Mariusz Wiśniewski KD-712

Wydział Geodezji i Kartografii PW

Podstawy rozpoznawania obrazów

4606-PS-00000CJ-C001

Agnieszka Jastrzębska

Raport

Poprawa detekcji twarzy i obecności maski na twarzy

**Wstęp**

Konwolucyjne sieci neuronowe, w skrócie CNN (ang. Convolutional Neural Network), doskonale radzą sobie z rozpoznawaniem obrazów. CNN to algorytm, który może obraz wejściowy sklasyfikować wedle predefiniowanych kategorii. Jest to możliwe dzięki przypisaniu wag różnym kształtom, strukturom, obiektom. Sieci konwolucyjne poprzez trening są w stanie nauczyć się, jakie cechy szczególne obrazu pomagają w jego klasyfikacji. Ich przewagą nad standardowymi sieciami głębokimi jest większa skuteczność w wykrywaniu zawiłych zależności w obrazach. Jest to możliwe dzięki zastosowaniu filtrów badających zależności pomiędzy sąsiednimi pikselami.

Obraz zawierający szkic, diagram, rysowanie, linia

Opis wygenerowany automatycznie

Zadaniem warstwy konwolucyjnej, w przypadku pierwszej warstwy, jest wyodrębnienie cech, takich jak krawędzie, kolory, gradienty. Warstwy konwolucyjne stosują filtry do skanowania lokalnych obszarów danych wejściowych. Filtr jest przesuwany przez dane, a wynik operacji konwolucji otrzymywany jest dla każdego położenia filtra. W wyniku tego procesu powstaje mapa cech, która zawiera informacje o występowaniu poszczególnych cech na obrazie. Kolejne warstwy sieci, korzystając z tego, co wyznaczyły poprzednie warstwy, mogą wykrywać coraz bardziej skomplikowane kształty. Po warstwie konwolucyjnej występuje warstwa aktywacyjna, wprowadzająca nieliniowość do sieci. Zazwyczaj stosuje się wiele filtrów, przez co warstwa konwolucyjna, zwiększa głębokość, czyli liczbę kanałów obrazu. Kolejna warstwa, nazywana łączącą, ma za zadanie zmniejszenie pozostałych wymiarów obrazu, przy zachowaniu kluczowych informacji potrzebnych np. do klasyfikacji obrazu. Sieć może stosować inne warstwy, takie jak warstwy łączące (pooling), które redukują rozmiar mapy cech poprzez wybór najważniejszych informacji. To pozwala na zmniejszenie liczby parametrów sieci i zwiększenie efektywności uczenia.  Po zastosowaniu wybranej ilości warstw otrzymana macierz zostaje „spłaszczona” do jednego wymiaru. „Dense Layer” pozwala na nauczenie algorytmu nieliniowych zależności pomiędzy cechami wyznaczonymi przez warstwy konwolucyjne. Ostatnią warstwą sieci, pozwalająca na uzyskanie wartości prawdopodobieństw przynależności do poszczególnych klas.

Podczas uczenia CNN sieć neuronowa jest zasilana dużym zbiorem danych obrazów oznaczonych odpowiednimi etykietami klas Podczas treningu CNN przechodzi przez kilka serii epok, dostosowując wagę do wymaganych niewielkich ilości. Po każdym kroku epoki sieć neuronowa staje się nieco dokładniejsza w klasyfikowaniu i poprawnym przewidywaniu klasy obrazów treningowych. W miarę ulepszania się CNN korekty dokonywane w wagach stają się odpowiednio coraz mniejsze.

Po przeszkoleniu CNN używamy testowego zbioru danych, aby zweryfikować jego dokładność. Testowy zestaw danych to zestaw oznaczonych obrazów, które nie zostały uwzględnione w procesie uczenia. Każdy obraz jest przesyłany do CNN, a wynik jest porównywany z rzeczywistą etykietą klasy obrazu testowego.

Otrzymałem pakiet zawierający program do wyszkolenia modelu IncepctionV3, program wykorzystujący ten model do określania czy osoby widoczne w sygnale z kamery internetowej noszą maskę czy nie oraz zestaw obrazów do szkolenia i testowania modelu.

Celem pracy jest poprawienie istniejącego programu, modelu lub zaproponowanie innego rozwiązania pozwalającego uzyskać lepszą skuteczność w wykrywaniu i klasyfikacji twarzy pod względem występowania na niej maski ochronnej a następnie przygotowanie metody umożliwiającej wykazanie lepszego działania nowej metody.

