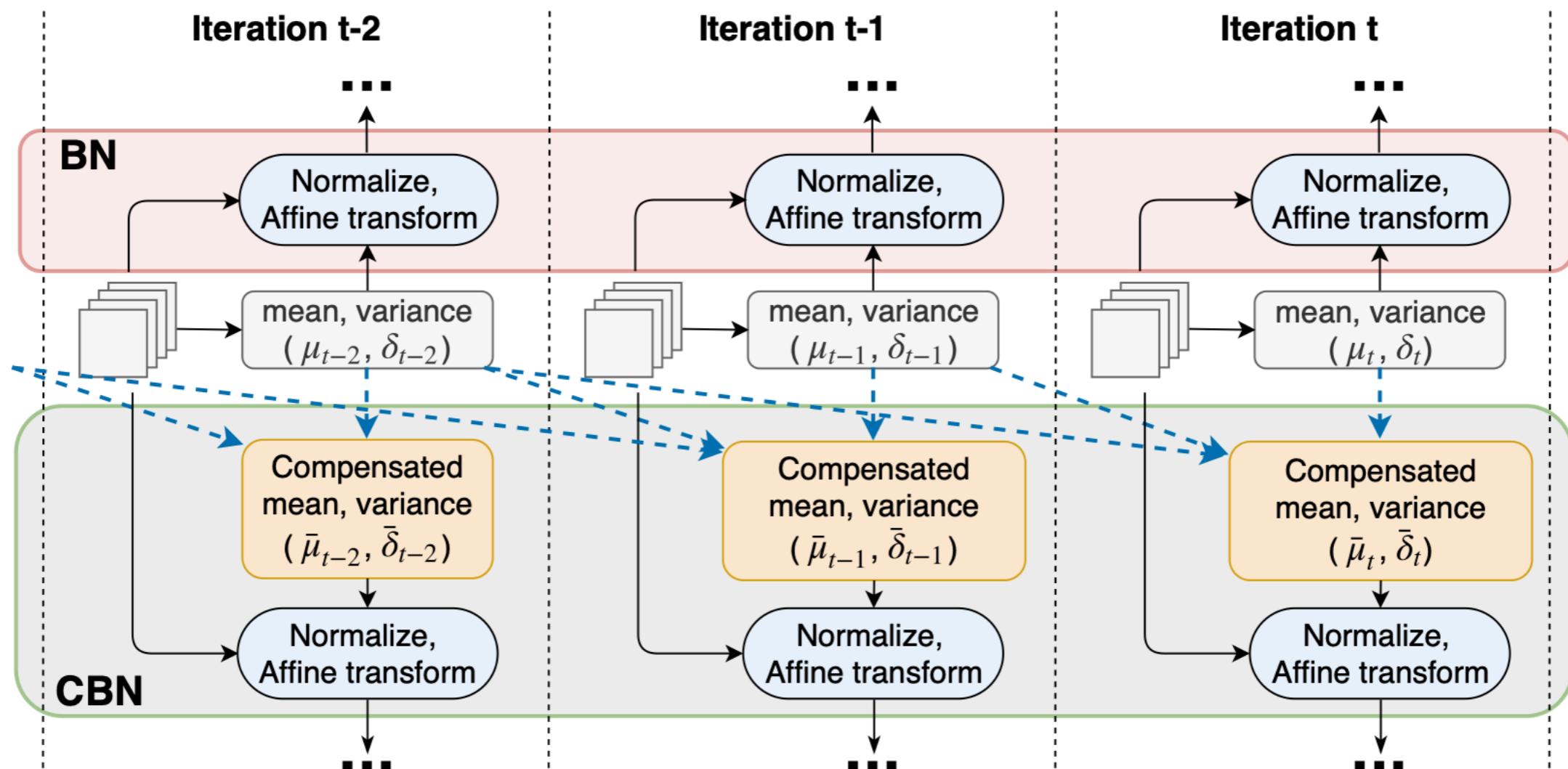
$$\bar{\nu}_{t,K}(\theta_t) = \frac{1}{K} \sum_{k=0}^{K-1} \max \{ \nu_{t-k}(\theta_t), \mu_{t-k}(\theta_t)^2 \}$$

(μ_{t-k} muss μ_{t-k} sein ⇒ wofür wir sorgen müssen)

Objektdetektion

YOLOv4

Cross-Iteration Batch Normalization



Einschub: Adversarial Attack

Gegnerisches Beispiel (adversarial example)

unbemerkbare, nicht-zufällige Veränderung eines Testbildes mit der Folge, dass sich die Vorhersage eines Klassifikationsnetzes ändert

Label	
Katze	0,93
Hund	0,03
Kaninchen	0,04



© C. Hiller, ReMIC, OTH Regensburg

Einschub: Adversarial Attack

Gegnerisches Beispiel (adversarial example)

unbemerkbare, nicht-zufällige Veränderung eines Testbildes mit der Folge, dass sich die Vorhersage eines Klassifikationsnetzes ändert

