Marcin Plata

Zastosowanie zaawansowanych metod sztucznej inteligencji do wybranych problemów bezpieczeństwa informacji

Rozprawa doktorska

Promotor:
prof. dr hab. inż. Marek Klonowski

Promotor pomocniczy:
dr inż. Piotr Syga

Instytut Podstaw Informatyki
Polskiej Akademii Nauk

Warszawa 2023

Oświadczenie autora:
Oświadczam, że niniejsza rozprawa doktorska jest moją pracą własną.

1 marca 2023 .....

Marcin Plata

Oświadczenie promotora:
Rozprawa doktorska jest gotowa do recenzji.

1 marca 2023 .....

prof. dr hab. inż. Marek Klonowski

Oświadczenie promotora pomocniczego:
Rozprawa doktorska jest gotowa do recenzji.

1 marca 2023 .....

dr inż. Piotr Syga

Spis treści

Streszczenie

W niniejszej rozprawie przedstawiamy metody zaliczane do szeroko pojętej sztucznej inteligencji, które z sukcesem aplikujemy do wybranych problemów z obszaru bezpieczeństwa informacji. Dotykamy dwóch głównych zagadnień – znakowania wodnego obrazów oraz biometrii. W przypadku biometrii pochylamy się nad trzema różnymi tematami – standardowej autoryzacji, odporności na atak podszywania się oraz ochrony prywatności. Każdy problem jest przedstawiony bazując na innych cechach biometrycznych, tj. odpowiednio konturze dłoni, głosie, mimice twarzy.

Trudność w tworzeniu rozwiązań do znakowania wodnego wynika głównie z konieczności uodpornienia opracowywanych metod na ataki oraz algorytmy kompresji, takie jak JPEG. Sam dobór ataków może znacząco wpłynąć na architekturę rozwiązania (np. operacja przycinania krawędzi obrazu zaburza jego strukturę przestrzenną, znacząco utrudniając odczytanie znaku wodnego, przez co jest często pomijana w pracach badawczych). W rozprawie prezentujemy nowe podejście oparte na trzech sieciach neuronowych – enkoder, dekoder oraz dyskryminator, które rozszerzamy o autorską metodę rozprzestrzeniania przestrzennego znaku wodnego, która wyraźnie zwiększa odporność na część ataków oraz kompresję JPEG w stosunku do ostatnich rozwiązań, a przy tym utrzymuje wysoki poziom odporności na atak przycinania. Następnie dodajemy do enkodera dodatkowy komponent, który przetwarza znak wodny niezależnie od obrazu wejściowego, tym samym uzyskując wyższą odporność oraz jego jakość, nie zwiększając rzeczywistego czasu nakładania znaku wodnego. Następnie proponujemy podział ataków na grupy, zależnie od operacji przetwarzania obrazu i pokazujemy, że w celu osiągnięcia wysokiej ogólnej odporności wymagane jest uwzględnienie ataków ze wszystkich grup. Zauważamy również fakt, że w rzeczywistych warunkach, treści oznaczone konkretną metodą znakowania wodnego występują stosunkowo rzadko, a tym samym istnieje zwiększone ryzyko błędnego oczytania znaku wodnego, prowadzące do fałszywego oskarżenia o piractwo. Ten problem jest przez nas adresowany poprzez wykorzystanie dyskrymina-

tora do wstępnego określenia istnienia znaku wodnego, następnie pokazujemy skuteczność naszego podejścia na podstawie opracowanych metryk.

Obecnie rozwiązania biometryczne są wykorzystywane na szeroką skalę, również jako metody autoryzacji do systemów wymagających szczególnych środków ostrożności, np. aplikacje bankowe. W rozprawie podejmujemy się tematu ochrony prywatności dla metod biometrycznych na przykładzie cech twarzy. W naszym rozwiązaniu staramy się ukryć typowe statyczne cechy i przetwarzać jedynie przesunięcia punktów charakterystycznych (mimikę). Takie podejście znacząco ogranicza możliwości rekonstrukcji twarzy z punktów charakterystycznych, np. w przypadku wycieku bazy danych. W celu zwiększenia prywatności dodatkowo normalizujemy cechy oraz proponujemy podejście, w którym odrzucamy najbardziej skorelowane cechy mimiki twarzy z cechami statycznymi. Nasze rozwiązanie zostało oparte na kilku klasycznych metodach uczenia maszynowego, takich jak maszyna wektorów nośnych czy drzewa losowe. Uzyskane wyniki pozwalają uznać naszą metodę jako tzw. biometrię słabą, która sprawdzi się również jako metoda wspomagająca autoryzację. Zastosowanie podejścia, w którym użytkownik musi wypowiedzieć określoną frazę w celu autoryzacji, pozwala rozważać nasze rozwiązanie jako biometrię usuwalną.

Metody do detekcji ataku spoofingowego są kluczowymi elementami systemów biometrycznych. Jednym z najbardziej skutecznych, a zarazem najtrudniejszym do wykrycia, atakiem jest powtórne odtworzeniowe nagranego głosu. Podejmujemy się stworzenia metody, która jest w stanie wykryć tego typu atak. Ze względu na specyfikę metody wykorzystujemy małe sieci neuronowe – LCNN oraz bayesowską sieć neuronową – umożliwiające szybką detekcję na podstawie kilkusekundowej próbki dźwiękowej. W celu wytrenowania modeli wykorzystujemy autorską metodę walidacji krzyżowej, uwzględniającą podział na ataki. Dodatkowo stosujemy szereg technik regularyzacyjnych, które umożliwiają wyuczenie rozwiązania wykrywającego również ataki spoza zbioru treningowego.

W przypadku metody do autoryzacji biometrycznej bazującej na konturze dłoni, pokazujemy, że wykorzystanie klasycznych metod przetwarzania obrazu jest wystarczające do stworzenia wydajnego obliczeniowo rozwiązania o wysokiej skuteczności. Nasz potok przetwarzania składa się z kilku szybkich algorytmów, takich jak rozmycie gaussowskie, liczenie otoczki wypukłej, wpisanie trójkąta o największym obwodzie w kontur. Takie podejście pozwala na wykorzystanie standardowego i taniego urządzenia, które nie wymaga GPU. Nasza metoda przyjmuje próbki pobrane skanerem biurowym, co dodatkowo pozwala na redukcję kosztów wdrożenia systemu.

Streszczenie w języku angielskim (Abstract)

In this dissertation, we present methods categorized as artificial intelligence, which we successfully apply to selected problems in the field of information security. We touch on two main issues – image watermarking and biometrics. Regarding biometrics, we delve into three different topics: standard authentication, spoofing attack robustness, and privacy protection. Each problem is presented based on different biometric features, i.e. hand contour, voice, and facial mimic, respectively.

The difficulty in creating watermarking solutions mainly stems from the need to make methods robust against attacks and compression algorithms such as JPEG. The choice of attacks could significantly affect the solution's architecture required to develop (e.g., the cropping operation disrupts the image spatial structure, significantly hindering the watermark extraction, which is often overlooked in research). We present a new approach based on three neural networks – encoder, decoder, and discriminator, which we extend with our novel method of watermark spatial spreading, which significantly increases robustness against some attacks and JPEG compression compared to recent solutions while keeping high robustness against the cropping attack. We expand the encoder with an additional component that processes the watermark independently of the input image, thus achieving higher robustness and quality without increasing the actual watermark encoding time. Then we classify attacks into groups depending on image processing operations and show that attacks from all groups should be considered in order to achieve high overall robustness. We also note that in real conditions, the content generated with a particular watermarking method occurs relatively rarely, thus increasing the risk of the watermark being wrongly extracted and falsely accusations of piracy. We address this issue by using the discriminator to preliminarily determine the existence of the watermark and we demonstrate the effectiveness of our approach based on proposed metrics.

Currently, biometric solutions are widely used, including authorization methods for systems that require special precautions, such as banking applications. In this dissertation, we address the issue of privacy protection for biometric methods using facial features. In our solution, we attempt to hide typical static features and process only the movements of characteristic points (facial expressions). Our approach significantly limits the possibility of reconstructing the face from characteristic points, for example in the case of a database leak. To increase privacy protection, we also normalize the features and propose an approach in which we reject the most correlated facial expressions with static features. Our solution is based on several classical machine learning methods, such as support vector machines and random forests. The obtained results allow us to consider our method as weak biometrics, which would also operate as a supporting method for authorization. A use case in which a user speaks a specific phrase for authorization allows our solution to be considered as cancelable biometrics.

Methods for detecting spoofing attacks are crucial elements of biometric systems. One of the most effective attacks and at the same time most difficult to detect is a replay of a recorded voice. We aim to create a method that is able to detect this type of attack. Due to the specificity of the method application, we use small neural networks – LCNN and Bayesian neural network – enabling fast detection based on a few seconds of an audio sample. To train the models, we use our attack-out cross-validation method that takes into account the division into attacks. Additionally, we apply several regularization techniques that enable to train a model that detects attacks non-included in a training set.

In the case of a biometric authorization based on hand contours, we show that using classical image processing methods is satisfactory to create a computationally efficient solution with high accuracy. Our processing pipeline consists of several fast algorithms, such as Gaussian blur, convex hull calculation, and finding the largest triangle inscribing into a contour. Our approach allows the use of a standard and low-cost device that does not require a GPU. Our method accepts samples taken with an office scanner, which additionally allows for the cost reduction of implementing the system.

Rozdział 1

Zawartość i struktura pracy

Niniejsza rozprawa dotyczy wykorzystania rozwiązań sztucznej inteligencji do wybranych problemów bezpieczeństwa informacji. W kolejnych rozdziałach pracy pokazujemy różne metody zaliczane do sztucznej inteligencji, tj. klasycznie algorytmy uczenia maszynowego (ang. *machine learning*), uczenie głębokie (ang. *deep learning*) oraz metody widzenia komputerowego (ang. *computer vision*) zastosowane w konkretnych zagadnieniach aplikacyjnych. Rozdziały 2 oraz 3 dotyczą znakowania wodnego obrazów, które to realizujemy z wykorzystaniem konwolucyjnych sieci neuronowych uczonych w ramach układu treningowego *enkoder-noiser-dekoder* oraz z użyciem tzw. *treningu adwersarialnego*. Rozdział 4 opisuje rozwiązanie do identyfikacji biometrycznej bazującej na mimice twarzy, którą charakteryzuje wyższy poziom ochrony prywatności w stosunku do standardowych rozwiązań biometrycznych opartych na analizie statycznych cech twarzy. W rozdziale 5 przedstawiamy rozwiązanie wykrywające atak polegający na głosowym podszywaniu się (ang. *spoofing*) do którego wykorzystaliśmy tzw. lekkie sieci neuronowe oraz sieci bayesowskie, które cechują się relatywnie krótkim czasem wnioskowania. Rozdział 6 został poświęcony rozwiązaniu do identyfikacji biometrycznej na podstawie konturu dłoni z wykorzystaniem szeregu metod zaliczanych do widzenia komputerowego oraz działającym na niskobudżetowych urządzeniach. W rozdziale 7 krótko podsumowujemy rezultaty opisane w pracy.

Większość treści przedstawionej w rozprawie jest oparta na wcześniej opublikowanych artykułach naukowych. Poniżej przedstawiamy listę tych prac wraz z krótkim opisem najistotniejszych wyników w nich zawartych:

1. [PS20] *Robust Spatial-Spread Deep Neural Image Watermarking*, praca zrealizowana wraz z Piotrem Sygą i zaprezentowana na 2020 IEEE 19th

International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). Głównym wynikiem opisanym w pracy jest metoda rozmieszczenia przestrzennego klucza identyfikującego kopię obrazu oraz adaptacja potoku (ang. *pipeline*) treningowego, w szczególności opracowanie nowej funkcji kosztu *średnia-wariancja* pozwalającej na efektywny trening nowego podejścia.

2. [PSK19] *How to Save Your Face: a Facial Recognition Method Robust Against Image Reconstruction*, współautorami pracy są Piotr Syga oraz Marek Klonowski. Artykuł został zaprezentowany na 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS). Celem badań zaprezentowanych w pracy było stworzenie metody do identyfikacji biometrycznej wykorzystującej cechy twarzy, która zapewnia wyższy poziom prywatności danych w stosunku do istniejących metod. Cel ten osiągnięto dzięki wykorzystaniu cech biometrycznych mimiki twarzy, a samo rozwiązanie można zaliczyć do tzw. słabych biometrii.

3. [BKM$^+$19] *Robust Bayesian and Light Neural Networks for Voice Spoofing Detection*, napisana wspólnie z Radosławem Białobrzeskim, Michałem Kośmiderem, Mateuszem Matuszewskim i Alexandrem Rakowskim oraz zaprezentowana na konferencji INTERSPEECH 2019. Artykuł dotyczy efektywnej czasowo metody zapobiegającej atakom spoofingowym dla biometrii głosu opartej na lekkiej sieci konwolucyjnej oraz sieci bayesowskiej. W pracy zaproponowano także nowe podejście do modelowania sieci – *attack-out cross-validation*. Dodatkowo przeprowadzono analizę danych, która pozwoliła na określenie poziomu trudności wykrycia ataku w zależności od jego parametrów.

4. [KPS18] *User authorization based on hand geometry without special equipment*, praca napisana wraz z Markiem Klonowskim oraz Piotrem Sygą i opublikowana w czasopiśmie Pattern Recognition. W pracy zaprezentowano szybką metodę identyfikacji, wykorzystującą cechy biometryczne konturu dłoni. Metoda została oparta na efektywnych czasowo algorytmach przetwarzania obrazu (w szczególności metodach geometrii obliczeniowej), takich jak liczenie otoczki wypukłej, szukanie najodleglejszego punktu oraz wpisanie trójkąta o największym obwodzie. Atutem rozwiązania jest możliwość wykorzystania zdjęć pobranych za pomocą standardowego skanera biurowego.

W rozprawie wykorzystano również elementy z artykułu *DDD-WM: Robust deep neural network–based image watermarking with double detector-discriminator scheme* [PS23] napisanego przez autora rozprawy wraz z Piotrem Sygą. Praca została zgłoszona do druku w czasopiśmie Expert Systems with Applications i na dzień złożenia rozprawy była w trakcie recenzji. Przeddruk publikacji można znaleźć w serwisie arXiv [PS23]. W pracy została przedstawiona koncepcja wykorzystania dyskryminatora jako dodatkowego komponentu pozwalającego na określenie istnienia znaku wodnego w obrazie, a tym samym zwiększenia dokładności detekcji w przypadku, w którym znakowane treści są znajdowane stosunkowo rzadko, co odpowiada rzeczywistemu scenariuszowi pracy takiego systemu. Zaprezentowano również rozszerzenie potoku treningowego z artykułu [PS20] o komponent, który odpowiednio przekształca klucz, nie zwiększając przy tym czasu przetwarzania z koniecznym dostępem do obrazu wejściowego.

Autor rozprawy był wspierany w ramach grantów:

1. „Ukrywanie informacji i prywatność w systemach (głównie) rozproszonych”, OPUS 2018/29/B/ST6/02969 (Narodowe Centrum Nauki),

2. „Vestigium – platforma wykorzystująca inteligentne algorytmy znakowania treści wideo umożliwiające szybką identyfikację i blokowanie źródła nielegalnych transmisji”, Program Operacyjny Inteligentny Rozwój POIR.01.01.01-00-1032/18 (Narodowe Centrum Badań i Rozwoju), realizowany we współpracy z start-upem Vestigit sp. z o.o.

Artykuł [BKM+19] został opublikowany jako wynik prac badawczych prowadzonych przez autora w zespole Audio Intelligence w firmie Samsung R&D Institute Poland.

Wkład naukowy wniesiony przez autora rozprawy w czterech wymienionych pracach [PS20, PSK19, KPS18, PS23] można uznać za przodujący, a samego autora należy wymienić jako głównego spośród wszystkich współautorów. W pracy [PS20] autor zaprezentował rozszerzenie istniejących rozwiązań do znakowania wodnego o nowe elementy, tj. metodę rozmieszczenia przestrzennego klucza oraz potok trenowany z wykorzystaniem autorskiej funkcji kosztu *średnia-wariancja*. Następnie autor zaproponował podział na grupy ataków. Był również odpowiedzialny za przeprowadzenie skutecznego treningu opracowanych sieci neuronowych, w tym opracowanie różniczkowalnej aproksymacji algorytmu kompresji JPEG. Artykuł [PS23] opisuje dwie znaczące nowości wniesione przez autora rozprawy – rozszerzenie sieci do nakładania znaku wodnego o *adapter* oraz opracowa-

nie potoku treningowego pozwalającego na wykorzystanie dyskryminatora do określenia istnienia klucza w obrazie. W pracy [PSK19] autor odpowiadał głównie za stworzenie metody do identyfikacji biometrycznej poprzez dobór kolejnych składowych rozwiązania, opracowanie reprezentacji cech mimiki twarzy i wytrenowanie klasyfikatora. Przeprowadził również analizę wydajności metody zależnie od osiąganego poziomu prywatności. Wkład w artykuł [KPS18] również polegał na opracowaniu metody do identyfikacji biometrycznej, w tym przypadku, bazującej na konturze dłoni. Autor był odpowiedzialny za dobór i ewaluację kolejnych technik przetwarzania obrazu oraz opracowanie reprezentacji cech. We wszystkich czterech pracach autor odpowiadał za pełną implementację zaprezentowanych rozwiązań, trening modeli uczenia maszynowego oraz ewaluację. Autorskie zbiory danych zebrane na potrzeby prac [PSK19, KPS18] zostały przygotowane w większości wysiłkiem autora rozprawy, przy pomocy promotora. W przypadku artykułu [BKM+19] autorzy uznali, że wniesiony wkład jest równomierny (co zostało oficjalnie nadmienione w samej publikacji) i wynosi po 20% dla każdego z nich. Wkład autora rozprawy polegał głównie na analizie cech technikami redukcji wymiarowości poprzez uczenie dedykowanych sieci neuronowych wykorzystując do tego dodatkowe dane o parametrach ataków.

Rozdział 2

Znakowanie wodne z wykorzystaniem sieci neuronowych

2.1 Wprowadzenie

W tym rozdziale prezentujemy nową metodę znakowania wodnego opartą na konwolucyjnych sieciach neuronowych. Znakowanie wodne, funkcjonujące również w literaturze polskojęzycznej w formie zapożyczenia z języka angielskiego – *watermarking*, rozważa się jako problem komunikacji, w którym nadawca umieszcza dodatkowe informacje w wybranym nośniku danych. Komunikację między nadawcą a odbiorcą może zakłócić adwersarz, który przeprowadza atak polegający na modyfikacji (zniekształceniu) nośnika. Celem ataku jest uniemożliwienie odczytania dodatkowej informacji przez odbiorcę końcowego, przy czym zaaplikowane zniekształcenie nie może znacząco wpływać na odczyt oryginalnych informacji umieszczonych w nośniku. W pracy skupiamy się na technikach znakowania wodnego, w których nośnik jest obrazem, przy czym zaprezentowane rozwiązania mogą zostać zaadaptowane do dziedziny wideo (rozumianej jako sekwencja obrazów), co również jest tematem, który podejmujemy w pracy. Z kolei dodatkowa informacja (nazywana też *wiadomością*) jest reprezentowana jako ciąg bitów.

Obecnie znakowanie wodne zyskuje na znaczeniu w kontekście ochrony praw autorskich twórców treści multimedialnych, takich jak wideo, obrazy lub materiały dźwiękowe, ponieważ jest narzędziem służącym do zapobiegania nielegalnej dystrybucji treści multimedialnych [RH13, Loe03]. Teoretyczny model komunikacyjny zdefiniowany dla problemu znakowania wodnego odpowiada realistycznemu i praktycznemu scenariuszowi ochrony treści multimedialnych, w którym do nośnika, jakim jest obraz, wideo lub dźwięk, dodajemy wiadomość w postaci identyfikatora, pozwalającego na

jednoznaczne określenie źródła stojącego za jego (bezprawnym) udostępnieniem lub wykorzystaniem.

W tym rozdziale przedstawiamy nowatorskie rozwiązanie typu *end-to-end* do osadzania i odzyskiwania znaku wodnego w obrazie cyfrowym oparte na konwolucyjnych sieciach neuronowych wytrenowanych w potoku (ang. *pipeline*) bazującym na architekturze enkoder-noiser-detektor oraz wykorzystującym tzw. trening adwersarialny (ang. *adversarial training*), po raz pierwszy zaprezentowany w [GPAM+14], a obecnie szeroko stosowany m.in. w obszarach związanych z generowaniem obrazu, tekstu, czy dźwięku [JTJ20].

W ostatnich latach rynek multimediów stale się rozwija. Dostęp do szerokiej gamy treści multimedialnych zapewniany jest w coraz wygodniejszy sposób, np. platforma Netflix oferuje dostęp offline do swoich zasobów w postaci filmów, seriali i programów telewizyjnych [Net]. To powoduje wzrost nielegalnej redystrybucji treści chronionych prawem autorskim. Jedną z najskuteczniejszych metod zapobiegania takim zachowaniom jest osadzanie w obrazie niewidocznego dla człowieka znaku wodnego, pozwalającego na wygenerowanie treści unikalnych i rozróżnialnych względem każdego użytkownika, które następnie można odzyskać i tym samym określić jednoznacznie źródło nielegalnego wycieku.

Znakowanie wodne wykorzystuje fakt, że pojemność informacyjna obrazu jest znacznie większa niż ilość informacji, które mogą być prawidłowo odebrane i zinterpretowane przez człowieka. Powszechnie wiadomo, że ludzki wzrok jest bardziej wrażliwy na składową *luminancji* przestrzeni kolorów, niż na *chrominancję* [AMCB02], tzn. człowiek potrafi rozróżnić nawet małe zmiany w jasności obrazu, za to nie jest w stanie rozpoznać niewielkich zaburzeń w warstwach reprezentujących kolor. Na tej podstawie można wysnuć wniosek, że kanał chrominancji jest najlepszą przestrzenią do modyfikacji. Niestety, w praktyce watermarking nie działa samodzielnie, a jest składową większego procesu, w którym między innymi występuje kompresja obrazu, która również wykorzystuje ten fakt, a zatem usuwa znacznie więcej informacji o chrominancji. Z tego powodu znak wodny najczęściej kodowany jest w kanale luminancji, a co za tym idzie, niezbędne jest balansowanie pomiędzy trwałością znaku wodnego, jego widocznością oraz ilością przesyłanych informacji. Znakowanie wodne jest jedną z wielu technik opierających się na wykorzystaniu „nadwyżek przepustowości" w danym kanale komunikacyjnym, inne godne uwagi to wspomniana wcześniej kompresja, a także steganografia.

W modelu komunikacyjnym dla znakowania wodnego użytkownik osa-

dza wiadomość w obrazie cyfrowym i wysyła ją do odbiorcy. Obraz ten może zostać przechwycony przez adwersarza i zniekształcony (wykorzystując do tego różnie techniki przetwarzania obrazu). Odbiorca, który posiada wiedzę nt. wcześniej zdefiniowanego zestawu strategii osadzania i wyodrębniania wiadomości z obrazu, powinien być w stanie odzyskać osadzoną wiadomość z (również zniekształconego) obrazu. Dopuszcza się, że adwersarz może zmodyfikować obraz maksymalnie w stopniu, w którym nadal możliwe jest odczytanie jego podstawowej treści, co implikuje, że wykorzystywane techniki przetwarzania obrazu nie mogą znacząco wpływać na jego pogorszenie.

Podczas pracy nad technikami znakowania wodnego musimy zapewnić trzy własności [PHC05]:

1. *przezroczystość* lub inaczej *transparentność* (ang. *transparency*) dotyczy jakości obrazu po dodaniu do niego znaku wodnego. Wszelkie zniekształcenia obrazu, wynikające z osadzenia znaku wodnego, nie powinny istotnie pogarszać poziomu wizualnego odbioru danej treści przez widza. W celu zobiektywizowanego porównania obrazów w naszej pracy wykorzystaliśmy szczytowy stosunek sygnału do szumu (ang. *peak signal-to-noise ratio*, PSNR), który w ogólności mierzy różnicę wartości pikseli między dwoma obrazami oraz wskaźnik podobieństwa strukturalnego (ang. *structural similarity index measure*, SSIM), który w ogólności porównuje obrazy w oparciu o ich cech statystyczne [HZ10];

2. *odporność* (ang. *robustness*) opisuje zdolność użytkownika do dekodowania wiadomości z zakodowanych obrazów po zastosowaniu na niej pewnych operacji przetwarzania sygnału. Operacje te mogą być stosowane intencjonalnie w celu zniszczenia znaku wodnego lub wynikać z wymagań lub ograniczeń technicznych (np. konieczności kompresji sygnału wideo). W pracy używamy określenia *ataki* odnoszącego się do tych operacji. Przykłady ataków obejmują kadrowanie (przycinanie), zmianę rozmiaru, rozmycie gaussowskie, czy kompresję JPEG;

3. *pojemność* (ang. *capacity*) została zdefiniowana w [CKLS97] jako liczba dodatkowo zakodowanych bitów w danym nośniku informacji. W ramach naszego rozwiązania wprowadziliśmy pojęcie *lokalnej* (*blokowej*) *pojemności bitów na piksel*, w celu wyjaśnienia ograniczeń wynikających z zastosowania warstw konwolucyjnych w sieci neuronowej. Wielkość bloku można wyznaczyć poprzez policzenie najdłuższej od-

ległości (w pikselach), na której informacja o danym pikselu może zostać rozpowszechniona, wykorzystując do tego architekturę enkodera opartą na kilku kolejnych warstwach konwolucyjnych, np. dla jednej warstwy o rozmiarze jądra równym 7 rozmiar bloku wynosi 3, a dla dwóch warstw o rozmiarze jądra równym 5 jest to 4.

W pracy przedstawiamy nową technikę osadzania dodatkowej wiadomości w obrazie cyfrowym i wyodrębniania jej za pomocą konwolucyjnych sieci neuronowych wytrenowanych bazując na potoku treningowym opartym na architekturze enkoder-noiser-detektor. W celu wyuczenia modeli używamy metody treningu adwersarialnego oraz stosujemy warstwy imitujące ataki między enkoderem a detektorem, w tym implementujemy „różniczkowalne" (w rozumieniu funkcji dla których wartość pochodnej jest różna od 0 na odpowiednio dużej części dziedziny) precyzyjne przybliżenie kompresji JPEG.

Proponujemy również nowe podejście do rozproszenia wiadomości w domenie przestrzennej obrazu, zmniejszając w ten sposób lokalną pojemność bitów na piksel, a jednocześnie zachowując ogólną pojemność bitową obrazu na ustalonym poziomie. Nasze podejście pozwala znacznie zwiększyć odporność na ataki przestrzenne, takie jak obracanie lub kadrowanie (przycinanie). Dodatkowo zastosowanie omawianej metody rozproszenia przestrzennego znacznie skraca czas treningu sieci neuronowych w porównaniu do poprzednich rozwiązań.

Nasza metoda została przetestowana pod kątem odporności na szeroką gamę ataków, w tym technik kompresji stratnej, takich jak próbkowanie 4:2:0 [Ker] i kompresja JPEG oraz ataków przestrzennych, takich jak obracanie i przycinanie. Te ataki były rozważane w wielu pracach nt. „klasycznego" watermarkingu, jednak niektóre z nich zostały zaniedbane przez autorów ostatnich rozwiązań opartych na sieciach neuronowych, pomimo tego, że są stosunkowo łatwe do zaaplikowania, a niektóre z nich są powszechnymi komponentami stratnych technik kompresji. W pracy uwzględniamy szersze spektrum typowych ataków, dzięki czemu osiągamy wysoką ogólną odporność, a także znacząco (względem innych algorytmów) uodparniamy się na kompresję JPEG, rozmycie gaussowskie, próbkowanie 4:2:0 i zmianę rozmiaru obrazu.

Pokazujemy też, że zastosowanie odpowiednich ataków może zwiększyć odporność na inne, nieuwzględnione podczas treningu, zniekształcenia w ramach jednej grupy ataków, zatem właściwy dobór ataków, zgodnie z ich zakresem wprowadzanych modyfikacji, pozwala osiągnąć wysoką ogólną odporność. Rozważane ataki dzielimy na pięć grup w zależności od sposobu

w jaki modyfikują obraz. Następnie pokazujemy, że wystarczające jest zastosowanie ataków ze wszystkich rozpatrywanych grup w celu zbudowania wydajnego systemu opartego na głębokich sieciach neuronowych do znakowania wodnego. Na koniec oceniamy odporność naszej metody na ataki pod względem jakości obrazu mierzonej za pomocą szczytowego stosunku sygnału do szumu (PSNR).

2.2 Znakowanie wodne a steganografia i szyfrowanie

Termin *znakowanie wodne* odnosi się do pewnego modelu komunikacji opartego na dodaniu dodatkowej wiadomości do nośnika informacji (np. obrazu). Watermarking jest często utożsamiany ze steganografią, która również podejmuje zagadnienie ukrywania dodatkowej wiadomości w danym nośniku. Pomiędzy obiema koncepcjami istnieją jednak znaczące różnice związanie z odmiennymi definicjami modeli komunikacji, w szczególności ról adwersarza w tych modelach. Głównym celem w steganografii jest ukrycie wiadomości, a następnie przekazanie jej odbiorcy w sposób uniemożliwiający adwersarzowi stwierdzenie, czy w danym nośniku zostały umieszczone dodatkowe informacje. Wysoka odporność w steganografii jest rozumiana jako zapewnienie, że dowolny praktyczny mechanizm wykrywający obecność dodatkowej wiadomości w nośniku osiąga skuteczność bliską losowej. W szczególności, mechanizm typujący może być oparty na ocenie wizualnej wykonanej przez człowieka, co indukuje, że wiadomość w modelu steganograficznym musi cechować się wysoką transparentnością.

Kolejnym obszarem powiązanym ze znakowaniem wodnym, jest steganoanaliza, która podejmuje temat szeroko rozumianej analizy osadzenia dodatkowej wiadomości w kanale komunikacyjnym, w tym próby wykrycia ukrytych dodatkowych danych, przeprowadzenia ataków w celu usunięcia wiadomości lub jej zniekształcenia [Böh10, CMB+08]. Wszystkie wymienione obszary zaliczane są do szeroko rozumianego bezpieczeństwa informacji.

Na poziomie definicji pojęcia takie jak watermarking, steganografia oraz steganoanaliza można skonfrontować z innymi pojęciami z obszaru bezpieczeństwa informacji – szyfrowaniem oraz kryptoanalizą. W przypadku szyfrowania celem nadrzędnym jest ukrycie treści przekazywanej wiadomości przed adwersarzem. Z kolei sam fakt wykrycia przez adwersarza, że wiadomość przekazywana kanałem komunikacyjnym jest zaszyfrowana jest nieistotny. W przypadku steganografii fundamentalnym problemem jest, aby przekazanie wiadomości kanałem komunikacyjnym odbyło się w sposób

„niezauważalny" dla adwersarza. Za to model komunikacyjny dla watermarkingu dopuszcza możliwość wykrycia faktu, że nośnik zawiera dodatkowe informacje, a nawet nie wymaga, żeby wiadomość pozostała nieznana dla adwersarza. Istotną znakowania wodnego jest możliwość odczytania wiadomości przez odbiorcę końcowego, mimo przeprowadzonych przez adwersarza modyfikacji nośnika w ramach dopuszczalnej szerokiej klasy ataków. Kryptoanaliza, podobnie jak steganoanaliza, skupia się na analizie systemów kryptograficznych w celu znalezienia luk w tych systemach [CMB+08].

2.3 Przegląd dotychczasowych rozwiązań

Problem niezauważalnego i odpornego na ataki osadzania dodatkowych informacji w domenie cyfrowej jest głęboko badany od wielu lat. Rozwiązania do znakowania wodnego można podzielić na dwa typy — *non-blind* oraz *blind*. Rozwiązania non-blind wymagają oryginalnej kopii obrazu podczas etapu odczytywania dodatkowej wiadomości, za to metody blind są w stanie wykryć wiadomość zakodowaną w obrazie bez żadnych dodatkowych danych i referencji. Ostatnie prace w większości skupiają się na podejściach typu non-blind ze względu na ich łatwiejsze zastosowanie w rzeczywistych warunkach.

Spora cześć rozwiązań do znakowania wodnego wykorzystuje transformacje z dziedziny przestrzennej do dziedziny częstotliwościowej, takie jak dyskretna transformata Fouriera (DFT) [Pun06], dyskretna transformata falkowa (DWT) [Naj17, SDNC15, KSK18, MK14, LT10], dyskretna transformata kosinusowa (DCT) [DSBG11, SS17, PPB10] oraz inne [NNS17, AKRB18]. Ekstremalne uczenie maszynowe (EML) to kolejna technika wykorzystywana do osadzania znaków wodnych w obrazach cyfrowych, która zyskuje na popularności w ostatnich latach [MGS+12, SDNC15, RMB17]. Inną metodą szeroko stosowaną do technik watermarkingowych jest rozkład według wartości osobliwych (SVD), która została wykorzystana między innymi w [MK14, GR12, MK13, LT10, LCT11, NL19, LHL+19]. Wiele prezentowanych prac bazowało na połączeniu dwóch lub więcej technik (np. [MK14, SDNC15]). W pracy [HT20] autorzy zastosowali nadmiarowość w postaci podwójnych znaków wodnych, aby poprawić odporność na przycinanie. Z kolei praca [YWZK22] przedstawia udoskonaloną metodę tzw. naturalnego watermarkingu (ang. *natural watermarking*).

Ukrywanie informacji za pomocą sieci neuronowych w kontekście steganografii, w której autorzy nie brali pod uwagę odporności na ataki i kompresję, zostało dokładnie zbadane w dziedzinie obrazu [HD17, Bal17, Bal20,

BCF19, ZBK+20, TLT+19, YZZ21], a także domenie wideo [WLCM19, AM19].

W ostatnich latach możemy zaobserwować również wzrost zainteresowania zastosowaniem metod głębokiego uczenia do znakowania wodnego. Autorzy [ZKJFF18] zaproponowali potok treningowy oparty na enkoderze i detektorze, który w sposób kompleksowy uczy się zarówno osadzania, jak i odczytywania znaku wodnego. W pracy dodano warstwę szumu między enkoderem i detektorem, a dodatkowy dyskryminator oceniał, czy w danym obrazie jest osadzony znak wodny, czy nie. Wiadomość została rozłożona na wszystkie piksele obrazu, co pozwoliło uzyskać imponującą odporność na ataki przycinania, jednak rozwiązanie przez to było mało odporne na kompresję JPEG oraz wymagało bardzo długiego treningu. W pracy [WA19] autorzy zastosowali metodę uczenia enkodera oraz detektorem z wykorzystaniem tzw. treningu adwersarialnego. Dzięki temu osiągnięto wysoką odporność na ataki, jednak uzyskano niską jakość zakodowanych obrazów mierzoną za pomocą PSNR. Innym interesującym podejściem do poprawy niezawodności wykrywania wiadomości jest użycie dodatkowej sieci neuronowej uczącej się na bieżąco nowych zniekształceń i ich generowania podczas treningu enkodera i detektora [LZC+20], jednak w danej metodzie nie ma kontroli pozwalającej określić, na które ataki sieci osadzająca i odzyskująca wiadomości powinny zostać uodpornione. Z kolei autorzy [ZS19] zaprojektowali w pełni automatyczny system oparty na głębokim uczeniu do ekstrakcji znaku wodnego z obrazów, które zostały wydrukowane i którym później zrobiono zdjęcie aparatem cyfrowym. W pracy [FNC+19] autorzy wykorzystali konwolucyjne sieci neuronowe do zerowego znakowania wodnego (ang. *zero-watermarking*), tj. techniki, w której obraz nie jest modyfikowany, za to wyodrębniane są unikalne cechy w celu określenia praw właścicielskich. Interesujące zagadnienie dotyczące odporności na jeden złożony atak polegający na odtworzeniu znaku wodnego ze zdjęcia obrazu wykonanego aparatem cyfrowym zostało zbadane w [WD19, TMN20, ZHMS20].

Znakowanie wodne wideo z wykorzystaniem sieci neuronowych było eksplorowane w [KMG17, VCF18], natomiast pierwsze wyniki konkurencyjne wobec metod klasycznych przedstawiono w [ZXCIV19], w której wykorzystano mechanizm atencji. Inne podejścia do znakowania wodnego wideo opierają się na modyfikacjach algorytmów kompresji (np. MPEG-4/HEVC), takich jak zniekształcanie współczynników DCT [XWW11, DG16, KHHW20] lub wektorów przesunięć [JHHN12, SLZ+13, LZL+19].

Artykuł [HK20] opisuje rozwiązanie wykorzystujące sieci neuronowe, przy czym skupia się na dwóch atakach – kompresji JPEG i obracaniu

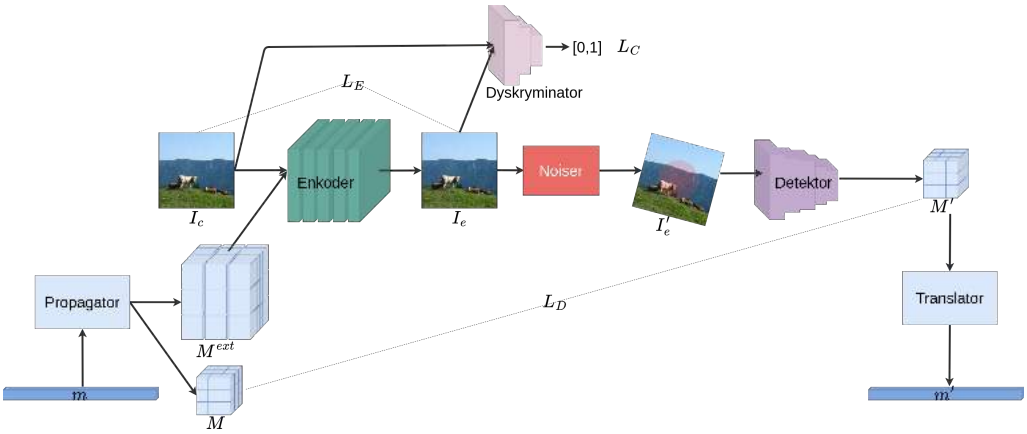
obrazu o określony kąt. W RedMark [ANK+20] zastosowano specjalną warstwę transformacji na obrazie przed przekazaniem sygnału do sieci neuronowej kodującej znak wodny i opracowano algorytm aproksymujący kompresję JPEG. W [KM19] autorzy zaproponowali ulepszoną metodę aproksymacji algorytmu do kompresji JPEG.

Nasze rozwiązanie opiera się na standardowym podejściu do treningu enkoder-noiser-detektor oraz wykorzystuje trening adwersarialny w celu poprawy jakości wygenerowanego obrazu. W pracy zaproponowaliśmy nowy mechanizm przestrzennego rozpraszania znaku wodnego, który znacząco zwiększa odporność na część ataków, w tym kompresję JPEG, oraz rozszerzyliśmy zbiór analizowanych ataków, które pogrupowaliśmy ze względu na sposób przetwarzania obrazu. Pokazujemy, że odpowiedni dobór ataków z każdej grupy pozwala uzyskać ogólny wysoki poziom odporności.

2.4 Architektura rozwiązania i mechanizm rozprzestrzeniania przestrzennego

Głównym celem metody znakowania wodnego jest dodanie dodatkowej informacji, zwanej *znakiem wodnym*, do obrazu cyfrowego, zwanego *obrazem podstawowym*, w sposób, który pozwala na późniejsze odzyskanie znaku wodnego przez uprawnionego użytkownika. Znak wodny musi być odporny na niektóre operacje przetwarzania sygnału, zwane *atakami*. Wzięliśmy pod uwagę następujące ataki: kadrowanie (przycinanie), cropout (podmiana całego zakodowanego obrazu na podstawowy za wyjątkiem wylosowanego prostokąta), dropout (podmiana losowo wybranych pikseli obrazu ze znakiem wodnym na piksele z obrazu podstawowego), obracanie, wygładzanie gaussowskie, próbkowanie 4:2:0, kompresja JPEG oraz zmiana rozmiaru. Obraz z osadzonym znakiem wodnym, również po przeprowadzeniu dodatkowych modyfikacji w postaci ataków, musi zapewnić pewien poziom transparentności, tj. musi dawać możliwość odczytania podstawowej treści.

Naszym celem jest dodanie wiadomości binarnej $m \in \{0,1\}^L$, gdzie $L \in \mathbb{N}_+$, do obrazu podstawowego I_c o wymiarach $(H \times W \times \mathrm{Ch}) \in \mathbb{N}_+^3$. Wynikiem tej operacji jest zakodowany obraz I_e zawierający dodatkową wiadomość m. Obrazy I_c oraz I_e powinny być wizualnie nie do odróżnienia. Następnie adwersarz zniekształca obraz I_e poprzez zastosowanie wybranych ataków w celu uniemożliwienia odczytania wiadomości m z zakodowanego obrazu. Obraz wynikowy po nałożeniu ataków nazywamy *obrazem zaszumionym* I'_e. Obraz I'_e składa się z trzech kanałów oraz ma nieokreśloną wysokość i szerokość. Ostatecznie dokonujemy próby odczytania dodanej



Rysunek 2.1: Schemat potoku treningowego. Propagator przekształca wiadomość m w dwie reprezentacje – M^{ext}, która jest przetwarzana w potoku treningowym oraz M, która jest używana jako obiekt celu w funkcji kosztu L_D. Enkoder dodaje M^{ext} do obrazu I_c, czego wynikiem jest obraz zakodowany I_e. Noiser nakłada na obraz I_e wylosowaną operację imitującą atak w celu wskazania w trakcie procesu uczenia sieciom neuronowym jednego (na iterację) z możliwych sposobów zniekształceń. Detektor odbiera zniekształcony obraz I'_e i odczytuje wiadomość $M^{'}$ o wymiarach równych M. Ostatecznie translator odczytuje wiadomość m' w oparciu o $M^{'}$. Dyskryminator jest komponentem, który w ramach treningu adwersarialnego uczy się rozróżniania między obrazami I_c oraz I_e, czego efektem jest zwiększenie jakości wygenerowanego obrazu I_e.

wiadomości $m' \in \{0,1\}^L$ z obrazu I'_e, która powinna spełniać warunek $\| m - m' \| < \delta$, dla przyjętej małej składowej δ.

2.4.1 Ogólna architektura

Zaproponowana w pracy architektura składa się z sześciu głównych komponentów. Trzy z nich to trenowalne sieci neuronowe: *enkoder* E_ϕ, *detektor* D_γ oraz *adwersarialny dyskryminator* C_ω, gdzie ϕ, γ, ω oznaczają trenowalne parametry kolejnych sieci. Dodatkowym komponentem jest *noiser* N służący do wykonywania ataków na zakodowanym obrazie wygenerowanym przez enkoder E_ϕ. Określamy również dwa deterministyczne algorytmy: *propagator wiadomości* P oraz *translator wiadomości* T. Ogólny schemat architektury został przedstawiony na rysunku 2.1.

2.4.2 Propagator P i translator T_o

Propagator P

Definiujemy m_i jako i-ty bit wiadomości m, następnie przedstawiamy wiadomość m za pomocą sekwencji krotek, gdzie krotka jest reprezentowana jako $s_i = (i, m_{ki}, m_{ki+1} \ldots, m_{ki+k-1})$ dla $i \in \{0, 1, \ldots, \lceil \frac{L}{k} \rceil - 1\}$ oraz $1 \leq k \leq L$. Przyjmijmy, że w przypadku, gdy k nie jest dzielnikiem L ostatnia krotka jest wypełniona dodatkowymi pustymi bitami (tzw. *padding*). W szczególności, dla $k = 1$, jesteśmy w stanie przedstawić wiadomość jako trywialną sekwencję krotek (i, m_i) dla $i \in \{0, 1, \ldots, L - 1\}$. Dodatkowo definiujemy funkcję $bin_n : \mathbb{N}_+ \to \{0, 1\}^n$, która dla danej wartości zwraca jej binarną reprezentację o długości równej n. Parametr $b \in \mathbb{N}_+$ definiuje rozmiar bloku zawierający zreplikowane krotki.

Propagator $P_{nkb} : \{0, 1\}^L \to \{0, 1\}^{\frac{H}{b} \times \frac{W}{b} \times (n+k)}$ jest funkcją, która wykonuje następujące kroki:

1. przekształca wiadomość m w sekwencję krotek $(s_0, s_1, \ldots, s_{\lceil \frac{L}{k} \rceil - 1})$,

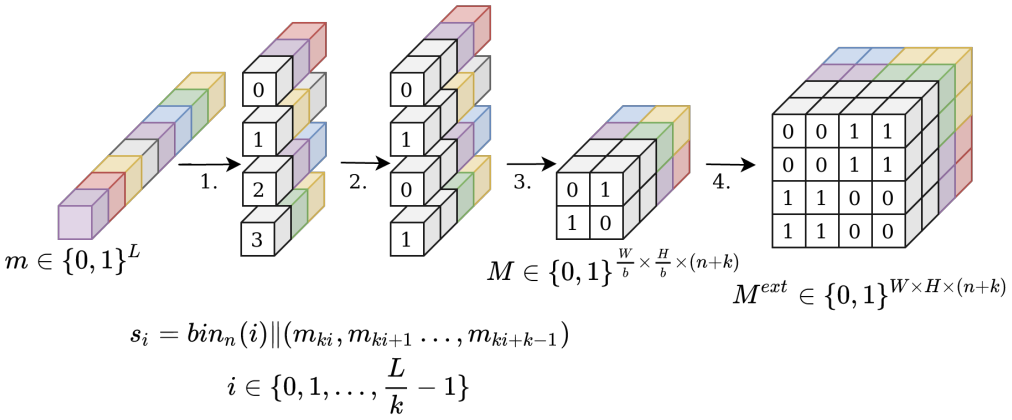
2. dla każdego i, konwertuje pierwszy element krotki s_i do jej binarnej reprezentacji $bin_n(s_{i0})$, spłaszcza krotkę s_i, a następnie zmienia jej kształt poprzez operację rozszerzenia wymiarowości, tak że $s_i \in \{0, 1\}^{1 \times 1 \times (n+k)}$,

3. generuje wiadomość przestrzenną $M \in \{0, 1\}^{\frac{H}{b} \times \frac{W}{b} \times (n+k)}$ poprzez losowo przypisanie krotek s_i do wycinków (ang. *slice*) tensora M_{xy}, gdzie $x \in \{0, 1, \ldots, \frac{H}{b} - 1\}$ oraz $y \in \{0, 1, \ldots, \frac{W}{b} - 1\}$. Zauważmy, że algorytm powiela informacje odnośnie poszczególnych krotek s_i w przestrzennej reprezentacji M, co jest oczekiwanym zachowaniem.

W celu dopasowania wiadomości przestrzennej M do formatu wejścia enkodera konieczne jest dokonanie jeszcze jednego dodatkowego kroku:

4. każdy wycinek M_{xy} jest powielany b-krotnie w kierunku poziomym oraz pionowym (mianowicie, krotka $M_{xy} \in \{0, 1\}^{1 \times 1 \times (n+k)}$ jest konwertowana do $M_{xy} \in \{0, 1\}^{b \times b \times (n+k)}$, gdzie wartość M_{ijl} jest identyczna dla każdego $i \in \{0, \ldots, b\}$ oraz $j \in \{0, \ldots, b\}$.

Jeśli dodatkowy krok jest wykonywany, propagator oznaczamy jako P^{ext}_{nkb}, a zwracaną wiadomość jako $M^{ext} \in \{0, 1\}^{H \times W \times (n+k)}$. Wizualizację działania propagatora przedstawiono na rysunku 2.2.



Rysunek 2.2: Wizualizacja kroków propagatora P^{ext}_{nkb} dla danych parametrów: długość bitowa indeksu $n = 2$, długość pojedynczego wycinka wiadomości $k = 2$, rozmiar bloku replikacji $b = 2$, długość wiadomości $L = 8$, szerokość obrazu $W = 4$ oraz wysokość obrazu $H = 4$. Liczby pod strzałkami odnoszą się do kroków propagatora.

Translator T_o

Ostatnim elementem architektury w potoku jest translator wiadomości T_o. Jest to funkcja deterministyczna, która oblicza końcową reprezentację wiadomości $m' \in \{0,1\}^L$ z wiadomości M' odczytanej przez detektor. Proces obliczeń jest podobny do algorytmu k-najbliższych sąsiadów [CH67]. Dla każdego elementu $i \in \{0, 1, \ldots, \frac{L}{k} - 1\}$, szukamy o krotek M'_{xy} w wiadomości M', gdzie n pierwszych wartości (odczytany przez detektor binarny indeks) są najbliższe rzeczywistej reprezentacji binarnej indeksu $bin_n(i)$, tzn. wybieramy krotkę o koordynatach xy jeśli wynik porównania $||bin_n(i) - M'_{xy[0,\ldots,n-1]}||_2$ jest jednym z o najmniejszych (indeksy są sobie bliskie). Następnie obliczamy średnią wartość z o wybranych krotek dla każdego indeksu i dla każdego elementu kodującego bit wiadomości, tzn. $\frac{1}{o} \sum_{xy} M'_{xyp}$ dla $p \in \{n, \ldots, n + k - 1\}$, co powala na odczytanie wszystkich bitów wiadomości końcowej m'.

Zalety rozwiązania

Zaproponowane podejście pozwala na zakodowanie komunikatu m w obrazie podstawowym I_c i zmniejszenie lokalnej (blokowej) pojemności bitów na piksel, co znacząco zwiększa odporność metody na ataki. Ostatnie i najbardziej zaawansowane rozwiązania do watermarkingu są oparte na konwo-

lucyjnych sieciach neuronowych [ZKJFF18, WA19, LZC+20], co oznacza, że we wszystkich tych rozwiązaniach enkoder osadza wiadomość lokalnie, bez dostępu i wiedzy o całym obrazie. Taka architektura enkodera prowokuje dwa sposoby kodowania wiadomości (dwa sposoby zbiegania sieci neuronowych do celu zdefiniowanego w funkcji kosztu):

1. Osadzany jest tylko podzbiór całej wiadomości zależnie od lokalnej charakterystyki obrazu podstawowego, np. dla danego koloru pikseli kodowane będą tylko wybrane bity wiadomości, stąd w różnych miejscach obrazu (zależnie od koloru) będą osadzane inne bity wiadomości. Ten sposób kodowania obrazu jest ryzykowny i zawodny, ponieważ nie ma pewności jaki będzie obraz podstawowy.

2. Cała wiadomość jest osadzana lokalnie (w bloku pikseli). Analiza architektur zaproponowanych w ostatnich latach oraz wyników odporności na ataki, w szczególności wysokie wyniki wydajności bitowej (ang. *bit accuracy*) dla przekształceń typu kadrowanie, wykazała, że ten sposób kodowania wiadomości jest dużo powszechniejszy. W związku z tym zaproponowaliśmy rozwiązanie zmniejszające lokalną pojemność bitów na piksel, dzięki czemu uzyskujemy lepszą odporność na część ataków, zwłaszcza przekształceń typu wygładzającego (patrz rozdział 2.6).

Zaproponowana przez nas architektura rozprasza wiadomość m po obrazie reprezentując ją w formie krotek $s_i = (i, m_{ki}, m_{ki+1} \ldots, m_{ki+k-1})$, gdzie $i \in \{0, 1, \ldots, \lceil \frac{L}{k} \rceil - 1\}$ oraz $1 \leq k \leq L$. Zauważmy, że rozprzestrzenianie odbywa się w sposób blokowy (w każdym bloku znajduje się inna cześć wiadomości), a nie poprzez przypisywanie całej wiadomości m do każdego pojedynczego piksela. Na przykład, możemy zakodować wiadomość o długości $L = 32$ poprzez podzielenie jej na 8 części o długości równej 4 ($k = 4$ i $n = 3$). Stąd jesteśmy w stanie zakodować każdą cześć wiadomości wykorzystując do tego 7 bitów, gdzie 3 bity są potrzebne do zakodowania bitowego indeksu danej części i 4 bity do zapisania odpowiedniej frakcji wiadomości, wtedy krotkę reprezentujemy jako $s_i = bin_3(i)||(m_{4i}, \ldots, m_{4i+3})$. Podczas eksperymentów osiągnęliśmy najlepsze wyniki dla parametru $k = 2$ (przy założeniu $L = 32$).

Takie podejście znacząco zwiększa odporność na niektóre ataki i kompresję, zachowując przy tym wcześniejszy poziom przejrzystości, pojemności i złożoności obliczeniowej. Zmniejsza się tylko odporność na ataki typu przycinanie, co jest spowodowane możliwością utraty niektórych fragmen-

tów wiadomości w przyciętej części obrazu po zastosowaniu rozproszenia przestrzennego. Za to takie podejście znacznie zwiększa odporność na algorytmy kompresji, wygładzanie gaussowiskie i zmianę rozmiaru, czyli ataki polegające na uśrednianiu sąsiednich pikseli (grupa ataków typu lokalnego). Jest to efekt uproszczenia metody kodowania poprzez zmniejszenie lokalnie liczby bitów do zakodowania (mniej bitów na pojedynczy piksel).

Nasz mechanizm rozprzestrzeniania przestrzennego jest alternatywą dla dodawania wiadomości w sposób *na każdy piksel* [ZKJFF18, LZC+20] lub przedstawiania wiadomości jako obrazu (np. obraz binarny) [ANK+20, Bal20]. Pierwszy mechanizm jest podatny na uśrednianie sąsiednich pikseli, podczas gdy dla drugiego wyzwaniem są ataki polegające na przycinaniu lub obracaniu (gdzie piksele mogą zmieniać pozycje). Nasze podejście do rozproszenia przestrzennego oferuje rozwiązanie, które pozwala znaleźć równowagę między tymi dwoma.

Zauważmy, że takie podejście przyspiesza trenowanie sieci neuronowych i czyni je stabilnym. W przeciwieństwie do podejścia polegającego na przypisaniu wiadomości do każdego piksela [ZKJFF18, LZC+20], w przypadku naszego podejścia już po pierwszych iteracjach treningu można zaobserwować efekt zbiegania (postępu w uczeniu), co pozwala na łatwe zarządzanie procesem uczenia i dostrajaniem hiperparametrów. W innych najnowszych podejściach potok jest uczony przez 200 epok, podczas gdy nasz trening potoku wymaga maksymalnie 100 epok w celu wyuczenia sieci neuronowych.

