



**AGH**

**Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie**

**WYDZIAŁ ELEKTRONIKI, INFORMATYKI I TELEKOMUNIKACJI**

Praca magisterska

*Analiza ruchu pieszego i dostaw towarów w strefach  
ograniczonego ruchu*

*Analysis of pedestrian traffic and goods deliveries in  
Limited Traffic Zones*

Autorka:

Kierunek studiów:

Typ studiów:

Opiekun pracy:

*Klaudia Katarzyna Bednarz*

*Teleinformatyka*

*Stacjonarne*

*dr hab. inż. Mikołaj Leszczuk*

Kraków, 2021

*Upředzona o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór; artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także upředzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchylbiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.*

*Serdecznie dziękuję mojemu promotorowi  
dr hab. inż. Mikołajowi Leszczukowi za  
pomoc i czas poświęcony w realizacji tej  
pracy.*



## Spis treści

<b>1. Wprowadzenie</b>	7
1.1. Opis problemu	7
1.2. Cele pracy	8
1.3. Zawartość pracy	9
<b>2. Detekcja obiektów - stan wiedzy</b>	11
2.1. Znane metody	11
2.1.1. Sliding window	11
2.1.2. Selective Search	12
2.1.3. R-CNN	13
2.1.4. YOLO	14
2.2. YOLOv5 implementacja	15
<b>3. Analiza danych wejściowych</b>	17
3.1. Wybór lokalizacji poddanych badaniom	17
3.2. Obróbka danych	18
3.2.1. Zbieranie danych	18
<b>4. Analiza ruchu pieszego i dostaw towarów w Strefach Ograniczonego Ruchu</b>	21
4.1. Jak to działa	21
4.2. Algorytm	21
4.3. Coś tam dalej	21
<b>5. Uzyskane wyniki</b>	23
<b>6. Podsumowanie</b>	25



# 1. Wprowadzenie

We współczesnym świecie nieustannie rośnie potrzeba opracowywania coraz to nowszych i bardziej zaawansowanych systemów, dzięki którym gromadzenie i analiza danych stają się szybsze. Każdego dnia, używając aplikacji mobilnych, robiąc zakupy online, czy w szerokim ujęciu korzystając z dobrodziejstw internetu, przyczyniamy się do powiększania ilości przetwarzanych danych. Z tego powodu wprowadzane są rozwiązania, które pozwalają na automatyzację związanych z tym procesów i tworzenie modeli analitycznych, eliminując przy tym „manualną” pracę wykonywaną przez człowieka. Ostatnimi czasy coraz większe zainteresowanie budzi *Data Science* - dziedzina, która ciągle się rozwija i daje szerokie pole do usprawnienia żmudnych procesów związanych z przetwarzaniem, i interpretacją danych oraz wykorzystująca techniki uczenia maszynowego.

Uczenie maszynowe i sztuczna inteligencja prawdziwie zawładnęły światem eksploracji danych i znajdują zastosowanie w wielu sektorach życia publicznego. Współczesne systemy bankowe pozwalają na szybkie przetwarzanie i interpretację danych klientów oraz umożliwiają identyfikowanie potencjalnych oszustów czy wyłudzeń. Dzięki wykorzystaniu tej technologii w dziedzinie ochrony zdrowia - obserwujemy szeroki rozwój telemedycyny. Uczenie maszynowe wykorzystywane jest także w administracji publicznej, gdzie znajduje zastosowanie np. w instytucjach zajmujących się zapewnianiem bezpieczeństwa obywateli, oraz w transporcie - np. do wyznaczania efektywnych tras dla dostawców[4].

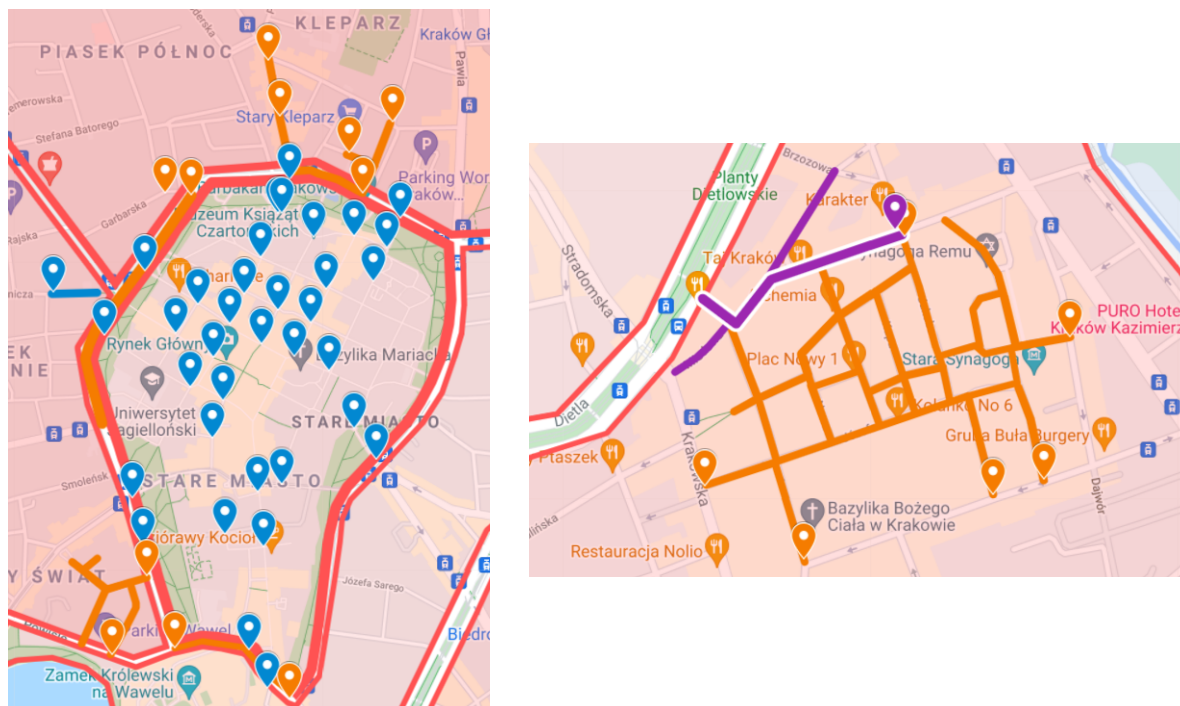
Techniki uczenia maszynowego wykorzystywane są także w zagadnieniu, z którym bezpośrednio związana jest niniejsza praca - widzeniu komputerowym (ang. *Computer vision*). Zadaniem tej dziedziny jest symulowanie procesu ludzkiego widzenia. Dzięki algorytmom uczącym się, możliwe jest odwzorowanie odkrywania i interpretacji zjawisk, a także klasyfikacji obiektów, które zachodzą podczas analizy otaczającego nas świata, poprzez zmysł wzroku. Gromadzenie informacji o rzeczywistości w tym wypadku odbywa się na podstawie analizy obrazu lub sekwencji obrazów. Dane pozyskiwane ze sceny poddanej interpretacji mogą być w tym procesie dzielone na mniejsze podzbiory, modyfikowane i wielokrotnie filtrowane. Wszystko po to, aby bez udziału człowieka udało się poprawnie „zrozumieć” kontekst sytuacji i np. rozpoznać sytuację zagrożenia albo zebrać dane do statystyki[2].

