AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Praca dyplomowa

Rozpoznawanie twarzy odporne na podszywanie się
Spoofing resistant face recognition

Autor: *Mateusz Pilecki*
Kierunek studiów: *Automatyka i Robotyka*
Opiekun pracy: *dr inż. Piotr Pawlik*

Kraków, 2022

Spis treści

1. Wprowadzenie

1.1. Wstęp

W ostatnich latach systemy rozpoznające oparte na biometryce zyskały ogromną popularność. Jest to uwarunkowane niezliczoną liczbą zastosowań, jakie posiadają takie aplikacje. Wspólną cechą takich systemów jest konieczność pobierania danych biometrycznych, czyli danych osobowych dotyczących cech fizycznych, fizjologicznych lub behawioralnych osoby fizycznej [1]. Dzięki tym danym systemy rozpoznające potrafią z wysokim prawdopodobieństwem jednoznacznie zidentyfikować osobę. Wśród powszechnie wykorzystywanych systemów biometrycznych możemy wyróżnić systemy rozpoznające na podstawie:

- linii papilarnych,
- geometrii dłoni,
- tęczówki oka,
- twarzy,
- głosu,
- układu naczyń krwionośnych dłoni.

Niestety pomimo licznych zalet oraz zastosowań, systemy te mają istotne niedoskonałości. Dla człowieka rozpoznanie osoby na podstawie głosu czy twarzy jest dosyć prostym zadaniem, jednak dla cyfrowej maszyny jest to o wiele trudniejsze. Dodatkowymi niedogodnościami są zmieniające się warunki, dla których następuje odczyt danych. Wszystkie powyższe czynniki powodują konieczność stałego rozwoju tej dziedziny nauki w celu stworzenia systemów biometrycznych jak najlepiej służących człowiekowi.

Bardzo popularnym rozwiązaniem wykorzystującym dane biometryczne stały się ostatnio systemy rozpoznawania twarzy. Takie aplikacje mają bardzo wiele praktycznych zastosowań i potrafią znacząco usprawnić procesy weryfikacji. Niestety są one bardzo podatne na ataki ze strony osób zewnętrznych. Ta luka w bezpieczeństwie zmusiła badaczy do stworzenia specjalnych zabezpieczeń, które mają zapobiegać podszywaniu się pod inne osoby.

1.2. Cel pracy

Celem pracy jest przegląd metod podszywania się w systemach rozpoznawania twarzy oraz stworzenie modelu uczenia maszynowego do rozpoznawania twarzy odpornego na ataki. Implementacja składa się z dwóch głównych części. Pierwsza część obejmuje wykonanie aplikacji potrafiącej rozpoznać użytkownika, porównując jego twarz ze zdjęciami przechowywanymi w bazie danych. Na drugą część składa się utworzenie modelu uczenia maszynowego potrafiącego rozpoznać, czy twarz ze zdjęcia jest prawdziwa, czy jest ona częścią ataku na system rozpoznawania twarzy. Ostateczny program ma zawierać obydwa odpowiednio zintegrowane systemy i powinien zwracać użytkownikowi informacje zarówno o wyniku rozpoznawania twarzy, jak o wyniku detekcji podszywania się. Program poddany zostanie odpowiednim testom dla różnych parametrów oraz danych testowych.

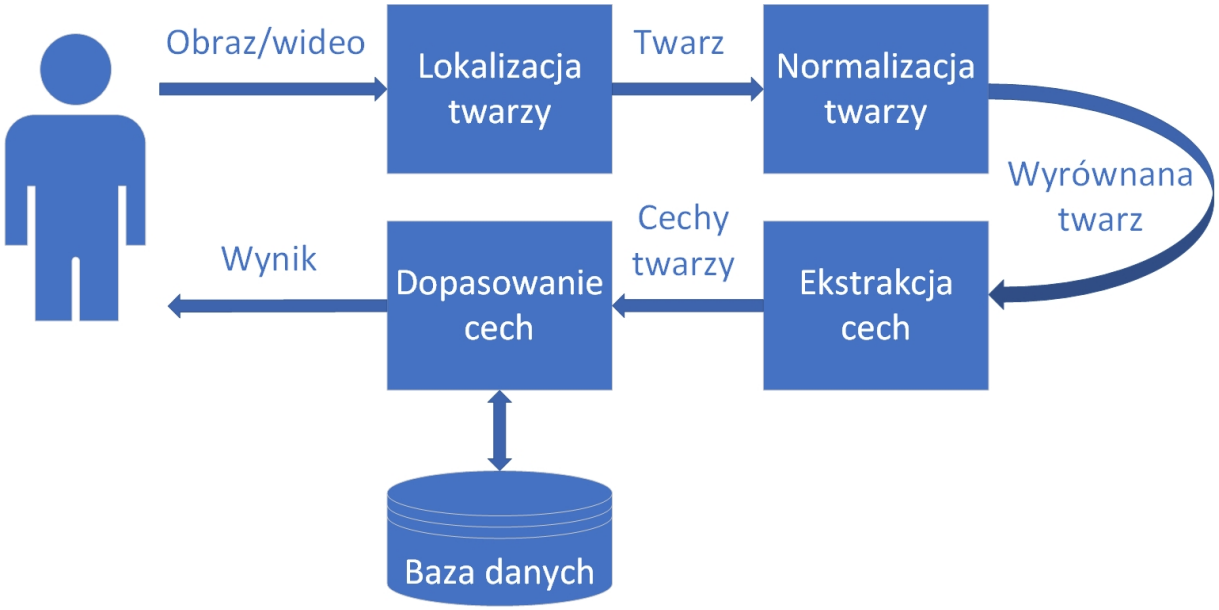
2. Rozpoznawanie twarzy

Rozpoznawanie twarzy jest bardzo ważnym odgałęzieniem przetwarzania wizyjnego. Największe międzynarodowe firmy chętnie korzystają lub wdrążają swoje algorytmy detekcji twarzy. Jest to spowodowane rosnącym popytem na wykorzystanie biometryki w celach weryfikacyjnych. Systemy rozpoznawania twarzy są wykorzystywane w wielu dziedzinach nauki oraz gospodarki. Korzysta się z nich między innymi w celu [2]:

- weryfikacji osób podczas kontroli granicznych, między innymi na lotniskach,
- zabezpieczenia dostępu do aplikacji, urządzenia czy danych osobistych,
- automatycznego wykrywania osób oraz ich cech twarzy na zdjęciach w aplikacjach dla osób niewidomych,
- egzekwowania prawa w tym do identyfikacji zatrzymanych osób przez policję,
- poszukiwania zaginionych osób,
- autoryzowania transakcji bankowych,
- mierzenia poziomu satysfakcji klientów.

2.1. Proces przetwarzania zdjęcia

Aplikacje służące do rozpoznawania twarzy możemy podzielić na dwie kategorie. Systemy identyfikujące, porównujące jedno zdjęcie z bazą danych wielu zdjęć oraz systemy uwierzytelniające, które sprawdzają zgodność pojedynczego zdjęcia z innym zdjęciem [3]. W obu przypadkach proces przetwarzania zdjęcia przebiega zgodnie z rysunkiem 2.1



Rys. 2.1. Proces przetwarzania zdjęcia w systemie rozpoznawania twarzy.

2.1.1. Lokalizacja twarzy

Algorytm lokalizacji twarzy może działać zarówno na podstawie pojedynczego zdjęcia, jak i strumienia wideo. Jego zadaniem jest selekcja fragmentu obrazu lub klatki video, na którym znajduje się twarz. Jeżeli wejściem algorytmu jest strumień wideo, aplikacja powinna śledzić twarz użytkownika przez cały proces działania. Celem tego etapu przetwarzania zdjęcia jest eliminacja zbędnych informacji dla systemu rozpoznawania twarzy. Dzięki tej operacji zmniejsza się rozmiar danych wejściowych późniejszych etapów przetwarzania, co wpływa pozytywnie na szybkość odpowiedzi całego systemu.

Na przestrzeni ostatnich lat ukształtowane zostały dwie główne metody detekcji twarzy [4]. Pierwsze podejście bazuje na analizie cech twarzy. Polega ono na ewaluacji wszystkich cech charakterystycznych twarzy, takich jak ich wygląd, lokalizacja względem całego zdjęcia czy względem pozostałych punktów charakterystycznych. Wszystkie cechy charakterystyczne zostają następnie przyporządkowane do ogólnego modelu geometrycznego obrazu. Drugie podejście jest podejściem holistycznym i polega na automatycznym zlokalizowaniu i ekstrakcji twarzy z całego obrazu za pomocą między innymi sieci neuronowych.

Podstawowym schematem wykrywania twarzy jest algorytm detekcji opierający się na modelu koloru skóry [5]. Bazuje on jednak wyłącznie na informacjach o odcieniu oraz nasyceniu koloru (chrominancja) przez co nie jest wystarczająco dokładny.

Kolejną metodą lokalizacji twarzy jest algorytm wyznaczania powierzchni własnej twarzy (ang. eigenfaces) [6]. W pewnym przybliżeniu twarz ludzką można uzyskać za pomocą złożenia

odpowiednich powierzchni własnych. Zestaw takich powierzchni otrzymuje się za pomocą algorytmu nazywanego analizą głównych składowych (ang. principal component analysis, PCA).

Inne popularne metody detekcji twarzy to między innymi:

– kaskady Haara,
– histogram zorientowanych gradientów (HOG),
– głębokie sieci neuronowe.

2.1.2. Normalizacja obrazu twarzy

Normalizacja obrazu twarzy jest implementowana w celu zestandaryzowania twarzy w kontekście zarówno kształtu geometrycznego, jak i oświetlenia. Jest to spowodowane potrzebą dostarczenia do kolejnego etapu procesu przetwarzania danych ujednoliconych twarzy. Normalizacja geometryczna opiera się przede wszystkim na wycięciu z obrazu twarzy na podstawie wyznaczonej wcześniej lokalizacji. Otrzymany fragment zdjęcia zostaje następnie poddany różnego rodzaju operacjom przekształcającym, mającym na celu ustawienie twarzy w pozycji frontalnej. Aby to osiągnąć należy algorytmicznie wyznaczyć stałą liczbę punktów charakterystycznych, które będą służyły jako punkt odniesienia do późniejszych transformacji obrazu. W większości przypadków punkty te skupiają się wokół wyróżniających się miejsc twarzy takich jak oczy, nos, usta czy brwi. Ze względu na różne warunki oświetleniowe panujące podczas próby detekcji twarzy obraz często jest również normalizowany pod kątem iluminacji.