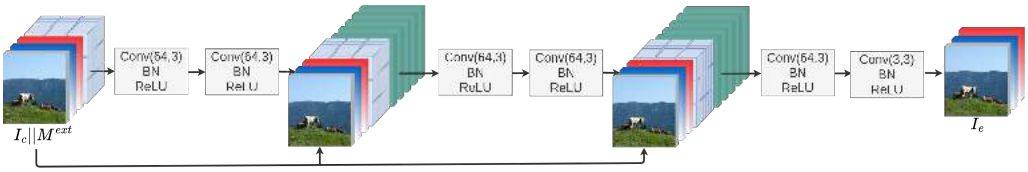
2.4.3 Modele sieci neuronowych do nakładania i dekodowania wiadomości

Blok *ConvBNReLU*

Główny blok, nazywany dalej *ConvBNReLU*, jaki został zastosowany w architekturach sieci neuronowych — enkodera E_ϕ, detektora D_γ i dyskryminatora C_ω, to struktura sekwencyjna dwuwymiarowej warstwy konwolucyjnej (*Conv2D*) z 64 kanałami, rozmiarem jądra równym 3×3, krokiem (ang. *stride*) równym 1×1 i dopełnieniem (ang. *padding*) równym 1×1. Następnie tensor wynikowy jest przetwarzany przez warstwę normalizacji po partii (ang. *batch normalization*, *BatchNorm*) [IS15] oraz funkcję aktywacji ReLU.

Enkoder E_ϕ

Enkoder E_ϕ składa się z 5 bloków *ConvBNReLU* kolejno po sobie umieszczonych. Pierwszy blok pobiera połączony (skonkatenowany, ang. *conca-*



Rysunek 2.3: Architektura enkodera E_ϕ. Pierwszy argument w funkcji *Conv* odnosi się do liczby kanałów, a drugi do rozmiaru jądra. Zachowujemy taki sam kształt obrazu, więc dla rozmiaru jądra równego 3×3 zawsze ustawiamy dopełnienie równe 1×1.

tenated) względem ostatnich wymiarów (tj. kanały obrazu i krotki wiadomości) tensor składający się z obrazu wejściowego I_c oraz wygenerowanej przez propagator P^{ext}_{nkb} wiadomości M^{ext}. Enkoder (jaki i pozostałe sieci) działa w przestrzeni kolorów YCbCr. Połączony tensor $I_c||M^{ext}$ o wymiarach $W \times H \times (3 + n + k)$ jest dodatkowo łączony z tensorem wynikowym zwróconym przez 2 i 4 warstwę *ConvBNReLU*, co wprost oznacza, że warstwy 1, 3 oraz 5 mają bezpośredni dostęp do obrazu wejściowego oraz wiadomości. Ostatnim elementem enkodera E_ϕ jest warstwa konwolucyjna z 3 kanałami (pozostałe parametry odpowiadają domyślnym parametrom zdefiniowanym dla bloku *ConvBNReLU*), która zwraca tensor o wymiarach $W \times H \times 3$ reprezentujący zakodowany obraz I_e w przestrzeni barw YCbCr. Następnie ta reprezentacja może zostać przekonwertowana do przestrzeni barw RGB. Tak zaprojektowana sieć posiada co najwyżej tyle warstw ile modele enkoderów zaprezentowane w innych ostatnich pracach [ZKJFF18, WA19, LZC+20, ANK+20]. Również złożoność obliczeniowa nakładania znaku wodnego jest zbliżona do innych ostatnich najszybszych rozwiązań. Jest to szczególnie ważne w przypadku tego typu aplikacji, ponieważ w wielu praktycznych scenariuszach (np. streaming wideo) enkoder musi działać w czasie rzeczywistym. Schemat architektury enkodera przedstawiono na rysunku 2.3.

Detektor D_γ

Detektor D_γ przyjmuje zakodowane zdjęcie I_e i przetwarza je przy użyciu sekwencji 6 bloków *ConvBNReLU*. Następnie stosowana jest adaptacyjna warstwa poolingu uśredniającego (ang. *adaptive average pooling*), która zwraca tensor o rozmiarach równych $\frac{H}{b} \times \frac{W}{b} \times 64$. W kolejnym kroku tensor jest przekazywany do bloku *ConvBNReLU* zwierającego 64 kanały oraz bloki jądra przekształcenia i dopełnienia (ang. *padding*) równe 1×1. Kolejny element to warstwa konwolucyjna zwracająca $k + n$ kanałów (pozostałe parametry

Tablica 2.1: Architektura detektora D_γ. Zauważmy, że w definicji modelu odwołujemy się do bloku *ConvBNReLU*, który jest opisany poniżej.

Warstwa	Parametry
*ConvBNReLU**	64 kanały, 3×3 rozmiar jądra, 1×1 padding [powtórzony $\times 6$]
AdaptiveAvgPool2D	$\frac{H}{b} \times \frac{W}{b}$ rozmiar wyjścia
*ConvBNReLU**	64 kanały, 1×1 rozmiar jądra, bez paddingu
Conv2D	$k + n$ kanały, 1×1 rozmiar jądra, bez paddingu
BatchNorm	-
Sigmoid	-
**Architektura ConvBNReLU(F_1, F_2, F_3)*:	
Conv2D	F_1 kanały, F_2 rozmiar jądra, F_3 padding
BatchNorm	-
ReLU	-

odpowiadają ustawieniom warstwy poprzedzającej), stąd tensor wyjściowy z detektora D_γ jest tego samego rozmiaru co M, tzn. $\frac{H}{b} \times \frac{W}{b} \times (k + n)$. Zauważmy, że ostatnie dwie warstwy konwolucyjne zachowują się tak samo jak warstwy liniowe dla poszczególnych wektorów po kanałach, tzn. takich składający się z elementów ze wszystkich kanałów na danej pozycji w przestrzeni. Stąd możemy przyjąć interpretację, że poszczególne wektory sa przetwarzane niezależnie przez dwie warstwy liniowe o macierzach przekształceń równych odpowiednio, 64×64 oraz $64 \times (k+n)$. Na końcu stosujemy warstwę normalizacji po partii oraz sigmoidalną funkcję aktywacji (ang. *sigmoid function*). Podczas naszych eksperymentów nie zmienialiśmy rozmiaru tensora wyjściowego poprzez zastosowanie innych parametrów warstwy adaptacyjnego poolingu uśredniającego, tj. detektor zwracał tensor wyjściowy zawsze o tym samym rozmiarze, również gdy zostały przeprowadzane ataki przycinania lub zmiany rozmiaru obrazu. Wykonywanie dodatkowych operacji zależnie od typów ataków mogłoby poprawić odporność naszego rozwiązania, ale wymagałoby rozpoznawania typu ataku oraz nie byłoby wtedy podejściem typu end-to-end, dlatego zdecydowaliśmy się zwracać M' o takim samym rozmiarze w każdym przypadku. Szczegółowy opis architektury modelu jest przedstawiony w tablicy 2.1.

Tablica 2.2: Architektura dyskryminatora C_ω. Zauważmy, że w definicji modelu odwołujemy się do bloku *ConvBNReLU*, który jest opisany poniżej.

Warstwa	Parametry
*ConvBNReLU**	64 kanały, 3×3 rozmiar jądra, 1×1 padding [powtórzony $\times 3$]
GlobalAveragePooling2D	1×1 rozmiar wyjścia
Linear	1 wartość wyjścia
Sigmoid	-
*Architektura *ConvBNReLU(F_1, F_2, F_3)*:	
Conv2D	F_1 kanały, F_2 rozmiar jądra, F_3 padding
BatchNorm	-
ReLU	-

Dyskryminator C_ω

Dyskryminator C_ω składa się z trzech kolejnych bloków *ConvBNReLU*, warstwy globalnego poolingu uśredniającego, która zwraca wektor o 64 wymiarach, następnie wykorzystujemy warstwę liniową wraz z sigmoidalną funkcją aktywacji zwracającą pojedynczą wartość w przedziale $[0, 1]$, którą interpretujemy jako podobieństwo obrazu wygenerowanego przez enkoder do obrazu rzeczywistego. Schemat architektury modelu został przedstawiony w tablicy 2.2.

W trakcie eksperymentach braliśmy również pod uwagę architektury modeli wykorzystanych w [ZKJFF18, LT10]. W tym scenariuszu nie zmieniliśmy architektury enkodera i dyskryminatora, jednak musieliśmy zmodyfikować ostatnie warstwy detektora, aby obsłużyć naszą metodę rozpraszania przestrzennego. Wymagało to jedynie zmiany warstwy globalnego poolingu uśredniającego na adaptacyjny. Następnie użyliśmy tej samej sekwencji warstw, co w poprzednio opisanych rozwiązaniach.

2.4.4 Warstwa zaszumiająca oraz rozważane ataki

W celu zwiększenia odporności naszego rozwiązania na ataki wykorzystaliśmy podejście, w którym w potoku treningowym między enkoderem E_ϕ oraz detektorem D_γ umiejscowiliśmy specjalny komponent – noiser N, który zaburza obraz z nałożonym znakiem wodnym I_e poprzez zastosowanie wybranej operacji przetwarzania obrazu. W ten sposób jesteśmy w stanie podczas treningu przekazać do sieci neuronowych informację o możliwych

sposobach zaburzania obrazu i tym samym wskazać kierunek w jaki sieci powinny być trenowane, a dzięki temu zwiększyć wydajność rozwiązania, tutaj rozumianej jako odporność na ataki. Typy zaburzeń, które wybraliśmy to: przycinanie, cropout, dropout, wygładzanie gaussowskie, rotacja, zmiana rozmiaru obrazu, próbkowanie 4:2:0 oraz kompresja JPEG, a dokładnie jej różniczkowalna aproksymacja (przy czym zaznaczmy, że rozwiązanie było testowane na oryginalnym algorytmie JPEG).

Atak polegający na przycinaniu zwraca wycinek z obrazu I_e o kształcie kwadratu oraz określonym stosunku powierzchni $p = \frac{H^{new}W^{new}}{HW}$. Cropout działa podobnie do standardowego przycinania – wycina kwadrat z obrazu I_e, ale pozostałe piksele, zamiast odrzucać, podmienia na piksele z obrazu podstawowego I_c. Podobnie jak w [ZKJFF18], zdecydowaliśmy się użyć obrazu I_c jako tła dla zakodowanego obrazu I_e, ze względu na większą trudność tego ataku, który odpowiada jednemu ze standardowych modeli w teorii informacji, mianowicie binarnemu kanałowi symetrycznemu (ang. *binary symmetric channel*, BSC), gdzie odbiorca nie ma wiedzy, czy otrzymany bit jest poprawny czy błędny. Podmiana pikseli na losowy kolor lub wybór jednolitego tła (np. czarnego) odpowiadałaby prostszemu modelowi, tj. binarnemu kanałowi z kasowaniem (ang. *binary erasure channel*, BEC), gdzie informacja o poprawności przesłanego bitu jest przekazywana. Zastosowanie wybranego przez nas podejścia pozwala zatem na przetestowanie wydajności rozwiązania dla bardziej złożonego scenariusza ataku. Cropout również jest parametryzowany poprzez wartość p określającą stosunek powierzchni obrazu zawierającego znak wodny do powierzchni początkowej. Z kolei w ataku dropout podajemy parametr $p \in [0, 1]$, który definiuje jaki procent pikseli obrazu I_e nie jest podmieniany, a pozostałe piksele są zastępowane odpowiednimi pikselami z obrazu I_c. Podobnie jak przy ataku cropout, ta procedura również symuluje model BSC. Wygładzanie gaussowskie zostało zdefiniowane parametrem σ odpowiadającym za szerokości jądra.

Kolejne cztery ataki są naszym rozwinięciem listy ataków względem wcześniejszych prac [ZKJFF18, WA19]. Pierwszy z nich to atak rotacyjny, który obraca obraz o α stopni. Próbkowanie 4:2:0 [Ker] jest stosowane w wielu algorytmach kompresji cyfrowej, takich jak JPEG lub MPEG, i jest to najbardziej popularny wariant próbkowania chromatycznego (inne to np. 4:2:2, 4:1:1). Atak polega na zmniejszeniu kanału obrazu *Cb* i *Cr* przez obliczenie średniej wartości z czerech pikseli w obrębie kwadratów 2×2. Procedura może być wykonana przy użyciu warstwy konwolucyjnej dwuwymiarowej z jednym kanałem, rozmiarem jądra równym 2×2 oraz krokiem

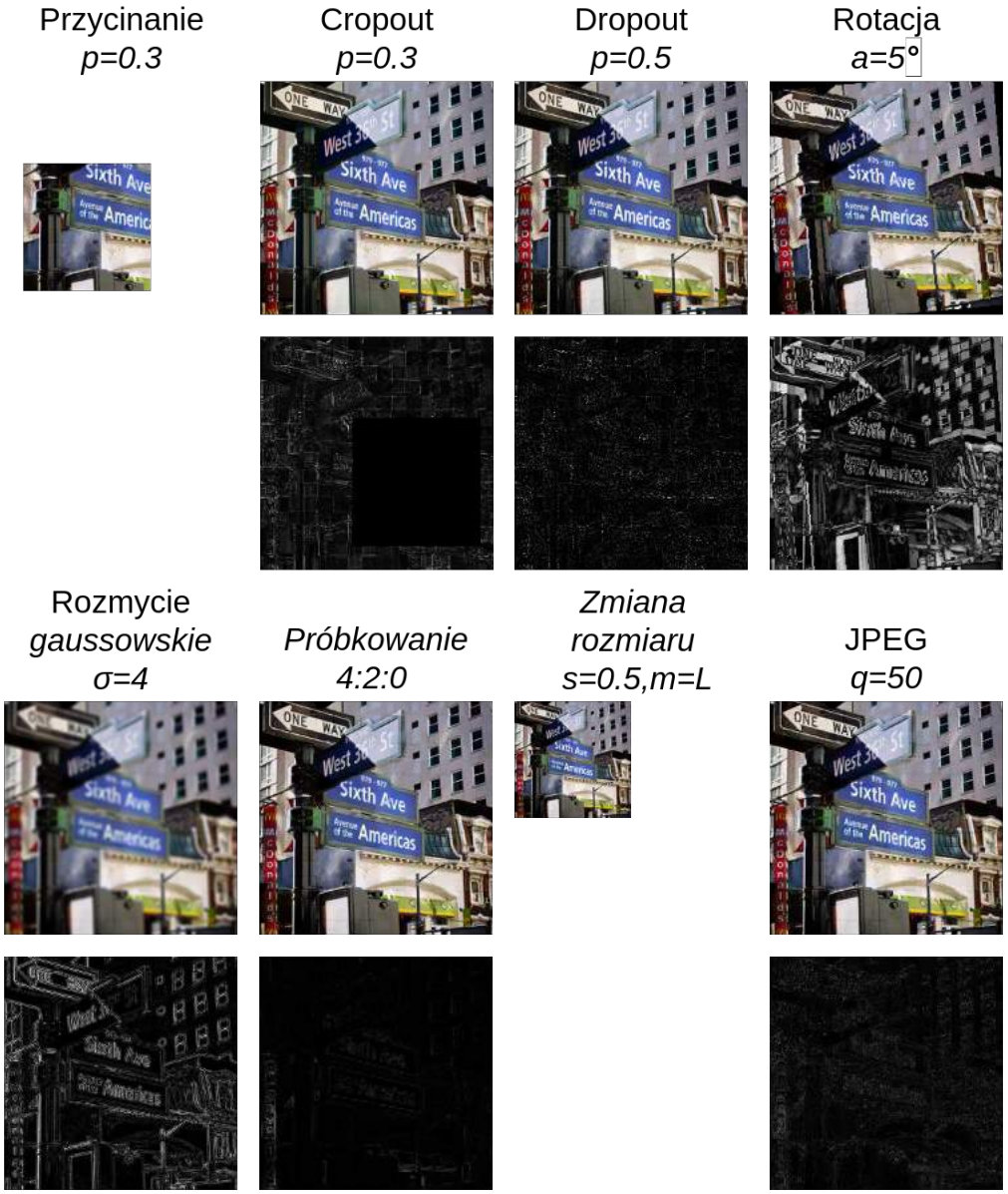
wynoszącym 2×2, dodatkowo wagi jądra muszą przyjmować wartości 0.25. Zastosowaliśmy również atak zmiany rozmiaru ze współczynnikiem skali $s = \frac{H^{new}}{H} = \frac{W^{new}}{W}$. Przy tym ataku uwzględniliśmy dwa rodzaje interpolacji – najbliższych sąsiadów i Lanczosa. Ataki zostały zobrazowane na rysunku 2.4.

Aproksymacja kompresji JPEG

Algorytmy kompresji stratnej można uznać za najskuteczniejszy rodzaj ataków na szeroką gamę protokołów znakowania wodnego. Wynika to z faktu, że algorytmy takie jak JPEG (ITU T.81, ISO/IEC 10918) [PM92] są bardzo wydajne w usuwaniu z obrazów artefaktów, które są niezauważalne dla odbiorcy. Jednocześnie podstawowym założeniem technik znakowania wodnego jest dodanie dodatkowej informacji, tak aby zmiana nie była łatwo widoczna dla widza, a przy tym dająca się później odzyskać. Łatwo zauważyć, że przestrzeń obrazu, która może być wykorzystana do umieszczenia dodatkowej informacji przez algorytmy watermarkingowe jest usuwana przez algorytmy kompresji stratnej. Z tego wynika, że konieczne jest uwzględnienie w potoku treningowym tego typu algorytmów w celu wskazania potencjalnych przestrzeni obrazu, z których znak wodny zostanie usunięty, tym samym, nie będzie możliwy później do odczytania. Takie podejście pozwala w znaczący sposób zwiększyć odporność metody znakowania wodnego. Głównym ograniczeniem w wykorzystaniu algorytmu JPEG w potoku treningowym jest operacja kwantyzacji, która jest wykorzystywana do zmniejszenia rozmiaru zapisywanego obrazu. Pochodna funkcji kwantyzacji jest nieokreślona dla punktów $x \in \mathbb{Z}$ i równa 0 w pozostałej części dziedziny. Zatem użycie algorytmu JPEG wewnątrz potoku treningowego jest niemożliwe, ponieważ blokuje aktualizację wag enkodera w procesie uczenia sieci neuronowych metodą propagacji wstecznej. Jednak możliwe jest zastosowanie aproksymacji, w której nie wykorzystujemy bezpośrednio kwantyzacji. Obecnie istnieje kilka propozycji aproksymacji algorytmu kompresji JPEG [ANK+20, KM19], jednak w pracy przedstawiamy ich ulepszenie, które uwzględnia m.in. próbkowanie 4:2:0.

Zaproponowane przez nas różniczkowalny algorytm aproksymujący metodę kompresji JPEG wykonuje następujące kroki dla obrazu I:

1. konwertuje obraz do przestrzeni barw YCbCr,
2. przeprowadza próbkowanie 4:2:0,
3. dzieli każdy kanał na bloki o wymiarach 8×8,



Rysunek 2.4: Rysunek przedstawia ataki rozważane w pracy. Górny wiersz odnosi się do zaszumionego obrazu I'_e, a wiersz poniżej odnosi się do znormalizowanej różnicy między zaszumionym obrazem I'_e i obrazem ze znakiem wodnym I_e. W celu uwypuklenia różnic zastosowaliśmy normalizację min-max.

4. stosuje dyskretną transformatę kosinusową (DCT),

5. dzieli otrzymane wartości zgodnie z tablicą kwantyzacji Q,

6. stosuje aproksymację zaokrąglania.

Kroki 5 i 6 wykonujemy zgodnie z następującym formułą:

$$I'_{ij} = \begin{cases} 0, & \text{if } -\frac{1}{2} \leq \frac{I_{ij}}{Q_{ij}} \leq \frac{1}{2}, \\ \frac{I_{ij}}{Q_{ij}} + \delta Q_{ij}, & \text{w innym przypadku,} \end{cases} \tag{2.1}$$

gdzie $\delta \sim N(0, \sigma^2)$, I_{ij} jest elementem obrazu w dziedzinie częstotliwości oraz Q_{ij} jest powiązanym elementem w tabeli kwantyzacji. W celu przeprowadzenia treningu sieci neuronowych przyjęliśmy, że $\sigma = 0.01$. W eksperymentach wykorzystujemy standardową tabelę kwantyzacji dla parametru jakości $q = 50$, a dla innych wartości parametru q modyfikujemy elementy tabeli Q zgodnie ze standardem JPEG [PM92, Par17]. Do celów testowych wykorzystujemy klasyczny algorytm JPEG [PM92].

2.4.5 Potok treningowy

Wiadomość M^{ext} wygenerowana przez propagator P^{ext}_{nkb} wraz z obrazem wejściowym I_c jest wykorzystywana przez enkoder E_ϕ do stworzenia zakodowanego obrazu I_e:

$$I_e = E_\phi(I_c, M^{ext}). \tag{2.2}$$

Następnie nakładamy ataki na obraz I_e wykorzystując do tego noiser N:

$$I'_e = N(I_e, I_c, M^{ext}). \tag{2.3}$$

Zauważmy, że część ataków wymaga obrazu podstawowego I_c, np. dropout. W celu przeprowadzenia kadrowania, dodatkowo podczas fazy treningu odpowiednio kadrujemy wiadomość M. Detektor D_γ próbuje wyodrębnić wiadomość M mając dostęp jedynie do zniekształconego obrazu I'_e:

$$M' = D_\gamma(I'_e) \in [0,1]^{\frac{H}{b} \times \frac{W}{b} \times (n+k)}. \tag{2.4}$$

Dodatkowo wykorzystujemy dyskryminator C_ω do ocenienia czy I_e jest podobny do I_c, co ma pomóc w uzyskaniu obrazu o wysokiej transparentności:

$$C_\omega(I \in \{I_e, I_c\}) \in [0,1]. \tag{2.5}$$

2.4.6 Funkcja kosztu

Sformułowaliśmy nową funkcję straty do trenowania sieci neuronowych z wykorzystaniem metody gradientu prostego, która uwzględnia specyfikę naszego zadania. Nasza ogólna funkcja kosztu zawiera trzy funkcje składowe L_E, L_D oraz L_C, które mają za zadanie określenie kierunku treningu dla odpowiednio enkodera E_ϕ, detektora D_γ oraz dyskryminatora C_ω. Noiser N, który jest częścią potoku treningowego nie zawiera parametrów trenowalnych. Ponadto propagator wiadomości P_{nkb} oraz translator T_o są algorytmami deterministycznymi poza potokiem uczenia.

Celem funkcji kosztu L_E jest utrzymanie obrazów I_c oraz I_e możliwie podobnych. Ta funkcja jest sformułowana w sposób następujący:

$$L_E(I_c, I_e) = \text{MSE}(I_c, I_e) = \frac{1}{H \cdot W \cdot \text{Ch}} ||I_c - I_e||_2^2, \tag{2.6}$$

gdzie MSE jest standardową funkcją błędu średniokwadratowego. Funkcja kosztu L_D określa podobieństwo między propagowanymi wiadomościami M oraz M', które również ma być jak najmniejsze. Jednakże, jako że M zawiera powielone dane, tzn. te same krotki, nie jest konieczne uzyskanie doskonałego odwzorowania między wiadomościami. Naszym celem było wyodrębnienie jedynie podzbioru krotek o „dużej pewności" odnośnie zawartych w nich informacjach. W ten sposób sformułowaliśmy nową funkcję straty L_D jako kombinację funkcji średniej i wariancji:

$$L_D^{mean}(M, M') = \frac{b^2}{H \cdot W} \sum_{h=0}^{H_b} \sum_{w=0}^{W_b} Mean(|M_{hw} - M'_{hw}|) \tag{2.7}$$

$$= \frac{b^2}{H \cdot W \cdot (n+k)} ||M - M'||_1 \tag{2.8}$$

oraz

$$L_D^{var}(M, M') = \frac{b^2}{H \cdot W} \sum_{h=0}^{H_b} \sum_{w=0}^{W_b} Var(|M_{hw} - M'_{hw}|), \tag{2.9}$$

gdzie $H_b = \frac{H}{b} - 1$ oraz $W_b = \frac{W}{b} - 1$, operator $|V|$ zwraca wektor wartości bezwzględnych wektora V, funkcja $Mean$: $\mathbb{R}^{(n+k)} \to \mathbb{R}$ zwraca średnią wektora, a funkcja Var: $\mathbb{R}^{(n+k)} \to \mathbb{R}$ zwraca wariancję wektora. Ostateczna funkcja kosztu jest zdefiniowana następująco:

$$L_D(M, M') = \lambda_D^{mean} L_D^{mean}(M, M') + \lambda_D^{var} L_D^{var}(M, M'). \tag{2.10}$$

Takie sformułowanie funkcji straty sprzyja uczeniu się *wszystkich elementów w niektórych krotkach* zamiast *niektórych elementów we wszystkich krotkach*, co ze względu na powielenie danych we wiadomości jest efektem pożądanym.

Zdefiniowaliśmy również trening adwersarialny enkodera E_ϕ z wykorzystaniem dyskryminatora C_ω, dzięki czemu osiągamy lepsze wizualne podobieństwo obrazów I_c oraz I_e. Dla enkodera E_ϕ spodziewamy się, że będziemy generować obrazy uzyskujące odpowiedni stopień transparentności, zatem definiujemy funkcję kosztu jako $L_C^E = log(1 - C_\omega(I_e))$. Z kolei rolą dyskryminatora C_ω jest rozróżnienie pomiędzy „rzeczywistym" obrazem I_c oraz obrazem zmodyfikowanym I_e, stąd w tym przypadku definiujemy funkcję kosztu jako $L_C^C = log(1 - C_\omega(I_c)) + log(C_\omega(I_e))$. Zdefiniowanie funkcji L_C^E oraz L_C^C w sposób sobie przeciwstawny, tzn. zbieganie jednej z nich do zdefiniowanego celu wpływa na wzrost wartości drugiej, pozwala na uczenia enkodera E_ϕ w oparciu technikę treningu adwersarialnego.

Ostatecznie, wykorzystujemy zmodyfikowany algorytm gradientu prostego Adam [KB15] do znalezienia parametrów ϕ oraz γ poprzez minimalizację funkcji kosztu nad dziedzinami obrazów wejściowych I_c i wiadomości M:

$$\mathbb{E}_{I_c,M}[\lambda_E L_E + \lambda_D^{mean} L_D^{mean} + \lambda_D^{var} L_D^{var} + \lambda_C L_C^E], \tag{2.11}$$

gdzie współczynniki λ są wagami dla poszczególnych składowych funkcji kosztu. Równolegle przeprowadzamy uczenie C_ω w celu znalezienia parametrów ω poprzez minimalizację funkcji kosztu nad dziedziną obrazów wejściowych I_c, czyli $\mathbb{E}_{I_c}[L_C^C]$.

2.4.7 Szczegóły treningu

Nasza metoda została wytrenowana i przetestowana wykorzystując do tego celu zbiór danych COCO [LMB+14]. Do podzbioru treningowego wylosowaliśmy 10000 obrazów, a do zbioru testowego 1000 obrazów. Oba podzbiory były rozłączne. Zarówno wybór bitów wiadomości, jaki i ich rozmieszczanie przestrzenne w całym procesie trenowania i testowania było losowane. Parametry λ_E, λ_D^{mean}, λ_D^{var} oraz λ_C zostały ustalone odpowiednio na 4.0, 1.0, 1.0 oraz 0.01. Wykorzystaliśmy optymalizator Adam [KB15] ze współczynnikiem uczenia równym 0.001 (inne wartości nie były zmieniane w stosunku do parametrów domyślnych). Modele zostały wytrenowane z partią (ang. *batch*) o rozmiarze równym 12. Ostateczny trening modeli z uwzględnieniem wszystkich rodzajów ataków trwał 100 epok.

2.5 Podział ataków pod kątem sposobu przetwarzania obrazu

Zaobserwowaliśmy, że większość rozważanych przez nas ataków można przypisać do bardziej ogólnych grup, ze względu na ich specyficzne cechy wpły-

wające na sposób w jaki przetwarzają obraz. W tym rozdziale przedstawiamy opracowany przez nas podział oraz prezentujemy wyniki odporności metody wytrenowanej na podzbiorach ataków dobranych zależnie od grupy do której zostały przypisane. Założyliśmy, że po każdym z ataków treść obrazu musi być widoczna, a jego jakość musi być akceptowalna dla odbiorcy.

2.5.1 Grupy ataków

Poniżej przedstawiamy zaproponowany przez nas podział na cztery grupy ataków zależnie od sposobu przedstawiania obrazu:

- **Określone na pikselu** (ang. *pixel-specific*), gdzie modyfikujemy tylko pojedyncze piksele (bez wiedzy i uwzględnienia innych) poprzez zmianę koloru, dodanie szumu, zastąpienie pikseli innymi (np. z innego obrazu lub stosując zaszumianie typu sól i pieprz), usunięcie niektórych pikseli lub zmianę ich pozycji na obrazie. W tej grupie możemy określić dwie podgrupy: jedną, która stosuje *jedną modyfikację na wszystkich pikselach*, a drugą, która stosuje *jedną modyfikację na wybranym podzbiorze pikseli*. Cechą charakterystyczną tej grupy jest to, że po atakach mamy dostęp do pomniejszanego podzbioru niezmodyfikowanych pikseli lub wszystkich pikseli przekształconych w ten sam specyficzny sposób (istnieje możliwość odtworzenia pierwotnych wartości). Do tej grupy wybraliśmy kilka ataków, takich jak konwersja przestrzeni kolorów, przycinanie, cropout, dropout i rotacja.

- **Lokalne**, gdzie modyfikujemy piksele z uwzględnieniem ich sąsiedztwa. W tej grupie wszystkie piksele są modyfikowane podczas ataków, ale tylko sąsiednie piksele wpływają na ostateczną wartość modyfikowanego piksela. Ataki z tej grupy to: próbkowanie 4:2:0, rozmycie gaussowskie i zmiana rozmiaru.

- **Dziedzinowe**, gdzie modyfikacje są specyficzne dla dziedziny (np. częstotliwościowej wygenerowanej za pomocą transformaty Fouriera), co może powodować, że niewielkie zmiany w ograniczonym sąsiedztwie (w danej dziedzinie) mogą mieć globalny wpływ na obraz reprezentowany w innej dziedzinie. Ta grupa ataków obejmuje wszystkie metody transformacji, m.in. powszechnie stosowaną w algorytmach kompresji obrazu dyskretną transformatę kosinusową (DCT).

- **Mieszane**, gdzie ostateczna modyfikacja jest kombinacją metod z innych grup. Tutaj możemy wyróżnić algorytm kompresji JPEG, który łączy konwersję przestrzeni kolorów, próbkowanie 4:2:0 i lokalnie stosowaną transformatę DCT.

Analiza typów ataków może być ważna i pomocna w kontekście projektowania potoku treningowego i doboru ataków do noisera. Większość najnowszych rozwiązań do znakowania wodnego opartych na uczeniu głębokim wykorzystuje dodatkowe warstwy zaszumiające w celu poprawy odporności na określone ataki (np. [ZKJFF18, WA19, ANK+20]). Takie podejście wymaga wyselekcjonowania skończonego zestawu ataków, które zostaną zastosowane podczas treningu. Co więcej, wszystkie ataki muszą umożliwiać trening, tzn. być różniczkowalne, ponieważ są one zwykle umiejscowione w potoku treningowym po sieci neuronowej odpowiedzialnej za nakładanie znaku wodnego na obraz. W związku z tym wymagane jest opracowanie różniczkowalnego przybliżenia niektórych ataków, np. kompresja JPEG, co nie jest zawsze możliwe lub wymaga znaczących nakładów pracy. Odpowiedni dobór ataków dla potoku treningowego, który uwzględnia ataki ze wszystkich grup, powinien znacząco zwiększyć odporność na inne ataki, które nie zostały bezpośrednio zastosowane do treningu modeli.

W [LZC+20] autorzy zaproponowali metodę, w której noiser, najczęściej składający się z zestawu wcześniej zdefiniowanych ataków, został zastąpiony dodatkową siecią neuronową generującą zniekształcenia. Jednak po analizie wyników wspominanej metody możemy zaobserwować, że nawet małe perturbacje generowane przez ataki zaliczane do grupy metod *lokalnych* zauważalnie zmniejszają dokładność wykrywania wiadomości. Oznacza to, że wytrenowana w ten sposób zaszumiająca sieć neuronowa generowała zniekształcenia należące do grup ataków *określonych na pikselu* lub *dziedzinowych*, ale ignorowała ataki podobne do tych z grupy ataków *lokalnych* (patrz tablica 2.4), stąd zaproponowane rozwiązanie nie można uznać za w pełni skuteczne. W naszym rozwiązaniu pokazujemy, że w celu uzyskania wysokiej odporności na rozważane ataki wymagane jest uwzględnienie przedstawicieli wszystkich czterech grup.

2.5.2 Odporność przy selektywnym dobrze ataków

Przeprowadziliśmy eksperyment, w którym sieci neuronowe były trenowane z wykorzystaniem tylko pewnego podzbioru ataków wybranych tylko z jednej grupy i przeanalizowaliśmy jak taki sposób treningu wpływa na końcowe wyniki odporności na ataki z tej samej grupy oraz innych grup. Wy-

niki zostały przedstawione w tablicy 2.3. Eksperymenty potwierdziły, że istnieje większa korelacja w osiąganych wynikach odporności między metodami przetwarzania obrazów zaliczanych do tej samej grupy ataków. Łatwo zauważyć, że ataki przycinania i cropout nie modyfikują całego obrazu, tzn. detektor ma dostęp do niezmodyfikowanego kwadratu, w którym znajduje się znak wodny. Atak dropout zmienia losowo piksele, ale detektor nadal może wykryć wiadomość na podstawie niezmodyfikowanych pikseli. Podobieństwo w sposobie przetwarzania obrazu pozwoliło na uzyskanie wyższej odporności na ataki cropout oraz rotację, mimo że nie zostały one bezpośrednio wykorzystanie w warstwie noiser podczas treningu. W ten sposób zastosowanie tylko podzbioru ataków z jednej grupy podczas treningu zapewnia bardziej ogólną odporność na szerszy zbiór ataków z danej grupy. Za to trening sieci z wykorzystaniem ataków należących tylko do jednej grupy nie powoduje znaczącego wzrostu odporności na ataki z innych grup. Wyniki eksperymentów potwierdzają, że w celu zapewnienia wysokiej ogólnej odporności istotny jest odpowiedni dobór ataków, tak aby pokrywały one wszystkie grupy ataków.

2.6 Wyniki analizy odporności metody

W tej części przedstawiamy wyniki odporności naszej metody na ataki i porównujemy je z innymi najnowszymi rozwiązaniami. Eksperymenty zostały przeprowadzone zgodnie z metodologią zaprezentowaną w pracy [ZKJFF18], tzn. na obrazach o rozmiarze 256×256 i dla wiadomości o długości $L = 32$ (w innych porównywanych pracach przyjęto pojemność bitową wynoszącą 30). Naszym głównym celem było zmniejszenie lokalnej pojemności bitów na piksel, dlatego ustaliliśmy $k = 2$. Dzięki temu, liczba bitów potrzebnych do przechowywania krotki wiadomości s_i była równa 6, a liczba krotek wymaganej to reprezentacji całej wiadomości wynosiła 16. Jedna krotka składała się z 2 bitów wiadomości oraz odpowiadającemu im indeksowi zapisanemu za pomocą 4 bitów. Parametr translatora o został ustalony na 3. Rozmiar bloku b został ustalony na 16. Aby rozmieścić wszystkie krotki na obrazie, musieliśmy ulokować 16 bloków o rozmiarze wynoszącym 16×16 pikseli. To oznacza, że obraz, na którym jest możliwe zakodowanie wiadomości musi mieć rozmiar wynoszący przynajmniej 64×64 pikseli. W przypadku rozmiarów obrazów wykorzystanych w eksperymentach każda krotka była umieszczona w obrazie średnio 16 razy. Ostateczna metoda została wytrenowana z uwzględnieniem w warstwie noisera wszystkich rozważanych ataków. Do mierzenia odporności algorytmu wykorzystaliśmy dokładność

Tablica 2.3: Wyniki odporności na ataki przy selektywnym dobrze ataków tylko z jednej grupy w trakcie procesu treningowego. Wartości w tablicy przedstawiają dokładność bitową (ang. *bit accuracy*). W każdej kolumnie przedstawiono wyniki dla niezależnego procesu treningowego, w którym do warstwy noisera zostały wykorzystane tylko określone ataki. Kolor czerwony oznacza te ataki, które zostały użyte podczas treningu sieci, a kolor niebieski oznacza najlepszą dokładność bitową osiągniętą dla danego ataku niewykorzystanego w procesie treningowy. Zauważmy, że najlepsze wyniki osiągnięto w ramach tych samych grup ataków, w których ich podzbiór był wykorzystywany w procesie treningu.

Ataki	Warstwa noisera		
	Identyczność	Przycinanie Dropout	Rozmycie Próbkowanie
Identyczność	0.999	0.991	0.985
Przycinanie(p=0.3)	0.847	0.894	0.833
Cropout(p=0.3)	0.793	0.875	0.672
Dropout(p=0.5)	0.530	0.972	0.574
Rotacja(α=5°)	0.754	0.821	0.780
Rozmycie(σ=5)	0.823	0.564	0.981
Próbkowanie(4:2:0)	0.524	0.623	0.980
Rozmiar(s=0.5, m=L)	0.511	0.532	0.735
JPEG(q=95)	0.502	0.512	0.783

bitową (ang. *bit accuracy*). Uzyskane wyniki odporności na ataki zostały przedstawione w tablicy 2.4.

Kompresja stratna a znakowanie wodne

Algorytmy kompresji stratnej i enkodery do znakowania wodnego działają w tej samej poddomenie obrazu, tj. w celu odpowiednio zmniejszenia rozmiaru obrazu lub zakodowania dodatkowych informacji, modyfikują obraz, tak żeby dokonanie zmiany nie były zauważalne (lub były mało zauważalne) dla odbiorców. Z tego powodu algorytmy kompresji uznaliśmy za specjalny rodzaj ataku. Chcąc stworzyć rozwiązanie o wysokiej transparentności, enkoder osadzający znak wodny powinien zmieniać dokładnie te piksele, które są usuwane lub modyfikowane przez algorytmy kompresji stratnej. W naszym rozwiązaniu zwracamy szczególną uwagę na odporność na techniki kompresji stratnej, ponieważ współczesne aplikacje lub usługi multimedialne domyślnie z nich korzystają, co więcej bardzo często nie jest

Tablica 2.4: Wyniki wydajności bitowej dla rozważanych typów ataków oraz porównanie naszej metody z najnowszymi rozwiązaniami – HiDDeN [ZKJFF18], DADW [LZC+20] oraz RedMark [ANK+20]. Wyniki w kolumnie $S+C$ uzyskano stosując metodę rozproszenia przestrzennego i architekturę enkoder E_ϕ z konkatenacją $I_c||M^{ext}$ co drugi blok *ConvBNReLU*. Z kolei w kolumnie S, zostały przedstawione wyniki dla modelu o architekturze zbliżonej do [ZKJFF18, LT10]. Wyniki odporności zostały zaprezentowane dla ustalonej pojemności bitowej równiej 30 lub 32. W pracach [LZC+20] i [ANK+20] nie zostały określone rodzaje interpolacji przy zmianie rozmiaru.

Ataki	*Nasza*		Metody		
	$S+C$	S	HiDDeN	DADW	RedMark
Identyczność	1.000	1.000	1.000	1.000	1.000
Przycinanie(p=0.3)	0.832	0.883	**1.00**	**1.00**	-
Cropout(p=0.3)	0.902	0.901	**0.940**	-	0.925
Dropout(p=0.5)	0.962	0.889	**1.0**	**1.0**	≈0.990
Rotacja(α=5°)	**0.842**	0.828	-	-	-
Rozmycie(σ=2)	**0.986**	0.982	0.960	0.600	0.500
Rozmycie(σ=4)	**0.982**	0.980	0.820	0.500	0.500
Próbkowanie(4:2:0)	**0.984**	0.980	-	-	-
Rozmiar(s=0.5, m=N)	0.849	**0.860**	-	0.671	0.819
Rozmiar(s=0.5, m=L)	0.908	**0.920**	-		
JPEG(q=50)	**0.831**	0.749	0.67	0.817	0.746
Pojemność (liczba bitów)	**32**	**32**	30	30	30

możliwe pominięcie kroku kompresji ze względu na techniczne ograniczenia przepustowości pasma transmisyjnego, np. w przypadku strumieniowania.

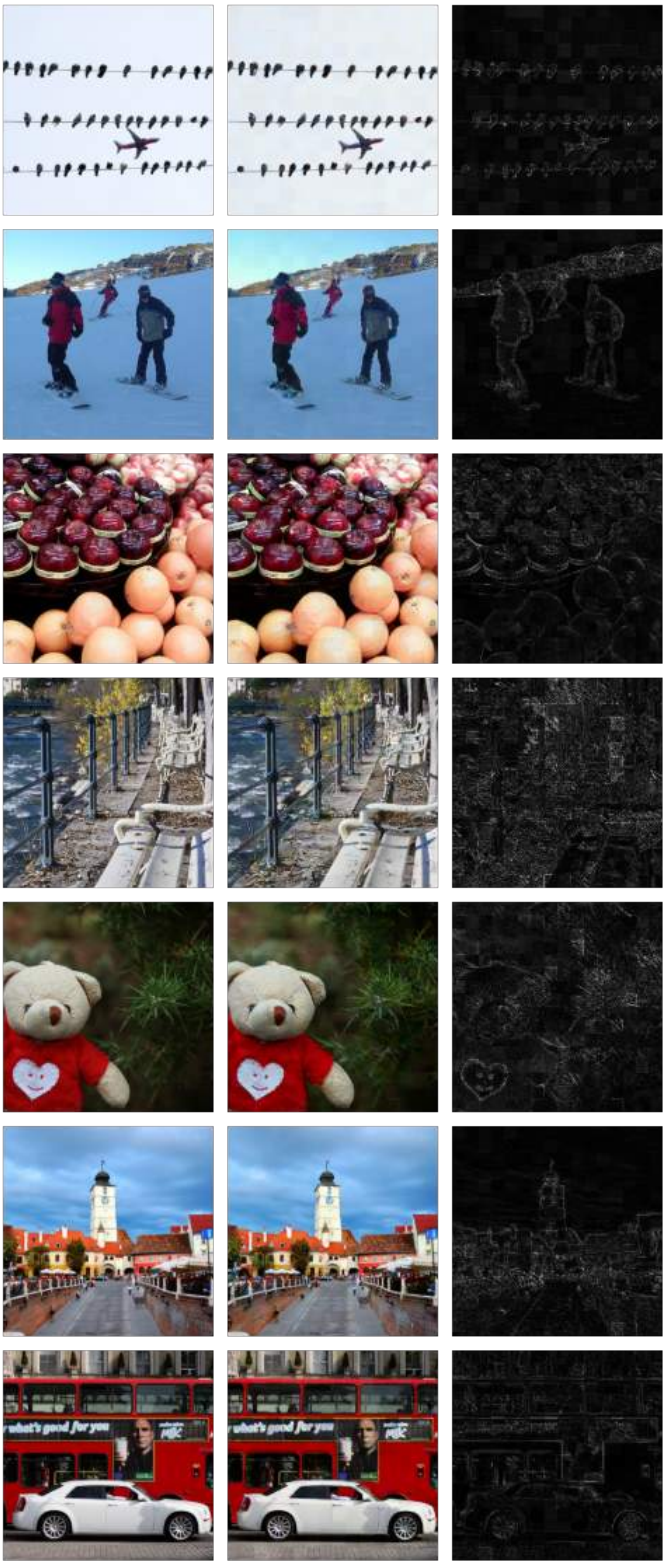
W wyniku kompresji stratnej większość *niedostrzegalnych* informacji w obrazie zostaje usunięta, stąd metoda znakowania wodnego nie jest w stanie wykorzystać tej części obrazu do osadzenia w niej dodatkowej wiadomości (co pozwoliłoby na osiągnięcie bardzo wysokiego współczynnika przezroczystości). W konsekwencji czego obserwujemy, że znak wodny powoduje subtelne zniekształcenia w *dostrzegalnej* przestrzeni obrazu, a zmiany te są zauważalne dla człowieka. Takie działanie algorytmu watermarkingowego jest wymuszone ze względu na konieczność zachowania znaku wodnego również po kompresji (odporność). Wpływ algorytmu kompresji stratnej jest także obserwowalny w przypadku pojemności. Zwróćmy uwagę, że w rozwiązaniach stenograficznych, gdzie zazwyczaj nie rozpatruje się ataków w celu usunięcia dodanych informacji, jesteśmy w stanie osadzić w obrazie

znacząco dłuższe wiadomości, często o rozmiarach równym obrazom wejściowym (lub nawet dwóm) [Bal17, Bal20]. W przypadku watermarkingu, ze względu na potrzebę zapewnienia odporności na ataki, jesteśmy w stanie obsłużyć tylko krótkie wiadomości (np. 30 bitów [ZKJFF18, LZC+20] lub 1024 bity [ANK+20] przy mocno zredukowanej liczbie typów ataków).

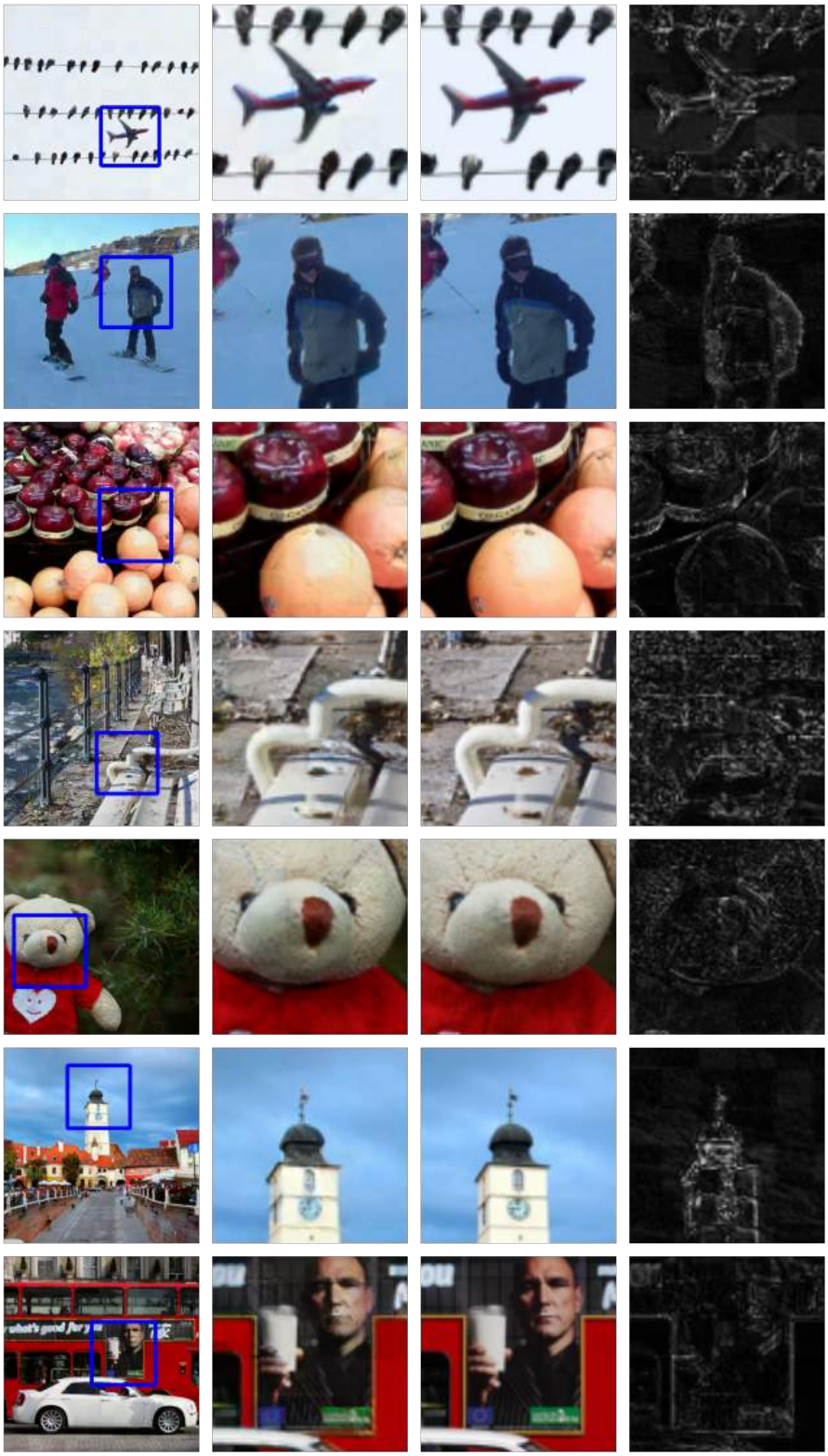
Odporność a jakość obrazu

Nasza metoda (dla pojemności wiadomości równej 32 bity i wyników odporności przedstawionych w tablicy 2.4) osiągnęła wyniki jakości obrazu liczonej przy użyciu PSNR równe 30.19 dB, 37.81 dB oraz 37.46 dB odpowiednio dla kanałów Y, U oraz V. Próbki zakodowanych obrazów zostały przedstawione na rysunku 2.5 oraz rysunku 2.6. Jakość zakodowanych obrazów z użyciem naszego rozwiązania jest zbliżona do wyników uzyskanych w [ZKJFF18]. W pracach [LZC+20, ANK+20], autorzy przedstawili nieco wyższe wartości PSNR. Wszystkie wspomniane metody osiągnęły jakość zbliżoną do algorytmów kompresji stratnej z zadanymi parametrami [Bar06], gdzie średnie uzyskiwane wartości PSNR dla wszystkich kanałów są zazwyczaj powyżej 30 dB. W wynikach nie uwzględniliśmy pracy [WA19], w której osiągnięto wysoką odporność na ataki, jednak metoda znacząco modyfikuje obraz, uzyskując przy tym niskie wartości PSNR.

Osiągane wyniki zniekształceń porównaliśmy z wynikami jakości osiąganymi przez algorytmy kompresji stratnej. Na próbkach ze zbioru testowego policzyliśmy wartości PSNR po przetworzeniu ich przez algorytm JPEG z współczynnikiem jakości $q = 50$ oraz próbkowaniem 4:2:0. Osiągnięte wyniki PSNR jest równy 36.35 dB, 36.78 dB i 36.93 dB odpowiednio dla kanałów Y, U i V. Przy niestosowaniu techniki próbkowania 4:2:0 osiągnięte wyniki były nieznacznie lepsze, tj. 37.90 dB, 38.17 dB i 38.29 dB odpowiednio dla kanałów Y, U i V. Warto podkreślić, że nasze rozwiązanie uzyskuje wysoki poziom odporności (pojemność bitową) nawet po zastosowaniu ataku kompresji stratnej JPEG oraz próbkowania 4:2:0. Łatwo zauważyć, że algorytmy znakowania wodnego muszą modyfikować obraz dużo bardziej niż metody kompresji stratnej przy założeniu, że znak wodny ma być odczytany również po skompresowaniu. W innym przypadku niewielkie zmiany wynikające z dodania informacji byłyby usuwane przez kompresję. Po analizie wartości PSNR można zauważyć, że nasz algorytm umieszcza dodatkowe informacje głównie w kanale Y. Ten kanał jest najbardziej modyfikowany, a co za tym idzie, osiąga najmniejszy wynik PSNR.



Rysunek 2.5: Porównanie jakościowe obrazów wejściowych I_c (lewa kolumna) oraz obrazów zakodowanych I_e (środkowa kolumna). Prawa kolumna zwiera znormalizowaną różnicę między obrazami wejściowymi I_c oraz obrazami zakodowanymi I_e. Normalizację przeprowadziliśmy metodą min-max.



Rysunek 2.6: Porównanie jakościowe fragmentów obrazów I_e oraz I_c — miejsce przycięcia obrazów (pierwsza kolumna od lewej), fragment I_e (druga kolumna), fragment I_c (trzecia kolumna) oraz znormalizowane metodą min-max różnice przyciętych fragmentów (ostatnia kolumna). Widzimy, że na zakodowanych obrazach widoczne są pewne zniekształcenia. Tylko w ten sposób metoda jest w stanie osiągnąć zadowalającą odporność na kompresję.

2.7 Podsumowanie

W rozdziale przedstawiliśmy naszą metodę znakowania wodnego opartą na przestrzennym rozprzestrzenianiu bitów wiadomości. Nasza architektura opiera się na konwolucyjnych sieciach neuronowych i działa dla dowolnego rozmiaru obrazu wejściowego. Opracowaliśmy specjalną architekturę dla modelu enkodera, w której obraz wejściowy oraz wiadomość (w formie przestrzennej) są przekazywane do co drugiej warstwy sieci neuronowej. Sformułowaliśmy również nową i niestandardową funkcję kosztu do trenowania sieci neuronowych uwzględniającą dużą redundancję znaku wodnego wynikającą z zastosowania metody rozprzestrzenienia przestrzennego wiadomości. W porównaniu z poprzednimi metodami, nasz system znakowania wodnego zapewnia znaczną poprawę odporności w przypadku rozmycia gaussowskiego, zmiany rozmiaru i kompresji JPEG (grupa ataków lokalnych). Praca została rozszerzona o dodatkowe typy ataków, takie jak próbkowanie 4:2:0 czy rotacja. Ostatecznie osiągnęliśmy dokładność bitową powyżej 0.83 dla wszystkich rozważanych ataków. Co oznacza, że nasza metoda osiąga wysoką ogólną odporność przewyższającą poprzednie rozwiązania. W ramach pracy przeprowadziliśmy eksperymenty z grupowaniem ataków ze względu na ich zakres przetwarzania obrazu i ujawniliśmy pewne korelacje (zbliżoną odporność) między atakami z tych samych grup zależnie od ich doboru. Tym samym pokazujemy, że aby osiągnąć ogólną odporność metody znakowania wodnego opartej na konwolucyjnych sieciach neuronowych, wymagany jest odpowiedni dobór zestawu ataków do potoku treningowego.

