

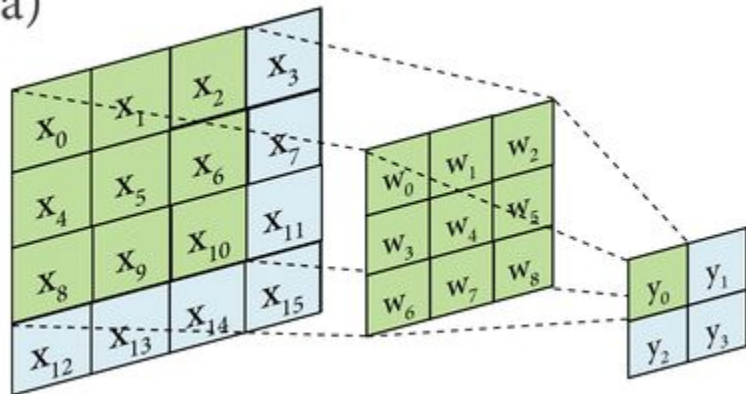
KNSI Golem Bootcamp Level-up

Spotkanie 1 – Computer Vision

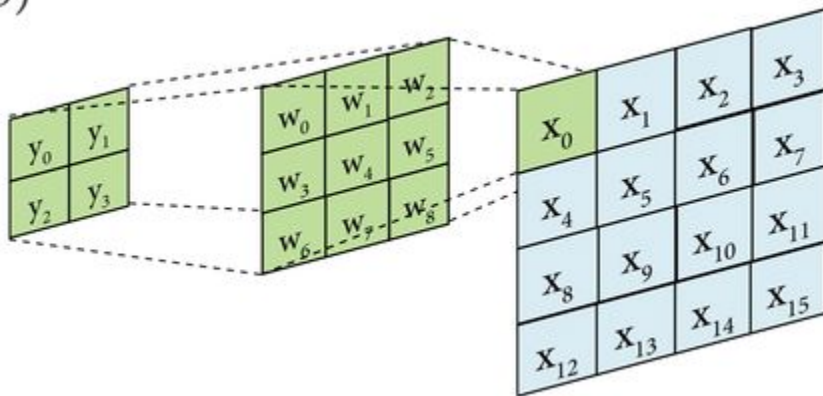
Agenda

- Konwolucja i konwolucyjne sieci (CNN)
- Poważniejsi gracze
- Transfer Learning (offtopic, ale nie do końca xD)
- Zadania typowe dla wizji komputerowej

a)



b)

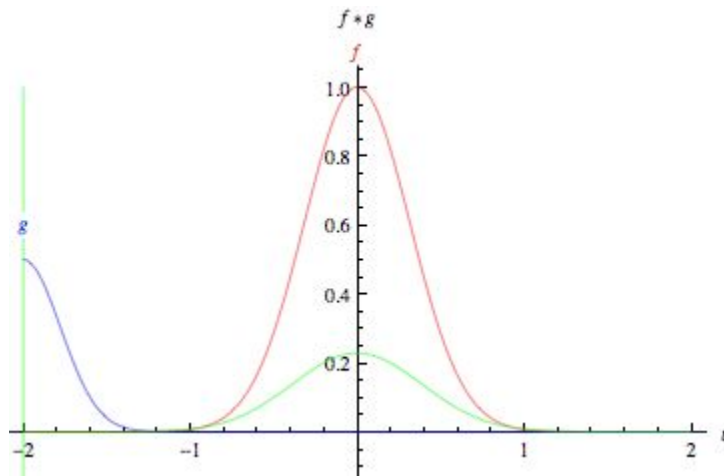


Konwolucja

Konwolucja

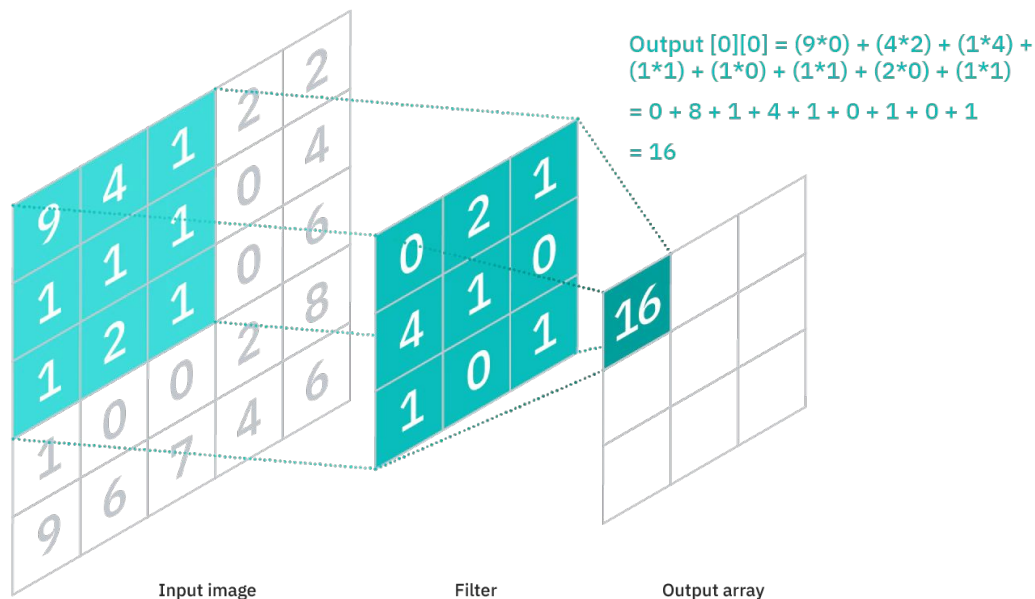
Idea wywodzi się z matematycznej definicji konwolucji

