

# Zaawansowane zagadnienia sieci neuronowych - Projekt - Dokumentacja końcowa

Zadanie: Przygotować program do detekcji maseczek ochronnych (real-time) na obrazie z wykorzystaniem detektora YOLO. Zaprezentować działanie na

publicznie dostępnym streamie albo własnym filmie. Dokonać oceny działania.

Michał Kozak  
*Wydział Elektrotechniki i Informatyki  
Politechnika Warszawska  
Warszawa, Polska  
01176006@pw.edu.pl*

## I. WSTĘP

Przygotowano projekt służący do detekcji maseczek ochronnych (real-time) z wykorzystaniem rodziny architektur do wykrywania obiektów YOLO, wstępnie wytrenowanych na zbiorze danych COCO. Dodatkowo w ramach chęci zrobienia czegoś więcej wprowadzono 3 klasę do wykrywania twarzy z nieprawidłowo założoną maseczką. Finalnie wybrano YOLO w wersji 5 oraz wykorzystano metodę uczenia transferowanego ze względu na ograniczone dostępne zbiory danych potrzebne do wykonania zadania.

Przeprowadzono testy, które mają za zadanie wskazać silne i słabe strony różnych architektur YOLOv5 Rys. 1, jak przekładają się różnice między ich rozmiarami a efektywnością, obciążeniem obliczeniowym oraz pamięciowym czy odpornością na skrajne scenariusze.

### 1) Przyjęte założenia projektowe:

- Program będzie w stanie rozpoznać osoby z niepoprawnie założoną maseczką - pomarańczowa obwódka, bez założonej maseczki - różowa obwódka, dla osób z poprane założoną maseczką obwódka będzie koloru czerwonego.
- Założono, że osoby na zdjęciach będą dobrze widoczne to znaczy, że zdjęcia nie były robione nocą lub w złym świetle. Decyzja ta została podjęta ze względu na ograniczone zbiory danych z takimi próbami oraz zapotrzebowanie na większą moc obliczeniową (większy model) w celu uzyskania satysfakcyjnej dokładności w trudnych przypadkach. Zostaną natomiast przeprowadzone testy dla trudnych przypadków w celu weryfikacji jak model sprawdza się ekstremalnych warunkach w obecnej formie.
- Wykorzystane zostaną w aplikacji tylko narzędzia bezpłatne.
- Głównym językiem programowania jest Python.

2) Użyta architektura: YOLOv5 [2] to konwolucyjna sieć neuronowa która jednocześnie przewiduje wiele ramek wokół obiektów i prawdopodobieństwa klas dla tych ramek. Nazwa YOLOv5 to skrót od *You Only Look Once* co odnosi się do

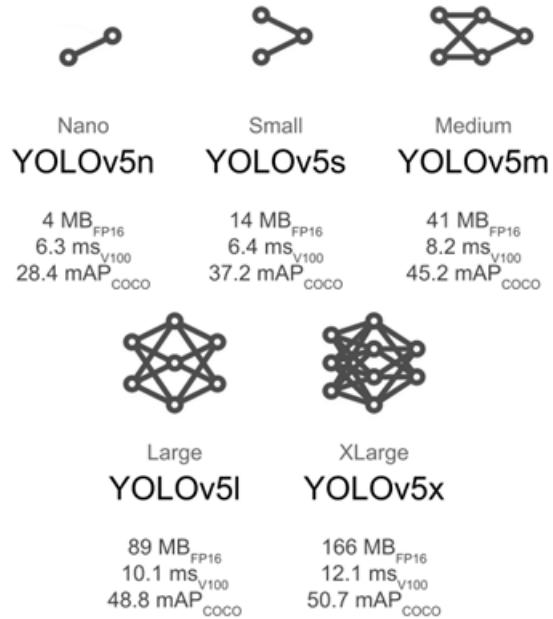


Fig. 1. Dostępne warianty YOLOv5

tego, że sieć jest wstanie za jednym podejściem rozpoznać wiele obiektów. YOLOv5 jest trenowana na pełnych obrazach i bezpośrednio optymalizuje wydajność wykrywania (nie trzeba przekazywać samych zdjęć twarzy).

Model ten ma szereg zalet w porównaniu z innymi metodami wykrywania obiektów.

- YOLOv5 jest bardzo szybkie dzięki czemu używane jest do wykrywania obiektów w czasie rzeczywistym.
- Widzi cały obraz w czasie uczenia i testowania, więc domyślnie koduje kontekstowe informacje o klasach oraz ich wyglądzie.
- Uczy się uogólniających reprezentacji obiektów, dzięki czemu algorytm po przeszkoleniu na naturalnych

obrazach i przetestowaniu na grafice przewyższa inne najlepsze metody wykrywania.

- YOLOv5 rozwiązuje dwa podstawowe a zarazem najważniejsze problemy: gdzie jest obiekt?, co to jest za obiekt? Dzięki czemu stanowi przeźne narzędzie.