Rozdział 3

Znakowanie wodne rozszerzone o schemat dyskryminator-detektor

3.1 Wprowadzenie

W tym rozdziale opisujemy rozszerzenie metody z rozdziału 2. Przy czym skupiamy się na rozszerzeniu funkcjonalności całego systemu znakowania wodnego i dopasowaniu go do warunków rzeczywistych w jakich tego typu system miałby działać. Obecnie najczęstszym zastosowaniem znakowania wodnego jest zapobieganie piractwu treści multimedialnych. Aby wykryć wyciek obrazu lub filmu i następnie zidentyfikować sprawcę, musimy przeszukać potencjalne źródła, na których owe treści mogą zostać opublikowane (np. torrenty lub nielegalne strony internetowe). To oznacza, że system watermarkingowy musi znacznie częściej przetwarzać treści, w których znak wodny nie został umieszczony niż z nałożonym znakiem wodnym. W rzeczywistych warunkach musimy rozszerzyć standardowy detektor znaku wodnego o inny podsystem, który najpierw pozwoli na określenie istnienia znaku wodnego w analizowanej treści lub alternatywnie wywnioskować, że dana treść pochodzi z naszego zasobu (np. w oparciu o podejście zero-watermarkingu lub analizy treści). W naszej pracy proponujemy metodę, która opiera się na połączeniu dwóch elementów – detektora oraz szybkiego dyskryminatora. Ten drugi ma za zadanie odróżnić, czy obraz zawiera znak wodny, czy też nie.

W tym rozdziale proponujemy nowy schemat detektor-dyskryminator (DD), który rozszerza powszechną architekturę enkoder-noiser-detektor oraz umożliwia określenie, czy w obrazie znajduje się znak wodny i następnie, w przypadku pozytywnym, wykrywa wiadomość. Mając na uwadze aplikacyjny charakter podejścia, proponujemy nową formułę, która uwzględnia

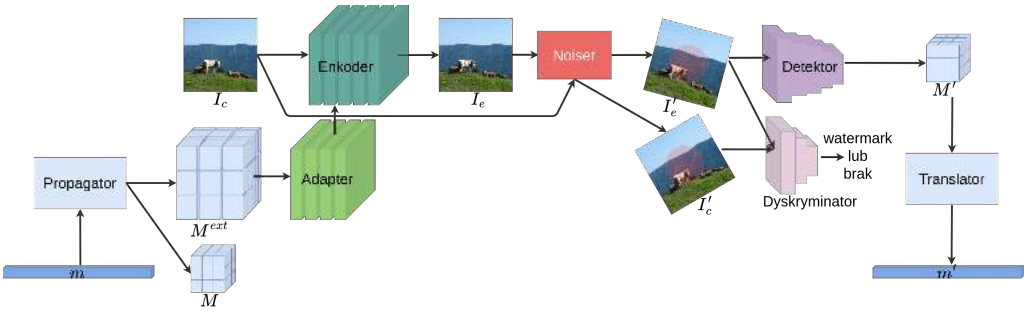
błąd fałszywych podejrzanych, tj. wykrycie błędnego (innego) klucza w zakodowanym obrazie lub niewłaściwą klasyfikację obrazu jako zawierającego znak wodny. Weryfikujemy nasze rozszerzone podejście i porównujemy je ze standardowym rozwiązaniem opartym na samym detektorze. Nasze podejście odzwierciedla bardziej *rzeczywiste* warunki zastosowania znakowania wodnego i do tej pory nie było badane dla systemów opartych na sieciach neuronowych.

Wprowadzamy również nową architekturę enkodera, która charakteryzuje się dodatkowymi warstwami przetwarzania wiadomości, które działają niezależnie od obrazów. To ulepszenie zostało celowo zaprojektowane w taki sposób, aby nie zwiększało złożoności obliczeniowej, gdy kodowanie obrazu jest stosowane wielokrotnie dla tego samego użytkownika (np. przetwarzanie kolejnych klatek wideo), co jest szczególnie istotne w aplikacjach wymagających wydajnego obliczeniowo kodowania (np. strumieniowanie wideo w czasie rzeczywistym). Nasze podejście przewyższa najnowsze metody w kontekście odporności na ataki (np. obracanie, zmiana rozmiaru) oraz kompresji JPEG. Ponadto, podobnie jak w rozdziale 2, rozważamy szersze spektrum ataków w porównaniu z poprzednimi pracami, co sprawia, że nasze rozwiązanie znacznie lepiej sprawdza się w rzeczywistych warunkach. Dokładność bitowa naszej metody wynosi co najmniej 0.86 dla każdego rodzaju badanego zniekształcenia. Osiągnęliśmy również dokładność bitową 0.90 dla kompresji JPEG, podczas gdy obecnie istniejące metody oparte na sieciach nieurnowych zapewniają co najwyżej 0.83. Nasza metoda utrzymuje również wysoką transparentność zakodowanych obrazów. Na koniec przedstawiamy, w jaki sposób nasze rozwiązanie można dostosować do domeny wideo oraz pokazujemy jej odporność na metody kompresji wideo MJPEG i MPEG [Ric03].

3.2 Metoda znakowania wodnego rozszerzona o dyskryminator

3.2.1 Ogólna architektura

W rzeczywistym scenariuszu (np. przeszukując Internet w celu indeksacji witryn, które mogą zawierać treści pirackie) musimy określić, czy analizowany obraz zawiera znak wodny, czy nie. Bez tego kroku konieczne byłoby przetworzenie każdej znalezionej treści w celu detekcji klucza (identyfikatora), co znacząco zwiększyłoby koszt obliczeniowy i mogło znacznie częściej prowadzić do fałszywego oskarżenia (patrz rozdział 3.4.2). Aby odróżnić od



Rysunek 3.1: Schemat architektury potoku treningowego. *Czerwone strzałki* oznaczają zniekształcenia powodowane przez noiser N. *Płaskie prostokąty* odnoszą się do deterministycznych (nieuczących się) komponentów potoku, a *kształty 3D* oznaczają trenowalne konwolucyjne sieci neuronowe.

siebie zniekształcone obrazy I'_c i I'_e, używamy *dyskryminatora* F, który wcześniej był używany tylko jako komponent treningu adwersarialnego i przyjmował obrazy bez zniekształceń I_c i I_e. W tym przypadku wykorzystujemy dyskryminator do obu zadań – ocenienia, czy obraz zawiera znak wodny oraz do treningu adwersarialnego.

Podobnie jak w rozwiązaniu opisywanym w rozdziale 2.4, nasz system zawiera 7 komponentów, gdzie 3 z nich są trenowalnymi sieciami neuronowymi – enkoder E, detektor D i dyskryminator F. Kolejny komponent to różniczkowalna warstwa – noiser N, który służy do dodawania sztucznego zniekształcenia do obrazów. Ostanie dwa elementy to propagator P oraz translator T, które są wymagane do wygenerowania wiadomości w postaci przestrzennej i późniejszego odwrócenia tej operacji. Dodatkowo wyróżniamy również pomocniczą sieć neuronową – adapter A, który jest rozszerzeniem enkodera E, ale przetwarza wiadomość M^{ext} niezależnie od obrazu wejściowego I_c. Architektura całego rozwiązania jest przedstawiona na rysunku 3.1.

3.2.2 Modele sieci neuronowych

Enkoder E i Adapter A

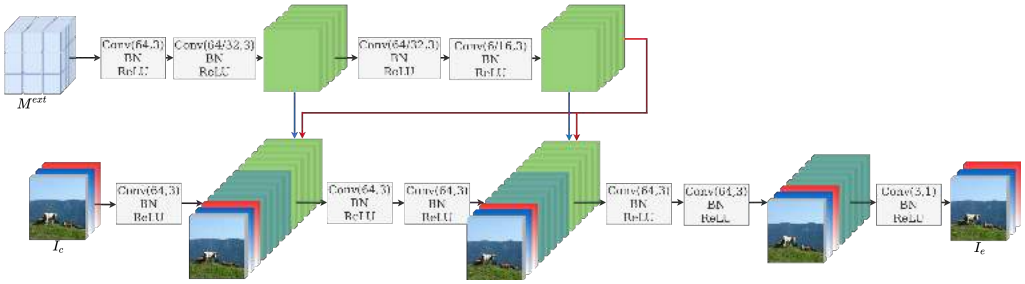
Operacja znakowania wodnego łączy obraz wejściowy I_c i wiadomość m, efektem czego jest zakodowany obraz I_e, który jest percepcyjnie podobny do I_c, ale zawiera przezroczystą „warstwę” zakodowanej wiadomości. W naszym rozwiązaniu korzystamy z propagatora, więc przed operacją kodowania konwertujemy wiadomość m do jej innej reprezentacji M^{ext}. Enkoder przyjmuje obraz wejściowy I_c w przestrzeni barw YCbCr. Oczywiście za-

kodowany obraz I_e może później zostać przywrócony do przestrzeni barw RGB.

W tym podejściu rozszerzamy standardowy enkoder oparty na sieci neuronowej o nowy moduł do adaptacji wiadomości. Rolą adaptera A jest dostarczenie „dogodnej" reprezentacji wiadomości M^{ext} do enkodera E. Adapter A jest zbudowany z czterech bloków *ConvBNReLU*. W pracy proponujemy dwa tryby adaptera: *przejście proste*, które przechodzi przez wszystkie warstwy i zwraca tensor wyjściowy oraz *przejście rzadkie*, które zwraca dwa tensory wyjściowe – po przejściu drugiej i ostatniej warstwy.

Enkoder E jest zbudowany z sześciu bloków *ConvBNReLU*. Tensor wyjściowy po pierwszej i trzeciej warstwie jest łączony z obrazem wejściowym i tensorami wyjściowymi z adaptera, tj. $I_c \| E(I_c)_i \| A(M^{ext})_{i+1}$, gdzie $E(\cdot)_i$ oraz $A(\cdot)_i$ są tensorami wynikowymi dla odpowiednio enkodera oraz adaptera po i-tej warstwie oraz $i \in \{1, 3\}$. Następnie tensory wyjściowe po piątej warstwie są łączone z obrazem wejściowym I_c, po czym stosowana jest szybka warstwa konwolucyjna z rozmiarem jądra równym 1×1 w celu wygenerowania zakodowanego obrazu I_e. Ogólną architekturę opisywanego komponentu do osadzania znaku wodnego oraz niektóre parametry sieci przedstawiono na rysunku 3.2.

Adapter jest komponentem możliwym do trenowania (w przeciwieństwie do propagatora), dzięki czemu może zostać nauczony właściwej reprezentacji wiadomości M^{ext}, wygodnej do przetwarzania przez enkoder. Jest to ważny aspekt, ponieważ ta reprezentacja jest przekazywana bezpośrednio do ukrytych warstw enkodera, dzięki czemu warstwy konwolucyjne mogą przetwarzać tensor zawierający dobrze dostosowaną i rozproszoną reprezentację wiadomości, zamiast tensora binarnego zawierającego tylko wartości 0 lub 1. Co więcej, owa reprezentacja jest obliczana bez dostępu do obrazu wejściowego I_c. Zatem adapter A przetwarza M^{ext} niezależnie od zawartości I_c, co umożliwia buforowanie wynikowych tensorów wiadomości w celu ich dalszego przetwarzania na kolejnych obrazach. Ma to kluczowe znaczenie w przypadku projektowania systemów znakowania wodnego do zastosowań w obszarze wideo, w którym często ustalone są pewne wymagania odnośnie przetwarzania w czasie rzeczywistym. Warto zaznaczyć, że znakowanie filmów jest obecnie znacznie bardziej komercyjnie rozpowszechnione niż watermarking obrazów, dlatego metody znakowania wodnego obrazu umożliwiające łatwe rozszerzenie do domeny wideo (np. wykonując przetwarzanie klatka po klatce) znacząco zyskują w kontekście ich komercjalizacji. W usługach *over-the-top* (OTT) strumieniowanie wideo jest poważnie ograniczone przez przepustowość serwerów treści (cache oraz origin), które ze względu



Rysunek 3.2: Architektura enkodera E rozszerzona o adapter A. Schemat zawiera dwa przejścia adaptera - czerwone strzałki odnoszą się do przejścia prostego, a niebieskie strzałki do przejścia rzadkiego. Pierwszy argument w funkcji *Conv* odnosi się do liczby kanałów (prosty/rzadki), a drugi do rozmiaru jądra. W trakcie przetwarzania utrzymujemy wymiary obrazu, więc dla rozmiaru jądra równego 3×3 zawsze ustawiamy dopełnienie równe 1×1, a dla rozmiaru jądra równego 1×1 nie używamy dopełnienia.

na ograniczenia pamięciowe i obliczeniowe nie są w stanie przetworzyć treści wideo dla każdego użytkownika z osobna, tj. wygenerować i wysłać strumień wideo z unikalnym kluczem dla każdego odbiorcy. W konsekwencji, aby umożliwić wytworzenie unikalnego strumienia wideo ze znakiem wodnym dla każdego użytkownika, enkoder musi działać po stronie klienta lub po stronie serwera wykorzystując tzw. kodowanie A/B, a przy tym obsługiwać wysokiej jakości wideo, wiele profili (rozdzielczości) oraz kanałów [Fri18]. To oznacza, że proponowana architektura enkodera (i pośrednio innych komponentów systemu) musi być stosunkowo niewielka i płytka (w rozumieniu modelowania sieci neuronowych), aby umożliwić przetwarzanie wideo w czasie rzeczywistym. To wyjaśnia powód, dla którego adapter A jest projektowany jako komponent niezależny od treści obrazu. Nasze podejście nie wykracza poza złożoność enkodera w przypadku watermarkingu wideo, ponieważ adapter przetwarza wiadomość tylko raz, a następnie wynik tej operacji jest wykorzystywany do generowania znaku wodnego w kolejnych klatkach.

Detektor D

Detektor D wraz z enkoderem E jest głównym elementem naszego systemu znakowania wodnego. Obie sieci muszą współpracować podczas treningu, aby znaleźć równowagę między przejrzystością i odpornością, przy ustalonej z góry pojemności i złożoności obliczeniowej. Obie sieci w ramach wspólnego treningu określają wspólny „schemat" procedury kodowania-dekodowania,

Tablica 3.1: Architektura detektora D. Zauważmy, że w definicji modelu odwołujemy się do bloku *ConvBNReLU*, który jest opisany w dolej części tablicy.

Warstwa	Parametry
*ConvBNReLU**	64 kanały, 3×3 rozmiar jądra, 1×1 padding [powtórzony $\times 8$]
AdaptiveAvgPool2D	$\frac{H}{b} \times \frac{W}{b}$ rozmiar wyjścia
*ConvBNReLU**	64 kanały, 1×1 rozmiar jądra, bez paddingu
Conv2D	k' kanały, 1×1 rozmiar jądra, bez paddingu
BatchNorm	-
Sigmoid	-
**Architektura ConvBNReLU(F_1, F_2, F_3)*:	
Conv2D	F_1 kanały, F_2 rozmiar jądra, F_3 padding
BatchNorm	-
ReLU	-

który jest unikalny dla każdego procesu treningowego i jest znany tylko komponentom uczestniczącym w danym treningu.

Podobnie jak w enkoderze i dyskryminatorze, wejściem do detektora jest obraz przekonwertowany do przestrzeni barw YCbCr. Model jest oparty na ośmiu blokach *ConvBNReLU*. Następnie tensor wyjściowy jest przetwarzany przez warstwę adaptacyjnego poolingu uśredniającego. Po czym jest przekazywany do kolejnego bloku *ConvBNReLU* i jednej warstwy konwolucyjnej z rozmiarami jąder równymi 1×1, zarówno dla bloku, jak i ostatniej warstwy. Na koniec stosujemy normalizację na partii (ang. *batch normalization*) i sigmoidalną funkcję aktywacji. Szczegółowa architektura modelu jest opisana w tablicy 3.1. Architektura detektora jest zbliżona do architektury tej samej sieci opisywanej w rozdziale 2.4.

Dyskryminator F

Zazwyczaj główną rolą dyskryminatora F jest zastosowanie podejścia opartego na *treningu adwersarialnym* [GPAM+14, HD17] w celu poprawy percepcyjnego podobieństwa zakodowanych i oryginalnych obrazów. W naszym rozwiązaniu rozszerzamy rolę dyskryminatora F w potoku treningowym, używając go do wskazania czy obraz zawiera wiadomość, czy nie. W naszym podejściu dyskryminator F na wejściu przyjmuje obrazy zniekształcone I'_c oraz I'_e, dzięki czemu zwiększamy jego odporność na ataki oraz kompre-

Tablica 3.2: Architektura dyskryminatora F. Zauważmy, że w definicji modelu odwołujemy się do bloku *ConvBNReLU*, który jest opisany na dole.

Warstwa	Parametry
*ConvBNReLU**	64 kanały, 3×3 rozmiar jądra, 1×1 padding [powtórzony $\times 3$]
GlobalMaxPooling2D	1×1 rozmiar wyjścia
Linear	1 wartość wyjścia
Sigmoid	-
*Architektura *ConvBNReLU(F_1, F_2, F_3)*:	
Conv2D	F_1 kanały, F_2 rozmiar jądra, F_3 padding
BatchNorm	-
ReLU	-

się. Z perspektywy generalizacji sieci neuronowej istotne jest, żeby obraz wejściowy I_c był zaszumiany w taki sam sposób jak obraz zakodowany I_e.

Sygnałem wejściowym do dyskryminatora są obrazy zniekształcone I'_c lub I'_e reprezentowane w przestrzeni barw YCbCr. Model łączy 3 bloki *ConvBNReLU*, warstwę globalnego poolingu maksymalizującego, warstwę liniową z pojedynczą wartością wyjściową i sigmoidalną funkcję aktywacji. Architekturę dyskryminatora przedstawiono w tablicy 3.2. Podobnie jak w przypadku detektora, również architektura dyskryminatora jest podobna do tej przedstawionej w rozdziale 2.4.

Wyniki odporności metody watermarkingowej wykorzystującej dyskryminator do określenia istnienia znaku wodnego w obrazie przedstawiono w rozdziale 3.4.

Zalety dyskryminatora o dwóch rolach oraz treningu adwersarialnego

W celu zbudowania modułu rozróżniającego, czy obraz zawiera znak wodny, moglibyśmy rozważyć proste podejście oparte na dodaniu kolejnej sieci neuronowej do potoku treningowego, wyspecjalizowanej tylko w tym celu. Zauważamy, że to proste podejście niesie ze sobą wiele niedogodności i jest mniej efektywne w porównaniu z naszym podejściem opartym na treningu adwersarialnym dyskryminatora i jego rozbudowanych funkcjonalnościach.

Proste podejście wymagałoby zaprojektowania potoku treningowego zawierającego cztery komponenty trenowalne. Takie rozwiązanie wydłużyłoby trening i wymagałoby większych zasobów obliczeniowych i pamięciowych. W proponowanym przez nas podejściu nie zwiększamy złożoności potoku

treningowego i wykorzystujemy istniejący komponent (dyskryminator) na dwa sposoby:

- Aby zwiększyć percepcyjne podobieństwo między obrazami zakodowanym i podstawowym. W przeciwieństwie do innych rozwiązań (np. [ZKJFF18]), porównujemy zniekształcone obrazy I'_{en} i I'_{co}. Podczas szkolenia potoku stosujemy te same ataki dla obu obrazów, co powoduje, że adekwatne porównanie jest nadal możliwe, a dyskryminator jest w stanie nauczyć się wzorców, które są efektem przetwarzania przez enkoder, a nie zniekształceniami spowodowanymi przez noiser.
- Aby określić, czy obraz zawiera znak wodny. Do tego celu wykorzystujemy zniekształcone obrazy I'_{en} i I'_{co}, dzięki czemu dyskryminator może pracować w środowisku rzeczywistym, np. jako część systemu indeksującego, który znajduje nielegalne kopie obrazów w Internecie.

Co więcej, trening adwersarialny pozwala na oddzielne trenowanie detektora i dyskryminatora, tj. uruchamiamy dwie niezależne iteracje wstecznej propagacji z gradientami obliczonymi dla dwóch różnych funkcji kosztu, a co za tym idzie aktualizujemy wagi dyskryminatora niezależnie od aktualizacji wag detektora. Dzięki temu detektor i dyskryminator mogą nauczyć się nieco innych wzorców znajdowania i wyodrębniania znaku wodnego, co może zwiększyć skuteczność rozwiązania.

3.2.3 Warstwa zaszumiająca oraz rozważane ataki

W tym rozwiązaniu również wykorzystujemy podejście do treningu oparte na schemacie enkoder-noiser-detektor oraz rozważamy ten sam zbiór ataków jaki zostały opisany w rozdziale 2.4.4.

3.2.4 Potok treningowy

W iteracji potoku treningowego najpierw wiadomość m przekazujemy do propagatora P, który zwraca dwa warianty wiadomości w postaci przestrzennej M i M^{ext}, tj.

$$M^{ext}, M = P(m). \tag{3.1}$$

Następnie wykorzystujemy adapter A do transformacji wiadomości M^{ext} oraz kodujemy obraz podstawowy wykorzystując do tego enkoder E:

$$I_e = E(I_c, A(M^{ext})). \tag{3.2}$$

Wynikiem operacji jest zakodowany obraz I_e, który jest później zniekształcany przez warstwę noisera N. Nosier najpierw wybiera losowo rodzaj ataku (co iterację treningową) i wykonuje tę samą operację na obrazach zarówno zakodowanych I_e, jak i wstępnych I_c, dzięki czemu mamy:

$$\begin{aligned} I'_e, M &= N(I_e; I_c, M), \\ I'_c &= N(I_c). \end{aligned} \tag{3.3}$$

Zwróćmy uwagę, że niektóre rodzaje ataków wymagają obrazu wejściowego I_c w celu zniekształcenia zakodowanego obrazu I_e (np. dropout). Co więcej, w przypadku niektórych rodzajów ataków, które wpływają przestrzennie na obraz, kalibrujemy również wiadomość M, która jest później używana jako *ground truth* do trenowania detektora (np. przy kadrowaniu przycinamy wiadomość M w celu dostosowania jej do wiadomości osadzonej w wykadrowanej części obrazu). Oczywiście te operacje nie są stosowane podczas testów. Wykonujemy je tylko w celu wyszkolenia enkodera, detektora i dyskryminatora.

Projektujemy rozwiązanie do znakowania wodnego typu blind (nie dostarczamy oryginalnego obrazu przy detekcji), dlatego detektor D na wejściu przyjmuje tylko zakodowany obraz po zniekształceniu I'_e, a następnie odczytuje zakodowaną wiadomość M':

$$M' = D(I'_e). \tag{3.4}$$

Dyskryminator F ocenia obrazy wejściowe po zniekształceniu I'_c oraz obrazy zakodowane po zniekształceniu I'_e

$$F(I' \in \{I'_c, I'_e\}) \in [0, 1]. \tag{3.5}$$

Przyjmujemy, że obraz I' zawiera znak wodny, jeśli $F(I')$ zwróci wartość bliską 1, z kolei $F(I') \approx 0$ wskazuje na brak znaku wodnego w obrazie. W naszym podejściu trenujemy dyskryminator w dwóch celach – do poprawy podobieństwa percepcyjnego obrazów oraz jako pomocniczy składnik naszego podejścia detektor-dyskryminator. W potoku treningowym dyskryminator jest trenowany na zniekształconych obrazach. Dzięki temu, podobnie jak detektor, może posłużyć do analizy treści multimedialnych z zasobów zewnętrznych (np. z przeszukiwanych stron internetowych). Jednocześnie wciąż możliwe jest wytrenowanie dyskryminatora w oparciu o trening adwersarialny i dzięki temu wpłynięcie na percepcyjną jakość zakodowanych obrazów.

3.2.5 Funkcja kosztu

Głównym celem treningu metody znakowania wodnego jest uzyskanie transparentności i odporności na najwyższym możliwym poziomie, przy założonej z góry pojemności wiadomości i złożoności bezpośrednio zdefiniowanej przez architekturę sieci. W procesie treningu staramy się kontrolować transparentności za pomocą błędu średniokwadratowego między I_c i I_e, a zatem:

$$L_E(I_e, I_c) = \frac{1}{H \cdot W \cdot \mathrm{Ch}} ||I_c - I_e||_2^2, \tag{3.6}$$

gdzie $|| \cdot ||_2$ jest normą Frobeniusa. Aby odpowiednio wytrenować sieć do dekodowania wiadomości, zastosowaliśmy podejście *średnia-wariancja* z rozdziału 2.4.6:

$$L_D^{\mu}(M, M') = \frac{b^2}{H \cdot W \cdot (n+k)} ||M - M'||_1 \tag{3.7}$$

oraz

$$L_D^{\sigma^2}(M, M') = \frac{b^2}{H \cdot W} \sum_{h=0}^{H_b} \sum_{w=0}^{W_b} Var(|M_{hw} - M'_{hw}|). \tag{3.8}$$

gdzie $W_b = \frac{W}{b} - 1$, $H_b = \frac{H}{b} - 1$, operator $|V|$ zwraca wektor wartości bezwzględnych wektora V, a funkcja $Var\colon \mathbb{R}^{(n+k)} \to \mathbb{R}$ zwraca wariancję wektora. Funkcja straty *średnia-wariancja* promuje uczenie *wszystkich elementów w niektórych krotkach* ponad *niektórych (powielonych) elementów we wszystkich krotkach*. Użycie takiej funkcji powoduje, że w trakcie treningu sieci zbiegają do stanu, w którym niektóre krotki M'_{xy} są w całości odczytywane poprawnie przed detektor. Tym samym unikamy przypadku, w którym poprawnie odczytywane są tylko wartości niektórych przekrojów (ang. *slices*) tensora wiadomości. Zauważmy, że ze względu na dużą nadmiarowość tych samych krotek w tensorze M', ten sposób uczenia jest pożądany.

W potoku treningowy stosujemy trening adwersarialny enkodera E przy użyciu dyskryminatora F. Celem enkodera jest wygenerowanie obrazów, które będą rozpoznane przez dyskryminator jako obrazy podstawowe (bez widocznych znaków wodnych). W naszym potoku treningowym obrazy te są wcześniej zniekształcane przez noiser N. Ten sposób treningu pozwala zwiększyć jakość zakodowanych obrazów i jednocześnie wykorzystać dyskryminator do określenia, czy w obrazie istnieje znak wodny. Stąd celem dyskryminatora jest odróżnienie zniekształconych obrazów podstawowych

I'_c od zniekształconych obrazków zakodowanych I'_e, co jest wyrażone w postaci funkcji:

$$L_F(I'_c, I'_e) = \log(F(I'_c)) + \log(1 - F(I'_e)). \tag{3.9}$$

Aby wytrenować enkoder E, adapter A i detektor D, minimalizujemy funkcję kosztu względem rozkładu obrazów i wiadomości, mianowicie:

$$\mathbb{E}_{I_c,M}[L_E + \lambda_F L_F + \lambda_D^{\mu} L_D^{\mu} + \lambda_D^{\sigma^2} L_D^{\sigma^2}], \tag{3.10}$$

ze współczynnikami λ. W celu wytrenowania dyskryminatora F, minimalizujemy funkcję kosztu względem rozkładu obrazów, czyli $\mathbb{E}_{I_c}[-L_F]$.

3.2.6 Szczegóły treningu

Aby wytrenować oraz przetestować naszą metodę użyliśmy zbioru danych COCO [LMB+14]. Ze zbioru wylosowaliśmy 10000 obrazów dla podzbioru treningowego i zmieniliśmy ich rozmiar do 256×256 pikseli. Do przetestowania naszej metody, losujemy 1000 kolejnych obrazów z tego samego zbioru danych. Nasze rozwiązanie wytrenowaliśmy dla wiadomości o długości $L = 32$, które podczas treningu losowaliśmy. Mechanizm rozproszenia przestrzennego również działał w sposób losowy. We wszystkich eksperymentach ustanowiliśmy parametry propagatora na $k = 2$ i $b = 16$ oraz parametr translatora $o = 3$. Podzieliliśmy wiadomość m na 16 krotek i zreplikowaliśmy, aby wypełnić bloki jednostkowe o rozmiarze $16 \times 16 \times 6$. Oczekiwana liczba redundantnych bloków była równa 16. Potok wytrenowaliśmy według następujących parametrów: $\lambda_E = 2.4$ (dla przejścia prostego) i bez adaptera), $\lambda_E = 1.6$ (dla przejścia rzadkiego), $\lambda_D^{\mu} = 1.0$, $\lambda_D^{\sigma^2} = 1.0$ oraz $\lambda_F = 0.05$. Wszystkie sieci neuronowe przyjmowały obrazy w przestrzeni barw YCbCr. Obrazy w tej samej przestrzeni były zwracane przez enkoder. Szczegóły architektury enkodera i adaptera są przedstawione na rysunku 3.2.

Do treningu wykorzystaliśmy optymalizator Adam [KB15] ze współczynnikiem uczenia równym 0.001 (pozostałe parametry nie zostały zmienione względem domyślnych). Potok był trenowany z wielkością partii równą 12 przez 100 epok i przy zastosowaniu wszystkich ataków (jeden rodzaj w każdej iteracji treningu). Przez kolejne 20 epok uczony był tylko dyskryminator (wagi innych sieci neuronowych nie były aktualizowane).

Do szkolenia systemu wykorzystaliśmy dwa procesory graficzne Nvidia RTX 2080Ti 11GB VRAM. Trening jednej epoki trwał około 370 sekund, podczas gdy w fazie wnioskowania jesteśmy w stanie przetworzyć około 110

obrazów na sekundę przez jeden komponent (np. enkoder) używając w tym celu jednego GPU tej samej klasy.

3.3 Wyniki analizy odporności oraz transparentności

W tym rozdziale przedstawiamy wydajność procedury kodowania i dekodowania naszego rozwiązania. Aby porównać naszą metodę z innymi rozwiązaniami, stosujemy powszechne podejście do oceny odporności, które polega na obliczaniu wydajności bitowej wykrytych wiadomości. Aby ilościowo porównać transparentność naszej metody, używamy szczytowego stosunku sygnału do szumu (PSNR). W naszych eksperymentach ustaliliśmy długość wiadomości na 32, co oznacza, że jest równa lub większa od innych ostatnich metod – 30 bitów jest wykorzystywane w [ZKJFF18, LZC+20, ANK+20] i 32 bity są używane w [ZXCIV19] oraz w naszej metodzie z rozdziału 2.4. Ocena rozwiązania odbywa się na 1000 losowo wybranych obrazach z bazy danych COCO i 50 filmach (2 sekundy, 50 klatek) z naszych zasobów własnych. W przypadku kodowania wideo znak wodny jest dodawany niezależnie do każdej klatki, a przetwarzanie odbywa się klatka po klatce, bez konieczności posiadania informacji o sposobie kompresji wideo (np. o rodzaju klatki I, P lub B) [Ric03].

3.3.1 Transparentność a odporność

Łatwo zauważyć, że transparentność jest ujemnie skorelowana z odpornością (przy założeniu stałej pojemności). Dlatego projektując system znakowania wodnego, musimy zachować równowagę między jakością obrazów a odpornością na ataki. Podobnie jak w rozdziale 2.6, podczas uczenia sieci neuronowych przyjmujemy minimalny akceptowalny poziom przezroczystości, który jest przyjęty na poziomie PSNR = 30 dB dla każdego kanału Y, Cb i Cr. Oznacza to, że bierzemy pod uwagę tylko te enkodery, które są w stanie generować zakodowane obrazy z wartością PSNR $>$ 30 dB dla wszystkich kanałów. Ten próg został pierwotnie zaprojektowany dla algorytmów kompresji [Bar06], ale nadaje się również do określenia jakości metod znakowania wodnego, gdzie jakość obrazów jest degradowana w celu dodania dodatkowych informacji. Mając ustalony próg akceptacji dla transparentności, pracujemy nad zwiększeniem odporności naszej metody znakowania wodnego.

Innym ważnym czynnikiem wpływającym na transparentność jest zbiór ataków, przeciwko którym metoda ma być odporna oraz to w jakim stopniu

te ataki same wpływają na transparentność obrazu. Odporność oznacza, że zakodowana wiadomość musi zostać wyodrębniona nawet po zastosowaniu ataku, co sugeruje pewną intuicję, że metoda znakowania wodnego musi bardziej modyfikować obraz, aby zachować wiadomość nawet po przeprowadzeniu ataku na obrazie. Dlatego poziom zniekształceń wytworzonych przez kompresję algorytmem JPEG z parametrem $q = 50$ (najbardziej złożony rozważany atak) traktujemy jako linię bazową (ang. *base line*) dla jakości obrazu (intuicyjnie maksymalną jaką możemy osiągnąć) i dążymy do tego, aby nasze wyniki przezroczystości były jak najbardziej zbliżone do ustalonej linii bazowej.

3.3.2 Analiza odporności

W tym rozdziale przedstawiamy wyniki odporności naszej metody. Oceniamy trzy zaproponowane podejścia – przejście proste, przejście rzadkie i enkoder bez adaptera. Nasze wyniki dla ostatniego przypadku pokazują, że uaktualnienie naszej architektury o dyskryminator nie zmniejsza odporności w porównaniu z metodą z rozdziału 2, która również używa propagatora i translatora jako komponentów metody znakowania wodnego. Metoda z zastosowanym przejściem rzadkim przewyższa inne rozwiązania w kwestii odporności na algorytmy kompresji – JPEG, MJPEG i MPEG, podczas gdy najlepsza ogólna dokładność (mierzona jako najniższa wydajność bitowa spośród wszystkich ataków) została osiągnięta przy zastosowaniu przejścia prostego. Uważamy, że przejście proste jest odpowiednie do celów ogólnych, dlatego w kolejnych rozdziałach opisujących dalsze wyniki (3.3.3 i 3.4) opieramy się na tym podejściu. Jednakże przejście rzadkie wyróżnia się bardzo wysoką odpornością na algorytmy kompresji. Ta metoda może być właściwa do zastosowania w niektórych specjalnych scenariuszach, w których dostarczany obraz jest mocno skompresowany, np. przy transmisjach strumieniowych przez Internet.

Ostatecznie obserwujemy, że nasze rozwiązanie charakteryzuje się znaczną poprawą skuteczności w przypadku rotacji, zmiany rozmiaru i kompresji JPEG w porównaniu do poprzednich metod. Co więcej, najniższa dokładność bitowa dla naszego rozwiązania jest równa 0.86 (przejście proste), podczas gdy wcześniej było to 0.831 dla metody z rozdziału 2 i poniżej 0.7 dla pozostałych metod. To oznacza, że nasza metoda osiąga najwyższą odporność niezależnie od ataków.

Wyniki wydajności bitowej policzonej dla naszego rozwiązania i innych ostatnich prac prezentujemy w tablicy 3.3 oraz tablicy 3.4. Przedstawiamy

wyniki dla naszych dwóch adapterów — prostego i rzadkiego oraz enkodera bez adaptera.

Warto zauważyć, że porównujemy nasze rozwiązanie tylko z tymi metodami, które uwzględniają praktyczny i realistyczny zestaw ataków oraz stosują metodologię badawczą podobną do naszej, zwłaszcza biorąc pod uwagę kompresję i atak przycinania, które sprawiają, że problem znakowania wodnego staje się dużo bardziej złożony, a jednocześnie odzwierciedla bardziej realistyczne warunki pracy. Porównanie z innymi metodami nie jest miarodajne ze względu na fundamentalne różnice i nieujednoliconą metodologię badawczą, np. ignorowanie ataku przycinania znacznie upraszcza problem, ponieważ pozwala na rozłożenie wiadomości przestrzennie na całym obrazie (mechanizm rozłożenia przestrzennego lub podejście polegające na przypisaniu wiadomości do każdego piksela [ZKJFF18, LZC+20] nie są wtedy wymagane). Z kolei zignorowanie kompresji uniemożliwia zastosowanie metody znakowania wodnego w domenie wideo, a także pozwala na uzyskanie znacznie wyższych wyników transparentności.

3.3.3 Jakość obrazu

Aby uzyskać porównywalne wyniki transparentności, obliczamy PSNR według schematu przedstawionego w [ZKJFF18] i później powtórzonego w innych ostatnich pracach, tj. używamy mocno zmniejszonych obrazów o stosunkowo wysokiej entropii. PSNR dla naszej metody jest równy 31.08 dB, 44.37 dB i 45.27 dB odpowiednio dla kanałów Y, U i V. Ilościowe wyniki transparentności naszej metody, a także innych ostatnich rozwiązań są przedstawione w tablicy 3.5. Natomiast jakościowe wyniki zakodowanych obrazów są przedstawione na rysunku 3.3 oraz rysunku 3.4.

Nasza metoda osiąga najwyższe wartości PSNR dla dwóch kanałów – U i V, podczas gdy PSNR dla kanału luminancji (Y) jest niższy. Jednym z powodów tego wyniku może być wysoka odporność na próbkowanie 4:2:0 oraz kompresję JPEG (które są najwyższe w przypadku naszej metody). Zauważmy, że JPEG i próbkowanie 4:2:0 znacznie redukują kanały chrominancji, podczas gdy kanał luminancji jest kompresowany w mniejszym stopniu. To oznacza, że metoda znakowania wodnego musi zmodyfikować kanał luminancji, aby zapobiec usunięciu zakodowanej wiadomości przez kompresję JPEG lub próbkowanie 4:2:0. Warto zauważyć, że nasza metoda przewyższa linię bazową transparentności liczonej dla kompresji JPEG z parametrem $q = 50$ dla kanałów U i V, natomiast dla kanału Y wyniki są gorsze.

Tablica 3.3: Wyniki odporności dla naszych metod i innych rozwiązań, tj. *S+C* (rozdział 2), HiDDeN [ZKJFF18], DADW [LZC+20] oraz RedMark [ANK+20], w odniesieniu do średniej wydajności bitowej. Wszystkie metody zostały ocenione pod kątem przezroczystości raportowanej w tablicy 3.5. Zauważmy, że metody interpolacji przy zmianie rozmiaru nie są zadeklarowane w [ANK+20, LZC+20].

Ataki	DD (*Nasza*)			*Nasza*
	Prosta	Rzadka	bez A	*S+C*
Identyczność	1.000	1.000	1.000	1.000
Przycinanie($p = 0.3$)	0.860	0.831	0.834	0.832
Cropout($p = 0.3$)	0.921	0.902	0.892	0.902
Dropout($p = 0.5$)	0.981	0.856	0.963	0.972
Rotacja($\alpha = 5°$)	**0.956**	0.716	0.884	0.842
Rozmycie($\sigma = 2$)	**0.991**	0.935	0.981	0.986
Rozmycie($\sigma = 4$)	**0.989**	0.974	0.984	0.982
Próbkowanie(4:2:0)	**0.993**	0.991	0.985	0.984
Rozmiar($s = 0.5$, $m = N$)	**0.913**	0.793	0.849	0.849
Rozmiar($s = 0.5$, $m = L$)	0.886	0.658	0.898	**0.908**
JPEG($q = 50$)	0.902	**0.935**	0.829	0.831
Pojemność (liczba bitów)	**32**	**32**	**32**	**32**

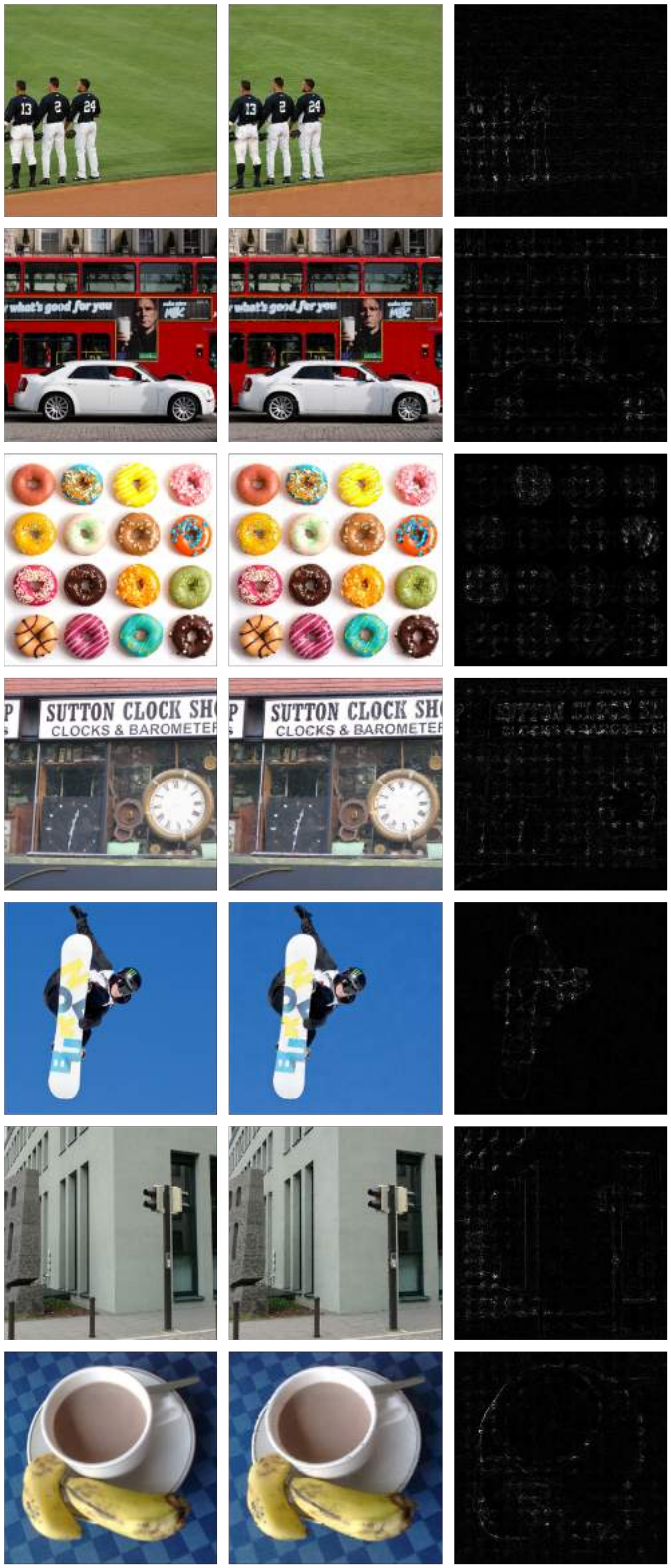
Ataki	Metody		
	HiDDeN	DADW	RedMark
Identyczność	1.000	1.000	1.000
Przycinanie($p = 0.3$)	**1.000**	**1.000**	-
Cropout($p = 0.3$)	**0.940**	-	0.925
Dropout($p = 0.5$)	**1.000**	**1.000**	≈0.99
Rotacja($\alpha = 5°$)	-	-	-
Rozmycie($\sigma = 2$)	0.960	0.600	0.500
Rozmycie($\sigma = 4$)	0.820	0.500	0.500
Próbkowanie(4:2:0)	-	-	-
Rozmiar($s = 0.5$, $m = N$)	-	0.671	0.819
Rozmiar($s = 0.5$, $m = L$)	-		
JPEG($q = 50$)	0.670	0.817	0.746
Pojemność (liczba bitów)	**30**	**30**	**30**

Tablica 3.4: Odporność mierzona za pomocą wydajności bitowej dla kompresji wideo MJPEG i MPEG dla naszej metody i ostatniego algorytmu opartego na metodzie głębokiego uczenia. Oba rozwiązania spełniają te same wymagania dotyczące pojemności i przejrzystości, podczas gdy nasze rozwiązanie jest około 4.5 razy szybsze.

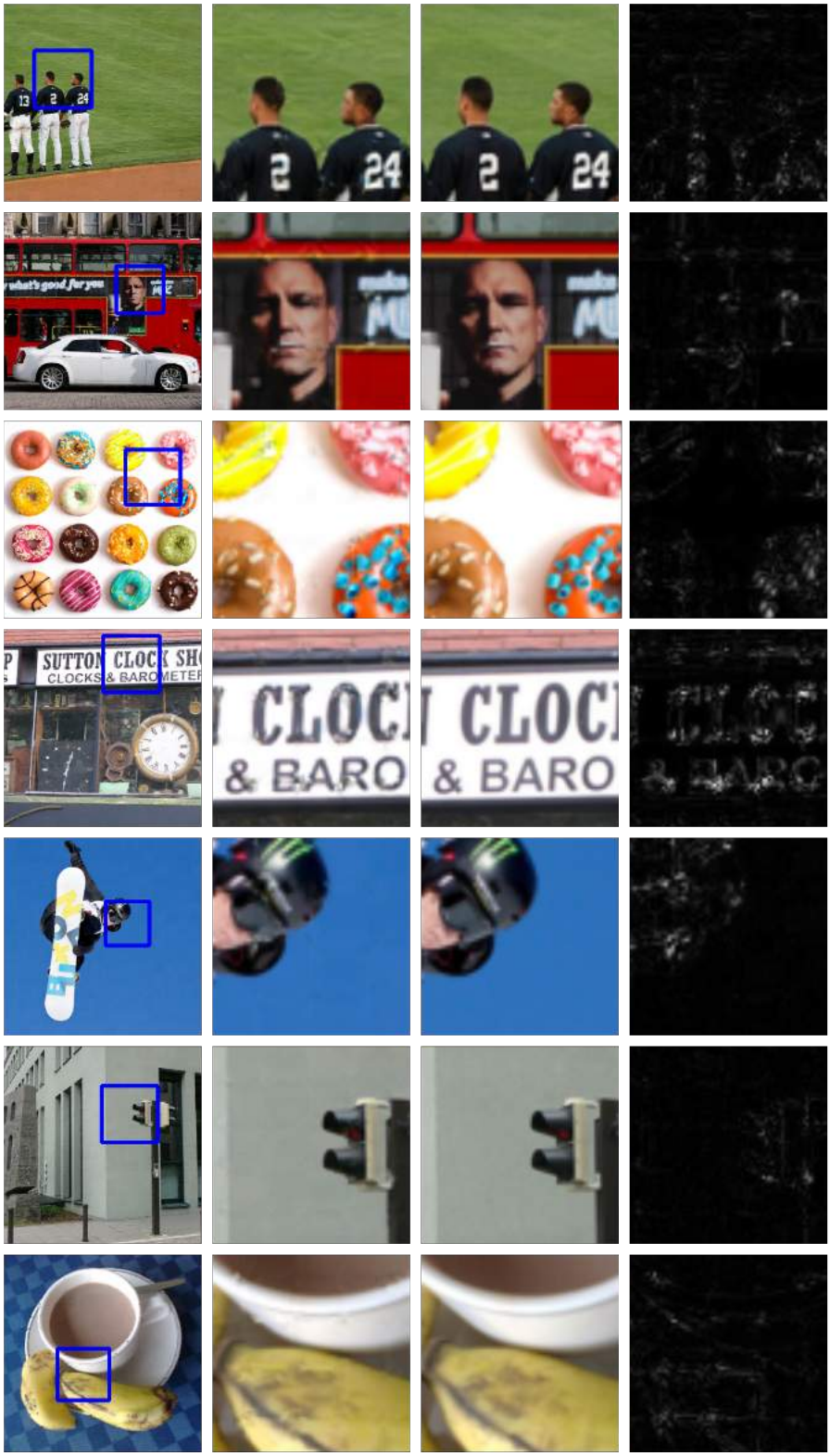
Ataki	DD (*Nasza*)			RivaGAN [ZXCIV19]
	Prosta	Rzadka	bez A	
MJPEG	0.968	**0.981**	0.746	0.965
MPEG	0.945	**0.951**	0.689	0.932

Tablica 3.5: Wyniki transparentności naszej metody i ostatnich rozwiązań, tj. *S+C* (rozdział 2), HiDDeN [ZKJFF18], DADW [LZC+20] oraz RedMark [ANK+20] w odniesieniu do PSNR. W tablicy przedstawiono wyniki obliczone zgodnie z metodologią przedstawioną w [ZKJFF18], natomiast szerszą analizę przezroczystości przedstawiono w tablicy 3.6 i tablicy 3.7. Zauważmy, że w [ANK+20] autorzy poinformowali, że osiągnięte wyniki są podobne do tych w [ZKJFF18]. Podajemy również linię bazową przezroczystości, czyli wartości PSNR policzone dla algorytmu kompresji JPEG z parametrem $q = 50$.

Kanał	DD (*Nasza*)			*Nasza*
	Prosta	Rzadka	bez A	*S+C*
PSNR(Y)	31.08	30.07	30.40	30.19
PSNR(U)	**44.37**	39.61	37.92	37.88
PSNR(V)	**45.27**	38.35	37.64	37.67
Kanał	HiDDeN	DADW	RedMark	JPEG(q=50)
PSNR(Y)	33.55	**35.70**	≈33.55	36.35
PSNR(U)	38.92	40.70	≈38.90	36.78
PSNR(V)	39.38	40.10	≈39.40	36.93



Rysunek 3.3: Porównanie jakościowe obrazów wejściowych I_c (lewa kolumna), obrazów zakodowanych I_e (środkowa kolumna) oraz ich różnice znormalizowane metodą min-max (prawa kolumna).



Rysunek 3.4: Porównanie jakościowe fragmentów obrazów I_e oraz I_c — miejsce przycięcia obrazów (pierwsza kolumna od lewej), fragment I_e (druga kolumna), fragment I_c (trzecia kolumna) oraz znormalizowane metodą min-max różnice przyciętych fragmentów (ostatnia kolumna). Widzimy, że na zakodowanych obrazach widoczne są pewne zniekształcenia. Tylko w ten sposób metoda jest w stanie osiągnąć zadowalającą odporność na kompresję.

Tablica 3.6: Wyniki transparentności naszej metody obliczone dla znaku wodnego nałożonego częściowo na obraz. Oceniamy trzy przypadki, w których 100%, 70% i 50% obrazu jest pokryte znakiem wodnym.

Współczynnik znaku wodnego do rozmiaru obrazu	PSNR			SSIM	Dokładność bitowa
	Y	U	V		
1.0	31.08	44.37	45.27	0.9129	0.944
0.7	32.10	45.21	46.10	0.9334	0.938
0.5	33.44	46.56	47.44	0.9514	0.886

Tablica 3.7: Wyniki transparentności naszej metody obliczone dla obrazów o rozmiarach równych 128×128, 256×256, 512×512 oraz 1024×1024.

Rozmiar obrazu	PSNR			SSIM	Dokładność bitowa
	Y	U	V		
128×128	31.05	44.33	45.20	0.9140	0.944
256×256	31.08	44.37	45.27	0.9129	0.944
512×512	31.81	44.85	45.81	0.9267	0.941
1024×1024	35.96	47.57	48.41	0.9400	0.936

Przeprowadzamy również szerszą analizę transparentności, w której rozważamy częściowe nałożenie znaku wodnego na obraz, a także dostarczamy wyniki przezroczystości dla obrazów o większej rozdzielczości, a tym samym mniejszej entropii. Oba warianty eksperymentów są bardziej adekwatne do rzeczywistych scenariuszy. Wyniki naszej analizy są przedstawione w tablicy 3.6 i tablicy 3.7. Możemy zaobserwować, że częściowe zastosowanie znaku wodnego może zwiększyć PSNR odpowiednio do 33.44dB, 46.56dB i 47.44dB dla odpowiednio kanałów Y, U i V, a przy tym nadal generowany jest odporny znak wodny. Drugi eksperyment pokazuje, że obrazy o większym rozmiarze (i mniejszej entropii ze względu na większą redundancję danych) cechują się wyższą transparentnością w stosunku do mniejszych, a jednocześnie dokładność bitowa nie jest zmniejszona. Aby pokazać podobieństwo strukturalne obrazów przed i po nałożeniu znaku wodnego, liczmy również SSIM.

3.4 Schemat dyskryminator-detektor

Wdrożony system do znakowania wodnego pracujący w warunkach rzeczywistych koduje jedynie niewielką część wszystkich dostępnych treści multimedialnych (np. do których mamy dostęp poprzez Internet), które musi realnie przeanalizować w celu zidentyfikowania sprawcy udostępniającego treści pirackie. Oznacza to, że w rzeczywistości musimy w pierwszej kolejności sprawdzić, czy w analizowanym obrazie znak wodny został w ogóle umieszczony. W takim scenariuszu, aby wykryć wiadomość, musimy zrealizować jedno z następujących podejść:

1. naiwne — stosujemy procedurę odczytania znaku wodnego na każdym obrazie i oznaczmy podejrzanych, gdy odczytany klucz ma co najmniej t bitów zgodnych z którymkolwiek z kluczy z bazy danych,

2. podwójne detektor-dyskryminator — najpierw wykonujemy procedurę określenia, czy dana treść jest oznaczona znakiem wodnym i następnie, w przypadku pozytywnym, przeprowadzamy procedurę odczytania klucza.

Pierwsza z nich polega na zastosowaniu wyłącznie detektora, który w tym przypadku musi cechować się bardzo wysoką skutecznością. Przykładowo, niech znak wodny składa się z 32 bitów i $t = 29$ (szacowanie dla skuteczności równej około 90%). Wtedy mając milion kluczy w bazie, wybranych losowo przy założeniu rozkładu jednostajnego, prawdopodobieństwo zdarzenia, że przynajmniej jeden klucz z bazy zawiera co najmniej 29 współdzielonych bitów jest równe $1 - (\sum_{i=0}^{28}[\binom{32}{i}0.5^i 0.5^{32-i}])^{10^6} \approx 0.72$. Tym samym prawdopodobieństwo niepowodzenia jest wysokie nawet przy stosunkowo niewielkiej bazie kluczy i dużej dokładności detektora.

Nasze drugie podejście może znacznie poprawić ogólną wydajność systemu znakowania wodnego, ale wymaga dodatkowego podsystemu do rozróżniania, czy treść jest znakowana. Zamiast używać dyskryminatora F tylko podczas treningu do zwiększenia jakości obrazu przez ocenianie, czy zakodowany obraz jest podobny do obrazu wejściowego, zmieniamy pozycję dyskryminatora w potoku treningowym i umieszczamy go „za” noiserem N. W ten sposób dostarczamy dyskryminatorowi F zniekształcony obraz podstawowy I'_c oraz zniekształcony obraz zakodowany I'_e. Obraz I'_c jest przetwarzamy przez noiser N w taki sam sposób jak obraz zakodowany I'_e, ma to na celu uniknięcie wyuczenia się niewłaściwych cech obrazów, tj. dyskryminator mógłby nauczyć się wzorców, które są efektem przetwarzania przez

nosier N, a nie enkoder E. Zaproponowana modyfikacja potoku treningowego nadal daje nam możliwość używania F do treningu adwersarialnego mającego na celu poprawę przejrzystości zakodowanych obrazów.