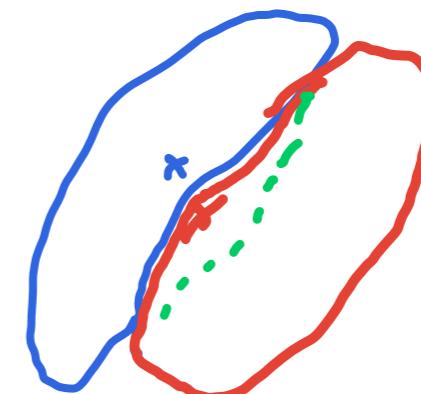


© C. Hiller, ReMIC, OTH Regensburg

Label	
Katze	0,7
Hund	0,91
Kaninchen	0,02

0,107

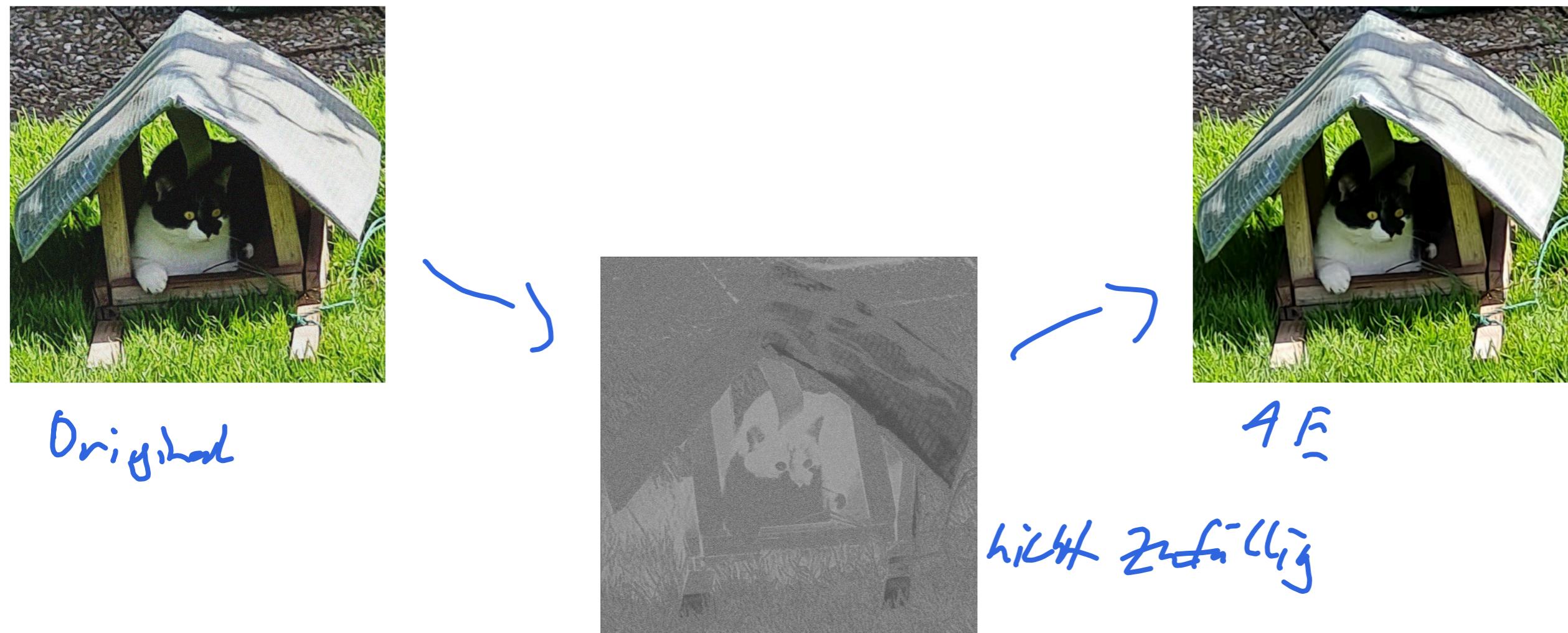
→ „blinde Fleche“ des
NN ausnutzen



Einschub: Adversarial Attack

Gegnerisches Beispiel

unbemerkbare, nicht-zufällige Veränderung eines Testbildes mit der Folge, dass sich die Vorhersage eines Klassifikationsnetzes ändert



Einschub: Adversarial Attack

Gegnerisches Beispiel

*Ergebnisse für J. Spaltk sind hier
ALLE „Vogelstrauß“*

unbemerkbare, nicht-zufällige Veränderung eines Testbildes mit der Folge, dass sich die Vorhersage eines Klassifikationsnetzes ändert

↗ verdeckt dargestellte
Veränderung



© C. Szegedy et al. 2013

Prof. Dr. Christoph Palm

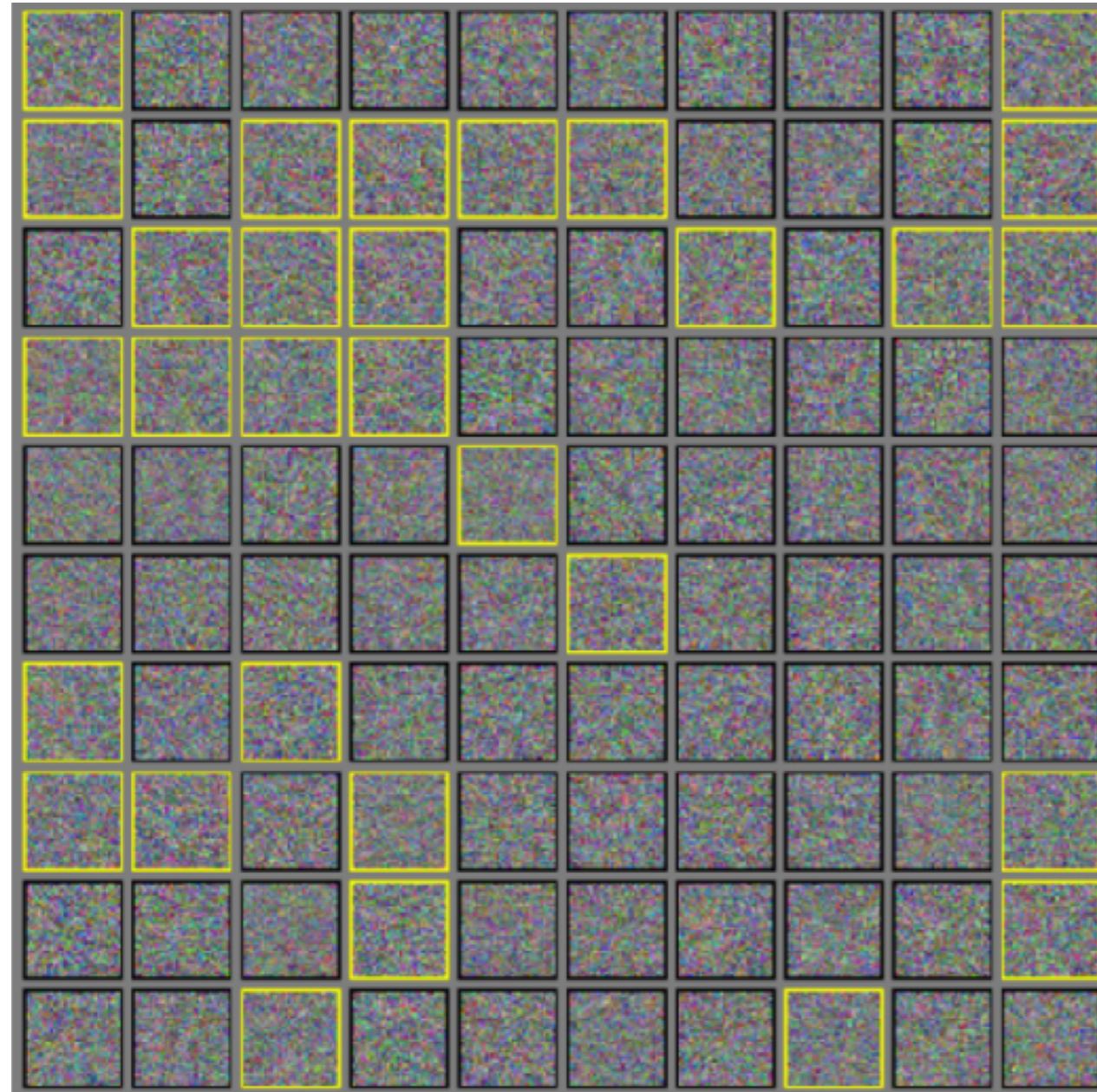
Regensburg Medical Image Computing (ReMIC)

Ostbayerische Technische Hochschule Regensburg (OTH Regensburg)