YOLOv5 dzieli obraz za pomocą siatki na komórki po czym dla każdej z komórek dokonuje detekcji obiektów. Jeśli obiekt znajduje się w danej jest on zaznaczany prostokątem ( bounding box ), w którym zawiera się obiekt co widać na Fig. 2. dla naszej sieci.



Fig. 2. Oznaczone prostokątami wykryte obiekty.

YOLOv5 pozwala na odtwarzanie filmów z YouTube, kamer bądź filmików. W takiej sytuacji na odtwarzanym obrazie wyświetlane są na bieżąco predykcje.

#### A. Wykorzystane zbiory danych

Wstępnie wybrano do pracy z modelami zbiór Face Mask Detection [1], lecz ze względu na nieproporcjonalny rozkład klas w tym zbiorze zdecydowano się dołączyć dodatkowe zbiory w celu zwiększenia jakości uzyskiwanego modelu. W celu zmniejszenia liczby przypadków fałszywie pozytywnych oraz fałszywie negatywnych dodano zdjęcia tła na których nie występują żadne twarze. Dążąc do wyeliminowania braku proporcjonalności wśród klas, wykorzystano zbiory z twarzami z nieprawidłowo noszonymi maseczkami oraz twarzami bez maseczki.

**1) Zbiór Face Mask Detection:** Zbiór pochodzi z Kaggle [1], wybrano ten zbiór ponieważ posiada on najwięcej pozytywnych ocen dla zbiorów oznaczonych *face mask*. Zbiór składa się z trzech klas (z maską, bez maski i maska noszona nie prawidłowo). Do każdego zdjęcia dołączony jest krótki opis podający klasę, współrzędne oraz rozmiar ramki. Rozkład klas w tym zbiorze danych przedstawia Fig. 3, zauważać można że tylko 3% stanowią etykiety osób z nieprawidłowo założoną maseczką i 17% bez maseczki, zdjęć łącznie w zbiorze jest 853.

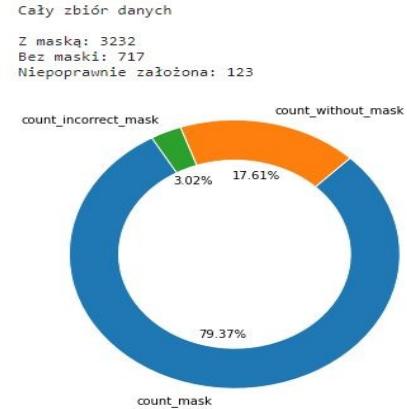


Fig. 3. Procentowy podział klas

**2) Zdjęcia tła:** Wybrano 223 obrazy tła [3] (brak jakichkolwiek twarzy na zdjęciach).

**3) Zbiór ludzi z cyfrowo nieprawidłowo założoną maską:** Niestety w sieci trudno zaleźć zbiór ze zdjęciami twarzy z nieprawidłowo założoną maseczką dlatego utworzono podzbior zawierający 955 zdjęć (955 etykiet) osób ze źle założoną maską ze zbioru MaskedFace-Net. Zbiór ten zawiera zdjęcia osób z maskami noszonymi nieprawidłowo lecz maseczki wygenerowane zostały cyfrowo. [4]. Fig. 4 przedstawia przykładowe zdjęcie z tego zbioru.



Fig. 4. Zdjęcie twarzy z cyfrowo dodaną maseczką

**4) Zbiór zdjęć osób bez masek:** Dodatkowy zbiór osób nie noszących jakichkolwiek masek składający się z 297 zdjęć w tym 1092 etykiet twarzy.

#### B. Przygotowanie danych

Zdjęcia oraz etykiety z wybranych 4 zbiorów połączono w jeden duży zbiór (rozkład klas w zbiorze łącznym przedstawia Fig. 5), następnie podzielono go na zbiory treningowy, walidacyjny oraz testowy w proporcjach 0.6/0.2/0.2 z random\_state=303. Dzięki wystarczającej ilości etykiet z różnych klas wszystkie trzy zbiory posiadały podobny ich rozkład różniący się maksymalnie o 1.8 punktu procentowego.