2.1.3. Ekstrakcja cech

Celem tego etapu przetwarzania jest wybór i ekstrakcja odpowiednich cech twarzy. Z ludzkiego punktu widzenia, patrząc na twarz innej osoby od razu wiadomo, jakie cechy charakterystyczne należałoby wybrać, na przykład rozmiar ucha czy długość nosa. Dla programu widzącego jedynie wartości pikseli nie jest to jednak takie oczywiste. W związku z tym problemem powstały specjalne algorytmy uczenia maszynowego potrafiące znaleźć odpowiednio wielki zestaw cech, na podstawie których dopasowanie twarzy powinno dać najlepsze efekty. Taki program powinien zminimalizować wariancję pomiędzy obrazami tej samej klasy (twarzami tej samej osoby), tym samym maksymalizując wariancję między obrazami różnych klas (twarzami różnych osób). W idealnym przypadku cechy powinny być łatwo wydobywalne z obrazu wejściowego, a rozmiar wektora, który je opisuje nie powinien być za duży. Niespełnienie powyższych warunków mogłoby doprowadzić do znaczącego wydłużenia czasu procesu rozpoznawania twarzy, co nie jest pożądanym efektem w systemach weryfikacji. Niestety proces ekstrakcji cech nie jest łatwy ze względu na dużą różnorodność obrazów wejściowych spowodowaną między innymi zmianami warunków oświetleniowych, orientacji twarzy, mimiki twarzy czy obecnością dodatkowych atrybutów, takich jak okulary lub nakrycie głowy. Mimo tego

współcześnie zaimplementowane metody potrafią wykonać to zadanie z bardzo zadowalającymi rezultatami.

Jedną z pierwszych propozycji rozwiązania problemu ekstrakcji cech była próba reprezentacji twarzy jako zależności geometryczne między punktami charakterystycznymi twarzy [7]. Następnie popularne stało się podejście bazujące na analizie tekstur miejsc charakterystycznych twarzy. Wyróżniające się metody postępujące w ten sposób to między innymi analiza głównych składowych (PCA), czy liniowa analiza dyskryminacyjna (ang. Linear discriminant analysis, LDA) [8]. Kolejnym usprawnieniem ekstrakcji cech są liniowe deskryptory cech twarzy. Rozpowszechnione zostały takie metody jak filtry Gabora czy lokalne wzory binarne (ang. local binary patterns, LBP) [9]. W ostatnich latach dzięki stale zwiększającej się ilości danych zaczęto stosować głębokie sieci neuronowe, a w szczególności konwolucyjne sieci neuronowe.

2.1.4. Dopasowanie twarzy

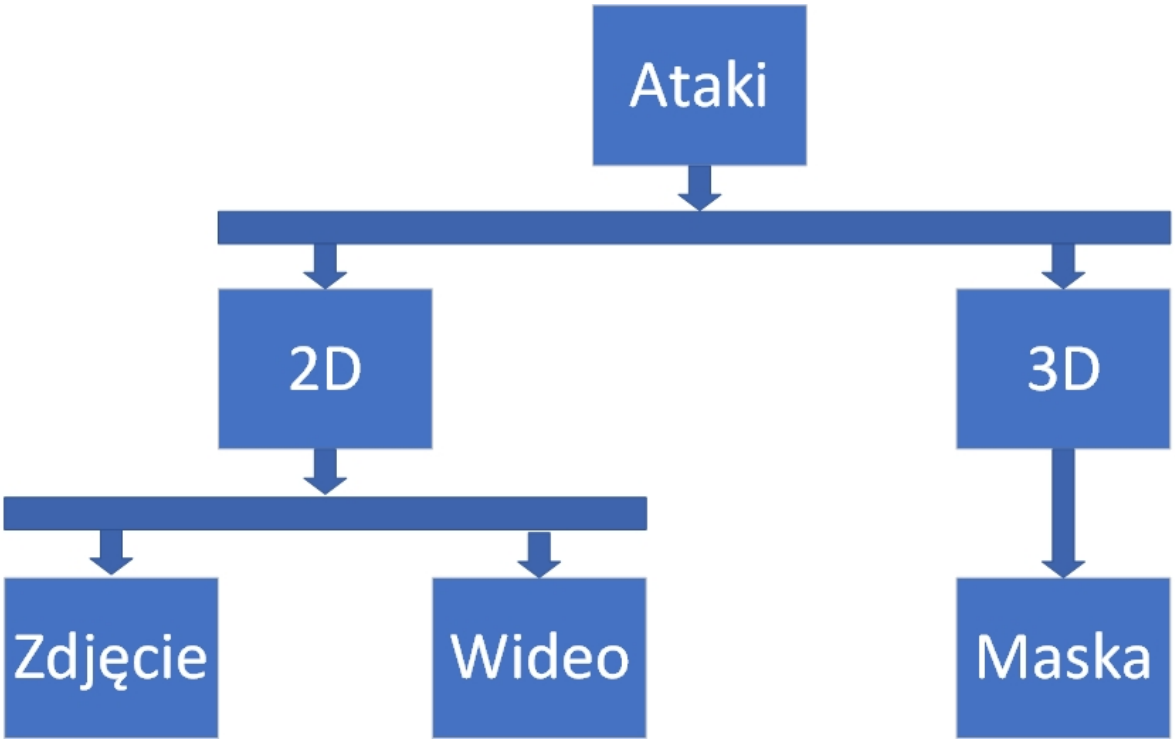
W procesie dopasowania twarzy aplikacja na podstawie wyznaczonych wcześniej cech porównuje twarz z jedną lub wieloma twarzami z bazy danych. Dla systemu uwierzytelniającego wynik dopasowywania może przyjąć dwie wartości - twarz pasuje do twarzy z bazy danych lub nie. W systemie identyfikującym natomiast wynik jest tablicą o rozmiarze równym wielkości bazy danych twarzy. Tablica ta zawiera dystans pomiędzy wektorem cech przetwarzanej twarzy, a wektorami cech zdjęć z bazy danych. Z takiej tablicy wybierany jest obraz z najmniejszą odległością. Następnie wartość ta porównywana jest z odpowiednio dobranym współczynnikiem tolerancji, w celu uniknięcia dopasowania nieznajdującej się w bazie danych twarzy.

3. Ataki na systemy rozpoznawania twarzy

Systemy rozpoznawania twarzy są w dzisiejszych czasach nieodłącznym elementem aplikacji wielu firm oraz instytucji. Niewątpliwie przyczyniły się one do ułatwienia wielu aspektów życia, były pomocne podczas procesów weryfikacji czy autoryzacji. Niestety wraz ze zwiększającą się popularnością systemów rozpoznawania twarzy wzrosło również zainteresowanie nielegalnymi działaniami związanymi z powyższymi aplikacjami, takimi jak podszywanie się pod inne osoby czy kradzież tożsamości. W związku z tym powstała potrzeba stworzenia metod, które zapobiegałyby takim atakom.

3.1. Metody podszywania się

W całym procesie przetwarzania twarzy istnieje wiele miejsc podatnych na ataki zewnętrzne. Jedne z nich potrzebują dostępu do wewnętrznego kodu programu lub danych systemu, inne bazują na możliwości oszukania czujników dokonujących pomiaru danych bez potrzeby dostępu do danych wewnętrznych. Pierwsze to tak zwane ataki pośrednie, które mają na celu między innymi uzyskanie dostępu do bazy danych, klasyfikatora czy kanałów przepływu danych. Zapobieganie tego typu atakom jest związane przede wszystkim z kwestią cyberbezpieczeństwa. Ataki bezpośrednie natomiast nie potrzebują dostępu do danych programu. Ich celem jest wprowadzenie w błąd systemu rozpoznawania twarzy już na poziomie odczytu informacji z czujników. W takich atakach wykorzystywane są różne techniki podszywania się przedstawione na rysunku 3.1.



Rys. 3.1. Rodzaje bezpośrednich ataków na system rozpoznawania twarzy.

Istnieją również inne metody podszywania się, takie jak makijaż czy operacja twarzy [10], jednak ze względu na swoją specyfikę nie są często stosowane.

3.2. Metody zapobiegania podszywaniu się

Ze względu na coraz łatwiejszy dostęp do danych personalnych, takich jak zdjęcia w internecie czy na portalach społecznościowych ataki na systemy rozpoznawania twarzy są ogromnym zagrożeniem. W związku z tym koniecznym stało się stworzenie algorytmów wykrywających próby podszywania się. Zgodnie z pracą Abdenoura Hadida i in. [11] sposoby detekcji takich prób dzielimy na metody:

- bazujące na czujnikach, wykrywające sygnały charakterystyczne dla podszywania się,
- korzystające z dodatkowego sprzętu komputerowego,
- wykorzystujące interakcję z użytkownikiem, na przykład prosząc go o wykonanie losowo wygenerowanej sekwencji ruchów,
- stosujące algorytmy zaimplementowane bezpośrednio w oprogramowaniu systemu rozpoznawania twarzy.

Zważywszy na łatwość implementacji oraz niskie koszty produkcji najczęściej stosowane są te ostatnie. Wśród tych metod wyróżniają się algorytmy potrafiące wykryć próbę podszywania się bez większej interakcji z użytkownikiem. Jest to spowodowane tym, że takie rozwiązania

nie potrzebują dodatkowego sprzętu, a wynik ich działania jest widoczny wkrótce po rozpoczęciu procesu przetwarzania danych. Metody takie można dodatkowo podzielić na statyczne oraz dynamiczne [12]. Pierwsze z nich do detekcji wykorzystują cechy statyczne, takie jak tekstury twarzy. Mając takie informacje są w stanie określić czy mają do czynienia z żywą osobą czy z wcześniej spreparowanym narzędziem do podszywania się. Niestety, ze względu na potrzebę ekstrakcji odpowiednich cech tekstur, nie radzą one sobie dobrze ze zdjęciami niskiej jakości. Na skuteczność tych algorytmów często ogromny wpływ mają też zmienne warunki oświetleniowe. Oprócz tekstur systemy statyczne mogą wykorzystywać również informacje o absorpcji, odbiciu, rozpraszaniu czy załamaniu światła. Metody dynamiczne natomiast bazują na analizie ruchu. Na podstawie próbek wideo próbują one wydobyć sekwencje naturalnych ruchów, takich jak mruganie oczami, zmiany pulsu, mimika twarzy czy ruch ust. Jednym z przykładów analizy dynamicznej jest poszukiwanie korelacji pomiędzy ruchami twarzy a ruchem tła [13]. Ruch autentycznej twarzy powinien być ściśle nieskorelowany z ruchem otoczenia.

4. Implementacja systemu rozpoznawania twarzy

4.1. Rozpoznawanie twarzy

Pierwszym etapem procesu rozpoznawania twarzy jest zlokalizowanie jej na zdjęciu. W tym celu posłużono się deskryptorem cech nazywanym histogramem zorientowanych gradientów (HOG). Głównym zadaniem każdego deskryptora cech jest przetworzenie obrazu podanego w przestrzeni RGB do wektora opisującego taki obraz. W tym celu zdjęcie zostaje najpierw skonwertowane do obrazu w skali szarości. Następnie dla każdego piksela obliczane są pionowe oraz poziome gradienty. Operację tę można wykonać za pomocą filtra Sobela o rozmiarze jądra równym 1. Kolejno obliczono wielkość oraz kierunek gradientu za pomocą wzorów:

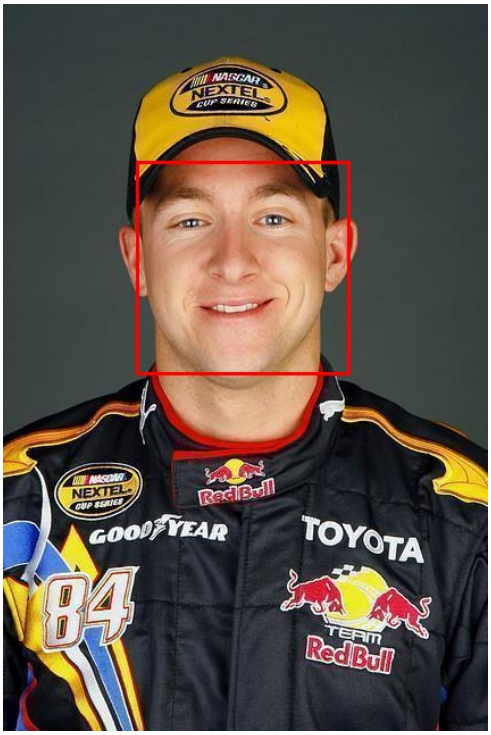
$$g = \sqrt{g_x^2 + g_y^2} \tag{4.1}$$

$$\theta = \arctan \frac{g_y}{g_x} \tag{4.2}$$

Aby zmniejszyć ilość szczegółów podzielono obrazek na fragmenty o wielkości 16x16 pikseli. Następnie dla każdego wycinka wyznaczono histogram. Przedziałami histogramu są kierunki gradientu (kąt 0-180 stopni), natomiast wartościami są zsumowane wielkości gradientu (po uwzględnieniu odpowiednich proporcji). Aby deskryptor był odporny na zmienne warunki oświetleniowe w kolejnym kroku znormalizowano histogram. Rezultat powyższych operacji został przedstawiony na rysunku 4.1.