Einschub: Adversarial Attack

Gegnerisches Beispiel

Starten mit zufälligem Rauschen, dann ein Update-Schritt



© Washington University

z.B. 5% w für die Klasse

$$y = \tau_0 + \tau_1 w_1 + \tau_2 w_2 + \dots$$

$$\frac{\partial L(y)}{\partial w_1} = \dots x_1$$

man kann Gradient nach Gewichten
oder nach Pixeln berechnen

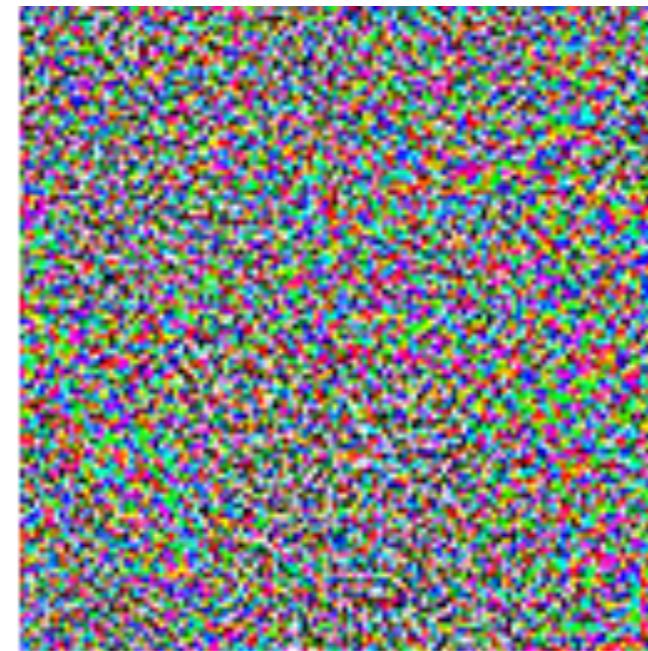
$$\frac{dy}{dx_1} = w_1$$

Einschub: Adversarial Attack

Gegnerisches Beispiel



$+\epsilon$



=



Panda : 57%

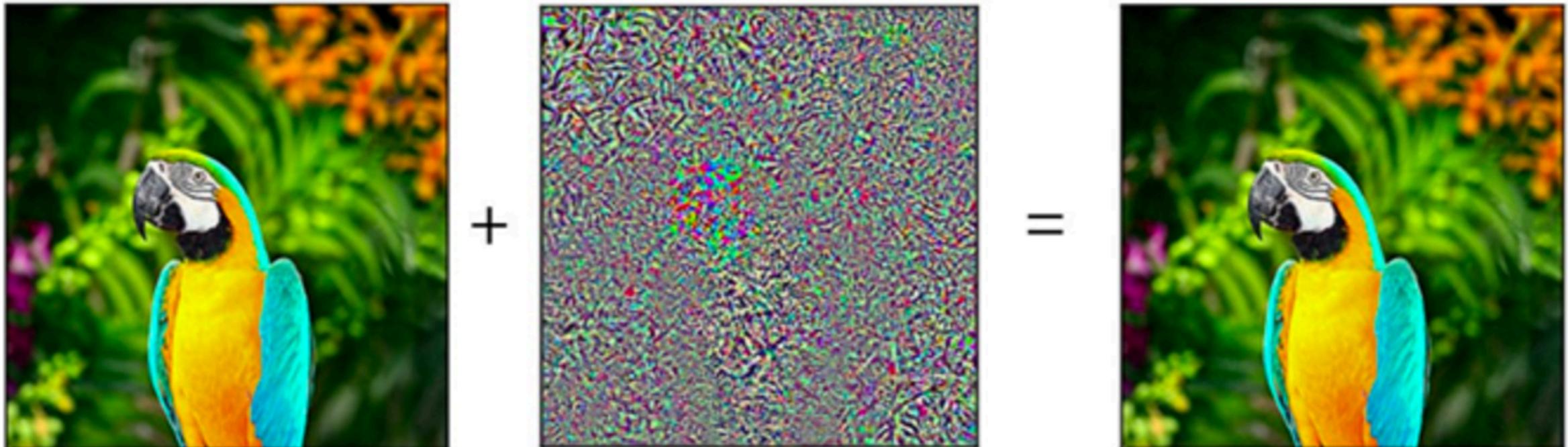
Gibbon: 99,3%

© Goodfellow et al.

© <https://www.youtube.com/watch?v=oZYgaD004Dw>

Einschub: Adversarial Attack

Gegnerisches Beispiel



© <https://www.youtube.com/watch?v=oZYgaD004Dw>

Ara: 97, 3%

Bücherei? 83, 9%

Einschub: Adversarial Attack

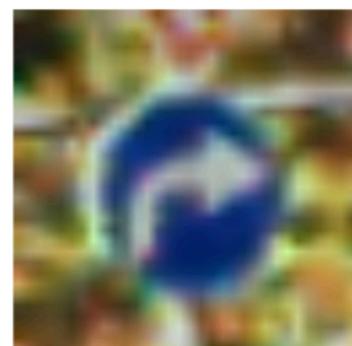
Gegnerisches Beispiel



0



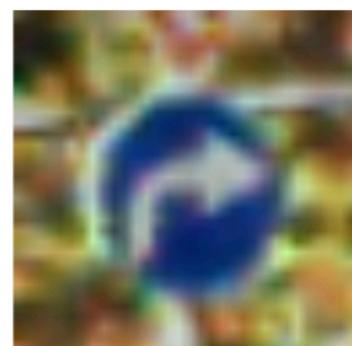
3



5



8



MNIST Dataset

GTSRD Dataset

© Papernot et al, 2017

Einschub: Adversarial Attack

White-Box Methode

→ Netz ist bekannt

LSDN ist + bekannt und trainiert

Definiere neue Loss-Funktion

$x: \mathbb{B}^d$

$r: \text{Störung des Bildes}$

$f(x): \text{Klassifikation}$

$l: \text{zu erzeugendes Label (falsch)}$

Ann: $f(x)$ Klassifikator ist gut trainiert und gibt nicht das falsche Label aus

Verstehen, wie sich die Lossfunktion zusammensetzt

$$\min \underbrace{\{c|r|}_{\text{Ziel: } r \text{ ist klein}} + \underbrace{D_f(x + r, l)}_{\substack{\text{Störung} \\ \text{Bild} \\ \text{Lossfunktion zu falschem Label}}} \quad \begin{array}{l} \text{Methode: Gradientenabstieg} \\ \text{Richtungsableitung; } L \text{ soll rauskommen} \\ \text{Approximation der Hesse-Matrix} \\ \text{Quasi-Newton-Verfahren} \end{array}$$

↪ DFgs: Dogrelle - Fletcher - Goldfarb
Schem

Einschub: Adversarial Attack

White-Box Methode

Gradient kann man nur berechnen, wenn man das Netz kennt -> nur bei White-Box

Fast Gradient Sign Methode

$$r = -\epsilon \cdot \text{sign}(\nabla_x D(\theta, x, y))$$

Gradient nach Pixeln
(statt gewichten)

↳ Entfernung von der korrekten Klasse

Einschub: Adversarial Attack

White-Box Methode

Beobachtungen:

- ↳ AEs sind häufig universell
 - ⇒ lässt sich auf verschiedenen Architekturen anwenden
 - ⇒ lässt sich auf Modelle aus unterschiedlichen Trainingsdaten anwenden

*Design muss nur gelernt sein, (<50 Autos/Schilder,...)
Architektur des Netz kann völlig verschieden sein
> evtl. eine möglichkeit Architektur universell zu beschreiben?*

Einschub: Adversarial Attack

Black-Box Methode

Gegeben:

Modell, das zu einem beliebigen Input eine Vorhersage produziert

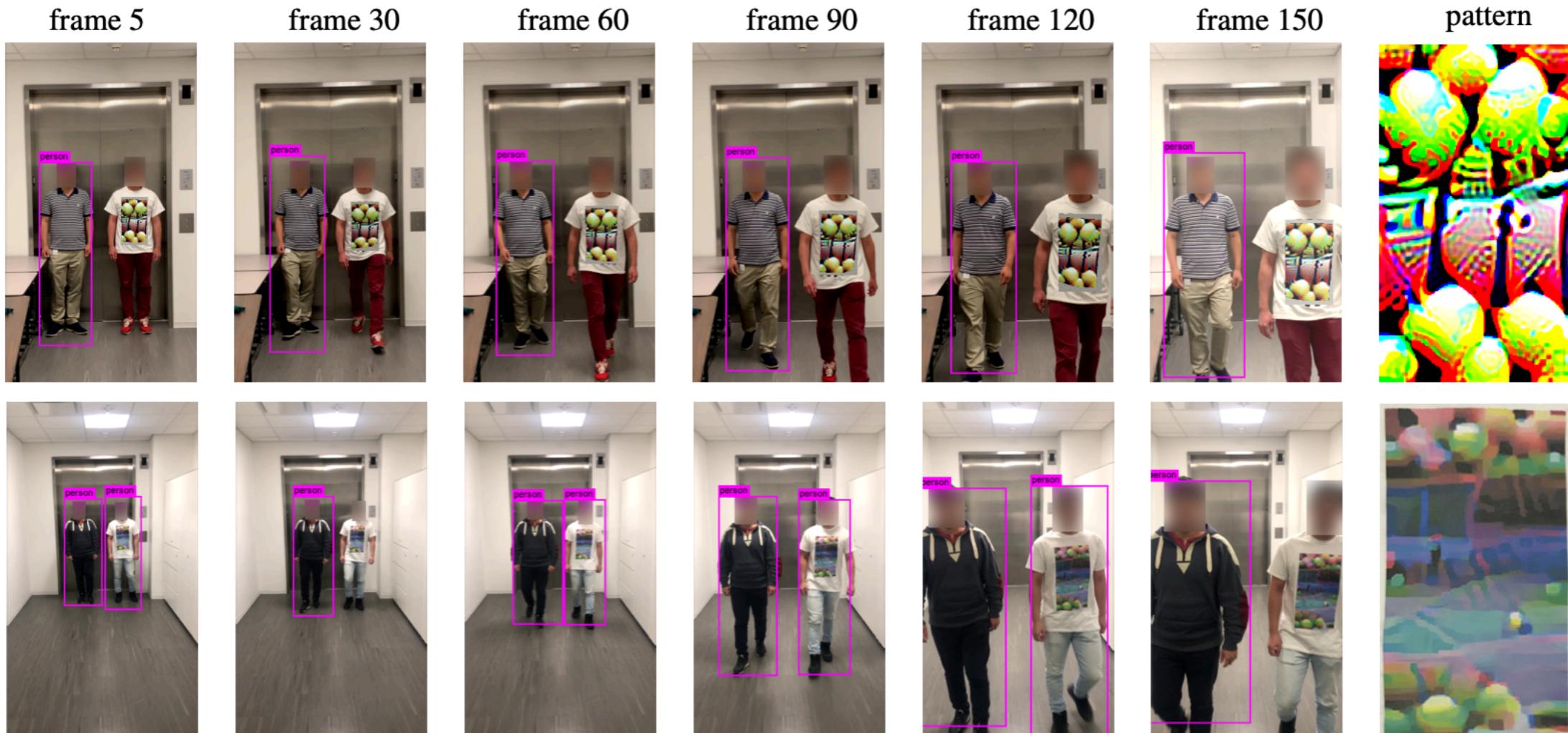
Idee:

Bilde ein neues „Ersatznetz“ mit den gleichen Eigenschaften

Nutze ein White-Box Verfahren zur Generierung von Adversarial Examples

Einschub: Adversarial Attack

Gegnerisches Beispiel in der realen Welt



Adversarial T-Shirt

© Xu et al., 2020

Einschub: Adversarial Attack

Gegnerisches Beispiel in der realen Welt



Clean Stop Sign



"Stop sign"



Real-world Stop Sign
in Berkeley

"Stop sign"



Adversarial Example

"Speed limit sign 45km/h"



Adversarial Example

"Speed limit sign 45km/h"

© Anmol Rajpurohit, 2018

nur mit Anführern erzeugt

→ in Praxis schon mal möglich

Einschub: Adversarial Attack

Gegnerisches Beispiel in der realen Welt



schwarz = Schildkröte

rot = Gelehr

© Washington University

Prof. Dr. Christoph Palm

Regensburg Medical Image Computing (ReMIC)

Ostbayerische Technische Hochschule Regensburg (OTH Regensburg)

Einschub: Self-Supervised Training

Training ohne gegebene Grundwahrheit (ground truth)

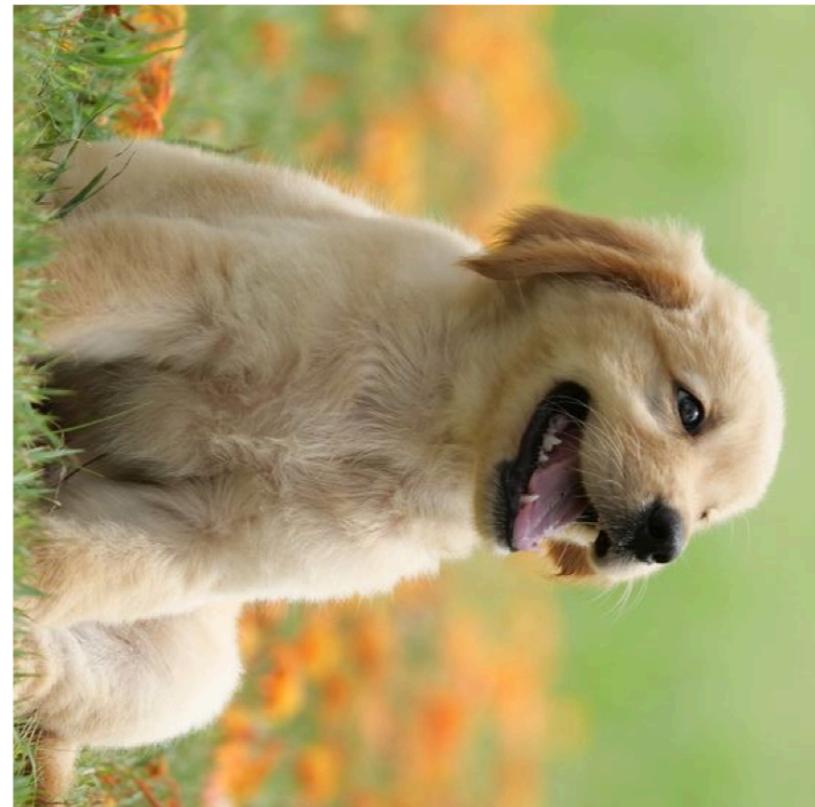
Idee:

Trainiere ein ganz anderes Problem

270°



90°



© Megan Leszczynski

Aufgabe: Erkenn e die Rotationen

Einschub: Self-Supervised Training

Training ohne gegebene Grundwahrheit (ground truth)

Idee:

Trainiere ein ganz anderes Problem



Aufgabe : Bring die Teilstücke in die richtige Reihenfolge

Einschub: Self-Supervised Adversarial Training

Gegeben: Originalbild und korrespondierendes Adversarial Example

Aufgabe: Maximiere die Mutual Information

2-stufiges Verfahren:

1) generiere Adversarial Example

2) Adaptiere die Schritte eines NIVs, das festig trainiert ist,
so dass es robust gegen AE ist

Einschub: Mutual Information

Entropie von Bild f und m

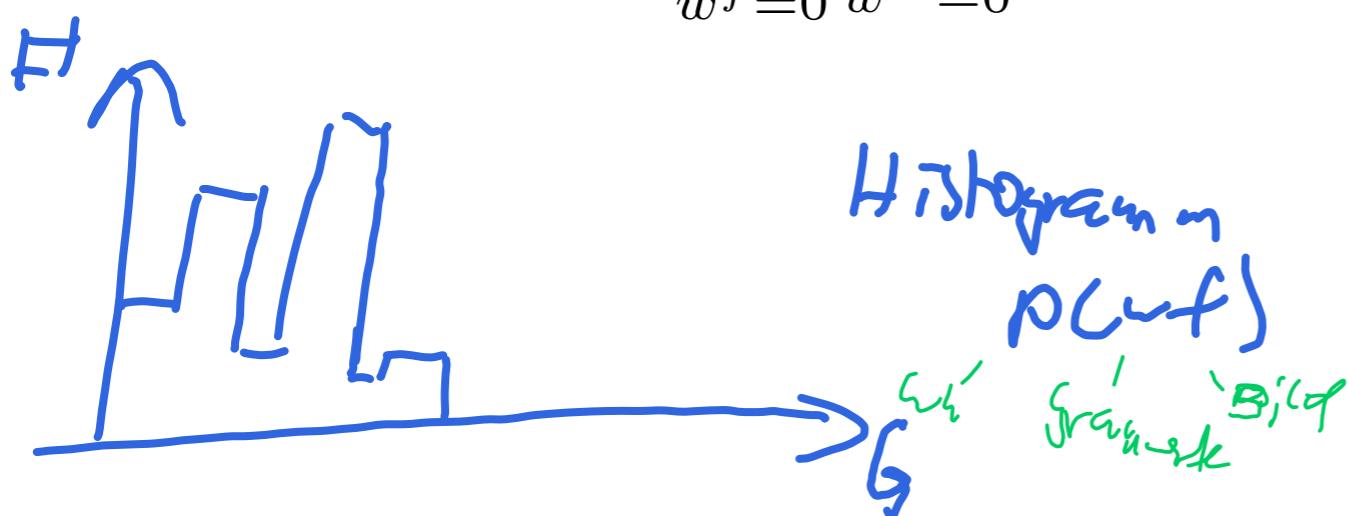
$$H(m) = - \sum_{w^m=0}^{W-1} p(w^m) \log p(w^m)$$

$$H(f) = - \sum_{w^f=0}^{W-1} p(w^f) \log p(w^f)$$

Ihre für das Auftreten des
Strukturen w in Bild m

Gemeinsame Entropie

$$H(f, m) = - \sum_{w^f=0}^{W-1} \sum_{w^m=0}^{W-1} p(w^f, w^m) \log p(w^f, w^m)$$