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau.$$



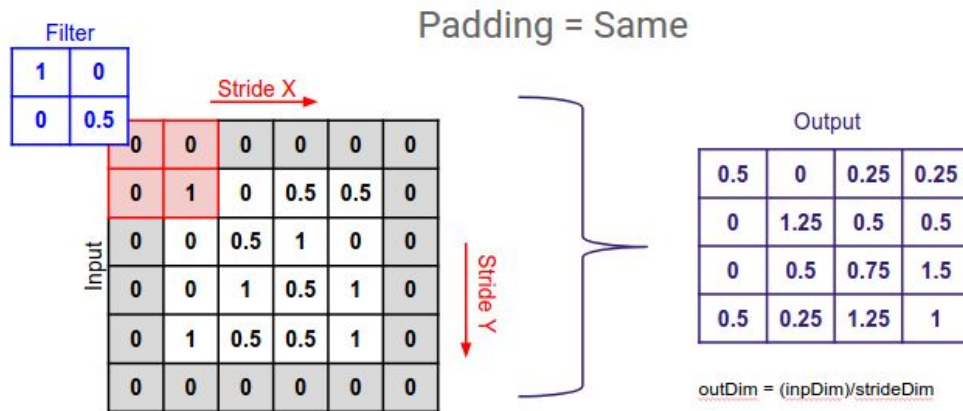
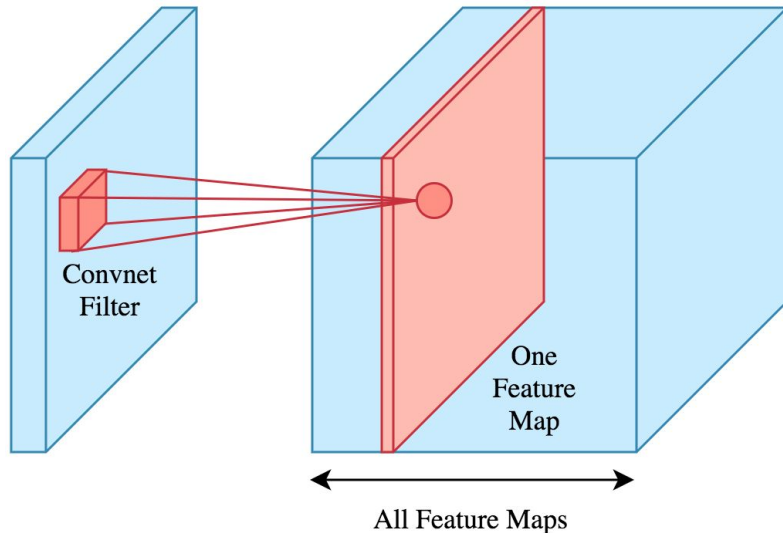
Konwolucja w MLu

“Filtr” to tensor $M \times N \times C$, który “przesuwamy” po wejściu i mnożymy wartości element po elemencie produkując skalar