## 1.1. Opis problemu

Tematyka niniejszej pracy magisterskiej bezpośrednio nawiązuje do ostatniego z wymienionych przykładów na zastosowanie widzenia komputerowego w życiu społecznym. Problem, który próbuje zaadresować to potrzeba kontroli i gromadzenia wiedzy na temat ruchu na drogach po części wyłączonych z użytkowania (tzw. strefy ograniczonego ruchu). Dzięki analizie

publicznie dostępnych nagrań z kamer zainstalowanych w tych miejscach, możliwe będzie zebranie danych i statystyk, które posłużą jako materiał do podejmowania przez władze miasta dalszych decyzji w zakresie organizacji ruchu drogowego.

Strefa ograniczonego ruchu w Krakowie obejmuje teren Starego Miasta oraz Kazimierz (wyznaczona przez ulice zaznaczone na pomarańczowo i fioletowo na Rys.1.1.). W strefach obowiązują przepisy dotyczące dozwolonego ruchu pojazdów, godzin dostaw oraz opłat postojowych. Restrykcje te mają na celu ograniczenie ruchu samochodowego w ścisłym centrum oraz promowanie ekologicznych środków transportu jako sposób przemieszczania się na terenie miasta.



Rys. 1.1. Obowiązujące strefy ograniczonego ruchu w Krakowie[16]

Analiza zachowania pieszych i ruchu pojazdów na terenie strefy daje możliwość zapoznania się z faktycznym stanem rzeczy w kontekście przestrzegania przepisów drogowych, a także planowania usprawnień. Posiadanie takiej wiedzy niesie za sobą szereg potencjalnych korzyści, takich jak np. optymalizacja częstotliwości planowanych dostaw, wprowadzenie nadprogramowych środków zapewniających bezpieczeństwo pieszym, czy dodatkowego oznakowania.

## 1.2. Cele pracy

Niniejsza praca stanowi rozwinięcie istniejącego projektu, którego przedmiotem było opracowanie algorytmu wykrywającego i zliczającego rowerzystów, pojawiających się na udostępnionych filmach z krakowskich kamer miejskich. Celem pracy było stworzenie systemu graficznego rozpoznawania obiektów ruchomych, który pozwoliłby na analizę ruchu pieszych oraz dostaw towarów w strefach ograniczonego ruchu. Praca została wykonana w konsultacji i z pozyskaniem danych z Urzędu Miasta Krakowa oraz miejskich jednostek i ma być pomocą w ocenie inwestycji prowadzonych przez władze miasta lub planowaniu przyszłych projektów.



Podczas jej realizacji korzystano z danych pochodzących z publicznie dostępnych kamer internetowych. Do jej implementacji posłużył popularny algorytm detekcji obrazów - *YOLO*.

## 1.3. Zawartość pracy

. W rozdziale drugim zostało umieszczone wprowadzenie do zagadnienia detekcji obiektów oraz zgromadzono tam informacje na temat różnych znanych metod. Opisano schemat działania każdej z nich, w szczególności skupiając się na *YOLO*, czyli metodzie, która posłużyła do wykonania części praktycznej niniejszej pracy magisterskiej.

Pozostałe rozdziały - TBD ...