Rys. 4.1. Wynik operacji HOG na przykładowym obrazku.

Ostatecznie wynik poddany został konwersji do wektora oraz podany na wejście algorytmu, który porównuje dane wraz z wytrenowanym wcześniej na wielu obrazkach wzorem twarzy. Na wyjściu otrzymujemy współrzędne, na których według klasyfikatora najprawdopodobniej znajduje się twarz. Efekt przykładowej lokalizacji współrzędnych twarzy jest zobrazowany na rysunku 4.2.



Rys. 4.2. Przykładowa wizualizacja zlokalizowanych współrzędnych twarzy.

Drugim etapem procesu rozpoznawania twarzy jest normalizacja. W tym celu użyto wytrenowanych drzew decyzyjnych, które znajdowały na każdym zdjęciu 68 charakterystycznych punktów, występujących na każdej twarzy, takich jak zewnętrzne krawędzie brwi, brzegi oczu, czubek nosa czy podbródek. Do uczenia maszynowego została wykorzystana technika gradient boosting z uwzględnieniem błędu średniokwadratowego (Mean Squared Error) [14]. W rezultacie otrzymano kaskadę drzew decyzyjnych będących w stanie zlokalizować odpowiednie cechy twarzy. Wyniki powyższych operacji przedstawione są na rysunku 4.3.

Rys. 4.3. Wizualizacja punktów charakterystycznych twarzy.

Następnie używając podstawowych przekształceń afinicznych takich jak między innymi rotacja, skalowanie czy powinowactwo scentralizowano wszystkie wykryte wcześniej punkty, a co za tym idzie uzyskano wyśrodkowaną względem całego obrazu twarz. Powyższe operacje nie są konieczne, jednak znacząco wpływają na jakość całego procesu rozpoznawania twarzy. Przykładowy obraz bez przekształceń oraz po wykonaniu wspomnianych operacji przedstawiony jest na rysunku 4.4.

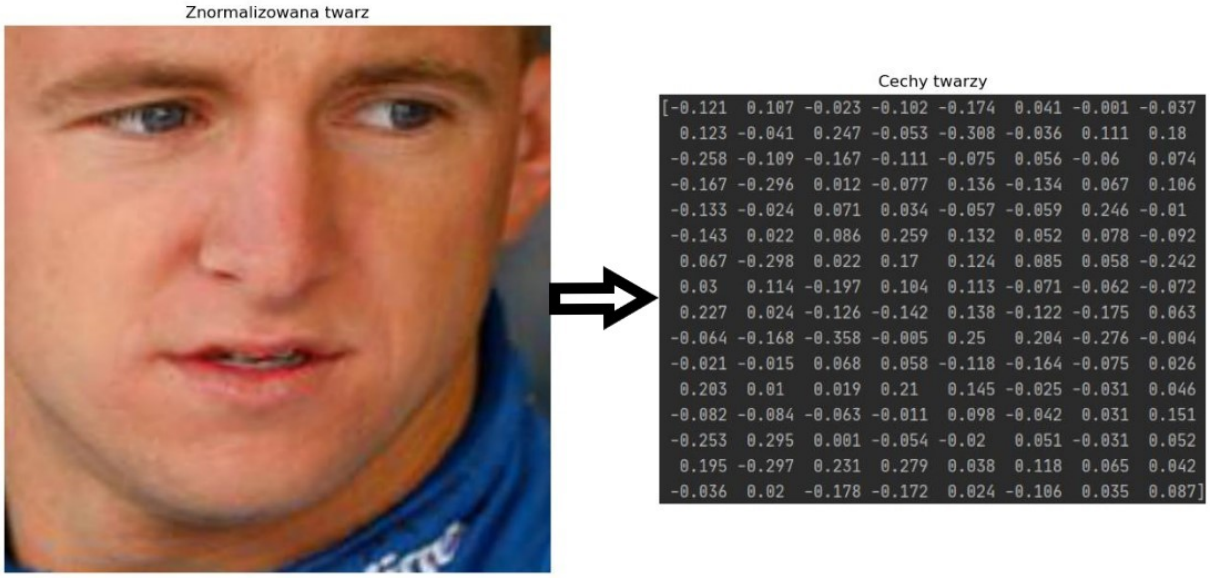
Rys. 4.4. Przykładowa twarz przed oraz po wykonaniu normalizacji.

Kolejnym etapem procesu przetwarzania obrazu twarzy jest ekstrakcja cech. Do realizacji tego zadania użyto wytrenowanej głębokiej konwolucyjnej sieci neuronowej. Sieć ta została nauczona generować 128 odpowiednich cech dla każdej twarzy jaką otrzyma. Na początku wagi modelu uczenia maszynowego są dobierane losowo. Następnie w procesie uczenia sieć dąży do minimalizacji tak zwanej potrójnej funkcji starty (ang. triplet loss) [15]. Przy takim postępowaniu dane wejściowe są porównywane z pasującymi im innymi danymi (klasa pozytywna) oraz z niepasującymi danymi (klasa negatywna). Funkcją straty jest różnica dwóch dystansów - między danymi wejściowymi a przykładem z klasy pozytywnej oraz między danymi wejściowymi, a przykładem z klasy negatywnej. Sieć dąży do minimalizacji pierwszego z nich oraz maksymalizacji drugiej odległości - minimalizując tym samym funkcję straty. W procesie ekstrakcji cech dla obrazu twarzy sieć neuronowa porównuje otrzymane na wejście zdjęcie twarzy znanej osoby z:

1. inną twarzą tej samej znanej osoby (klasa pozytywna)
2. losową twarzą innej osoby (klasa negatywna)

Za każdym razem gdy sieć neuronowa zmienia swoje parametry, porównuje aktualne cechy wejściowego obrazu twarzy z cechami zdjęcia twarzy klasy pozytywnej próbując zminimalizować różnicę między nimi. W podobny sposób algorytm porównuje jej cechy z przykładem z klasy negatywnej, tym razem próbując zmaksymalizować ich różnicę. Po wielokrotnym powtórzeniu tego cyklu dla wielu obrazów wielu osób, program jest w stanie znaleźć optymalne

cechy dla twarzy. Cały proces ekstrakcji jest kluczowy dla systemów rozpoznawania twarzy. Pozwala on na łatwe porównywanie twarzy oraz zmniejsza złożoność obliczeniową tego etapu, poprzez redukcje danych do wektora 128 cech. Rezultat działania algorytmu widoczny jest na rysunku 4.5. Dla czytelności dane liczbowe zostały zaokrąglone do 3 miejsc po przecinku.



Rys. 4.5. Rezultat ekstrakcji cech.

Ostatnią fazą systemu jest próba dopasowania twarzy do jednej z tych, znajdujących się w bazie danych. Odbywa się to poprzez porównanie cech otrzymanych z poprzedniego etapu. Algorytm kolejno oblicza odległości pomiędzy wektorem cech obrazu testowego, a wektorami cech zdjęć znajdujących się w bazie danych. Następnie porównuje je z progiem, poniżej którego dana osoba zostaje uznana za osobę z bazy danych. Eliminuje to wiele przypadków, w których postać niebędąca w bazie danych została niepoprawnie dopasowana. Jeżeli odległość wektorów cech dwóch obrazów jest mniejsza niż 0.6 algorytm uznaje te zdjęcia jako zdjęcia tych samych osób. Ostatecznie na wyjściu systemu rozpoznawania twarzy otrzymujemy nazwę pliku z bazy danych, jeśli twarz została odpowiednio dopasowana lub napis "Unknown"w przeciwnym przypadku.

4.2. Detekcja podszywania się

W celu implementacji detekcji podszywania się zdecydowano się na rozwiązanie wykorzystujące algorytmy zaimplementowane bezpośrednio w oprogramowaniu systemu rozpoznawania twarzy. Przy takim podejściu jedyną interakcją z programem, jaka jest wymagana ze strony użytkownika, jest pokazanie swojej twarzy do czujników znajdujących się w urządzeniu, na którym wdrożona jest aplikacja. Do realizacji tego zadania posłużono się algorytmami uczenia

maszynowego. W związku z tym konieczny był przegląd dostępnych zbiorów danych oraz zapoznanie się z ich atrybutami.

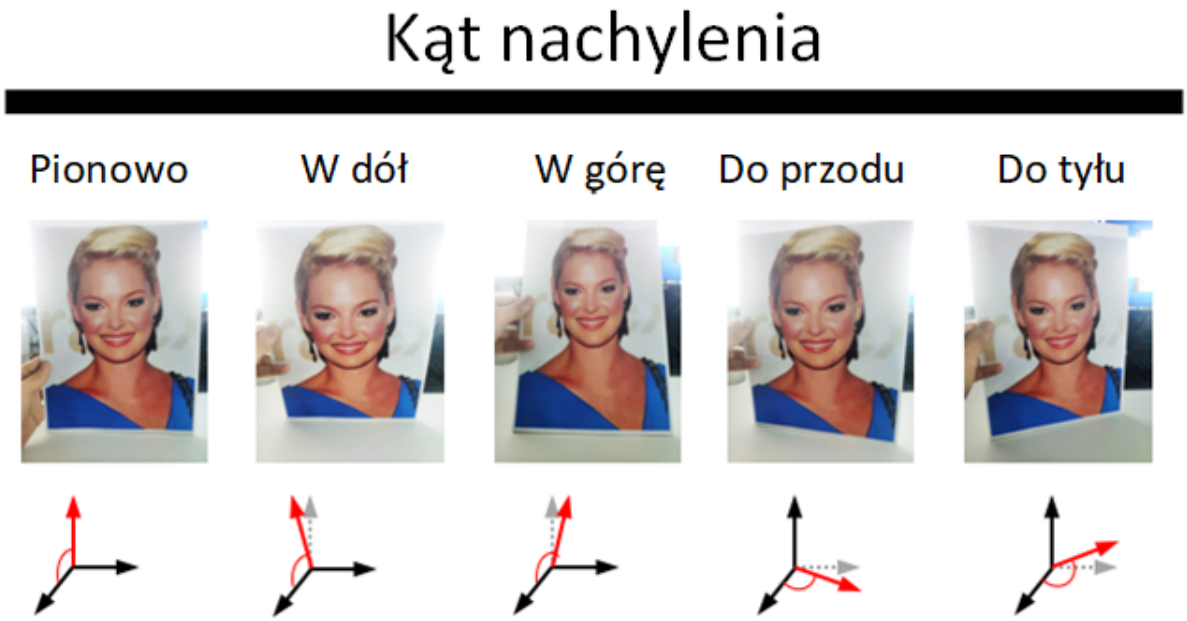
4.2.1. Zbiory danych

Istnieje wiele publicznie dostępnych zbiorów danych skoncentrowanych wokół tematu detekcji ataków na systemy rozpoznawania twarzy. Każdy taki zestaw różni się od pozostałych między innymi ilością plików, rodzajem podszywania się, rodzajem danych wejściowych czy ilością osób, które zostały wykorzystane podczas ich zbierania. Podsumowanie różnic między niektórymi zbiorami danych przedstawione jest w tabeli 4.1. Oznaczenia W oraz O w kolumnie Dane oznaczają odpowiednio dane w postaci wideo oraz obrazów.