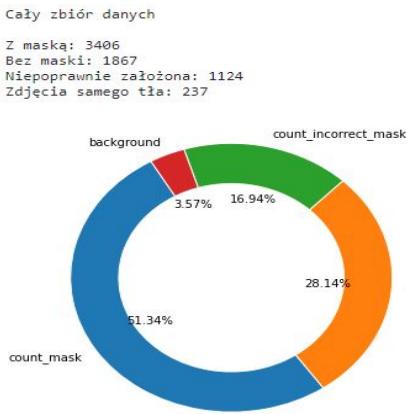


Fig. 5. Procentowy podział klas

### C. Wykorzystane miary jakości modelu

Miary które zostały wykorzystane do badań architektur oraz ich działania z różnymi zbiorami danych:

- mAP@0.5 [6] ("uśrednione pole względem klas dla pól pod krzywymi precision-recall" dla progu IoU = 0.5 )
- mAP@(0.5:0.95) [6] ("uśrednione pole względem klas dla pól pod przywymy precision-recall" dla progów IoU = 0.5, 0.55, ..., 0.9, 0.95)
- Czas treningu
- Wykorzystywana pamięć karty graficznej
- %-obciążenia karty graficznej, zużycie prądu oraz temperatura układu

### D. Procedura treningu

Autorzy biblioteki YOLOv5 zalecają trenować model na 600/1200 epokach z maksymalną liczbą epok bez progresu 100, niestety ze względu na ograniczone możliwości obliczeniowe nie było to w naszym zasięgu, czas 200 epok treningu sieci large dla 4 zbiorów wyniósł około 6 godzin i został zakończony przez wzgląd na 30 epok bez progresu. Niżej wymienione parametry treningu są wyłonione empirycznie i są one najlepsze możliwe jakie były w naszym zasięgu, przez wzgląd na dostępny sprzęt oraz czas. Wybrane przez nas parametry treningu dla modeli:

- liczba epok - 300
- maksymalna liczba epok bez progresu (early stopping) - 30
- hiperparametry - domyślne
- wymiary przetwarzanego zdjęcia - 640x640 px
- batch\_size - 32 (lecz dla wersji z 4 zbiorami, model large 16 ponieważ 32 nie mieściło się w pamięci)

Inne aspekty uczenia architektur YOLOv5:

- Do trenowania, walidacji, testowania oraz wykrywania na zdjęciach/filmach/obrazie - wykorzystywane są dwa progi IoU\_threshold oraz confidence\_threshold
- IoU threshold dla treningu wynosił 0.6, dla walidacji 0.6 i dla detekcji 0.45
- Do walidacji i testowania confidence\_threshold wynosił 0.001, a dla detekcji 0.25

### E. Wykorzystany sprzęt

Do realizacji projektu wykorzystano platformę Kaggle (poza obsługą filmów z youtube oraz kamery), platforma Kaggle udostępnia niżej opisany sprzęt na 30 godzin tygodniowo, do realizacji projektu wykorzystano 8 kont na których równocześnie były uruchomione obliczenia. Łączny czas obliczeń wykorzystując platformę Kaggle to około 300 godzin.

- Intel(R) Xeon(R) CPU @ 2.30GHz 16 rdzeni
- 16GB RAM
- Nvidia TESLA P100 16GB

### F. Wykonane eksperymenty

Każda architektura używana do eksperymentów została wytrenowana wykorzystując odpowiedni zbiór treningowy oraz walidacyjny, a na końcu została przetestowana na zbiorze testowym. Ze względu na zasoby sprzętowe przetestowaliśmy cztery architektury YOLOv5: Nano, Small, Medium oraz Large, niestety wersja Extra-Large była poza naszym zasięgiem. Wykorzystano wersje operujące na rozdzielcości 640x640 px. Przeprowadzono takie eksperymenty jak:

- Ocena jakości architektur dla zbioru Face Mask Detection.
- Ocena jakości architektur dla zbioru Face Mask Detection wzbogaconego o obrazy tła.
- Ocena jakości architektur dla połączonych 4 zbiorów.
- Ocena jakości najlepszego modelu (według mAP) wytrenowanego na połączonych 4 zbiorach na zbiorze Face Mask Detection.
- Testy działania najlepszego modelu (według mAP) dla zdjęć, filmów oraz obrazu z kamery.
- Testy działania najlepszego modelu (według mAP) dla przypadków ekstremalnych/skrajnych w których będzie umieszczone wiele twarzy na zdjęciu, cięższe warunki oświetleniowe.
- Próby oszukania modelu poprzez dorysowanie różnych elementów na maseczce/twarzie.
- Badanie zależności między wielkością modelu i wybranym zbiorem danych, a intensywnością wykorzystywania zasobów obliczeniowych.

## II. WYNIKI

W tym rozdziale przedstawiono wyniki dla wszystkich modeli i zestawów zbiorów.

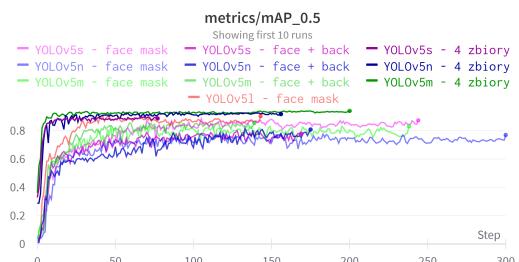


Fig. 6. Wyniki mAP 0.5 dla wszystkich modeli

## A. YOLOv5 dla Face Mask Detection

Wyniki uzyskane dla pojedynczego zbioru są niewystarczające, co można zobaczyć na Fig. 7. nawet YOLOv5l (największy model) uzyskało znaczny odsetek fałszywych negatywnych Fig. 8. Próbą rozwiązania tego problemu było dodanie obrazów, które nie przedstawiają żadnych twarz - tzw. zdjęcia tła. Dla YOLOv5 autorzy zalecają aby od 3% do 10% danych przedstawało wyłącznie tło [1].