1. **Zestaw danych do szkolenia modelu klasyfikującego twarze**

Otrzymany zestaw twarzy składał się z katalogu 3276 obrazów do trenowania oraz 818 obrazów do walidacji, razem 4096 obrazów. Każdy z katalogów podzielony został na dwa katalogi zawierające wyłącznie obrazy twarzy z maską lub bez maski. Zbiór obrazów nie zawierał katalogu przeznaczonego do testowania oprogramowania.

Wykonałem poszukiwania większych zbiorów obrazów z podziałem na twarze z i bez maski. Udało mi się pobrać taki katalog z adresu:

<https://www.kaggle.com/datasets/omkargurav/face-mask-dataset/data>

Znaleziony zestaw 7553 obrazów podzieliłem losowo na 5551 obrazów przeznaczonych do szkolenia modeli, 924 obrazów do walidacji w czasie szkolenia oraz 1078 obrazów do testowania wyniku szkolenia.

Dodatkowo udało mi się znaleźć zestaw 853 obrazów z oznaczonymi 4072 położeniami twarzy z podziałem na twarze bez maski, z maskami oraz z maskami nieprawidłowo założonymi. 79% próbki stanowią twarze z maską.

Zestaw obrazów pobrałem z adresu:

<https://datasetninja.com/face-mask-detection#download>

Dane te zostaną wykorzystane do szkolenia modelu opartego na detekcji zamiast na klasyfikacji.

Przeprowadzone zostanie szkolenie modeli, wybór najlepszych do badań, wykorzystanie modeli do pracy z kamerą internetową, wykonane zostanie badanie skuteczności detekcji w oparciu o symulowane obrazy i wskazany zostanie najskuteczniejszy z zaproponowanych programów.

Celem raportu nie jest opisywanie detali w kodzie napisanym w języku Python.

1. **Szkolenie modeli klasyfikujących**

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznieSzkolenie modelu na podstawie otrzymanego programu.

Program został zmodyfikowany tak by wyświetlać historię szkolenia oraz wykonywać ewaluację wyników na koniec na podstawie katalogu z obrazkami walidacyjnymi ze względu na brak wydzielonej części testowej.

Uzyskana została dokładność 99.3% przy błędach na poziomie 2.2%

Wynikowy model stosowany będzie dalej jako referencyjny by wykazać poprawienie działania otrzymanego programu.

Wynik: IncepctionV3\_0

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznie

Analiza kodu wykazała, że funkcja wywołująca uczenie modelu bierze w każdej epoce tylko 12 obrazków. Usunięcie tego błędu sprawiło, że model mógł się uczyć na podstawie wszystkich obrazków z pierwotnej bazie obrazków.

Dzięki tej zmianie udało się uzyskać dokładność ewaluacji na zbiorze testowym na poziomie 99.7% przy błędach na poziomie 0.9%

Wyniki: IncepctionV3\_0p

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznie

Zbiór obrazów do uczenia został zastąpiony większym znalezionym w internecie. Nowy zbiór podzielony był nie tylko na część treningową i walidacyjną ale również oddzielną testową.

W wyniku ponownego szkolenia uzyskano dokładność na poziomie 99.0% przy błędach na poziomie 5.3%. Gorszy wynik może być spowodowany tym, że wcześniejsza ewaluacja była przeprowadzona nieprawidłowo, bo na zbiorze walidacyjnym zamiast wydzielonym testowym.

Wynik: IncepctionV3\_0b

Obraz zawierający diagram, tekst, linia, Wykres

Opis wygenerowany automatycznieWykonałem szereg prób modyfikacji zaproponowanego modelu. Próbowałem obcinać przetrenowany model w innym miejscu oraz stosować inne dodatkowe warstwy i funkcje aktywacyjne. Większość prób prowadziła do wyników o gorszej dokładności. Dla jednego przypadku udało się uzyskać zadowalający wynik. Zaproponowany model zachował wszystkie pierwotne warstwy oraz dodatkowe, a 32 ostatnie warstwy zostały ustawione na douczanie.

Udało się uzyskać dokładność na poziomie 99.7% przy błędach na poziomie 3.5%.

Wynik: IncepctionV3\_nowymod2\_big\_01

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznie

Podjąłem próbę zastosowania zupełnie innego przeszkolonego modelu. Do testów wybrałem MobileNetV2. Doszkalanie modelu z wykorzystaniem dodatkowych warstw jak poprzednio, z zablokowaniem modelu przed uczeniem pozwoliło na uzyskanie dokładności na poziomie 98.7% przy błędach na poziomie 3.1%

Ten wyszkolony model został wykorzystany w dalszych badaniach.