3.4.1 Wykorzystane metryki

Podczas testów zakładamy, że obraz I zawiera znak wodny jeśli $F(I) > t_F$, gdzie $t_F \in [0, 1]$ jest progiem akceptacji dla wyniku dyskryminatora. Następnie definiujemy klasyczne metryki opisujące 4 typy wyników: *prawdziwie dodatni* (ang. *true postive*, TP), kiedy I jest zakodowanym obrazem oraz $F(I) > t_F$; *fałszywie dodatni* (ang. *false positive*, FP), kiedy I nie zawiera zakodowanej wiadomości oraz $F(I) > t_F$; *prawdziwie ujemny* (ang. *true negative*, TN), kiedy I nie zawiera zakodowanej wiadomości oraz $F(I) \leq t_F$; *fałszywie ujemny* (ang. *false negative*, FN), kiedy I jest zakodowanym obrazem oraz $F(I) \leq t_F$.

Projektując system znakowania wodnego dążymy do maksymalizacji wykrywania naruszeń praw autorskich oraz minimalizacji prawdopodobieństwa fałszywych oskarżeń o piractwo (wynik fałszywie dodatni). Podążając za tymi warunkami, proponujemy miarę efektywności podejścia detektor-dyskryminator, który dobrze przekłada się na walidację systemu znakowania wodnego w warunkach rzeczywistych:

1. *wskaźnik poprawnych identyfikacji* (ang. *true identification rate*, TIR), definiowany jako prawdopodobieństwo odczytania właściwego klucza z prawdziwe zakodowanego obrazu:

$$\mathrm{TIR} = \Pr(\text{odczytany prawdziwy klucz} \mid \mathrm{TP}), \tag{3.11}$$

2. *wskaźnik fałszywych identyfikacji* (ang. *false identification rate*, FIR), definiowany jako prawdopodobieństwo błędnego odczytania znaku wodnego z prawdziwe zakodowanego obrazu lub fałszywie sklasyfikowanie obrazu jako zakodowany:

$$\begin{aligned}\mathrm{FIR} = {} & \Pr(\text{odczytany błędy klucz} \mid \mathrm{TP}) \Pr(I_e) \\ & + \Pr(\mathrm{FP}) \Pr(I_c).\end{aligned} \tag{3.12}$$

TIR mierzy skuteczność poprawnego wykrycia nielegalnych treści. Próba jest liczona jako sukces, gdy dyskryminator wykryje znak wodny w zakodowanym obrazie, a następnie detektor odczyta poprawnie klucz (wiadomość). Intuicyjnie moglibyśmy powiedzieć, że ta miara mierzy skuteczność wykrywania pirackich treści. FIR liczy błędy systemu interpretowane jako omyłkowe oskarżenie kogoś o piractwo. Rozważamy dwa scenariusze:

- zawodzi detektor, który odczytuje zły klucz, przy tym oskarża niewłaściwego użytkownika, a dyskryminator działa poprawnie.
- zawodzi dyskryminator, który fałszywie wykrywa znak wodny w niezakodowanym obrazie i zmusza detektor do odczytania klucza z niezakodowanego obrazu (co ostatecznie prowadzi do błędnego wytypowania użytkownika).

Łatwo zauważyć, że dla wskaźnika TIR dążymy do jego maksymalizacji, podczas gdy FIR powinien być jak najmniejszy. Praca z oboma wskaźnikami daje nam możliwość precyzyjnej walidacji systemu znakowania wodnego. Ze względu na brak wiedzy o proporcjach między obrazami zakodowanymi oraz oryginalnymi, rozważamy FIR oddzielnie dla zakodowanych obrazów I_e i obrazów wejściowych I_c. Dlatego definiujemy osobny wskaźnik, aby przetestować wskazanie błędnych kluczy dla zakodowanych obrazów definiujemy:

$$\mathrm{FIR}_{en} = \Pr(\text{odczytany błędy klucz}| \text{ TP}) \tag{3.13}$$

oraz drugi wskaźnik do sprawdzenia błędnie sklasyfikowanych obrazów wejściowych:

$$\mathrm{FIR}_{co} = \Pr(\mathrm{FP}). \tag{3.14}$$

3.4.2 Odporność podejścia detektor–dyskryminator

Zgodnie z naszą wiedzą, ostatnie prace nad znakowaniem wodnym opartym na sieciach neuronowych nie obejmowały eksperymentów w warunkach opisanych w rozdziale 3.4.1. Zdecydowana większość badań skupia się na poprawie jedynie procedury odczytywania znaku wodnego i przejrzystości zakodowanych obrazów. Nasz system przetestowaliśmy bazując na procedurach znanych z innych prac [ZKJFF18, LZC+20, ANK+20] (rozdział 3.3), ale również wykorzystaliśmy do tego celu zaproponowane nowe wskaźniki TIR oraz FIR.

Korzystając z zaproponowanej formuły testowej, pokazujemy skuteczność naszego rozwiązania opartego na parze detektor-dyskryminator i porównujemy z naiwnym systemem z pojedynczym detektorem. Do celów testowych używamy systemu z adapterem wykorzystującym przejście proste. Przyjmujemy, że rozmiar puli kluczy jest równy 10^6. Eksperymenty zostały wykonywane na 1000 obrazach. Osiągnięte wyniki zostały przedstawione w tablicy 3.8. Próg t_F jest dostosowywany tak, aby uzyskać najwyższy

Tablica 3.8: Wyniki odporności (mierzonej z wykorzystaniem wydajności bitowej) na wybrane ataki licznej przy użyciu nowej formuły testowej. Eksperymenty zostały przeprowadzane dla adaptera wykorzystującego przejście proste. Wyraźnie lepsze wyniki zaznaczono pogrubioną czcionką. Należy pamiętać, że dla naiwnego podejścia przedstawiamy wyniki wyliczone empiryczne na naszym zbiorze danych testowych, jednakże wyniki te można również oszacować na podstawie dokładności samego detektora.

Ataki	podwójne		
	TIR ↑	FIR_{en} ↓	FIR_{co} ↓
Identyczność	0.999	0.001	0.000
Przycinanie($p = 0.3$)	0.219	**0.122**	**0.000**
Cropout($p = 0.3$)	0.550	**0.193**	**0.001**
Dropout($p = 0.5$)	0.940	0.007	0.008
Rotacja($\alpha = 5°$)	0.983	0.016	**0.000**
Rozmycie($\sigma = 2$)	0.950	0.000	0.474
Rozmycie($\sigma = 4$)	0.956	0.001	0.410
Próbkowanie(4:2:0)	0.971	0.000	0.000
Rozmiar($s = 0.5, m = N$)	0.630	**0.119**	**0.602**
Rozmiar($s = 0.5, m = L$)	0.540	**0.126**	0.701
JPEG($q = 50$)	0.712	0.069	**0.000**
Ataki	naiwne		
	TIR ↑	FIR_{en} ↓	FIR_{co} ↓
Identyczność	0.999	0.001	0.005
Przycinanie($p = 0.3$)	0.219	0.594	0.707
Cropout($p = 0.3$)	0.556	0.372	0.720
Dropout($p = 0.5$)	0.951	0.004	0.006
Rotacja($\alpha = 5°$)	0.979	0.009	0.119
Rozmycie($\sigma = 2$)	0.998	0.000	**0.130**
Rozmycie($\sigma = 4$)	0.999	0.001	**0.005**
Próbkowanie(4:2:0)	0.975	0.000	0.001
Rozmiar($s = 0.5, m = N$)	0.645	0.284	0.717
Rozmiar($s = 0.5, m = L$)	0.542	0.371	0.707
JPEG($q = 50$)	0.712	0.044	0.118

TIR dla podejścia naiwnego, jednak ze względu na wysoką czułość FIR_{co} wybieramy niższy próg w przypadku, gdy wartość wskaźnika TIR zmniejszyła się co najwyżej 5%. Następnie dostosowujemy wskaźniki dla podejścia podwójnego utrzymując podobny TIR.

Nasze wyniki potwierdzają wysoką skuteczność podejścia opartego na

dwóch sieciach – dyskryminatorze i detektorze. W większości przypadków uzyskane wyniki są wyższe niż w przypadku naiwnym, a dla niektórych ataków różnica jest nawet znacząca. Podejście podwójne znacznie poprawia ogólną wydajność w przypadku ataków, w których detektor wykazuje stosunkowo niską dokładność bitową, tj. niższą niż 0.95. Wynika z tego, że podejście dyskryminator-detektor przewyższa naiwne podejście dla kilku ataków, tj. kompresji JPEG, zmiany rozmiaru, przycinania, rotacji i ataku cropout. Na przykład, mając podobne wartości dla TIR, całkowicie zredukowaliśmy FIR_{co} z około 0.7 i zmniejszyliśmy FIR_{en} od 2 do 5 razy dla przycinania i ataku cropout. Obserwujemy też, że podejście naiwne jest równie skuteczne co podejście oparte na dwóch sieciach detektor-dyskryminator, jeśli detektor osiąga wysoką odporność, tj. wydajności bitową powyżej 0.95, co można zauważyć w przypadku próbkowania 4:2:0 i ataku dropout. Ponadto obserwujemy, że przy niewydajnym dyskryminatorze i odpornym detektorze wyniki mogą być lepsze dla naiwnego podejścia, np. dla wygładzania gaussowskiego dokładność bitowa detektora jest bliska 0.99, a dyskryminatora wynosi 0.8 (na zbalansowanych danych), co przekłada się na wyższą ogólną wydajność dla tego ataku w przypadku podejścia naiwnego.

Zgodnie z naszą wiedzą, jako pierwsi rozszerzamy standardowe podejście obejmujące pojedynczy detektor o inny komponent, aby rozróżnić, czy obraz zawiera znak wodny, czy nie. Możemy jednak z łatwością oszacować, że nasze podwójne podejście można z powodzeniem zastosować do innych metod. Widzimy, że odporność metody z rozdziału 2 można znacznie poprawić dla wszystkich ataków poza wygładzaniem gaussowskim i próbkowaniem 4:2:0, dla których sam detektor osiąga skuteczność powyżej 0.98. Rozwiązania zaprezentowane w [ZKJFF18], [LZC+20] oraz [ANK+20] wskazują dokładność bitową zbliżoną do 1.0 dla ataków kadrowania i dropout. W przypadku tych dwóch ataków żadne dodatkowe rozwiązania nie są wymagane, ponieważ jesteśmy w stanie praktycznie bezbłędnie wyodrębnić wiadomość za pomocą samego detektora. Jednak wszystkie te metody w połączeniu z dyskryminatorem mogą osiągnąć znacznie wyższą wydajność dekodowania wiadomości w przypadku pozostałych ataków – JPEG, zmiana rozmiaru, wygładzanie gaussowskie, a także croput, dla których osiągana odporność jest niższa.

3.5 Podsumowanie

W pracy wprowadziliśmy nowatorskie rozwiązanie typu end-to-end do znakowania wodnego oparte na sieciach neuronowych. Znacznie poprawiliśmy odporność na niektóre ataki, np. kompresję JPEG, rotację. Metoda wy-

różnia się najwyższą ogólną dokładnością, która jest równa 0.86. Charakteryzuje się również jedną z najwyższych transparentności zakodowanych obrazów. Do enkodera dodaliśmy nowy komponent o nazwie *adapter* i opracowaliśmy nową architekturę potoku, dzięki której byliśmy w stanie później wykorzystać dyskryminator do rozróżniania między obrazami zakodowanymi i oryginalnymi. Zaproponowaliśmy metodę oceny, która znacznie bardziej oddaje rzeczywiste warunki pracy systemu znakowania wodnego. Metodę oparliśmy o trzy metryki, mianowicie TIR, FIR_{en} oraz FIR_{co}. Porównawszy naiwny system znakowania wodnego i nasze podejście detektor-dyskryminator wykorzystując do tego zaproponowane wskaźniki, potwierdziliśmy wysoką skuteczność naszego nowego podejścia. Na koniec dostosowaliśmy nasze rozwiązanie do znakowania wodnego wideo i eksperymentalnie udowodniliśmy jego wysoką niezawodność. Nasze rozwiązanie przewyższa inne najnowsze podejścia oparte na uczeniu głębokim w przypadkach odporności i złożoności, zachowując przy tym wysoką przejrzystość i pojemność.

Rozdział 4

Prywatność w kontekście rozpoznawania twarzy

4.1 Wprowadzenie

W tym rozdziale omówimy zagadnienie prywatności w kontekście biometrii na przykładzie rozpoznawania twarzy. Biometria znalazła już powszechne zastosowanie w systemach do autoryzacji użytkowników. Najbardziej rozpowszechnione systemy biometryczne są oparte na odcisku palca i cechach twarzy, a proces autoryzacji odbywa się najczęściej za pomocą dedykowanego rozwiązania w miejscu autoryzacji (ang. *on-site*) lub poprzez urządzenie mobilne [GB17]. Kolejna istotna grupa komercyjnych systemów biometrycznych jest oparta na analizie cech głosu. Tego typu systemy można znaleźć w rozwiązaniach do komunikacji telefonicznej z bankami lub w „inteligentnych" głośnikach (np. Amazon Alexa). Metody weryfikacji czy identyfikacji wykorzystujące biometrię uważa się za bezpieczne oraz wygodne w użyciu, co uczyniło je bardzo popularnymi. Obecnie ten sposób autoryzacji pozwala już na dostęp do bardzo wrażliwych danych użytkowników, takich jak konta bankowe lub prywatne wiadomości, a nawet umożliwia dostęp do sklepów internetowych i kupno produktów poprzez inteligenty głośnik (bez dodatkowych metod autoryzacji). Ponadto, ze względu na rosnące zainteresowanie badawcze metodami biometrycznymi, niektóre algorytmy są wykorzystywane poza klasycznym obszarem autoryzacji, między innymi, biometria może być wykorzystywana do skanowania tłumu w miejscach publicznych i śledzenia ludzi. Możliwość zidentyfikowania osoby w łatwy i zdalny sposób (nawet bez jej wiedzy) przyczynia się do popularności takich zastosowań jak analiza zachowania z użyciem kamer przemysłowych czy kontrolna graniczna [LGMn+16]. W wielu dziedzinach metody rozpoznawania wykorzystują więcej niż jedną biometrię, łącząc ich właściwości, a przy tym zapewniając wyższą dokładność identyfikacji. Przykłady wy-

branych multimodalnych podejść można znaleźć w [OH16, BCTPL16].

Tak szerokie zastosowanie metod biometrycznych w autoryzacji rodzi pytania o kwestie prywatności tych metod. Jest to szczególnie ważne, ze względu na fakt, że cechy biometryczne są unikalne dla każdej osoby. To powoduje ryzyko, że w momencie wycieku bazy z systemu biometrycznego dostęp do unikalnych danych wrażliwych (jakimi są dane biometryczne) mogą uzyskać osoby nieupoważnione. To już samo w sobie stanowi zagrożenie prywatności. Ten problem jest szczególnie istotny, ponieważ oprócz swojej unikalności, są to też dane zasadniczo niezbywalne (niezmienne). A co za tym idzie, naruszenie bezpieczeństwa w tego typu systemie, który zazwyczaj jest oparty na konkretnej metodzie, w praktyce oznacza konieczność jego likwidacji i zastąpienie nowym rozwiązaniem. Co więcej, istnieją metody pozwalające na odtworzenie wzorca biometrycznego (zazwyczaj wektora cech) [MCYJ19], co powoduje, że adwersarz mający dostęp do danych biometrycznych z bazy może dokonać próby ich odtworzenia, a przy tym uzyskać dostęp do innych systemów wykorzystujących autoryzację bazującą na tych samych cechach biometrycznych.

Jedna z najpopularniejszych metod identyfikacji biometrycznej wykorzystuje charakterystyczne rysy twarzy. Kilka popularnych metod rozpoznawania użytkownika na podstawie rysów twarzy zostało przedstawionych w [KWT+05, WP07].

W zwykłym przypadku cechy biometryczne są interpretowane jako pewne punkty zlokalizowane na twarzy, które są pobieranie i przechowywane w postaci wektora cech (oczywiście, dokładna reprezentacja tego wektora zależy od wykorzystanej metody). W przypadku włamania się do bazy i ujawnienia danych, tak reprezentowane cechy biometryczne mogą być wykorzystanie przez adwersarza do zrekonstruowania twarzy [GMF+10, MHC15].

Badania nad prywatnością w schematach biometrycznych przeprowadzono w [RCB01], jednak większość zaproponowanych metod ogranicza się do odpowiedniego przechowywania już uzyskanych danych lub wymaga znacznego poziomu świadomości użytkownika. Autorzy w artykule wprowadzili termin *biometrii usuwalnej* (ang. *cancelable biometrics*), który był później badany m.in. w [PRC15]. Ponadto istnieją rozwiązania polegające na ochronie profili użytkowników, w tym [EFG+09, HLP12], gdzie wektor cech jest szyfrowany i przechowywany w bezpieczny sposób. Te metody zwiększają poziom bezpieczeństwa bazując na niejawności dodatkowego klucza szyfrującego, co jest niewystarczające wobec adwersarza, który jednak uzyskał dostęp do dodatkowych danych kryptograficznych (koniecznych do odszyfrowania wektora cech biometrycznych). Alternatywnym podejściem

jest zastosowanie obliczeń wielostronnych (ang. *multiparty computation*) podczas przeprowadzania biometrycznej identyfikacji użytkownika [BCP13].

My proponujemy inne podejście, w którym zamiast identyfikowania i przechowywania stałych pozycji określonych cech twarzy, chcemy śledzić przemieszczenia takich cech w czasie, w trakcie wypowiadania przez użytkownika określonego zadania. Wtedy naszymi cechami biometrycznymi będzie ruch ust, marszczenie brwi, itp. Intuicyjnie wektor przemieszczenia wyraźnie ujawnia mniej informacji, które można wykorzystać do bezpośredniej rekonstrukcji twarzy po wycieku danych, a ponadto nie jest podatny na metody stosowane w znanych atakach na statyczne cechy twarzy. Pokazujemy również jak zmniejszyć możliwość powiązania wektorów przemieszczeń i odpowiadającym im statycznych cech twarzy za pomocą normalizacji oraz odrzucenia cech o najwyższej korelacji. Ponadto koncentrujemy się na maksymalnym uproszczeniu potoku przetwarzania i metodach przekształcania wektorów przemieszczeń. Naszą metodę można również uznać za rodzaj biometrii usuwalnej, w której wyrażone zdanie jest traktowane jako zaszyfrowany klucz służący do transformacji cech.

Mimo że zgodnie z naszą najlepszą wiedzą, nie przeprowadzono żadnych wcześniejszych badań dotyczących identyfikacji użytkownika na podstawie analizy przemieszczeń określonych punktów twarzy, to tego typu cechy zostały wykorzystane do wykrywania, śledzenia oraz wzmacniania ruchu różnych obszarów twarzy. Śledzenie punktów charakterystycznych twarzy (ang. *face landmarks*) służy przede wszystkim do identyfikacji stanu medycznego lub emocjonalnego badanej osoby, bądź też sterowania w trakcie interakcji człowiek-maszyna, np. [BGDP15]. W pracy [RPCP13] zaproponowano metodę bezpiecznego przechowywania informacji o wyrazie (ekspresji) twarzy użytkowników wykorzystującą do tego celu metody zaszyfrowania. Więcej na temat wydobywania i wzmacniania (mikro-)ekspresji twarzy można znaleźć w [MHP11, MNRD12, BDVS13]. Dodatkowo kilka metod wykrywających ataki spoofingowe swoje działanie opiera na analizie mimiki twarzy (np. [PSWL07, ZYL+12, CAM12]). Przegląd metod wykrywania ataków spoofingowych wykorzystując do tego analizę mimiki twarzy można znaleźć w [GMF14].

4.2 Schemat identyfikacji oparty na mimice twarzy

W tym rozdziale przedstawiamy pomysł na zbudowanie metody rozpoznawania twarzy, która zwiększa odporność na wybrane ataki polegające na odtwarzaniu (odbudowaniu) obrazu twarzy. Tworząc naszą metodę autory-

zacji biometrycznej działamy zgodnie z zasadą *privacy-by-design*, tzn. skupiamy się na ochronie prywatności użytkowników już w fazie projektowania systemu. Zwiększenie prywatności jest uzyskiwane poprzez wyodrębnienie specjalnego zbioru cech w sposób pozwalający na ukrycie standardowych statycznych danych biometrycznych, jakimi są punkty charakterystyczne twarzy, pozwalające na określenie rozmiarów poszczególnych części twarzy (np. nosa, ust), czy odległości miedzy nimi. W ten sposób ograniczamy możliwość rekonstrukcji twarzy z szablonów biometrycznych (reprezentacji cech), które mogłyby zostać pobrane lub skradzione z bazy danych. Zamiast wyodrębniać cechy, które bazują na rysach twarzy lub zależnościach między nimi, postanowiliśmy dokonać analizy mimiki twarzy w celu znalezienia różnic w tych cechach wśród populacji. Jako dane wejściowe do naszego systemu wykorzystujemy nagrania wideo osób wypowiadających jedno wybrane zdanie. Zebrana przez nas baza danych liczy łącznie 191 filmów zawierających nagrania 34 osób.

4.2.1 Zbiór danych

W celu przeprowadzenia dowodu poprawności (ang. *proof-of-concept*) naszego podejścia zebraliśmy bazę 191 nagrań wideo od 34 wolontariuszy. Wszystkie nagrywane osoby miały białą karnację skóry, a ich wiek wynosił między 20 a 50 lat. W zebranym zbiorze danych 21 osób było płci żeńskiej. Dla każdej z osób zostało nagrane od 5 do 7 próbek w postaci filmów, na których osoba wypowiada to samo zdanie składające się z 8 słów. Sposób wypowiadanej frazy nie został narzucony, za to każda z osób miała wypowiedzieć zdanie w sposób jak najbardziej dla niej naturalny. Uczestnicy badania zostali jedynie poinstruowani, żeby w trakcie wypowiadania frazy kierowali wzrok wprost w kierunku obiektywu kamery, a samo nagranie odbywało się na stojąco. Do momentu zakończenia nagrywania osoby nie były świadome w jakim dokładnie celu zostaną wykorzystane nagrania. Co więcej, nie wprowadziliśmy żadnych ograniczeń dotyczących fryzury, makijażu, posiadania okularów oraz biżuterii. Filmy zostały nagrane w środowisku nielaboratoryjnym, gdzie również nie określiliśmy odgórnie sztywnej odległości między stojącą osobą a obiektywem aparatu, ani nie podjęliśmy kroków w celu redukcji hałasu z otoczenia.

Wszystkie filmy zostały nagrane w rozdzielczości 1280×720 pikseli i przy próbkowaniu 24 fps oraz zapisane w formacie AVI bazującym na algorytmie kompresji MPEG-4 część 2 (MPEG-4 ASP). Wybór powszechnego formatu kodowania oraz relatywnie niskich parametrów wideo pozawala na wyko-

(a) (b)

(c) (d)

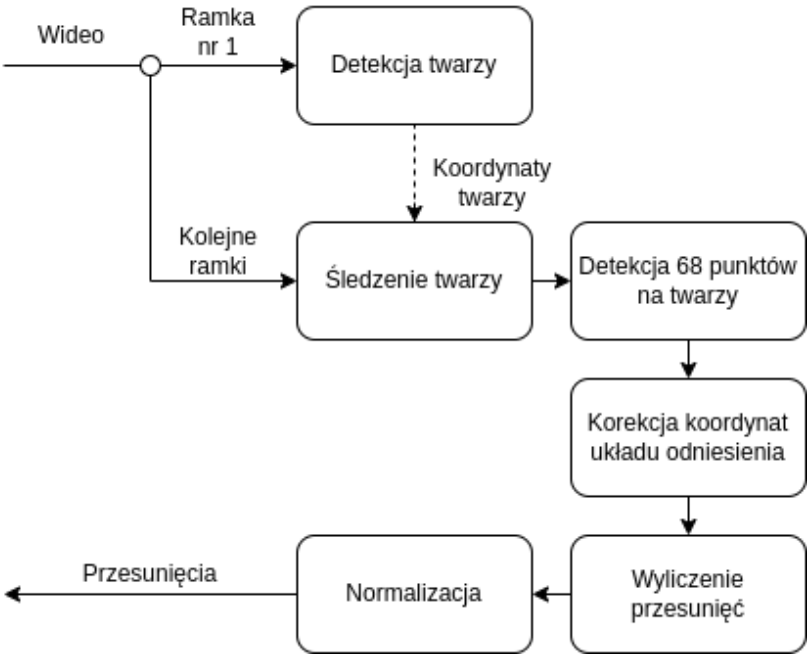
Rysunek 4.1: Przykładowe klatki wycięte z jednego z nagranych filmów. Nagrania zostały przygotowane w sposób oddający warunki rzeczywiste – cienie wokół ust oraz oczu nie zostały zredukowane lub usunięte, a pomieszczenie przeznaczone do nagrywania nie zawierało żadnych dodatkowych punktów oświetlenia, poza lampą sufitową[‡].

rzystanie niskokosztowych kamer, a przy tym nie zmniejsza znacząco wydajności naszej metody. Do pobierania próbek wideo wykorzystano kamerę Sony Exmor RS IMX214 z przetwornikiem obrazu 13M-Pixel CMOS. Taki wybór został dokonany na podstawie ceny i popularności wspomnianego urządzenia, które często było montowane w telefonach komórkowych klasy budżetowej. Wynikowe klatki wyodrębnione z jednego z nagranych filmów wideo znajdują się na rysunku 4.1.

4.2.2 Charakterystyka cech

Nasza metoda opiera się na śledzeniu przemieszczeń wybranych punktów charakterystycznych zlokalizowanych na ludzkiej twarzy. Ponieważ naszym celem jest dostarczenie tensora cech, który ma ograniczać możliwości rekonstrukcji obrazu twarzy, unikamy wyodrębniania jakichkolwiek dodatkowych cech, które są powszechnie stosowane w innych systemach rozpoznawania twarzy, np. odległość między dowolnymi dwoma punktami. W naszym po-

[‡]Autor rozprawy dziękuje za użyczenie wizerunku swojej Żonie.



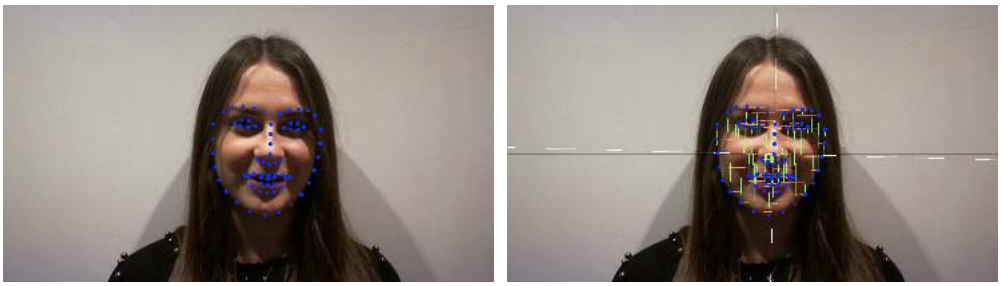
Rysunek 4.2: Schemat ekstraktacji cech biometrycznych mimiki twarzy.

dejściu przechowywanie typowych cech twarzy w bazie danych zastępujemy zbieraniem informacji jedynie o przesunięciu punktów charakterystycznych między dwiema kolejnymi ramkami. Ten sposób reprezentacji cech twarzy nie ujawnia bezpośrednio cennych danych o rysach i kształcie twarzy koniecznych do jej rekonstrukcji, nawet w przypadku wycieku bazy danych. Nasz sposób reprezentacji cech biometrycznych nie zawiera tych informacji, a jedynie informacje o mimice twarzy, tzn. dane o tym, jak zmienia się względne położenie punktów podczas wymawiania frazy, zamiast ich bezpośrednich współrzędnych (względem zadanego układu, np. z punktem początkowym na czubku nosa).

Przypomnijmy, że w tradycyjnych systemach biometrycznych opartych na rysach i kształcie twarzy adwersarz może wykorzystać ujawnione w trakcie wycieku informacje i podjąć próbę odtworzenia wizerunku osoby, co samo w sobie jest zagrożeniem prywatności (np. umożliwia identyfikację biometryczną poza systemem). Ryzyko takiego ataku zostało opisane między innymi w [NNKJ10]. W pracy [MCYJ19] autorzy wykazali, że zaawansowane głębokie sieci neuronowe do rozpoznawania twarzy generujące abstrakcyjny wektor cech (trudno interpretowalny dla człowieka) również nie są odporne na tego typu ataki. Autorzy dokonali ataku wykorzystując wiedzę, że algorytm do identyfikacji opiera się na konwolucyjnych sieciach neuronowych i wykorzystali generatywne sieci przeciwstawne (ang. *Generative Adversa-*

(a) Ramka wejściowa

(b) Detekcja pozycji twarzy na obrazie



(c) Detekcja 68 punktów charakterystycznych twarzy

(d) Estymacja pochylenia twarzy i przekształcenie punktów charakterystycznych

Rysunek 4.3: Wizualizacja kroków algorytmu ekstrakcji cech biometrycznych twarzy z jednej ramki filmu.

rial Network, GAN) do zrekonstruowania oryginalnego obrazu twarzy, co udało się im z wysokim wskaźnikiem powodzenia. Przechowywanie jedynie przemieszczenia punktów charakterystycznych, bez informacji o ich (pierwotnym) położeniu, wprowadza dodatkową warstwę ochrony prywatności. Dodatkowo jednym z etapów naszej metody jest przeskalowanie (normalizacja) względem jednej z cech biometrycznych twarzy, tj. długości nosa, co powoduje zbliżone wartości przeskalowanych cech nawet dla różnych rzeczywistych rozmiarów twarzy.

4.2.3 Ekstrakcja cech

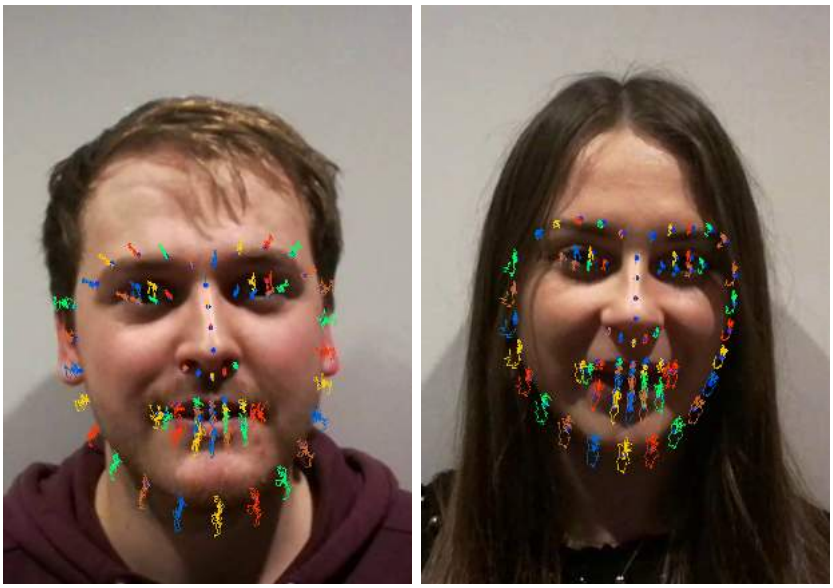
Proces wyodrębniania cech został podzielony na kilka etapów. Najpierw nasza metoda wykrywa pozycję twarzy nagrywanej osoby na pierwszej klatce danego filmu. Pozycja twarzy jest reprezentowana jako prostokąt, który obejmuje w większości twarz. W pracy wykorzystujemy prostą i szybką metodę detekcji twarzy zaproponowaną w [VJ01]. W kolejnych klatkach nie wykrywany ponownie położenia twarzy, a zamiast tego wykorzystujemy algorytm do śledzenia obiektów [HCMB12], co pozwala na jeszcze lepszą

optymalizację metody pod kątem złożoności obliczeniowej. Ten algorytm również reprezentuje położenie twarzy jako prostokąt o tych samych wymiarach co znaleziony w pierwszej klatce filmu, w efekcie czego z całego filmu wyodrębniamy tę część obrazu, na której znajduje się tylko twarz.

W kolejnym kroku dla każdej klatki wykrywamy punkty charakterystyczne twarzy za pomocą szybkiego algorytmu opartego na drzewach regresyjnych [KS14], który zwraca łącznie 68 punktów wskazujących dokładne pozycje istotnych części twarzy (np. oczu, nosa). Bardziej szczegółowy opis punktów charakterystycznych rozważamy w rozdziale 4.2.4. Następnie określamy układ współrzędnych dla każdej klatki w taki sposób, aby oś pionowa była równoległa do nosa, a środek układu współrzędnych znajdował się w jego czubku. Wcześniej wykryte punkty charakterystyczne są odpowiednio obracane w celu dopasowania do nowego układu współrzędnych. Następnie obliczamy przemieszczenia między dwiema kolejnymi klatkami dla każdego punktu charakterystycznego. Wartości przemieszczeń normalizujemy dzieląc przez średnią długość nosa (liczona na podstawie punktów definiujących początek i koniec nosa) dwóch kolejnych ramek. Proces normalizacji został szczegółowo opisany w rozdziale 4.2.6.

Schemat blokowy procesu wyodrębniania cech został przedstawiony na rysunku 4.2. Główne etapy naszej metody wyodrębniania cech biometrycznych z jednej z ramek zostały przedstawione na rysunku 4.3, natomiast rysunek 4.4 obrazuje przykłady trajektorii (śladów) przemieszczeń dla dwóch wybranych osób. Zwróćmy uwagę na widoczne różnice w przedstawionych trajektoriach przemieszczeń, m.in. okrągły *versus* podłużny pionowy kształt trajektorii brwi, wskazujący na różnorodność cech mimiki twarzy w populacji.

W tablicy 4.1 zaprezentowaliśmy wartości minimalne i maksymalne, 5. percentyl, 95. percentyl, średnią oraz odchylenie standardowe wartości przemieszczeń wybranych punktów w kierunku pionowym i poziomym. Dodatkowo przedstawiono średnią i odchylenie standardowe bezwzględnych wartości przemieszczeń. Wyliczone statystyki wskazują na różnice między poszczególnymi próbkami. Wartości średnich i odchyleń standardowych z kolei mogą sugerować, że zakresy wartości przemieszczeń są głównie małe, a maksima i minima są dalekie od wartości średnich, które wskazują na istnienie danych odstających (ang. *outliers*). Tego typu reprezentacja cech pozwala na rozróżnienie osób w populacji i przeprowadzenie identyfikacji.



(a) (b)

Rysunek 4.4: Przykłady śladów przemieszczeń punktów charakterystycznych dwóch osób. Trajektorię wyznaczają wektory przemieszczeń w kolejnych klatkach. Rożne kolory zostały wykorzystane w celu lepszej przejrzystości wizualizacji.

4.2.4 Reprezentacja twarzy oparta na punktach charakterystycznych

W naszej metodzie analizujemy standardowy zestaw punktów charakterystycznych twarzy, które można wykryć za pomocą dedykowanego do tego celu rozwiązania. W pracy wykorzystaliśmy algorytm oparty na drzewach regresyjnych uczonych metodą wzmocnienia gradientu (ang. *gradient boosting*) [KS14], która została wytrenowana na zbiorze punktów charakterystycznych twarzy iBUG 300-W [SAT+16]. Takie podejście pozwala na wskazanie 68 punktów definiujących miejsca szczególne na twarzy, takich jak nos, oczy, usta. Szkic rozmieszczenia punktów charakterystycznych z odniesieniami do ich numerów identyfikacyjnych (indeksów) przedstawiono na rysunku 4.8a.

Punkty charakterystyczne o indeksach od 1 do 17 definiują kontur twarzy. Punkty 18–22 i 23–27 lokalizują odpowiednio lewą i prawą brew. Następnie punkty od 28 do 31 wyznaczają pionowy przebieg nosa, gdzie 31 punkt jest jego czubkiem, który przyjmujemy jako środek układu odniesienia dla algorytmu. Zatem, te punkty są szczególnie ważne ze względu na procedurę wyrównania układu współrzędnych (patrz rozdział 4.2.3). Punkty charakterystyczne oznaczone jako 32–36 odnoszą się do dolnej części nosa

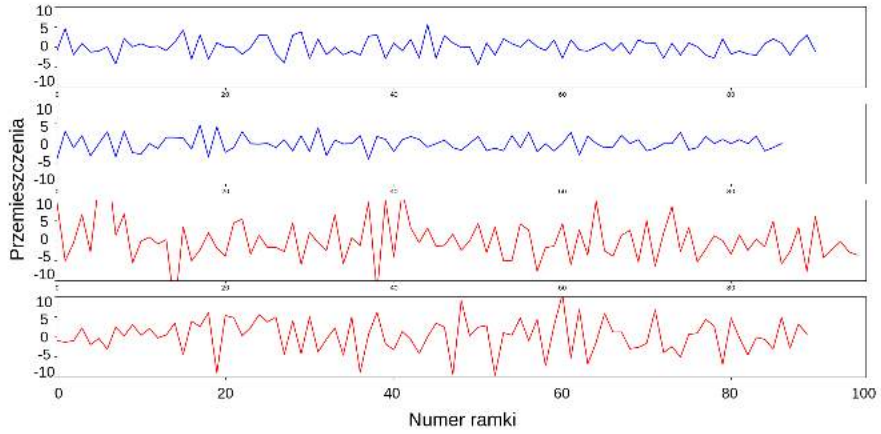
Punkt	Orientacja	Min.	5. Perc.	95. Perc.	Maks.
8	pozioma	-13.953	-0.890	0.890	17.307
8	pionowa	-4.961	-0.749	0.758	5.846
23	pozioma	-1.430	-0.375	0.371	1.806
23	pionowa	-7.769	-0.409	0.405	6.373
62	pozioma	-6.892	-0.470	0.471	8.740
62	pionowa	-3.119	-0.434	0.455	4.100
Punkt	Orientacja	μ	σ	Bezw. μ	Bezw. σ
8	pozioma	0.002	0.422	0.635	0.475
8	pionowa	0.000	0.343	0.498	0.361
23	pozioma	0.000	0.175	0.233	0.154
23	pionowa	-0.001	0.192	0.276	0.199
62	pozioma	0.001	0.224	0.328	0.239
62	pionowa	-0.000	0.211	0.310	0.226

Tablica 4.1: Wybrane parametry statystyczne – minimum, 5. percentyl, 95. percentyl, maksimum, średnia (μ), odchylenie standardowe (σ), średnia dla wartości bezwzględnych i odchylenie standardowe dla wartości bezwzględnych, wyliczone dla znormalizowanych przemieszczeń wybranych punktów charakterystycznych (8, 23 i 62) w poziomie i pionie. Parametry ilustrują istotne różnice w zakresach przemieszczeń. Zauważmy, że zestaw punktów dla których osiągnięto najlepszy wynik identyfikacji obejmował wszystkie przedstawione w tablicy punkty (patrz rozdział 4.4).

i określają jego poziomy kształt. Punkty od 37 do 42 i od 43 do 48 wyznaczają pozycje odpowiednio lewego i prawego oka. Największy podzbiór punktów defiluje położenie ust. Punkty charakterystyczne od 49 do 60 wyznaczają zewnętrzny kontur ust, a punkty od 61 do 68 określają położenie wewnętrznego konturu ust.

4.2.5 Reprezentacja cech biometrycznych

Cechy przemieszczeń twarzy są reprezentowane jako trójwymiarowy tensor $D \in \mathbb{R}^{N \times F \times 2}$, gdzie N – liczba cech (punktów charakterystycznych), F – liczba ramek w filmie pomniejszona o 1 (metoda liczy przesunięcia między ramkami). Zwróćmy uwagę, że N jest stałe, za to F jest zależne od długości danego filmu, która może być różna. Ostatni wymiar zawsze wynosi 2 i jest konieczny do zapisania przemieszczenia punktu osobno w pionie i poziomie. W pracy przyjmujemy, że D_{nf1} to wartości przesunięcia w poziomie dla n-tego punktu pomiędzy f-tą oraz $(f+1)$-tą ramką, a D_{nf2} to z kolei przesunięcie w pionie. Macierz $D_n \in \mathbb{R}^{F \times 2}$ definiuje pełną macierz prze-

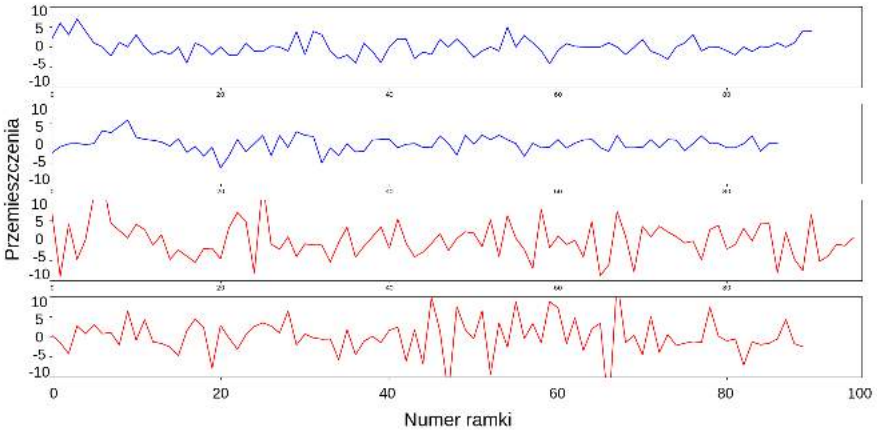


Rysunek 4.5: Wykresy przemieszczeń w poziomie dla 49 punktu charakterystycznego definiującego kącik ust dla czterech próbek – po dwie dla dwóch różnych osób (odpowiednio niebieski i czerwony).

mieszczeń dla n-tego punktu po wszystkich klatkach. Na rysunku 4.5 oraz rysunku 4.6 przedstawiamy wykresy przemieszczeń przykładowego punktu o indeksie 49 – kącik ust – w kierunku poziomym i pionowym odczytach z czterech różnych nagrań z bazy danych dla dwóch rożnych osób (po dwie próbki na osobę). Łatwo zauważyć pewne podobieństwa w przebiegu funkcji przesunięć na wykresach dla próbek należących do tych samych osób. Przykładowo, na wykresach cech poziomych linie w kolorze niebieskim zawierają dwa charakterystyczne piki między 10 a 20 ramką. Dwa tego typu piki nie są widoczne na wykresach z liniami czerwonymi. Dodatkowo zauważalna jest ogólna tendencja mniejszych różnic wartości między lokalnymi minimami i maksymami na wykresach z liniami koloru niebieskiego, które kontrastują z dużą amplitudą linii o kolorze czerwonym. Następnie na wykresach cech pionowych obie niebieskie linie wypłaszczają się wokół 40 i 60 klatki, podczas gdy line czerwone nie wykazują takiej tendencji, za to na liniach czerwonych widoczne są z kolei dynamiczne zmiany między 20 a 30 ramką oraz 50 a 60 ramką, których nie ma w przypadku pierwszej osoby (linie koloru niebieskiego).

4.2.6 Normalizacja cech

W celu zmniejszenia wpływu wynikającego z różnych odległości między nagrywaną osobą a kamerą, ostatnim etapem w naszej metodzie jest normalizacja tensora przemieszczeń. Zastosowanie normalizacji, oprócz zwięk-



Rysunek 4.6: Wykresy przemieszczeń w pionie dla 49 punktu charakterystycznego definiującego kącik ust dla czterech próbek – po dwie dla dwóch różnych osób (odpowiednio niebieski i czerwony).

szenia efektywności naszego rozwiązania, wpływa pozytywnie na kwestie prywatności. Znormalizowane wartości przesunięć ograniczają odtworzenie kształtu twarzy ze względu na ukrycie nominalnych wartości przemieszczeń i zachowanie w bazie tylko znormalizowanego tensora. To podejście dodatkowo zmniejsza możliwości określenia przez adwersarza rzeczywistego kształtu twarzy (wyrażonego w wartościach nominalnych) ze względu na brak dostępu do rzeczywistych wartości przemieszczeń (np. większe przesunięcia powieki lub brwi mogą być efektem większych oczodołów).

Dla każdego filmu wprowadzamy dodatkowy parametr, który nazywamy wektorem normalizacji i oznaczamy przez $\eta \in \mathbb{R}^{F+1}$, gdzie $F+1$ to liczba ramek w filmie. Przebadaliśmy wiele podejść do normalizacji, takich jak wykorzystanie długości całej twarzy, którą mierzyliśmy jako odległość między 1 i 17 punktem (patrz rysunek 4.8a), pola lub obwodu figury określonej przez zbiór punktów na twarzy, np. pole trójkąta wyznaczonego przez punkty 37, 46 i 52. Eksperymenty pozwoliły na wytypowanie w sposób empiryczny rozwiązania, które dało najlepsze wyniki. Ostatecznie wykazaliśmy, że odległość miedzy punktami, która w przybliżeniu określa długość nosa, jest najefektywniejszym wyborem dla wektora normalizacji η. Parametr η_f jest liczony dla f-tej ramki jako odległość między dwoma puntami – czubkiem nosa (punkt o indeksie 34) i początkiem nosa (punkt o indeksie 28). Ta odległość jest określana dla każdej ramki niezależnie, stąd η jest wektorem długości $F+1$.

W ostatnim etapie normalizacji każdą wartość przemieszczenia dzielimy przez średnią wartość dwóch elementów wektora normalizacji odnoszących się do dwóch kolejnych klatek, pomiędzy którymi obliczono przemieszczenie:

$$D_{nfi} := \frac{D_{nfi}}{\frac{1}{2}(\eta_f + \eta_{f+1})}, \tag{4.1}$$

gdzie η_f to f-ta wartość normalizowanego wektora.

4.3 Metoda porównania próbek

4.3.1 Porównanie przemieszczeń punktów charakterystycznych

Mając obliczone cechy biometryczne mimiki twarzy, nasza metoda dokonuje porównania dwóch znormalizowanych wektorów przesunięć D oraz E w celu określenia, czy próbki pochodzą od tej samej osoby. W tym celu zdecydowaliśmy się wykorzystać algorytm Dynamic Time Warping (DTW) [KL83, BK15], który porównuje sekwencje dwóch ciągów i zwraca wartość określającą ich podobieństwo. Istotną zaletą tego algorytmu jest jego zdolność do porównania dwóch ciągów o różnej długości, co jest szczególnie ważne w przypadku naszego rozwiązania, w którym bazujemy na filmach, które nie mają ustalonej odgórnie długości, ze względu oczekiwaną naturalność i swobodę przy wypowiadaniu zdania przez nagrywane osoby. Tempo mówienia jest inne dla każdego użytkowania oraz może różnić się zależnie od okoliczności, w których fraza jest wypowiadana (np. zdanie będzie szybciej wypowiadane, gdy użytkownik się spieszy lub ma je utrwalone w pamięci). Brak standaryzacji długości filmu jest zamierzony, gdyż zależy nam również na niezniekształcaniu naturalnych ruchów mimicznych, zwłaszcza tych, które trwają przez kilka czy kilkanaście klatek. Interpolacja w dziedzinie czasowej powodowałaby „rozciąganie" tych ruchów lub nienaturalne „przeskoki" klatek, co dodatkowo utrudniałoby pracę naszej metodzie identyfikacji.

Procedura porównawcza na wejściu przyjmuje reprezentację dwóch próbek biometrycznych D oraz E, które zawierają dane o przemieszczeniach punktów charakterystycznych. W pierwszej kolejności spłaszczamy tensory wejściowe, w taki sposób, że otrzymujemy $D \in \mathbb{R}^{2N \times F}$ i $E \in \mathbb{R}^{2N \times F'}$, dzięki czemu przemieszczenia w pionie oraz poziome traktujemy jako równoważne i wzajemnie niezwiązane. Następnie wykorzystujemy szybką aproksymację algorytmu DTW [SC07] działającą w czasie liniowym do policzenia podobieństwa s_n dla dwóch danych ciągów przemieszczeń niezależnie dla każ-

dej n-tej cechy, tj.

$$s_n \;=\; DTW(D_n, E_n) \;\in\; \mathbb{R}. \tag{4.2}$$

Algorytm DTW liczy podobieństwo dla każdego z 136 punktów charakterystycznych, stąd finalny wektor podobieństwa między dwoma próbkami to $s = [s_1, \ldots, s_{136}]^T \in \mathbb{R}^{136}$.

4.3.2 Funkcja decyzyjna

Mając wektor podobieństwa s musimy stwierdzić, czy próbki dla których został on policzony pochodzą od tej samej osoby. W tym celu wykorzystujemy algorytm uczenia maszynowego, który przyjmując na wejściu 136 cech zwróci prawdopodobieństwo tej przynależności. Manualne określenie progów akceptacji dla wszystkich cech i opracowanie algorytmu decyzyjnego jest zadaniem trudnym i czasochłonnym, ze względu na dużą ilość danych do przeanalizowania (już przy bazie kilkudziesięciu użytkowników otrzymujemy kilka tysięcy wektorów wzajemnych podobieństwa). Wydaje się, że właściwym podejściem jest wykorzystanie rozwiązań zaliczanych do uczenia maszynowego, które opierając się na zbiorze danych treningowych, będą w stanie znaleźć funkcję decyzyjną o stosunkowo wysokiej skuteczności.

Zdecydowaliśmy się na wykorzystanie zmodyfikowanego algorytmu maszyny wektorów nośnych (ang. *Support Vector Machine*, SVM), który estymuje prawdopodobieństwo przynależność do danej klasy [WLW04]. W naszym przypadku rozwiązujemy problem klasyfikacji binarnej, ale zamiast zwracać bezpośrednio etykietę ze zbioru $\{-1, 1\}$, liczymy prawdopodobieństwo przynależności do danej klasy. Stąd naszą funkcję decyzyjną h definiujemy jako $h : \mathbb{R}^{136} \to [0, 1]$. Funkcja decyzyjna przyjmuje wektor podobieństw, czyli $h(s)$, oraz zwraca podobieństwo zdarzenia, że dwie próbki dla których został policzony wektor s należą do tej samej osoby.

4.3.3 Trening algorytmu SVM

Postanowiliśmy przetestować dwa warianty naszej metody – wykorzystując trzy lub cztery próbki dla każdej osoby w procesie rejestracji w systemie biometrycznym. Zbiór treningowy składał się z par – wektor podobieństwa s oraz etykieta ze zbioru $\{0, 1\}$. Etykietę równą 1 przypisaliśmy wektorom podobieństwa próbek s należących do tej samej osoby, co miało odzwierciedlać 100% prawdopodobieństwa przynależności próbek do jednej osoby. Z kolei etykietę równą 0 przyporządkowaliśmy każdemu z wektorów podobieństw

próbek s pochodzących od różnych osób, co dawało informację podczas treningu, że dla tych wektorów prawdopodobieństwo przynależności próbek do jednej osoby wynosi 0%. Wektory s są liczone dla dwóch próbek znajdujących się w bazie danych, stąd w symulacji procesu rejestracji bierzemy 102 nagrania (po 3 filmy dla każdej z 34 osób) lub 136 nagrań (po 4 filmy), co pozwala nam na stworzenie zbioru treningowego liczącego odpowiednio $\binom{102}{2} = 5151$ oraz $\binom{136}{2} = 9180$ rekordów, co jest wystarczającą liczbą do efektywnego treningu algorytmu SVM [NK19].

4.3.4 Proces identyfikacji

Aby zidentyfikować osobę na podstawie cech biometrycznych mimiki twarzy w nowym nagraniu wideo, nasza metoda wymaga wyodrębnienia cech z nagrania wideo i policzenia wartości podobieństw $h(s)$ między każdą próbką zarejestrowaną w bazie danych a wyodrębnionymi danymi biometrycznymi. Następnie korzystamy z algorytmu k najbliższych sąsiadów, który mając wektor prawdopodobieństw $h(s^{(i)})$, gdzie $s^{(i)}$ jest wektorem podobieństwa między próbką wejściową oraz i-tą próbka zapisaną w bazie, wybiera k par próbek, które cechują się najwyższym prawdopodobieństwem $h(s^{(i)})$. Bazując na danym podzbiorze k-elementowym dokonujemy identyfikacji osoby poprzez wybór tej, która występuje w nim najwięcej razy. W przypadku, gdy w podzbiorze mamy tyle samo elementów odnoszących się do przynajmniej dwóch użytkowników z bazy i jest to zarazem największa liczba próbek, wtedy identyfikujemy próbkę wejściową jako należącą do osoby, dla której otrzymujemy najwyższe prawdopodobieństwo $h(s^{(i)})$. W naszych eksperymentach przyjmujemy, że k jest równe 3.

4.4 Eksperymenty

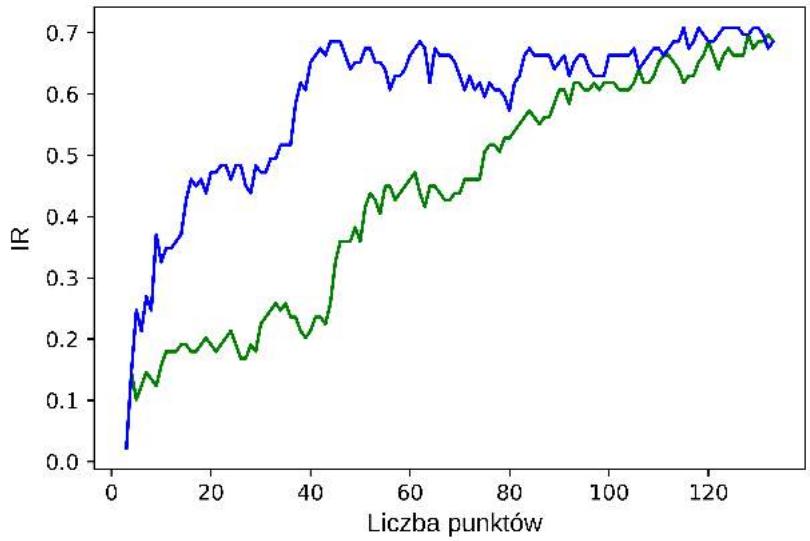
Aby zweryfikować wiarygodność schematu identyfikacji przeprowadziliśmy dowodu poprawności, tzw. *proof-of-concept*. Pobraliśmy 3 lub 4 próbki (zależnie od badanego wariantu) nagrane jako pierwsze dla każdej osoby i zasymulowaliśmy proces rejestracji. Pozostałe próbki wykorzystaliśmy w testach autoryzacji. W tym przypadku chronologiczny dobór próbek jest celowy i ma spowodować jeszcze lepsze oddanie warunków rzeczywistych, w których użytkownik najpierw rejestruje próbki w bazie danych, a później przeprowadza autoryzację już w trakcie funkcjonowania systemu.

