Nummernschilderkennung mit Python

Anne-Sophie Bollmann, Susanne Klöcker, Pia von Kolken, Christian Peters

18. Februar 2021

Pipeline

Pipeline

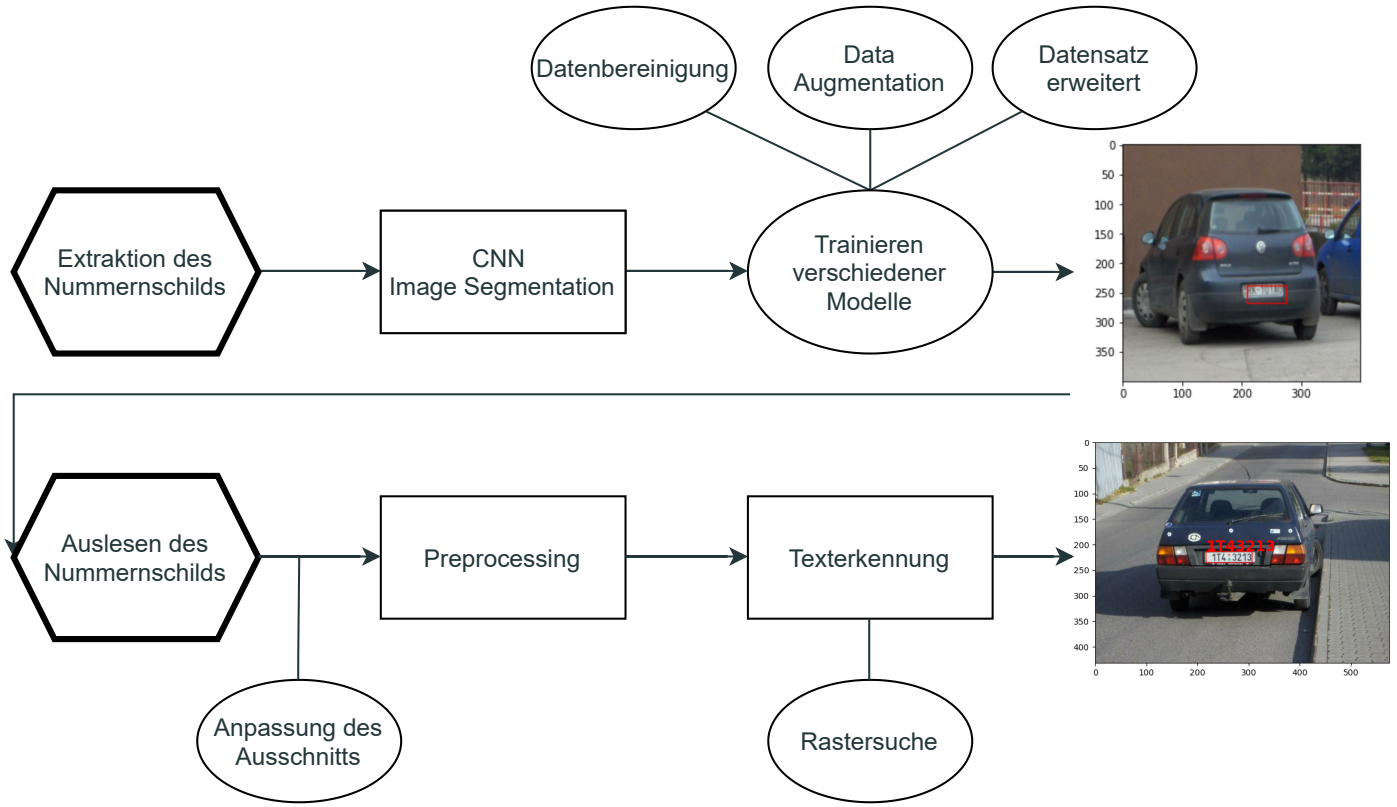
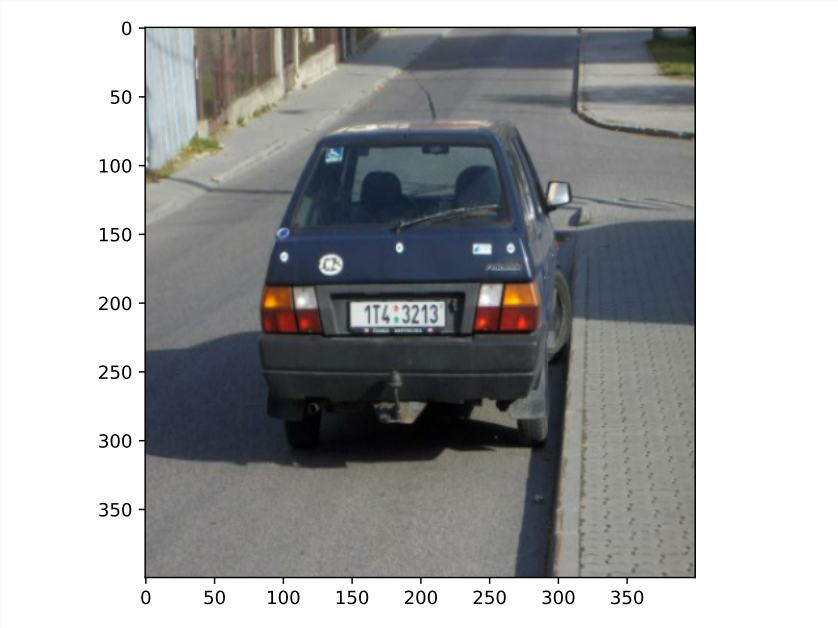
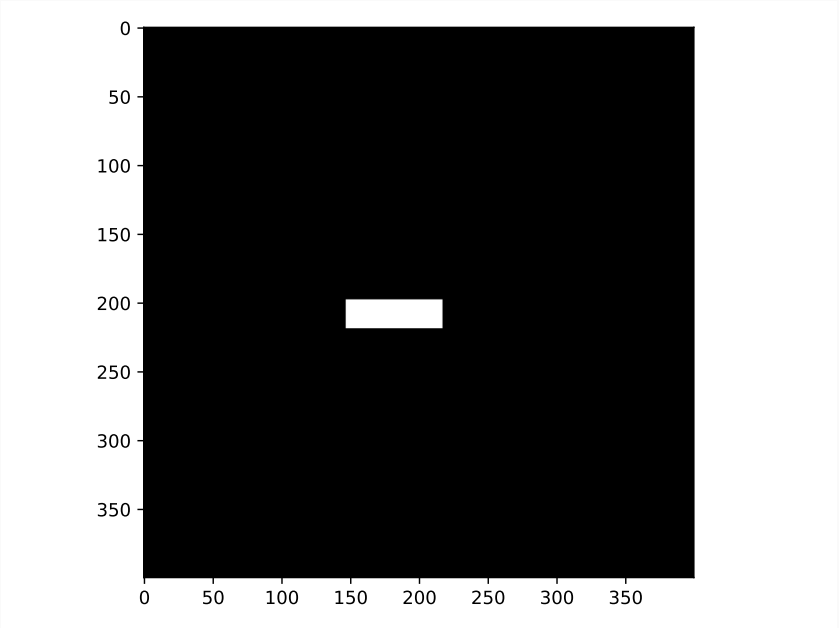