Wynik: MobileNetV2\_nowymod1\_01

Próby modyfikacji tego modelu i szkolenia części warstw nie doprowadziły do lepszych wyników.

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznieZe względu na ograniczenia w dostępie do komputera dysponującego szybkim procesorem lub kartą graficzną (darmowy pakiet na Google Colab wykorzystałem w jeden dzień) uczenie wykonywane było tylko dla 5 epok. Wykonałem jedno szkolenie podstawowego podstawowego modelu trwające 50 epok. Szkolenie trwało około 12 godzin. Uzyskałem w ten sposób dokładność na poziomie 99.5% oraz błąd na poziomie 3.7%.

Ponieważ, model ten był uczony najdłużej ze wszystkich został wykorzystany w dalszych badaniach.

Wynik: IncepctionV3\_0\_nowytest\_02

1. **Szkolenie modelu detektora**

Ponieważ udało się znaleźć gotowy zestaw obrazów twarzy z maską i bez wraz z plikami z zaznaczeniem miejsc występowania tych kategorii mogłem zastosować odmienne podejście do wykrywania twarzy z maską lub bez pomijające etap wyszukiwania jedynie twarzy, a przeprowadzający detekcję od razu twarzy z maską lub twarzy bez maski.

Wykorzystany zestaw składał się z 853 obrazów. Zastosowałem model SSD\_MoblileNetV2\_FPNLite, który sprawdził mi się w innym realizowanym projekcie. Szkolenie modelu przeprowadzone zostało w środowisku Google Colab dla 15 tysięcy epok. Kopia skryptu do szkolenia modelu znajduje się w katalogu z programami. W wersji gotowej do użycia jest dostępny pod adresem:

https://colab.research.google.com/github/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/blob/master/Train\_TFLite2\_Object\_Detction\_Model.ipynb

Obraz zawierający tekst, diagram, Wykres, linia

Opis wygenerowany automatycznie

Model został pobrany i uruchomiony jako lokalny detektor.

Nie spisałem wyników przed zamknięciem środowiska i nie znam końcowych statystyk.

1. **Modele do dalszego testowania**

Do dalszego testowania wybrane zostały 4 modele:

1. IncepctionV3 – pierwotny źle przeszkolony model na mniejszym zestawie obrazów szkolących
2. IncepctionV3 – ten sam model ale przeszkolony przez 50 epok na pełnym większym zestawie obrazów szkolących
3. MobileNetV2 – model przeszkolony przez 5 epok na większym zestawie obrazów szkolących
4. SSD\_MoblileNetV2\_FPNLite – model przeszkolony przez 15 tysięcy epok na zestawie 2559 opisanych obrazów.

Trzy pierwsze modele są klasyfikatorami. Wymagają wstępnego znalezienia twarzy na obrazie. Ostatni model sam znajduje twarze na obrazie.

1. **Wykrywanie twarzy na obrazie z kamery internetowej**

**real\_time\_v1.py**

Otrzymany program do wykrycia twarzy na obrazie wykorzystywał kaskady haara (haar cascades). Informacja o parametrach wszystkich znalezionych twarzy trafia do zmiennej. Następnie po kolei przeprowadzane jest wycinanie twarzy z obrazu wejściowego i przekazywane do klasyfikatora pracującego na modelu 1.Program rysuje prostokąt w miejscu wykrytej twarzy informując o rozpoznanej kategorii.

**real\_time\_v2.py i real\_time\_v3.py**

Dokonałem modyfikacji pierwotnego programu po przez zastosowanie biblioteki cvzone zawierającej przeszkolony model wykrywający twarze na obrazie. Funkcja findFaces modułu FaceDetector również zwraca zmienną zawierającą informację o wszystkich wykrytych twarzach. Na tej podstawie wycinane są twarze z wejściowego obrazu i przekazywane do klasyfikatora. Program rysuje prostokąt w miejscu wykrytej twarzy informując o rozpoznanej kategorii.

Program został przygotowany do pracy z modelami 2 i 3 wytypowanymi do dalszych badań. Może również pracować z modelem 1.

**real\_time\_TFL.py**

Trzeci program pozwala na sprawdzenie w działaniu modelu 4, przeprowadzającego detekcję twarzy z maską lub bez. Program rysuje prostokąt w miejscu wykrytej twarzy informując o rozpoznanej kategorii.

