

# Raport z projektu:

## Stresowanie modelu rozpoznającego obiekty

Autor: Edyta Cal

### 1. Opis projektu

#### 1.1. Cel projektu

Celem projektu było zrozumienie czy wielkość, położenie a także tło na którym znajduje się obiekt ma wpływ na jego detekcję.

#### 1.2. Założenia projektowe

Użyto modelu detekcji obiektów YOLOv3 wytrenowanego na bazie COCO 2017.

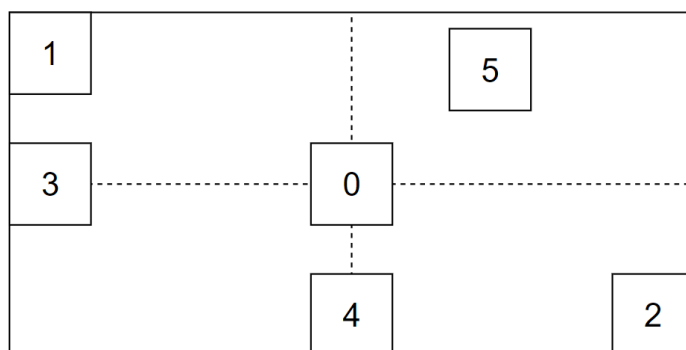
Detekcję obiektów wykonywano na zdjęciach w rozdzielczości Full HD (1920x1080 px)

Wybrano cztery, znacząco różniące się od siebie klasy: jabłko, samochód, pies oraz krzesło

Dla każdej z klas wybrano po 20 obiektów, o wymiarach 1024x1024 px (jabłko pies), 1024x2048 px (krzesło) oraz 2048x1024 px (samochód).

Wybrano 13 tła: białe, szum RGB, po dwa charakterystyczne dla danej klasy i trzy losowe

Wybrano 6 lokalizacji obiektu wg. schematu nr 1, gdzie liczba 5 oznacza miejsce losowe



Schemat nr 1: Oznaczenia lokalizacji obiektu na zdjęciu. Lokalizacja 5, oznacza lokalizację losową

Obiekty były skalowane do wielkości 25%, 18,75%, 12,5% oraz 6,25% oraz umieszczane w lokalizacjach przedstawionych na schemacie nr 1.

Obiekty umieszczane były także “w skali 100%” (skalowane były na tyle by zmieściły się w rozdzielczości Full HD), jednak umieszczane były wtedy tylko w lokalizacji nr 0 (środek zdjęcia).

## 2. Przebieg projektu

### 2.1. Tworzenie danych do analizy

Projekt zaczęto od znalezienia po 20 zdjęć z obiektami dla każdej z klas, wycięciu obiektów i umieszczeniu ich na przezroczystym tle. Obiekty w zależności od klasy, były rozmiaru 1024x1024 px w przypadku klasy jabłko i pies, 1024x2048 px w przypadku klasy samochód oraz 2048x1024 px w przypadku klasy krzesło. Spreparowane obiekty można znaleźć w folderze *objects*.

Napisano algorytm *generator.py*, którego celem było skalowanie obiektów i umieszczenie ich na tłach, w lokalizacjach wskazanych na schemacie nr 1. . Jego drugim celem było zebranie danych użytych do generacji zdjęć w pliku *.csv*.

Napisano także algorytm *detektor\_yolo.py*, który używając modelu YOLOv3 przeprowadzał rozpoznawanie obiektów na wygenerowanych wcześniej zdjęciach. On także zbierał dane do pliku *.csv*.

W celu ułatwienia analizy napisano także algorytm *heatmap.py*, który służył wygenerowaniu map ciepła występowania interesujących nas klas na zdjęciach z bazy COCO.

Napisano algorytmy takie jak *detection\_image\_test.py*, służący detekcji i podglądowi wyniku działania algorytmu rozpoznającego obiekty na pojedynczych zdjęciach (głównie w celu analizy anomalii) oraz kilka innych służących głównie organizacji wytworzonych danych.

### 2.2. Analiza zebranych danych

Analiza danych została przeprowadzona na podstawie wygenerowanych danych, informacji na temat bazy COCO ( w tym mapy ciepła dla klas oraz liczba obiektów w bazie reprezentujących daną klasę) oraz zasadzie działanie modelu YOLOv3.