Tabela 4.1. Przegląd zbiorów danych

Zbiór danych	Rok wydania	Osoby	Dane(W/O)	Czujniki	rodzaje ataków
Replay-Attack	2012	50	1200(W)	2	zdjęcia papierowe oraz cyfrowe
Msspoof	2015	21	4704(O)	2	zdjęcia papierowe
MSU-USSA	2016	1140	10260(O)	2	zdjęcia papierowe oraz cyfrowe
CASIA-SURF	2018	1000	21000(W)	1	papierowe maski 2D
CelebA-Spoof	2020	10177	625537(O)	>10	zdjęcia papierowe, cyfrowe maski 2D oraz 3D

Z powyższej tabeli wynika, że najobszerniejszym, a zarazem najbardziej wszechstronnym zbiorem danych jest wydany w 2020 roku CelebA-Spoof [16]. Jest on publicznie dostępny do niekomercyjnego użytku naukowego. Zawiera 625 537 zdjęć pobranych od 10 177 celebrytów. Wśród tych obrazów 202 599 to twarze niebędące próbą podszywania się. Pozostałe obrazy przedstawiają różnego rodzaju ataki na system rozpoznawania twarzy. Wszystkie zdjęcia będące próbą podszywania się zostały poddane generalizacji, aby zwiększyć różnorodność zbioru danych. Każdy obraz został zdywersyfikowany pod względem kąta nachylenia, zniekształcenia oraz urządzenia pobierającego dane. Wprowadzono 5 kątów nachylenia: pionowo, w dół, w górę, do przodu oraz do tyłu. Wszystkie rodzaje kątów nachylenia z przykładami przedstawione są na grafice 4.6.



Rys. 4.6. Kąty nachylenia.

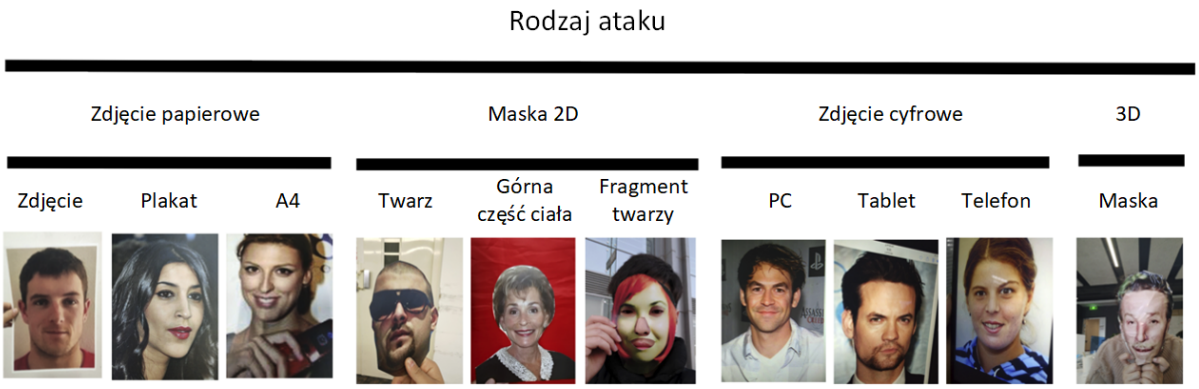
Próby podszywania się zostały zarejestrowane w 4 różnych zniekształceniach: normalnym, w środku, na zewnątrz oraz w rogach. Wszystkie powyższe parametry przedstawione są na rysunku 4.7.

Rys. 4.7. Zniekształcenie narzędzia ataku.

Zabieg ataku na system rozpoznawania twarzy został zarejestrowany na 24 urządzeniach, które można przypisać do 4 głównych kategorii. Kategorie te zostały zebrane na ilustracji 4.8.

Rys. 4.8. Kategorie urządzeń.

Dodatkowo każde zdjęcie zawiera 43 atrybuty. 40 z nich opisuje cechy twarzy, takie jak kształt brwi, wielkość nosa, kolor oraz kształt włosów, a nawet czy osoba na zdjęciu nosi kolczyki lub nakrycie głowy. Cechy te dotyczą jedynie prawdziwych zdjęć przedstawiających osoby fizyczne. Pozostałe 3 atrybuty są powiązane ze zdjęciami ilustrującymi próby podszywania się. Pierwszy z nich opisuje rodzaj ataku, a jego wartości są przedstawione na grafice 4.9.



Rys. 4.9. Rodzaje ataków.

Drugi atrybut dostarcza wiadomości o rodzaju oświetlenia, natomiast trzeci mówi o miejscu, w jakim nastąpiła próba oszukania systemu rozpoznawania twarzy. Powyższe parametry przedstawione są na rysunku 4.10.

Rys. 4.10. Warunki oświetleniowe oraz miejsce ataku.

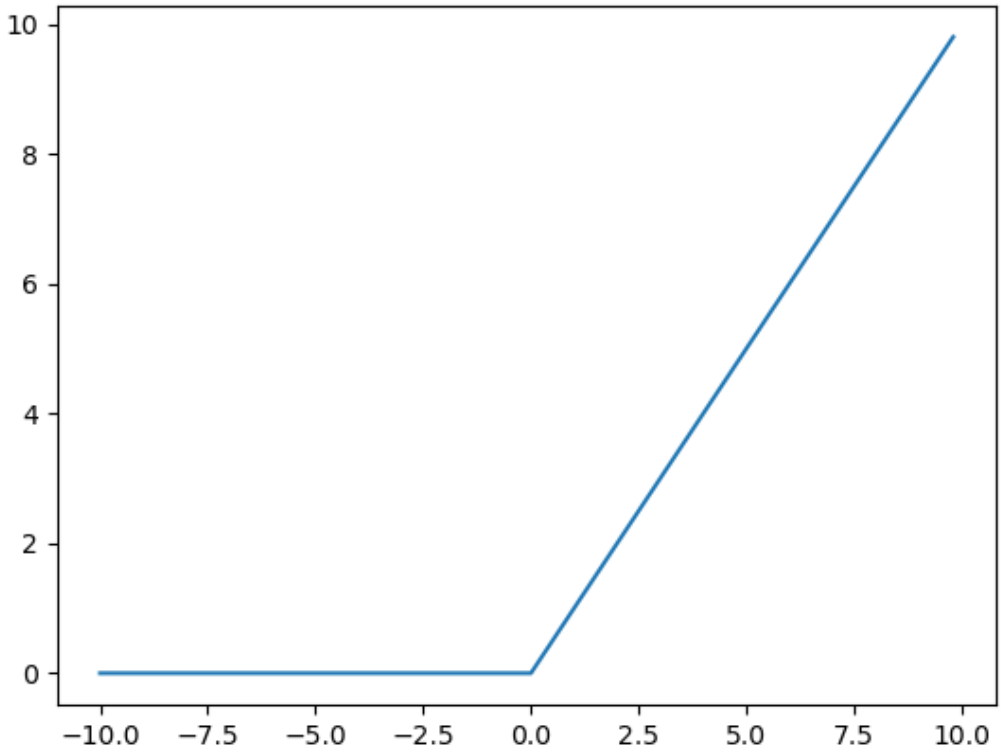
4.2.2. Przetwarzanie oraz format danych

Zanim zdjęcia znajdujące się w zbiorze danych CelebA-Spoof trafią do pierwszej warstwy sieci neuronowej, muszą zostać odpowiednio przetworzone. Pierwszą operacją, jakiej obrazy zostały poddane było wycinanie twarzy z obrazu. Współrzędne wszystkich twarzy ze zbioru danych CelebA-Spoof zostały wcześniej obliczone oraz zapisane w odpowiednich plikach. Do ich detekcji autorzy użyli detektora RetinaFace [17]. Na podstawie informacji otrzymanych z detektora twarz została wycięta i przekonwertowana do formatu trójwymiarowej tablicy nazywanej tensorem. Następnie rozmiar obrazu został zmieniony do wymiarów 224x224x3, a jego wartości zostały znormalizowane tak, aby mieściły się w przedziale <0, 1>.

Tak przetworzone zdjęcia wraz z etykietami mówiącymi, czy dana twarz jest próbą podszywania się zostały następnie zapisane do tak zwanych TFRecordów. Jest to prosty format do przetrzymywania sekwencji danych binarnych. Raz zapisane dane do formatu TFRecord można wielokrotnie w bardzo krótkim czasie wczytać do postaci zestawu danych biblioteki TensorFlow, który w bardzo łatwy sposób można przekazać do sieci neuronowej. Każdy rekord w pliku TFRecord zawiera dwie wartości: obraz, który zapisywany był w postaci danych binarnych oraz odpowiadającą mu etykietę zapisaną jako liczba całkowita. Przy odczytywaniu plików dane binarne są deserializowane do postaci pierwotnej, czyli do formatu tensorów. Funkcja ładująca dane z TFRecordów kolejno znajduje wszystkie pliki z rozszerzeniem .tfrecords, deserializuje dane oraz wczytuje je do zbioru danych pakietu TensorFlow, uwzględniając ilość danych propagowanych przez sieć (ang. batch size) oraz automatyczne strojenie danych (ang. automatic tuning).