1. **Testowanie przygotowanego oprogramowania**

Testowaniu poddane zostaną cztery programy wykrywające i rozpoznające twarze w maskach lub bez. Celem eksperymentu jest określenie skuteczności działania całego oprogramowania zawierającego zarówno detektor twarzy jak i klasyfikator obecności maski na twarzy.

1. IncepctionV3 – program z detektorem twarzy opartym o kaskadę haara i klasyfikator oparty o model IncepctionV3, szkolony pierwotną wersją programu szkolącego na mniejszym zestawie obrazów.
2. IncepctionV3 + cvzone – porgram z detekcją twarzy opartą o bibliotekę cvzone i model IncepctionV3 przeszkolony na dużym zbiorze obrazów przez 50 epok.
3. MobileNetV2 + cvzone - porgram z detekcją twarzy opartą o bibliotekę cvzone i model MobileNetV2 przeszkolony na dużym zbiorze obrazów przez 5 epok.
4. SSD\_MoblileNetV2\_FPNLite – program z detekcją opartą o model detektora SSD\_MoblileNetV2\_FPNLite przeszolony na mniejszym zbiorze obrazów przez 15 tysięcy epok.

Programy te odpowiadają czterem programom real\_time\_v1.py real\_time\_v2.py real\_time\_v3.py real\_time\_TFL.py przygotowanym do pracy z kamerami internetowymi.

Programy zostały zintegrowane z generatorem obrazów symulujących pracę w warunkach naturalnych, który będzie opisany w dalszej części. Programy test\_model\_v1.py test\_model\_v2.py test\_model\_v3.py test\_model\_TFL.py testują wylistowane powyżej rozwiązania i modele.

Wyniki detekcji i klasyfikacji trafiają do plików wynikowych by możliwa była analiza rzeczywistej skuteczności działania oprogramowania.

1. **Tworzenie symulowanych warunków naturalnych do testowania rzeczywistej skuteczności oprogramowania.**

Skuteczność rozpoznawania twarzy i określania czy są one zasłonięte maską czy nie zbadana zostanie po przez wygenerowania sztucznych obrazów testowych.

Z katalogu obrazów testowych wybrane zostały zdjęcia twarzy, prawidłowo naświetlone, o dobrej rozdzielczości i ostrości, o zbliżonych proporcjach, przedstawiających głowy osób wypełniające kadr w zbliżony sposób bez pozostawiania nadmiarowej wolnej przestrzeni. Przykładowe twarze zostały wybrane tak by skierowane były raczej w stronę kamery, reprezentowały różne karnacje i kolory masek. Preferowane były obraz z różnorodnym tłem.

Obraz zawierający ubrania, osoba, maska, człowiek

Opis wygenerowany automatycznieObraz zawierający osoba, maska, plastik, dziewczyna

Opis wygenerowany automatycznieObraz zawierający ubrania, osoba, maska, nosić

Opis wygenerowany automatycznieObraz zawierający ubrania, osoba, Ludzka twarz, nosić

Opis wygenerowany automatycznieObraz zawierający medyczny, Ludzka twarz, Sprzęt medyczny, służba zdrowia

Opis wygenerowany automatycznie

Obraz zawierający Ludzka twarz, osoba, brew, rzęsa

Opis wygenerowany automatycznie Obraz zawierający Ludzka twarz, osoba, ubrania, Podbródek

Opis wygenerowany automatycznie Obraz zawierający Ludzka twarz, osoba, człowiek, brew

Opis wygenerowany automatycznie Obraz zawierający osoba, Ludzka twarz, uśmiech, Warga

Opis wygenerowany automatycznie Obraz zawierający Ludzka twarz, osoba, brew, Podbródek

Opis wygenerowany automatycznie

Wyselekcjonowane zostało 56 twarz w masce i 55 twarzy bez maski. Razem 111.

Obrazy twarzy poddawane były trzem modyfikacjom:

1. Zmiana rozmiaru obrazka z twarzą w zakresie od 25 do 275 pikseli, z krokiem 25.
2. Zmiana jasności obrazka w zakresie od pociemnienia o 150 poziomów do rozjaśnienia o 100, z krokiem 50, reprezentująca zakres od dużego niedoświetlenia do dużego prześwietlenia
3. Dodanie kolorowego szumu o odchyleniu standardowym w zakresie od 0 do 125 z krokiem 25.