Datenbereinigung
Data Augmentation
Datensatz erweitert
Extraktion des Nummernschilds
CNN Image Segmentation
Trainieren verschiedener Modelle
Auslesen des Nummernschilds
Preprocessing
Texterkennung
Anpassung des Ausschnitts
Rastersuche

Image Segmentation

Trainingseingaben



(a) Eingabe



(b) Ziel

Modellarchitektur

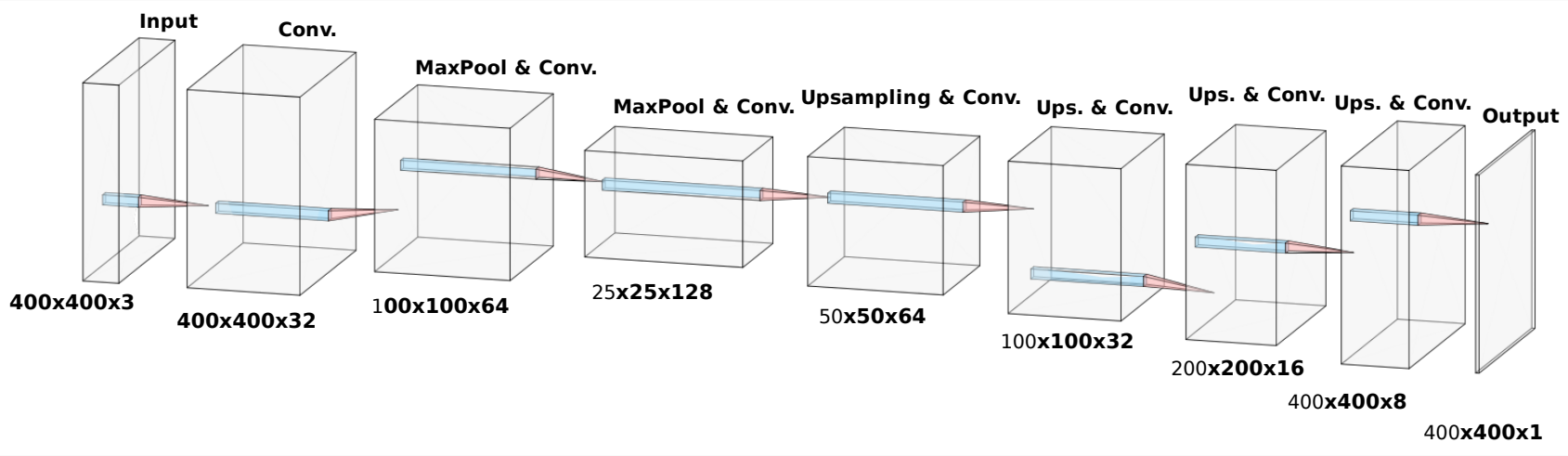
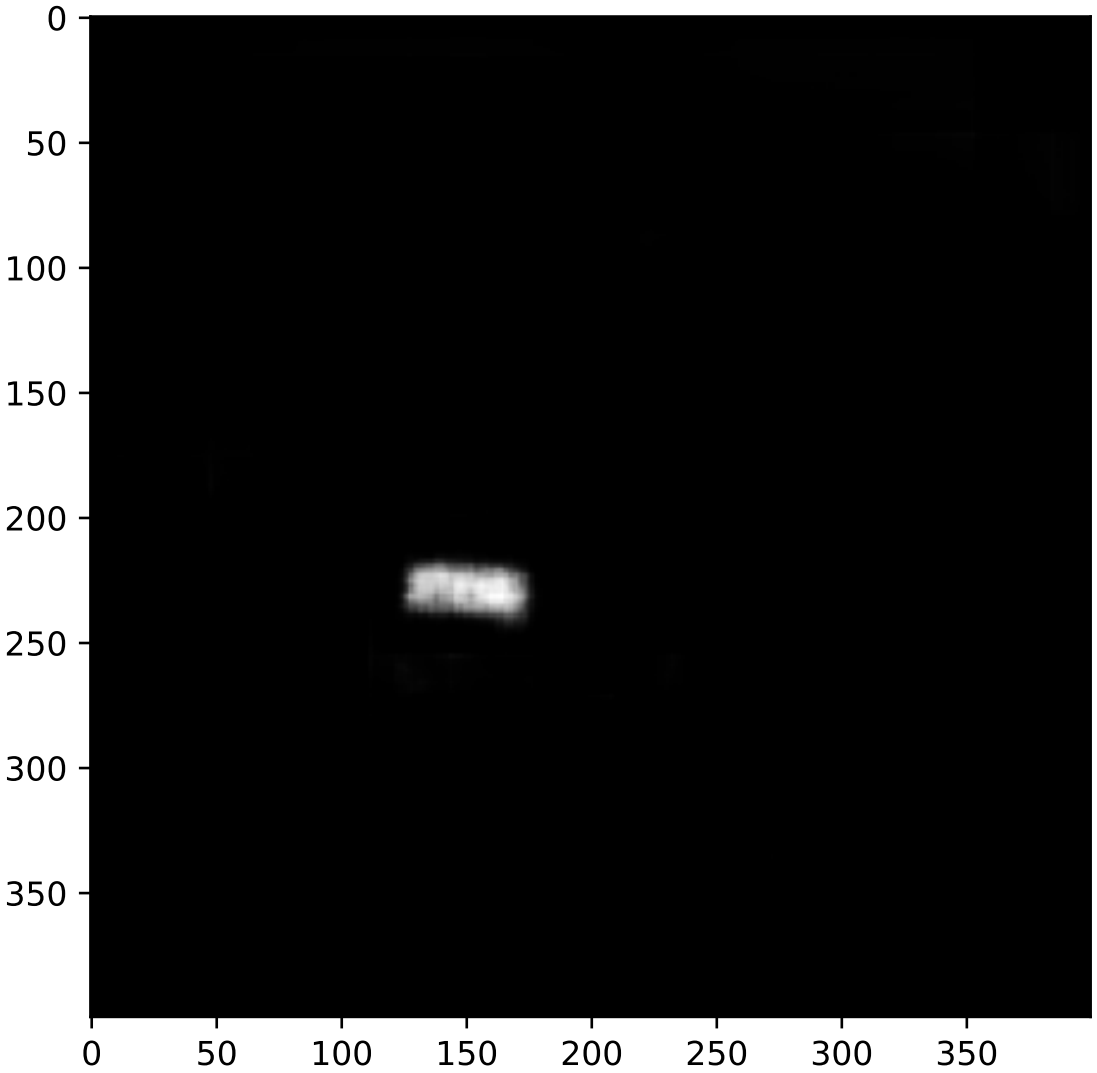
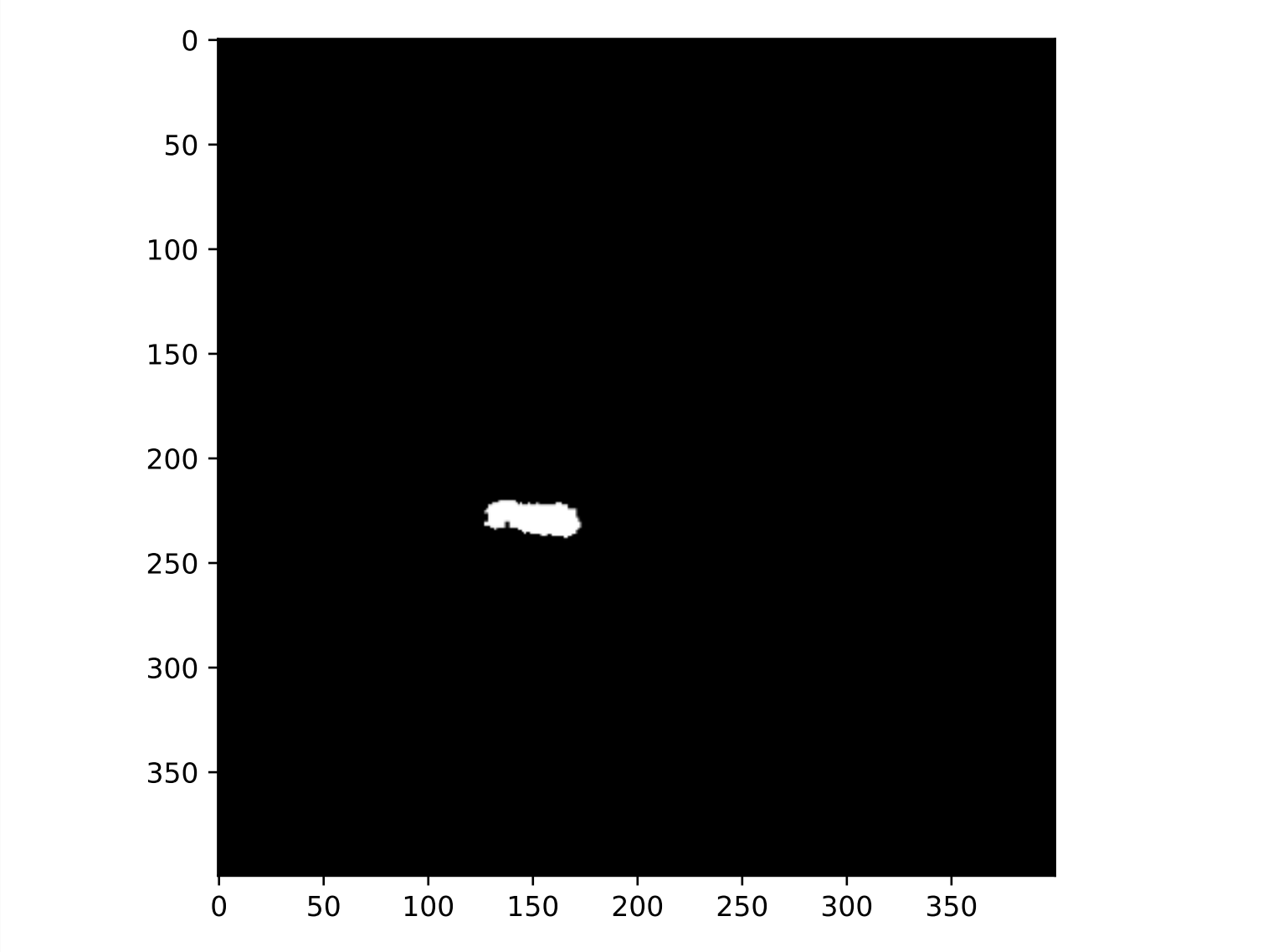