Pozostałe próbki, nieuwzględnione w procesie rejestracji, zostały wykorzystane do ewaluacji procesu autoryzacji. Takie podejście pozwoliło na

zbudowanie zbiorów testowych zawierających 9078 i 7480 rekordów odpowiednio dla 3 i 4 próbek użytych przy rejestracji użytkownika. Jeden rekord w zbiorze testowym to para dwóch próbek – jednej znajdującej się w bazie, a drugiej spoza bazy tworzonej w fazie rejestracji. Proces testowania przebiegał zgodnie z krokami opisanymi w rozdziałach 4.3.1, 4.3.2 oraz 4.3.4. W pracy zdecydowaliśmy się na wykorzystanie standardowego wskaźnika biometrycznego do określenia skuteczności identyfikacji – współczynnik identyfikacji (ang *Identification Rate*, IR), który jest równy liczbie udanych prób identyfikacji w stosunku do liczby wszystkich prób.

W trakcie przeprowadzania eksperymentów zaobserwowaliśmy, że część cech negatywnie wpływa na wyniki skuteczności naszej metody. Co oznaczało, że część cech nie dawało pozytywnej informacji w kontekście uczenia się algorytmu, a nawet wprowadzało błędną informację lub powodowało nadmierne dopasowanie modelu (ang. *overfitting*). Co więcej, niektóre punkty charakterystyczne zostały odczytane na poszczególnych klatkach z dużymi błędami ze względu na zmienne elementy znajdujące się na nagraniach, takie jak okulary czy fryzura. W efekcie algorytm do identyfikacji również osiągał mniejszą skuteczność. Ponadto zauważyliśmy, że pierwsze próbki mogły być mało reprezentatywne i nie odwzorowywały prawdziwych przemieszczeń, ponieważ niektóre osoby potrzebowały chwili na przystosowanie się do nietypowej sytuacji w jakiej się znaleźli, a ich mimika twarzy stała się spójna dopiero w późniejszych nagraniach. Stąd oprócz uruchomienia procesu treningowego z uwzględnieniem wszystkich 136 wyodrębnionych cech, przeprowadziliśmy również dodatkowe eksperymenty, w których do treningu wykorzystaliśmy wszystkie cechy z wyjątkiem jednej. Pozwoliło nam to zbadać, które punkty charakterystyczne wpływają na zmniejszenie skuteczności naszej metody. Następnie posortowaliśmy cechy malejąco według ich wpływu i iteracyjnie trenowaliśmy nasz model, usuwając kolejne elementy w ustalonej kolejności.

Ostatecznie uzyskaliśmy najlepsze wyniki dla modelu przyjmującego 115 cech, dla którego osiągnęliśmy IR równy 70.78% przy 3 próbkach zarejestrowanych w bazie danych. Interesujące wyniki zostały również uzyskane dla podzbioru 44 cech, dla których otrzymaliśmy IR wynoszący 68.54%. W tym przypadku skuteczność naszego systemu zmniejszyła się zaledwie o około 2 punkty procentowe, podczas gdy zredukowaliśmy liczbę wykorzystywanych cech o około 61%. Ten sam eksperyment przeprowadziliśmy dla scenariusza z 4 próbami wykorzystanymi podczas rejestracji i uzyskaliśmy IR równy 69.08%, co pozwoliło stwierdzić, że zwiększenie liczby próbek w procesie rejestracji nie poprawia wyników identyfikacji. Wyniki eksperymentu

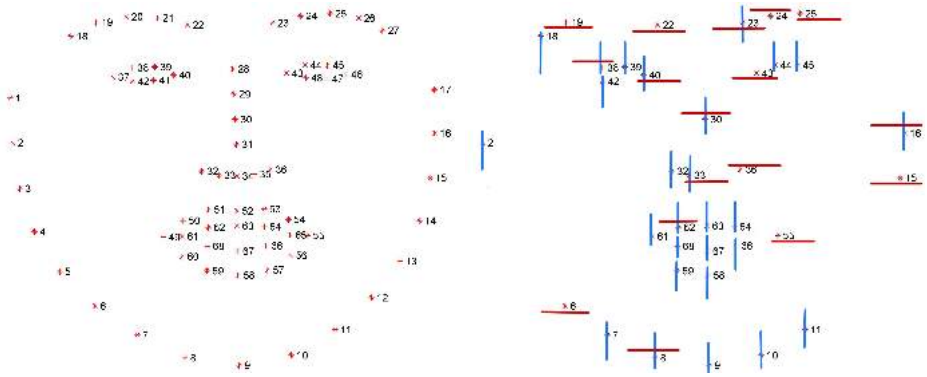


Rysunek 4.7: Wykres współczynnika identyfikacji względem liczby cech. Linia zielona wskazuje trend IR, gdy usuwane są najlepiej skorelowane cechy (patrz rozdział 4.5.1). Linia zielona wskazuje trend IR, gdy usuwane są cechy mało informatywne oraz dezinformacyjne (patrz rozdział 4.4).

dla 3 próbek wpisanych do bazy danych przedstawiono na rysunku 4.7, a na rysunku 4.8b został przedstawiony szkic z listą 44 punktów charakterystycznych (z uwzględnieniem kierunku pionowego lub poziomego) dających wynik IR równy 68.54%. Zauważmy, że nasze wyniki są znacznie wyższe niż w przypadku naiwnej strategii losowej, która dla naszego zbioru danych dawałaby IR wynoszące 2.94%.

4.5 Rozszerzona dyskusja nt. bezpieczeństwa

W pracy wykazaliśmy, że zaprezentowane przez nas nowe podejście osiąga rozsądny wynik IR (porównywalny z behawioralnymi metodami biometrycznymi). Jednak oczekujemy, że nasz protokół zapewni dodatkowo inne, dalekosiężne właściwości bezpieczeństwa. Łatwo zauważyć, że dzięki nagraniom twarzy en face i konieczności wypowiedzenia frazy, nasz schemat jest odporny na niektóre szersze klasy ataków, np. większość naiwnych ataków spoofingowych. Głównym celem przy projektowaniu naszej metody była ochrona użytkownika przed rekonstrukcją twarzy przez adwersarza, który uzyskał dostęp do bazy danych próbek biometrycznych. Innymi słowy, dążymy do „ukrycia twarzy” przed adwersarzem. W celu określenia bardziej



(a) (b)

Rysunek 4.8: Szkic rozmieszczenia punktów charakterystycznych na twarzy użytych do śledzenia przemieszczeń przed (4.8a) i po (4.8b) wyselekcjonowaniu zbioru 44 cech. Szkic 4.8b przedstawia podzbiór 44 cech, dla których osiągnięta skuteczność IR jest bliska najlepszemu wynikowi. Linie czerwone odnoszą się do przesunięć punktów w poziomie, a linie niebieskie do przesunięć punktów w pionie. Szkic 4.8a na podstawie [KS14].

formalnie poziomu bezpieczeństwa naszej metody względem ataku rekonstrukcyjnego należy postawić sobie dwa pytania:

1. Czy możliwe jest zrekonstruowanie ważnych cech twarzy przy użyciu znanych ataków rekonstrukcyjnych?

2. Czy można zrekonstruować ważne cechy twarzy za pomocą **jakiegokolwiek** algorytmu?

Odpowiedź na pierwsze pytanie jest negatywna, co można łatwo sprawdzić poprzez analizę działania obecnie wykorzystywanych algorytmów rekonstrukcji, takich jak [MJ13, FLY14, MCYJ19]. Jednak pełne bezpieczeństwo rozumiane jako odporność na wszystkie ataki rekonstrukcyjne sprowadza się do odpowiedzi na drugie pytanie. Wydaje się mało prawdopodobne, aby formalnie wykazać, że taki atak nie istnieje. Udało nam się jednak dostarczyć argumenty za tym, że rekonstrukcja twarzy na podstawie tylko danych zbieranych i zapisywanych w bazie przez naszą metodę jest trudna. Mianowicie w podrozdziale 4.5.1 pokazujemy, że korelacja między **znormalizowanym** przemieszczeniem niektórych punktów a ich położeniem oraz odległością jest zwykle niewielka.

Co więcej, wykazujemy, że potrafimy skutecznie wybrać podzbiór cech o szczególnie małej korelacji tak, aby zredukować „wymianę" informacji pomiędzy naszą biometrią a odpowiadającą jej statyczną biometrią twarzy.

Można to zrobić bez znaczącego obniżenia dokładności naszej metody, gdy cechy o niskiej korelacji z dopowiadającymi im statycznymi cechami twarzy są starannie dobierane. Uważamy, że takie podejście może być przedmiotem kolejnych badań i może prowadzić do ogólnego projektu zapisów biometrycznych, które nie są „przekształcalne" na inne dane biometryczne, tak aby adwersarz posiadający jeden zapis biometryczny nie mógł go dopasować do zapisów innych danych biometrycznych.

4.5.1 Porównanie przesunięć do cech statycznych twarzy

Zbadaliśmy korelację między standardowymi, statycznymi rysami twarzy a cechami przemieszczeń twarzy (mimiką). Cechy statyczne, które odpowiadają punktom orientacyjnym zwracanym przez algorytm [KS14], określają takie elementy charakterystyczne twarzy jak: wielkość i kształt głowy (1-17), pozycje i wymiary prawej oraz lewej brwi (18-27), wartości określające pozycje i wielkość nosa (28-36), umiejscowienie i rozmiar oczu (37-48) oraz pozycja i wielkość ust (49-68).

Dla każdej n-tej cechy obliczyliśmy standardowy współczynnik korelacji zgodnie ze wzorem:

$$\rho_n = \frac{1}{\Sigma_{i\in\Omega} F^{(i)}} \frac{\Sigma_{i\in\Omega}\Sigma_{f=1}^{F^{(i)}}(D_{nf}^{(i)} - \mu_{D_n})(S_{nf}^{(i)} - \mu_{S_n})}{\sigma_{D_n}\sigma_{S_n}}, \tag{4.3}$$

gdzie Ω określa zbiór próbek, $D_{nf}^{(i)}$ to wartość przemieszczenia n-tego punktu pomiędzy f-tą i $(f+1)$-tą ramką dla i-tej próbki, $S_{nf}^{(i)}$ to wartość statycznej cechy twarzy dla n-tego punktu i-tej próbki. Symbole μ i σ oznaczają odpowiednio średnią i odchylenie standardowe, które są liczone odpowiednio według następujących wzorów:

$$\mu_{D_n} = \frac{1}{\Sigma_{i\in\Omega} F^{(i)}} \Sigma_{i\in\Omega}\Sigma_{f=1}^{F^{(i)}} D_{nf}^{(i)} \tag{4.4}$$

oraz

$$\sigma_{D_n} = \left(\frac{1}{\Sigma_{i\in\Omega} F^{(i)}} \Sigma_{i\in\Omega}\Sigma_{f=1}^{F^{(i)}} (D_{nf}^{(i)} - \mu_{D_n})^2 \right)^{\frac{1}{2}}. \tag{4.5}$$

Średnia współczynnika korelacji dla wszystkich 136 punktów wyniosła 0.007546, a odchylenie standardowe było równe 0.022897. W przypadku liczenia wartości bezwzględnych dla przemieszczeń ($|D_n|$) oraz cech statycznych ($|S_n|$), średnia współczynnika korelacji wyniosła 0.186707, a odchylenie standardowe było równe 0.046074. Przykładowe wyniki współczynników korelacji dla wybranych punktów charakterystycznych zostały przedstawione

w tablicy 4.2. Wyniki sugerują, że większość cech przemieszczeń jest słabo skorelowana ze statycznymi cechami twarzy.

Punkt	Orientacja	Współ. korelacji ρ_n dla D_n	Współ. korelacji ρ_n dla $\|D_n\|$
7	pionowa	-0.000682	0.261425
7	pozioma	0.010158	0.217390
18	pionowa	-0.000229	0.174853
18	pozioma	0.072713	0.191044
30	pionowa	0.043991	0.120583
30	pozioma	-0.000379	0.015931
53	pionowa	0.017458	0.130539
53	pozioma	-0.014167	0.139086

Tablica 4.2: Wyniki analizy korelacji dla wybranych cech. 7 punkt w orientacji pionowej osiągnął najwyższą wartość bezwzględną spośród wszystkich cech, 18 punkt w orientacji pionowej i poziomej miał, za to, odpowiednio najmniejszą i najwyższą wartość względną. Dla punktu 30 widać dużą rozbieżność wartości bezwzględnych między pionową i poziomą orientacją.

Aby poprawić bezpieczeństwo naszej metody, a przy tym utrzymać zadowalający poziom skuteczności identyfikacji (przyjęliśmy, że jest to IR $\geq$ 50%), sprawdziliśmy jakie wartości współczynnika identyfikacji IR osiąga nasz system po usunięciu dobrze skorelowanych cech. W tym celu iteracyjnie usuwaliśmy cechy o najwyższej korelacji dla ich bezwzględnych wartości. Wyniki eksperymentu pokazują, że akceptowalny współczynnik identyfikacji utrzymał się do momentu usunięcia aż **50 najbardziej skorelowanych** cech. Wyniki zostały przedstawione na rysunku 4.7 (zielona linia).

Zauważmy, że nasze podejście nie może być postrzegane jako formalnie udowodnione, matematycznie bezpieczne podejście (ang. *provable secure*), ponieważ mogą istnieć inne relacje, które potencjalnie mogą być wykorzystane przez adwersarza. Zaprezentowane podejście jednak znacznie ogranicza informacje o statycznych cechach twarzy, które można wyekstrahować (dowolnymi metodami) po wycieku bazy danych.

4.5.2 Metoda jako *cancelable biometrics*

Naszą metodę można również rozważać w kategorii biometrii usuwalnej (ang. *cancelable biometrics*), co oznacza, że przed zapisaniem cech biometrycznych w bazie danych system biometryczny odpowiednio przekształca te cechy (wykorzystując w tym celu funkcję parametryzowaną kluczem). Dzięki temu w momencie wycieku bazy danych możemy zmienić klucz tak,

aby wykradzione dane nie mogły być już wykorzystane do nielegalnej autoryzacji przez adwersarza. W naszym przypadku możemy przyjąć, że wypowiada fraza jest kluczem. Dzięki czemu w momencie wycieku bazy istnieje możliwość zmiany wypowiadanej frazy. Co więcej, możemy zbudować system, w którym wypowiadane zdania będą różnić się zależnie od osób, co jeszcze dodatkowo zwiększy bezpieczeństwo w rozumieniu biometrii usuwalnej (osobny klucz dla każdego użytkownika).

Podobnie jak w innych systemach biometrycznych typu *cancelable*, zakładamy, że klucz (zdanie) można bezpiecznie przechowywać, m.in. w ludzkiej pamięci lub poza bazą danych. Co więcej, metody biometrii z możliwością usuwania są szczególnie ważne w przypadku biometrii twarzy ze względu na zwiększone ryzyko „pobrania" tego typu cech od nieświadomej tego działania osoby. Metody oparte na takich cechach jak odcisk palca czy geometria dłoni są znacząco mniej podatne na ten problem. Wiąże się to głównie z faktem, że twarz jest najbardziej rozpowszechnionym nośnikiem cech biometrycznych ze względu na popularność mediów społecznościowych oraz powszechność kamer w przestrzeni publicznej. Mimo tego, istnieje znacznie mniej metod usuwania danych biometrycznych zastosowanych do systemów rozpoznawania twarzy w porównaniu do innych rodzajów biometrii. Zauważmy, że artykuł przeglądowy [PRC15] wymienił jedynie dwa takie rozwiązania dla biometrii twarzy, w przeciwieństwie do aż ośmiu schematów ochrony tęczówki oka.

4.6 Podsumowanie

W tym rozdziale przedstawiliśmy nowatorską metodę wykorzystania cech mimiki twarzy do skonstruowania systemu biometrycznego, który ma na celu zapewnienie *prywatności na etapie projektowania (privacy-by-design)*. Eksperymentalnie wykazaliśmy, że nasza metoda osiągnęła współczynnik identyfikacji równy 70.79% dla wektora cech użytkownika składającego się z 3 próbek zarejestrowanych w bazie danych. Następnie pokazaliśmy, że wykorzystując jedynie 44 (ze wszystkich 136) punktów charakterystycznych nasza metoda uzyskuje IR równy 68.54%. Istotnym wynikiem pracy jest analiza korelacji cech mimiki twarzy względem cech statycznych, pozwalająca na ograniczenie najbardziej skorelowanych liniowo cech i tym samym zwiększenie prywatności rozwiązania poprzez ograniczenie możliwości rekonstrukcji twarzy. Przetestowaliśmy nasz system na zbiorze danych zawierającym 191 filmów wideo 34 różnych osób. Omówiliśmy również wykorzystanie naszej metody w kontekście biometrii usuwalnej.

Rozdział 5

Zapobieganie atakowi spoofingowemu w rozpoznawaniu mówcy

5.1 Wprowadzenie

Ten rozdział jest poświęcony jednemu z najistotniejszych zagadnień z obszaru biometrii, mianowicie zapobieganiu atakom spoofingowym, czyli podszywaniu się pod innego użytkownika. Istotą problemu jest stworzenie dodatkowej warstwy bezpieczeństwa w systemie biometrycznym, która będzie odpowiedzialna za wyłapywanie próbek, które nie są autentycznie (spoofingowych), dzięki czemu takie próbki będą odrzucanie i nie będą poddawane procesowi autoryzacji biometrycznej.

Opracowanie systemu biometrycznego, który zapewnia ochronę przed spoofingiem jest szczególnie istotne w dobie szerokiego i łatwego dostępu do danych biometrycznych poszczególnych osób w mediach społecznościowych, np. Facebook, YouTube, TikTok. Użytkownicy w tych portalach udostępniają zdjęcia, filmy oraz inne multimedia, na których znajdują się między innymi cechy biometryczne twarzy, mimiki oraz głosu. Co więcej, twórcy systemów biometrycznych chcąc zwiększyć wygodę ich użytkowania dopuszczają przeprowadzenie procesu autoryzacji na próbkach biometrycznych pobieranych bez ustalonego reżimu, tj. w dowolny, a przy tym wygody dla użytkowania, sposób. Przykładami takiego podejścia są systemy do rozpoznawania głosu, które działają niezależnie od wymawianej frazy, tzw. podejście *text-independent* [NCZ17], oraz systemy biometryczne oparte na zdjęciu twarzy pobieranej niezależnie od jej ustawienia i pozycji względem sensora, tzw. *faces in the wild* [HRBLM07, LM14]. Dwa wyżej opisane fakty powodują, że dzisiaj uwzględnienie ochrony przed atakami spoofingowymi wydaje się być koniecznością podczas opracowywania rozwiązania do

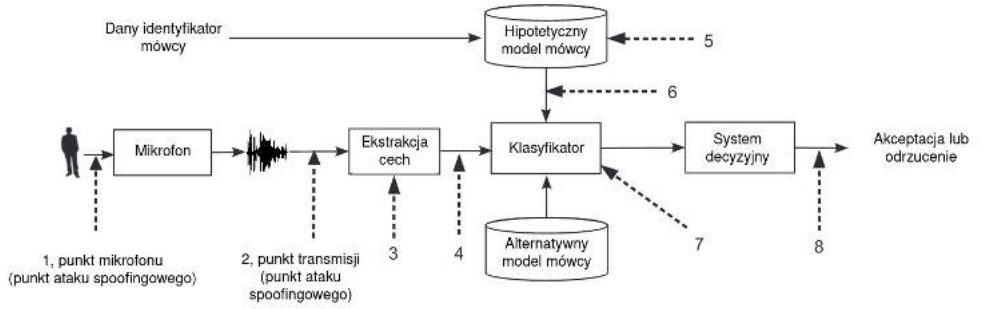
autoryzacji biometrycznej.

W tym rozdziale skupiamy się na metodzie do detekcji ataku podszywania się dla systemu biometrycznego opartego na analizie głosu. Metoda zapobiega jednemu z najbardziej powszechnych i praktycznych scenariuszy ataków, tj. próbie dostępu poprzez odtworzenie bezpośrednio do mikrofonu nagranego wcześniej głosu (dostęp fizyczny, ang. *physical access*). Dźwięk jest propagowany w przestrzeni fizycznej („świecie rzeczywistym") i odczytywany przez sensor (mikrofon), który finalnie konwertuje odebrany dźwięk do jego reprezentacji cyfrowej i w takiej formie trafia on do systemu do autoryzacji biometrycznej. W przypadku ataku, głos użytkownika jest nagrywany przez adwersarza za pomocą innego mikrofonu, następnie jest zapisywany w formie cyfrowej i ponownie odtwarzany. Ta subtelna różnica w postaci podwójnej konwersji dźwięku oraz zniekształceń wynikających z technicznych ograniczeń sensorów odbierających dźwięk oraz urządzeń emitujących daje możliwość określenia, czy dana próbka głosu pochodzi od użytkownika, czy jest próbą podszycia się. W pracy dodatkowo pokazujemy praktyczne aspekty metody, uwzględniając jej rolę jako systemu wspomagającego autoryzację biometryczną oraz fakt ograniczonej dostępności do zasobów obliczeniowych.

Rozdział w dużej mierze został oparty na pracy [BKM+19], w której skupiliśmy się na temacie regularyzacji modelu pod kątem ataków oraz na samej efektywności sieci neuronowych. W pracy zaprezentowaliśmy nowe podejście wspomagające proces modelowania sieci neuronowych – *Attack-Out Cross-Validation* oraz zaproponowaliśmy rozwiązanie oparte na lekkiej konwolucyjnej sieci neuronowej (LCNN) oraz bayesowskiej sieci neuronowej. Następnie przeprowadziliśmy analizę złożoności danych ukazującą, które parametry ataku najbardziej wpływają na jego powodzenie.

5.2 Ataki na system automatycznej weryfikacji mówcy

System *antyspoofingowy* jest zazwyczaj rozważany jako rozszerzenie systemu do automatycznej weryfikacji mówcy (ang. *automatic speaker verifiacation*, ASV). System ASV składa się z wielu komponentów, które mogą stać się potencjalnym celem ataku dla adwersarza. Dodatkowo, również, kanały komunikacyjne między poszczególnymi komponentami mogą stanowić przestrzeń do przeprowadzenia ataku na ów system. Rysunek 5.1 przedstawia typowy schemat systemu ASV wraz z oznaczonymi potencjalnymi miejscami, w których adwersarz może wykonać atak na system. Punkty te można podzielić na dwie kategorie – *ataki bezpośrednie* oraz *ataki pośrednie* [WEK+15].



Rysunek 5.1: Schemat typowego systemu do automatycznej weryfikacji mówcy (ASV) oraz osiem możliwych punktów ataku. Punkty o numerach 1 oraz 2 są miejscami, w których możliwy jest do przeprowadzenia atak podszywania się. Te punkty są określane jako ataki bezpośrednie. Z kolei punkty o numerach od 3 do 8 są zaliczane do ataków pośrednich i są rozważane jako ataki na system. Na podstawie [WEK+15].

Ataki bezpośrednie

Ataki bezpośrednie są kategorią ataków charakterystycznych dla systemów biometrycznych. Cechują się tym, że adwersarz wykrada dane biometryczne, modyfikuje je lub syntetycznie generuje w celu podszycia się pod rzeczywistego użytkownika. Konsekwencją tego jest wykonanie przez system ASV autoryzacji fałszywego wzorca biometrycznego, co w efekcie prowadzi do nieuprawnionego dostępu do chronionych zasobów. W tym przypadku, adwersarz nie ma dostępu do systemu ASV, nie musi też mieć wiedzy o sposobie jego działania i zastosowanych algorytmach do weryfikacji mówcy, w tym ekstraktora cech, klasyfikatora oraz funkcji decyzyjnej. W przypadku ataków bezpośrednich rozważamy dwa punkty dostępu [WEK+15]:

1. Punkt mikrofonu – fałszywa próbka biometryczna jest odczytywana bezpośrednio poprzez sensor wejścia, adwersarz nie ingeruje w żaden kolejny komponent systemu ASV.

2. Punkt transmisji – próbka biometryczna jest pobierana poprzez mikrofon, następnie jest przechwytywana przez adwersarza i modyfikowana przed przekazaniem do ekstraktora cech. Ten przypadek opisuje sytuację, w której sensor wejściowy znajduje się w infrastrukturze niezależnej od sytemu ASV, nad którą nie ma wystarczającej kontroli, np. próbka jest pobierana za pomocą mikrofonu w telefonie, następnie jest przekazywana do systemu ASV z wykorzystaniem sieci telekomunikacyjnej (m.in. opartej na standardowym GSM). Przykładowym

wdrożonym rozwiązaniem, które jest potencjalnie podatne na atak w tym punkcie jest system Voice ID banku HSBC [HSB].

Ataki pośrednie

Kategoria ataków pośrednich definiuje szeroką grupę ataków do której zaliczamy podmianę lub obejście jednego z komponentów systemu ASV, dodanie, podmianę lub usunięcie informacji o użytkowniku w bazie danych, a także ataki wykorzystujące techniki inżynierii wstecznej [MDFGOG11], czy też ataki oparte na standardowych metodach *hill-climbing* [GMF+10, MHC15].

W przypadku znaczącej części ataków zaliczanej do tej kategorii, adwersarz w celu wykonania ataku musi mieć zapewniony dostęp do infrastruktury obliczeniowej, sieciowej lub pamięciowej w obrębie której działa system. ASV jest systemem informatycznym, stąd może on być chroniony standardowymi metodami zwiększającymi bezpieczeństwo takich systemów. Dodatkowo ochrona może być rozszerzona o dedykowane dla metod biometrycznych rozwiązania, takie jak techniki ochrony wzorców biometrycznych – *hashable biometrics*, które pozwalają na utworzenie skrótu wzorca biometrycznego wykorzystując do tego algorytmy cechujące się wysoką tolerancją na zaburzenia danych oraz stosując mechanizmy korekcji błędów [MRLW01, WCD+17], czy też *cancelable biometrics* [Adl09], która pozwala na zmianę wykorzystywanego wzorca biometrycznego w przypadku wycieku danych z systemu. W efekcie czego za dużo bardziej niebezpieczne dla systemów biometrycznych uważa się ataki bezpośrednie, jakimi są ataki spoofingowe [FZHK06].

5.3 Rozszerzenie automatycznej weryfikacji mówcy o system antyspoofingowy

System do zapobiegania atakom spoofingowym (ang. *spoof countermeasures*, CM), dalej nazywany systemem CM, ma na celu wykrycie próbek głosu, które zostały ponownie odtworzone lub zmodyfikowane cyfrowo w celu podszycia się pod innego użytkowania. System CM działa zawsze jako wsparcie dla systemu do automatycznej weryfikacji mówcy, zaprojektowanego do oceny zgodności między dwiema próbkami głosu pobranymi od mówców. Pierwsza wypowiedź jest pobierana w celu rejestracji próbki do bazy danych (tzw. *enrollment phase*). Z kolei druga próbka jest pobierana w momencie, gdy ma zostać przeprowadzona weryfikacja mówcy przez system ASV (tzw.

trial phase). W tej fazie system ASV porównuje pobraną próbkę z próbką referencyjną dostarczoną w fazie rejestracji w celu ustalenia, czy obie pochodzą od tego samego mówcy.

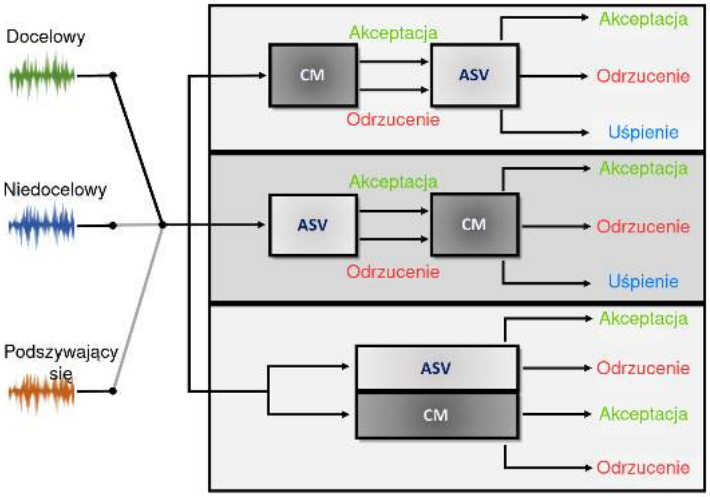
W przypadku rozwiązań łączących systemy ASV oraz CM, wyróżniamy trzy rodzaje danych wejściowych do obu systemów:

- Docelowy (ang. *target*) – grupa obejmująca próbki, które powinny zostać zweryfikowane pozytywnie w trakcie weryfikacji przez system ASV. Są to próbki zgodnie z próbkami pobranymi wcześniej w celu rejestracji do bazy danych.
- Niedocelowy (ang. *nontarget*) – grupa obejmująca próbki, wykorzystane w celu nieautoryzowanego dostania się do systemu. Są to próbki, które powinny zostać odrzucone przez system ASV ze względu na ich niespójność ze wzorcami zarejestrowanymi w bazie.
- Podszywający się, spoofowy (ang. *spoof*) – grupa próbek wykorzystana w celu przeprowadzenia ataku podszycia się.

Próbki docelowe są jedynymi, które powinny zostać pozytywnie zweryfikowane w ramach systemu łączącego podsystemy ASV oraz CM, z kolei próbki niedocelowe oraz spoofowe powinny zostać odrzucenie przez jeden z wymienionych podsystemów.

Standardowe systemy ASV są zazwyczaj zaprojektowane jedynie do rozróżniania między próbkami docelowymi i niedocelowymi. Te systemy są ograniczone w kontekście wykrycia oraz późniejszego odrzucenia próbki spoofowej, która dla owych systemów jest „podobna" do próbki docelowej, ze względu na mocno zbliżone cechy biometryczne zakodowane w obu próbkach. Zatem metody ASV są zdolne do rozróżnienia próbek niedocelowych od pozostałych dwóch typów – docelowych oraz spoofowych. Co wprost indukuje, że systemy ASV są podatne na ataki spoofingowe. W przeciwieństwie, do metod ASV, rozwiązania CM są zaprojektowane do rozróżnienia między próbkami spoofowymi oraz grupą próbek, do których zaliczamy zarówno próbki docelowe, jak i niedocelowe (tzw. grupa próbek autentycznych, ang. *bona fide*). Opisane wyżej ograniczenia w działaniu systemów ASV oraz CM wymuszają opracowanie pewnych scenariuszy dla różnych przypadków współdziałania obu systemów [KLD+18]. Rozważamy trzy takie scenariusze. Zostały one zobrazowanie na rysunku 5.2, a ich dokładniejszy opis umieszczony w rozdziale 5.4.

Warto podkreślić, że w części prac nad systemami ASV oraz CM podjęto wysiłek stworzenia metod wielozadaniowych, tzn. łączących systemy



Rysunek 5.2: Schemat ilustrujący trzy scenariusze komunikacji między systemami ASV oraz CM. Na podstawie [KLD+18].

do automatycznej weryfikacji mówcy oraz antyspoofingowy w ramach jednej metody [SKK+15, LSZW20, GAGLD+21]. Takie podejście z pewnością poprawia złożoność obliczeniową całego systemu oraz rozwiązuje cześć problemów wynikających z konieczności opracowania optymalnego podejścia do połączenia obu systemów. Jednak obecnie standardowe podejście łączące dwa niezależne systemy ASV oraz CM cieszy się większą popularnością, ze względu na możliwość niezależnej pracy nad dwoma rozwiązaniami oraz możliwości wykorzystania podejścia „plug-and-play", które pozwala na podmianę jednego z systemów niezależnie od pracy drugiego. Jest to szczególnie istotne w przypadku rozwiązania ASV, które działa dla określonej bazy wzorców próbek biometrycznych, a jego aktualizacja może wymagać zbudowania tej bazy na nowo. Z kolei system CM może wymagać cyklicznych aktualizacji, np. ze względu na konieczność uwzględnienia nowych typów ataków spoofingowych opracowanych już po jego wdrożeniu i uruchomieniu. Najbardziej powszechnym scenariuszem łączenia metod CM oraz ASV jest ten, w którym system CM analizuje próbkę mowy przed ASV (scenariusz pierwszy na rysunku 5.2).

5.4 Formalizacja problemu rozszerzenia systemu ASV o system wspomagający CM

System do automatycznej weryfikacji mówcy operuje na parze wypowiedzi głosowych $X = (\mathcal{X}_{en}, X_{tr})$, gdzie $\mathcal{X}_{en}$ jest zbiorem próbek (w szczególności jednoelementowym) pobieranym podczas fazy rejestracji i jest skojarzony

z identyfikatorem mówcy, a X_{tr} jest próbką pobraną w celu jego weryfikacji. Para X jest nazywana *docelową*, gdy każda próbka $X_{en} \in \mathcal{X}_{en}$ oraz X_{tr} pochodzą od tego samego mówcy, a w przypadku, gdy mówcy dla obu próbek są różni, mówimy o parze *niedocelowej*. Próbka dźwiękowa może być reprezentowana, między innymi, jako czysta fala dźwiękowa, sekwencja cech spektralnych, cechy typu i-vector, parametry modelu typu Gaussian Mixture, czy też jako wektor wynikowy sieci neuronowej. System ASV przyjmuje X i na jego podstawie zwraca wynik detekcji:

$$r = \mathrm{ASV}(\mathcal{X}_{en}, X_{tr}) \in \mathbb{R}, \tag{5.1}$$

który określa, czy $\mathcal{X}_{en}$ oraz X_{tr} pochodzą od tego samego mówcy.

Wynik detekcji r jest powiązany z dwiema wzajemnie przeciwnymi hipotezami – hipotezą zerową (docelową) mówiącą, że $\mathcal{X}_{en}$ oraz X_{tr} są pobrane od tego samego mówcy oraz hipotezą alternatywną (niedocelową), która opisuje sytuację, gdy próbki pochodzą od różnych mówców. Wyższy wyniki r indukuje mocniejsze wsparcie dla hipotezy docelowej. Decyzja o akceptacji lub odrzuceniu hipotez jest podejmowana przy ustalonym progu (ang. *threshold*) $t \in \mathbb{R}$. Jeśli $r > t$ to hipoteza docelowa jest akceptowana. W przeciwnym przypadku zostaje zaakceptowana hipoteza niedocelowa [KLD+18, KDE+20].

System do zapobiegania atakom spoofingowym działa z wykorzystaniem innych modeli oraz inaczej zdefiniowane hipotezy. W przypadku systemów ASV zazwyczaj rozważamy budowę modeli *per mówca* (przy czym istnieją również prace opisujące rozwiązania oparte na pojedynczych modelach porównujących dwie wypowiedzi, np. bazujące na sieciach syjamskich [CS11, KH20]), z kolei w przypadku systemu CM rozważamy wyuczenie dwóch modeli – pierwszy dla próbek spoofowych oraz drugi dla wypowiedzi autentycznych. Oznacza to metoda CM jest klasyfikatorem binarnym (możliwe są dwa wyniki – próbka jest autentyczna lub spoofowa). Stąd system binarny można zdefiniować wprost jako metodę rozróżniającą między dwiema klasami. Rozszerzając wcześniejszą notację $X = (\mathcal{X}_{en}, X_{tr})$, gdzie $\mathcal{X}_{en}$ jest zbiorem próbek autentycznych lub spoofowych. Próbka X_{tr}, podobnie jak w przypadku systemu ASV, jest pobrana w celu poddania jej ocenienie autentyczności. W takim przypadku definiujemy system CM jako

$$q = \mathrm{CM}(\{\mathcal{X}_{en}^{bona}, \mathcal{X}_{en}^{spoof}\}, X_{tr}) \in \mathbb{R}, \tag{5.2}$$

gdzie $\{\mathcal{X}_{en}^{bona}, \mathcal{X}_{en}^{spoof}\}$ to zbiór próbek zarówno autentycznych, jak i zmodyfikowanych w celu wykonania ataku podszycia się. W pracy upraszczamy

notację stosując zapis CM(X_{tr}), pomijając pierwszy argument dla funkcji CM.

Wynik q jest powiązany z dwiema hipotezami. Pierwsza z nich, hipoteza zerowa (o autentyczności) opisuje pochodzenie pobranej próbki X_{tr} jako autentyczne, hipoteza alternatywna (o podszywaniu się) mówi o tym, że próbka została przygotowana celu przeprowadzenia ataku podszycia się. System CM zwraca wynik $q \in \mathbb{R}$, który jest zdefiniowany w taki sposób, że wyższa zwrócona wartość wzmacnia hipotezę o autentyczności. Tak samo jak w systemie ASV, definiujemy próg akceptacji próbki autentycznej $s \in \mathbb{R}$. Jeśli $q > s$, wtedy przyjmujemy za prawdziwą hipotezę zerową (próbka jest autentyczna). W przeciwnym wypadku, akceptujmy hipotezę o podszywaniu się (próbka uznana jest za próbkę spoofową) [KLD$^+$18, KDE$^+$20].

Odrębne systemy ASV oraz CM mogą ze sobą współpracować opierając się na trzech scenariuszach — jako połączone w ramach kombinacji kaskadowej (dwa scenariusze) lub jako systemy równoległe. Wszystkie trzy zostały przedstawione na rysunku 5.2. W przypadku zastosowania podejścia kaskadowego, jeśli pierwszy z systemów odrzuci próbkę, kolejny przechodzi w tryb uśpienia i nie wykonanie obliczeń, a ostateczna decyzja o odrzuceniu próbki jest zależna tylko od wyniku pierwszego systemu. Oczywiście, gdy pierwszy system zaakceptuje próbkę, drugi ją dalej przetwarza. W scenariuszu opartym na kombinacji równoległej, oba systemy pracują niezależnie od siebie, stąd próbka jest zawsze przetwarzana przez system ASV oraz system CM. Finalnie w scenariuszu równoległym próbka jest akceptowana tylko w przypadku podwójnej akceptacji przez oba systemy. Najpowszechniejszym scenariuszem łączenia obu systemów jest kaskada, w której próbka pobrana od mówcy najpierw analizowana jest przez system CM [KDE$^+$20].

W przypadku systemu łączącego ASV oraz CM, mówimy o dwóch przypadkach błędów:

- Fałszywa akceptacja (ang. *false accpetance*) – błąd ten występuje w przypadku, gdy dwa systemy zaakceptują próbkę, która w rzeczywistości należny do jednej z dwóch grup:
  - niedocelowych – system CM akceptuje próbkę nie popełniając przy tym błędu, gdyż próbka nie jest próbką spoofową, a błąd popełnia system ASV;
  - spoofowych – przyjmując, że system ASV nie popełnia błędu tylko przy akceptacji próbki docelowej, ten rodzaj próbki powinien zostać odrzucony przez system ASV. Jednak próbka spo-

ofowa będzie charakteryzować się mocno zbliżonymi cechami biometrycznymi do cech odczytanych z próbki pobranej podczas fazy rejestracji, dlatego istnieje duże ryzyko, że system ASV zaakceptuje próbkę spoofową (ze względu na jej duże podobieństwo do próbki docelowej). Stąd w tym przypadku polegamy na działaniu systemu CM, który popełni błąd, gdy zaakceptuje próbkę spoofową.

- Fałszywe odrzucenie, pominięcie (ang. *false reject*, *miss*) – opisuje błędy wynikające z odrzucenia właściwej próbki, tzn. system ASV odrzuci próbkę docelową, a system CM odrzuci próbkę inną niż spoofowa (docelową lub niedocelową). Warto zaznaczyć, że w przypadku odrzucenia próbki niedocelowej przez podsystem CM lub nie dopuszczenia do pozytywnej autoryzacji mówcy dla próbki spoofowej przez podsystem ASV, cały system zadziała poprawnie, przy czym, jest to nieoczekiwane działanie poszczególnych komponentów systemu.

5.5 Metody walidacji

System CM wspomaga główny system ASV. Wspólnie działanie dwóch komponentów zwiększa bezpieczeństwo całego rozwiązania do weryfikacji biometrycznej. Omawiane podsystemy skupiają się na różnych zadaniach, jednak wspólnie mają dążyć do jednego celu nadrzędnego, tj. poprawnej i odpornej na ataki weryfikacji użytkownika. Wydajność systemu antyspoofingowego naturalnie wpływa na wydajność systemu do automatycznej weryfikacji mówcy. Sytuacja, w której system CM nie wykryje próbki spoofowej, zwiększa ryzyko dokonania pomyłki przez system ASV oraz spowodowania błędu fałszywej akceptacji. Z kolei przypadek zablokowania próbki docelowej przez system CM, spowoduje fałszywe odrzucenie, a sama próbka nie zostanie przetworzona przez system ASV. To oznacza, że moduł CM wpływa pośrednio na niezawodność i użyteczność systemu ASV. Co więcej, nie zawsze jest prawdziwa zależność, że bardziej wydajny system CM (osiągający niższy błąd zrównoważony *ang. equal error rate, EER*) spowoduje większą wydajność (oraz niezawodność) całościowego systemu łączącego podsystemy CM i ASV.

Oznacza to, że konieczne jest opracowanie metryki, która odzwierciedli wydajność systemu jako całości. Przy czym metryka ta powinna wspierać możliwość niezależnego rozwoju obu komponentów – ASV i CM. Co więcej, metryka powinna odzwierciedlać wspomagający charakter systemu CM

względem ASV, co oznacza, że finalnie oba podsystemy powinny wzmacniać wydajność komponentu ASV. W przypadku modułu CM, metryka powinna działać poprawnie niezależnie od formy ataku spoofingowego (np. konwersja głosu, synteza mowy, odtworzenie). Kolejny istotnym warunkiem jest wpływ systemu CM na działanie całego systemu autoryzacyjnego CM+ASV w sensie bayesowskiego minimalnego ryzyka, co oznacza uwzględnienie, że poszczególnie ataki spoofowe nie wpływają w takim samym stopniu na wydajność całego systemu CM+ASV [KLD+18]. Przykładowo przyjmijmy, że „słaby" atak spoofingowy przypomina atak zerowysiłkowy (ang. *zero-effort*), co oznacza, że analizowana próbka głosu jest próbką niedocelową (pobraną od innego mówcy), która będzie cechowała się wysoką jakością i naturalnie brzmiącą mową. Taka próbka zostanie zaakceptowana przez system CM, ale będzie odrzucona przez moduł ASV, który rozpozna, że próbka jest daleka od próbki głosu docelowego mówcy. W tym sensie, błąd pominięcia przez system CM spowoduje jedynie niewielki koszt dla systemu jako całości, gdyż finalnie moduł ASV odrzuci próbkę. Sytuacja będzie wyglądała zupełnie inaczej dla ataku spoofwego cechującego się dobrą jakością (w kontekście podobieństwa cech biometrycznych), który w przypadku błędu fałszywej akceptacji przez system CM, spowoduje wysokie ryzyko popełnienia kolejnego błędu przez system ASV i zaakceptowania próbki ze względu na jej podobieństwo do mówcy docelowego. Oznacza to, że poszczególne przypadki, z którymi nie radzi sobie system CM, mogą powodować wyższy lub niższy koszt błędu dla całościowego systemu. To oznacza, że metryka powinna odzwierciedlać koszt poszczególnych decyzji w sensie bayesowskiego minimalnego ryzyka.

Rozwiązanie spełniające wszystkie wymienione wcześniej kryteria dla systemu łączącego podsystemy ASV i CM jest oparte na funkcji kosztu detekcji (ang. *detection cost function*, DCF) [Bd06] zaproponowanej dla pojedynczego systemu ASV i zatwierdzonej przez amerykański Narodowy Instytut Standaryzacji i Technologii (ang. *National Institute of Standards and Technology*, NIST) w ramach kampanii dotyczącej ewaluacji rozwiązań do rozpoznawania mówców [DPMR00]. Nowa metryka jest rozszerzeniem DCF, które pozwala na ocenienie wydajności dwóch systemów realizujących odmienne zadania i działających jako całość. Zaproponowana nazwa dla nowej metryki to funkcja kosztu detekcji tandemu (ang. *tandem detection cost function*, t-DCF) [KLD+18, KDE+20].

5.5.1 Koszt detekcji – wprowadzenie

Funkcja kosztu detekcji

W zagadnieniu *bayesowskiego minimalnego ryzyka klasyfikacji* skupiamy się na dokonaniu predykcji, w taki sposób, że wybór określonych klas będzie minimalizował ryzyko błędów, rozumianych jako poniesiony koszt. Niech $\mathcal{A} = \{\alpha_1, \ldots, \alpha_L\}$ będzie *zbiorem akcji* reprezentującym decyzje możliwe do podjęcia przez dany klasyfikator. Następnie $\Theta = \{\theta_1, \ldots, \theta_M\}$ będzie *zbiorem założeń* reprezentującym faktyczny stan (etykiety, *ground truth*). Wybór określonej akcji dla danego założenia niesie za sobą koszt, tzn. każdej akcji α przy prawdziwym założeniu θ przypisujemy nieujemny *koszt* $C(\alpha|\theta) \in \mathbb{R}^+$. Oczywiście, akcji właściwej dla danego założenia przypisujemy koszt, który jest równy 0.

W rozważanym przez nas systemie, zbiór akcji to decyzje (akceptacja, odrzucenie, uśpienie) możliwe do zwrócenia przez oba komponenty systemu – ASV i CM. Rozważamy zbiór akcji o rozmiarze $|\mathcal{A}| = 6$, gdzie możliwe akcje to:

$$\begin{aligned}
\alpha_1 &= \{akceptacja, akceptacja\},\\
\alpha_2 &= \{akceptacja, odrzucenie\},\\
\alpha_3 &= \{odrzucenie, akceptacja\}\\
\alpha_4 &= \{odrzucenie, odrzucenie\},\\
\alpha_5 &= \{odrzucenie, uśpienie\},\\
\alpha_6 &= \{uśpienie, odrzucenie\}.
\end{aligned} \tag{5.3}$$

Przyjmujemy, że pierwszy element akcji tyczy się decyzji podjętej przez system ASV, a drugi przez system CM. Warto podkreślić, że zbiór $\mathcal{A}$ może być zredukowany zależnie od typu wybranego scenariusza łączenia podsystemów (patrz rozdział 5.4 oraz rysunek 5.2). Przykładowo, akcje α_5 i α_6 nie będą możliwe w scenariuszu równoległym. Z kolei zbiór założeń ma rozmiar $|\Theta| = 3$ i jest równy:

$$\Theta = \{\theta_{tar}, \theta_{non}, \theta_{spoof}\}. \tag{5.4}$$

Dla próbki, której etykieta jest jednym elementem ze zbioru założeń, podejmujemy akcję, która niesie za sobą pewien koszt. Koszt podjętej akcji jest ustalany *a priori* i jest on zależny od środowiska w jakim ma pracować system do automatycznej weryfikacji mówcy rozszerzony o komponent do zapobiegania atakowi spoofingowemu. Jest to ocena subiektywna podejmowana przez twórców, testerów oraz odbiorców systemu. Przykładowo,

ustalony koszt $C(a_1|\theta_{non})$ może być dużo większy niż $C(\alpha_3|\theta_{tar})$ w systemie autoryzacji do konta bankowego, gdzie każde błąd typu fałszywej akceptacji niesie za sobą poważne konsekwencje, niż w przypadku logowania do panelu multimedialnego lodówki (np. Family Hub oferowany przez firmę Samsung [Sam]).

Następnie, określamy prawdopodobieństwo podjęcia przez system łączący komponenty ASV i CM akcji α_i dla założenia θ_j jako $P(\alpha_i|\theta_j)$ oraz definiujemy prawdopodobieństwo wstępne (ang. *prior probability*) dla każdego z założeń $\pi_i = P(\theta_i)$. Oczywiście, $\sum_i \pi_i = 1$, gdyż założenia θ_i są rozłączne. W przypadku walidacji systemu na konkretnym zbiorze testowym, wartość prawdopodobieństwa wstępnego dla poszczególnych założeń jest liczona wprost jako stosunek liczby danych próbek do wszystkich próbek w zbiorze. Rozkład prawdopodobieństwa wstępnego w naszym przypadku nawiązuje do częstotliwości występowania poszczególnych kategorii próbek. Z kolei prawdopodobieństwo podejmowania akcji przez system przy określonych założeniach wprost opisuje wydajność tego systemu – jak często popełniane są błędy oraz jaki jest to typ błędów. Przykładowo, dla systemu ASV współczynnik FAR $= P(\alpha_1|\theta_{non}) + P(\alpha_2|\theta_{non})$.

Mając na uwadze powyższe założenia, definiujemy koszt detekcji dla akcji (koszt podjęcia konkretnej akcji):

$$\mathrm{DCF}(\alpha_i) = \sum_{j=1}^{M} \pi_j C(\alpha_i|\theta_j) P(\alpha_i|\theta_j). \tag{5.5}$$

W celu otrzymania pełnej funkcji kosztu detekcji (DCF) liczymy sumę poszczególnych kosztów detekcji po wszystkich podejmowanych akcjach:

$$\mathrm{DCF} = \sum_{i=1}^{L} \mathrm{DCF}(\alpha_i) = \sum_{i=1}^{L} \sum_{j=1}^{M} \pi_j C(\alpha_i|\theta_j) P(\alpha_i|\theta_j). \tag{5.6}$$

Zauważmy, że wartość DCF służy do określenia błędów w działaniu sytemu, uwzględniając przy tym koszt poszczególnych rodzajów błędów. Widzimy, że im wyższa wartość funkcji, tym mniej wydajny (odporny) jest system. Przy czym w zbiorze akcji rozważamy również przypadki, które nie są błędami dla poszczególnych założeń, np. $P(\alpha_1|\theta_{tar})$ nie jest prawdopodobieństwem wystąpienia błędu systemu tylko podjęcia poprawnej akcji dla tego założenia. Jednak w takim przypadku koszt $C(\alpha_1|\theta_{tar}) = 0$, co w efekcie powoduje nieuwzględnienie tego przypadku w całościowej funkcji kosztu detekcji.

NIST DCF dla systemu ASV

W przypadku sztandarowej funkcji DCF liczonej dla konwencjonalnego systemu ASV bez wykorzystania dodatkowego komponentu CM (zatwierdzonej przez Narodowy Instytut Standaryzacji i Technologii [DPMR00]) rozważamy tylko dwa rodzaje próbek – docelowe i niedocelowe, a sam system dokonuje tylko dwóch działań – odrzuca lub akceptuje użytkownika. W tym przypadku zbiór akcji jest zdefiniowany w następujący sposób:

$$\mathcal{A} = \{akceptacja, odrzucenie\}, \tag{5.7}$$

a zbiór założeń jako:

$$\Theta = \{\theta_{tar}, \theta_{non}\}. \tag{5.8}$$

System ASV podejmuje decyzję opierając się na ustalonym progu akceptacji t. Koszt C jest dodatni tylko w dwóch przypadkach, gdy system podejmuje jedną z dwóch błędnych decyzji:

- gdy dla próbek docelowych θ_{tar} system ASV zwróci wartość poniżej progu akceptacji (błąd typu fałszywego odrzucenia) – prawdopodobieństwo tego zdarzenia definiujemy jako $P^{asv}_{miss}(t) = P(r \leq t|\theta_{tar})$, a koszt tego rodzaju błędu to $C(odrzucenie|\theta_{tar})$;
- gdy dla próbek niedocelowych θ_{non} system ASV zwróci wartość powyżej progu akceptacji (błąd typu fałszywej akceptacji) – prawdopodobieństwo tego zdarzenia definiujemy jako $P^{asv}_{fa}(t) = P(r > t|\theta_{non})$, a koszt definiujemy jako $C(akceptacja|\theta_{non})$.

Mając powyższe, definiujemy funkcję kosztu detekcji (NIST DCF):

$$\begin{aligned} DCF(t) =& \pi_{tar} C(odrzucenie|\theta_{tar}) P^{asv}_{miss}(t) \\ &+ \pi_{non} C(akceptacja|\theta_{non}) P^{asv}_{fa}(t). \end{aligned} \tag{5.9}$$

Metryka jest zadana dla progu akceptacji t i wymaga określenia trzech parametrów - dwóch wartości kosztu C oraz π_{tar} (wiemy, że $\pi_{non} = (1 - \pi_{tar})$). Wybór owych parametrów jest zależny od istności poszczególnych błędów i jest dokonywany z uwzględnieniem specyfiki użytkowania danego systemu. Przy czym w procesie ewaluacji rozwiązań do rozpoznawania mówców realizowanego przez NIST (*Speaker Recognition Evaluation*) parametry kosztu zostały zaproponowane jako: $C(akceptacja|\theta_{non}) = 10$ oraz $C(odrzucenie|\theta_{tar}) = 1$ [RDPM00].

5.5.2 Funkcja kosztu detekcji tandemu (t-DCF)

System składający się z komponentów ASV i CM definiujemy jako para $S = (ASV, CM)$. Przyjmujemy, że nie mamy bezpośredniego dostępu do poszczególnych podsystemów i polegamy jedynie na wynikach zwróconych przez oba komponenty $(r_i, q_i) \in \mathbb{R}^2$, gdzie $i = 1, 2, \ldots, N$ oraz N jest liczbą prób przygotowanych w celu ewaluacji systemu. Zbiór ten składa się z N_{tar} próbek docelowych, N_{non} próbek niedocelowych oraz N_{spoof} próbek spoofowych. Wszystkie trzy podzbiory są rozłączne, zatem $N = N_{tar} + N_{non} + N_{spoof}$.