#### ■ Wielkość obiektu

Wielkość obiektu ma największy i najbardziej zauważalny wpływ na jego rozpoznawanie. Następuje gwałtowny spadek w skuteczności rozpoznawania obiektów między skalami 18,75% a 12,5% (wielkość obiektu pomiędzy ok. 380 a 120 px (największy wymiar))

Skala:	Jabłko	Pies	Samochód	Krzesło
100%	84,23	95,38	75,76	68,07
25%	77,67	57,05	78,46	65,93
18,75%	54,48	37,56	75,75	54,48
12,5%	14,74	8,58	56,6	23,71
6,25%	0,57	0,32	15,38	1,35
Ogółem	38,75	28,66	57,31	37,66

Tabela nr 1: Skuteczność rozpoznawania klas w zależności od skali

### ■ Tekstura/Złożoność obiektu

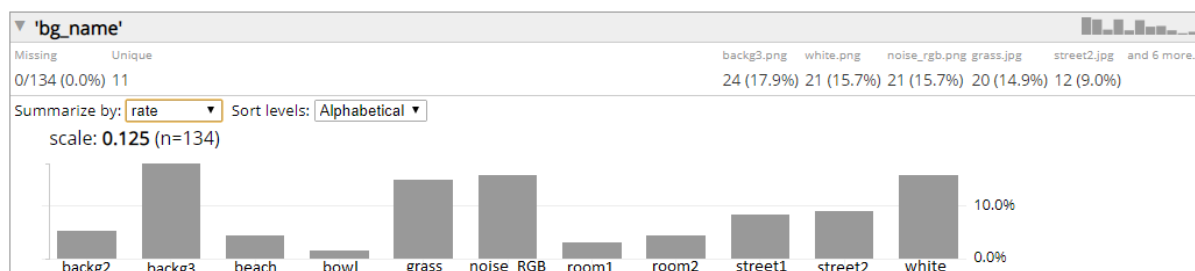
Warto uwzględnić na ilu obiektach danej klasy wytrenowano model. W bazie COCO znajduje się 13354 zdjęć krzesel, 12786 zdjęć samochodów, 4562 zdjęć psów oraz 1662 zdjęć jabłek. Sugeruje to, że rozpoznawalność krzesel powinna być największa z tych czterech klas, tymczasem jest mniejsza od rozpoznawalności jabłka, którego zdjęć znajduje się najmniej w bazie. Wynika to z tego, że jabłko jest stosunkowo prostym obiektem. Jednak różnica w rozpoznawalności samochodów a krzesel jest ogromna. Wygląd obu obiektów jest skomplikowany, jednak model detekcji działa w inny sposób niż ludzki mózg. Dla ludzi ważniejsze są kształty, a dla modelu tekstury. Samochody są na ogół budowane z tych samych materiałów (blacha, szkło), tymczasem krzesła mogą być zbudowane z różnych materiałów (np. drewno, plastik) mogą też być dodatkowo obite różnym materiałem.

### ■ Tło

Detektor osiągał radził sobie dobrze w przypadku tła białego i szumu RGB (*white* i *noise\_RGB*) w przypadku wszystkich klas. Dla każdej z klas najslabsze wyniki były dla tła z graffiti (tło *backg1*).

Dla małych skal (6,25% , 12,5%), można było zauważyć zależność między zawartością bazy COCO, skutecznością rozpoznawania detektora.

Najwyraźniej było to widać dla klasy pies:



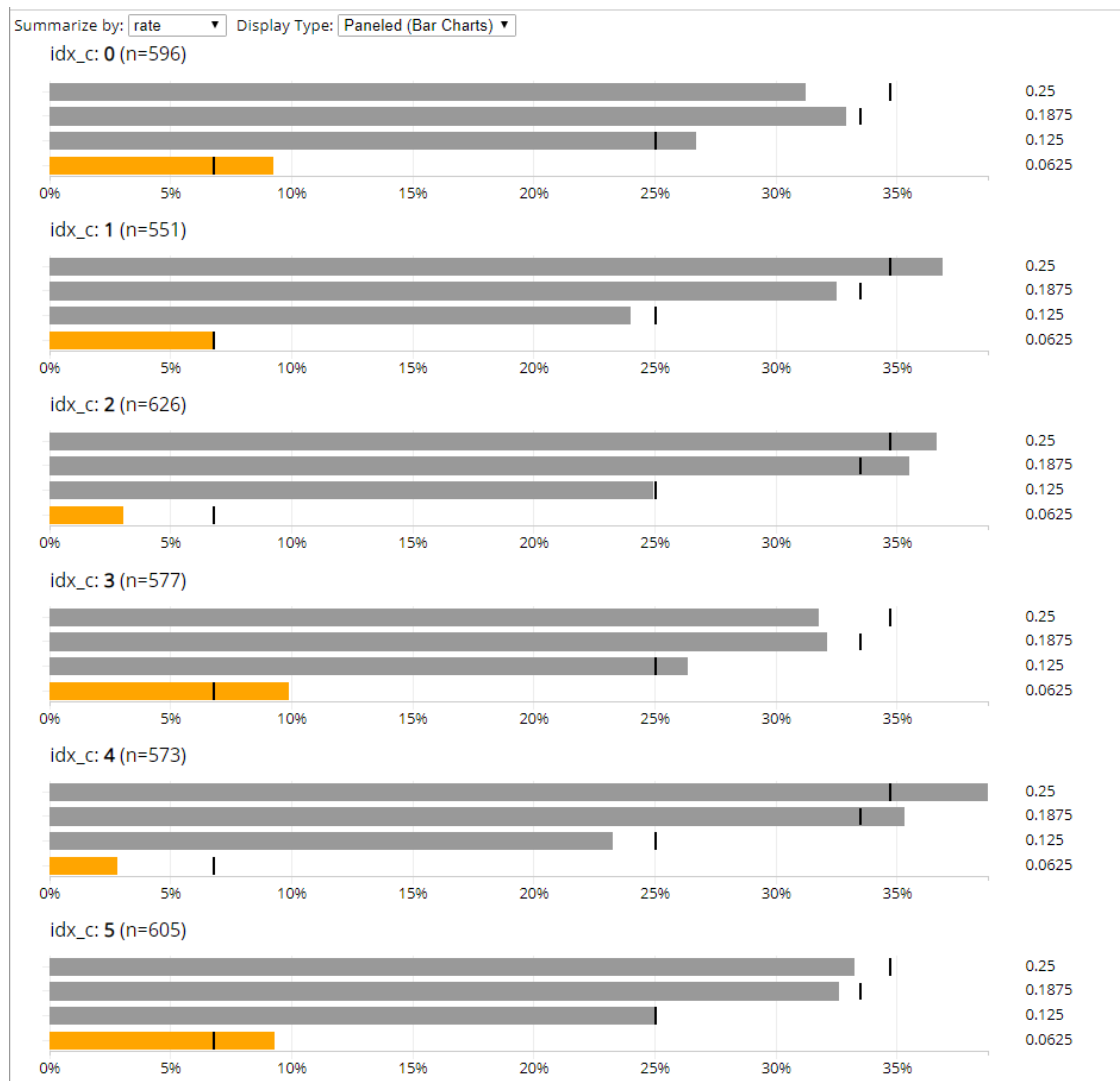
Rysunek 1: Skuteczność rozpoznawania klasy pies dla skali 12,5% w zależności od tła.

W bazie COCO, można wyróżnić dużą podgrupę zdjęć, które zawierają obiekt klasy pies na tle trawy. Natomiast tła *backg3* i *grass* to zdjęcia w zdecydowanej większości przedstawiające trawę.

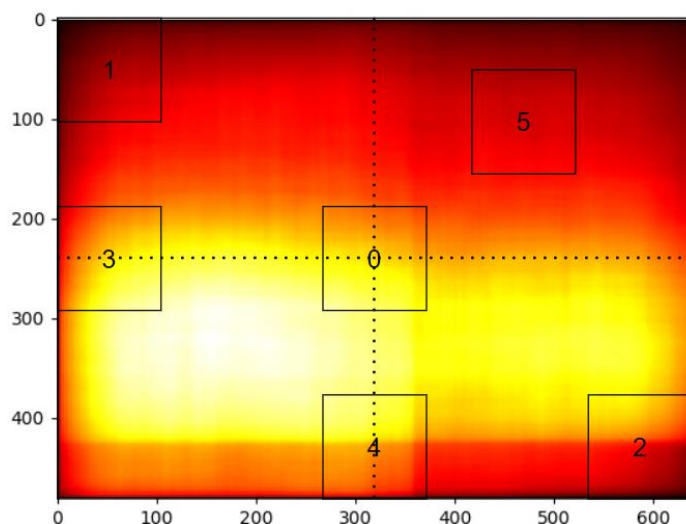
Podobną zależność można wyraźnie zauważyć także dla klasy krzesło; dla tła *backg3*, *beach*, *grass*.

