

Pixelweise Klassifikation von Straße

Martin, Marvin, Sebastian, Vitali | 22. Juli 2015



Contents

1 Einleitung

2 Paper

3 Unser Ansatz

4 Ergebnisse

5 Ausblick

Daten



Überlagerte Labels



■ Caffe

- + Großes Framework / viele Funktionen
- + Viele Beispiele
- Schlecht Dokumentiert
- Sehr viel Code / mäßige Code-Lesbarkeit

■ nolearn, Lasagne, Theano

- + Exzellente Dokumentation
- + Entwickler Antworten schnell auf Anfragen
- + Gute Code-Qualität

⇒ Street Segmentation Toolkit (SST)

- Modelle trainieren,
- Modelle evaluieren / testen,
- Videos erstellen

■ Caffe

- + Großes Framework / viele Funktionen
- + Viele Beispiele
- Schlecht Dokumentiert
- Sehr viel Code / mäßige Code-Lesbarkeit

■ nolearn, Lasagne, Theano

- + Exzellente Dokumentation
- + Entwickler Antworten schnell auf Anfragen
- + Gute Code-Qualität

⇒ Street Segmentation Toolkit (SST)

- Modelle trainieren,
- Modelle evaluieren / testen,
- Videos erstellen

■ Caffe

- + Großes Framework / viele Funktionen
- + Viele Beispiele
- Schlecht Dokumentiert
- Sehr viel Code / mäßige Code-Lesbarkeit

■ nolearn, Lasagne, Theano

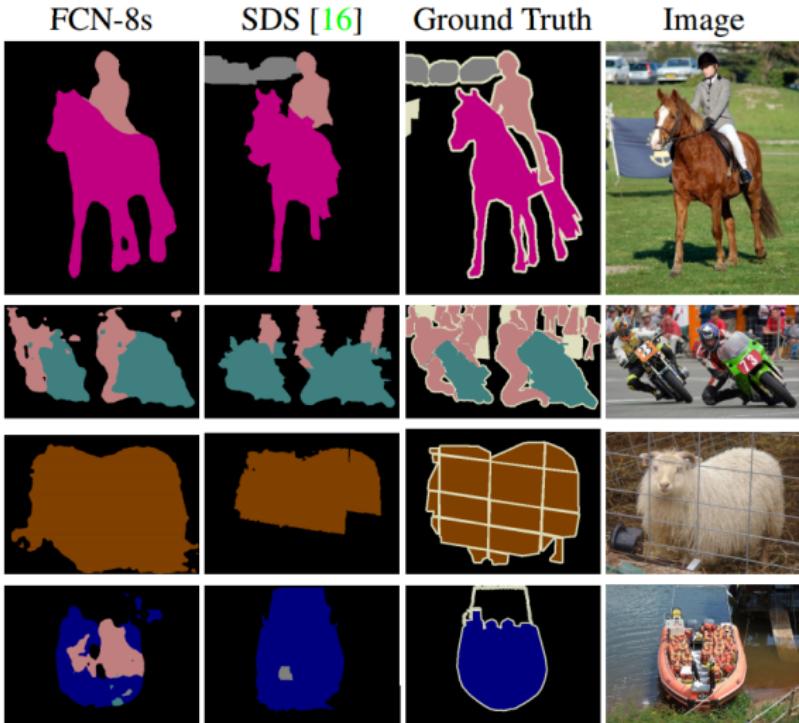
- + Exzellente Dokumentation
- + Entwickler Antworten schnell auf Anfragen
- + Gute Code-Qualität

⇒ Street Segmentation Toolkit (SST)