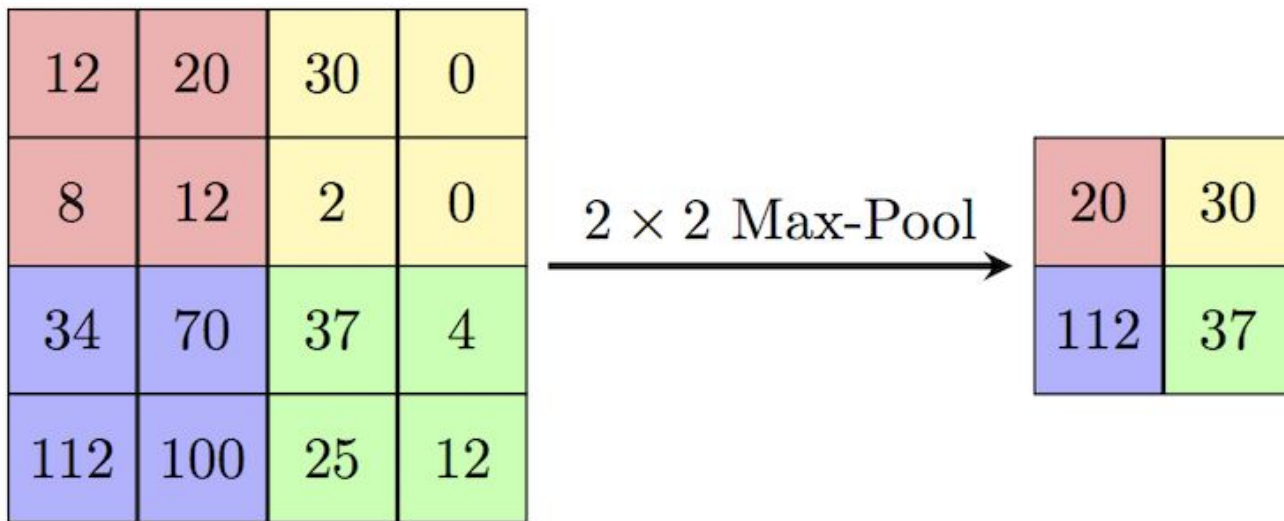
Warstwa konwolucyjna

- Składa się z 1+ filtrów, zazwyczaj wszystkie mają te same wymiary
- Definiujemy tzw. padding
- Wyjścia filtrów “stackujemy”, na wyjściu mamy tyle kanałów ile filtrów
- Możemy zdefiniować jakieś poboczne cechy warstwy, ale to są tzw. pierdoły



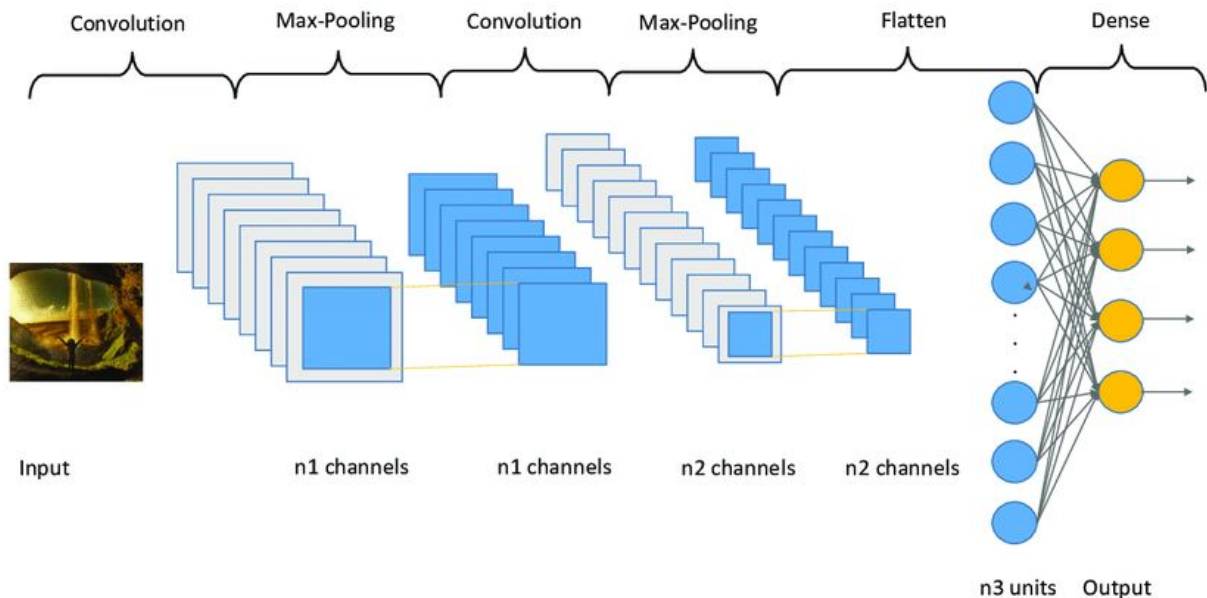
Pooling Layer

- Składa się z jednego filtra
- Służy do zmniejszania wymiarowości danych w warstwach sieci
- Ogranicza ilość parametrów sieci oraz w teorii ogranicza overfitting
- Różne odmiany: average, max, min