Abbildung 2: Das Ergebnis nach wochenlangem Ausprobieren.

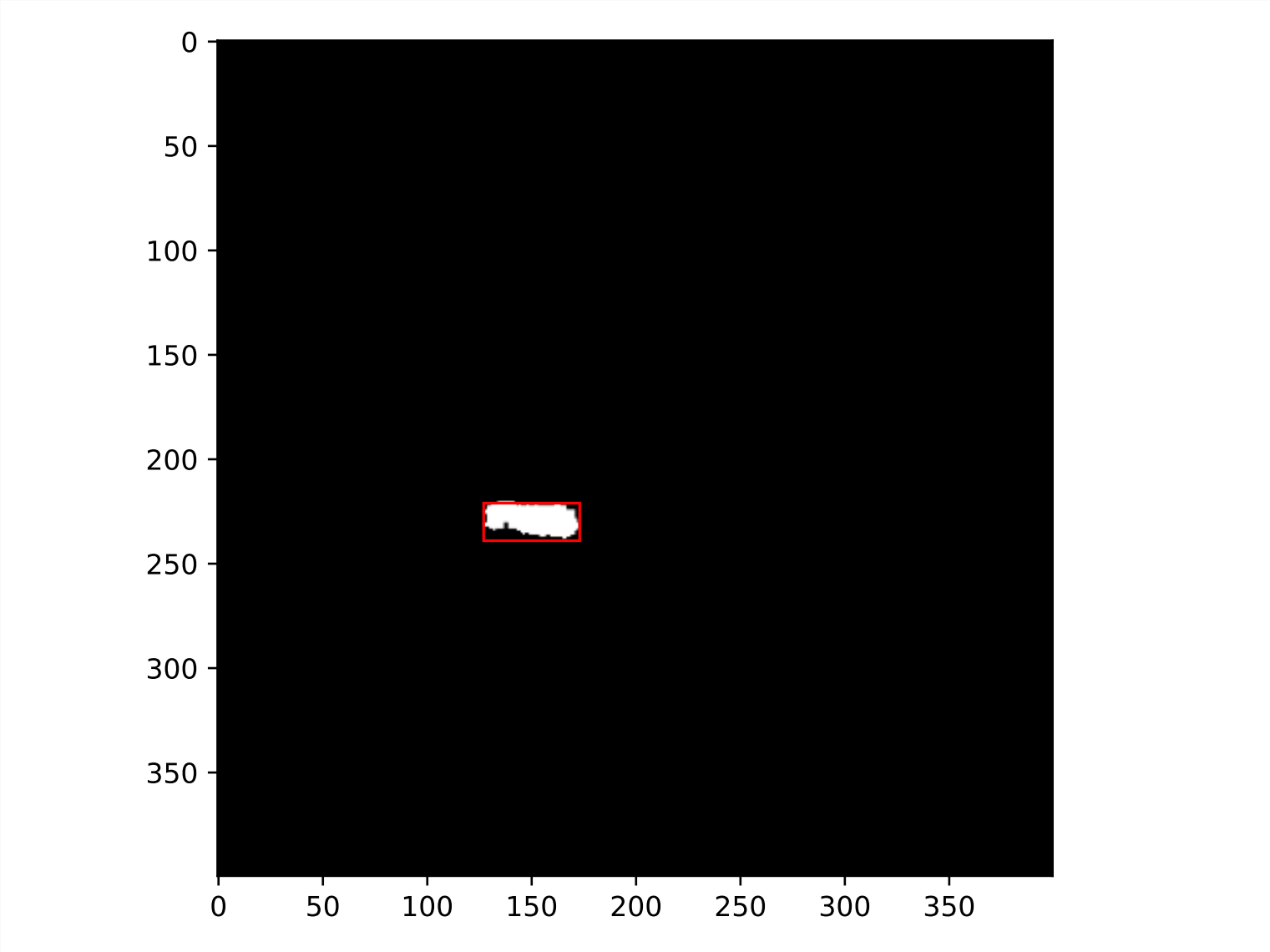
Modellvorhersage



Schwellenwert



Umschließendes Rechteck



Resultat

Details zum Training

Implementierung in Tensorflow, Training auf Google Colab GPU

- 534 Zusätzliche Trainingsbilder hinzugefügt
- Data Augmentation: 949 $\Rightarrow$ 22.776 Bilder