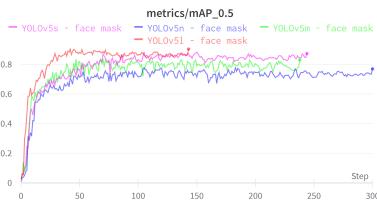


Fig. 7. Wyniki mAP 0.5 dla zbioru Face Mask Detection

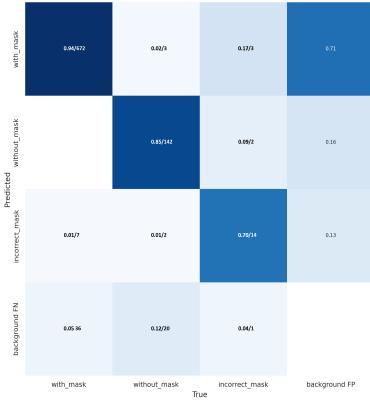


Fig. 8. Macierz pomyłek dla YOLOv5l dla Face Mask Detection, wartości procentowe i liczba przypadków dla danej predykcji

## B. YOLOv5 dla Face Mask Detection + background

Dzięki użyciu obrazów przedstawiających wyłącznie tło ilość fałszywych pozytywnych zmniejszyła się Fig. 9, model rzadziej myli posiadanie maski z tłem. Niestety liczba prawidłowo wykrywanych niepoprawnie założonych masek również się zmniejszyła przez co wyniki nie są jednoznaczne. Przez to, że wyniki FP nie są zapisywane, trudno jest stwierdzić wyłącznie na podstawie porównania obu macierzy pomyłek jak zachowywało się FP, dlatego użyto precyzji. Na Fig 10. widać, że precyzja nie zwiększyła się.

Obawy które powstały analizując obie macierze pomyłek i porównania precyzji dla treningu na 2 zbiorach sprawdziły się, nie widać wzrostu w mAP 0.5 Fig. 11.

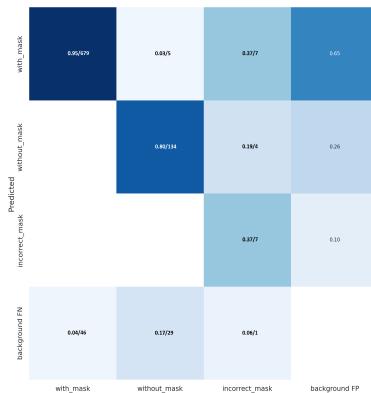


Fig. 9. Macierz pomyłek dla YOLOv5l dla Face Mask Detection + background, wartości procentowe i liczba przypadków dla danej predykcji



Fig. 10. Porównanie precyzji dla obu modeli YOLOv5l

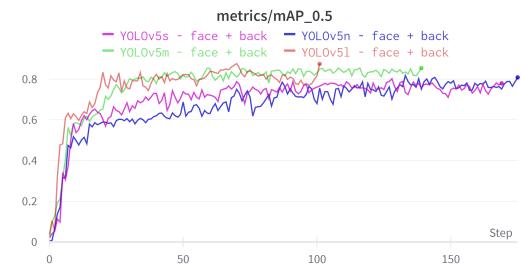


Fig. 11. Wyniki mAP 0.5 dla Face Mask Detection + background

## C. YOLOv5 dla połączonych 4 zbiorów

Po połączeniu czterech zbiorów uzyskano zdecydowanie najwyższe wyniki mAP 0.5 Fig. 12.

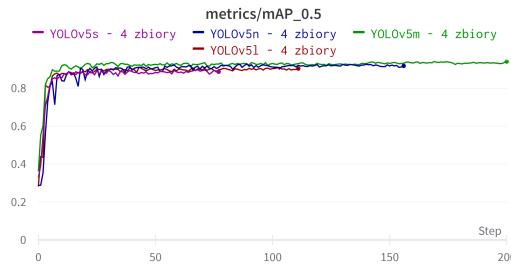


Fig. 12. Wyniki mAP 0.5 dla 4 zbiorów

#### D. Najlepszy wytrenowany model przetestowany na podstawowym zbiorze danych

Wykorzystując najlepszy wytrenowany model (najwyższy mAP0.5:0.95 ze wszystkich) na połączonych 4 zbiorach do przetestowania na zbiorze Face Mask Detection udało się drastycznie przewyższyć wyniki architektur trenowanych wyłącznie na podstawowym zbiorze. Najwyższy wynik to 0.625 mAP0.5:0.95 architektury large, zaś wykorzystując najlepszy model wytrenowany na największym zbiorze danych uzyskano wynik 0.743 mAP0.5:0.95 Fig. 13.