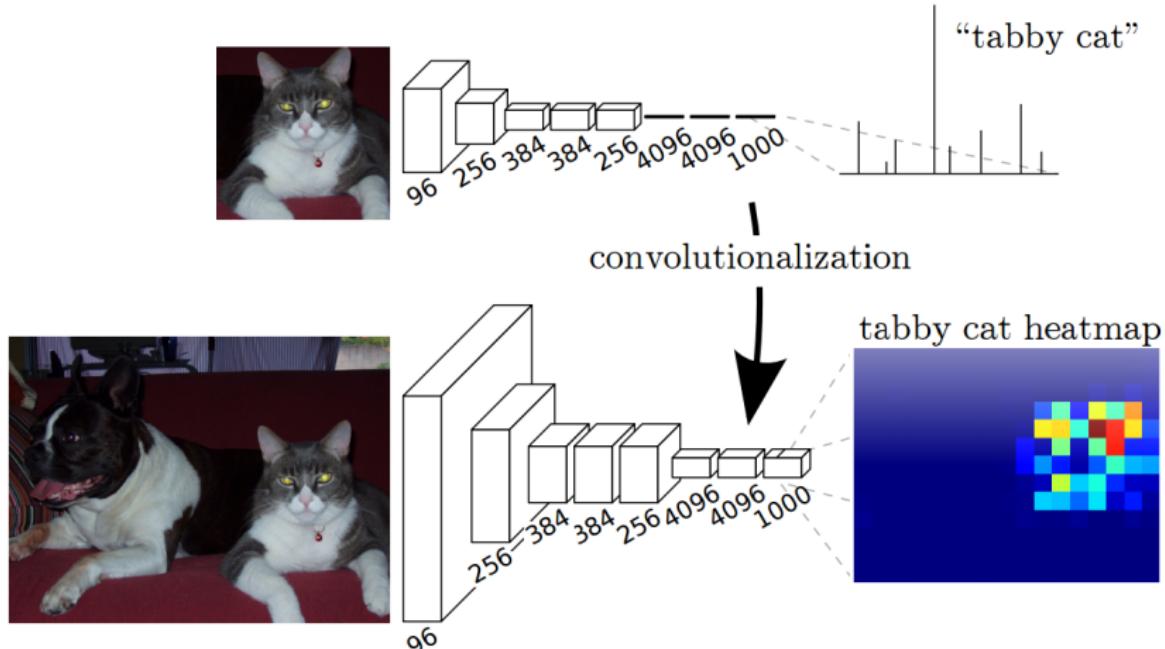
- Modelle trainieren,
- Modelle evaluieren / testen,
- Videos erstellen

- Fully Convolutional Networks for Semantic Segmentation:
Jonathan Long, Evan Shelhamer, Trevor Darrell
- pixelwise segmentation of multiple classes

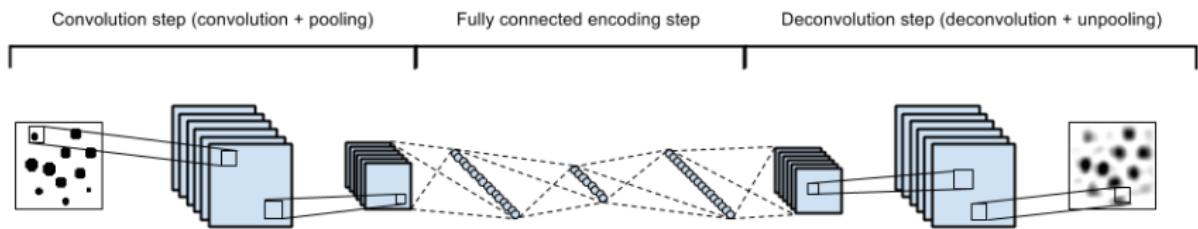
Paper - Results



Paper - Heatmap



Paper - Deconvolution



Klassifikation - Sliding Window

- Klassifizierte mittigen Pixel der Eingabe

Input: 51×51



Output: 1×1



{0,1}

- Laufe über das Bild (Sliding Window)
- Überspringe Pixel zur Beschleunigung

Klassifikation - Sliding Window

- Klassifizierte mittigen Pixel der Eingabe
- Laufe über das Bild (Sliding Window)
- Überspringe Pixel zur Beschleunigung