## 2. Detekcja obiektów - stan wiedzy

Detekcja obiektów jest zagadnieniem związanym z rozpoznawaniem wzorców, mającym zastosowanie w etykietowaniu obrazów, rozumieniu scen i rozpoznawaniu działań[5]. Polega na lokalizowaniu i klasyfikowaniu obiektów na obrazach i zdjęciach, gotowych plikach wideo oraz poprzez analizę wideo w czasie rzeczywistym. Każdy z tych sposobów wymusza inne podejście do wykrywania obiektów:

- obraz - wyszukiwanie obiektów na wyodrębnionej scenie jest najprostszym z wymienionych ujęć. Przebieg procesu detekcji polega w uproszczeniu na właściwym odczycie obrazu i przekazaniu go do algorytmu. Wynikiem przetwarzania jest obraz, na którym prawidłowo rozpoznane zostały przedmioty należące do odpowiadających im kategorii.
- wideo - detekcja obiektów w tym wypadku wymaga podzielenia filmu na klatki (ang. *frames*) oraz przekazania ich kolejno do algorytmu wykrywania aż do zakończenia pliku. Pojedyncze klatki wynikowe są zbierane i zapisywane w celu otrzymania nowego wideo z prawidłowo wychwyconymi przedmiotami.
- *real-time by camera* - podejście to wymaga w zasadzie tego samego zestawu czynności jak w przypadku detekcji wideo. W tym wypadku jednak pojedyncze klatki nie są odczytywane z pliku, ale bezpośrednio z kamery. Przetworzone klatki są „odkładane” w osobnym oknie, tak aby możliwa była obserwacja zmian w czasie rzeczywistym.

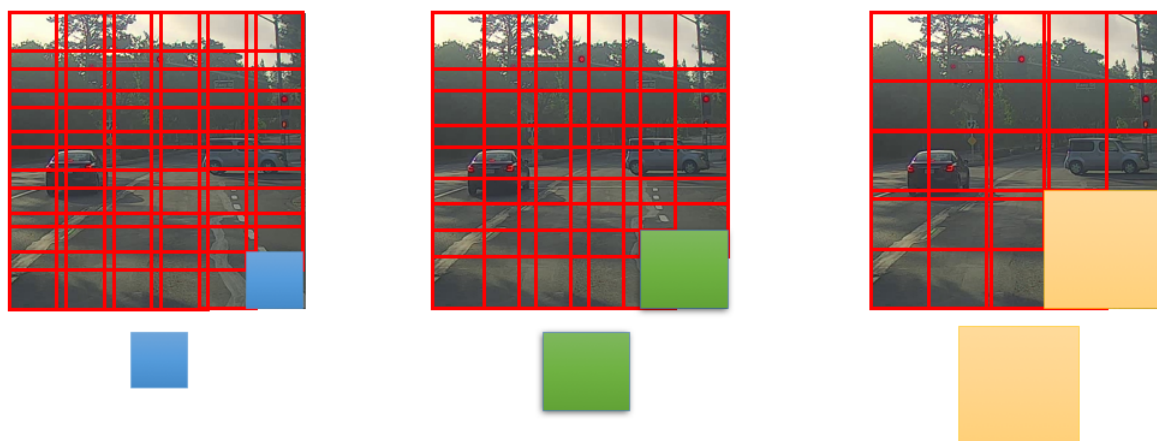
Zwykle wizualizacja wykrytych obiektów odbywa się poprzez otoczenie ich prostokątnymi ramkami *bounding box*, wyznaczającymi ich kontury. Obiekty, które posiadają te same cechy zostają przypisane do jednej kategorii. Prawidłowe przypisanie kategorii danemu obiektowi może być utrudnione, w przypadku kiedy mamy do czynienia z np. zakłóceniami tła lub zmianami perspektywy[5].

### 2.1. Znane metody

#### 2.1.1. Sliding window

Detekcja obiektów może odbywać się poprzez porównywanie punktów charakterystycznych obrazu będącego przedmiotem badań i obrazu referencyjnego. Niekiedy, stosując się do tego podejścia, możemy w wyniku otrzymać zbyt dużo podobnych do siebie obiektów (tzw. problem koncentracji cech), co utrudnia wykrycie konkretnego przedmiotu zainteresowania. Metoda okna przesuwanego (ang. *sliding window*) pozwala na wyeliminowanie tego problemu poprzez ograniczenie cech charakterystycznych, branych pod uwagę przy pojedynczym porównaniu wycinka obrazu z obrazem docelowym. Okno o określonym, a najlepiej - odpowiednio dla

danego problemu zoptymalizowanym rozmiarze, jest przesuwane w obrębie przeszukiwanego obrazu w każdej iteracji. Im większe okno, tym mniejsza złożoność detekcji, ale i też więcej objętych nim cech charakterystycznych - występuje problem z dopasowaniem. Zbyt małe okna nie zawierają wystarczającej liczby cech charakterystycznych i wiążą się z większym nakładem obliczeniowym. Wybór optymalnego rozmiaru okna warto podejmować bazując na rozmiarze obrazu będącego przedmiotem zainteresowania detekcji. Możemy manipulować także odległością przesuwu, modyfikując w ten sposób liczbę iteracji, tak aby nie ryzykować utraty informacji i jednocześnie minimalizować złożoność obliczeniową[6].