4.2.3. Model uczenia maszynowego

Do stworzenia modelu uczenia maszynowego, mającego pełnić rolę klasyfikatora czy dana twarz jest próbą podszywania się, czy jest twarzą autentycznego użytkownika użyto biblioteki Keras. Jest to interfejs biblioteki TensorFlow dostarczający funkcjonalności potrzebnych do zbudowania sieci neuronowych w języku Python. Ze względu na nietypową specyfikacje klasyfikatora zdecydowano się skorzystać z techniki nazywanej transfer learning. Polega ona na wykorzystaniu wiedzy zdobytej podczas rozwiązywania jednego problemu i zastosowaniu jej do innego, powiązanego problemu. W tym celu załadowano z biblioteki Keras gotowy model sieci neuronowej ResNet50. Składa się ona z wielu bloków resztkowych (ang. residual blocks), które połączone w całość tworzą rezydualną sieć neuronową. Blok resztkowy jest zbiorem warstw, połączonych w taki sposób, że wyjście z danej warstwy jest dodawane do innej warstwy, znajdującej się głębiej w sieci neuronowej. W sieci rezydualnej istnieją dwa rodzaje bloku resztkowego, różniące się połączeniem omijającym część rezydualną. Pierwszy z nich jako dodatkowe połączenie stosuje przekształcenie tożsamościowe nie zmieniające rozmiaru wyjścia. Taki rodzaj bloku jest korzystny w sytuacji, w której przekształcenie tożsamościowe byłoby optymalne. Wówczas łatwiej jest wyzerować omijaną przez to połączenie część rezydualną niż wytrenować omijane warstwy, aby przyjęły odpowiednie wagi. Drugie możliwe połączenie zawiera warstwę konwolucyjną z następującą po niej warstwą normalizacji wsadu (ang. batch normalization layer). W sieci ResNet50 warstwa konwolucyjna wykorzystana w tym połączeniu ma filtr o rozmiarze 1x1. W każdym bloku resztkowym część rezydualna składa się z następujących po sobie warstw konwolucyjnych o rozmiarach filtra równych kolejno: 1x1, 3x3 oraz 1x1. Po każdej z trzech warstw konwolucyjnych zastosowano normalizację wsadu oraz funkcję aktywacji ReLU, której wykres przedstawiony jest na rysunku 4.11.

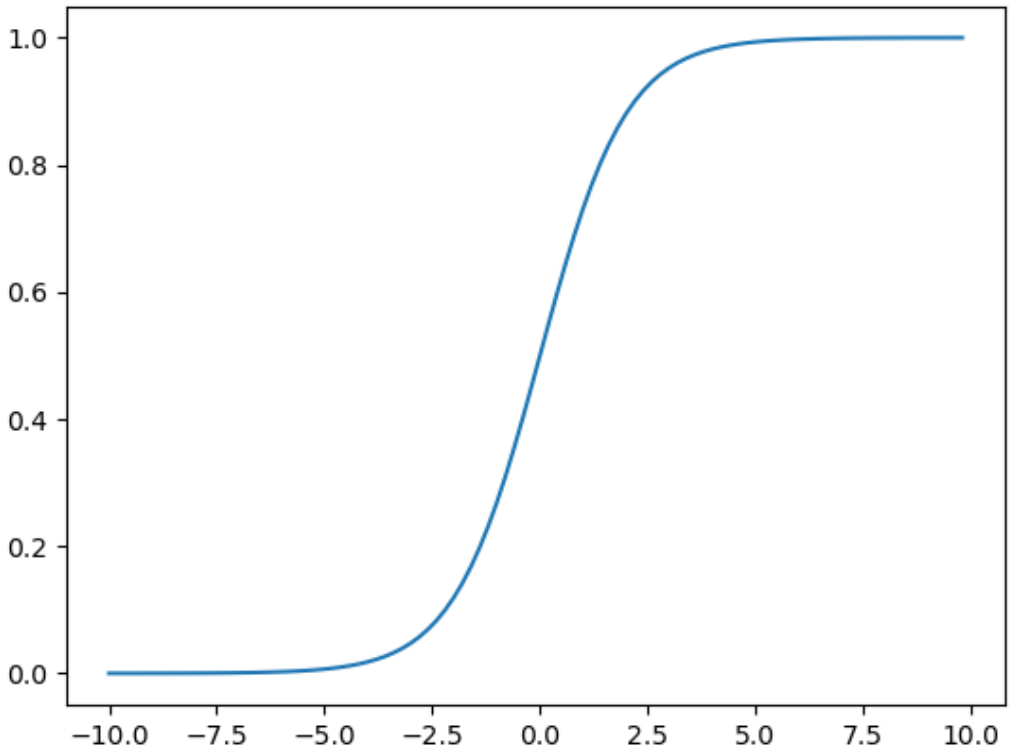


Rys. 4.11. Funkcja aktywacji ReLU.

W miejscu złączenia wyjść części rezydualnej z dodatkowym połączeniem następuje dodanie obu aktywacji oraz zastosowanie dla wyniku sumy nieliniowości przy użyciu funkcji ReLU.

Na wejście sieci neuronowej podano obraz w postaci tensora o rozmiarze 224x224x3. Po warstwie wejściowej znajdowała się pojedyncza warstwa konwolucyjna o wielkości filtra 7x7, po której następowała normalizacja wsadu oraz funkcja ReLU. Liczba filtrów była równa 64, natomiast krok przesuwania filtru (ang. stride) wyniósł 2. Kolejną warstwą był MaxPolling o rozmiarze 3x3 oraz kroku przesuwania filtru równym 2. Następnymi warstwami modelu były powtarzające się bloki resztkowe, różniące się od siebie jedynie liczbą filtrów w warstwach konwolucyjnych części rezydualnych. Każdy blok resztkowy posiadał 3 warstwy konwolucyjne. Liczba filtrów tych warstw wyniosła kolejno: 64, 64 oraz 256 dla pierwszego rodzaju bloku, 128, 128, 512 dla drugiego, 256, 256, 1024 dla trzeciego oraz 512, 512, 2048 dla czwartego bloku. W sieci neuronowej ResNet50 występują kolejno 3 bloki pierwszego typu, 4 bloki drugiego, 6 bloków trzeciego oraz 3 bloki czwartego rodzaju. W każdym rodzaju bloku pierwszy blok miał dodatkowe połączenie zawierające warstwę konwolucyjną, natomiast wszytkie pozostałe bloki danego typu miały połączenia będące przekształceniem tożsamościowym. Po ostatnim bloku sieci ResNet50 dodano warstwę w pełni połączoną o 2 wyjściach, pełniącą role

warstwy klasyfikacyjnej. Z tego względu jako funkcję aktywacji użyto w niej funkcji sigmoid, której wykres przedstawiony jest na ilustracji 4.12.



Rys. 4.12. Funkcja aktywacji sigmoid.

Całkowita liczba parametrów sieci neuronowej wyniosła 23 587 712. Wśród nich 23 534 592 to parametry podlegające uczeniu, natomiast liczba parametrów niepodlegających uczeniu była równa 53 120. Ze względu na ograniczone zasoby obliczeniowe do uczenia zastosowano wielkość wsadu równą 2. W sieci neuronowej użyto optymalizatora korzystającego z algorytmu Adam o współczynniku uczenia 0.001. Jako funkcję straty zastosowana binary crossentropy. Do ewaluacji modelu podczas uczenia oraz walidacji wybrano metrykę dokładności.

Podczas nauki modelu wykorzystano technikę ModelCheckpoint. Dzięki niej po każdej epoce możemy zapisać cały model lub wagi, jakie dobrał w tej epoce, pod warunkiem, że dokładność danego zbioru uległa poprawie. W tym przypadku użyto checkpointa monitorującego dokładność zbioru walidacyjnego. W celu zatrzymania trenowania w momencie, gdy model nie zyskuje lepszych efektów wykorzystano EarlyStopping. Zatrzymuje on uczenie sieci neuronowej w chwili, w której dana metryka nie poprawia się. Wybraną metryką jest funkcja straty

zbioru walidacyjnego. Jeżeli przez 10 epok jej wartość nie spada o więcej niż 0.001 uczenie jest przerywane.

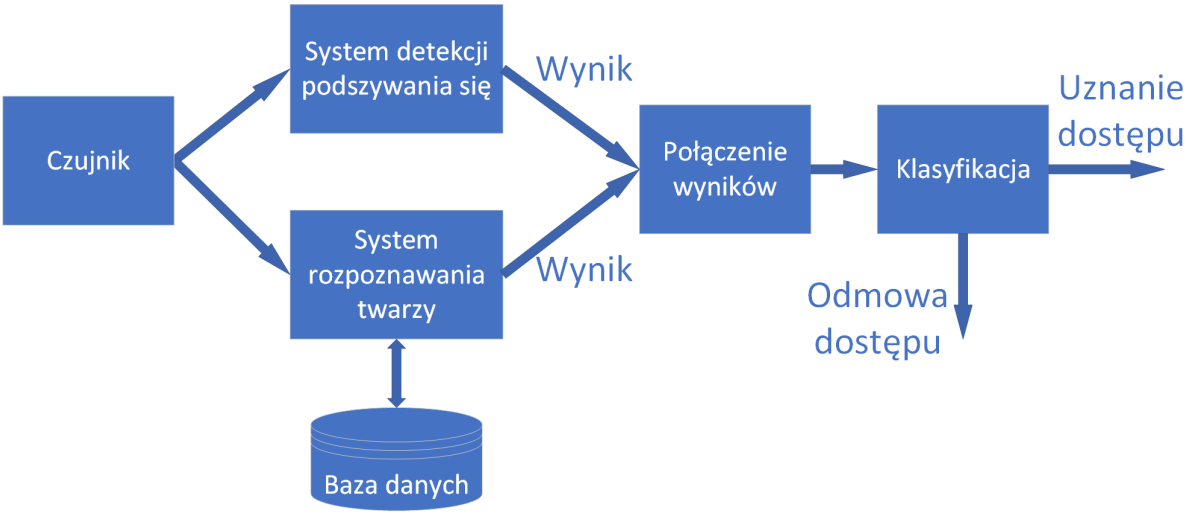
W celu wykorzystania transfer learningu przed rozpoczęciem trenowania sieci neuronowej do warstw sieci ResNet50 załadowano wagi, które zostały dobrane podczas uczenia modelu na zbiorze danych ImageNet [18]. Jest to baza zdjęć stworzona na potrzeby klasyfikacji obiektów w systemach wizyjnych. W sieci ResNet50 pierwotnie została wykorzystywana do rozpoznawania 1000 różnych podmiotów.

4.3. Integracja systemów

Aby system rozpoznawania twarzy odporny na podszywanie się działał poprawnie należy odpowiednio zintegrować ze sobą obie jego części. W praktyce wyróżnione są dwa sposoby połączenia tych podsystemów: szeregowy oraz równoległy. Obie koncepcje mają swoje wady oraz zalety.