Einschub: Mutual Information

verschiedene Definitionen

I ist groß, wenn die Verteilungen $p(w^f)$ und $p(w^m)$ abhängig sind

$$I(f, m) = H(f) - H(f|m) = H(m) - H(m|f)$$

$$I(f, m) = H(f) + H(m) - H(f, m)$$

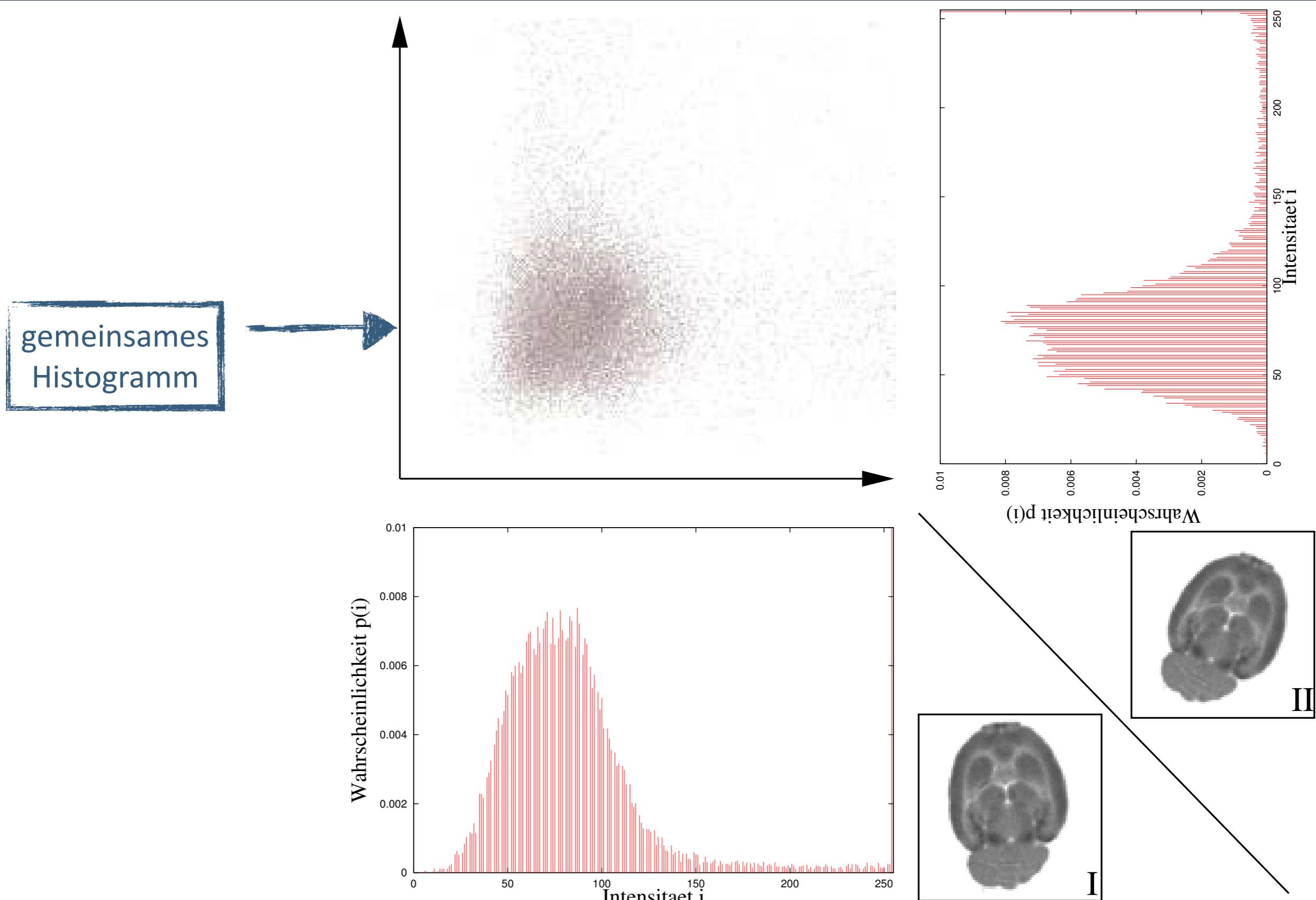
$$I(f, m) = \frac{H(f) + H(m)}{H(f, m)}$$

K-L-Divergenz zwischen der
Multiplikation der Einzelverteilungen
und der gemeinsamen Verteilung

$$I(f, m) = \sum_{w^f, w^m} p(w^f, w^m) \log \frac{p(w^f, w^m)}{p(w^f)p(w^m)}$$

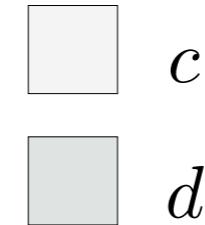
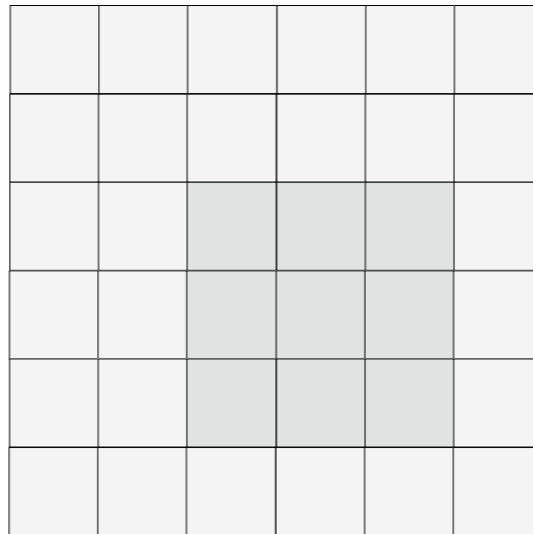
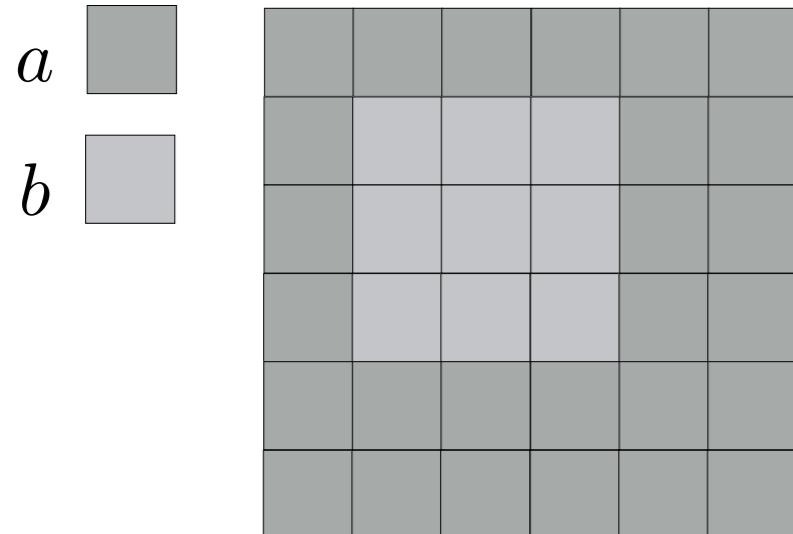
Kullback-Leibler-Divergenz