| Class          | Images | Labels | P     | R     | mAP@.5 mAP@ |
|----------------|--------|--------|-------|-------|-------------|
| all            | 853    | 4072   | 0.971 | 0.89  | 0.923       |
| with_mask      | 853    | 3232   | 0.983 | 0.958 | 0.982       |
| without_mask   | 853    | 717    | 0.97  | 0.936 | 0.964       |
| incorrect_mask | 853    | 123    | 0.96  | 0.777 | 0.824       |
|                |        |        |       |       | 0.673       |

Fig. 13. Wyniki mAP 0.5 dla 4 zbiorów

#### E. YOLOv5 dla zdjęć, filmów oraz kamera

YOLOv5 poprawnie przewiduje klasy we wszystkich 3 przypadkach (choć niestety czasem się myli), działanie z filmem oraz kamerą zostanie zaprezentowane podczas konsultacji na teams.

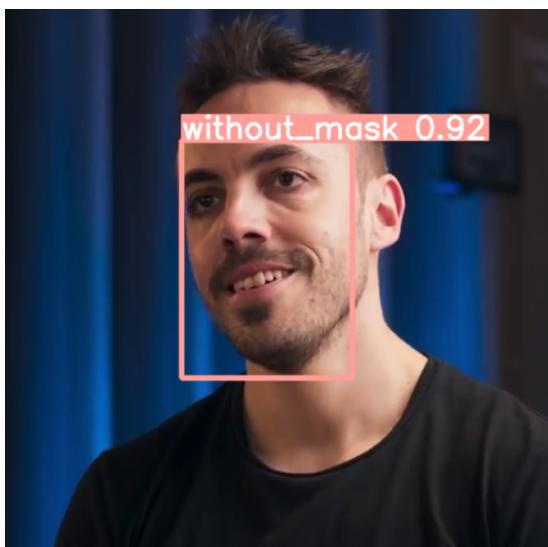


Fig. 14. Przykład dla osoby bez maski.



Fig. 15. Osoby w maskach



Fig. 16. Różne warianty noszenia maseczki na jednym zdjęciu

#### F. Skrajne przypadki oraz próby oszukania modelu

Fig. 17 oraz Fig. 18 pokazuje że model dobrze sobie radzi ze złymi warunkami oświetleniowymi w wykrywaniu założonej maseczki, niestety wykrywanie czy maseczka jest poprawnie założona nie wychodzi mu już tak dobrze. Niestety brak jest zbioru z dużą liczbą zdjęć w złych warunkach oświetleniowych aby obiektywnie ocenić działanie w takich przypadkach. Fig. 19 pokazuje że model świetnie radzi sobie z dużą liczbą twarzy na zdjęciu, ze względu ograniczeń widoczności przykład z jeszcze większą liczbą twarzy zostanie pokazany na teamsach podczas prezentacji projektu.



Fig. 17. Różne warianty noszenia maseczki na jednym zdjęciu

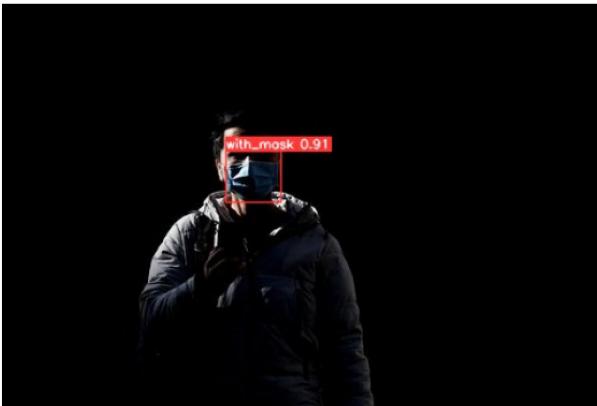


Fig. 18. Różne warianty noszenia maseczki na jednym zdjęciu



Fig. 19. Różne warianty noszenia maseczki na jednym zdjęciu



Fig. 20. Wariant noszenia maseczki z profili nr.1

Na zdjęciu z wariantu pierwszego możemy zauważać grupę osób z maseczkami w trzech różnych kolorach, w tym również jedną osobę w czarnej masce z profilu która jednak została sklasyfikowana poprawnie z dość wysoką pewnością.



Fig. 21. Wariant noszenia maseczki z profili nr.2

Wariant nr.2 przedstawia różne kolory masek, w tle widać osobę z maską koloru czarnego która jest wykryta prawidłowo niestety pewność modelu nie jest wysoka.



Fig. 22. Wariant noszenia maseczki z profili nr.3

Na rysunku z wariantem nr.2 można zauważać że model prawidłowo rozpoznał osobę w masce, lecz jego pewność nie jest bardzo wysoka pomimo wysokiej rozdzielczości.