4.3.1. Podejście równoległe

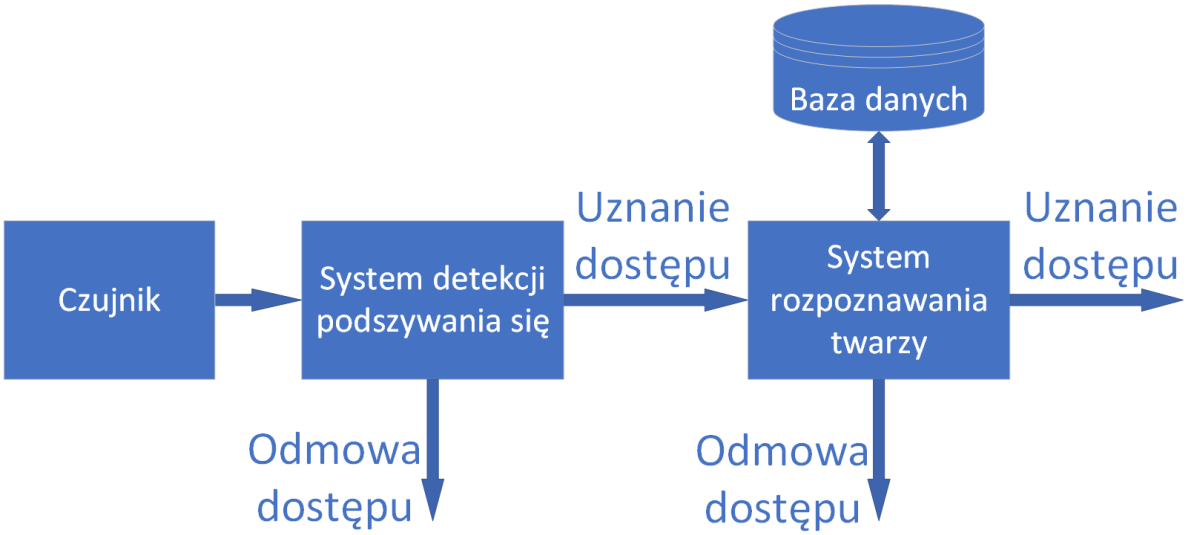
Pierwszą propozycją na połączenie obu funkcjonalności jest schemat równoległy zobrazowany na rysunku 4.13. Zakłada on, że dane biometryczne pobrane od użytkownika trafiają w tym samym momencie do obu podsystemów. Działając równolegle każdy z nich przetwarza dane oraz przekazuje uzyskany wynik na wyjście. Następnie rezultaty otrzymane z podsystemów są łączone i na ich podstawie program decyduje czy dane przyszły od autentycznego użytkownika oraz czy znajduje się on w systemie. Główną zaletą tego podejścia jest krótki czas, po jakim użytkownik zostaje poinformowany o rezultacie rozpoznawania twarzy. Fakt ten może zostać wykorzystany w systemach z możliwościami wykonywania obliczeń równoległych, na przykład wyposażonych w wielordzeniowe procesory.



Rys. 4.13. Podejście równoległe.

4.3.2. Podejście szeregowe

Innym sposobem na połączenie systemu rozpoznawania twarzy z detekcją podszywania się jest schemat szeregowy, przedstawiony na grafice 4.14. W tym podejściu dane użytkownika podawane są na wejście systemu detekcji podszywania się, który przetwarza je oraz podaje wynik swojego działania. Jeżeli wykryje on próbę ataku na system rozpoznawania twarzy nie przekazuje danych wejściowych dalej tylko podejmuje odpowiednie działania dążące do zablokowania dostępu do systemu dla atakującego. Jeżeli natomiast obraz pochodzi od autentycznego użytkownika zdjęcie jest przekazywane dalej do systemu rozpoznawania twarzy, gdzie jest ono porównywane z bazą danych, a wynik tego działania jest podawany na wyjście programu. Zaletą takiego rozwiązania jest fakt, że system rozpoznawania twarzy zostanie wywołany tylko w przypadku, gdy dane wejściowe zostaną dostarczone przez autentyczną osobę. Dzięki temu czas działania programu w przypadku próby oszukania systemu rozpoznawania twarzy będzie krótszy. Wadą tego podejścia jest jednak dłuższy czas działania aplikacji podczas rozpoznawania realnego użytkownika.

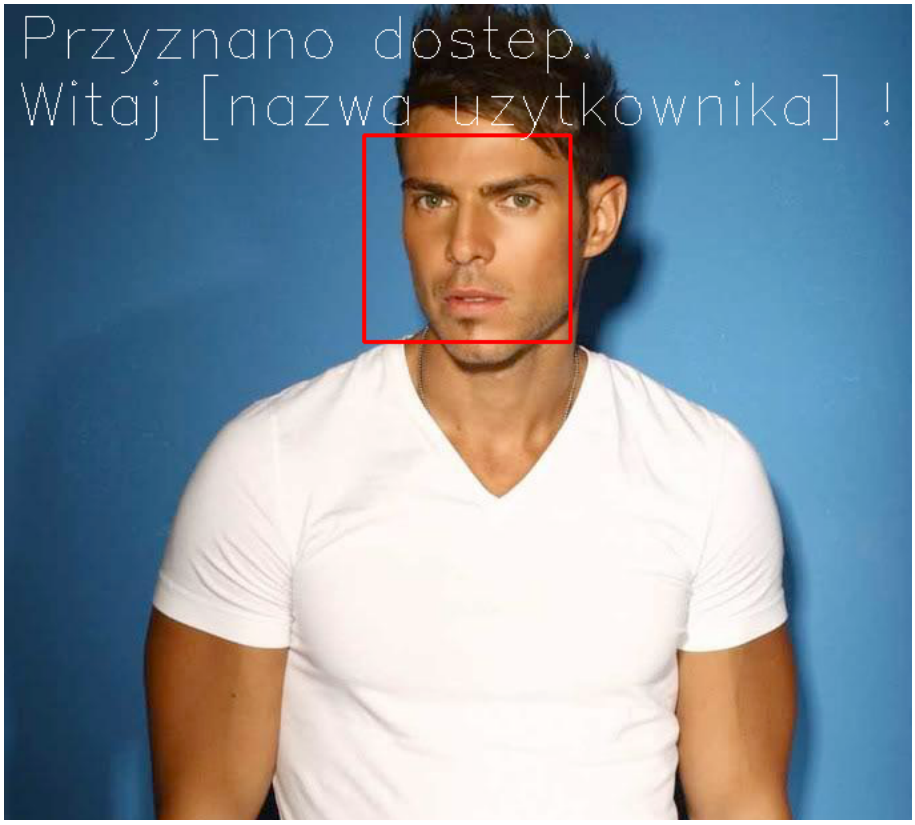


Rys. 4.14. Podejście szeregowe.

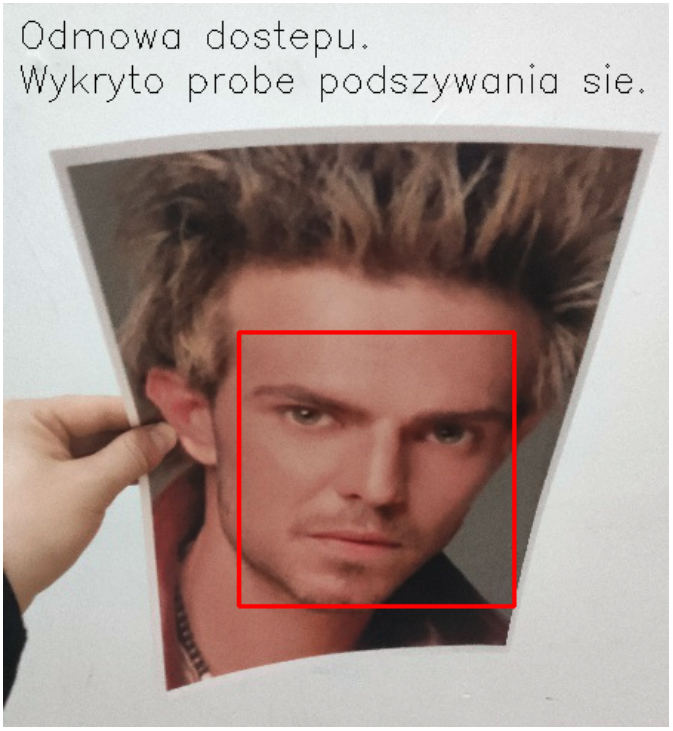
Ze względu na fakt, że w praktycznych zastosowaniach systemów rozpoznawania twarzy przypadki podszywania się nie występują tak często w porównaniu do podawania danych przez autentycznych użytkowników, optymalnym podejściem byłoby podejście równoległe. Jednak ze względu na ograniczone zasoby obliczeniowe w tej pracy zastosowano podejście szeregowe.

4.4. Interfejs użytkownika

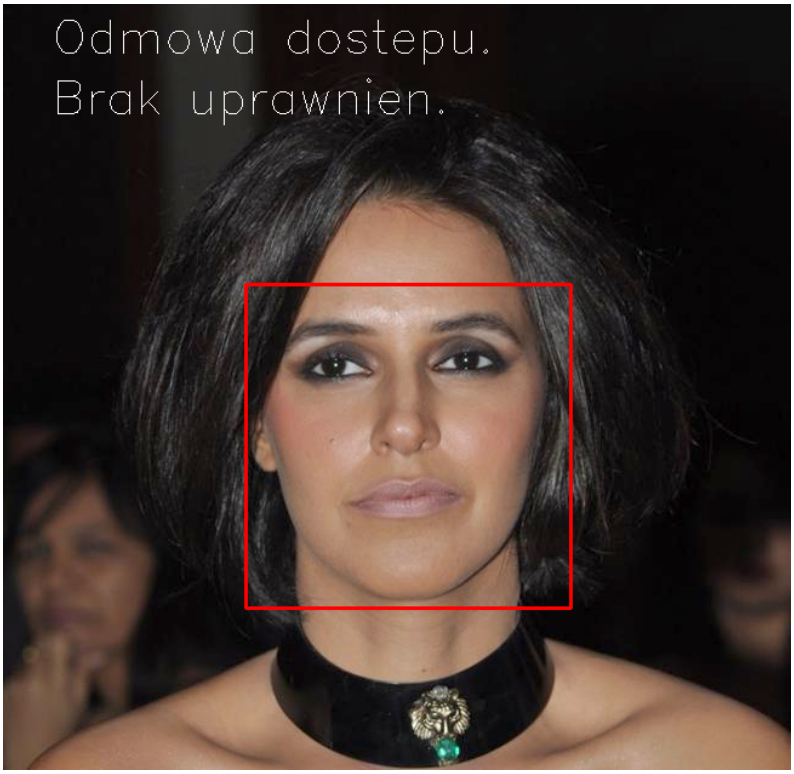
Do stworzenia interfejsu użytkownika skorzystano z biblioteki OpenCV. Podczas inicjalizacji systemu program pobiera wszystkie twarze z bazy danych oraz oblicza ich cechy charakterystyczne, które zapisuje do tablicy. Następnie na ekranie użytkownika ukazuje się widok z kamery komputera. System działa w czasie rzeczywistym - pobiera ramkę z kamery, którą podaje na wejście klasyfikatora podszywania się. Zanim to nastąpi obraz jest poddany odpowiednim operacjom przetwarzania wstępnego. Jeżeli detektor nie wykrył próby podszywania się zdjęcie przekazywane jest do systemu rozpoznawania twarzy. Po przejściu wszystkich jego etapów użytkownik dostaje odpowiednią informację zwrotną od systemu. Wszystkie możliwe odpowiedzi systemu, w zależności od jego wyniku przedstawione są na grafikach 4.15, 4.16 oraz 4.17. W przypadku gdy system nie jest w stanie znaleźć twarzy na obrazie, informuje o tym użytkownika za pomocą analogicznego komunikatu tekstowego jak na ilustracjach poniżej.



Rys. 4.15. Odpowiedź systemu na poprawną próbę uzyskania dostępu przez osobę znajdującą się w bazie danych.



Rys. 4.16. Odpowiedź systemu na próbę podszywania się.