#### ■ Lokalizacja obiektu

Po analizie otrzymanych danych okazało się, że dla małych skal (12,5% i 6,25%), rozmieszczenie obiektów na zdjęciach ma wpływ na skuteczność rozpoznawania obiektów. Jednak po porównaniu ich z mapami ciepła wygenerowanych na podstawie zdjęć z bazy COCO, zależności te nie pokrywają się w pełni. Przez to nie można jednoznacznie stwierdzić, iż skuteczność detekcji obiektów zależy od ich lokalizacji.



Rysunek nr 2: Skuteczność rozpoznawania obiektów klasy samochód w zależności od lokalizacji obiektu i skali (na pomarańczowo zaznaczono dane dla skali 6,25%). Sądząc po mapie ciepła (Rysunek nr 3), wynik dla lokalizacji 1 i 2 powinien być zbliżony.

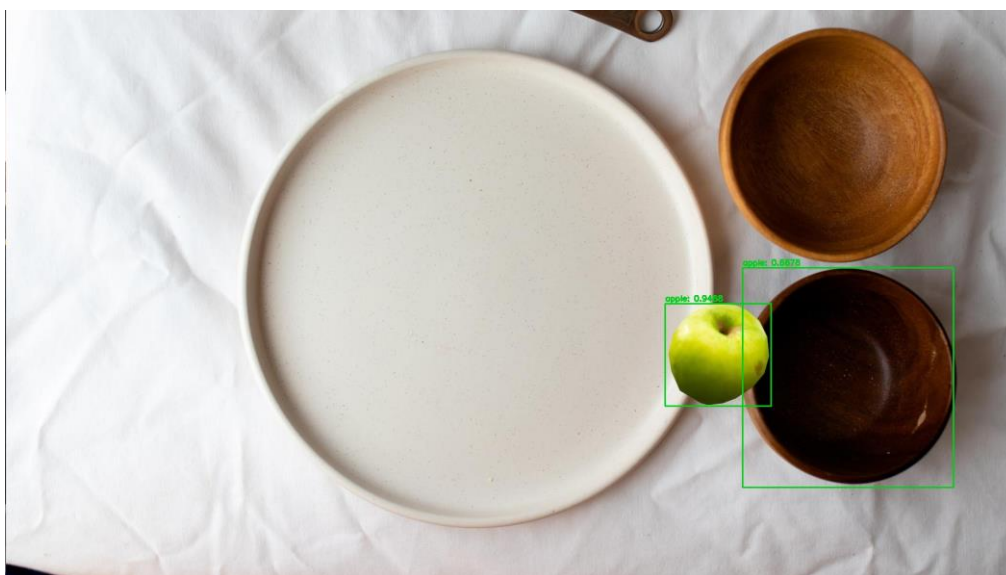


Rysunek nr 3: Mapa ciepła dla obiektów klasy samochód wygenerowana na podstawie zdjęć z bazy danych COCO z naniesionym schematem oznaczeń lokalizacji obiektów.

### 2.3. Anomalie

Na 26000 wygenerowanych zdjęć wystąpiło tylko 16 anomalii (wystąpiły one tylko dla 7 obiektów). Polegały one najczęściej na wykryciu błędnej obiektu, gdy w istocie go tam nie było (15), i tylko raz na rozpoznaniu obiektu jako dwóch różnych.

Jedną z ciekawszych anomalii jest wynik detekcji przeprowadzonej na zdjęciu *E15\_002879*. Użyte do jego generacji tło *plate* zostało użyte do generacji także 1999 innych zdjęć, jednak tylko w tym jednym przypadku miska została rozpoznana jako jabłko.



Zdjęcie nr 1: *E15\_002879*.

### 3. Wnioski

Jak dla każdego modelu detekcyjnego, najważniejsze jest odpowiednie przygotowanie bazy danych do jego treningu. Rozmiar obiektu odgrywa ogromną rolę w detekcji. W przypadku niewielkich obiektów, tło zaczyna odgrywać dużą rolę. Jednak trzeba też, wziąć pod uwagę złożoność klasy obiektów - im bardziej różnorodne obiekty wchodzi w skład klasy, tym więcej zdjęć tej klasy obiektów powinna zawierać baza treningowa modelu.

### 4. Materiały

[GitHub](#) - kod źródłowy, wygenerowane dane oraz obiekty i tła użyte do ich generacji

[COCO](#) - baza danych na której trenowany był model

[Model YOLOv3](#) - wytrenowany YOLOv3 na bazie COCO

[Unsplash](#) - źródło tła oraz obiektów