Zmodyfikowany obrazek z twarzą nanoszony był w losowe miejsce obrazka z tłem przedstawiającym ulicę. Szum opisany w punkcie 3 dodawany był na całość już po naniesieniu obrazka z twarzą po dwóch pierwszych modyfikacjach. W ten sposób otrzymaliśmy obrazki testowe o 11 różnych rozmiarach twarzy, 6 różnych poziomach jasności i 6 różnych poziomach zaszumienia. Razem 396 przypadków.

Obraz zawierający na wolnym powietrzu, drzewo, niebo, droga

Opis wygenerowany automatycznie

Obrazek wykorzystany jako tło dla nanoszonych twarzy.

1. **Analiza skuteczności programów w wykrywaniu twarzy**

Obraz zawierający Wielobarwność, zrzut ekranu, tekst, Prostokąt

Opis wygenerowany automatycznie

**8.1 Skuteczność programów przy braku szumów**

Program InceptionV3 wykazał się bardzo słabym średnim poziomem wykrywania twarzy. Przyczyna tego stanu wynika z tego, że bardzo dobrze poradził somie z wykrywaniem twarzy bez maski, nawet dla bardzo małych rozmiarów w pełnym przedziale jasności, ale zawodził gdy twarz była przysłoniona maską. Zaledwie jedna trzecia twarzy zakrytych maską została rozpoznana jako twarz człowieka.

Programy InceptionV3\_cvzone i MobileNetV\_cvzone dość dobrze poradziły sobie z wykryciem twarzy. Znowu lepiej wykryte zostały twarze bez maski niż zakryte maską. Detektor twarzy z biblioteki cvzone bardzo słabo wykrywał twarze jeśli ich rozmiar był mniejszy niż 125 pikseli.

Program SSD\_MoblileNetV2\_FPNLite najlepiej poradził sobie z wykryciem twarzy niezależnie od tego czy były widoczne czy zasłonięte maską. Był w stanie wykryć twarz nawet małych rozmiarów.

Wszystkie programy nie były w stanie wykryć twarzy o rozmiarze 25 pikseli

W tym porównaniu najlepiej wypadł program SSD\_MoblileNetV2\_FPNLite.

Obraz zawierający Wielobarwność, zrzut ekranu, tekst, Prostokąt

Opis wygenerowany automatycznie

**8.2 Skuteczność programów przy obrazie zaszumionym**

Program InceptionV3 wykrywał twarz bez maski niezależnie od poziomu szumu. Tak jak wcześniej detekcja twarzy powiodła się tylko dla jednej trzeciej przypadków twarzy zasłoniętych maską.

Programy InceptionV3\_cvzone i MobileNetV\_cvzone wykrywały twarz bez maski niezależnie od poziomu szumu. Dla twarzy zasłoniętej maską skuteczność wykrywania twarzy malała ze wzrostem szumu.

Program SSD\_MoblileNetV2\_FPNLite źle zareagował na pojawienie się szumu. Dla szumu o poziomie większym niż std=50 wykrywanie twarzy gwałtownie spada, a przy std=125 program jest prawie ślepy. Co ciekawe zaszumione twarze zasłonięte maską wykrywane są lepiej niż odsłonięte.

W tym porównaniu najlepiej wypadły InceptionV3\_cvzone i MobileNetV\_cvzone, co prawda mają problem z wykryciem małych twarzy, ale po przekroczeniu progu dość dobrze wykrywają twarze zarówno z maskami jak i bez niezależnie od zaszumienia.

1. **Analiza skuteczności programów w rozpoznawaniu twarzy z i bez maski.**

Obraz zawierający Wielobarwność, zrzut ekranu, Prostokąt, wzór

Opis wygenerowany automatycznie

**9.1 Skuteczność programów dla obrazów bez szumu**

Program InceptionV3 nieznacznie mniejszą skuteczność w rozpoznaniu czy znaleziona twarz jest w masce czy bez dla obrazów ciemnych i prześwietlonych.

Programy InceptionV3\_cvzone bardzo dobrze poradził sobie z rozpoznawaniem twarzy bez maski niezależnie od poziomu jasności. Wyraźne pogorszenie skuteczności dla twarzy prześwietlonych i niedoświetlonych wystąpiło dla twarzy zasłoniętych maską.

Program MobileNetV\_cvzone w porównaniu z InceptionV3\_cvzone wykazał się pogorszeniem rozpoznawania również dla twarzy bez maski.

Program SSD\_MoblileNetV2\_FPNLite najskuteczniej rozpoznał zarówno twarze z maskami jak i bez, ale wystąpiło zmniejszenie skuteczności dla obrazów prześwietlonych i niedoświetlonych.