Input: 51×51

Output: 1×1



{0,1}



Klassifikation - Sliding Window

- Klassifizierte mittigen Pixel der Eingabe
- Laufe über das Bild (Sliding Window)
- Überspringe Pixel zur Beschleunigung

Input: 51×51

Output: 1×1



{0,1}



Regression - Fully - Patch Evaluation

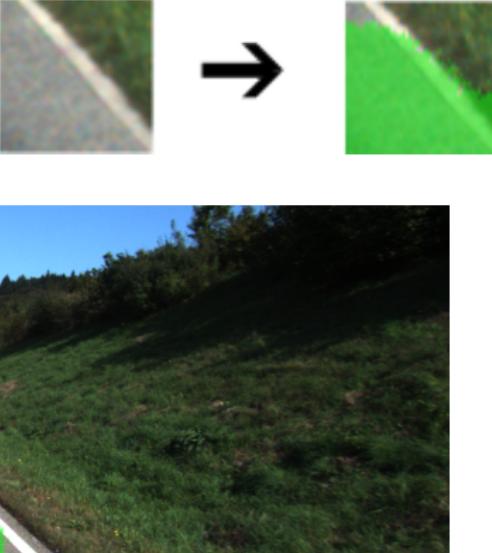
- Nutze Regression und Mean Squared Error Zielfunktion
 - Erhalte Information pro Pixel
 - Füge Bild zusammen
- Input: 51 x 51 Output: 51 x 51
- 

Regression - Fully - Patch Evaluation

- Nutze Regression und Mean Squared Error Zielfunktion
 - Erhalte Information pro Pixel
 - Füge Bild zusammen
- Input: 51 x 51 Output: 51 x 51
- 



Regression - Fully - Patch Evaluation

- Nutze Regression und Mean Squared Error Zielfunktion
 - Erhalte Information pro Pixel
 - Füge Bild zusammen
- Input: 51 x 51 Output: 51 x 51
- 



Aufbau unserer Neuronale Netze

Ergebnisse-Convolutional Layer



Ergebnisse-Convolutional Layer



Vergleich



Vergleich

| Netztyp | TN | FP | FN | TP | ACC |
|------------------|-------|-------|-------|-------|--------|
| Conv - Stride 10 | 0.977 | 0.022 | 0.197 | 0.802 | 0.9471 |
| Conv - Stride 37 | 0.973 | 0.026 | 0.176 | 0.823 | 0.9475 |
| Conv - Stride 51 | 0.969 | 0.030 | 0.165 | 0.834 | 0.946 |
| SW - Stride 10 | 0.981 | 0.018 | 0.241 | 0.758 | 0.942 |
| SW - Stride 37 | 0.959 | 0.040 | 0.212 | 0.787 | 0.929 |
| SW - Stride 51 | 0.982 | 0.017 | 0.451 | 0.548 | 0.906 |

Tabelle: Ergebnisse der verschiedene neuronalen Netze auf Testset (58 Bilder, 6,7 Mil. Pixel)

Laufzeiten

| Netztyp/Stride | 10 | 37 | 51 |
|----------------|------|------|------|
| Conv | 1.99 | 0.29 | 0.18 |
| SW | 1.83 | 0.2 | 0.11 |

Tabelle: Ausführungszeit pro Bild (621x188 Pixel) in Sekunden.

Video

- Zu wenig GPU-RAM für größere Patches
- Neuronales Netz verbessern
- Effizienteres Zusammensetzen der Patches
- Mehr Daten (Lense Flare, Beleuchtung, Fahrbahnmarkierungen)

- Paper - Results and Heatmap by Jonathan Long, Evan Shelhamer, Trevor Darrell
- Paper - Deconvolution by Mike Swarbrick Jones

Danke für die Aufmerksamkeit!