  - Horizontal Flip, Random Cropping, Random Contrast, Random Brightness
  - **Benötigt ¿14GB GPU Speicher!**
- Gradient Descent mit ADAM Optimizer
- Loss: Binary cross entropy

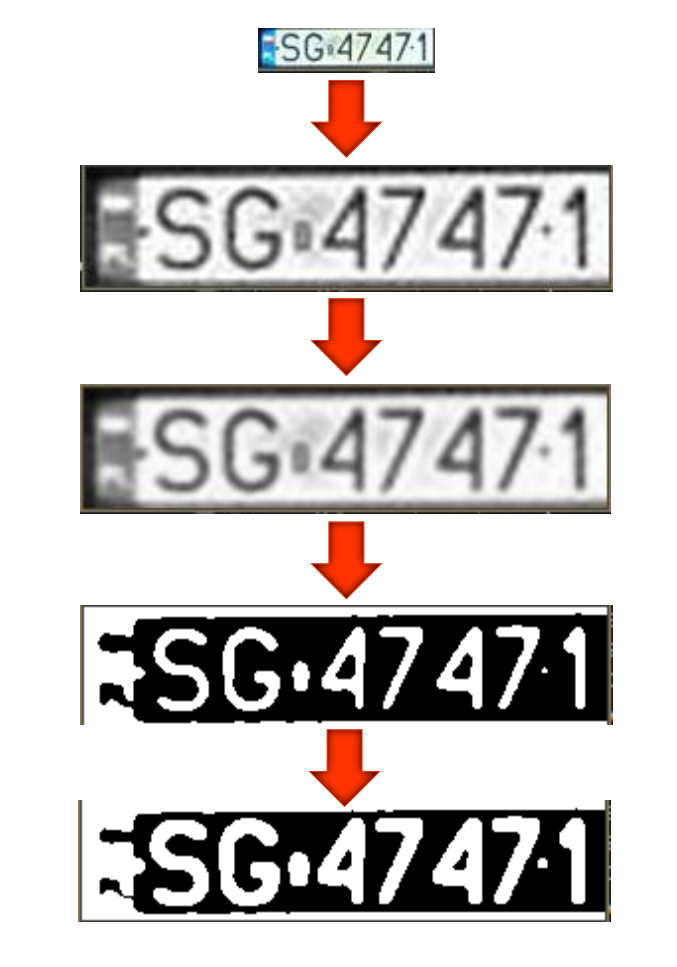
$$-\sum_{\text{Pixel}} y_{true} \log(y_{pred}) + (1 - y_{true}) \log(1 - y_{pred})$$

- Zur Validierung 20 Bilder aus EU/RO per Hand selektiert
- „Early Stopping“ nach 19 Epochen