W tym porównaniu najlepiej wypadł SSD\_MoblileNetV2\_FPNLite gdyż pozwalał na rozpoznanie dla mały twarzy i zapewniał podobną skuteczność do InceptionV3\_cvzone i MobileNetV\_cvzone dla większych twarzy.

Obraz zawierający Wielobarwność, zrzut ekranu, tekst, Prostokąt

Opis wygenerowany automatycznie

**9.2 Skuteczność programów przy obrazie zaszumionym**

Pojawienie się szumu jeszcze bardziej ograniczyło skuteczność rozpoznawania twarzy z maskami programu InceptionV3. Szum nie ma wpływu na rozpoznawanie twarzy bez masek.

Programy InceptionV3\_cvzone bardzo dobrze rozpoznał twarze bez masek niezależnie od zaszumienia. Zaszumienie wpływa na spory spadek skuteczności rozpoznawania twarzy w maskach.

Program MobileNetV\_cvzone w porównaniu z InceptionV3\_cvzone ma dodatkowo ograniczoną skuteczność rozpoznawania twarzy bez maski wraz ze wzrostem szumu oraz dla twarzy z maską ma gorszą skuteczność działania.

Program SSD\_MoblileNetV2\_FPNLite daje radę rozpoznać twarze z maskami i bez nawet dla małych twarzy jednak bardzo szybko traci skuteczność ze wzrostem poziomu szumu.

Dla obrazów mało zaszumionych najlepiej sprawdziłby się SSD\_MoblileNetV2\_FPNLite dla bardziej zaszumionych InceptionV3\_cvzone, jednak stracimy rozpoznanie małych twarzy.

Obraz zawierający Wielobarwność, zrzut ekranu, tekst, Prostokąt

Opis wygenerowany automatycznie

**9.3 Skuteczność programów przy obrazie zaszumionym dla różnych jasności**

InceptionV3 radzi sobie bardzo słabo w rozpoznawaniu twarzy z maskami na obrazach zaszumionych.

InceptionV3\_cvzone i MobileNetV\_cvzone dla obrazów mocno zaszumionych jako jedyne pozwalają rozpoznać twarze bez masek i w maskach.

SSD\_MoblileNetV2\_FPNLite znacznie gorzej radzi sobie z twarzami bez masek niż z maskami na obrazach zaszumionych, co jest dość zaskakujące.

To porównanie wskazuje na największą skuteczność InceptionV3\_cvzone w radzeniu sobie z zaszumionymi obrazami.

Obraz zawierający Wielobarwność, zrzut ekranu, Prostokąt, wzór

Opis wygenerowany automatycznie

**9.4 Skuteczność programów przy obrazie niedoświetlonym**

InceptionV3 bardzo słabo radzi sobie z rozpoznawaniem twarzy w maskach a przy wzroście poziomu szumu całkiem ślepnie. Również dla twarzy bez masek widać pogarszanie się skuteczności ze wzrostem szumu.

InceptionV3\_cvzone i MobileNetV\_cvzone dość dobrze radzą sobie z rozpoznawaniem twarzy zarówno w maskach jak i bez. Widać wyraźny spadek skuteczności ze wzrostem szumu. InceptionV3\_cvzone jest trochę bardziej skuteczny dla twarzy z maskami, a MobileNetV\_cvzone dla twarzy bez masek.

SSD\_MoblileNetV2\_FPNLite przy ciemnych obrazach wypada najgorzej w tym porównaniu. Już przy niewielkim poziomie szumu całkowicie traci zdolność do rozpoznawania twarzy z maskami i bez.

Najlepsze wyniki w tym zestawieniu uzyskał InceptionV3\_cvzone.

Obraz zawierający Wielobarwność, zrzut ekranu, Prostokąt, wzór

Opis wygenerowany automatycznie

**9.5 Skuteczność programów przy obrazie prześwietlonym**

InceptionV3 dość dobrze rozpoznaje prześwietlone twarze bez masek jednak prawie całkowicie ślepnie przy twarzach zakrytych maskami.

InceptionV3\_cvzone i MobileNetV\_cvzone również zmniejszają swoją skuteczność w rozpoznawaniu twarzy z maskami dla obrazów prześwietlonych. Wraz z jasnością wzrósł dla nich próg rozmiar powyżej którego możliwe jest rozpoznanie.