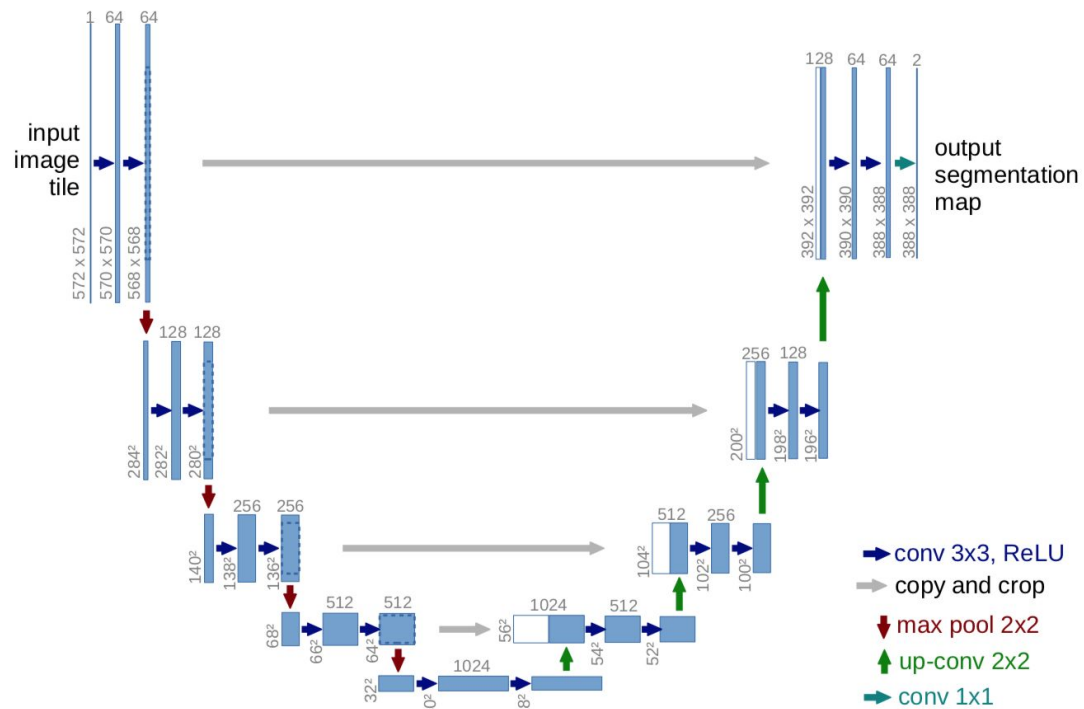
Sieć konwolucyjna – Convolutional Neural Network (CNN)

- Składa się z kilku warstw konwolucyjnych
- Do ostatniej warstwy dołączamy “głowę”, czyli zwykłą sieć neuronową





Przykład w kodzie + zadanie



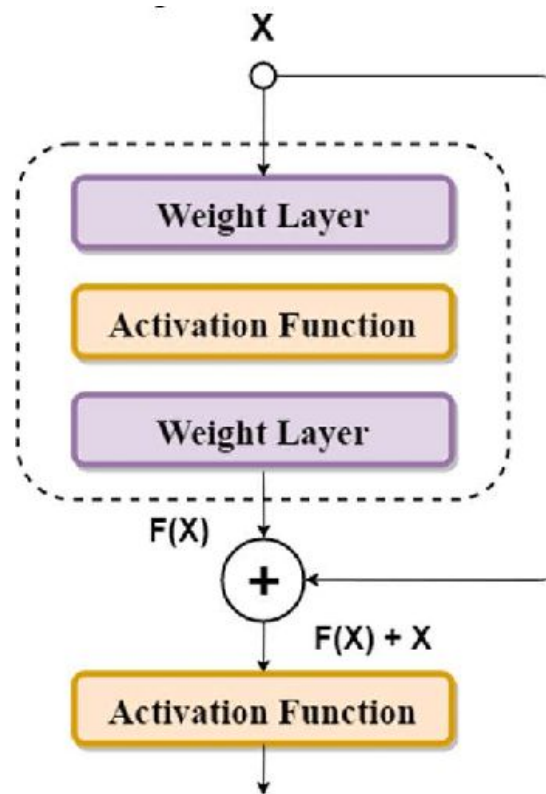
Poważne architektury do klasyfikacji

Background

- Wiele konkursów, m. in. ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
- Zazwyczaj odbywają się co roku
- Ogromne zbiory danych
- Mnóstwo klas (ImageNet ma ich aż 1000)
- Zwycięskie architektury to efekt bardzo dynamicznej ewolucji CNNów i ich pochodnych