**Rys. 2.1.** Ilustracja działania metody okna przesuwnego z wykorzystaniem różnych wielkości okna[3].

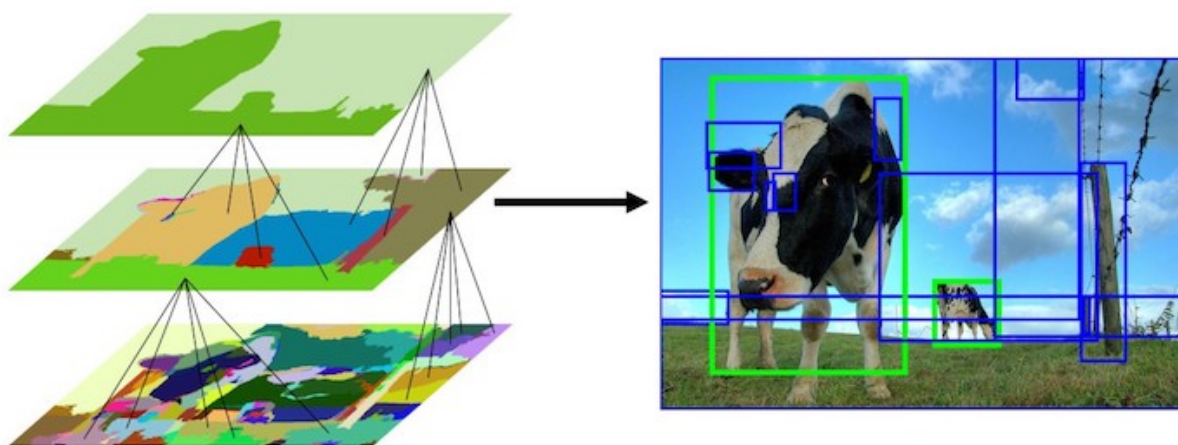
Wyszukiwanie metodą okna przesuwnego następuje po wytrenowaniu klasyfikatora na zbiorze próbek pochodzących z obrazu referencyjnego oraz obiektach losowych. Próbki pozyskane z porównywanych w ten sposób obrazów należą do osobnych klas - przedmiot detekcji zawiera tzw. przykłady pozytywne, natomiast przeszukiwane obiekty losowe, które nie są dla nas interesujące - negatywne. Po zakończeniu trenowania oraz wyborze rozmiaru okna i odległości przesuwu, wyuczony klasyfikator może posłużyć do wyszukiwania pożądanego przedmiotu detekcji na obrazie docelowym[12].

### 2.1.2. Selective Search

Metoda *Selective Search* jest podejściem należącym do grupy algorytmów polegających na wyznaczaniu podobnych do siebie regionów przeszukiwanego obrazu (ang. *Region proposal algorithms*) w procesie tzw. segmentacji. Miara podobieństwa regionów ustalana jest na podstawie kombinacji kryteriów:

- podobieństwa kolorów,
- podobieństwa tekstur,
- rozmiaru i miary wypełnienia.

Algorytm *Selective Search* polega na hierarchicznym generowaniu propozycji regionów rozpoczynając od nadsegmentacji obrazu, stopniowo zwiększając rozmiar proponowanych regionów. Uruchamiany w różnych warunkach, np. w komplementarnych przestrzeniach barw, pozwala wygenerować propozycje regionów wartych przeszukania o zróżnicowanych rozmiarach i proporcjach[10]. Metoda ta wiąże się z ryzykiem wyznaczenia licznych „fałszywych” segmentów, które w późniejszych krokach zostaną odrzucone przez algorytm rozpoznawania obiektów. Wraz z rosnącą liczbą nieprawidłowo wyznaczonych segmentów rośnie czas wykonywania algorytmu. Jest to jednak spodziewany i akceptowalny rezultat - tak długo, dopóki *Selective Search* zwraca regiony, w których istnieją poszukiwane obiekty.

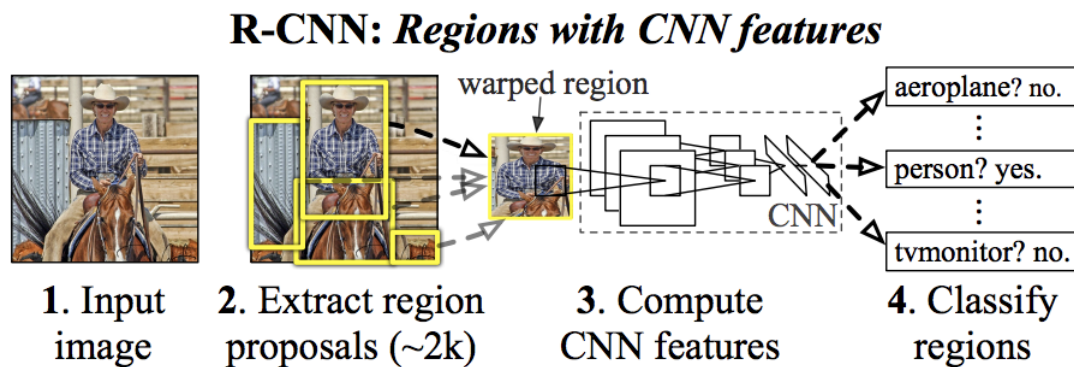


Rys. 2.2. Przykład przebiegu procesu segmentacji hierarchicznej obrazu[11].

### 2.1.3. R-CNN

Metoda *R-CNN* (ang. *Region Based Convolutional Neural Networks*) do rozwiązania problemu detekcji może wykorzystywać dowolne podejścia do wyboru regionów, zarówno *sliding windows* jak i wyszukiwanie selektywne. Istotą metody jest przedstawienie regionu za pomocą wektora cech, stworzonego dzięki sieci neuronowej (*CNN*), który jest uproszczoną reprezentacją analizowanego obrazu[7]. Proces wyodrębniania wektora cech z obrazu może przebiegać następująco:

- przekazanie obrazu wejściowego do sieci CNN,
- przepuszczenie obrazu przez kilka warstw konwolucyjnych oraz warstw łączących - w każdym kroku następuje filtrowanie danych w celu wyodrębnienia map cech, które posłużą do rozpoznawania wzorców,
- redukcja rozmiaru - łączenie map cech pochodzących z różnych propozycji regionów w wektory,
- przekazanie wektora cech do „w pełni połączonej warstwy” (*fully connected layer*), która zwraca wynik klasyfikacji[1].



Rys. 2.3. Schemat działania R-CNN[8].

Po klasyfikacji następuje postprocessing służący wyznaczeniu obwiedni *bounding box* otaczających obiekty, pozbyciu się duplikatów oraz ponownej oceny pól na podstawie innych obiektów obecnych na obrazie.[15]

- Fast R-CNN - TBD
- ...
- ...
- Faster R-CNN - TBD
- ...
- ...