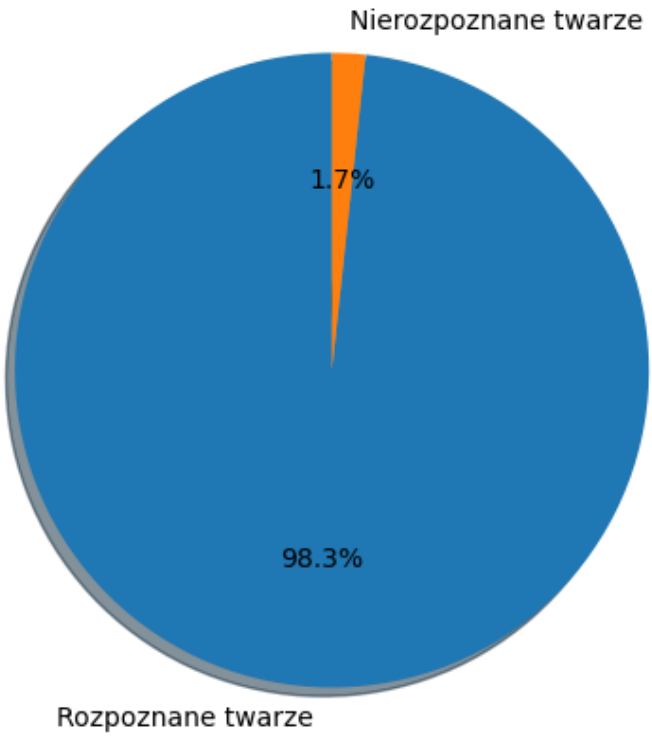


Rys. 4.17. Odpowiedź systemu na próbę uzyskania dostępu przez osobę nie znajdującą się w bazie danych.

5. Testy

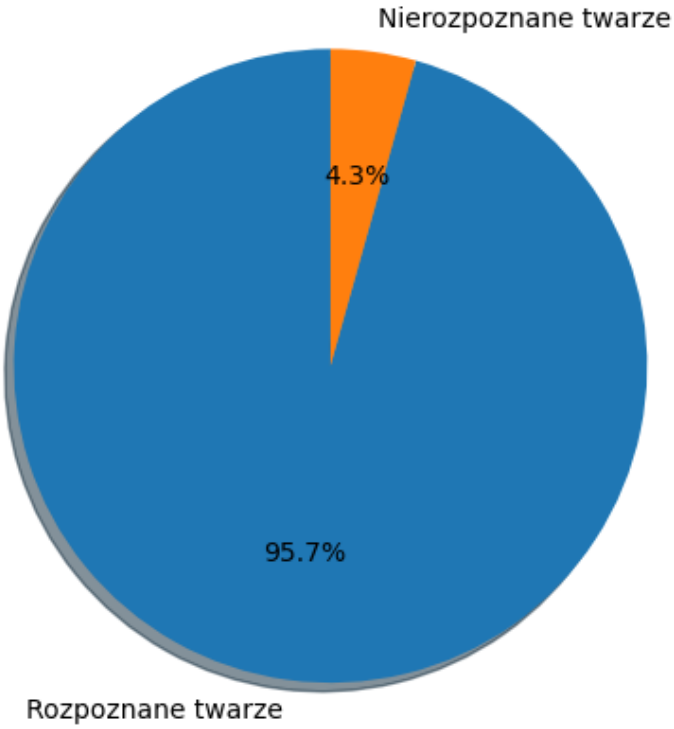
5.1. Ewaluacja podsystemu rozpoznawania twarzy

Aby sprawdzić poprawność działania aplikacji przeprowadzono osobne testy na obu podsystemach. Do sprawdzenia rozpoznawania twarzy użyto niewielkiej części zbioru danych CelebA-Spoof. W tym celu zbadano wyniki działania systemu dla zdjęć 5-ciu wybranych osób. Dla każdej osoby losowo wybrano jedno autentyczne zdjęcie, które dodano do folderu pełniącego rolę bazy danych. Następnie na wejście podsystemu rozpoznawania twarzy podano pozostałe zdjęcia 5-ciu wybranych osób, przedstawiające zarówno autentyczne obrazy, jak i próby podszywania się. Dodatkowo wylosowano pięć innych, nie będących w bazie danych osób, dla których wybrano wszystkie autentyczne zdjęcia, w celu sprawdzenia działania systemu dla nieznanych osób. Jak się okazało aplikacja nie rozpoznała ani jednej twarzy nie znajdującej się w bazie danych. Ze wszystkich autentycznych osób znajdujących się w bazie danych tylko dwie twarze nie zostały odpowiednio rozpoznane. Wyniki rozpoznawania osób znajdujących się w systemie przedstawione są na wykresie 5.1.



Rys. 5.1. Wyniki rozpoznawania twarzy dla zdjęć autentycznych użytkowników znajdujących się w bazie danych.

W celu sprawdzenia, jak system rozpoznawania twarzy radziłby sobie bez dodatkowej detekcji podszywania się dla wszystkich 5-ciu osób przeprowadzono testy na obrazach będących próbą ataku na aplikację. Jak się okazało w prawie 96% przypadków twarz została rozpoznana jako znajdująca się w bazie danych. Wyniki tego eksperymentu zilustrowano na wykresie 5.2.

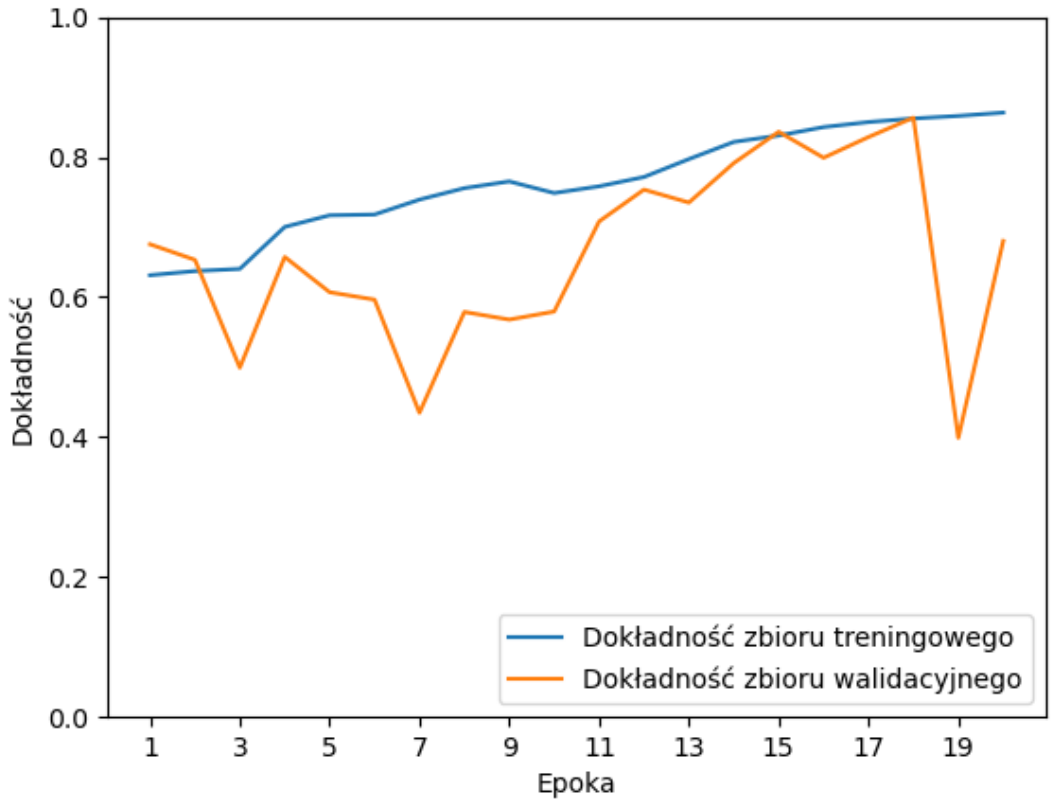


Rys. 5.2. Wyniki rozpoznawania twarzy dla zdjęć przedstawiających próbę podszywania się.

Uzyskany wynik pokazuje, jak bardzo potrzebny w systemie rozpoznawania twarzy jest odpowiedni detektor podszywania się.

5.2. Testowanie detekcji podszywania się

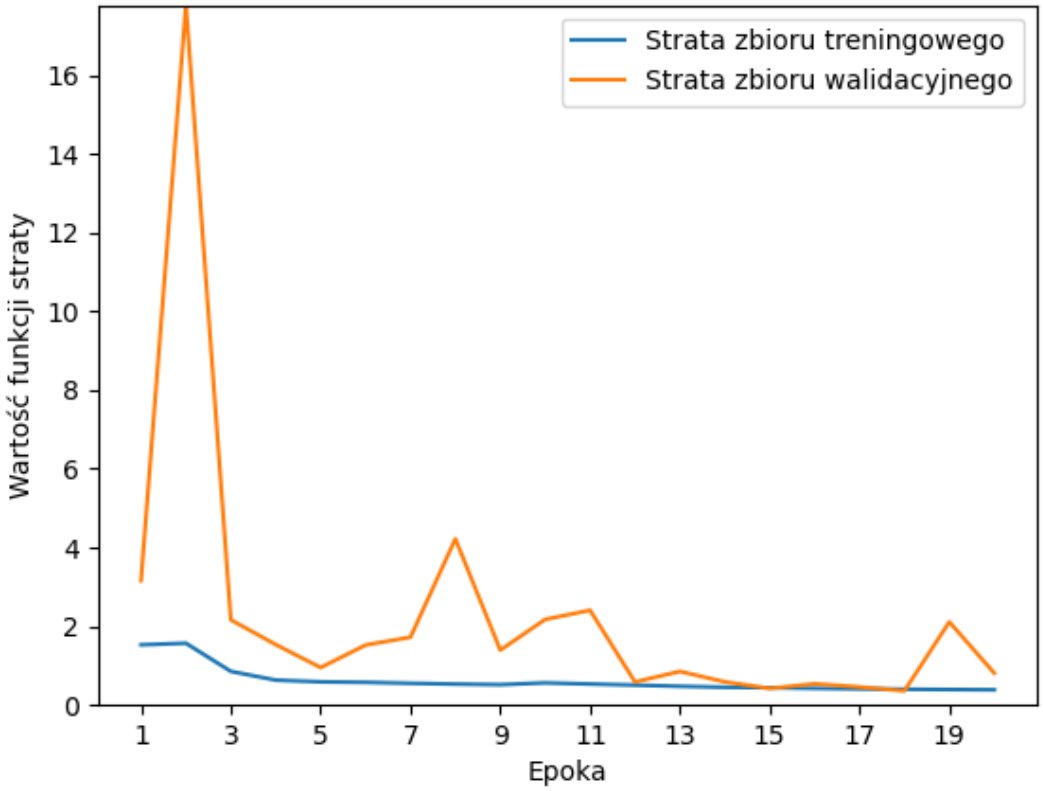
W celu przeprowadzenia walidacji detekcji podszywania się dane początkowo podzielono na zbiór treningowy oraz testowy. Pierwszy z nich zawierał 490 000 obrazków, zbiór walidacyjny natomiast posiadał 67 000 danych. Dane zostały podzielone tak, aby nie było w nich zdjęć tej samej osoby. Niestety proces uczenia sieci neuronowej z tak dużą ilością parametrów na tak dużej ilości danych był niemożliwy, ze względu na zbyt wielką złożoność obliczeniową trenowania. W związku z tym zdecydowano się zawęzić zbiory do 32 000 obrazów dla zbioru uczącego oraz 8000 zdjęć dla zbioru walidacyjnego. Cały proces uczenia sieci neuronowej trwał 20 epok. Wykres przedstawiający zmiany dokładności modelu na zbiorze treningowym oraz zbiorze walidacyjnym jest przedstawiony na grafice 5.3.