Fig. 23. Wariant noszenia maseczki z profiliu nr.4

Wariant czwarty przedstawia błąd wykrywania nieprawidłowo założonej maseczki koloru białego z profilu (model uznał że osoba maski nie nosi), jednocześnie model prawie w tym samym kwadracie wykrywa osobę z nieprawidłowo założoną maseczką.



Fig. 24. Wariant noszenia maseczki z profiliu nr.5

W wariantie nr.5 można zauważyć że maseczki widoczne z profilu koloru niebiesko-białego w dali zostały wykryte nawet z daleka, zaś koloru zielono-czarnego niestety już nie.

#### G. Próba oszukania modelu

W pewnym momencie udało się oszukać model, pierw próbowało na maseczce dorysować znaczki, następnie zmienić jej kolor a na koniec dodać paski idące od okularów do maseczki.



Fig. 25. Maska założona nieprawidłowo



Fig. 26. Predykcja



Fig. 27. Predykcja dla maski z dorysowaniami paskami



Fig. 29. Predykcja dla inaczej zamazanej maski



Fig. 28. Predykcja dla zamazanej maski

#### H. Aspekty techniczne treningu modeli

Na rysunkach poniżej można zauważać takie aspekty jak:

- czas treningu dla sieci wahał się od półtora do blisko sześciu
- temperatura, zużycie prądu, zasobów czy pamięci karty graficznej bardziej zdaje się zależeć od wielkości modelu niż wielkości zbioru danych
- zużycie procesora przez cały trening pozostaje stałe

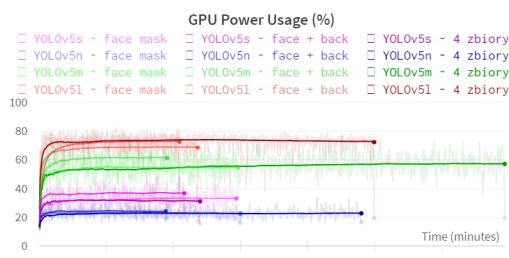


Fig. 30. Wykorzystanie mocy przez GPU (%)

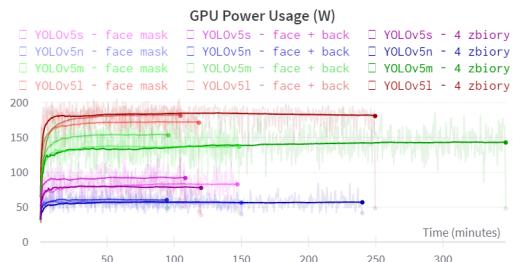


Fig. 31. Wykorzystanie mocy przez GPU (W)

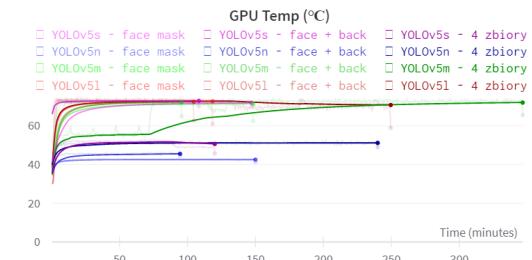


Fig. 32. Temperatura GPU

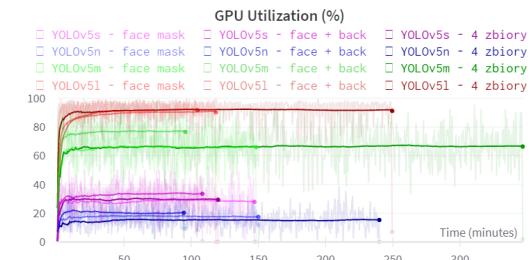


Fig. 33. Wykorzystanie GPU

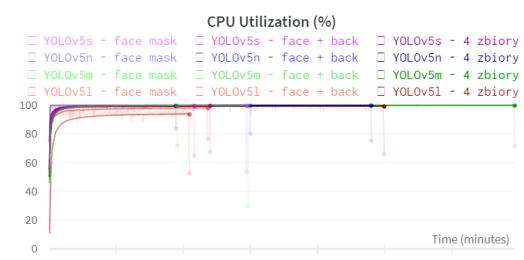


Fig. 34. Wykorzystanie CPU

### III. WNIOSKI

Rozdział przedstawia wnioski jaki wysnuto z otrzymanych wyników.

1) *Porównanie wielkości zbiorów:* Dodając większą liczbę zbiorów zaobserwowano wzrost jakości dla każdego modelu. Dla łatwiejszego rozróżnienia, ciemniejszy kolor oznacza

większą liczbę zbiorów użytych do trenowania. Można zauważać, że po pewnej liczbie epok, modele ustabilizowały się i nie następowała dalsza poprawa jakości.