Optical Character Recognition

Bearbeitungsschritte

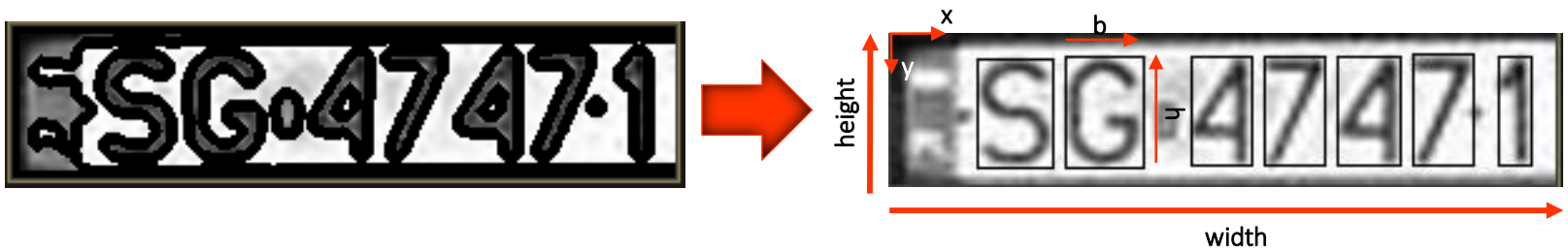
1. Vergrösserung und Graustufen
2. Blurring (Bildglättung)
   - Gaußsche Unschärfe und Median Unschärfe
3. Thresholding (Schwellenwertverfahren)
   - Otsu: wählt automatisch einen geeigneten Schwellenwert aus
   - Einfaches Verfahren mit Schwellenwerten $\{60, 80, 100, 120\}$
   - Binary-Inv.: Die Werte derart getauscht, dass schwarz zu weiß wird und umgekehrt $\rightarrow$ führt zur besseren Erkennung von Konturen
4. Dilation (Morphologische Transformation)



Aussortierung der Konturen

- *Genutzt werden die Werte x, y, w, h (Werte der Kontur) sowie width und height (Breite und Höhe des Bildes)*
- Es werden nur Konturen berücksichtigt, die folgende Bedingungen erfüllen:

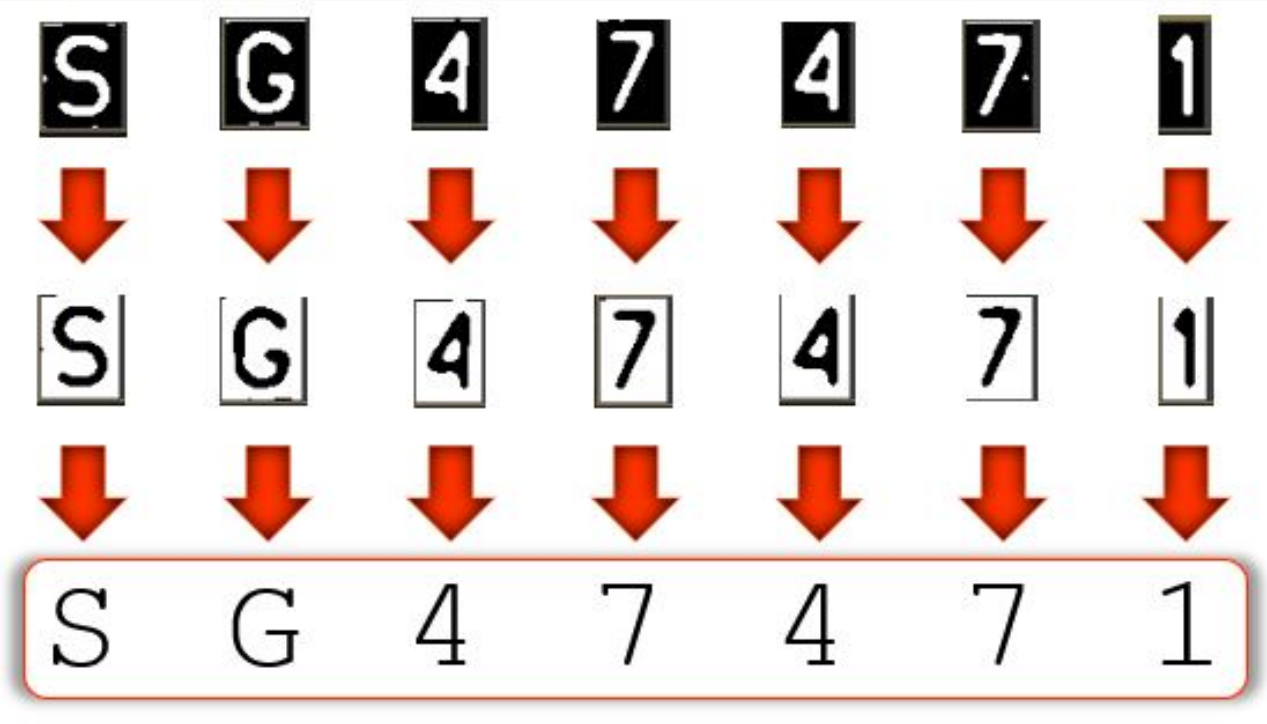
$$\frac{height}{h} > 3 \qquad \frac{h}{w} < 1.2 \qquad \frac{width}{w} > 50$$