Mimo że w wykorzystywanej notacji posługujemy się parą (r_i, q_i), w celach testowych możemy policzyć wyniki dla systemów ASV i CM niezależnie. W szczególności, wyniki r_i dla systemu ASV oraz q_i dla systemu CM mogą zostać podane dla różnych zbiorów testowych, uwzględniając konieczność policzenia wyników dla systemu CM na zbiorach próbek docelowych oraz niedocelowych wykorzystanych przy liczeniu wyników dla systemu ASV. Jest to szczególnie ważne w przypadku prowadzenia badań nad oboma systemami w sposób niezależny, co jest powszechne zarówno w badaniach akademickich, jak i komercyjnych. W przypadku pracy nad poszczególnymi komponentami możemy wykorzystywać rożne zbiory testowe, dzięki czemu ograniczeniem przestaje być fakt, że zbiory danych dla problemu rozpoznawania mówcy często nie zawierają próbek spoofowych. I odwrotnie, zbiory z próbkami ataków, są często dużo mniej liczne (w rozumieniu liczby osób udostępniających swój głos), za to cechują się różnorodnością pod kątem przeprowadzonych ataków.

Metryka t-DCF uwzględnia progi akceptacji dla dwóch systemów (w naszym przypadku, t dla ASV oraz s dla CM), stąd definiujemy poszczególne współczynniki błędu (ang. *error rates*) jako funkcje zależne od progu. Takie rozwiązanie nadal aplikuje się do systemów, w których nie ustalone zostały owe progi, co oznacza, że takie podejście nie traci na swojej ogólności.

Współczynniki błędów detekcji dla systemu ASV

Dla systemu do automatycznej weryfikacji mówcy definiujemy trzy współczynniki błędów. Pierwszym z nich jest współczynnik fałszywszych odrzuceń (ang. *false rejection rate*, FRR):

$$P_{miss}^{asv}(t) = \int_{-\infty}^{t} p_R(r|\theta_{tar})dt \approx \frac{1}{N_{tar}} \sum_{i \in \Lambda_{tar}} \mathbb{1}_{\{r:\ r \leq t\}}(r_i), \tag{5.10}$$

gdzie $p_R(\cdot|\theta_{tar})$ jest funkcją gęstości prawdopodobieństwa wyników zwracanych przez system ASV dla próbek docelowych, z kolei Λ_{tar} określa zbiór indeksów próbek docelowych, które powinny zostać zaakceptowane. Funkcja $\mathbb{1}_{\{r:\ r\leq t\}} : \mathbb{R} \rightarrow \{0, 1\}$ jest funkcją charakterystyczną dla zbioru wyników spełniających zadany warunek, w tym wypadku $r \leq t$.

Drugi rodzaj błędu zdefiniowany dla systemu ASV to błąd typu fałszywych akceptacji, mierzony z wykorzystaniem współczynnika fałszywych akceptacji (ang. *false acceptance rate*, FAR):

$$P_{fa}^{asv}(t) = \int_t^{\infty} p_R(r|\theta_{non})dt \approx \frac{1}{N_{non}} \sum_{i\in\Lambda_{non}} \mathbb{1}_{\{r:\ r>t\}}(r_i), \tag{5.11}$$

gdzie $p_R(\cdot|\theta_{non})$ jest funkcją gęstości prawdopodobieństwa wyników zwracanych przez system ASV dla próbek niedocelowych, a Λ_{non} określa zbiór indeksów próbek niedocelowych, które powinny zostać odrzucone przez system ASV.

Warto nadmienić, że w przypadku liczenia wskaźnika FAR rozważane są tylko przypadki błędów na próbkach niedocelowych, a nie są brane pod uwagę próbki spoofowe (które również powinny być odrzucone). Jest to spowodowane tym, że próbki spoofowe są mocno zbliżone, pod kątem swoich cech biometrycznych, do próbek docelowych, co wpływa również na podobieństwo rozkładów wyników zwracanych przez system ASV dla właśnie tych klas próbek głosu, tzn. $p_R(r|\theta_{spoof}) \approx p_R(r|\theta_{tar})$. Stworzenie systemu ASV, który poprawnie obsłuży próbki spoofowe opierając się tylko na cechach biometrycznych, wymagałoby ustalenia bardzo wysokich progów akceptacji (lub bardzo niskich, zależnie od założeń systemu), efektem czego byłby mocno niezbalansowane wskaźniki FAR i FRR.

Rozważmy skrajny przypadek, w którym jesteśmy w stanie przeprowadzić wyjątkowo skutecznie atak typu podszywanie się (atak spoofingowy). Przyjmijmy, że $p_R(r|\theta_{spoof}) = p_R(r|\theta_{tar})$. W tym przypadku wskaźnik poprawnych akceptacji ($1 - \text{FRR}$) dla próbek docelowych (pobranych wprost od rzeczywistych użytkowników systemu) będzie wynosił dokładnie tle samo co FAR dla próbek przygotowanych do przeprowadzenia ataku spoofingowego. Przykładowo, dla systemu ASV z FRR = 0.01, dostajemy FAR = 0.99 na podzbiorze próbek spoofowych, które *de facto* zostaną zaakceptowane za względu na ich znaczące podobieństwo względem próbek docelowych.

Zaprezentowany przykład krańcowy pozwala wyrobić sobie pewną intuicję, że próbki spoofowe rozważane jako konieczne do odrzucenia przez system ASV mogą znacząco zaburzyć standardowe wskaźniki liczone dla

systemu biometrycznego (FAR i FRR), a co za tym idzie uniemożliwić właściwą ewaluację takiego systemu. Jednak przy założeniu, że system ASV będzie wspierany przez moduł CM nie ma potrzeby, aby badać wydajność (odporność) tego pierwszego w kontekście ataków spoofowych, ze względu na to, że system CM ma za zadanie odrzucać takie próbki. Przy czym nadal wartościową wiedzą jest informacja o tym jak system ASV działa, gdy na wejściu zostanie wprowadzona próbka spoofowa. Dzięki temu, jesteśmy w stanie określić, czy próbki przygotowane to wykonania ataku podszywania się są niosą za sobą zagrożenie dla systemu ASV, czy też dany atak jest łatwy do zidentyfikowania i odrzucenia nawet przez sam system ASV. W tym celu uwzględniamy w metryce prawdopodobieństwo zdarzenia, że próbka spoofowa zostanie fałszywie zaakceptowana (nie zostanie odrzucona) przez system ASV, czyli

$$P^{asv}_{fa,spoof}(t) = \int_t^{\infty} p_R(r|\theta_{spoof})dt \approx \frac{1}{N_{spoof}} \sum_{i \in \Lambda_{spoof}} \mathbb{1}_{\{r:\ r>t\}}(r_i), \quad (5.12)$$

gdzie $p_R(\cdot|\theta_{spoof})$ jest funkcją gęstości prawdopodobieństwa wyników zwracanych przez system ASV dla próbek spoofowych, a Λ_{spoof} określa zbiór indeksów próbek spoofowych. Warto nadmienić, że w przypadku, gdy systemy ASV i CM są ewaluowane na osobnych zbiorach danych i nie jest możliwe policzenie $P^{asv}_{fa,spoof}(t)$, wtedy zakładamy, że system ASV nie jest w stanie poradzić sobie z próbkami spoofowymi, czyli $P^{asv}_{fa,spoof}(t) = 1$.

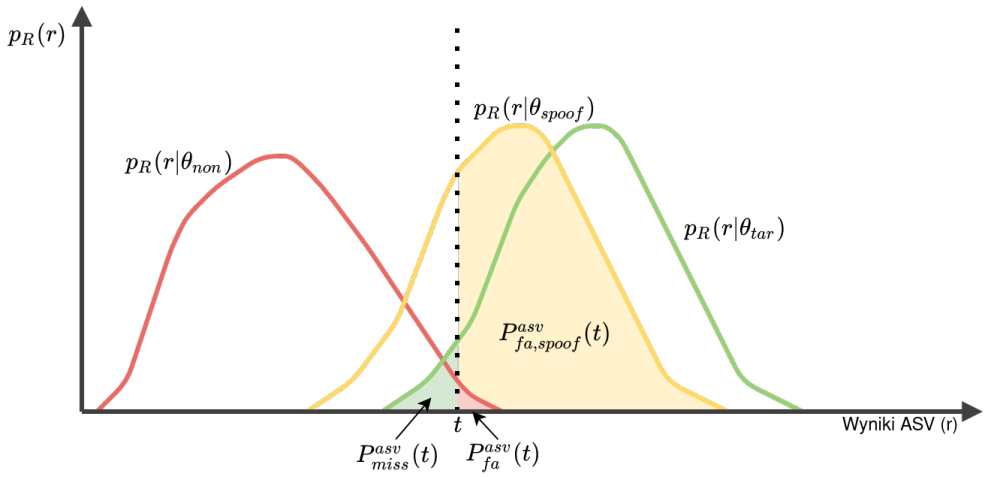
Opisane współczynniki błędów dla systemu ASV zostały zobrazowane na rysunku 5.3.

Poziomy błędu detekcji dla systemu CM

Zadaniem systemu CM jest rozróżnienie między próbką wprowadzoną do systemu bezpośrednio przez użytkownika, a próbką odpowiednio przygotowaną lub zmodyfikowaną w celu podszycia się pod niego. Co istotnie, system ten nie ma na celu wykrycia próbek niedocelowych, które są rozważane jako ataki zero-wysiłkowe (ang. *zero-effort attacks*). Stąd z perspektywy systemu do zapobiegania atakom spoofingowym próbki docelowe θ_{tar} oraz próbki niedocelowe θ_{non} traktujemy jako jedną kategorię i reprezentujemy jako zbiór próbek autentycznych $\theta_{bona} = \{\theta_{tar}, \theta_{non}\}$.

Dla systemu CM, tak jak dla systemu ASV, rozważamy dwa współczynniki błędów. Pierwszy z nich to współczynnik fałszywszych odrzuceń próbek autentycznych (FRR):

$$P^{cm}_{miss}(s) = \int_{-\infty}^{s} p_Q(q|\theta_{bona})dq \approx \frac{1}{N_{tar} + N_{non}} \sum_{i \in \Lambda_{tar} \cup \Lambda_{non}} \mathbb{1}_{\{q:\ q \leq s\}}(q_i), \quad (5.13)$$



Rysunek 5.3: Wizualizacja współczynników błędów dla systemu ASV. Wykres przedstawia funkcje gęstości prawdopodobieństwa wyników zawracanych przez system ASV dla poszczególnych próbek – docelowych, niedocelowych i spoofowych. Na wykresie zostały również przedstawione obszary współczynników błędów dla danego progu akceptacji t (całka po zbiorze akceptowalnych wartości).

gdzie $p_Q(\cdot|\theta_{bona})$ jest funkcją gęstości prawdopodobieństwa wyników zwracanych przez system CM dla próbek autentycznych (docelowych i niedocelowych).

Drugi z rozważanych współczynników to współczynnik fałszywych akceptacji próbek spoofowych (FAR):

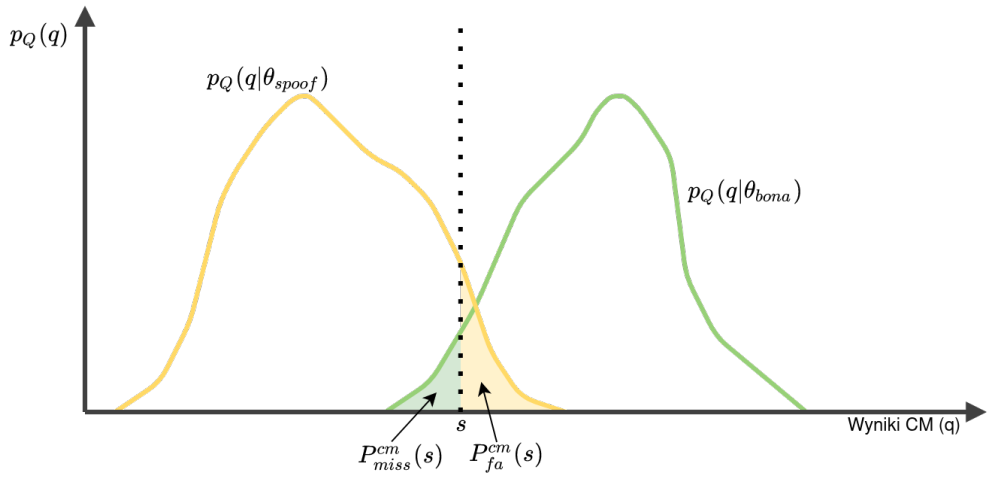
$$P^{cm}_{fa}(s) = \int_s^{\infty} p_Q(q|\theta_{spoof})dq \approx \frac{1}{N_{spoof}} \sum_{i \in \Lambda_{spoof}} \mathbb{1}_{\{q:\ q>s\}}(q_i), \tag{5.14}$$

gdzie $p_Q(\cdot|\theta_{spoof})$ jest funkcją gęstości prawdopodobieństwa wyników zwracanych przez system CM dla próbek spoofowych.

Opisane współczynniki błędów dla systemu CM zostały zaprezentowane na rysunku 5.4.

t-DCF dla systemu ASV i CM

Wcześniej zdefiniowany zbiór akcji $\mathcal{A} = \{\alpha_1, \ldots, \alpha_6\}$ zawiera sześć akcji możliwych do wykonania przez system dwukomponentowy (ASV i CM), przy czym wykonanie części z akcji ze zbioru jest możliwe tylko w przypadku niektórych scenariuszy współpracy między dwoma komponentami. Przykładowo akcje, w których jeden z komponentów przechodzi w tryb uśpienia,



Rysunek 5.4: Wizualizacja współczynników błędów dla systemu CM. Wykres przedstawia funkcje gęstości prawdopodobieństwa wyników zawracanych przez system CM dla poszczególnych próbek – autentycznych i spoofowych. Na wykresie zostały również przedstawione obszary współczynników błędów dla danego progu akceptacji s (całka po zbiorze akceptowalnych wartości).

nie są możliwe do wykonania w scenariuszu równoległym, a w scenariuszach kaskadowych, w przypadku odrzucenia próbki przez pierwszy system, drugi zawsze podejmuje akcję uśpienia, tzn. nie przetwarza próbki. Autorzy tworząc miarę skuteczności t-DCF przyjęli najpowszechniejszy scenariusz współpracy obu systemów – moduły współpracują kaskadowo i system CM wykonuje swoje zadanie jako pierwszy [KLD+18, KDE+20]. Ograniczenie się do jednego scenariusza upraszcza opracowanie funkcji, ze względu na redukcję możliwych przypadków wystąpienia błędów. Również nasze rozwiązanie do wykrywania ataków spoofingowych jest testowane w ramach scenariusza kaskadowego z pierwszeństwem wykonania zadania przez system CM. Niemniej jednak, adaptacja funkcji t-DCF do innego scenariusza współpracy jest możliwa.

W rozważanym przypadku zbiór akcji to $\mathcal{A} = \{\alpha_1, \alpha_3, \alpha_6\}$, co oznacza, że gdy moduł CM odrzuca próbkę, ASV przechodzi w tryb uśpienia, z kolei, gdy moduł CM zaakceptuje próbkę, ASV podejmuje jedną z dwóch możliwych decyzji – akceptacja lub odrzucenie. Następnie definiujemy zbiór założeń $\Theta = \{\theta_{tar}, \theta_{non}, \theta_{spoof}\}$ oraz prawdopodobieństwa wstępne (prawdopodobieństwo wystąpienia konkretnego rodzaju próbki) – π_{tar}, π_{non} i π_{spoof}. Następnie określamy cztery możliwe niezerowe koszty podjętych akcji:

- C_{miss}^{asv} - koszt odrzucenia próbki docelowej przez system ASV,
- C_{fa}^{asv} - koszt akceptacji próbki niedocelowej przez system ASV,
- C_{miss}^{cm} - koszt odrzucenia próbki autentycznej (docelowej lub niedocelowej) przez system CM,
- C_{fa}^{cm} - koszt akceptacji próbki spoofowej przez system CM.

Kolejnym krokiem jest zdefiniowanie i policzenie prawdopodobieństwa zwrócenia niepoprawnego wyniku przez współpracujące ze sobą komponenty. Tutaj, rozważamy cztery przypadki błędów:

1. Podjęcie akcji α_3 dla próbek θ_{tar}. CM akceptuje próbkę docelową, którą fałszywie odrzuca ASV:

$$P_a(s,t) = (1 - P_{miss}^{cm}(s)) \cdot P_{miss}^{asv}(t). \tag{5.15}$$

2. Podjęcie akcji α_1 dla próbek θ_{non}. CM akceptuje próbkę niedocelową, którą następnie fałszywie akcentuje ASV:

$$P_b(s,t) = (1 - P_{miss}^{cm}(s)) \cdot P_{fa}^{asv}(t). \tag{5.16}$$

3. Podjęcie akcji α_1 dla próbek θ_{spoof}. CM fałszywie akceptuję próbkę spoofową, która później jest akceptowana przez ASV:

$$P_c(s,t) = P_{fa}^{cm}(s) \cdot P_{fa,spoof}^{asv}(t). \tag{5.17}$$

4. Podjęcie akcji α_6 dla próbek θ_{tar}. CM fałszywie odrzuca próbkę docelowa, a system ASV nie podejmuje działania (uśpienie):

$$P_d(s,t) = P_{miss}^{cm}(s). \tag{5.18}$$

Ostatecznie liczmy t-DCF zgodnie ze wzorem:

$$\begin{aligned} \text{t-DCF}(s,t) = {} & C_{miss}^{asv} \cdot \pi_{tar} \cdot P_a(s,t) \\ & + C_{fa}^{asv} \cdot \pi_{non} \cdot P_b(s,t) \\ & + C_{fa}^{cm} \cdot \pi_{spoof} \cdot P_c(s,t) \\ & + C_{miss}^{cm} \cdot \pi_{tar} \cdot P_d(s,t). \end{aligned} \tag{5.19}$$

5.6 System CM

5.6.1 Wprowadzenie

W tym rozdziale prezentujemy rozwiązanie do wykrywania ataków spoofingowych oparte na dwóch konwolucyjnych sieciach neuronowych. Metoda wykrywa atak typu odtworzenie (ang. *replay attack*) z dostępnym fizycznym (ang. *physical access*). Przy tworzeniu naszego rozwiązania skupiamy się na kwestiach złożoności obliczeniowej – zakładamy, że rozwiązanie wspomagające autoryzację użytkownika musi działać szybko, w czasie rzeczywistym, oraz mając do dyspozycji ograniczone moce obliczeniowe. Kolejną istotną kwestią braną przez nas pod uwagę podczas opracowywania rozwiązania jest generalizacja modelu na rożne warianty ataku – wykorzystywany zbiór danych zawiera jedynie 9 różnie sparametryzowanych ataków (zależnie od jakości mikrofonu i jego odległości od nagrywanej osoby), za to nasze rozwiązanie ma działać poprawnie również dla innych wariantów ataku typu odtwarzanie z dostępem fizycznym, które nie zostały uwzględnione w danym zbiorze danych, a mogą wystąpić podczas pracy systemu w warunkach rzeczywistych. Jest to szczególnie istotne w przypadku projektowania rozwiązania bazującego na sieciach neuronowych, gdzie nie mamy bezpośredniej kontroli nad dobrem analizowanych cech, a jedynie określamy oczekiwany rezultat działania metody (poprzez funkcję kosztu).

Pierwszy wykorzystywany model bazuje na małej bayesowskiej sieci neuronowej, co jest umotywowane hipotezą twierdzącą, że modele sieci bayesowskich są odporne na zagadnienie nadmiernego dopasowania (ang. *overfitting*) [KW14, BCKW15]. Architektura drugiego wykorzystanego modelu bazuje na standardowych warstwach konwolucyjnych przeplatanych dodatkowymi warstwami *Max Feature Map* [WHST18], które są zaprojektowane w sposób pozwalający na redukcję złożoności obliczeniowej sieci. Drugi model jest trenowany z wykorzystaniem kilku technik regularyzacyjnych w celu poprawy jego zdolności do generalizacji. W przypadku modelu opartego na sieci bayesowskiej, przyjmujemy, że dodatkowe techniki regularyzacji nie są potrzebne, że względu na naturalne własności tego typu architektury do osiągania wysokiej odporności na nadmierne dopasowanie, w szczególności, że ten model posiada niewielką liczbę warstw. W procesie modelowania sieci neuronowych wykorzystujemy nowo zaproponowaną przez nas zmodyfikowaną metodę sprawdzenia krzyżowego – *Attack-Out Cross Validation*, która polega na podzieleniu zbioru na podzbiory treningowy, testowy i walidacyjny w taki sposób, żeby w każdym z nich znalazły się inne warianty ataku.

Takie podejście pozwala na przeprowadzenie dokładniejszej analizy wyników trenowanych modeli pod kątem ich uogólnienia na różne ataki. Opisywanie rozwiązanie zostało przetestowane podczas konkursu ASVspoof 2019 Challenge [TWV+19] organizowanego przez europejskie oraz japońskie instytucje badawcze – EURECOM, Narodowy Instytut Informatyki w Japonii, INRIA oraz Uniwersytet Wschodniej Finlandii. W ramach konkursu została opracowana specjalna baza danych zawierająca dane dla dwóch typów dostępu – logicznego (LA) oraz fizycznego (PA). Pierwszy z nich nawiązuje do przypadku, w którym adwersarz wysyła cyfrową próbkę dźwięku bezpośrednio do systemu (np. podczas zdalnej autoryzacji poprzez telefon), a ataku podszycia się dokonuje z użyciem technik konwersji dźwięku (również oparte na sieciach neuronowych). W drugim przypadku adwersarz odtwarza wcześniej nagrany dźwięk autentycznej osoby [WYT+20]. Nasze rozwiązanie pozwala na uodpornienie się na atak z dostępem fizycznym, a wyniki jakie osiągnęło rozwiązanie to min-tDCF = 0.0219 oraz EER = 0.88%, co oznacza 10 krotną poprawę względem modeli bazowych wybranych przez autorów konkursu. **Rozwiązanie zajęło 6 miejsce w konkursie spośród 52 drużyn, cechując się przy tym najmniejszą złożonością obliczeniową zaproponowanych modeli sieci neuronowych.**

5.6.2 Problem uogólnienia cech dla modeli antyspoofingowych

Problem nadmiernego dopasowania sieci neuronowej do danych treningowych (ang. *overfitting*) dotyka praktycznie każdego zagadnienia rozwijanego z wykorzystaniem modeli uczenia maszynowego, szczególnie sieci neuronowych. W zdecydowanej większości przypadków problem ten jest wynikiem ograniczonych i mało liczebnych zbiorów danych traktowanych przy użyciu modeli o wysokiej pojemności (liczonej jako liczba wag uczących modelu) [CLG00, Yin19].

W przypadku danych dźwiękowych problem ten staje się jeszcze bardziej istotny. Jest to spowodowane reprezentacją cyfrową sygnału audio, na którą mocno wpływają parametry techniczne sensora wykorzystanego do pobrania próbki dźwiękowej. Mikrofony pracują w ramach określonej szerokości pasma przenoszenia, próbkowania, czy poziomu ciśnienia akustycznego. Wszystkie te oraz inne parametry powodują, że reprezentacja cyfrowa tej samej próbki dźwiękowej może różnić się zależnie od mikrofonu wykorzystanego do jej nagrania. Warto podkreślić, że parametry techniczne mikrofonów są dobierane w taki sposób, żeby później odtwarzany dźwięk nie różnił od jego pierwotnego źródła, przy czym owe różnice mają być niezau-

ważalne w zakresie słyszalności ludzkiego ucha, które jest węższe od pełnego możliwego zakresu widma sygnału dźwiękowego. To też oznacza, że wykorzystując mikrofony specjalistyczne (o szerszym paśmie) możemy z pomocą sieci neuronowej przeprowadzić analizę dużo szerszego widma, niż gdyby tej analizy miał dokonać ekspert akustyk poprzez swój narząd słuchu. Sieć neuronowa w pewnych przypadkach może nauczyć się podejmowania decyzji na bazie częstotliwości spoza zakresu ludzkiego ucha [Pla19]. To z kolei może nieść za sobą dodatkowe wyzwania w identyfikacji źródła powodującego efekt nadmiernego dopasowania.

Jednak najistotniejszym utrudnieniem w modelowaniu rozwiązań uczenia maszynowego dla danych audio z perspektywy problemu nadmiernego dopasowania jest specyfika cyfrowej reprezentacji próbki audio, która, tak jak wspomniano powyżej, jest wyjątkowo wrażliwa na parametry techniczne mikrofonu. Gdy dane treningowe zebrane do uczenia modelu będą zawierać jedynie próbki pobrane z wykorzystaniem jednego lub kilku mikrofonów może okazać się, że wytrenowany model „dopasuje się" do parametrów technicznych owych urządzeń, co wprost oznacza, że wybrane podczas treningu wzorce nie zostaną uogólnione dla danego zagadnienia. W takim przypadku model będzie działał dobrze, ale tylko, gdy dane wejściowe zostaną zarejestrowane przez odpowiednie mikrofony – te same, które zostały wykorzystane do pobrania danych treningowych. Opisany efekt jest dużo bardziej wyraźny w dziedzinie audio, niż innych dziedzinach, np. obraz lub wideo, gdzie reprezentacja danych jest dużo bardziej odporna na parametry urządzeń nagrywających. Jednym z rozwiązań tego problemu jest odpowiednia adaptacja fali dźwiękowej zależnie od rodzaju wykorzystywanego mikrofonu, np. metodą korelacji spektrum [NPK20], przy czym tego typu metody mają ograniczone zastosowanie, np. wymagają dodatkowej wiedzy o urządzeniach nagrywających.

Wyżej opisany problem jest potęgowany, gdy celem modelu jest rozpoznanie ataku typu odtworzenie z dostępem fizycznym, gdzie za próbkę autentyczną uznaje się bezpośrednie nagranie mówcy z wykorzystaniem mikrofonu docelowego, które następnie jest przetwarzane cyfrowo. W przypadku ataku, próbka pobrana od mówcy jest nagrywana przez mikrofon adwersarza, potem odtwarzana przy użyciu głośnika, aby ostatecznie ponownie została zarejestrowana przez mikrofon docelowy w celu przetworzenia przez system ASV (lub ASV+CM). Jak łatwo zauważyć, model uczenia maszynowego musi, w tym przypadku, oprzeć swoją analizę o pewne subtelne artefakty (zniekształcenia) sygnału, które są wynikiem pobierania próbki przez pośredni mikrofon adwersarza oraz jej odtwarzania przez pośredni

głośnik. Oznacza to, że idealny model antyspoofingowy musi nauczyć się rozpoznawać uogólnione zniekształcenia (na które wpływają cechy fizyczne dźwięku, np. pogłos, ciśnienie akustyczne, zakres widma), które występują w przypadku wielokrotnego nagrania i odtworzenia próbki dźwiękowej we wszystkich typach urządzeń nagrywających i odtwarzających, a przy tym rozróżnić i zignorować artefakty, które są efektem wrażliwej reprezentacji cyfrowej danych dźwiękowych wynikającej z parametrów technicznych samych mikrofonów oraz głośników.

Ostatnim godnym zaznaczenia faktem wpływającym negatywnie na problem nadmiernego dopasowania jest mały rozmiar dostępnych zbiorów danych dla zadania rozpoznania ataku typu podszywanie się, co samo w sobie niesie ograniczenia w modelowaniu rozwiązań opartych na technikach uczenia maszynowego. Dodatkowo zbiory danych są stosunkowo mało różnorodne pod kątem wariantów (parametrów) ataku, wykorzystywanych urządzeń oraz mówców użyczających swojego głosu. W takim przypadku wyuczenie sieci neuronowej, która podejmuje decyzje opierając się na cechach znajdujących się bezpośrednio w zbiorze treningowym (rozróżniając dane jedynie względem etykiet – *spoof* i *bonafide*) jest znacząco utrudnione z powodu braku rozkładu danych uwzględniającego szeroką gamę ataków. Niesie to za sobą ryzyko, że sieć rozpozna cechy specyficzne dla danego wąskiego zbioru dostarczonych przypadków, ale już nie dla ogólnej reprezentacji danej etykiety. W przypadku rozwiązania do zapobiegania atakowi spoofingowemu, sieć neuronowa podczas treningu nie ma możliwości „zobaczenia" wszystkich możliwych wariantów ataku. W trakcie modelowania sieci istotne jest zwrócenie szczególnej uwagi na to, aby wyuczone cechy dobrze uogólniały przypadek *spoofa*, a nie ograniczały się do cech specyficznych dla wąskiego grona wariantów ataku uwzględnionych w zborze danych. Istotność tego problemu została również dostrzeżona przez organizatorów konkursu ASVspoof 2019 Challenge [ASV19], którzy w zbiorach treningowych nie uwzględnili części wariantów ataku, za to wykorzystali je podczas ewaluacji [WYT+20]. Ten problem minimalizujemy poprzez wykorzystanie naszej metody Attack-Out Cross Validation.

5.6.3 Wcześniejsze prace

W ostatnich latach badano wiele różnych podejść do zagadnienia wykrywania ataków odtwarzania z dostępem fizycznym. Analiza nowych cech sygnału, rozwój architektur modeli i modyfikacje procedur treningowych przyczyniły się do znacznego wzrostu jakości wykrywania ataków spoofin-

gowych. Na poziomie reprezentacji cech, wydaje się, że istotną zmianą jest spostrzeżenie, że tradycyjnie wykorzystywane cechy w metodach rozpoznawania mówcy lub mowy, takie jak współczynniki mel-cepstralne (ang. *Mel-Frequency Cepstral Coefficients*, MFCC), mogą nie być optymalne w zastosowaniach polegających na wykrywaniu próbek spoofowych, ponieważ większość informacji dyskryminacyjnej jest zawarta w wyższych częstotliwościach [WKZ+17, LNM+17]. W ostatnich latach zaproponowano również nowe cechy, w szczególności stałe współczynniki Q-cepstralne (ang. *Q-Cepstral Coefficients*), ale ich zastosowanie jest ograniczone ze względu podatność na nadmierne dopasowanie i słabą generalizację [TDE17]. Dążenie do znalezienia optymalnej architektury systemu zaowocowało wieloma nowatorskimi klasyfikatorami opartymi na sieciach neuronowych, konsekwentnie przewyższającymi tradycyjne odpowiedniki takie jak GMM, czy i-wektory [LNM+17, CCL+17]. Procedury treningowe klasyfikatorów opartych na głębokich sieciach neuronowych, wymagające dużych zasobów danych, zostały ulepszone poprzez techniki augmentacji [CCL+17] oraz uczenie wielozadaniowe [SJH+18].

5.6.4 Opis zboru danych ASVspoof Challenge 2019

Nasze rozwiązanie trenujemy oraz testujemy wykorzystując zbiór danych opracowany w ramach konkursu ASVspoof Challenge 2019 [WYT+20]. Korzystamy ze zbioru stworzonego dla zagadnienia antyspoofingowego z dostępem fizycznym (ang. *physical access, PA*). Zbiór danych jest podzielony na trzy podzbiory – *treningowy*, *testowy* (nazwany też *rozwojowym*) oraz *ewaluacyjny*, gdzie dwa pierwsze posiadają ogólnodostępne etykiety (próbka autentyczna vs. próbka spoofa) oraz metadane zawierające parametry ataków, informacje o środowisku akustycznym oraz identyfikator mówcy. Zbiór treningowy oraz testowy zawierają autentyczne nagrania 40 mówców oraz ich powtórne odtworzenia (atak spoofingowy), przy czym nagrania są rozłącznie podzielone miedzy dwa zbiory (treningowy i testowy) po 20 osób. W naszym rozwiązaniu proponujemy jednak inny podział podzbioru treningowego i testowego oraz wykorzystanie opracowanej przez nas metody Attack-Out Cross-Validation (patrz rozdział 5.6.6).

Środowisko akustyczne, w którym odbywały się nagrania oraz przeprowadzano atak było symulowane w celu zapewnienia kontroli nad parametrami akustycznymi oraz parametrami ataku spoofingowego. Każda próbka została pobrana w środowisku akustycznym, które ma zdefiniowane następujące parametry – powierzchnia pokoju (metry kwadratowe), czas pogłosu

T_{60} (milisekundy), odległość mówcy od mikrofonu systemu ASV (centymetry). Dla wszystkich parametrów zostały określone następujące przedziały: $2-5\ m^2$, $5-10\ m^2$, $10-20\ m^2$ dla powierzchni pokoju, $50-200\ ms$, $200-600\ ms$, $600-1000\ ms$ dla czasu pogłosu T_{60} oraz $10-50\ cm$, $50-100\ cm$, $100-150\ cm$ dla parametru odległości mówcy od mikrofonu systemu ASV.

Metody ataku są sparametryzowane dwoma cechami. Pierwsza z nich to odległość mikrofonu adwersarza od osoby mówiącej, gdzie określamy trzy przedziały odległości: $10-50\ cm$, $50-100\ cm$, powyżej $100\ cm$. Druga z nich to jakość urządzenia odtwarzającego, dla którego wyróżniamy trzy klasy: *perfekcyjna*, *wysoka*, *niska*. Metoda ataku jest definiowana jako dwójka $(A|B|C)^2$, gdzie kolejne litery odpowiadają parametrom w kolejności wymienionej powyżej, gdzie pierwsza z liter nawiązuje do przedziału odległości adwersarza od mówcy, a druga litera określa klasę jakości wykorzystanego urządzenia odtwarzającego. Przykładowo metoda zdefiniowana jako AC oznacza, że adwersarz znajdował się w odległości $10-50\ cm$ od mówcy oraz wykorzystał niskiej jakości urządzenie. Dzięki wyszczególnionym parametrom możemy wyróżnić 9 metod wariantów ataków.

5.6.5 Analiza złożoności danych

Inspirując się [HCE+17, GEF+17] przeprowadzamy analizę, w której próbujemy zdefiniować wektor własności próbek (ang. *embeddings*) uwzględniając informacje o typach ataków, co umożliwi znalezienie podobieństw między poszczególnymi atakami. Taka analiza pozwoli na wskazanie, czy jakieś warianty ataków są do siebie podobnie (z perspektyw modelu) oraz które z nich mogą sprawić potencjalnie najwięcej trudności w wykryciu próbki spoofowej. W tym celu trenujemy sieć neuronową, której zadaniem jest wyuczenie się mapowania próbki dźwięku (w postaci spektrogramu) na abstrakcyjny wektor reprezentujący cechy danej próbki. Wykorzystujemy sieć neuronową o prostej architekturze zawierającej 3 konwolucyjne warstwy zawierające kolejno 8, 16 i 32 kanały. Następnie sieć składa się z 4 warstw głębokich zawierających kolejno 512, 256, 128 oraz 48 cech wyjściowych. Sieć neuronowa zwraca wektor własności składający się z 48 wartości, który normalizujemy do jedności. W celu wytrenowania sieci stosujemy funkcję kosztu *n-pair angular loss* [WZW+17], która jest bardziej efektywną alternatywą względem standardowej funkcji *triplet loss* [SKP15]. Funkcję kosztu formułujemy w taki sposób, żeby znaleźć wzorce miedzy poszczególnymi metodami ataków oraz próbkami autentycznymi, stąd celem treningu nie

jest jedynie klasyfikacja binarna (próbka autentyczna vs. spoofowa), ale rozróżnienie między kilkoma klasami, gdzie metody ataków są niezależnie etykietowane z wykorzystaniem ich parametrów. W celach wizualizacji zależności pomiędzy typami ataków, ostatnim krokiem jest redukcja wektora własności do dwóch głównych składowych wykorzystując do tego algorytm t-SNE [vdMH08].

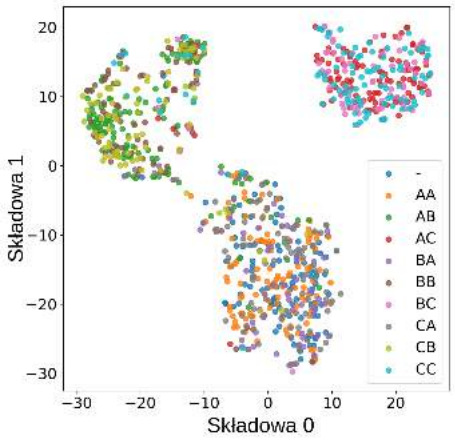
Wyniki analizy danych są przedstawione na rysunku 5.5. Wnioski w odniesieniu do trzech faz treningu przestawiamy w punktach poniżej:

- W początkowej fazie treningu zauważamy, że wektory własności są mocno zależne od parametru jakości urządzenia odtwarzającego, dodatkowo obserwujemy, że wektory reprezentujące próbki autentyczne są bliskie wektorom próbek spoofowych odtworzonych na urządzeniu o *perfekcyjnej* jakości akustycznej. Z kolei nie jest widoczna fragmentacja próbek po drugim parametrze – odległości adwersarza od mówcy. Wnioskujemy, że sieć po trzech epokach treningu jest w stanie rozróżnić między jakością urządzeń (przy czym próbki autentyczne reprezentuje podobnie do próbek odtworzonych urządzeniem o *perfekcyjnej* jakości) (rys. 5.5a).

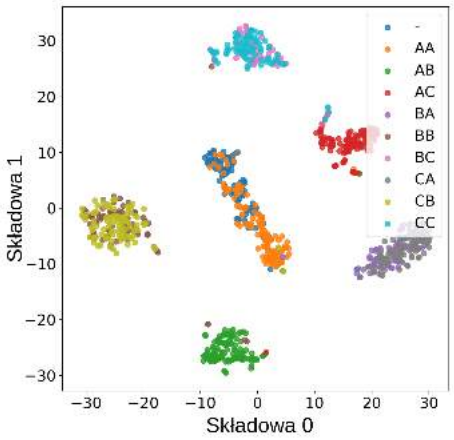
- Po 50 epokach treningu nadal jest widoczna zależność z jakością urządzenia odtwarzającego, ale dodatkowo zauważalny jest dodatkowy podział względem parametru odległości adwersarza od mówcy. Obserwujemy, że rozdzielenie następuje przy odległości wynoszącej 50 cm. Za to ciągle zauważamy, że próbki ataku AA (najbliższa odległość oraz najwyższa jakość urządzenia odtwarzającego) nie są odseparowane od próbek autentycznych (rys. 5.5b).

- Po 200 epokach treningu staje się widoczny niezależny klaster zawierający tylko próbki autentyczne, jednak ciągle nie ma widocznego rozdzielenia próbek nagranych z odległości w przedziałach $50-100$ cm oraz powyżej 100 cm (rys. 5.5c).

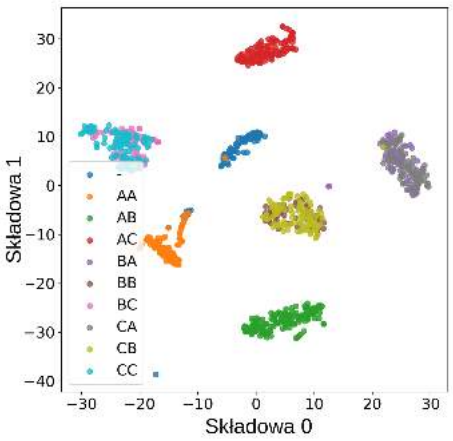
Analiza pokazuje, że separacja względem jakości urządzenia odtwarzającego jest łatwiejsza. Co istotne, mimo długiego treningu, nie udaje się całkowicie odseparować próbek względem odległości adwersarza od mówcy. Co więcej, wyniki potwierdzają dość intuicyjną tezę, że największe trudności w wykryciu sprawia atak przeprowadzony urządzeniem o najwyższej jakości odtwarzania oraz gdy adwersarz znajduje się najbliżej mówcy w momencie nagrywania. Analiza pozwala również wysnuć wniosek, że przeprowadzenie



(a) Po 3 epokach.



(b) Po 50 epokach.



(c) Po 200 epokach.

Rysunek 5.5: Wizualizacja wyników przeprowadzonej analizy danych. Wykresy przedstawiają wektory własności (ang. *embeddings*) po treningu sieci neuronowej przez 3, 50 oraz 200 epok. Wektory zostały dodatkowe zredukowane do dwóch głównych komponentów z wykorzystaniem algorytmu t-SNE [vdMH08]. Analizowane typy ataków zostały szczegółowo opisane w rozdziale 5.6.4.

ataku przez adwersarza z sukcesem jest bardziej prawdopodobne w przypadku wykorzystania urządzenia odtwarzającego wysokiej jakości, niż skupieniu się na zmniejszeniu odległości od mówcy, co w warunkach rzeczywistych jest bardziej ryzykowne dla adwersarza ze względu na większe ryzyko wykrycia próby przeprowadzenia ataku.

5.6.6 Attack-Out Cross-Validation

Problem nadmiernego dopasowania modelu do danych treningowych jest widoczny w rozwiązaniach *state-of-the-art* dedykowanych zagadnieniom wykrywania ataku typu podszywanie się. Przykładowo analizując wyniki ewaluacji modeli antyspoofingowych zgłoszonych do konkursu ASVspoof Challenge 2017 [KSD+17] można łatwo zauważyć, że tylko jeden model uzyskuje nieznacznie lepszy wynik EER na zbiorze ewaluacyjnym w porównaniu do zbioru treningowego, a pozostałe 19 modeli wypada znaczniej gorzej, często uzyskując ponad dwukrotnie gorszy wyniki na zbiorze ewaluacyjnym [LAR+19]. Co więcej, stały rozwój dziedziny przyczynia się również do wprowadzania coraz to nowszych metod podszywania się opartych np. na rozwiązaniach wykorzystujących techniki uczenia maszynowego, czy generowania mowy z tekstu (ang. *text-to-speech*, TTS). Stąd dobrze uogólniający model sieci neuronowej musi być odporny nie tylko na wcześniej niewidzianych mówców, czy artefakty wynikające ze specyfiki sensora, ale przede wszystkim na wcześniej niewidziane ataki.

W przypadku konkursu ASVspoof Challenge 2019, zbiory danych treningowych i testowych zawierają ograniczoną liczbę metod spoofingowych względem zbioru ewaluacyjnego, który zawiera dodatkowe metody, nieudostępnione podczas modelowania rozwiązania. Stąd wyniki liczone na zbiorze testowym (podczas modelowania) mogą źle odwzorowywać faktyczną wydajność modelu, ze względu na nadmierne dopasowanie do danych dostępnych podczas treningu. Przy czym model może dopasowywać się do danych nie tylko w klasycznym rozumieniu, tj. uzyskiwać wysoką wydajność tylko na danych użytych do treningu sieci, ale również wynikających z braku dostępu do danych dobrze „opisujących" problem spoofingu.

W celu otrzymania lepszej estymacji faktycznej wydajności modelu podczas procesu modelowania, wykorzystujemy opracowaną przez nas technikę Attack-Out Cross-Validation, która jest adaptacją standardowej k-krotnej walidacji krzyżowej (ang. *k-fold cross validation*) do potrzeb zagadnienia antyspoofingowego. Metoda polega na treningu i testowaniu modelu wykorzystując potrójną walidację krzyżową, gdzie dla każdego *foldu* dobieramy podzbiory treningowy, walidacyjny, testowy w taki sposób, żeby metody ataku o tych samych parametrach nie występowały w różnych podzbiorach w ramach tego samego foldu. Co istotne w tej metodzie ignorujemy podział zaproponowany przez twórców bazy danych na zbiory treningowy i testowy.

Dla każdego foldu wybieramy 4 metody spoofingowe do zbioru treningowego, 2 metody do zbioru walidacyjnego i 3 metody do zbioru testo-

Tablica 5.1: Konfiguracja metody Attack-Out Cross-Validation dla poszczególnych foldów. Dwójka $(A|B|C)^2$ oznacza identyfikatory ataków z bazy danych ASVspoof 2019 (dostęp fizyczny). Pierwsza litera nawiązuje do parametru odległości atakującego od mówcy, natomiast druga litera określa parametr jakości urządzenia odtwarzającego. Dokładny opis parametrów jest zwarty w rozdziale 5.6.4.

Fold	Zbiór treningowy	Zbiór walidacyjny	Zbiór testowy
0	BC, AA, CB, AB	BA, CA	BB, AC, CC
1	AB, CB, AC, BA	CC, BB	AA, BC, CA
2	CC, AA, CA, BB	BC, AC	CB, BA, AB

wego. Każdy wariant ataku spoofingowego występuje w podzbiorze testowym w jednym z foldów, co pozwala na pełną weryfikację w trakcie modelowania, które metody ataku sprawiają trudności, a przy tym zachowane jest założenie o uczeniu i testowaniu na rozłącznych zbiorach danych. Tablica 5.1 przedstawia dokładną konfigurację poszczególnych foldów. W przypadku próbek autentycznych, podział jest dokonywany w sposób losowy przy zachowaniu odpowiednich proporcji względem identyfikatorów mówców, które zostały dodane do metadanych wykorzystywanego zbioru danych. W przypadku braku odpowiednich metadanych, próbki autentyczne mogą zostać rozdzielone w sposób całkowicie losowy.

Attack-Out Cross-Validation jest wykorzystywane w celu wyboru architektury modelu o jak najwyższym potencjale uogólnienia względem ataków (niewidocznych podczas treningu oraz modelowania). Ostatecznie oba modele – bayesowska sieć neuronowa oraz LCNN są trenowane na pełnym zbiorze treningowym przy zachowaniu parametrów wybranych w trakcie procesu modelowania z wykorzystaniem procedury Attack-Out Cross-Validation.

5.6.7 Modele sieci neuronowych

W tym rozdziale opisujemy dwie sieci neuronowe wykorzystane do zbudowania metody do wykrywania ataków spoofingowych.

Bayesowskie sieci neuronowe

Pierwszy model jest oparty na bayesowskiej sieci neuronowej [KW14]. Wykorzystanie sieci bayesowskich w naszym rozwiązaniu jest częściowo motywowane możliwością wykorzystania ich do efektywnego treningu opierając się na stosunkowo małych zbiorach danych o ograniczonej liczbie wariantów ataku. Jedynie takie są dostępne dla zagadnienia zapobieganiu ata-

kowi spoofingowemu dla biometrii głosu. Empirycznie zostało dowiedzione, że sieci bayesowskie są mniej podatne na nadmierne dopasowanie [KW14, BCKW15]. To oznacza, że takie podejście odpowiada na jeden z najważniejszych problemów poruszanych w ramach konkursu ASVspoof Challenge 2019 [ASV19], czyli generalizację rozwiązania na szeroką gamę wariantów ataku spoofingowego. Naszym podstawowym celem jest stworzenie rozwiązania pozwalającego na określenie rzeczywistych niepewności w predykcji zwracanych przez sieci neuronowe. Empirycznie pokazano, że tradycyjne modele sieci neuronowych mają tendencję do zwracania wyników o wysokim współczynniku pewności (technicznie, największa wartość w zwróconym wektorze predykcji jest dużo większa od pozostałych wartości) nawet dla przypadków, które nie były dobrze reprezentowane w zbiorze treningowym, a co za tym idzie, co do których model nie powinien mieć pewności odnośnie właściwego wyniku [GSS15, BCKW15]. Co więcej, mimo zwrócenia wysokiej pewności dla wybranej etykiety, nie oznacza to, że ta jest prawdziwa – dostajemy błędną predykcję z wysoką pewnością odnośnie tej predykcji. Powoduje to duże trudności w doborze odpowiedniego poziomu akceptacji dla wyniku zwróconego przez model.

W modelu zastępujemy standardowe warstwy konwolucyjne oraz liniowe ich specjalnymi bayesowskimi modyfikacjami – warstwami *Flipout* [WVB+18]. Proces modelowania sieci neuronowej oparliśmy na zaproponowanej przez nas technice Attack-Out Cross-Validation (patrz rozdział 5.6.6).

Koncepcja bayesowskich sieciach neuronowych. Bayesowska sieć neuronowa z wagami W zwraca wartości interpretowane jako prawdopodobieństwo zdarzenia $P(\hat{y} \mid x, W)$, że próbka x jest atakiem spoofowym $\hat{y}$. Wagi sieci neuronowej W można zdefiniować jako listę zmiennych losowych w_i, czyli $W = \{w_1, w_2, w_3, \dots\}$. Każda zmienna w_i jest niezależna i ma początkowy rozkład normalny $w_i \sim \mathcal{N}_{\theta_i^{init}}$ z parametrem początkowym θ_i^{init}. Celem treningu bayesowskiej sieci neuronowej, nie jest ustalenie optymalnych wartości wag dla modelu, jak w przypadku standardowej optymalizacji sieci neuronowej, ale znalezienie *optymalnych* rozkładów prawdopodobieństw dla omawianych zmiennych losowych, tj. $w_i \sim \mathcal{N}_{\theta_i}$, gdzie θ_i to parametry znalezione podczas treningu modelu. W naszym przypadku, parametry θ są szukane poprzez minimalizację funkcji Evidence Lower Bound (ELBO).

Wynikiem bayesowskiej sieci neuronowej jest wartość oczekiwana wyników prawdopodobieństw liczona na przestrzeni zmiennych losowych W, którą aproksymujemy poprzez n sieci neuronowych z wylosowanymi wa-

gami W_i, czyli:

$$P(\hat{y} \mid x) \approx \frac{1}{n} \sum_{i=1}^{n} P(\hat{y} \mid x, W_i). \tag{5.20}$$

W przypadku budowy bayesowskiego modelu do zadania wykrywania ataku spoofingowego, które jest zadaniem binarnym, prawdopodobieństwo zwracane przez poszczególnie sieci jest z rozkładu Bernoulliego.

Warstwa Flipout. W warstwie Flipout zaprezentowanej w [WVB+18] stosuje się pseudo-niezależne perturbacje wag sieci neuronowej w ramach partii (ang. *batch*). Rozwiązanie wykorzystuje fakt, że wagi w bayesowskich sieciach neuronowych można reprezentować jako $W = \overline{W} + \Delta W$, gdzie $\overline{W}$ jest wektorem średnich wag (wprowadzonym jako parametr lub uczonym na bieżąco w trakcie treningu) oraz ΔW jest wektorem perturbacji. Przyjmijmy, że perturbacje dla poszczególnych wag są niezależne i ich rozkład jest symetryczny wokół wartości 0 oraz niech E będzie niezależnym losowym wektorem wartości ze zbioru $\{-1, 1\}$. Wtedy można zaobserwować, że $\Delta W = \widehat{\Delta W} \circ E$ ma identyczny rozkład do $\widehat{\Delta W}$. Rozwiązanie Flipout wykorzystuje ten fakt by użyć podstawowej perturbacji $\widehat{\Delta W}$, która jest współdzielona przez wszystkie próbki w partii i efektywnie policzyć indywidualną perturbację dla poszczególnych próbek, tj. $\Delta W_i = \widehat{\Delta W} \circ E_i$, gdzie ΔW_i oraz E_i odnoszą się do i-tej próbki w partii. Dodatkowo obliczenia są usprawniane na poziomie operacji macierzowych. Przyjmijmy, że $\mathbf{E}_i$ to losowa macierz wartości ze zbioru $\{-1, 1\}$, wtedy może zostać reprezentowana jako $\mathbf{E}_i = R_n S_n^\top$, gdzie R_n i S_n to wektory elementów ze zbioru $\{-1, 1\}$.

Dane wejściowe do sieci. Przed przekazaniem próbki audio, jako dane wejściowe od sieci neuronowej, jest ona odpowiednio przetwarzana z wykorzystaniem klasycznego podejścia z obszaru przetwarzania sygnałów dźwiękowych. Wejściem do sieci jest logarytmiczne skalowany spektrogram melowy (ang. *mel spectrogram*). Spektrogram jest parametryzowany przez 512 urny melowe (ang. *mel bins*) i jest tworzony z wykorzystaniem krótkookresowej transformaty Fouriera (ang. *Short Time Fourier Transform*, STFT) ze skokiem czasowym ustawionym na 512 jednostek oraz oknem Hanna o rozmiarze 2048 jednostek. Wartości umieszczone w spektrogramie to moduły widma częstotliwościowego zwróconego przez transformatę. Spektrogram jest powielany lub przycinany do 280 kolumn (na osi czasu).

Tablica 5.2: Architektura wykorzystywanej bayesowskiej sieci neuronowej.

Typ warstwy	Parametry
Conv2DFlipout	8 kanałów, 5×5 jądro konwolucji, padding, ReLU
MaxPooling2D	2×2 okno poolingu, 2×2 okno kroku, padding
BatchNorm	-
Conv2DFlipout	16 kanałów, 5×5 jądro konwolucji, padding, ReLU
MaxPooling2D	2×2 okno poolingu, 2×2 okno kroku, padding
BatchNorm	-
Conv2DFlipout	48 kanałów, 5×5 jądro konwolucji, padding, ReLU
MaxPooling2D	2×2 okno poolingu, 2×2 okno kroku, padding
GlobalMaxPool2D	-
BatchNorm	-
LinearFlipout	24 cech wyjściowych, ReLU
BatchNorm	-
LinearFlipout	1 cecha wyjściowa, Sigmoid

Oznacza to, że przy próbkowaniu źródła dźwięku równym 48000 Hz, analizujemy około 3 sekundy danej próbki i reprezentujemy jako macierz liczb rzeczywistych o rozmiarze 512×280.

Architektura modelu. Istotną cechą systemu do bezpieczniej autoryzacji biometrycznej jest szybkość jego działania. Musi on działać praktycznie w czasie rzeczywistym lub z niewielkim opóźnieniem. System CM, jako składowa systemu do autoryzacji, również musi spełniać warunek działania w czasie bliskim czasu rzeczywistego. Zamierzony cel osiągamy stosując architekturę sieci neuronowej, która jest stosunkowo płytka – składa się z trzech bayesowskich warstw konwolucyjnych Flipout i dwóch bayesowskich warstw gęstych Flipout oraz zawiera warstwy redukujące rozmiar tensora wynikowego na określonych wymiarach – w naszym przypadku wykorzystujemy warstwę poolingu maksymalizującego na wymiarach szerokości i wysokości. Ograniczamy też liczbę kanałów w warstwach konwolucyjnych, stosując kolejno 3, 16 i 48 kanałów. Pełna architektura wykorzystywanego modelu zawierająca specyfikację parametrów dla poszczególnych warstw została przedstawiona w tablicy 5.2.

