# Poprawka

5 grudnia 2020

## Spis treści



## 1 Refaktoryzacja kodu

Pomocnicze funkcje przeniosłem do package-a **utills**, modele i generatory do **models** a często użyte stałe do pliku **config**.

## 2 Requirements

Wymagania wygenerowałem z **pipreqs** – bardzo fajne narzędzie, dzięki za podrzucenie!

## 3 Augmentacja

Z augmentacją miałem najwięcej problemów, nie udało mi się zmusić ImageDataGenerator-a do modyfikowania label-ów przez co większość standardowych transformacji powodowała konflikt z y-kami wprowadzanymi do sieci. Z podobnych powodów nie użyłem channel_shift_range do zmiany wartości RGB – nie chciałem chciałem tracić własności w której sztucznie dodane brzegi zdjęć miały wartości zerowe (dla sieci trenowanej z takim generatorem błąd na zbiorze validacyjnym był wyraźnie większy), żeby tego uniknąć musiałbym przenieś większość preprocessing-u na generator.
Próbowałem użyć odręcznie napisanego generatora (custom_generator z pliku generators.py) ale działał zdecydowanie zbyt wolno (generowałem obrazki ręcznie, bez wykonywanie operacji równolegle na macierzach). Na koniec użyłem do zmiany jasności ImageDataGenerator natomiast wycięte kawałki zdjęć dodałem do oryginalnego datasetu (w moim przypadku to była ta kluczowa operacja bez której sieć uczyła się przewidywać środek zdjęcia, mam nadzieję, że można to podpiąć pod ten przypadek w którym transformacja wykonana na wszystkich danych treningowych daje lepsze wyniki).

# 4 Uwagi techniczne

Wytrenowanego modelu używającego resnet jako backbone nie wrzuciłem na repozytorium – zapisany model ważył troszeczkę za dużo jak na ograniczenia githuba.
Wytrenowany po zmianie augmentacji model to cascade_model.