Rys. 5.3. Zmiana dokładności modelu w zależności od epoki.

Sieć neuronowa w każdej epoce zwiększała nieznacznie swoją dokładność na zbiorze treningowym, zaczynając od skuteczności niewiele większej niż 60% do około 86%. Jak widać na wykresie zmiany dokładności modelu na zbiorze walidacyjnym nie były takie stabilne, jednak widoczny jest ogólny trend wzrostowy. W ostatnich dwóch epokach dokładność nie zwiększyła się, co może oznaczać, że sieć nie jest w stanie już dobrać lepszych parametrów. Jednak biorąc pod uwagę fakt, że detekcja podszywania się w systemach rozpoznawania twarzy nie jest typowym problemem klasyfikacji, wcale nie musi to oznaczać, że model nie jest w stanie uzyskać lepszych wyników przy zwiększeniu ilości epok. Dodatkowo, pozytywnie na poprawę jakości detektora mogłoby wpłynąć zwiększenie zbiorów danych. Najlepszy wynik sieć uzyskała w 18 epoce, w której jej dokładność na zbiorze testowym wyniosła 85.6%.

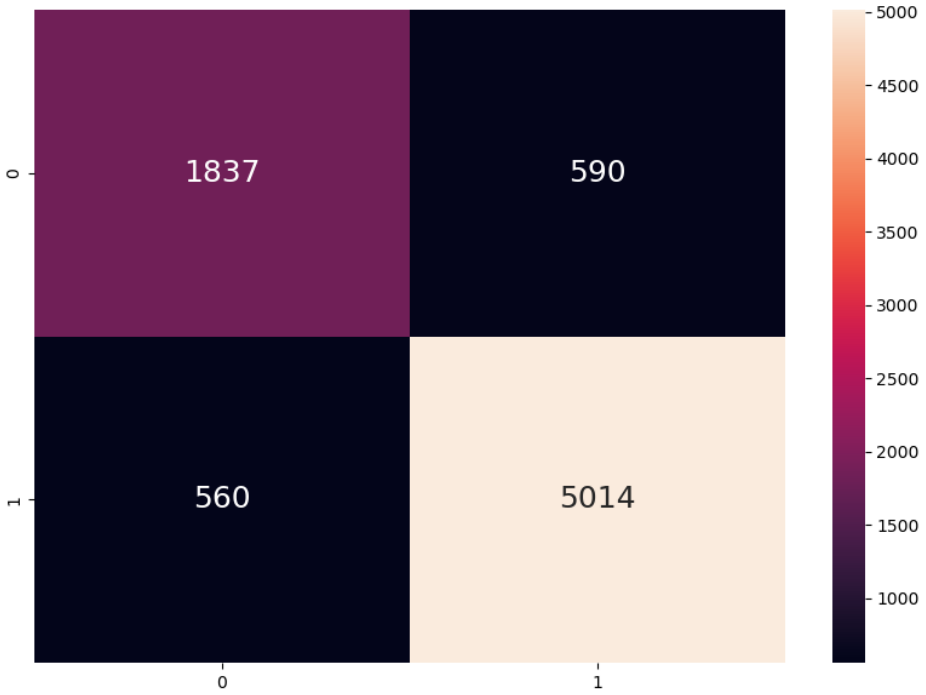
W celu wizualnej oceny zmiany błędu klasyfikatora stworzono wykres 5.4, który opisuje wartości funkcji straty modelu w zależności od numeru epoki.



Rys. 5.4. Zmiana wartości funkcji straty w zależności od epoki.

Na powyższym wykresie można zaobserwować, że model cały czas zmniejszał wartość funkcji straty. Wyjątkiem od tej reguły jest druga epoka, w której jej wartość była zaskakująco duża. Mogło to być spowodowane błędnym doborem parametrów na początkowym etapie uczenia przez sieć neuronową. W kolejnych epokach strata była już zdecydowanie mniejsza, a jej wartość dla zbioru walidacyjnego osiągnęła minimum w epoce 18 i wyniosła 0.36.

Ze zbioru testowego 1837 obrazów zostało poprawnie sklasyfikowane jako autentyczny użytkownik, natomiast 560 realnych zdjęć zostało nie prawidłowo oszacowane jako próba podszywania się. Z drugiej strony 5014 zdjęć zostało stosownie rozpoznane jako próba podszywania się, natomiast 590 zdjęć przedstawiających atak na system rozpoznawania twarzy było nie poprawnie oznaczone jako zdjęcia autentycznych osób. Wartości te zostały przedstawione w macierzy pomyłek na ilustracji 5.5.



Rys. 5.5. Macierz pomyłek.

Klasa zerowa odpowiada w niej zdjęciu autentycznego użytkownika, natomiast klasa pierwsza jest atakiem na system rozpoznawania twarzy. Jak widać modelowi zdarza się źle sklasyfikować obrazy, jednak jego pomyłki są stosunkowo rzadkie. Dodatkowo sieć popełniała błędy w obu klasach, co może oznaczać, że jej dalszy trening mógłby przynieść pozytywne efekty.

6. Podsumowanie

Systemy rozpoznawania twarzy bazujące na algorytmach uczenia maszynowego są efektywne oraz stosunkowo łatwe w implementacji, lecz co za tym idzie są bardzo podatne na wpływ czynników zewnętrznych. Zmienne warunki otoczenia, różnorodne oświetlenie czy zmiany ekspresji twarzy to tylko jedne z wielu aspektów mogących negatywnie wpływać na działania takich aplikacji. Pomimo tego aktualnie tworzone systemy potrafią osiągać bardzo zadowalające efekty. Wymaga to jednak wykorzystania stosunkowo dużych zasobów, zarówno jeśli chodzi o dane, jak i jednostki obliczeniowe. Stworzony system rozpoznawania twarzy zdecydowanie nie byłby odpowiedni do aplikacji komercyjnych, jednak uzyskał satysfakcjonujące rezultaty.

Istnieje wiele funkcjonalności oraz usprawnień, które można dodać do stworzonego systemu rozpoznawania twarzy. Pierwszym, jakie należałoby rozważyć jest poprawa modelu uczenia maszynowego odpowiedzialnego za detekcję podszywania się. Zwiększenie liczby parametrów, czy dodanie dodatkowych, działających równolegle sieci neuronowych i odpowiednia analiza ich wyników mogłoby zwiększyć dokładność całego systemu. Dodatkowo rozważenie wpływu atrybutów geometrycznych, takich jak mapa głębi (ang. depth map) czy wcześniej omawianych parametrów semantycznych, przedstawionych na grafikach 4.9 oraz 4.10 mogłoby wpłynąć pozytywnie na działanie programu.

Innym kierunkiem rozwoju jest możliwość rozbudowania interfejsu użytkownika. Na ten moment można go wywołać jedynie z komputera, na którym jest odpowiednio skonfigurowane środowisko programistyczne. Do implementacji realnego systemu kod programu należałoby przenieść na urządzenia docelowe potencjalnego użytkownika. Takie aplikacje najczęściej stosowane są w telefonach mobilnych, wobec czego dobrym pomysłem jest wdrożenie oprogramowania na takie systemy operacyjne jak Android czy iOS. W związku z tym, wymagane byłoby również utworzenie oraz utrzymanie odpowiedniej bazy danych do przechowywania informacji o użytkownikach.

Na poprawę działania systemu powinno wpłynąć również dodanie oraz analiza innych scenariuszy testowych. Jednym z nich może być ewaluacja detektora podszywania się na różnych urządzeniach. Takie testy zakładałyby uczenie modelu na zdjęciach pobranych z czujników o niskiej, średniej oraz wysokiej jakości oraz walidację wytrenowanej sieci na odpowiednich

zbiorach testowych. Powyższe procedury testowe powinny znacząco zniwelować wpływ czynników zewnętrznych na wynik działania systemu rozpoznawania twarzy.

Bibliografia

[1] „*Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2016/679 z dnia 27 kwietnia 2016 r. w sprawie ochrony osób fizycznych w związku z przetwarzaniem danych osobowych i w sprawie swobodnego przepływu takich danych oraz uchylenia dyrektywy 95/46/WE*". Parlament Europejski, Rada Unii Europejskiej. 2016.

[2] Kaspersky Lab. All Rights Reserved. „*What is Facial Recognition – Definition and Explanation*". URL: *https://www.kaspersky.com/resource-center/definitions/what-is-facial-recognition* (term. wiz. 2021).

[3] „*ISO/IEC 19794-5*". International Organization for Standardization. 2005.

[4] Anil K. Jain David Zhang. „*Advances in Biometrics*". W: *International Conference, ICB 2006*. Hong Kong, China, sty. 2006, s. 183.

[5] Jie Yang i Alexander H. Waibel. „*A real-time face tracker*". W: *Proceedings Third IEEE Workshop on Applications of Computer Vision. WACV'96* (1996), s. 142–147.

[6] M. Turk i A. Pentland. „*Face recognition using eigenfaces*". W: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Czer. 1991, s. 586–591.

[7] T. Kanade. „*Picture processing system by computer complex and recognition of human faces.*" Prac. dokt. Kyoto University, 1973.

[8] Kamran Etemad i Rama Chellappa. „*Discriminant analysis for recognition of human face images*". W: *J. Opt. Soc. Am. A* 14.8 (1997), s. 1724–1733.

[9] Timo Ahonen, Abdenour Hadid i Matti Pietikäinen. „*Face Recognition with Local Binary Patterns*". W: *Computer Vision - ECCV 2004*. T. 3021. Maj 2006, s. 469–481.

[10] Antitza Dantcheva, Cunjian Chen i Arun Ross. „*Can facial cosmetics affect the matching accuracy of face recognition systems?*" W: *2012 IEEE Fifth international conference on biometrics*. Wrz. 2012, s. 391–398.

[11] Abdenour Hadid i in. „*Biometrics Systems Under Spoofing Attack: An evaluation methodology and lessons learned*". W: *Signal Processing Magazine, IEEE* 32 (wrz. 2015), s. 20–30.

[12] Sébastien Marcel i in. „*Handbook of Biometric Anti-Spoofing*". Springer, Cham, 2019.

[13] André Anjos, Murali Chakka i Sébastien Marcel. „*Motion-Based Counter-Measures to Photo Attacks in Face Recognition*". W: *Biometrics, IET* 3 (wrz. 2014), s. 147–158.

[14] Vahid Kazemi i Josephine Sullivan. „*One millisecond face alignment with an ensemble of regression trees*". W: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, s. 1867–1874.

[15] Florian Schroff, Dmitry Kalenichenko i James Philbin. „*FaceNet: A unified embedding for face recognition and clustering*". W: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (czer. 2015).

[16] Yuanhan Zhang i in. „*CelebA-Spoof: Large-Scale Face Anti-Spoofing Dataset with Rich Annotations*". W: *European Conference on Computer Vision (ECCV)*. 2020.

[17] Sefik Ilkin Serengil i Alper Ozpinar. „*HyperExtended LightFace: A Facial Attribute Analysis Framework*". W: *2021 International Conference on Engineering and Emerging Technologies (ICEET)*. IEEE. 2021.

[18] Jia Deng i in. „*ImageNet: A large-scale hierarchical image database*". W: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, s. 248–255.