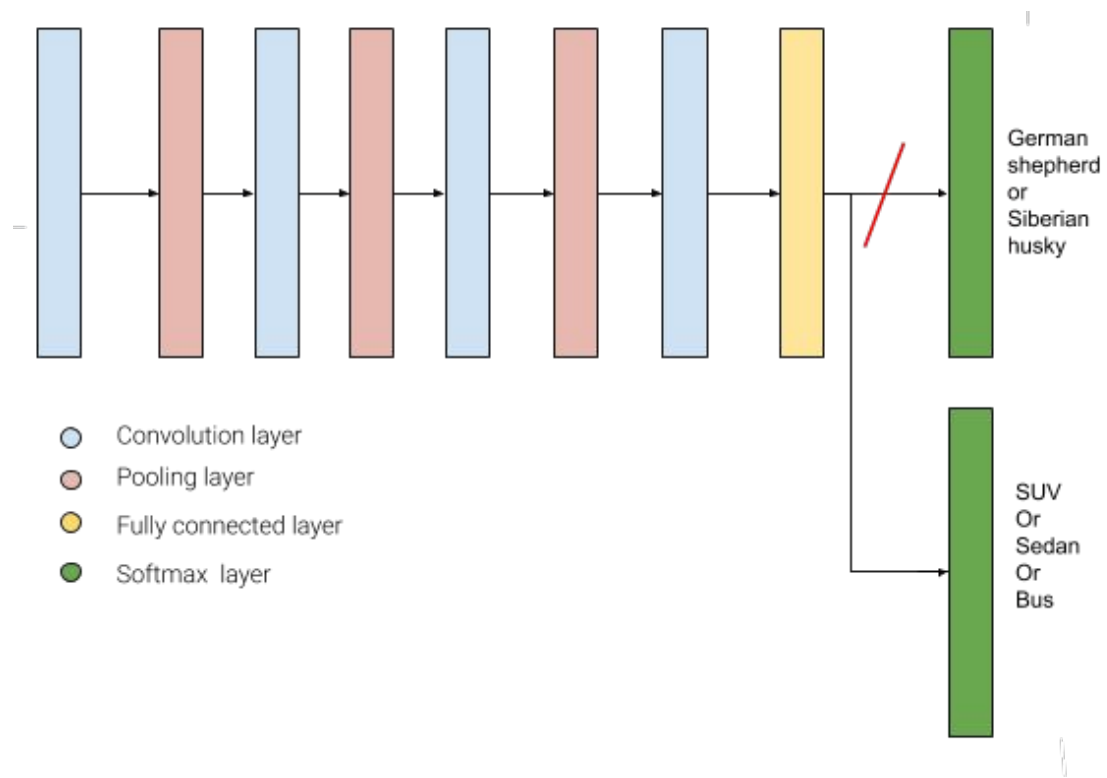
ResNet (zwycięzca ILSVRC 2015)

- Residual network
- Nowy typ warstwy, Skipping Layer
- Zwycięska architektura miała 50 warstw
- W następnych latach powstawały ResNet101, ResNet152