SSD\_MoblileNetV2\_FPNLite po raz kolejny zaskakuje wykazując mniejszą skuteczność dla twarzy bez masek niż dla twarzy z maskami. Powodem może być znacznie mniejsza liczba twarzy bez maski w zbiorze uczącym.

W tym porównaniu InceptionV3\_cvzone jest najlepszy gdyż pozwala na rozpoznanie większości twarzy bez masek przy zachowaniu dość wysokiego poziomu skuteczności rozpoznawania twarzy z maskami.

1. **Poprawność działania programów**

Obraz zawierający zrzut ekranu, tekst, Wielobarwność, Prostokąt

Opis wygenerowany automatycznie

**10.1 Poprawność działania programu InceptionV3**

InceptionV3 nie radzi sobie z wykrywaniem twarzy w maskach co skutkuje niskim maksymalnym poziomem poprawności działania. W sprzyjających warunkach tylko dla dwóch trzecich przypadków podaje prawidłową informację o obecności twarzy w masce lub bez. Wykazuje się wówczas również bardzo niskim poziomem błędnych odpowiedzi.

Zmniejszenie rozmiaru, niedoświetlenie, prześwietlenie a w szczególności większy szum znacząco wpływają na pojawienie się dużego udziału błędnego rozpoznania maski lub jej braku.

Obraz zawierający zrzut ekranu, Wielobarwność, tekst, Prostokąt

Opis wygenerowany automatycznie

**10.2 Poprawność działania programu InceptionV3\_cvzone**

InceptionV3\_cvzone Nie jest w stanie prawidłowo rozpoznać twarzy z maską lub bez gdy jest ona mniejsza niż 100 pikseli. Dla większych twarzy o dobrym naświetleniu poprawność działania jest na bardzo dobrym poziomie. Nawet pojawienie się większego szumu nie powoduje dużego spadku poprawnych odpowiedzi.

W optymalnych warunkach nie występują fałszywe detekcje, a wzrost szumu powoduje ich niewielką liczbę.

Obraz zawierający zrzut ekranu, tekst, Wielobarwność, Prostokąt

Opis wygenerowany automatycznie

**10.3 Poprawność działania programu MobileNetV\_cvzone**

MobileNetV\_cvzone, tak jak wcześniejszy, nie jest w stanie prawidłowo rozpoznać twarzy z maską lub bez gdy jest ona mniejsza niż 100 pikseli. Dla większych twarzy o dobrym naświetleniu poprawność działania jest na bardzo dobrym poziomie. Wzrost szumu i pogorszenie warunków oświetleniowych ma większy negatywny wpływ na poprawność wyników niż dla InceptionV3\_cvzone.

Pojawienie się większego szumu powoduje znaczny wzrost fałszywych detekcji, szczególnie dla obrazów niedoświetlonych.

Obraz zawierający zrzut ekranu, tekst, Wielobarwność, Prostokąt

Opis wygenerowany automatycznie

**10.4 Poprawność działania programu SSD\_MoblileNetV2\_FPNLite**

SSD\_MoblileNetV2\_FPNLite wykazuje się najlepszą poprawnością pracy dla małych obrazów ze wszystkich czterech programów. Niestety działanie to ograniczone jest jedynie do obrazów prawidłowo oświetlonych z niewielkimi szumami. Warunki słabe oświetlenie ogranicza wykrywanie twarzy i liczbę poprawnych detekcji. Prześwietlenia i wzrost zaszumienia wywołują znaczny wzrost fałszywych detekcji.

1. **Określenie jakości pracy programów**

Uzyskane wyniki rozpoznawania symulowanych danych zostały podzielone na dwie grupy ze względu na symulowane warunki:

Optymalne warunki pracy:

- jasność w przedziale (-50, 50)   
- rozmiar w przedziale (150, 275)   
- szum w przedziale (0, 25)

Trudne warunki pracy (otoczenie optymalnych warunków pracy):

- jasność w przedziale (-100, 100)  
- rozmiar w przedziale (50, 275)  
- szum w przedziale (0, 75)  
- z wyłączeniem części wspólnej z optymalnymi warunkami pracy.