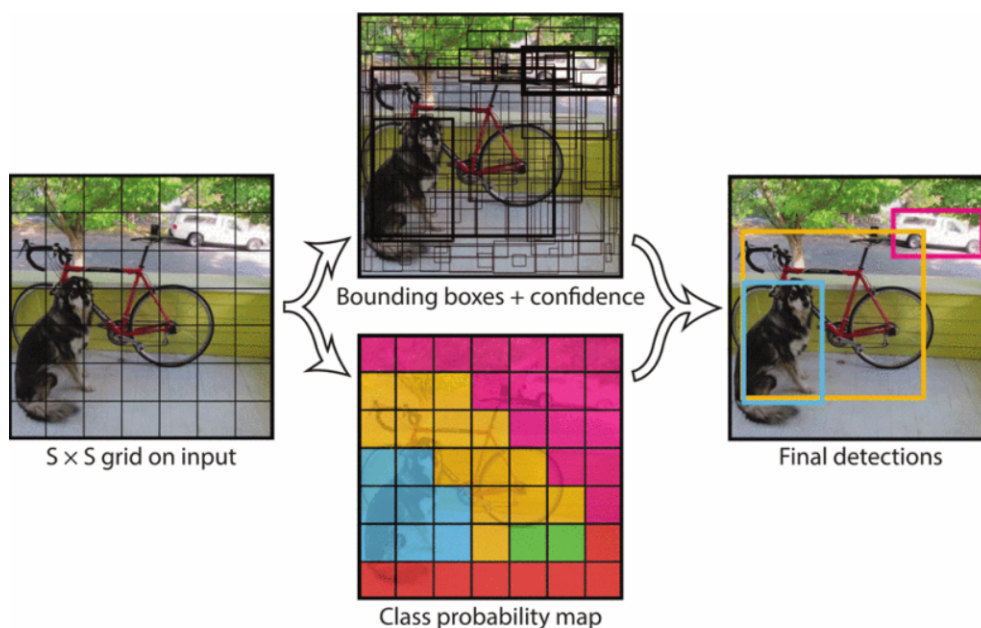
#### 2.1.4. YOLO

Metoda *YOLO* (ang. *You Only Look Once*) wykorzystuje konwolucyjne sieci neuronowe i polega na lokalizowaniu obiektów oraz etykietowaniu ich w tym samym czasie. Umożliwia wykrywanie wielu obiektów różnych kategorii na jednym obrazie. *YOLO* traktuje problem rozpoznawania obiektów jako pojedynczy problem regresyjny. Do detekcji wykorzystywana jest sieć konwolucyjna, która odpowiada za równoczesne przewidywanie obwiedni (*bounding box*) oraz prawdopodobieństwa, że wyznaczony obiekt należy do danej kategorii.

*YOLO* działa w następujący sposób:

- Na początku działania algorytmu następuje podział obrazu wejściowego za pomocą „siatki” na wiele małych komórek.
- W każdej z komórek w tym samym czasie (zgodnie z nazwą metody *You Only Look Once*) rozpoznawane są fragmenty obwiedni otaczających potencjalne obiekty. Jednocześnie określone jest prawdopodobieństwo, z jakim wyznaczony wycinek obwiedni otacza faktyczny obiekt danej klasy. Proces ten „zwraca” rozkład prawdopodobieństwa po wszystkich klasach, na których trenowana była sieć.

- Wynikiem działania algorytmu jest obraz, na którym wyznaczono obwiednie o różnej grubości linii - im grubsza linia tym większe prawdopodobieństwo, że wewnątrz znajduje się obiekt.
- Na koniec odrzucone w procesie *non-maximum suppression* zostają najcieńsze obwiednie (np. te z prawdopodobieństwem wystąpienia tam obiektu mniejszym niż 30%)[15].



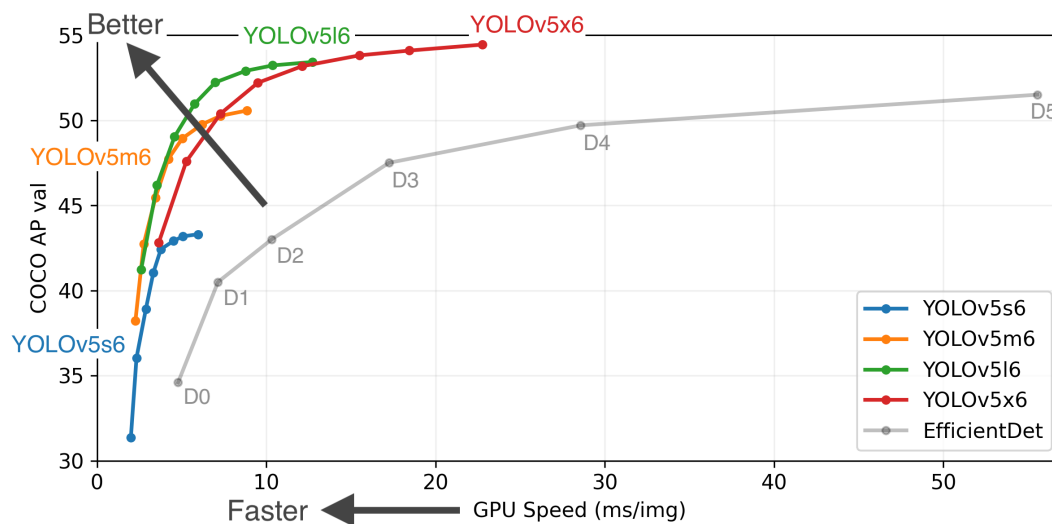
Rys. 2.4. Model systemu detekcji oparty o YOLO[15]

YOLO jest najszybszą znaną do tej pory metodą detekcji obiektów. Istnieje kilka wersji algorytmów z rodziny YOLO, które różnią się między sobą pod względem architektury i wydajności. Najnowsza oficjalnie opublikowana wersja, wykorzystana do implementacji części technicznej niniejszej pracy, to YOLOv5.