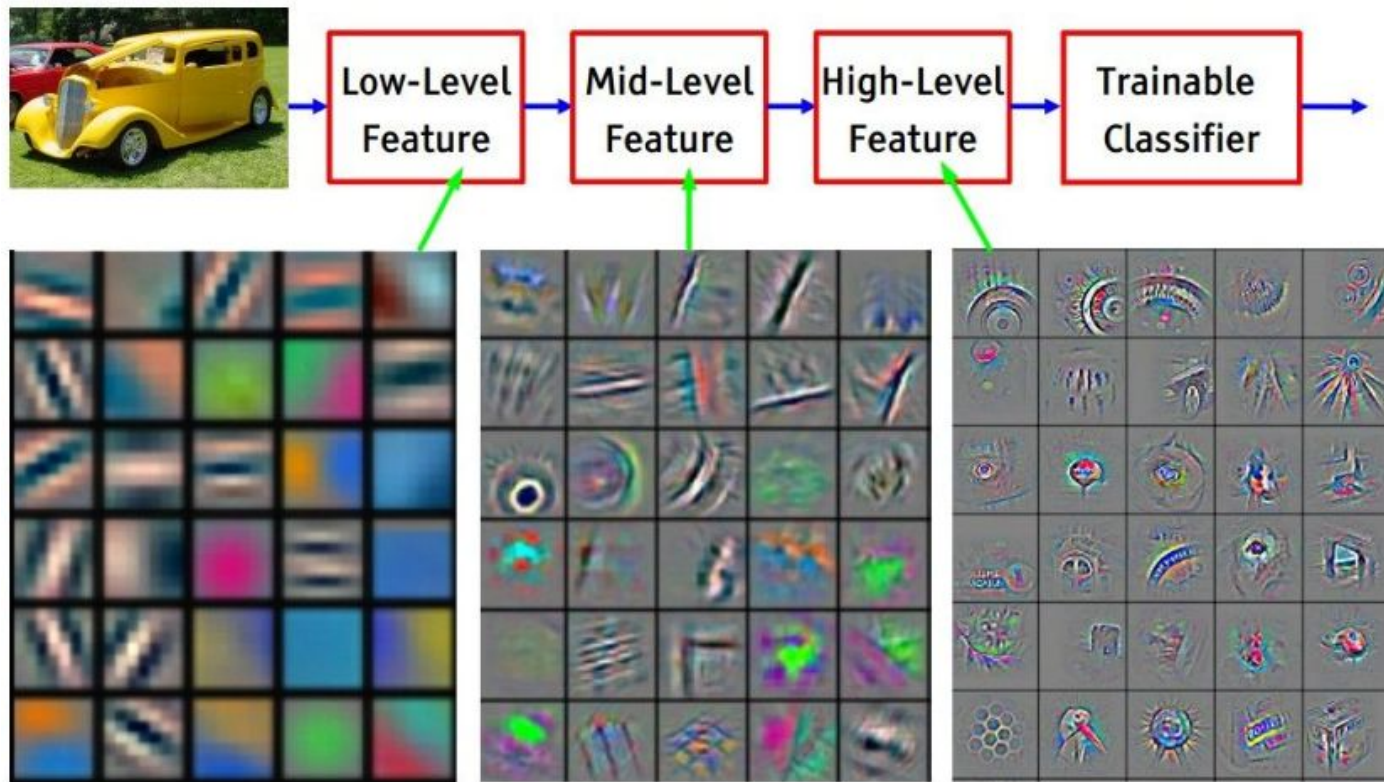


Przykład w kodzie + zadanie



Transfer Learning

Warstwy konwolucyjne jako “feature extractors”

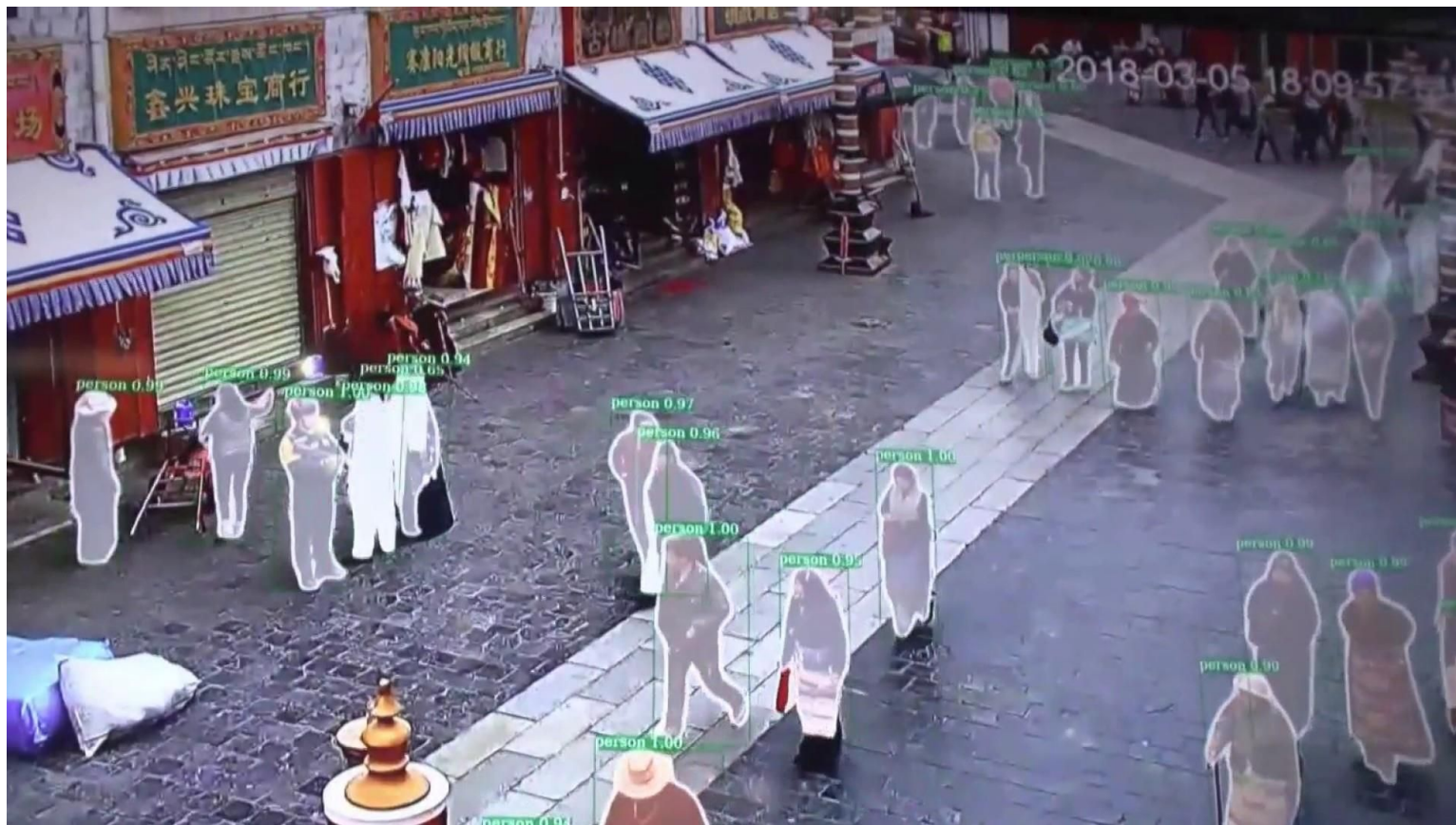


Idea

1. Mamy dwa bardzo zbliżone problemy
2. Do jednego z nich bierzemy **bardzo** duży model i **bardzo** duży dataset
3. Trenujemy go najlepiej jak się da
4. Wszystkie warstwy to ekstraktory cech
5. “Zamrażamy” tyle ile potrzebujemy i podmieniamy ostatnie warstwy
6. Trenując na nowym datasecie aktualizujemy tylko wagi nowych warstw