Einschub: Mutual Information



Einschub: Mutual Information

Gemeinsames Histogramm

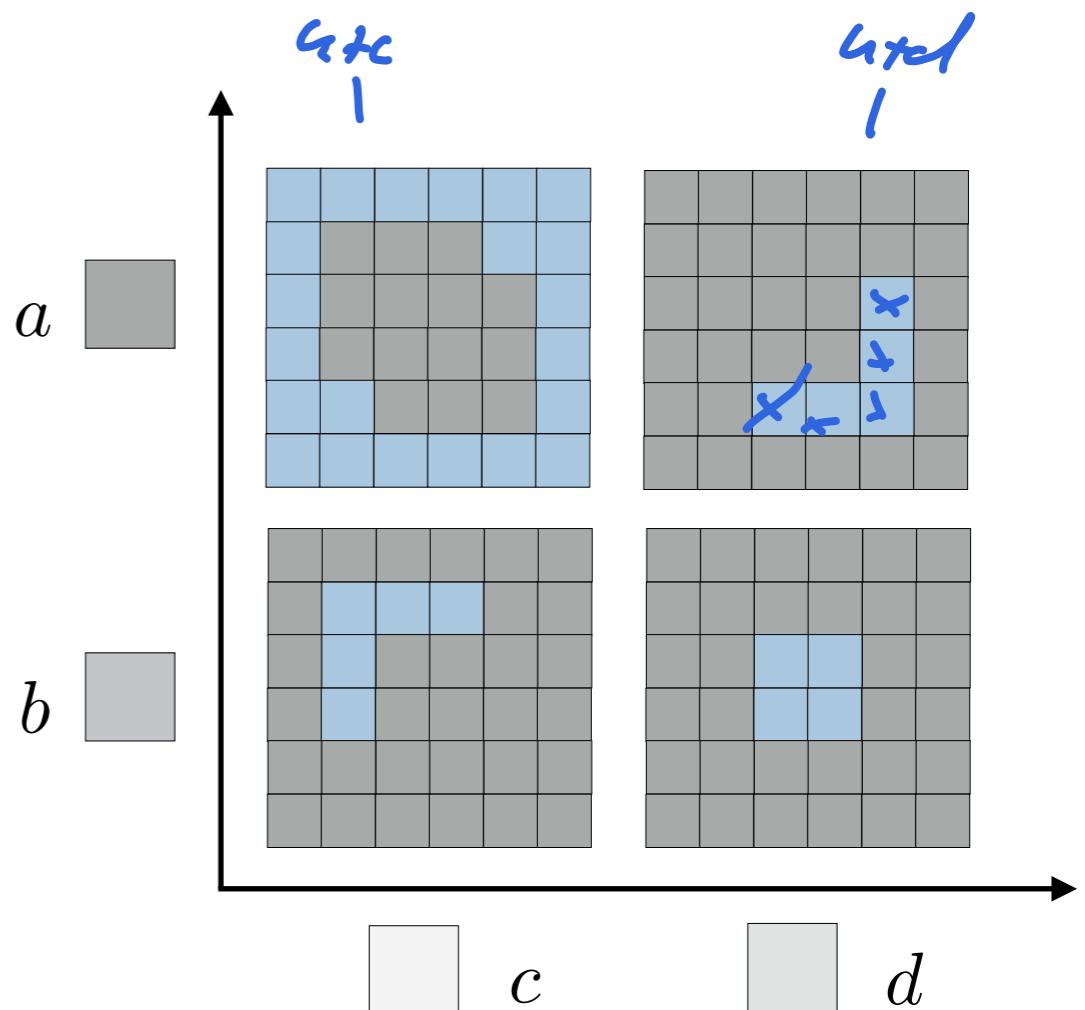


$$p(a,d) = \frac{5}{36}, \text{ analog für die anderen}$$

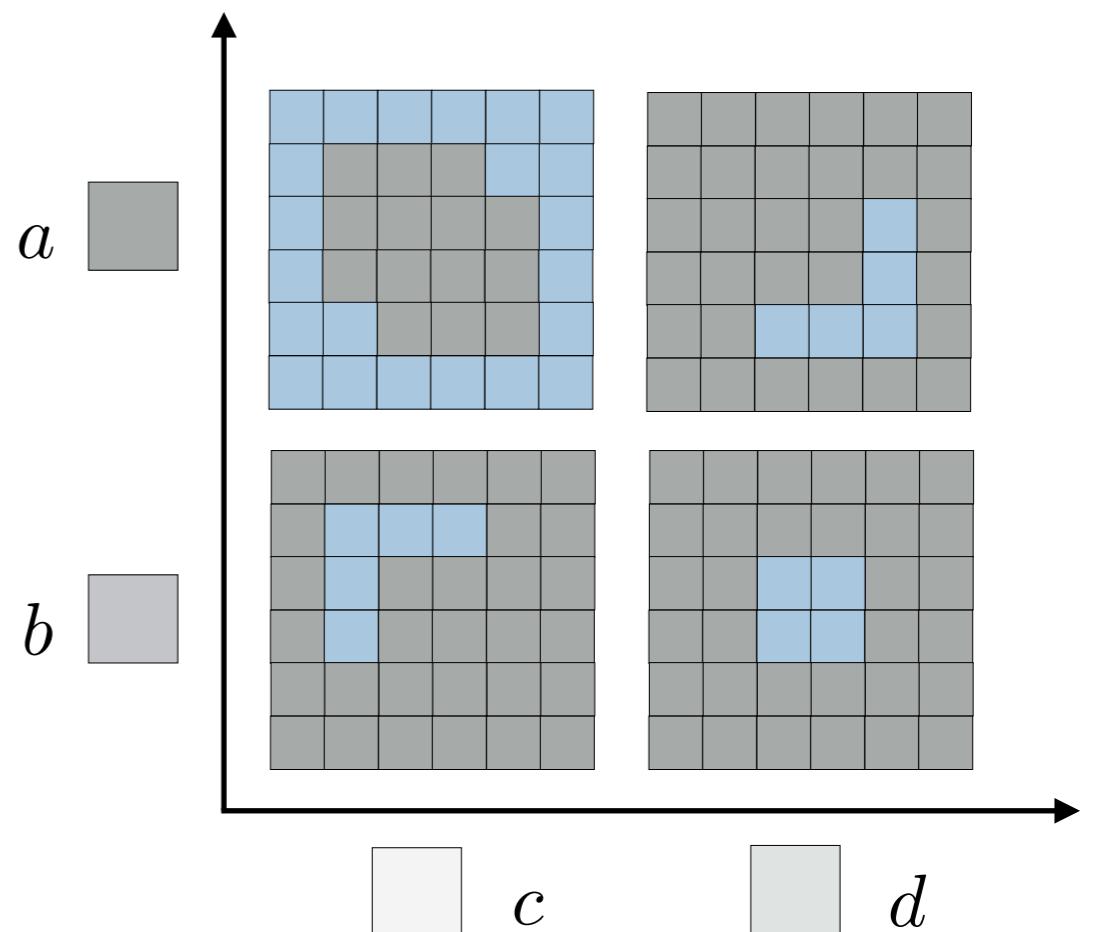
$$p(b) = \frac{2}{36}, \text{ analog für alle anderen}$$

$$I = p(b,c) \cdot \log \frac{p(b,c)}{p(b) \cdot p(c)} + \dots$$

$$= 0.865$$



Einschub: Mutual Information



Einschub: Self-Supervised Adversarial Training

Gegeben: Originalbild und korrespondierendes Adversarial Example

Aufgabe: Maximiere die Mutual Information

Methode:

Handcode:

Schleife über die Epochen

Schleife über die Mini-Batches (MB)

für $x \in MB$: generiere zu x ein $A\tilde{x}$ durch Fast Gradient Sign

$\Rightarrow M, I$, Mutual Information

Gradientenabstieg zur Minimierung des MI-Fehlermaßes

(Minimierung des Fehlers statt Maximierung der MI)

Self-supervised Adversarial Training

a) 2 Forward-Backward Schritte

- Generiere ein Adversarial Example

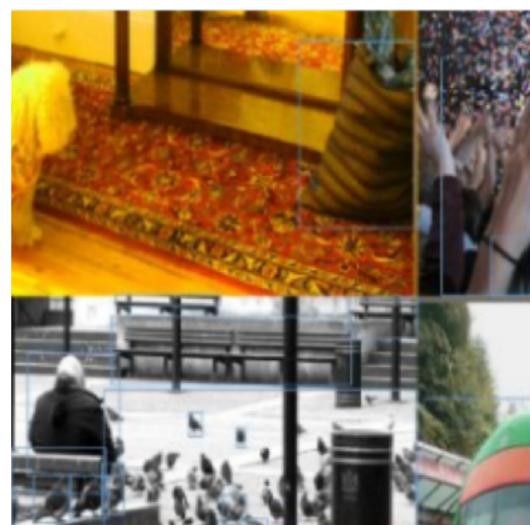
- Optimiere das Netz so, dass das richtige Label für das Adversarial Example erkannt wird

⇒ Augmentierung der Trainingsdaten mit
Adversarial Examples

Objektdetektion

YOLOv4

→ Zusammenhang der Objekte verändern
Mosaic Augmentation



© A. Bochkovskiy et al., 2020

↳ 4 verschiedene Kontexte
⇒ Erkennungsrohstr gegen wechselnden Kontext

Objektdetektion

YOLOv4

z.B. vom Trainingsdaten unsaubrig sind

Label Smoothing

bisher:

korrektes Label = 100% als Zielwert der Wahrscheinlichkeit der Vorhersage

neu:

korrektes Label = 90% als Zielwert der Wahrscheinlichkeit der Vorhersage

Objektdetektion

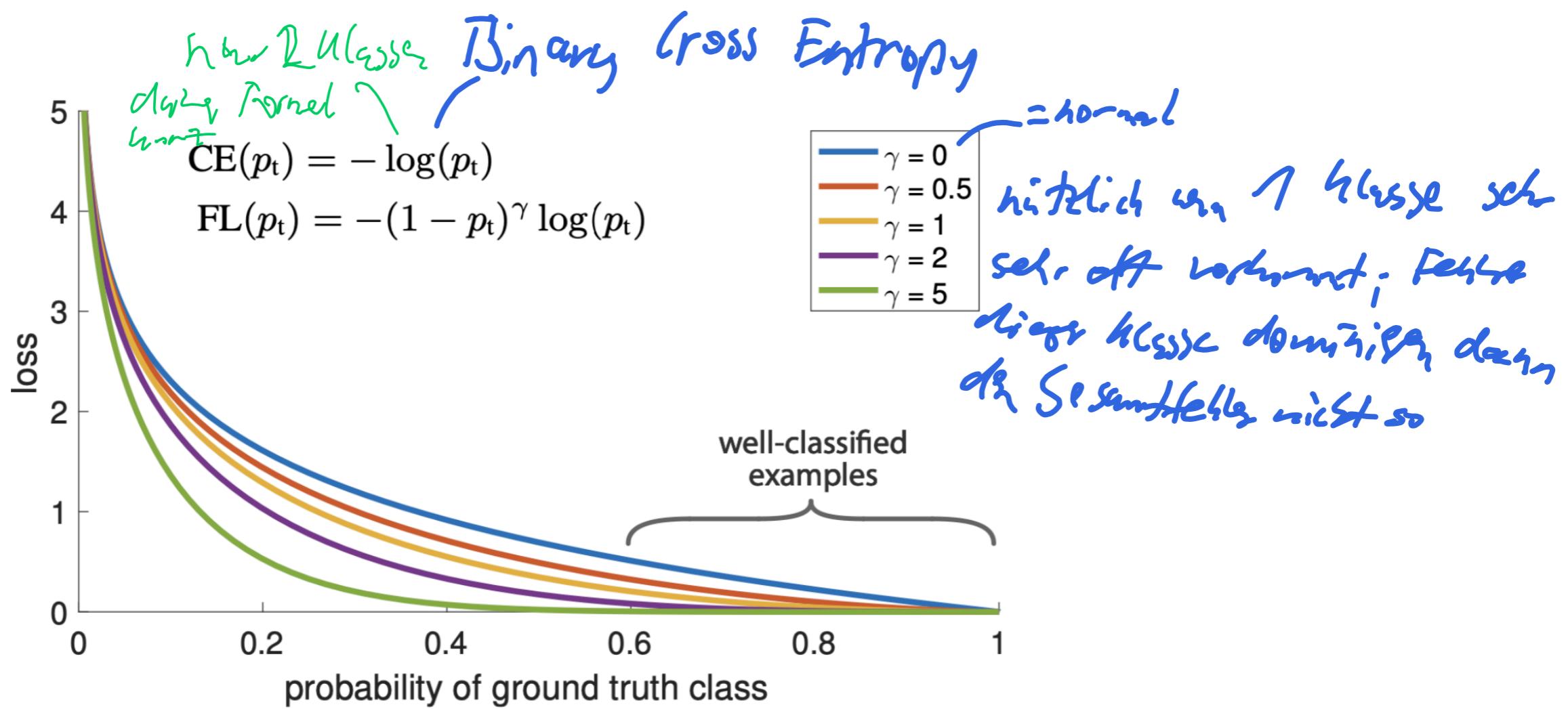
YOLOv4

$\Rightarrow \gamma$ groß: Loss wird gleich gehalten, und wenn die W ϵ für eine Klasse gar nicht so groß ist

Focal Loss

\Rightarrow Fehleranteil von kleineren Klassen kann sofort in den Hintergrund treten gegenüber sehr großen Klassen