## 2.2. YOLOv5 implementacja

YOLOv5 to model detekcji obiektów stworzony przez firmę Ultralytics - autorów YOLOv3 opracowanego z wykorzystaniem PyTorch - popularnego framework'a znajdującego zastosowanie w rozwoju zagadnień uczenia maszynowego. Deweloperzy Ultralytics zajmują się prowadzeniem badań nad sztuczną inteligencją i przyszłymi metodami detekcji oraz udostępnianiem rozwiązań w zakresie uczenia maszynowego, nie wymagających od programistów wdrażania się w skomplikowane architektury sieci neuronowych. Repozytorium YOLOv5 jest open-source'owe, a udostępnione modele mogą być wykorzystywane i rozszerzane przez programistów do własnych projektów.

YOLOv5 udostępnia 4 modele wytrenowane na zbiorze danych COCO (*s*, *m*, *l* oraz *x*). Każdy z nich wyróżnia się inną dokładnością i wydajnością[14].



Rys. 2.5. Porównanie wariantów YOLOv5[14]

Do treningu modelu przygotowanego w ramach niniejszej pracy magisterskiej wykorzystany został wariant *YOLOv5l6*. Wybór podyktowany został dobrymi metrykami modelu oraz kwestiami wydajnościowymi - trenowanie z zastosowaniem *YOLOv5x6* trwało zbyt długo, nawet przy wykorzystaniu układu *GPU* w *Google Colab*.



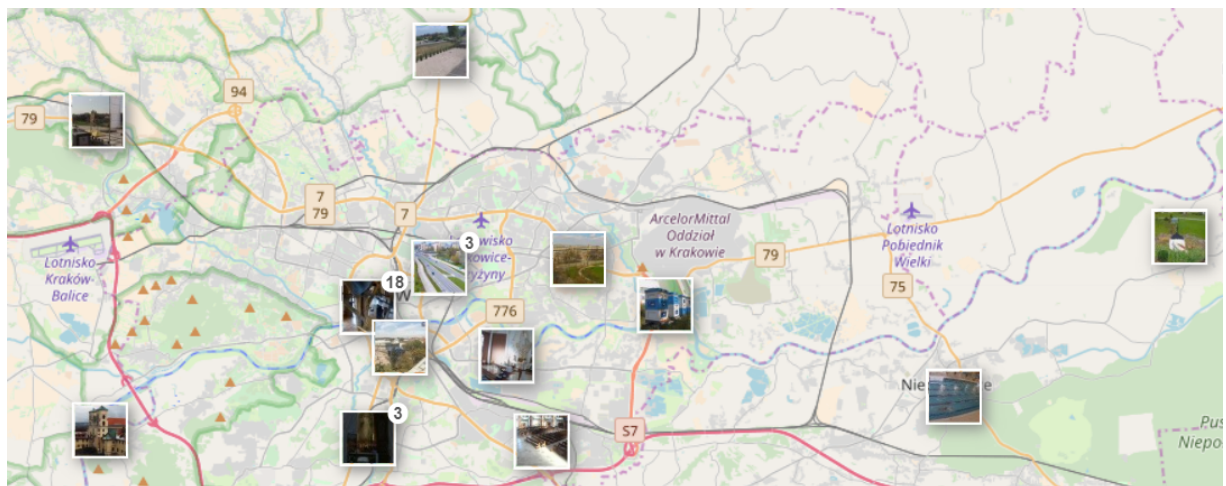
## 3. Analiza danych wejściowych

### 3.1. Wybór lokalizacji poddanych badaniom

Od 2014 roku, kiedy Rada Miasta Krakowa podjęła temat stworzenia monitoringu wizyjnego w referendum zorganizowanym celem poprawy bezpieczeństwa obywateli, na terenie Krakowa obserwujemy zwiększanie się liczby zainstalowanych kamer w przestrzeni publicznej. Łącznie, biorąc pod uwagę różne podmioty sfery komunalnej, takie jak Policja i Straż Miejska, czy placówki oświatowe, jest ich 12698 (stan na rok 2018)[9].

Kamery monitoringu nie są ogólnodostępne, a wykorzystanie ich do pracy magisterskiej wymagałoby uzyskania pozwolenia Straży Miejskiej/ZDMK. Jednakże, niniejszy projekt ma na celu zaprezentowanie działania modelu i algorytmu, które w zastosowaniu będą uniwersalne. Umożliwi wykorzystanie ich na obrazach pochodzących z różnych źródeł. W związku z tym do realizacji tego projektu posłużono się obrazami publicznie dostępnych kamer internetowych, z których obraz jest transmitowany w czasie rzeczywistym.

W Krakowie zainstalowane są 34 publicznie dostępne kamery internetowe[13].



**Rys. 3.1.** Rozmieszczenie kamer internetowych w Krakowie[13]

Przedmiotem badań tej pracy jest analiza ruchu pieszych i dostaw towarów w strefach ograniczonego ruchu. Wybrane kamery internetowe znajdują się w strefach przedstawionych w rozdziale 1. Lokalizacje poddane analizie to:

- Rynek Główny - widok z okna (kamera nieruchoma),
- Rynek Główny - widok od strony ul. Brackiej (kamera ruchoma),

- Ulica Grodzka (z Hotelu Senacki),
- Ulica Grodzka - Rynek Główny.