Przykład w kodzie + zadanie

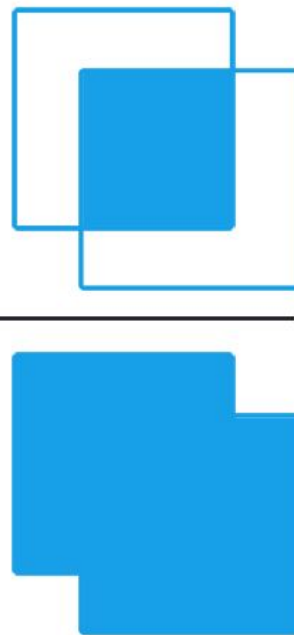


Co poza klasyfikacją?

Detekcja obiektów na obrazie

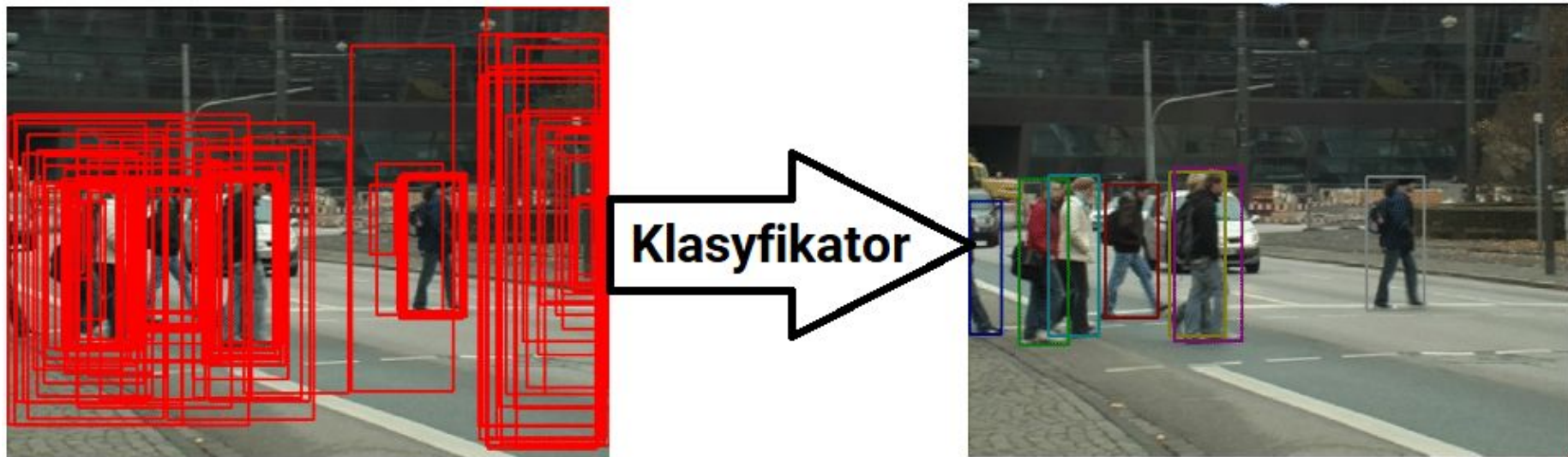
- Zazwyczaj połączone z klasyfikacją
- Polega na wyznaczeniu tzw. bounding box
- Funkcja straty: Intersection over Union
- Główne architektury:
 - Faster R-CNN
 - YOLO
- Trudne zadanie, wymaga bardzo dużych zasobów do trenowania from scratch