Parametry treningu. W treningu bayesowskiej sieci neuronowej wykorzystujemy technikę nadpróbkowania (ang. *oversampling*). Sieć jest uczona z wykorzystaniem standardowego optymalizatora Adam [KB15] z krokiem

uczącym równym 0.001 oraz parametrami $\beta_1 = 0.9$ i $\beta_2 = 0.999$. Rozmiar partii treningowej wynosi 64.

Lekka konwolucyjna sieć neuronowa (LCNN)

Drugie wykorzystywane rozwiązanie opiera się na lekkiej konwolucyjnej sieci neuronowej (LCNN) [WHST18]. W tym przypadku, w samej architekturze sieci neuronowej nie wprowadzamy znaczących zmian względem prac wcześniejszych wykorzystujących sieci LCNN do wykrywania ataków typu podszywanie się na próbkach dźwiękowych [LNM+17]. Zmiany na poziomie architektury modelu dotyczą jedynie liczby warstw w sieci neuronowej oraz doboru parametrów dla poszczególnych warstw. Również cały proces wstępnego przetworzenia próbek wejściowych został opracowany zgodnie z [LNM+17]. Istotną zmianą wprowadzoną w naszym rozwiązaniu jest za to sposób treningu samego modelu, który realizujemy poprzez rozszerzoną procedurę wykorzystującą zaawansowane techniki regularyzacji oraz augmentacji danych wejściowych, mając przy tym na uwadze problem nadmiernego dopasowania modelu występujący w zagadnieniu rozpoznawania ataku spoofingowego oraz cel w postaci stworzenia rozwiązania, które będzie odpowiednio generalizowało się względem ataków.

Dane wejściowe do sieci. Podobnie jak w [LNM+17], dane wejściowe są przekazywane do modelu w postaci logarytmu ze spektrogramu mocy widma sygnału liczonego za pomocą krótkookresowej transformaty Fouriera (STFT). Transformata jest liczona ze skokiem czasowym równym 1728 oraz z wykorzystaniem okna Hamminga o rozmiarze 864. Wartości są normalizowane używając do tego tzw. *standaryzacji z-score*, a średnia oraz odchylenie standardowe konieczne do przeprowadzenia standaryzacji zostały policzone na całym zbiorze danych treningowych. Spektrogram jest powielany lub przycinany do 400 kolumn (na jednostce czasu), stąd analizie podlega około 14 sekund sygnału dźwiękowego, a jego reprezentacja to macierz o rozmiarze 864×400.

Architektura modelu. Architektura modelu jest oparta na konwolucyjnej sieci neuronowej, która obok dobrze znanych i powszechnie używanych warstw konwolucyjnych, normalizacji na partii (ang. *batch normalization*), poolingu maksymalizującego, czy warstw głębokich, zawiera dodatkowo warstwy *Max Feature Map* [WHST18], które są szczególnym przypadkiem warstwy *Maxout* [GWFM+13]. Ta warstwa pozwala na redukcję liczby kanałów z N do

Tablica 5.3: Architektura lekkiej konwolucyjnej sieci neuronowej (LCNN).

Typ warstwy	Parametry
Conv2D	32 kanałów, 5×5 jądro konwolucji
BatchNorm	-
MaxFeatureMap	-
MaxPooling2D	2×2 okno poolingu, 2×2 okno kroku
*ConvBlock**	32, 48
*ConvBlock**	48, 64
*ConvBlock**	64, 32
*ConvBlock**	32, 32
Linear	64 wartości wyjściowe
Dropout	0.7
MaxFeatureMap	
Linear	1 cecha wyjściowa, Sigmoid
**Struktura ConvBlock*(F_1, F_2):	
Conv2D	F_1 kanałów, 1×1 jądro konwolucji
BatchNorm	-
MaxFeatureMap	-
Conv2D	F_2 kanałów, 3×3 jądro konwolucji
BatchNorm	-
MaxFeatureMap	-
MaxPooling2D	2×2 okno poolingu, 2×2 okno kroku

$\frac{N}{2}$ w tensorze (przyjmijmy X) poprzez zastosowanie szybkiej i efektywnej operacji:

$$X'_{whi} \leftarrow max(X_{whi}, X_{wh(i+\frac{N}{2})}), \tag{5.21}$$

gdzie $w \in \{0, 1, \ldots, W-1\}$, $h \in \{0, 1, \ldots, H-1\}$ i $i \in \{0, 1, \ldots, \frac{N}{2} - 1\}$ są pozycjami na kolejnych wymiarach tensora (szerokość, wysokość, kanały). Po zastosowaniu tej operacji, wynikowy tensor X' ma rozmiar $W \times H \times \frac{N}{2}$. Podobnie jak w przypadku pierwszego modelu, zastosowanie warstwy *Max Feature Map* oraz warstwy poolingu maksymalizującego mają na celu przyspieszenie przetwarzania danych przez sieć neuronową poprzez redukcję wymiarowości tensora między warstwami ukrytymi.

Techniki regularyzacji. Mając na uwadze problem nadmiernego dopasowania danych dla zadania antyspoofingu (patrz rozdział 5.6.2), w przypadku tego modelu stosujemy szereg technik, które zwiększają odporność na omawiany problem oraz pozwalają na lepszą generalizację sieci neuronowej:

- *Uczenie wielozadaniowe (ang. multi-task learninig).* Podobnie jak w [SJH+18], definiujemy funkcję kosztu w taki sposób, aby oprócz uczenia się rozpoznawania właściwej etykiety (próbka spoofowa vs. autentyczna), model dodatkowo zdobywał wiedzę o dodatkowych metadanych zawartych w plikach dźwiękowych lub dodatkowo zgromadzonych przez anotatorów. W naszym przypadku dodatkowe metadane to odległość mówcy od mikrofonu, jakość urządzenia odtwarzającego, identyfikator mówcy oraz informacje o środowisku akustycznym – powierzchnia pomieszczenia, czas pogłosu, odległość mówcy od mikrofonu systemu ASV.

- *Technika Manifold Mixup* [VLB+19]. Metoda rozszerza jedną z standardowych technik augmentacji, tj. *Mixup* [ZCDLP18], polegającą na wykonaniu kombinacji liniowej dwóch tensorów – X i Y – zgodnie ze wzorem:

$$Z_{whc} \leftarrow pX_{whc} + (1-p)Y_{whc}, \tag{5.22}$$

  gdzie X_{whc}, Y_{whc} to elementy tensorów wejściowych (w pracy jest to spektrogram o 3 wymiarach – szerokość, wysokość, liczba kanałów) na pozycji whc, z kolei Z_{whc} to element tensora wynikowego (kombinacji liniowej) na pozycji whc. Wartość $p \in [0, 1]$ jest z rozkładu beta z parametrami α i β definiującymi tzw. rozkład U-kształtny (np. $\alpha = \beta = 0.5$). Tensor wynikowy jest rezultatem połączenia dwóch próbek, do których mogą być przypisane różne etykiety, stąd kombinację liniową z tym samym parametrem p liczy się również dla wektorów etykiet odpowiadającym tensorom wejściowym. Zauważmy, że po tej operacji wynikowy wektor etykiet nie musi już być kodowaniem typu „1 z n" (ang. *one-hot encoding*). Wykorzystywana w pracy technika Manifold Mixup dodatkowo rozszerza standardowe podejście poprzez wykonywanie kombinacji liniowej na tensorach pomiędzy warstwami ukrytymi (zamiast wyłącznie na tenorach wejściowych).

- *Augmentacja.* Wykorzystywana procedura jest inspirowana *techniką mieszania* z [IVW+18]. Próbki spoofowe krótsze niż 10 sekund są losowo, z prawdopodobieństwem równym 0.5, dopełnianie próbkami autentycznymi (zamiast np. wartościami równymi 0). Takie podejście zwiększa trudność w rozpoznaniu próbek spoofowych, ze względu na lokalne artefakty odpowiadające próbkom autentycznym.

- *Oversampling.* W celu wyrównania niezbalansowanego zbioru danych treningowych próbki autentyczne zostały powielone 10-krotnie.

Parametry treningu. Sieć LCNN jest trenowana przez 20 epok z wykorzystaniem algorytmu optymalizacyjnego Adam [KB15] z krokiem uczącym (ang. *learning rate*) równym 0.0001 oraz rozmiarem partii wynoszącym 8. Parametry rozkładu beta dla techniki *Manifold Mixup* zostały określone następująco $\alpha = \beta = 1$ (co jest równoważne z rozkładem jednostajnym ciągłym $\mathcal{U}_{[0,1]}$). Funkcja kosztu składa się z głównego komponentu binarnej klasyfikacji oraz innych klasyfikatorów zadaniowych związanych z uczeniem wielozadaniowym. Uczenie wszystkich zadań odbywa się z wykorzystaniem funkcji entropii krzyżowej (ang. *cross entropy*), przy czym w przypadku klasyfikatorów zadaniowych (w ramach uczenia wielozadaniowego) wyniki funkcji są mnożone przez wagę 0.1.

5.6.8 Eksperymenty i uzyskane wyniki

Bayesowska sieć neuronowa vs. standardowa

W celu sprawdzenia naszej intuicji o odporności bayesowskich sieci neuronowych na problem dopasowania [BCKW15, WVB+18], przeprowadzamy eksperyment, w którym podmieniamy warstwy *Flipout* na standardowe odpowiedniki i porównujemy wzajemnie oba modele w różnych scenariuszach testowych. Eksperyment przeprowadzamy z wykorzystaniem naszej metody Attack-Out Cross-Validation na konkursowym zbiorze danych ASVspoof 2019 (dostęp fizyczny, PA). Wyniki uzyskane na poszczególnych foldach wskazują, że sieć neuronowa ze standardowymi warstwami uzyskuje lepsze wyniki niż jej bayesowska wersja. Zauważyliśmy również, że sieć bayesowska szybciej dopasowuje się do zbioru treningowego, co jest nieoczekiwanym zachowaniem. W celu sprawdzenia, czy opisywane obserwacje występują niezależnie od ilości danych, powtarzamy eksperyment na mniejszym zbiorze treningowym zawierającym 20% losowo wybranych próbek. W tym przypadku, obserwujemy znaczącą poprawę efektywności podejścia opartego na warstwach *Flipout*, które wypada lepiej w dwóch na trzy przypadki względem podejścia z wykorzystaniem standardowych warstw. Dokładne wyniki eksperymentów zostały przedstawione w tablicy 5.4.

Uzyskane rezultaty świadczą, że bayesowskie sieci neuronowe rzeczywiście są bardziej odporne na dopasowanie w porównaniu do standardowych sieci neuronowych, jednakże zbiór danych wykorzystany w eksperymentach zdaje się być wystarczająco duży, żeby ten efekt stał się nieistotny dla rozwiązania. Mimo to mając w świadomości dobre teoretyczne, jak i empiryczne, wyniki badań nad poprawą generalizacji modeli głębokiego uczenia z wykorzystaniem podejścia bayesowskiego [BCKW15, WI20] przyjmu-

Tablica 5.4: Wyniki porównania rozwiązania opartego na bayesowskiej sieci neuronowej z *nie-bayesowską* siecią neuronową wykorzystującą standardowe warstwy. Foldy są dokładnie opisane w tablicy 5.1.

Typ	Pełny zbiór danych		20% zbioru danych	
	min-tDCF	EER (%)	min-tDCF	EER (%)
		Fold 0		
bayesowska	0.02561	0.00961	**0.04821**	**0.01536**
standardowa	**0.00912**	**0.00399**	0.05075	0.01667
		Fold 1		
bayesowska	0.00014	7.36×10^{-5}	**0.00029**	**0.00014**
standardowa	$\mathbf{7.36 \times 10^{-5}}$	$\mathbf{3.68 \times 10^{-5}}$	0.00066	0.00042
		Fold 2		
bayesowska	0.00072	0.00029	0.00315	0.0012
standardowa	**0.00018**	$\mathbf{9.09 \times 10^{-5}}$	**0.00261**	**0.00075**

Tablica 5.5: Wyniki „ścieżki ukrytej" w konkursie ASVspoof Challenge 2019.

Model	*„Ścieżka ukryta"*		*Ewaluacyjny*	
	min-tDCF	EER (%)	min-tDCF	EER (%)
[CXZ19]	0.5039	19.68	0.1470	0.52
Bayesowska NN (nasza)	**0.6379**	**32.64**	**0.1729**	**1.66**
[LCVD19]	0.7134	41.11	0.1666	1.29
[LNT+19]	0.7139	25.03	0.1610	1.23
[CWCL19]	0.8826	37.04	0.1598	1.08

jemy, że dla zagadnienia szczególnie podatnego na problem nadmiernego dopasowania (jakim jest wykrywanie ataku spoofingowego w próbce głosu) ważne jest zastosowanie wszelkich możliwych mechanizmów regularyzacyjnych, w tym użycie warstw bayesowskich w modelu.

Końcowe wyniki konkursu ASVspoof Challenge 2019 potwierdzają skuteczną regularyzację zaproponowanego przez nas podejścia opartego na bayesowskiej sieci neuronowej. Nasze rozwiązanie osiągnęło 6 wynik (liczony względem min-tDCF i EER) na pełnym zbiorze ewaluacyjnym, jednak w przypadku tzw. ścieżki ukrytej (ang. *hidden track*), w której wyselekcjonowano tylko ataki niedostępne w zbiorach treningowym oraz testowym, nasze rozwiązanie zajęło 2 miejsce (pod względem min-tDCF) i 3 (pod względem EER) [NWE+21]. Przy czym warto zauważyć, że wszystkie modele osiągają znacząco niższe wyniki ewaluacji na ścieżce ukrytej w stosunku do pełnego zbioru ewaluacyjnego. Wyniki zostały przedstawione w tablicy 5.5.

Wyniki dla zadania antyspoofingowego

Końcowe wyniki są uzyskiwane jako średnia ważona predykacji. Wagi dla modeli są otrzymywane z wykorzystaniem prostej procedury przeszukiwania siatkowego (ang. *grid search*), a finalny zestaw wag jest dobierany tak, żeby minimalizować średni EER po wszystkich foldach z metody Attack-Out Cross-Validation. Wagi są określane na podstawie predykcji wyliczonych dla zbioru testowego, tj. tego, który nie został wcześniej wykorzystany do treningu modeli (modele są trenowane na zbiorze treningowym, a parametry modeli są dobierane opierając się na metodzie Attack-Out Cross-Validation, czyli bazując na wynikach otrzymanych na zbiorze ewaluacyjnym). Finalnie wybrane wagi to 59% dla bayesowskiej sieci neuronowej oraz 41% dla LCNN. Bayesowska sieć neuronowa zwraca końcową predykcję jak średnia z 128 rezultatów cząstkowych.

Tablica 5.7 prezentuje wyniki naszych dwóch modeli oraz ich wagowego połączenia (ang. *ensemble*). Uzyskane wyniki wskazują, że połączone modele uzyskują 10-krotną względną poprawę w stosunku do modeli bazowych, zarówno dla metryki min-tDCF, jak i EER, na obu zbiorach – testowym oraz ewaluacyjnym. Nasz model plasuje się na 6 miejscu w konkursie ASVspoof Challenge 2019 dla zadania z dostępem fizycznym, przy czym cechuje się najmniejszą architekturą i największą szybkością działania (razem z metodą [LNT+19] opartą na trzech sieciach LCNN). Pozostałe rozwiązania opierają się na głębokiej architekturze ResNet [HZRS16], która jest dużo „cięższa" w sensie liczby parametrów, a przy tym czas wnioskowania jest znacząco dłuższy w porównaniu do modeli LCNN lub bayesowskich sieci neuronowych. W tablicy 5.6 przedstawiamy oszacowanie liczby operacji zmiennoprzecinkowych (ang. *floating point operations*, FLOPS) wykonywanych przy przetwarzaniu tensora wejściowego o tym samych rozmiarze w przypadku każdej z sieci.

Warto podkreślić, że wyniki połączonych dwóch modeli uzyskują 40% względną poprawę w stosunku do najlepszego pojedynczego modelu, co oznacza, że cechy wyuczone przez modele wzajemnie dopełniają się.

5.7 Podsumowanie

W tym rozdziale skoncentrowaliśmy się na uzyskaniu dobrego uogólnienia modelu na różnie metody ataków spoofingowych polegającym na powtórnym odtworzeniu próbki dźwiękowej. W celu osiągnięcia opisanego założenia, testowane modele zostały poddawane ewaluacji przy użyciu zapropo-

Tablica 5.6: Liczba operacji zmiennoprzecinkowych (FLOPS) wykonywanych podczas przetwarzania tensora wejściowego o rozmiarach $224 \times 224 \times 3$. Wyniki zostały policzone zgodnie z formułą zaproponowaną w [HZRS16]. Zwróćmy uwagę, że bayesowska sieć neuronowa wymaga kilkukrotnego uruchomienia modelu w celu otrzymania wyniku.

Model	FLOPS
ResNet-18	1.8×10^9
ResNet-34	3.6×10^9
ResNet-50	3.8×10^9
VGG-19	19×10^9
LCNN	0.270×10^9
Bayesowska NN	0.130×10^9

Tablica 5.7: Wyniki skuteczności modeli na zbiorze testowym oraz ewaluacyjnym.

Model	*Zbiór testowy*		*Zbiór ewaluacyjny*	
	min-tDCF	EER (%)	min-tDCF	EER (%)
Modele bazowe:				
LFCC-GMM	0.2554	11.96	0.3017	13.54
CQCC-GMM	0.1953	9.87	0.2454	11.04
Modele konkursowe:				
[CXZ19]	0.0049	0.20	0.0096	0.39
[LNT+19]	0.00054	0.30	0.0122	0.54
[LCVD19]	0.0032	0.13	0.0161	0.59
[CWCL19]	0.0065	0.24	0.0168	0.66
T24 [TWV+19]	0.0114	0.44	0.0215	0.77
Nasze modele:				
Bayesowska NN	0.0316	1.18	0.0433	1.66
LCNN	0.0516	2.46	0.0600	2.33
Ensemble	**0.0170**	**0.78**	**0.0219**	**0.88**

nowanej przez nas procedury Attack-Out Cross-Validation, która pozwala oszacować skuteczność sieci neuronowej w stosunku do ataków spoofingowych, do których model nie miał dostępu w czasie treningu. To pozwoliło nam wybrać dwa potencjalnie odporne modele – bayesowską konwolucyjną sieć neuronową o małej architekturze oraz sieć LCNN wytrenowaną z wykorzystaniem zestawu technik regularyzacji. Wyniki otrzymane w ramach walidacji krzyżowej pomogły nam również dostroić wagi dla obu modeli konieczne do wyliczenia wyników końcowych (*ensemble*). Zaprezentowane podejście może być wykorzystane również w innych zagadnieniach, w których

wydajność systemu na wcześniej nieznanych danych ma kluczowe znaczenie. Skuteczność naszej techniki walidacji krzyżowej została również potwierdzona podczas konkursu ASVspoof Challenge 2019. Przeprowadzono także dalszą analizę sieci bayesowskiej w celu zidentyfikowania czynników odpowiedzialnych za jej dobre działanie. Analiza ta wydaje się wskazywać, że kompaktowy rozmiar modelu może być korzystny z perspektywy generalizacji w scenariuszach przeciwdziałania atakom spoofingowym. Rodzi to również pytanie dotyczące wydajności bayesowskich sieci neuronowych – trenowanie tego modelu na ograniczonym zbiorze danych może być potencjalnie interesującym kierunkiem przyszłych prac.

Rozdział 6

Identyfikacja oparta na konturze dłoni

6.1 Wprowadzenie

Rozdział jest poświęcony zagadnieniu identyfikacji opartej na konturze dłoni z wykorzystaniem niskokosztowych urządzeń, takich jak skaner biurowy. W tym rozdziale prezentujemy metodę wykorzystującą klasyczne algorytmy z dziedziny widzenia komputerowego. W rozwiązaniu nie wykorzystujemy typowych algorytmów zaliczanych do uczenia maszynowego, w tym uczenia głębokiego, a skupiamy się na analizie geometrycznej stosunkowo złożonej figury, jaką jest kontur dłoni. Nasze rozwiązanie opiera się na liczeniu odległości między punktami charakterystycznymi zlokalizowanymi na konturze dłoni. Taka analiza pozwala na identyfikację osób z bardzo dużą skutecznością, a dodatkowo jest wydajna obliczeniowo. Rozdział w dużym stopniu został poświęcony rozwiązaniu opisanemu w naszej pracy [KPS18].

Biometria stała się powszechnym sposobem identyfikacji, z którego korzysta znaczna część populacji. Najbardziej znane oraz wdrożone na szeroką skalę metody biometryczne bazują na cechach twarzy, DNA, odcisku palca, głosie, czy też na własnoręcznym podpisie. Jednakże istnieje jeszcze szereg innych biometrii – kształt zębów, dźwięk serca, tęczówka, wzorzec użylenia ręki [USA14]. Biometria oparta na cechach konturu dłoni jest również jedną z mniej popularnych, jednak jej wykorzystanie niesie za sobą wiele korzyści, takich jak możliwość działania na niespecjalistycznym urządzeniu (w przypadku naszej pracy wykorzystujemy skaner biurowy), wysoką skuteczność, szybkość i wydajność obliczeniową oraz fakt, że cechy biometryczne konturu dłoni poszczególnych osób nie są rozpowszechnione publicznie (np. w serwisach społecznościowych), w przeciwieństwie do obrazów twarzy, próbek głosu, czy też informacji o tęczówce oka.

Nasze rozwiązanie na wejściu przyjmuje dane pobrane skanerem biuro-

wym. Jest wydajne czasowo, zarówno w fazie rejestracji, jak i weryfikacji. Algorytm analizuje standardowe cechy geometryczne ręki – długość i szerokość palców, powierzchnię dłoni, a także kilka nowych cech – krzywiznę palców, odległość od początku palca serdecznego do błon między palcami.

Nasza metoda została opracowana i przetestowana na zbiorze 60 dorosłych użytkowników w wieku między 20 a 55 lat. Najbardziej znaczące wyniki, które udało się nam uzyskać dla zadania weryfikacji to współczynnik fałszywszych akceptacji (ang. *False Acceptance Rate*, FAR) wynoszący 0.0% przy poziomie współczynnika fałszywych odrzuceń (ang. *False Rejection Rate*, FRR) równym 1.19% oraz współczynnik błędu (ang. *Equal Error Rate*, EER) wynoszący 0.59%. Wszystkie wyniki zostały obliczone dla przypadku, w którym podczas fazy rejestracji pobierane są 3 wzorce biometryczne ręki dla jednego użytkownika. Dodatkowo dla zadania identyfikacji został obliczony współczynnik identyfikacji (ang. *Identification Rate*, IR), który wyniósł 100.0%. Co istotne, użytkownicy nie byli w żaden sposób ograniczani oraz instruowani pod kątem pozycji ręki na skanerze, a skany były pobierane bez wymogu zdjęcia biżuterii. Co więcej, obsługa systemu opartego na naszym algorytmie nie wymaga wiedzy technicznej, a otrzymane wyniki świadczą o tym, że system może być wykorzystawszy w aplikacjach wymagających wysokiego poziomu bezpieczeństwa.

6.2 Wcześniejsze prace

Pomysł wyodrębnienia cech dłoni w celu identyfikacji osób został opublikowany pierwszy raz w 1971 roku w postaci dwóch patentów [Ern71, Mil71]. Pierwszy z nich koncentrował się na wykorzystaniu cech dłoni jako dodatkowej, obok wizerunku twarzy i podpisu, warstwy zabezpieczenia biometrycznego dowodów osobistych. Z kolei drugi patent dotyczył systemu biometrycznego bazującego na liczeniu długości palców.

Prace nad uwierzytelnianiem na podstawie biometrii ręki było kontynuowane w [JRP99], gdzie autorzy zaprezentowali metodę weryfikacji wykorzystującą 16 cech dłoni. W pracy użyto aparatu cyfrowego do zrobienia zdjęcia dłoni umieszczonej w specjalnej platformie z kołkami, pozwalającymi ustabilizować rękę i ujednolicić jej sposób ułożenia. Autorzy jako pierwsi przeprowadzili testy metody i pokazali wyniki bezpieczeństwa tego rodzaju systemów. W [SRSAGM00] zwiększono liczbę cech do 25, jednak wyniki testów były podobne do tych uzyskanych w [JRP99].

Nowe podejście, w którym zamiast specjalnej platformy użyto skanera biurowego zostało zaprezentowane w [WS02]. System w celu weryfika-

cji użytkownika mierzył szerokości oraz długości palców. Dodatkowe cechy oparte na promieniach największych okręgów wpisanych w wybrane miejsca wewnątrz konturu dłoni zostały wprowadzone w [BJKS04]. Podejście [OEB03] opierało się natomiast na rozszerzeniu metody ekstrakcji cech biometrycznych konturu dłoni o wielomianowe funkcje pozwalające na modelowanie kształtu palców oraz na wykorzystaniu odległości Mahalanobisa do stworzenia funkcji decyzyjnej. Kolejne rozwiązania bazujące na różnej reprezentacji cech biometrycznych dłoni zaprezentowano w [XXO05, ABEN06, ABEN09]. W pracy [AAVT08] przedstawiono system biometryczny bazujący na analizie konturów obu rąk. W pracy [HJZ+12] zaproponowano metodę opartą na analizie spójności kształtu. Pomysł reprezentacji ręki jako nieskierowanego grafu ważonego został przedstawiony w [AH15]. Z kolei w pracy [VL07] zaprezentowano trzy podejścia oparte na odległości euklidesowej, odległości Hamminga oraz podejściu wykorzystującym model uczenia maszynowego – Gaussian Mixture Model (GMM). Spośród trzech zaproponowanych metod najlepszą wydajność osiąga ostania. Kolejną pracą bazującą na uczeniu maszynowym jest [STKB14], w której wykorzystano maszynę wektorów nośnych (ang. *Support Vector Machine*, SVM). W artykule [FZ14] zaprezentowano rozwiązanie oparte na sieci neuronowej. Warto zauważyć, że wszystkie wspominane metody wykorzystujące algorytmy uczenia maszynowego nie przewyższają swoją wydajnością rozwiązań klasycznych. Przykładowo, w pracy [FZ14] zaraportowano, że $IR = 93.00\%$. Zauważalna cześć prac nt. właściwości cech biometrycznych dłoni koncertowała się na łączeniu różnych podejść [OCJL03, DGJ11].

Ostatnie prace dotyczące biometrii dłoni wskazują inny kierunek badań, w którym wykorzystany jest odcisk dłoni (zamiast samego konturu) lub układ naczyniowy ręki. Rozwiązania wykorzystujące odcisk dłoni zostały przedstawione w [ZGG16, AH16]. W pracy [RRM17] opisano system bazujący na minucjach odcisku ręki (jedno z podejść popularnych w rozwiązaniach bazujących na odcisku palca). Metody [KP10, PK13, GSG16] prezentują systemy identyfikacji oparte na geometrii dłoni i układzie naczyniowym ręki. Zauważmy jednak, że zastosowanie wzorca naczyniowego wymaga specjalnego urządzenia lub aparatu cyfrowego z funkcją podczerwieni. Szerszy przegląd podejścia unimodalnego i multimodalnego w identyfikacji biometrycznej można znaleźć w [OH16].

Alternatywnym rozwiązaniem prezentowanym w istniejących pracach jest ekstrakcja trójwymiarowej reprezentacji ręki za pomocą kamery 3D oraz opracowanie metody opartej na cechach trójwymiarowych (najczęściej wraz z cechami dwuwymiarowymi) [ZXZ16, SBD15].

Istotne wyniki wydajności dla poszczególnych metod zostały przedstawione w tablicy 6.6.

6.3 Źródło danych oraz sposób pobierania

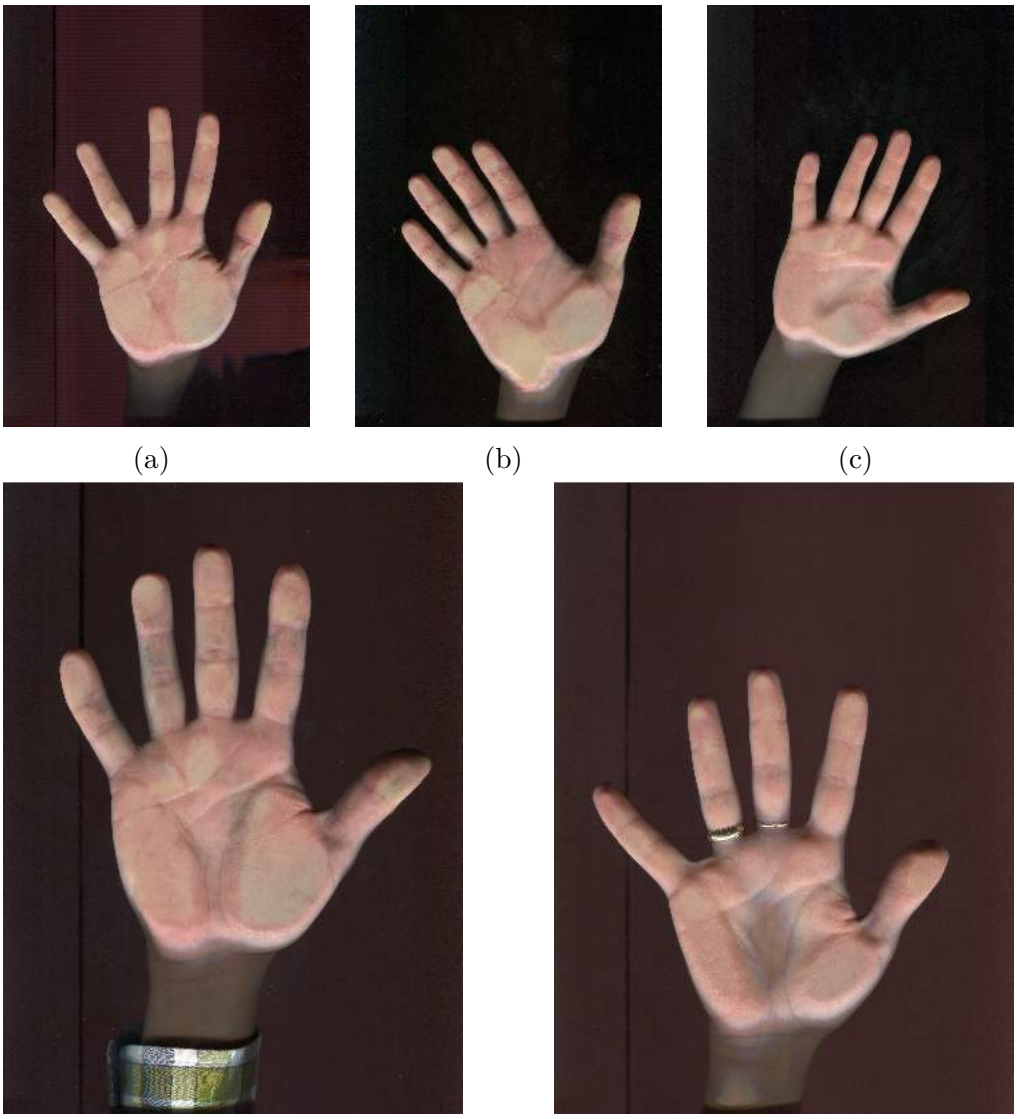
Do pobrania skanów dłoni wykorzystujemy ogólnodostępny skaner biurowy. Urządzenie zostało wyposażone w jeden z dwóch najpowszechniejszych typów układów optycznych, oparty na matrycy CCD (and. *charge-coupled device*). Obrazy skanów dłoni zostały pobrane w rozdzielczości 2500×3000 pikseli i 300 DPI. Zdjęcia o takich parametrach są możliwe do pobrania przez większość skanerów biurowych oraz nie wymagają znaczących zasobów pamięci do ich zapisu.

Przygotowana baza danych zawiera od 5 do 10 skanów prawej ręki 60 wolontariuszy. Najmłodsza osoba w chwili pobierania próbek była w wieku 20 lat, z kolei najstarsza miała 55 lat. Połowa wolontariuszy to kobiety. Zdecydowana większość osób nie miała przygotowania technicznego, ani nie deklarowała wykształcenia informatycznego. Co więcej, żadna z osób nie miała wcześniej świadomego kontaktu z biometrią geometrii ręki oraz dla wielu wolontariuszy było to pierwsze doświadczenie z biometrią w ogólności.

W celu pobrania pojedynczej próbki każda z osób została poproszona o wykonanie następujących czynności:

- położenie całej ręki oraz około 5 cm nadgarstka na szkle skanera,
- rozdzielenie palców,
- ułożenie ręki w sposób dowolny i dla niej komfortowy.

Skany były pobierane w publicznym miejscu, tj. na uniwersytecie. Osoby nie były proszone o zdjęcie obrączek, pierścionków czy bransoletek, w efekcie czego na części skanów widać biżuterię. Obrazy, na których dostrzegalne są kawałki ubrań również nie były odrzucane. Dodatkowo skany zawierają artefakty wynikające z zabrudzeń szyby skanera. W procesie tworzenia zbioru obrazów stawialiśmy duży nacisk, aby jak najlepiej odzwierciedlić realistyczne warunki, w których potencjalnie tego typy dane mógłby być pobierane. Co więcej, naszym celem było, aby dane były zebrane od osób, które nie były specjalnie w tym celu przeszkolone, a proces pobierania był dla nich jak najbardziej prosty i intuicyjny. Od części osób skany zostały ponownie pobrane po około 2 miesiącach od pierwszej próby, dzięki czemu wprowadzono dodatkowy efekt urealistyczniający proces pobierania próbek. Przykładowe skany dłoni zostały przedstawione na rysunku 6.1.



(a) (b) (c)

(d) (e)

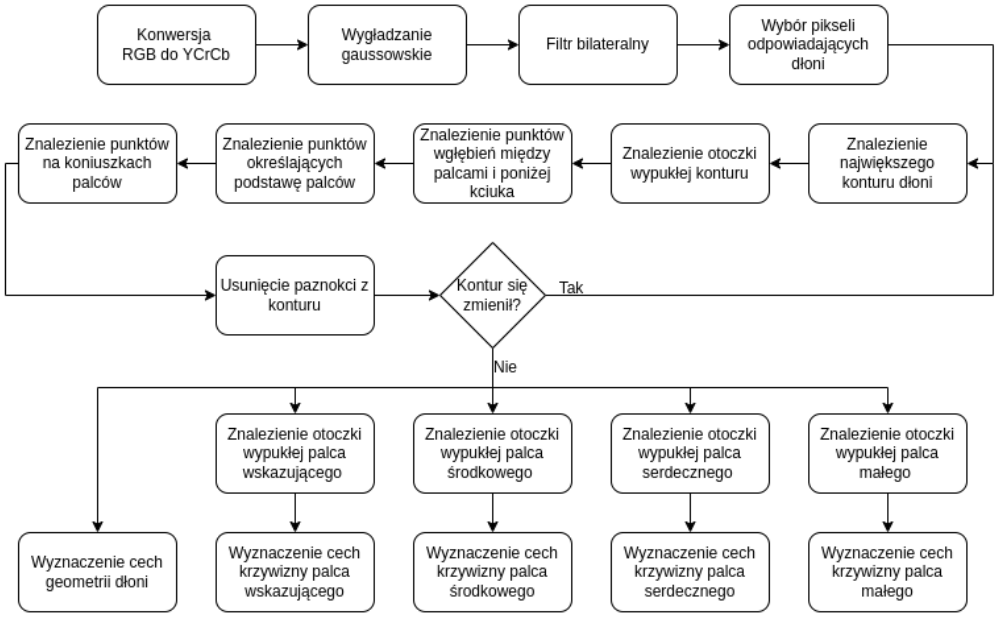
Rysunek 6.1: Przykładowe próbki pobrane do bazy danych. Zauważmy, że skany zawierające części ubrania (6.1d) oraz biżuterię (6.1e) nie zostały odrzucone.

6.4 Opis algorytmu ekstrakcji cech z konturu dłoni

6.4.1 Ekstrakcja punktów charakterystycznych dłoni

Nasz protokół można podzielić na trzy głównie składowe: wstępne przetwarzanie obrazu, wybór punktów charakterystycznych dłoni oraz ekstrakcja cech biometrycznych. Wysokopoziomowy zamysł naszego algorytmu został

przedstawiony na rysunku 6.2 w formie schematu blokowego, a poszczególne kroki zostały zobrazowane na rysunku 6.3.



Rysunek 6.2: Schemat blokowy przedstawiający kolejne kroki algorytmu do wyznaczania cech geometrycznych konturu dłoni.

Wstępne przetwarzanie skanu dłoni

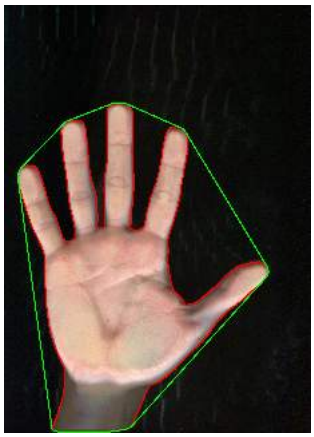
Pierwszym krokiem w fazie przetwarzania skanu dłoni jest zaaplikowanie rozmycia gaussowskiego w celu usunięcia małych smug i innych artefaktów spowodowanych zabrudzeniem skanera. Następnie wykorzystujemy filtr bilateralny, aby uwypuklić krawędź ręki.

W kolejnym kroku wydzielamy obszar obrazu reprezentujący rękę. Wybieramy piksele mieszczące się w zakresie barw odpowiadających kolorowi skóry, który został określony w [GS06]. W celu użycia metody, najpierw dokonujemy konwersji przestrzeni barw z RGB do YCbCr (rysunek 6.3a), po czym wybieramy te piksele, które mieszczą się w zakresie barw określających kolor skóry (rysunek 6.3b):

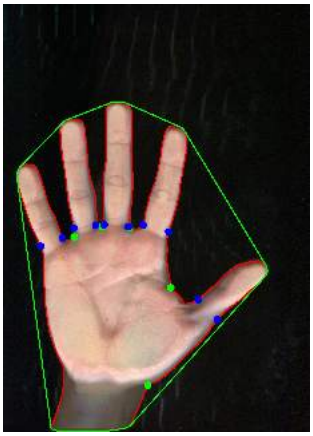
- $50 \leq Y \leq 170$,
- $77 \leq Cb \leq 127$,
- $110 \leq Cr \leq 173$.

(a) Konwersja z przestrzeni barw RGB do YCbCr.

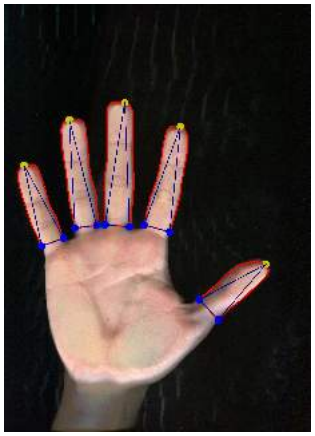
(b) Wybór pikseli interpretowanych jako dłoń.



(c) Znalezienie konturu dłoni oraz otoczki wypukłej dla danego konturu.



(d) Obliczenie punktów wgłębień oraz najkrótszych szerokości palców (początek palców).



(e) Wykrycie punktów reprezentujcych koniuszek palca poprzez wpisanie trójkąta o największej powierzchni w palce.

(f) Określenie precyzyjnych pozycji punktu wgłębienia pod kciukiem.

Rysunek 6.3: Wizualizacja kolejnych kroków pozwalających określić pozycję dłoni oraz palców na obrazie. Oznaczamy punkty wgłębień poprzez różowe oraz zielone kropki. Niebieskie kropki określają początek palców. Żółte punkty są oznaczeniem dla koniuszków palców.

Takie podejście jest możliwe ze względu na kolor tła zeskanowanego obrazu, które jest czarne, stąd prosty mechanizm selekcji oparty na zakresie kolorów skóry jest wystarczająco dokładny.

Wyznaczenie punktów charakterystycznych dłoni

Kolejnym krokiem jest obliczenie konturu dłoni oraz otoczki wypukłej dla danego konturu (rysunek 6.3c). Opierając się na znalezionych obiektach wyznaczamy część punktów charakterystycznych dłoni (punkty wgłębień) poprzez wybranie najbardziej oddalonych punktów od otoczki wypukłej leżących na konturze dłoni między dwoma kolejnymi palcami. Każdy punkt liczymy dla danego odcinka otoczki wypukłej oraz dla części konturu znajdującego się między jego końcami, stąd proces liczenia jest szybki, tzn. ma złożoność liniową względem punktów części konturu. Określamy 4 punkty wgłębień, które znajdują się pomiędzy palcami.

W kolejnym etapie procedury wyznaczania punktów charakterystycznych na konturze dłoni dla każdego palca szukamy dwóch punktów definiujących jego podstawę (rysunek 6.3d). Te punkty są szukane poprzez policzenie najkrótszej szerokości palca przy śródręczu. W tym kroku algorytm rozpoczyna swoje działanie od obliczenia odległości między punktami wgłębienia okalającymi dany palec, następnie szuka punktu na konturze, który jest bliżej lewego punktu wgłębienia, rozpoczynając obliczenia od prawego punktu wgłębienia. Znaleziony punkt przyjmujemy jako położenie prawego końca najkrótszego odcinka. Następnie algorytm przyjmuje nowo znaleziony punkt jako punkt odniesienia oraz szuka na konturze punktu dla niego bliższego, zaczynając od lewego punktu wgłębienia, ustanawiając lewy koniec odcinka w znalezionym punkcie. Algorytm powtarza swoje działanie jeden raz, rozpoczynając je od nowo wybranych końców odcinka.

Mając wyznaczone punkty podstawy palców, wykorzystujemy je do wybrania punktów konturu należących do poszczególnych palców i wyznaczamy punkty określające koniuszki palców (rysunek 6.3e). Metoda wpisuje w figurę danego palca trójkąt o największej powierzchni, przy założeniu, że jednym z boków trójkąta jest odcinek wyznaczony przez punkty podstawy palców. Trzeci punkt trójkąta jest interpretowany jako położenie koniuszka danego palca.

Dodatkowo wyznaczamy jeden punkt wgłębienia pomiędzy przedramieniem i kciukiem, w okolicy nadgarstka (rysunek 6.3f). Na pozycję punktu pod kciukiem wpływa położenie ręki na skanerze oraz odległość między kciukiem a palcem wskazującym (położenie kciuka względem całej dłoni), stąd szukamy pozycji, która jest blisko nadgarstka. Algorytm szukający dokładnego położenia tego punktu swoje działanie opiera na analizie części konturu oraz odcinka, które w kolejnych krokach odpowiednio ogranicza w celu znalezienia najlepszego punktu wgłębienia pod kciukiem. Dokładny

algorytm działa w następujący sposób:

1. wyznaczamy punkt wgłębienia tak samo jak w przypadku pozostałych punktów – jako najodleglejszy punkt na konturze względem odcinka otoczki wypukłej,

2. sprawdzamy, czy jeden z końców odcinka otoczki wypukłej jest odpowiednio blisko koniuszka kciuka, i jeśli jest to prawda, obliczamy nowy punkt końcowy odcinka jako środek pomiędzy punktem wgłębienia a aktualnym końcem odcinka (patrz rysunek 6.4b),

3. analizujemy drugi koniec odcinka na przedramieniu i ustanawiamy nowy punkt końca odcinka jako środek pomiędzy punktem wgłębienia oraz dotychczasowym punktem końca odcinka,

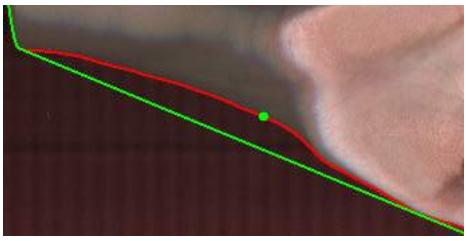
4. po operacji w 2. kroku na nowo liczymy punkt wgłębienia dla nowych punktów końcowych odcinka oraz odpowiednio pomniejszonej części konturu dłoni (patrz rysunek 6.4c),

5. kroki 3–4 powtarzamy 3 razy, chyba że punkt wgłębienia nie zmienia położenia, wtedy procedura jest przerywana,

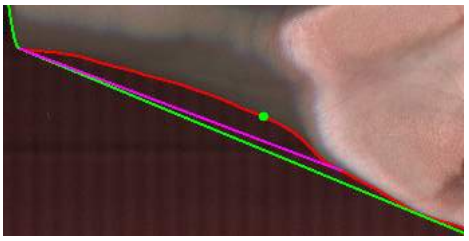
6. kroki 2–5 powtarzamy dwukrotnie.

Ze względu na ustanowienie stałej liczby kroków procedura wykonuje się w czasie stałym, a przy tym zapewnia odpowiednią precyzję. Główne kroki algorytmu wyboru punktu wgłębienia pod kciukiem zostały zaprezentowane na rysunku 6.4.

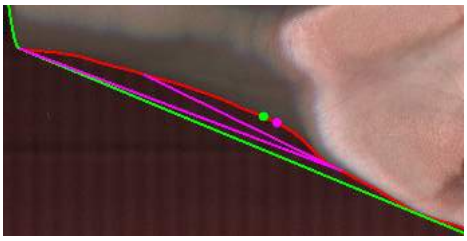
W celu eliminacji potencjalnych błędów spowodowanych długimi paznokciami, aplikujemy kolejną procedurę, która dokonuje segmentacji paznokci i je usuwa z konturu dłoni. Algorytm rozpoczynamy od policzenia otoczki wypukłej dla otoczenia koniuszków palców oraz liczymy punkty wgłębienia około koniuszka palca wykorzystując tę samą metodę co poprzednio, czyli znajdując najodleglejszy punkt na odcinku konturu dłoni od otoczki wypukłej. W przypadku, w którym znajdujemy dwa punkty wgłębienia, których odległość od odcinka otoczki wypukłej jest odpowiednio duża (tzn. ponad 10% długości odcinka) oraz odległość obu punktów od punktu koniuszka palca jest zbliżona, to interpretujemy ten fakt jako artefakt spowodowany długimi paznokciami i usuwamy punkty konturu znajdujące się między dwoma znalezionymi punktami wgłębienia. Kroki procedury zostały przedstawione na rysunku 6.5. Jeśli kontur się zmienił (z konturu został usunięty przynajmniej jeden paznokieć), wtedy powtarzamy proces



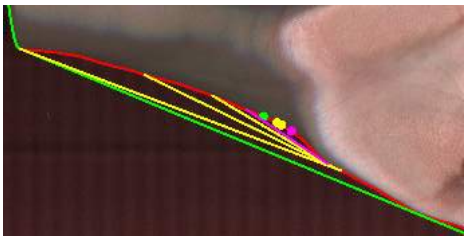
(a) Punkt wgłębienia pod kciukiem obliczony jako najodleglejszy punkt na cześci konturu względem odcinka otoczki wypukłej.



(b) Wyznaczenie nowego punktu końca odcinka przy kciuku wykorzystując do tego wyznaczony punkt wgłębienia oraz odległość aktualnego końca odcinka od koniuszka kciuka.



(c) Wyznaczenie nowego punktu końca odcinka przy przedramieniu jako średnia aktualnego punktu końca odcinka oraz punktu wgłębienia.



(d) Określenie ostatecznego położenia punktu wgłębienia pod kciukiem po przejściu wszystkich kroków algorytmu.

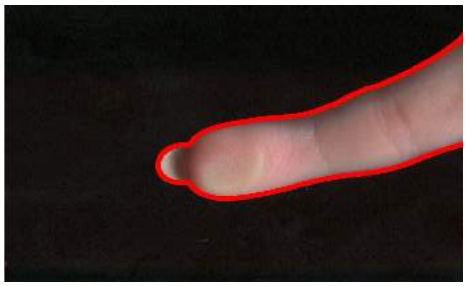
Rysunek 6.4: Obrazy przedstawiają główne kroki algorytmu określania pozycji punktu wgłębienia pod kciukiem.

liczenia punktów charakterystycznych rozpoczynając od obliczenia ponownie otoczki wypukłej dla całej ręki.

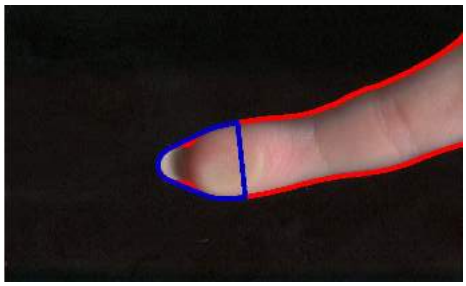
Cały proces kończymy wyznaczeniem 20 punktów charakterystycznych, a dokładniej 5 punktów wgłębień, 10 punktów podstawy palców oraz 5 punktów koniuszków palców. Te punkty oraz kontur dłoni są wystarczające do wyznaczenia cech biometrycznych dla luźno usytuowanej dłoni, przez to nie jest wymagane wykorzystanie dodatkowego sprzętu, takiego jak specjalistyczna platforma do usytuowania ręki [MKCK01] lub dedykowane kołki (ang. *pegs*) [VL07].

Wyznaczenie cech biometrycznych geometrii dłoni

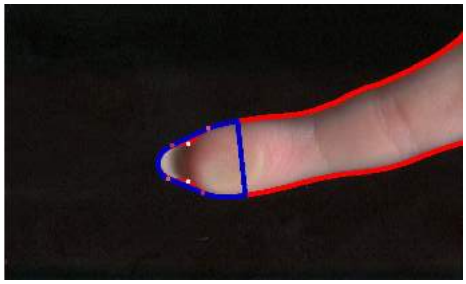
W celu wyboru odpowiednich cech biometrycznych geometrii dłoni przeanalizowaliśmy szereg konturów dłoni różnych osób i określiliśmy różnice między nimi. W konsekwencji zidentyfikowaliśmy cechy, które znacząco



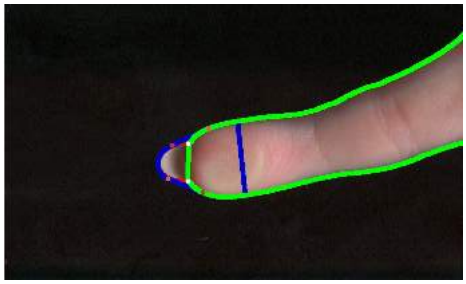
(a) Kontur całego palca zawierający paznokieć.



(b) Obliczenie otoczki wypukłej dla górnego fragmentu palca w okolicy jego koniuszka.



(c) Obliczenie wgłębień dla otoczki wypukłej fragmentu palca.

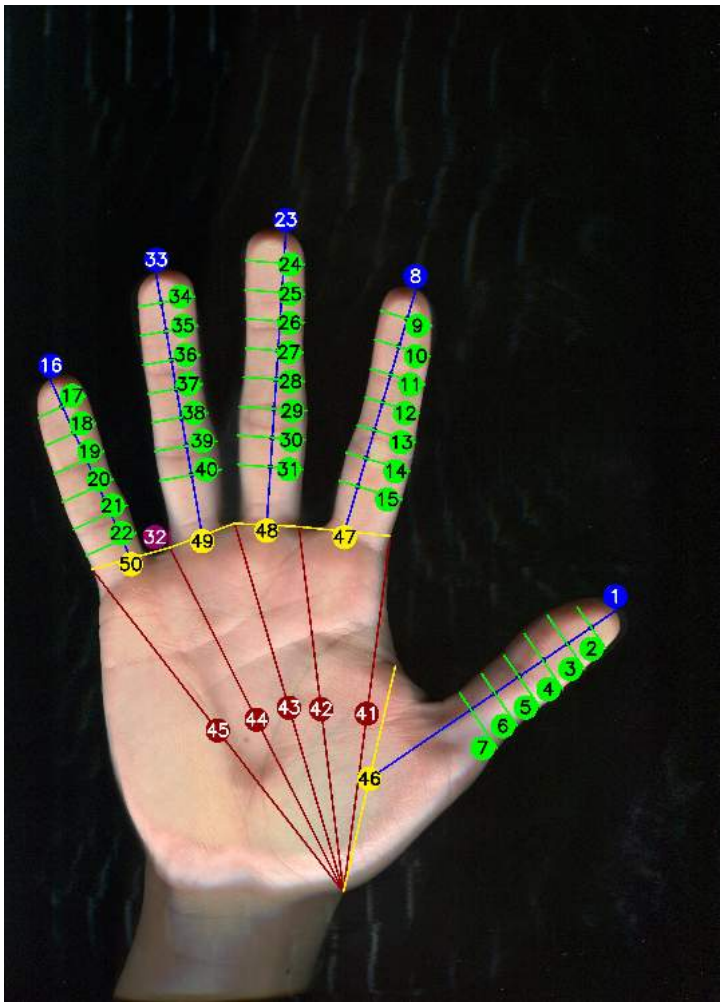


(d) Usunięcie części kontura zintepretowanego jako paznokieć.