Character auslesen

Einstellungen für das Auslesen mit Tesseract 5:

- Jeder Character wird einzeln ausgelesen
  $\rightarrow$ *PageSegmentationMode*(`--psm10`)*EngineMode*`--oem3`
- Zeichen-Whitelist (Großbuchstaben + Zahlen 0-9)
- Außerdem: Bild darf nicht zu nahe am Character ausgeschnitten werden



Boundingboxes verschieben und Levenshtein-Distanz

Falls keine Character in der gefundenen Bounding-Box erkannt werden, verschiebe die Bounding-Box anhand unterschiedlicher Methoden: *hoch*, *runter*, *rechts*, *links*, *oben-rechts*, *unten-rechts*, *unten-links*, *oben-links*

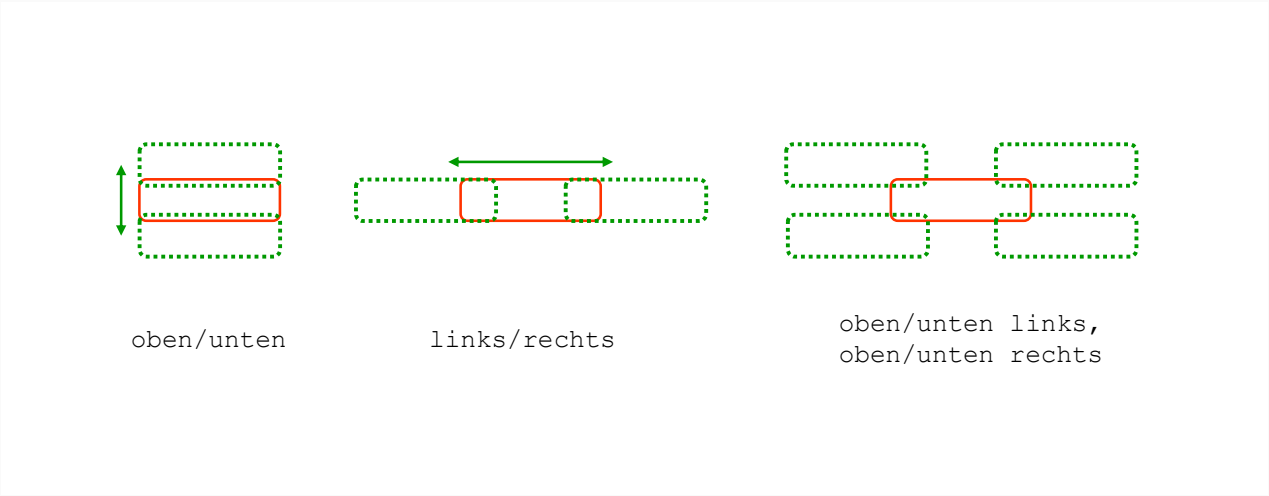


Tabelle 1: Levenshtein-Distanz

ohne Verschiebung	mit Verschiebung	mit Verschiebung & Thresholding-Variierung
4.05	3.55	3.2

Learnings

Learnings

Dies und das...

Evaluator Results

Evaluator Results

Dies und das...

Literatur i