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$



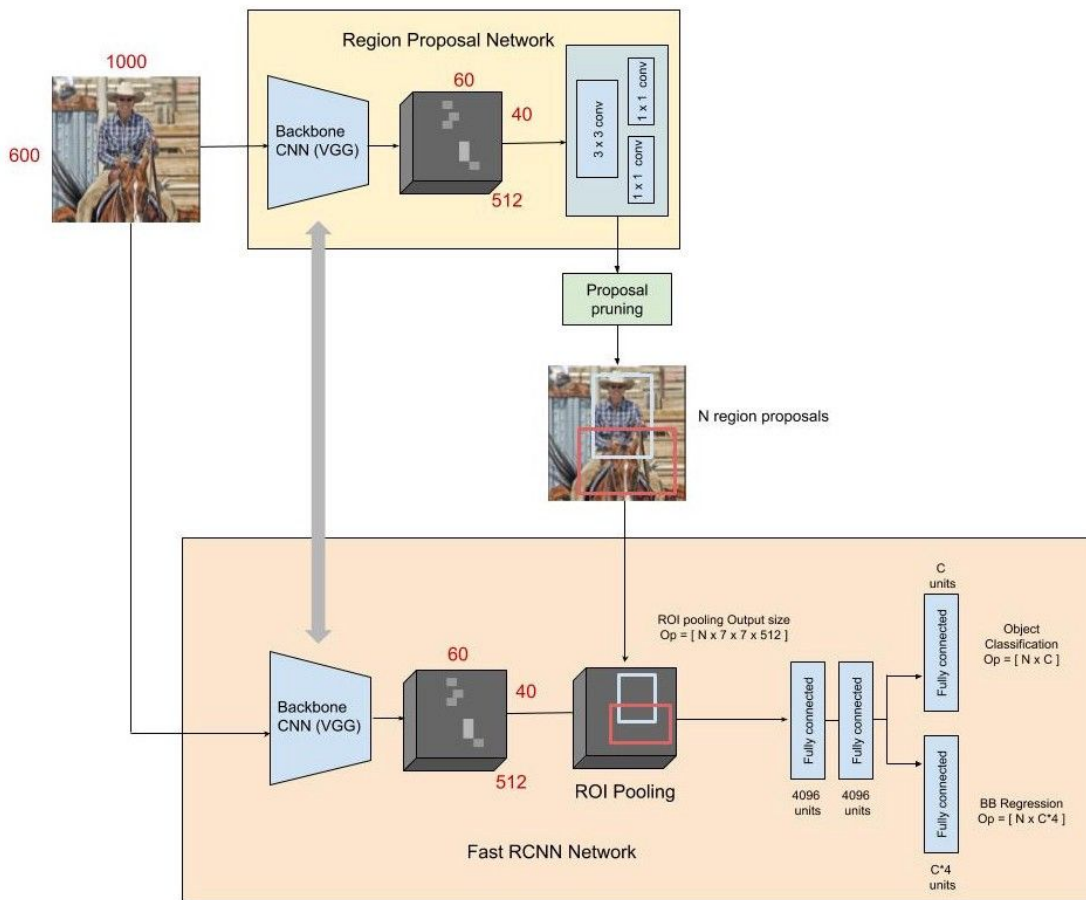
Pierwsze kroki w detekcji

- Dość skromne ;)
- Losowano X bounding boxów dla każdego obrazka
- Każdą zawartość bounding boxa przepuszczano przez sieć
- Wybierano bounding boxa i klasę o najwyższym zwróconym prawdopodobieństwie (lub wiele)



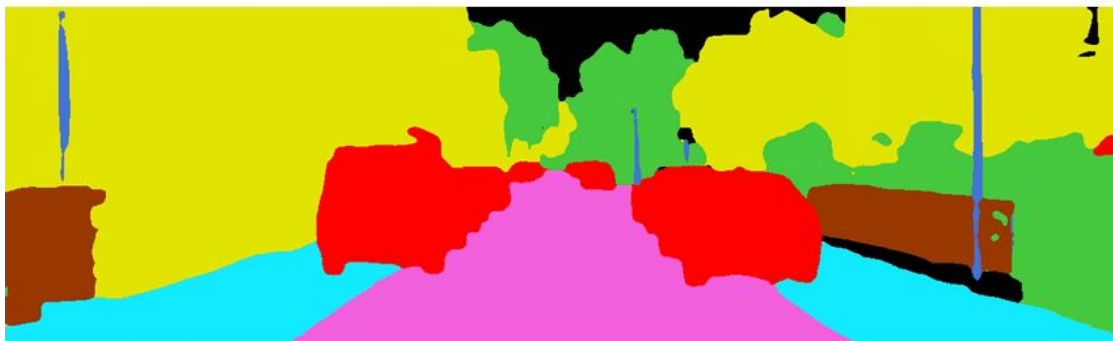
R-CNN









- Pierwsze “ciekawsze” podejście do problemu
- Nowy element: Region Proposal Network
- Następne iteracje: Fast R-CNN, Faster R-CNN



Segmentacja

- Inna wersja detekcji obiektów na obrazie
- Zadanie trudniejsze od zwykłego wyznaczania bounding boxów
- Przypisuje klasę **każdemu pikselowi**
- Główne architektury:
 - Mask R-CNN
 - UNet



 Road	 Sidewalk	 Building	 Fence
 Pole	 Vegetation	 Vehicle	 Unlabel



Przykład w kodzie + zadanie

Q&A



Feedback