Dla każdej grupy wyznaczony został średni poziom poprawnego i fałszywego rozpoznania twarzy w masce lub bez maski.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Program | Rozpoznanie dla  optymalnych warunków | | Rozpoznanie dla  trudnych warunków | |
| poprawne | fałszywe | poprawne | Fałszywe |
| InceptionV3 | 65.82% | 0.30% | 55.88% | 4.24% |
| **InceptionV3 cvzone** | **98.20%** | **0.05%** | **58.42%** | **2.47%** |
| MobileNetV2 cvzone | 95.72% | 1.43% | 54.85% | 6.19% |
| SSD MobileNetV2 FPNLite | 95.77% | 4.05% | 56.58% | 8.34% |

**Wniosek**

Najlepsze wyniki uzyskane zostały dla InceptionV3 cvzone. Zapewniają zarówno najwyższy poziom poprawnych odpowiedzi jak i najniższy poziom fałszywych detekcji zarówno w warunkach optymalnych jak i trudnych.

**Dodatek**

**Opis i instrukcja użytkowania przygotowanego oprogramowania**

Struktura katalogów:

- dataset mały zestaw obrazów

- train zestaw szkolący

- with\_mask

- without\_mask

- validate zestaw walidacyjny

- with\_mask

- without\_mask

- dataset\_big duży zestaw obrazów

- train zestaw uczący

- with\_mask

- without\_mask

- validate zestaw walidacyjny

- with\_mask

- without\_mask

- test zestaw testowy

- with\_mask

- without\_mask

- test\_50 zestaw 111 obrazów testowych do symulacji

- with\_mask

- without\_mask

- dataset\_det zestaw do szkolenia detektora SSD MobileNetV2 FPNLite

- annotations

- images

- modele wyszkolone modele, katalog docelowy szkolonych modeli (nadpisują się)

- wykresy ostateczne wykresy do raportu

- wykresy TFL wykresy do raportu

- wykresy v1 wykresy do raportu

- wykresy v2 wykresy do raportu

- wykresy v3 wykresy do raportu

Wszystkie programy znajdują się w katalogu głównym. Wszystkie programy uruchamiają się bez konieczności podawania parametrów wejściowych. Wymagane biblioteki znajdują się w pliku *requirements.txt*.

**Szkolenie modeli**

**train\_InceptionV3.py** – pierwotny kod szkolenia modelu InceptionV3 zmodyfikowany jedynie by zwracać wyniki do raportu. Wykorzystuje zbiór z katalogu *dataset*

**train\_InceptionV3\_v2.py** – program przygotowany do szkolenia modelu InceptionV3 z poprawionymi ustawieniami. Wykorzystuje zbiór z katalogu *dataset\_big*

**train\_InceptionV3\_v3.py** – program przygotowany do szkolenia modelu MobileNetV2. Wykorzystuje zbiór z katalogu *dataset\_big*

**Train\_TFLite2\_Object\_Detction\_Model.ipynb** – skrypt jupiter do szkolenia modelu SSD MobileNetV2 FPNLite. Wykorzystuje zbiór z katalogu *dataset\_det*. Uruchamiany na Google Colab.

**Testowanie działania programów z sygnałem z kamery internetowej**

Programy wczytują modele z katalogu modele

**real\_time\_v1.py** – InceptionV3 z kaskadą Haara

**real\_time\_v2.py** – InceptionV3 z cvzone

**real\_time\_v3.py** – MobileNetV2 z cvzone

**real\_time\_TFL.py** – SSD MobileNetV2 FPNLite

**Testowanie skuteczności wykrywania i rozpoznawania twarzy z maską lub bez**

Do testów wykorzystywany jest zestaw z katalogu *dataset\_big/test\_50*

**test\_model\_v1.py** – InceptionV3 z kaskadą Haara

**test\_model\_v2.py** – InceptionV3 z cvzone

**test\_model\_v3.py** – InceptionV3 z kaskadą Haara

**test\_model\_TFL.py** – InceptionV3 z kaskadą Haara

**modyfikacje.py** – zawiera funkcje generujące symulowany obrazek do testowania. Wykorzystywany jest przez wszystkie cztery programy testujące.

Wyniki testowania trafiają odpowiednio do *plików test\_results\_v1.txt test\_results\_v2.txt test\_results\_v3.txt test\_results\_TFL.txt*. W momencie uruchomienia programu wcześniejsza zawartość plików ulega skasowaniu.

**analiza\_statystyczna.py** przetwarza pliki wynikowe z testowania na wykresy i wartości do podsumowania wyników w pliku *ostateczny\_wynik\_<nazwa\_modelu>.txt*. Program jest wspólny dla wszystkich modeli.

**składanie\_obrazków.py** – składa wygenerowane wykresy w zbiorcze ilustracje do raportu.

Wszystkie pliki z programami dostępne są na GitHub pod adresem:

Dane z obrazkami oraz obliczone modele są czasowo udostępnione w sieci PW.