## 3.2. Obróbka danych

Elementem niezbędnym do rozpoczęcia pracy nad modelem było zgromadzenie odpowiedniej „bazy” filmów, w której istotne było przechwycenie kontrastujących warunków pogodowych, zacielenia danego miejsca a także pór dnia. Dzięki temu obiekty wybierane z poszczególnych klatek i przekazywane dalej do treningu modelu, były zróżnicowane. Zbieranie danych przeprowadzono na przełomie marca i kwietnia tego roku. Skompletowano godzinne filmy przechwytywane bez przerwy ze wszystkich czterech kamer z kilkunastu dni. Wybór takiego przedziału czasu umożliwił zebranie danych z tygodnia zwykłego oraz wielkanocnego.

Dodatkową wartością płynącą z eksperymentu bazującego na danych z tamtego okresu jest trwająca pandemia i osiągane wtedy rekordy w liczbie zachorowań na covid-19, a także obowiązujące w Polsce obostrzenia.

### 3.2.1. Zbieranie danych

W pierwszej fazie kompletowania danych zrealizowano przechwytywanie transmisji z wybranych kamer oraz zapisywanie ich lokalnie na dysku w formacie .mp4. Narzędzie, które zostało do tego wykorzystane to *youtube-dl* - command-line'owy program, który pozwala na pobieranie filmów z m.in. z *YouTube*, ale także z innych źródeł.

```
#!/bin/bash
timeout 3599 /usr/local/bin/youtube-dl -o "/media/klaudia/8fcc6fb9-4004-493e-b4b6-b7efdc1df57b/klaudiamgr/camers/grodzka_rynek/%(title)s_$(date '+%Y%m%dT%H%M%S%z').%(ext)s" http://185.70.181.30:8080/hls/live.stream.m3u8
```

Rys. 3.2. Przykład wykorzystania narzędzia *youtube-dl*

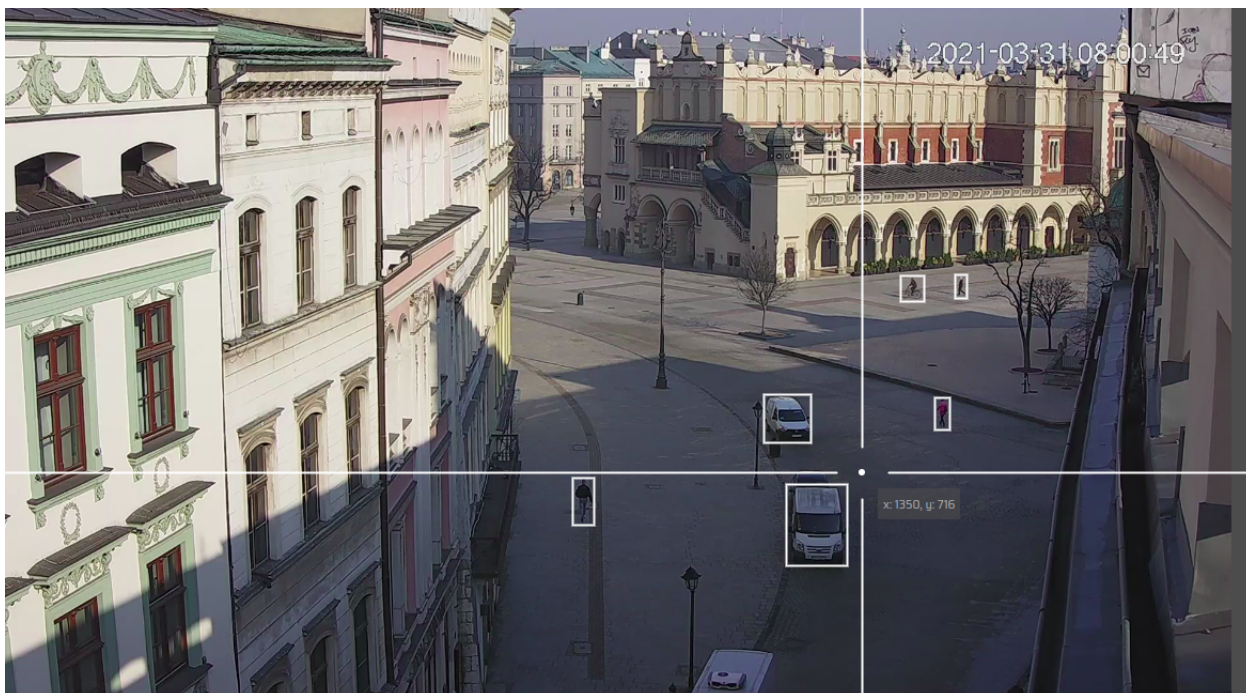
Aby uniknąć manualnego uruchamiania skryptów realizujących pobieranie zapisów kamer, wprowadzono automatyzację poprzez wykorzystanie *crontaba*. Opracowano także skrypt, umożliwiający zapisywanie filmów ze wszystkich kamer równolegle. W początkowej fazie kompletowania danych podjęto próbę zapisu filmów 24-godzinnych, jednak takie podejście wiązało się z wysokim ryzykiem utraty danych. Przerwanie połączenia internetowego skutkowało natychmiastowym zakończeniem zapisu. W związku z tym, w ostatecznym rozwiązaniu zadanie uruchamiane było co godzinę, a utworzona w ten sposób „baza” filmów składa się z kilkuset plików .mp4.