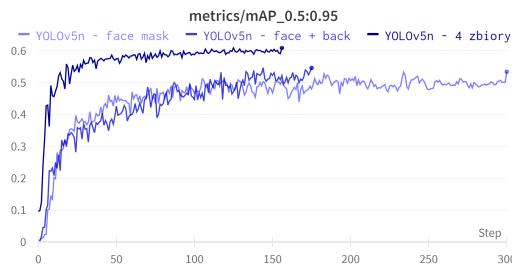


Fig. 35. Wyniki mAP 0.95 modeli YOLOvn

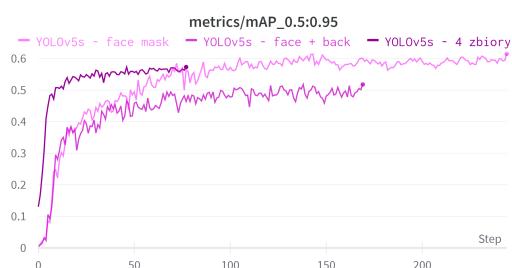


Fig. 36. Wyniki mAP 0.95 modeli YOLOvs

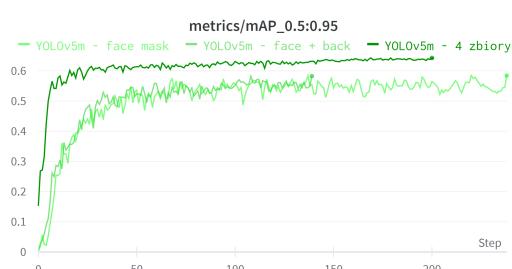


Fig. 37. Wyniki mAP 0.95 modeli YOLOvm

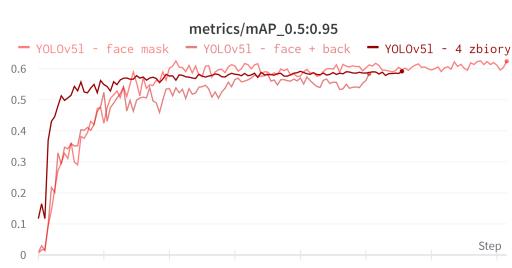


Fig. 38. Wyniki mAP 0.95 modeli YOLOv1

2) Porównanie wielkości modeli: Wykorzystując różne modele i różne zbiorы danych nie zawsze zauważano aby

największy model uzyskiwał najlepsze rezultaty fig. 39. Może to być spowodowane szybkim wpadaniem w nadmierne dopasowanie i uczenie się zbioru trenującego na pamięć. Problemem może być mały zbiór danych - twórcy YOLOv5 zalecają 10tyś. etykiet na klasę oraz 1500 zdjęć na klasę, brak stabilności obliczeniowej (niestety nie mamy możliwości sprawdzenia tego przez ograniczenia obliczeniowe), nieodpowiedni dobór hiperparametrów. Dodatkowo można zauważać, że na wybranych zbiorach danych model medium osiąga podobne lub czasem nawet lepsze wyniki od large, pomimo tego że wydaje się być o 40% 'lżejszy' dla karty graficznej. W celu lepszego zbadania przyczyny oraz tego zawsze tak jest, należałoby wykonać uczenie sieci w liczbie epok którą podają twórcy biblioteki tj. 600/1200, oraz ustawić maksymalną liczbę epok bez progresu na 100/150 zgodnie z zaleceniem autorów. Jest to spowodowane tym że architektura YOLO często przez wiele iteracji nie robi postępów, lecz po przykładowo 60 epokach nagle może dać lepszy rezultat niż wszelkie wcześniejsze, potem to samo za 20 epok lub w przeciwną stronę - 120.

3) Wpływ na predykcje zdjęć przedstawiających tylko tła:  
 Okazało się, że mimo zmniejszania się liczb fałszych pozytywnych ogólna jakość predykcji pogorszyła się po dodaniu zdjęć przedstawiających tylko tła dla niektórych modeli a dla innych delikatnie się poprawiła. Może to być spowodowane zbyt małą liczbą zdjęć tła (3% zbioru danych, gdzie twórcy zalecają od 3% do 10%), zbyt małym zbiorom danych lub pomyłkami w zbiorze danych (możliwe że niektóre zdjęcia zawierają twarze lecz nie posiadają odpowiednich adnotacji). Możliwe, że trening dla większej liczby epok poprawiłby wyniki predykcji ponieważ autorzy zalecają trening przez 2/4 krotnie większa ilość epok z 3-4 razy większą możliwą liczbą epok bez progresu. Istnieje możliwość że po pewnym czasie wyniki dla poszczególnych modeli jeszcze by urosły, lecz wymagałoby to dłuższych obliczeń na które nie mogliśmy sobie pozwolić. Kolejną przyczyną braku poprawy może być różniące się oznaczanie ramek dla zbiorów np. zbiorów mogą zakreślać inny zakres twarzy.