Ungleichmäßige Verteilung bzgl. Hintergrund und Objekten

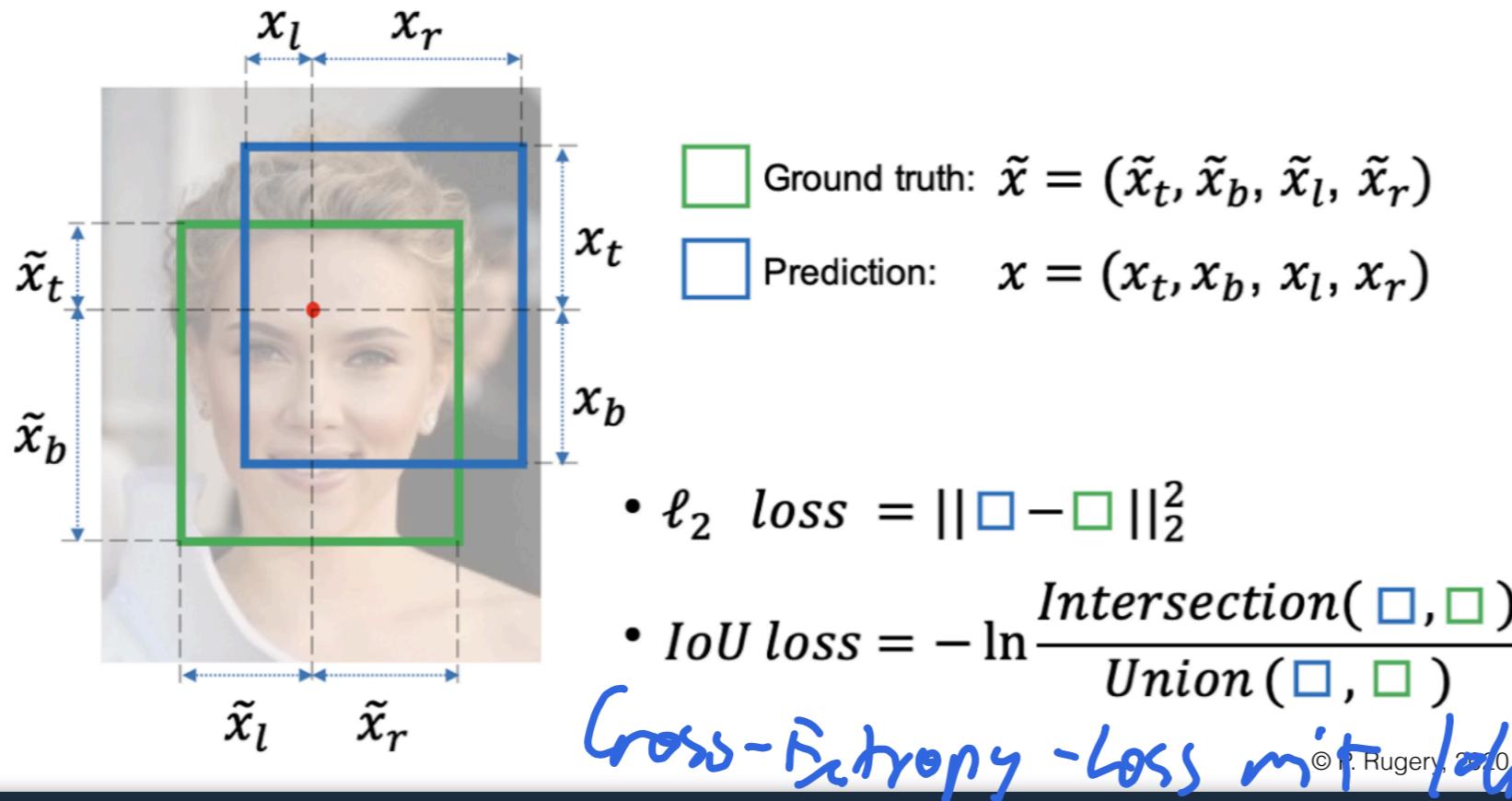


Objektdetektion

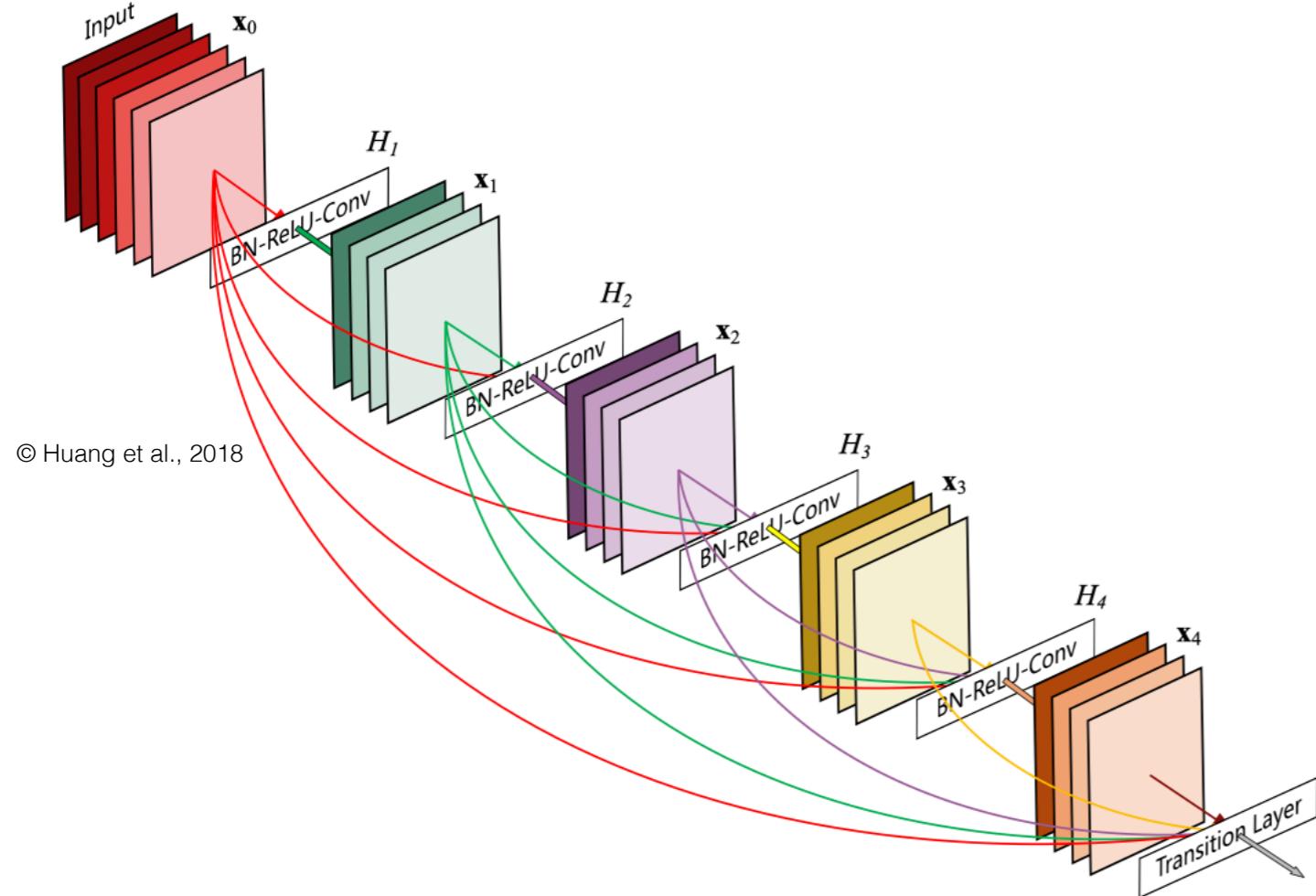
YOLOv4

Intersection over Union Loss

⇒ man für Performance wird auch hier
Optimierung benutzt

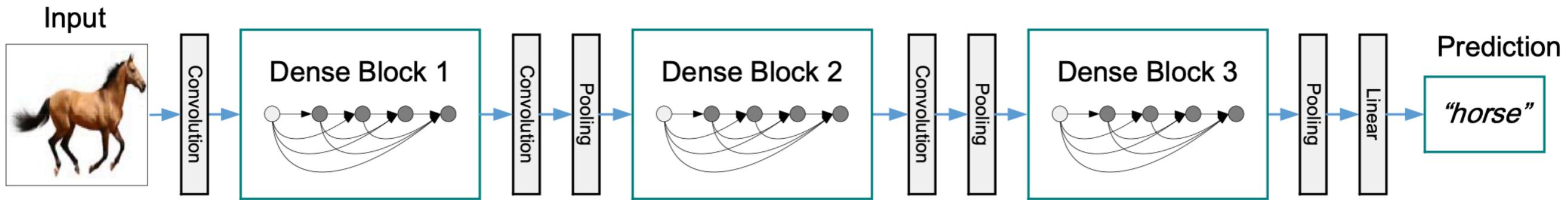


Einschub: DenseNet



Idee: Merkmalskarten der vorherigen Schichten werden in jedg
Später Schicht weiterverarbeitet

Einschub: DenseNet



© Huang et al., 2018

Dense Block:

Grundidee des Dense Nets auf einer
Tiefenlernstufe

Übergangsschicht:

- Reduktion der Ortsauflösung
- Batch-Normalization
- 1×1 - Faltungsschicht
- 2×2 - Average-Pooling

Objektdetektion

YOLOv4

Cross Stage Partial Connections

folgt der Idee von DenseNets

Unterschied:

Input wird doppelt verwendet: *Originalbild*
*Input wird am Ende mit der letzten Merkmalsbank
verhöhnt \Rightarrow es muss Input-NIL markiert werden*