Kolejnym etapem pracy nad zbiorem danych przekazanych później do modelu było wyodrębnienie i zapis klatek z poszczególnych filmów przy pomocy narzędzia *VLC*. Aby uzyskane w ten sposób zdjęcia nie były zbyt podobne do siebie, wybrano do tego filmy pochodzące z różnych dni i godzin, przedstawiające odmienne warunki pogodowe.

Po zgromadzeniu odpowiedniej ilości zdjęć, przystąpiono do procesu etykietowania widocznych na nich obiektów z wykorzystaniem open-source'owego narzędzia *makesense.ai*. Do nowo

utworzonego projektu kategorii „Object detection” wprowadzono klasy „person”, „bicycle”, „car” oraz „cargovelo”, wykorzystane później do trenowania modelu.

Późniejsze postępowanie w programie *makesense.ai* wiązało się ze żmudnym procesem otaczania obiektów poszczególnych klas *bounding-box*’ami oraz oznaczania ich odpowiadającymi etykietami. Po zakończeniu każdego etapu prac należało wyeksportować przetworzone dane - wykorzystane narzędzie umożliwia eksport w formacie *YOLO*.



Rys. 3.3. Proces etykietowania obiektów

*YOLO* posiada własny format etykietowania danych, w którym dla danego obrazu tworzony jest odpowiadający mu plik .txt o takiej samej nazwie. Każda linia pliku jest reprezentacją innego obiektu rozpoznanego na obrazie. Następujące po sobie kolumny oznaczają kolejno:

- klasę obiektu,
- współrzędne x i y punktu przecięcia przekątnych *bounding-box*’a otaczającego obiekt,
- szerokość *bounding-box*’a,
- wysokość *bounding-box*’a.

```
0| 0.784120 0.394917 0.011159 0.046872
0 0.767811 0.571820 0.016309 0.061992
1 0.743348 0.396429 0.017167 0.037800
0 0.480258 0.698827 0.016309 0.077111
2 0.643348 0.581647 0.038627 0.072576
2 0.745923 0.800886 0.061803 0.185975
```

Rys. 3.4. Wygenerowane etykiety w formacie YOLO



## **4. Analiza ruchu pieszego i dostaw towarów w Strefach Ograniczonego Ruchu**

...

### **4.1. Jak to działa**

...

### **4.2. Algorytm**

fragmenty kodu

### **4.3. Coś tam dalej**

...



## **5. Uzyskane wyniki**

...





## **6. Podsumowanie**

...



## Bibliografia

- [1] P. Dong i W. Wang. „Better region proposals for pedestrian detection with R-CNN”. *2016 Visual Communications and Image Processing (VCIP)*. 2016, s. 1–4. DOI: 10.1109/VCIP.2016.7805452.
- [2] X. Li i Y. Shi. „Computer Vision Imaging Based on Artificial Intelligence”. *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*. 2018, s. 22–25. DOI: 10.1109/ICVRIS.2018.00014.
- [3] R. Arora. *Convolutional implementation of the sliding window algorithm*. <https://medium.com/ai-quest/convolutional-implementation-of-the-sliding-window-algorithm-db93a49f99a0>. [Online; accessed 28-February-2021]. Sty. 2020.
- [4] *Machine Learning Teoria i praktyka*. [https://www.sas.com/pl\\_pl/insights/analytics/machine-learning.html](https://www.sas.com/pl_pl/insights/analytics/machine-learning.html). [Online; accessed 06-March-2021].
- [5] Z. Zhao i in. „Object Detection With Deep Learning: A Review”. *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), s. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.
- [6] J. Lee, J. Bang i S. Yang. „Object detection with sliding window in images including multiple similar objects”. *2017 International Conference on Information and Communication Technology Convergence (ICTC)*. 2017, s. 803–806. DOI: 10.1109/ICTC.2017.8190786.
- [7] Sharif Elfouly. *R-CNN (Object Detection)*. <https://medium.com/@selfouly/r-cnn-3a9beddf55a>. [Online; accessed 04-March-2021]. Lip. 2019.
- [8] Ross Girshick i in. „Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Czer. 2014.
- [9] *ROZWÓJ MONITORINGU WIZYJNEGO W KRAKOWIE*. [https://www.bip.krakow.pl/?dok\\_id=87015](https://www.bip.krakow.pl/?dok_id=87015). [Online; accessed 11-July-2021].
- [10] K. E. A. van de Sande i in. „Segmentation as selective search for object recognition”. *2011 International Conference on Computer Vision*. 2011, s. 1879–1886. DOI: 10.1109/ICCV.2011.6126456.
- [11] J. R. R. Uijlings i in. „Selective Search for Object Recognition”. *International Journal of Computer Vision* 104.2 (2013), s. 154–171.
- [12] A. Helwan i D. U. Ozsahin. „Sliding Window Based Machine Learning System for the Left Ventricle Localization in MR Cardiac Images”. 9 (2017). DOI: <https://doi.org/10.1155/2017/3048181>.

- [13] *WorldCam cały świat w jednym miejscu, Kraków - Kamery internetowe*. <https://www.worldcam.pl/kamery/polska/krakow/lista/25>. [Online; accessed 11-July-2021].
- [14] *YOLOv5*. <https://github.com/ultralytics/yolov5>. [Online; accessed 11-July-2021].
- [15] J. Redmon i in. „You Only Look Once: Unified, Real-Time Object Detection”. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, s. 779–788. DOI: 10.1109/CVPR.2016.91.
- [16] *Zarząd Dróg Miasta Krakowa* (<https://zdmk.krakow.pl/>). <https://www.google.com/maps/d/viewer?mid=1NIH-VbeBqdyei7IrSb4KFEwg-Sg&ll=50.058324615317105\%2C19.974245575037344&z=13>. [Online; accessed 13-March-2021].