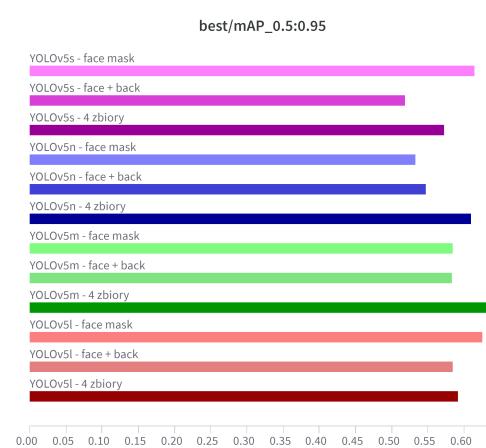


Fig. 39. Najlepsze wyniki mAP 0.95 dla modeli

4) *Potrzebne zasoby do pracy z modelem:* Pomimo tego że zbiory danych były niewielkie wykorzystywana pamięć karty graficznej sięgała prawie 16GB - na rynku nie jest zbyt wiele kart które są w stanie obsłużyć tak duże modele, a te które są w stanie są bardzo drogie. W pewnym momencie musieliśmy zejść o połowę z batch size, jeśli chcielibyśmy w projekcie analizować kilkakrotnie większy zbiór danych w celu uzyskaniawiększej jakości potrzebowalibyśmy kilku kart przez zapotrzebowanie na pamięć oraz szybkość samego procesu uczenia. W projekcie wykorzystaliśmy około 300h obliczeń, średnio na jedną godzinę wykorzystywane było 200W mocy, co daje około 60kWh prądu, można przeliczyć że koszt wykonania obliczeń do projektu wyniósł około 50zł. Dzięki platformie Kaggle udało nam się to zrealizować za darmo, lecz niestety niezgodnie z regulaminem ponieważ Kaggle pozwala utworzyć tylko jedno konto na użytkownika a my wykorzystywaliśmy po 4 na osobę. Dzięki platformie Kaggle oraz wandb.ai mogliśmy wspólnie w czasie rzeczywistym pracować wspólnie nad programem czy też obserwować jak uczą się modele i na bieżąco korygować błędy oraz wyciągać wnioski.

5) *Wpływ artefaktów/zmiany wyglądu maski na predykcje:* Jeśli używane maski nie odbiegają wyglądem od tych z używanych zbiorów predykcje są zadowalające. Jednak gdy maski mają inny wygląd nawet najlepszy uzyskany model nie radzi sobie już tak dobrze - nie ugólnia wystarczająco dobrze. W celu rozwiązania tego problemu - uzyskania lepszego uogólniania należałoby powiększyć zbiór danych o zdjęcia ludzi w maskach różnych kolorów/rodzajów, dobrze założonych/źle założonych/z różnych profili czy też zasłaniającym je dłońmi lub przedmiotami.

#### IV. PODSUMOWANIE

Podsumowując prace nad projektem i nabycie doświadczeń:

- Wybór większych modeli generalnie poprawia jakość predykcji.
- Ważne jest aby używać jak największych oraz jak najbardziej zrównoważonych zbiorów ponieważ podnosi to znacząco wyniki.
- Wszystkie modele YOLOv5 generują stabilne wyniki po odpowiednio długim treningu.
- YOLOv5 należy uruchamiać na wiele epok, domyślne wartości (300) wraz z *patience* (100) pozwalają znaleźć optymalne parametry.
- YOLOv5 jest w stanie pracować ze zdjęciami, filmami (z dysku lub z linkami np. do YouTube) czy obrazem z kamery.
- Udało się stworzyć aplikację która wykonuje swoje zadanie, lecz aby działała jeszcze lepiej potrzeba lepszego zbioru danych, lepszego sprzętu do trenowania oraz można próbować dostrajać hiperparametry.

#### A. Możliwe dalsze kierunki rozwoju:

- Testy na zbiorach zawierające różnorakie maski np. kolorowe, mające wzory.

- Zdobycie większych zbiorów danych dla wszystkich 3 klas, w szczególności z nieoprawnie założonymi maskami z życia zamiast wygenerowanego cyfrowo.
- Trening sieci na większej liczbie epok dla połączonych wszystkich czterech zbiorów.
- Wykorzystanie destylacji wiedzy w celu ograniczenia wielkości modelu zachowując jak najwięcej jakości.
- Przetestowanie odporności modeli na ataki przeciwstawne (white box lub black box).

#### BIBLIOGRAFIA

- [1] <https://www.kaggle.com/datasets/andrewmvd/face-mask-detection>
- [2] <https://github.com/ultralytics/yolov5>
- [3] <https://www.kaggle.com/datasets/yodazmc/myunder>
- [4] <https://www.kaggle.com/datasets/yodazmc/yolox1>
- [5] <https://www.kaggle.com/datasets/yodazmc/yolox2>
- [6] <https://www.v7labs.com/blog/mean-average-precision>