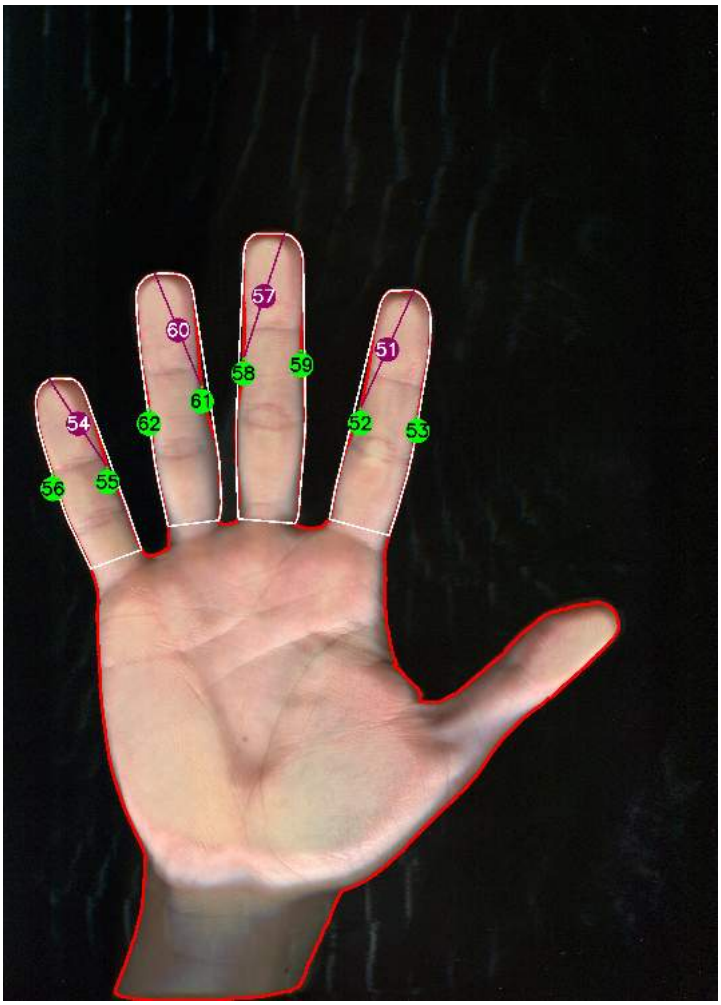
Rysunek 6.5: Obrazy przedstawiają główne kroki algorytmu usuwania długich paznokci.

różnią się zależnie od osoby (np. odległość podstawy palca serdecznego od błony zlokalizowanej między tym palcem oraz palcem małym), a przy tym są możliwe do dokładnego wyznaczenia bazując na 20 punktach charakterystycznych obliczonych przez nasz algorytm. Ostatecznie wyznaczyliśmy 50 cech, które pozwalają na dokładną identyfikację osoby na podstawie konturu jej dłoni. W zbiorze tych cech znajduje się 6 szerokości kciuka mierzonych na różniej wysokości oraz od 6 do 8 szerokości dla pozostałych czterech palców, długości tych palców, 5 odległości między kolejnymi punktami wgłębienia, 5 odległości śródręcza oraz 1 odległość między błoną a palcem serdecznym. Wszystkie wartości są liczone wykorzystując do tego jedynie wyznaczone punkty charakterystyczne oraz kontur dłoni.

Co więcej, znajdujemy dodatkowe 12 cech, które definiują krzywiznę palców. Krzywizna dla czterech palców (poza kciukiem) jest liczona na podstawie konturu oraz otoczki wypukłej obliczonej dla każdego z palców z osobna (od podstawy do koniuszka). Jako cechy krzywizny przyjmujemy odległości dwóch punktów wgłębienia (największych co do wartości) od otoczki oraz odległość punktu najodleglejszego do koniuszka palca (od-



Rysunek 6.6: Przedstawienie 50 standardowych cech biometrycznych geometrii dłoni wyliczonych naszą metodą:
1 – długość kciuka,
2 do 7 – szerokości kciuka,
8 – długość palca wskazującego,
9 do 15 – szerokości palca wskazującego,
16 – długość palca małego,
17 do 22 – szerokości palca małego,
23 – długość palca środkowego,
24 do 31 – szerokości palca środkowego,
32 – odległość od błony,
33 – długość palca serdecznego,
34 do 40 – szerokości palca serdecznego,
41 do 45 – długości śródręcza,
46 do 50 – odległości między punktami wgłębienia.



Rysunek 6.7: Przedstawienie 12 cech biometrycznych określających krzywiznę palców wyliczonych naszą metodą:
51, 54, 57, 60 – odległości między punktem wgłębienia a koniuszkiem palca,
52, 53, 55, 56, 58, 59, 61, 62 – największa odległość punktu wgłębienia od otoczki wypukłej palca.

ległość od drugiego najodleglejszego punktu wydaje się być informacją redundantną, przez co została pominięta).

Wszystkie opisane cechy zostały przedstawione na rysunku 6.6 (cechy standardowe) oraz rysunku 6.7 (cechy krzywizny palców).

6.4.2 Rejestracja i reprezentacja danych

W trakcie fazy rejestracji pobieramy od 1 do 3 skanów każdego użytkownika w celu stworzenia wzorca biometrycznego. System pobiera skany (użytkownik powinien dla każdego ze skanów na nowo ułożyć rękę na szkle skanera), dokonuje przetwarzania pobranego obrazu i określa 62 cechy biometryczne. W przypadku pobierania więcej niż jednego skanu, każda z cech jest uśredniania i w bazie danych jest zapisywany jedynie wektor uśrednionych 62 cech biometrycznych. Wzorzec do zapisu wymaga zaledwie 62 wartości zmiennoprzecinkowych, stąd w bazie danych zajmuje mniej niż pół kilobajta pamięci.

6.5 Weryfikacja i identyfikacja oraz analiza cech

6.5.1 Proces weryfikacji i identyfikacji

Po wykonaniu skanu dłoni następuje proces weryfikacji (lub identyfikacji), w którym algorytm liczy podobieństwo między profilem zadeklarowanego użytkownika (w przypadku identyfikacji między wszystkimi profilami) a pobraną próbką. Poniżej prezentujemy sposób obliczania podobieństwa (odległości) oraz ustalenia progu akceptacji.

W celu policzenia odległości między dwoma wektorami stosujemy następujący wzór:

$$\sigma_{ERR} = \frac{2\sum_{i=1}^{62} w_i|q_i - r_i|}{\sum_{i=1}^{62} w_i(q_i + r_i)}, \tag{6.1}$$

gdzie:
$Q = (q_1, q_2, \ldots, q_{62})$ – wzorzec, wektor uśrednionych cech zapisanych w bazie danych oraz $q_i = \frac{1}{k}\sum_{j=1}^{k} q_i^j$, gdzie q_i^j jest i-tą cechą dla j-tej próbki pobranej do bazy danych,
$R = (r_1, r_2, \ldots, r_{62})$ – wektor cech biometrycznych geometrii dłoni obliczonych ze skanu pobranego od użytkownika,
$W = (w_1, w_2, \ldots, w_{62})$ – wektor wag cech (patrz rozdział 6.5.2).

Przedstawioną formułę można rozumieć jako średnią różnicę między poszczególnymi cechami biometrycznymi konturu dłoni, która jest odpowied-

nio znormalizowana poprzez rozmiar ręki liczony jako średnia suma wartości cech biometrycznych pobranej próbki oraz wzorca. Intuicyjnie przyjmujemy, że dla większych rąk odnotujemy większe różnice w wartościach poszczególnych cech.

W przypadku zadania weryfikacji formuła jest liczona tylko raz dla pobranej próbki oraz wzorca zadeklarowanego użytkownika. W tym przypadku użytkownik w ramach procesu deklaruje swoją tożsamość na podstawie, której wybierany jest odpowiedni wzorzec z bazy danych. Użytkownik przechodzi poprawnie autoryzację (jest akceptowany) jeśli wartość σ_{ERR} jest mniejsza od zadeklarowanego progu akceptacji t. W celu identyfikacji użytkownika obliczony wektor cech porównujemy ze wszystkimi wzorcami znajdującymi się w bazie danych. Na końcu wybieramy ten, dla którego obliczona wartość σ_{ERR} jest najmniejsza. W procesie identyfikacji również wykorzystujemy próg akceptacji, na podstawie którego sprawdzamy, czy otrzymana najmniejsza wartość σ_{ERR} jest mniejsza od progu akceptacji, w przeciwnym razie użytkownik jest odrzucany i uznawany za niedopasowany do żadnego wzorca w bazie.

Próg akceptacji został wybrany eksperymentalnie. Założyliśmy, że wartość FRR ma wynosić około 1% i do tej wartości dobieraliśmy próg w taki sposób, żeby uzyskać jak najmniejszą wartość współczynnika FAR.

6.5.2 Dobór wag cech biometrycznych

W celu określenia, które cechy biometryczne konturu dłoni są najbardziej wyróżniające w populacji przeprowadziliśmy testy statystyczne wartości tych cech. Policzyliśmy histogramy odległości (różnic) między wartościami cech konturu dłoni dla próbek pobranych od tych samych oraz różnych osób. Dla każdej cechy konturu dłoni została również policzona średnia oraz wartość oczekiwana odległości pomiędzy tymi samymi osobami oraz z uwzględnieniem całej populacji (wartość różnicy cech między dwiema różnymi osobami). Przykładowe wyniki są przedstawione w tablicy 6.1 oraz na rysunku 6.8 w formie histogramów.

Łatwo zauważyć, że cechy o dużym znaczeniu charakteryzują się niskimi wartościami średniej oraz odchylenia standardowego dla odległości liczonych dla tych samych osób oraz wysoką średnią wartości odległości cech pobranych od różnych osób. Z kolei wartość odchylenia standardowego dla odległości cech różnych osób powinna być możliwe najmniejsza. Wysoka średnia odległość cechy konturu dłoni oraz niska wartość oczekiwana dla próbek pobranych od różnych osób pozwala stwierdzić, że dana

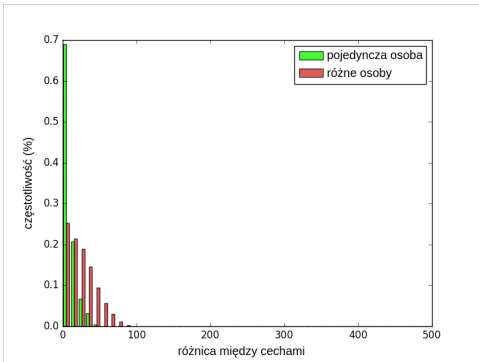
Tablica 6.1: Przykładowe dane statystyczne policzone dla czterech długości palców. Tablica przedstawia przykłady dwóch znaczących cech oraz dwóch o bardzo niskim znaczeniu (w szczególności długość palca małego). Ostatnia kolumna zawiera referencję do wykresów histogramów analizowanych cech.

Cechy	*Ręce tych samych osób*		*Ręce różnych osób*		Hist.
	Średnia	Odchylenie standardowe	Średnia	Odchylenie standardowe	
Szerokość kciuka	8.964	8.720	25.207	18.360	6.8a
Szerokość palca wskazującego	2.934	3.001	20.831	16.538	6.8b
Szerokość palca serdecznego	10.186	33.848	80.125	86.201	6.8c
Szerokość palca małego	2.015	20.240	2.459	22.342	6.8d

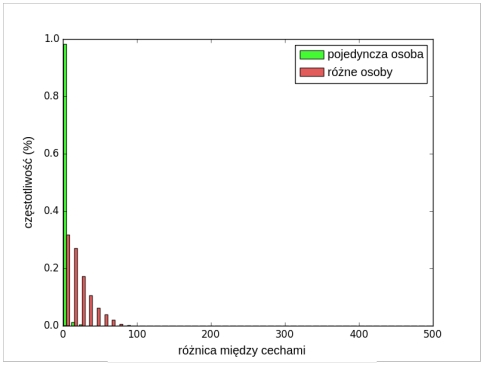
cecha z dużym prawdopodobieństwem pozwoli na rozróżnienie analizowanych osób, ze względu na fakt, że wartość odległości dla poszczególnych przypadków będzie się niewiele różniła od wartości średniej, która sama w sobie jest wysoka. Ze względu na specyfikę pojedynczych cech konturu dłoni, które poddajemy analizie, nie oczekujemy, że odchylenie standardowe będzie co do wartości bardzo małe, jednak przyjmujemy, że powinno być możliwe najmniejsze. W celu określenia, które cechy mają wysokie znaczenie dla przeprowadzenia poprawnej weryfikacji biometrycznej opracowaliśmy współczynnik znaczenia cechy:

$$e = \frac{\mu_d}{1 + \mu_s \sigma_s \sigma_d}, \tag{6.2}$$

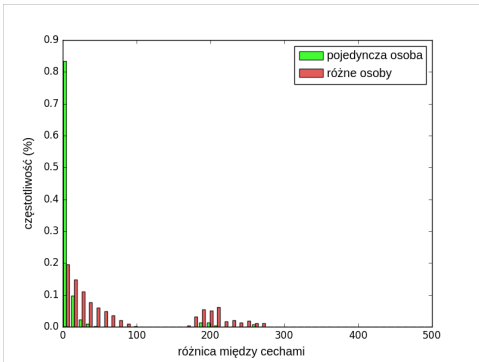
gdzie:
μ_d – średnia odległości cech pobranych od różnych osób,
μ_s – średnia odległości cech pobranych od tych samych osób,
σ_d – odchylenie standardowe odległości cech pobranych od różnych osób,
σ_s – odchylenie standardowe odległości cech pobranych od tych samych osób.
Mając wynik współczynnika jesteśmy w stanie określić znaczenie poszczególnych cech. Im wyższa wartość współczynnika, tym dana cecha ma większe znaczenie z perspektywy analizy biometrycznej. Uzyskane wartości współ-



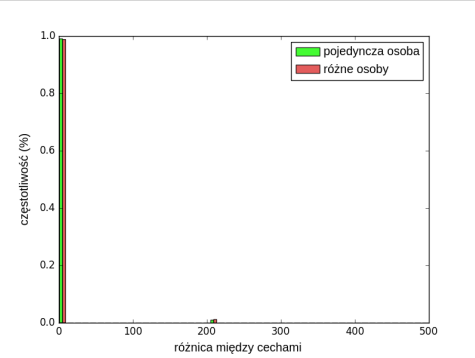
(a) Histogram dla cechy szerokości kciuka.



(b) Histogram dla cechy szerokości palca wskazującego. Większość odległości pomiędzy wartościami dla tych samych osób jest bliskie 0.



(c) Historgram dla cechy szerokości palca serdecznego. Obserwujemy relatywnie duże wartości różnić pomiędzy tymi samymi osobami.



(d) Histogram dla cechy szerokości palca małego. Wszystkie odległości, niezależnie czy wartośc jest liczona dla tych samych czy różnych osób, są bliskie 0.

Rysunek 6.8: Histogramy przedstawiają rozkłady empiryczne wartości odległości liczone pomiędzy próbkami pobranymi od tych samych (kolor zielony) oraz różnych (kolor czerwony) osób.

czynnika e powyżej 0.1 (średnia i odchylenie standardowe dla próbek pobranych od tych samych osób poniżej 4) określamy jako te o wysokim znaczeniu. Współczynnik e powyżej 0.01 to cechy, które mają pozytywny wpływ na wyniki weryfikacji oraz identyfikacji. Z kolei empirycznie sprawdziliśmy, że wartość współczynnika e poniżej 0.005 oznacza, że dana cecha wpływa negatywnie na wyniki analizy biometrycznej.

Zaprezentowana wyżej analiza pozwoliła na wybranie 19 cech biometrycznych, które charakteryzują się wysokim współczynnikiem e. Wszystkie wybrane cechy to szerokości palców (bez kciuka). Po przeprowadzeniu ana-

Tablica 6.2: Porównanie wyników naszej metody przed ustaleniem wag oraz uśnięciem wybranych cech względem liczby próbek wykorzystanych przy tworzeniu wzorca (1, 2 lub 3 skany) dla procedury weryfikacyjnej.

Liczba próbek we wzorcu	FAR	FRR	Próg akceptacji
1	3.25%	1.04%	4.773
2	0.45%	0.87%	3.82
3	0.19%	1.19%	3.57

Tablica 6.3: Porównanie wyników naszej metody względem liczby próbek wykorzystanych przy tworzeniu wzorca biometrycznego (1, 2 lub 3 skany) dla procedury weryfikacyjnej.

Liczba próbek we wzorcu	FAR	FRR	Próg akceptacji
1	1.00%	1.04%	3.392
2	0.007%	0.87%	2.635
3	0.0%	1.19%	2.6

lizy danych, przypisaliśmy tym cechom wartości wag w_i równe 3, a pozostałym równe 1. Przykładowe wartości średniej oraz odchylenia standardowego odległości cechy o dużym znaczeniu zostały zaprezentowane w tablicy 6.1 (wiersz drugi), a histogram odległości został przestawiony na rysunku 6.8b.

Dodatkowo przeprowadzona analiza pozwoliła na usunięcie 4 początkowo wytypowanych cech (w rozdziale 6.4.1 nie zostały one opisane), które negatywnie wpływają na wyniki weryfikacji oraz identyfikacji. Zauważyliśmy, że powodem negatywnego wpływu tych cech na działanie metody są nieprzewidziane artefakty na samym obrazie oraz sama nietypowość cech biometryczne rąk. Przykładowo, powodem odrzucenia jednej cechy jest szerokość małego palca u jego podstawy, ponieważ u części populacji ten palec jest zbyt krótki. Z tego powodu w ostatecznej wersji metody dla małego palca liczone jest 6 szerokości, zamiast 7 lub 8, jak w przypadku pozostałych palców z wyjątkiem kciuka. Przykład takiej cechy został zaprezentowany w tablicy 6.1 (wiersz czwarty) oraz na rysunku 6.8d. Wartości współczynników FAR oraz FRR dla procedury weryfikacji dla przypadku przed oraz po ustanowieniu wag i odrzuceniu części cech zostały przedstawione, odpowiednio, w tablicy 6.2 oraz tablicy 6.3.

6.6 Wyniki eksperymentów

6.6.1 Wydajność metody

W procesie testowania wydajności naszego algorytmu opieramy się na dwóch standardowych wskaźnikach biometrycznych: współczynnik fałszywych akceptacji (FAR) oraz współczynnik fałszywych odrzuceń (FRR), które intuicyjnie pozwalają odpowiedzieć odpowiednio na pytania: „Jak często dochodzi do naruszenia bezpieczeństwa systemu?" oraz „Jak niewygodny w użytkowaniu jest system?" (tzn. jak często proces autoryzacji trzeba powtórzyć, mimo że autoryzowany jest właściwy użytkownik). Zwróćmy uwagę, że omawiane współczynniki odpowiadają współczynnikom z rozdziału 5. W przypadku trybu identyfikacji liczymy dodatkowo współczynnik identyfikacji (IR), który informuje jak dużo użytkowników zostało poprawnie skojarzone z ich wzorcami z bazy danych. Ten wskaźnik został już wykorzystany w rozdziale 4. Za pomocą tego wskaźnika mierzymy wydajność identyfikacji w scenariuszu ze zbiorem zamkniętym (ang. *closed set*), w którym przyjmujemy, że wzorzec użytkownika od którego została pobrana próbka znajduje się w bazie danych. Dodatkowo dla procesu identyfikacji testujemy również scenariusz zbioru otwartego (ang. *open set*), w którym posługujemy się wskaźnikami FAR i FRR. W tym przypadku, te dwa wskaźniki są interpretowane odpowiednio jako frakcja użytkowników niepoprawnie zidentyfikowanych – przypadek, w którym użytkownik nie znajduje się w bazie danych lub jest identyfikowany jako ktoś inny, a ponadto σ_{ERR} jest mniejsza niż próg akceptacji, oraz frakcja osób niedopasowanych do swoich wzorców (σ_{ERR} jest większa od progu akceptacji), mimo że ich wzorce zostały umieszczone w bazie danych.

W trakcie testowania procedury weryfikacyjnej skupiamy się na poziomie bezpieczeństwa metody, wykorzystujemy do tego współczynnik FAR przy odgórnie ustalonym poziomie wygody użytkowania systemu (współczynnik FRR). Na potrzeby testów zakładamy, że wartość współczynnika FRR ma wynosić około 1% i przy tym założeniu przeprowadzamy testy sprawdzające poziom bezpieczeństwa metody (analizujemy współczynnik FAR). Takie założenia pozwalają na dobranie progu akceptacji w sposób, który będzie pozwalał na racjonalny komfort użytkowania metody – średnio 1 na 100 prób autoryzacji będzie niesłusznie odrzucona i będzie wymagała ponowienia. Oczywiście, dla innych scenariuszy działania systemu mogą zostać poczynione inne założenia odnośnie współczynnika fałszywych odrzuceń. Jest to wyłącznie kwestia dostosowania progu akceptacji, który

Tablica 6.4: Porównanie wyników naszej metody względem liczby próbek wykorzystanych przy tworzeniu wzorca biometrycznego (1, 2 lub 3 skany) dla procedury identyfikacyjnej.

Liczba próbek we wzorcu	FAR	FRR	Próg akceptacji	IR
1	0.0%	8.30%	2.7	98.96%
2	0.0%	0.87%	2.635	100.0%
3	0.0%	1.19%	2.6	100.0%

może zostać w łatwy sposób określony (zmodyfikowany) nawet już w trakcie działania systemu, po jego wdrożeniu.

W ramach ewaluacji procedury identyfikacyjnej przeprowadzamy dwa rodzaje testów. Pierwszy z nich tyczy się podejście ze zamkniętym zbiorem danych, co oznacza, że w zbiorze testowym znajdują się tylko te skany użytkowników, którzy mają zarejestrowane wzorce w bazie danych. Dla każdego skanu ze zbioru testowego liczymy wektor cech biometrycznych i szukamy wektora wzorca, który jest najbardziej zbliżony do policzonego wektora. W tym celu liczmy wartość σ_{ERR} i wybieramy użytkownika dla którego ten współczynnik jest najmniejszy. Dla testów na zbiorze zamkniętym zwracamy współczynnik IR. W drugim przypadku analizujemy scenariusz zbioru otwartego, w którym rozważamy skany rąk użytkowników, którzy nie posiadają zarejestrowanego wzorca w bazie danych. W tym celu wykorzystujemy próbki pobrane od dodatkowych pięciu osób, liczymy wektory cech i porównujemy je ze wzorcami z bazy danych. Mając wyliczoną najmniejszą wartość współczynnika σ_{ERR}, sprawdzamy, czy jest ona mniejsza od progu akceptacji. Jeśli tak, przyjmujemy, że próbka pochodzi od użytkownika skojarzonego ze zidentyfikowanym wzorcem. Zauważmy, że scenariusz otwartego zbioru wymaga ustalenia progu akceptacji, co nie jest konieczne w przypadku scenariusza zbioru zamkniętego. Po serii testów określiliśmy, że wartości progów akceptacji ustalone w ramach testów procedury weryfikacyjnej są również odpowiednie dla procedury identyfikacyjnej w scenariuszu otwartego zbioru.

Testy zostały przeprowadzone dla wzorców zbudowanych z 1, 2 oraz 3 skanów dłoni. Wyniki dla procedury weryfikacyjnej zostały przedstawione w tablicy 6.3, a dla identyfikacji w tablicy 6.4.

Przy utrzymaniu założonej wartości współczynnika FRR równego w przybliżeniu 1%, możemy uznać, że nasza metoda jest „przyjazna użytkownikowi" (ang. *user-friendly*). Dla tej wartości współczynnika FRR osiągane

Tablica 6.5: Porównanie wyników wydajności procedury weryfikacyjnej dla naszej metody działającej wyłącznie z wykorzystaniem cech geometrii dłoni oraz uwzględniając tylko cechy krzywizny palców.

Liczba próbek we wzorcu	Geometria dłoni		Krzywizna	
	FAR	FRR	FAR	FRR
1	0.79%	1.04%	21.14%	11.07%
2	0.15%	0.87%	9.90%	10.48%
3	0.03%	1.19%	7.92%	10.65%

wyniki współczynnika FAR oznaczają, że nasza metoda, działająca na próbkach pobranych za pomocą podstawowego skanera biurowego, może zapewniać bezpieczeństwo na poziomie średnim do wysokiego. Ze względu na bardzo niskie (a nawet równe 0) wartości FAR, w szczególności przy tworzeniu wzorca biometrycznego wykorzystując więcej niż jeden skan dłoni, jesteśmy w stanie znacząco ograniczyć (a nawet wyeliminować) naruszenia bezpieczeństwa systemu. Dodatkowo scenariusze testowe, w których wykorzystywane są 4 lub więcej próbek wykazują, że uzyskiwany niewielki wzrost wydajność nie rekompensuje dużego dodatkowego narzutu obliczeniowego przy tworzeniu wzorca oraz konieczności gromadzenia dodatkowych danych. Warto zauważyć, że zwiększenie liczby wzorców mocno ogranicza zbiór testowy, ze względu na ograniczoną liczebność próbek w bazie danych, jednak równoważne wyniki otrzymaliśmy na większej bazie danych (patrz rozdział 6.6.4). Stąd przyjmujemy, że optymalna liczba próbek, z której powinien składać się wzorzec biometryczny zapisywany do bazy danych to 2 lub 3.

6.6.2 Znaczenie krzywizny palców

Nasze rozwiązane jest oparte na dwóch typach cech biometrycznych (patrz rozdział 6.4.1) – cechy geometrii dłoni (np. długość palca) oraz nowe cechy krzywizny palców, które niosą za sobą informację o tym jak krzywe są 4 palce ręki (poza kciukiem). W tablicy 6.5 zostały przedstawione wyniki wydajności procedury weryfikacyjnej dla naszego algorytmu działającego wyłącznie z wykorzystaniem cech geometrii dłoni oraz, w drugim przypadku, tylko opierając się na cechach krzywizny palców.

Otrzymane wysokie wartości współczynników FAR i FRR dla naszej metody bazującej wyłącznie na 12 cechach krzywizny palców wskazują, że

te cechy nie powinny być wykorzystywane jako samodzielna metoda do autoryzacji biometrycznej. Jednakże cechy krzywizny pozwalają uzyskać lepszą ogólną wydajność, gdy są one wykorzystywane jako cechy wspomagające dla metod opartych na standardowej biometrii konturu dłoni. Wyniki wskazują, że rozwiązanie oparte na standardowej geometrii dłoni i rozszerzone o dane nt. krzywizny palców pozwala na uzyskanie dużo lepszych wartości dla współczynnika FAR, niż gdyby ta metoda wykorzystywała tylko kontur dłoni. Przy czym wzorzec biometryczny powinien być tworzony z przynajmniej 2 próbek. Brak poprawy w przypadku budowy wzorca z tylko jedną próbką można argumentować tym, że w takim przypadku na odczyt wartości cech krzywizny palców z danej próbki mogą wpływać pewnie zaburzenia wynikające z ułożenia ręki na płycie skanera, np. palce zbyt blisko siebie lub zbyt mocno przyciśnięta dłoń do skanera. W przypadku, gdy wzorzec jest opracowywany z wykorzystaniem większej liczby skanów, wartości poszczególnych cech są uśredniane, stąd efekt zaburzeń staje się mniej widoczny. Metoda liczenia krzywizny palców jest szybka i zwiększa rozmiar wzorca zapisywanego w bazie danych jedynie o 48 bajty. Co więcej, jakość skanu pozwalającego na odczytanie cech krzywizny palców nie jest większa od jakość skanu wymaganego do obliczenia standardowych cech konturu dłoni, w przeciwieństwie do metod bazujących na analizie odcisku dłoni (ang. *palmprint*), które wymagają skanów o bardzo wysokiej jakości [Soc10]. Stąd wnioskujemy, że cechy krzywizny palców są godnym uwagi rozszerzeniem metody do autoryzacji biometrycznej bazującej na skanie dłoni.

6.6.3 Złożoność obliczeniowa metody

Testy wydajności naszej metody wskazują, że czas potrzebny na dokonanie ekstraktacji cech biometrycznych wynosi niecałe 1.2 sekundy. Z tego wynika, że cały proces liczenia wzorca biometrycznego zajmuje od 1.2 do 3.6 sekund, w zależności od liczby skanów wymaganych do jego stworzenia. W przypadku weryfikacji, w której obliczony wektor cech biometrycznych porównujemy jedynie z deklarowanym wzorcem z bazy danych, czas weryfikacji nadal nie przekracza 1.2 sekundy. Z kolei proces identyfikacji, który wymaga porównania z całą bazą danych, jest zależny od jej wielkości. Przy czym, przeszukanie nawet dużej bazy danych – rzędu kilkuset wzorców, zajmuje dodatkowo jedynie kilka milisekund (około 0.05 milisekundy/wzorzec), stąd przyjmujemy, że ten czas jest pomijalny. Istotnym czynnikiem przy pobieraniu próbki biometrycznej jest czas działania samego skanera. Obecnie skanery biurowe wykonują skan w około 2 do 10 sekund (zależnie od jego

jakości). Stąd całkowity proces rejestracji wzorca biometrycznego nie powinien przekraczać 40 sekund (przyjmując, że zostaną pobrane 3 próbki), z kolei proces weryfikacji i identyfikacji zajmie maksymalnie 12 sekund.

Zauważmy, że metoda nie wymaga wysokiej jakości skanu (jak w przypadku biometrii odcisku dłoni), stąd możliwym jest zastosowanie szybkiego oraz taniego skanera, a co za tym idzie, autoryzacja użytkownika w takiej konfiguracji może zająć jedynie 3.2 sekundy. Osiągnięte wartości wskazują, że nasze rozwiązanie może być wykorzystywane w warunkach rzeczywistych, gdzie istotnym czynnikiem jest czas działania systemu oraz jego koszt.

Wszystkie symulacje zostały przeprowadzone na komputerze wyposażonym w standardowy procesor Intel Core-i5 3317U 1.70 GHz. Kod źródłowy został napisany w języku Python oraz C++ z wykorzystaniem biblioteki OpenCV [Bra00].

6.6.4 Testy na Bosphorus Hand Dataset

Poza autorską bazą danych przygotowaną na potrzeby badań, nasze rozwiązanie zostało przetestowane na bazie danych Bosphorus Hand Dataset, która została stworzona w ramach pracy nad multimodalnym systemem biometrycznym opartym na cechach dłoni oraz twarzy [YKSD06]. Większość próbek w tym zbiorze danych zostało pobrane w warunkach zbliżonych do określonych w naszej pracy, tj. z wykorzystaniem skanera biurowego, bez użycia dodatkowego sztucznego oświetlenia, itd. Jednakże ze względu na niepełne kontury dłoni umieszczone na skanerze (niewidoczne koniuszki palców, nadgarstki) oraz braku przynajmniej 2 skanów dłoni pojedynczej osoby, część próbek musiała zostać odrzucona. Ostatecznie nasze rozwiązanie przetestowaliśmy na 2214 próbkach należących do 738 osób (z 914 w bazie danych). Ze względu na ograniczoną liczbę próbek na osobę naszą metodę sprawdziliśmy dla wzorca biometrycznego stworzonego z 2 skanów dłoni. Wszystkie pozostałe parametry nie zostały zmienione (w tym próg akceptacji). Istotną różnicą pomiędzy oboma zbiorami danych jest rozdzielczość DPI, która dla naszego zbioru wynosi 300, a dla Bosphours Hand Dataset jest to 43.

Otrzymane wyniki na opisanej bazie danych to IR = 99.86% dla zadania identyfikacji oraz FAR = 0.41% i FRR = 1.41% w przypadku weryfikacji.

Tablica 6.6: Porównanie naszej metody z innymi wybranymi rozwiązaniami opartymi na biometrii ręki. W tablicy zostały uwzględnione wyniki dla zadania weryfikacji w postaci FAR i FRR oraz dla identyfikacji w postaci IR. Wyniki zestawiono z parametrami rozwiązań – rodzajem urządzenia wraz z informacją o kołkach stabilizujących oraz liczbą próbek i wzorców zapisanych w bazie danych.

Metoda	Kołki	Rejestracja	Urządzenie	Weryf.		Ident.
				FAR	FRR	IR
Jain et al. [JRP99]	Tak	1 wzorzec, 5 próbek	kamera cyfrowa	0%	5%	-
Wong i Shi [WS02]	Nie	9 próbek	skaner biurowy	2.2%	11.1%	-
Kumar et al. (kontur) [KWSJ03]	Nie	1 wzorzec, 5 próbek	skaner biurowy	5.29%	8.34%	-
Kumar et al. (dłoń) [KWSJ03]	Nie	1 wzorzec, 5 próbek	skaner biurowy	0%	1.41%	-
HandKey II [MKCK01]	Tak	-	dedykowane	<1%	<1%	-
Varchol i Levicky [VL07]	Tak	15 próbek	skaner biurowy	0.27%	10.42%	80.18%
Dale [DGJ11]	Nie	6 próbek	kamera cyfrowa	0.57%	0.57%	100.0%
Adan [AAVT08]	Nie	5 próbek	dwie kamery cyfrowe	1.3%	1.3%	97.60%
Hu [HJZ+12]	Nie	1 próbka	kamera cyfrowa	1.0%	0.85%	99.60%
Singh [STKB14]	Nie	7 próbek	kamera cyfrowa	-	-	96.53%
Angadi i Hatture [AH15]	Nie	1 wzorzec, 6 próbek	skaner biurowy	-	-	97.92%
Nasza	Nie	1 wzorzec, 2 próbki	skaner biurowy	0.007%	0.87%	100.0%
Nasza	Nie	1 wzorzec, 3 próbki	skaner biurowy	0.0%	1.19%	100.0%

(a) HandKey II [MKCK01].

(b) Jain et al. [JRP99].

Rysunek 6.9: Przykładowe rozwiązania wykorzystujące kołki stabilizujące do pobrania próbek geometrii dłoni.

6.7 Porównanie metod

W tablicy 6.6 zostały przedstawione zbiorcze wyniki naszej metody oraz innych rozwiązań autoryzacyjnych opartych na biometrii ręki. Poza samymi wynikami skuteczności metody dla zadania weryfikacji oraz identyfikacji uwzględniono parametry metod, takie jak liczbę próbek wymaganych do rejestracji w bazie danych, liczbę osobnych wzorców zapisywanych w bazie danych, wykorzystywane urządzenie wraz z uwzględnieniem konieczności zastosowania kołków stabilizujących. Przykładowe systemy wykorzystujące kołki stabilizujące są przedstawione na rysunku 6.9.

Nasza metoda na tle innych charakteryzuje się niskimi wymaganiami, tj. nie jest konieczne użycie kołków oraz umożliwia działanie na próbkach pobranych skanerem biurowym. Dodatkowo proces rejestracji wzorca w bazie jest możliwy z wykorzystaniem jedynie 2 próbek (tylko [HJZ+12] wymaga mniejszej liczby, jednak ta metoda wykazuje się również niższą skutecznością). Wyniki zbliżone do naszych pod kątem współczynników FAR i FRR są uzyskiwane w [KWSJ03], jednak ta metoda bazuje na biometrii całej dłoni, tj. kontur oraz odcisk, a przy działaniu jedynie na konturze dłoni wyniki są znacząco gorsze. Co więcej, rozwiązanie wymaga dostarczenia skanu wyżej jakości (analiza odcisku ręki) oraz działa tylko w trybie weryfikacji (wyniki dla zadania identyfikacji nie zostały przedstawione). Metoda [DGJ11] również charakteryzuje się wysoką wydajnością zarówno podczas weryfikacji, jak i identyfikacji, a przy tym wykorzystuje kamerę cyfrową, która pozwala na szybsze pobranie próbki (kilka milisekund) względem skanera biurowego (2 sekundy). Jednak podobnie jak [KWSJ03], to rozwiązanie działa wykorzystując biometrię odcisku dłoni, a nie jedynie jej

konturu. Warto podkreślić, że przy wykorzystaniu większej liczby rodzajów biometrii (odcisk i kontur) w ramach jednej metody, jej wykorzystanie staje się bardziej ograniczone ze względu na wykluczenie większej części populacji z powodu różnych zaburzeń lub kalectwa. Algorytm wymaga 6 zdjęć dłoni w fazie rejestracji. Co więcej autorzy wskazali, że FAR wynosi 0.57%, co oznacza, że metoda jest mniej bezpieczna w stosunku do naszej. W pracy został podany EER (wartość błędu takiego, że FAR = FRR), stąd nie jest pewne, czy próg akceptacji może zostać zmieniony w celu zwiększenia bezpieczeństwa kosztem wygody użytkowania. Niemniej, uważamy, że istotniejszym parametrem w systemie autoryzacji biometrycznej jest FAR, który wypada korzystniej dla naszego rozwiązania.

6.8 Dlaczego nie używamy uczenia maszynowego?

Nasza metoda charakteryzuje się dużą szybkością działania na niskokosztowym procesorze oraz niewielkim zapotrzebowaniem na pamięć, a przy tym cechuje ją wysoka dokładność działania, zarówno pod kątem zapewnienia bezpieczeństwa (FAR = 0.0%), jak i wygodny użytkowania (FRR = 1.19%). Przy budowie jakiegokolwiek rozwiązania z dziedziny widzenia komputerowego, przetwarzania sygnału, czy rozpoznawania wzorców opartego na szeroko pojętej sztucznej inteligencji, czy też metodach klasycznych, istotną kwestią jest określenie rzeczywistych warunków oraz zakresu akceptowalnych parametrów, w których będzie działał system. Konieczne jest ustalenie górnego limitu mocy obliczeniowej możliwej do wykorzystania, maksymalnego czasu w jakim może działać metoda oraz oczekiwanej skuteczności systemu. O ile w przypadku metod wykrywających atak spoofingowy (rozdział 5.4) użycie sieci może wydawać się nieuniknione, ze względu na wykorzystanie złożonych i głębokich sieci neuronowych do tworzenia samych ataków na próbki biometryczne, które są w stanie wygenerować dane trudne do rozpoznania prostymi metodami (np. liniowymi), to wykorzystanie metod klasycznych do autoryzacji może być słusznym wyborem. Widzimy, że nasze rozwiązanie osiąga bardzo wysoką skuteczność autoryzacji biometrycznej, a przy tym jej działanie jest oparte na kilku szybkich algorytmach z dziedziny widzenia komputerowego. Stąd wnioskujemy, że w rzeczywistości, takie rozwiązanie będzie znajdowało dużo szersze zastosowanie i będzie przychylniej odbierane w porównaniu do złożonych metod uczenia maszynowego, często wymagających dużo większych zasobów obliczeniowych. W tym miejscu zauważmy pewien paradoks, mianowicie, systemy do autoryzacji biometrycznej są dzisiaj „lżejsze”, niż metody wykrywające ataki na

te systemy.

Kolejną istotną kwestią jest ograniczenie się do wykorzystania jedynie cech konturu dłoni. W naszej metodzie dokonujemy binaryzacji obrazu, który usuwa wszystkie pozostałe dane biometrycznie (odciski palców i dłoni, kolor skóry). W tym celu stosujemy kilka klasycznych metod, tj. rozmycie gaussowskie, filtr bilateralny oraz binaryzację koloru skóry. Podobne techniki przetwarzania wstępnego musiałby zostać zastosowanie również w przypadku wykorzystania metod uczenia maszynowego, co oznacza, że metoda nie byłaby w pełni *end-to-end*. W przeciwnym wypadku model mógłby uczyć się innych cech, które nie są konturem dłoni, co oznaczałoby de facto stworzenie metody opartej na innych cechach biometrycznych.

Ostatnim ważnym problemem jest wielkość dostępnych zbiorów danych dla tego zagadnienia, na którym mógłby zostać wytrenowany algorytm uczenia maszynowego, a w szczególności sieć neuronowa. Obecnie największym dostępnym zbiorem danym jest Bosphorus Hand Dataset [YKSD06], który zawiera łącznie 2742 skany dłoni pobranych od 914 osób. Przeprowadzone wstępne rozeznanie doprowadziło nas do konkluzji, że jest to znaczące ograniczenie, uniemożliwiające zastosowanie metod uczenia maszynowego, szczególnie uczenia głębokiego, w tym konkretnym zagadnieniu.

6.9 Podsumowanie

W rozdziale przedstawiliśmy system biometryczny oparty na geometrii dłoni, który działa z wykorzystaniem standardowego skanera biurowego. Nasza metoda wykorzystuje fakt unikatowości konturu dłoni w populacji oraz bazuje na standardowych algorytmach, w tym najważniejszych w kontekście rozwiązania – obliczeniu otoczki wypukłej i szukaniu defektów na konturze dłoni poprzez liczenie charakterystycznych odległości między określonymi punktami. Opracowany algorytm ekstrahuje 62 cechy dłoni, w tym niektóre niestandardowe, takie jak odległości między błonami międzypalcowymi w okolicy palca serdecznego. Zaproponowaliśmy też nowe cech geometryczne dłoni, tj. krzywiznę palców. Poprzez wyodrębnienie 3 odległości dla każdego palca definiujących ich krzywiznę urozmaiciliśmy wektor cech biometrycznych. Ponadto porównaliśmy i wskazaliśmy, które cechy mają większe znaczenie statystyczne w porównaniu z innymi, co poskutkowało dobraniem wag dla tych cech, w efekcie czego zwiększyło wydajności metody.

W celu przeprowadzenia eksperymentalnej oceny wydajności naszego algorytmu zebraliśmy dane od 60 osób w wieku od 20 do 50 lat i wyko-

naliśmy testy obejmujące dwa tryby pracy systemu, tj. weryfikację oraz identyfikację użytkowników. Przeprowadzone testy pokazują, że nasza metoda może być stosowana w systemach wymagających średniego i wysokiego poziomu bezpieczeństwa dzięki wynikom weryfikacji FAR i FRR wynoszącym odpowiednio 0.0% oraz 1.19% dla szablonu złożonego z 3 skanów. W przypadku identyfikacji użytkowników, przeprowadziliśmy testy zarówno z próbami obejmującymi identyfikację osób tylko w ramach zamkniętej bazy danych, jak i identyfikację osób w ramach scenariusza zbioru otwartego, uzyskując następujące wyniki FAR = 0.0%, FRR = 1.19% i IR = 100.0% dla szablonu złożonego z 3 skanów. Biorąc pod uwagę stosunkowo niewielki czas eksploracji bazy danych (podczas procesu identyfikacji) wynikający z wykonywania tylko podstawowych operacji arytmetycznych, algorytm identyfikacji jest odpowiedni dla systemów opartych na małych i średnich bazach danych, np. dostęp do biura dla pracodawców. Taki system jest znacznie bardziej przyjazny dla użytkownika, gdyż nie wymaga stosowania innych zabezpieczeń, takich jak identyfikatory czy hasła.

Rozdział 7

Podsumowanie rozprawy

W pracy przedstawiliśmy możliwości zastosowania metod zaliczanych do sztucznej inteligencji w dwóch zagadnieniach z obszaru bezpieczeństwa informacji – znakowania wodnego oraz biometrii.

W dwóch pierwszych rozdziałach przedstawiliśmy dwie metody znakowania wodnego treści wizualnych oparte na konwolucyjnych sieciach neuronowych, trenowane w schemacie enkoder-noiser-dekoder oraz rozszerzone o autorskie komponenty – propagator oraz translator – pozwalające na rozprzestrzeniane wiadomości przestrzennie, zwiększając przy tym transparentność oraz odporność na poszczególne ataki. Opracowaliśmy również różniczkowalną aproksymację metody kompresji obrazu JPEG, dzięki czemu ten komponent mógł zostać umieszczony w potoku treningowym i wykorzystany jako jeden z ataków w warstwie noiser. Co istotne, nasze metoda nadal zapewnia wysoki poziom odporności na atak przycinania. Następnie zaproponowaliśmy rozszerzenie enkodera o sieć neuronową – adapter – działającą niezależnie od obrazu wejściowego, dzięki czemu zwiększamy odporność oraz transparentność znaku wodnego, a przy tym utrzymujemy wysoką wydajność czasową kodowania. Nasze rozwiązania uzyskują wysokie wyniki odporności na ataki wynoszące odpowiednio 0.83 oraz 0.86 ogólnej dokładności bitowej. Warto podkreślić, że nasze rozwiązanie uzyskuje wydajność bitową równą aż 0.90 w przypadku kompresji JPEG z parametrem $q = 50$. W pracy zwróciliśmy też uwagę na kwestie związane z weryfikacją istnienia klucza (identyfikatora) w treści multimedialnej, a przy tym na problem fałszywie podejrzanych o piractwo. Zaproponowaliśmy przy tym rozszerzenie klasycznego schematu enkoder-noiser-dekoder o dodatkowy komponent – dyskryminator – pozwalający na weryfikację istnienia wiadomości w obrazie.

Następnie podjęliśmy się trzech tematów dotykających zagadnień w ob-

szarze biometrii – kwestii prywatności i ochrony danych biometrycznych, problemu odporności na ataki spoofingowe oraz opracowania klasycznej metody do autoryzacji biometrycznej. Co więcej, każde z wymienionych trzech zagadnień zostało przedstawione dla innego rodzaju danych biometrycznych, odpowiednio, cech twarzy, głosu oraz konturu dłoni. Pierwsza metoda zwiększająca prywatność i ograniczająca możliwość odtworzenia rysów twarzy z wzorca biometrycznego opiera się na kombinacji kilku metod uczenia maszynowego, w tym uczenia głębokiego, oraz algorytmów klasycznych, takich jak DTW. Metoda pozwala na stworzenia wzorca biometrycznego opartego o mimikę twarzy, a przez to, że zawiera tylko informacje o przesunięciu punktów charakterystycznych twarzy, a nie ich dokładne pozycje, znacząco ogranicza możliwości odtworzenia rysów twarzy ze wzorca. Przedstawiliśmy też propozycję wykorzystania naszej metody jako rozwiązania typu *cancelable bometrics*. Do wykrywania ataków spoofingowych wykorzystaliśmy dwie sieci neuronowe, tj. LCNN oraz bayesowską sieć neuronową z warstwami typu *Flipout*. Wejściem do sieci był sygnał w dziedzinie częstotliwości (wykorzystując do tego odpowiednio sparametryzowaną transformatę STFT). W pracy podjęliśmy również temat generalizacji modelu proponując przy tym technikę treningu *Attack-Out Cross-Validation*. Nadmierne dopasowanie modelu do danych treningowych jest szczególnie istotne w przypadku wykrywania spoofingu, gdzie musimy uwzględnić pojawianie się nowych ataków. Skuteczność podejścia został potwierdzona zbliżonymi wynikami błędów na zbiorach ewaluacyjnych, w tym na zbiorze zawierającym ataki niewystępujące w zbiorze treningowym. Z kolei, metoda do autoryzacji w oparciu o kontur dłoni wykorzystuje kilka klasycznych algorytmów widzenia komputerowego oraz geometrii obliczeniowej, takich jak binaryzacja obrazu, liczenie otoczki wypukłej, czy wpisanie trójkąta w wielokąt. W tym wypadku metody klasyczne zapewniały wystarczająco wysokie wyniki weryfikacji oraz identyfikacji użytkowników, wynoszące FAR = 0.0%, FRR = 1.19% oraz IR = 100.0%. Dzięki zastosowaniu szybkich algorytmów klasycznych nasza metoda może obliczyć wzorzec biometryczny i porównać go z bazą danych w niecałe 1.2 sekundy. Dodatkowo metoda może pracować na danych pobranych zwykłym skanerem biurowym.

Literatura

[AAVT08] Miguel Adan, Antonio Adan, Andres S. Vazquez, and Roberto Torres, *Biometric verification/identification based on hands natural layout*, Image and Vision Computing **26** (2008), no. 4, 451 – 465. [cytowanie na str. 131, 152]

[ABEN06] G. Amayeh, G. Bebis, A. Erol, and M. Nicolescu, *Peg-free hand shape verification using high order zernike moments*, 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06), June 2006, pp. 40–40. [cytowanie na str. 131]

[ABEN09] Gholamreza Amayeh, George Bebis, Ali Erol, and Mircea Nicolescu, *Hand-based verification and identification using palm-finger segmentation and fusion*, Computer Vision and Image Understanding **113** (2009), no. 4, 477 – 501. [cytowanie na str. 131]

[Adl09] Andy Adler, *Cancelable biometrics*, pp. 175–178, Springer US, Boston, MA, 2009. [cytowanie na str. 92]

[AH15] S. A. Angadi and S. M. Hatture, *User identification using wavelet features of hand geometry graph*, SAI Intelligent Systems Conference (IntelliSys), Nov 2015, pp. 828–835. [cytowanie na str. 131, 152]

[AH16] Shanmukhappa A. Angadi and Sanjeevakumar M. Hatture, *Biometric person identification system: A multimodal approach employing spectral graph characteristics of hand geometry and palmprint*, International Journal of Intelligent Systems and Applications **8** (2016), no. 3, 48–58. [cytowanie na str. 131]

[AKRB18] B. Ahmaderaghi, F. Kurugollu, J. M. D. Rincon, and A. Bouridane, *Blind image watermark detection algorithm based on discrete shearlet transform using statistical decision theory*, IEEE Transactions on Computational Imaging **4** (2018), no. 1, 46–59. [cytowanie na str. 14]

[AM19] Fariha Aiman and G. R. Manjula, *Video steganography using convolutional neural network and temporal residual method*, International Journal of Computer Applications **178** (2019), no. 46, 24–29. [cytowanie na str. 15]

[AMCB02] Mohammed Ebrahim Al-Mualla, C. Nishan Canagarajah, and David R. Bull, *Chapter 2 - video coding: Fundamentals*, Video Coding for Mobile Communications (Mohammed Ebrahim Al-Mualla, C. Nishan Canagarajah, and David R. Bull, eds.), Signal Processing and its Applications, Academic Press, San Diego, 2002, pp. 9–42. [cytowanie na str. 10]

[ANK+20] Mahdi Ahmadi, Alireza Norouzi, Nader Karimi, Shadrokh Samavi, and Ali Emami, *Redmark: Framework for residual diffusion watermarking based on deep networks*, Expert Systems with Applications **146** (2020), 113157. [cytowanie na str. 16, 21, 22, 26, 32, 35, 36, 52, 55, 56, 62, 64]

[ASV19] ASVspoof Consortium, *ASVspoof 2019: Automatic Speaker Verification Spoofing and Countermeasures Challenge Evaluation Plan*, 01 2019. [cytowanie na str. 111, 118]

[Bal17] Shumeet Baluja, *Hiding images in plain sight: Deep steganography*, Advances in Neural Information Processing Systems (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017, pp. 2069–2079. [cytowanie na str. 14, 36]

[Bal20] S. Baluja, *Hiding images within images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **42** (2020), no. 7, 1685–1697. [cytowanie na str. 14, 21, 36]

[Bar06] Mauro Barni, *Document and image compression*, CRC press, 2006. [cytowanie na str. 36, 52]

[BCF19] Mehdi Boroumand, Mo Chen, and Jessica Fridrich, *Deep residual network for steganalysis of digital images*, IEEE Transactions on Information Forensics and Security **14** (2019), no. 5, 1181–1193. [cytowanie na str. 15]

[BCKW15] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, *Weight Uncertainty in Neural Networks*, Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, JMLR.org, 2015, pp. 1613–1622. [cytowanie na str. 108, 118, 124]

[BCP13] J. Bringer, H. Chabanne, and A. Patey, *Privacy-preserving biometric identification using secure multiparty computation: An overview and recent trends*, IEEE Signal Processing Magazine **30** (2013), no. 2, 42–52. [cytowanie na str. 69]

[BCTPL16] Jorge Blasco, Thomas M. Chen, Juan Tapiador, and Pedro Peris-Lopez, *A survey of wearable biometric recognition systems*, ACM Comput. Surv. **49** (2016), no. 3, 43:1–43:35. [cytowanie na str. 68]

[Bd06] Niko Brümmer and Johan du Preez, *Application-independent evaluation of speaker detection*, Computer Speech And Language **20** (2006), no. 2, 230–275, Odyssey 2004: The speaker and Language Recognition Workshop. [cytowanie na str. 98]

[BDVS13] Samarth Bharadwaj, Tejas I. Dhamecha, Mayank Vatsa, and Richa Singh, *Computationally efficient face spoofing detection with motion magnification*, Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops (Washington, DC, USA), CVPRW '13, IEEE Computer Society, 2013, pp. 105–110. [cytowanie na str. 69]

[BGDP15] Priyanka R. Borude, S.T. Gandhe, P.A. Dhulekar, and G.M. Phade, *Identification and tracking of facial features*, Procedia Computer Science **49** (2015), 2 – 10, Proceedings of 4th International Conference on Advances in Computing, Communication and Control (ICAC3'15). [cytowanie na str. 69]

[BJKS04] Yaroslav Bulatov, Sachin Jambawalikar, Piyush Kumar, and Saurabh Sethia, *Hand recognition using geometric classifiers*, Biometric Authentication: First International Conference, ICBA 2004, Hong Kong, China, July 15-17, 2004. Proceedings (Berlin, Heidelberg), Springer Berlin Heidelberg, 2004, pp. 753–759. [cytowanie na str. 131]

[BK15] Karl Bringmann and Marvin Künnemann, *Quadratic conditional lower bounds for string problems and dynamic time warping*, 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, 2015, pp. 79–97. [cytowanie na str. 79]

[BKM+19] Radosław Białobrzeski, Michał Kośmider, Mateusz Matuszewski, Marcin Plata, and Alexander Rakowski, *Robust Bayesian and Light Neural Networks for Voice Spoofing Detection*, Proc. Interspeech 2019, 2019, pp. 1028–1032. [cytowanie na str. 6, 7, 8, 90]

[Böh10] Rainer Böhme, *Introduction*, pp. 1–7, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. [cytowanie na str. 13]

[Bra00] G. Bradski, *The OpenCV Library*, Dr. Dobb's Journal of Software Tools (2000). [cytowanie na str. 151]

[CAM12] I. Chingovska, A. Anjos, and S. Marcel, *On the effectiveness of local binary patterns in face anti-spoofing*, 2012 BIOSIG - Proceedings of the International Conference of Biometrics Special Interest Group (BIOSIG), Sept 2012, pp. 1–7. [cytowanie na str. 69]

[CCL+17] Weicheng Cai, Danwei Cai, Wenbo Liu, Gang Li, and Ming Li, *Countermeasures for automatic speaker verification replay spoofing attack : On data augmentation, feature representation, classification and fusion*, Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017, 2017, pp. 17–21. [cytowanie na str. 112]

[CH67] T. Cover and P. Hart, *Nearest neighbor pattern classification*, IEEE Transactions on Information Theory **13** (1967), no. 1, 21–27. [cytowanie na str. 19]

[CKLS97] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, *Secure spread spectrum watermarking for multimedia*, IEEE Transactions on Image Processing **6** (1997), no. 12, 1673–1687. [cytowanie na str. 11]

[CLG00] Rich Caruana, Steve Lawrence, and Lee Giles, *Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping*, Proceedings of the 13th International Conference on Neural Information Processing Systems (Cambridge, MA, USA), NIPS'00, MIT Press, 2000, p. 381?387. [cytowanie na str. 109]

[CMB+08] Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich, and Ton Kalker, *Chapter 1 - introduction*, Digital Watermarking and Steganography (Second Edition) (Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich, and Ton Kalker, eds.), The Morgan Kaufmann Series in Multimedia Information and Systems, Morgan Kaufmann, Burlington, second edition ed., 2008, pp. 1–13. [cytowanie na str. 13, 14]

[CS11] Ke Chen and Ahmad Salman, *Learning speaker-specific characteristics with a deep neural architecture*, IEEE Transactions on Neural Networks **22** (2011), no. 11, 1744–1756. [cytowanie na str. 95]

[CWCL19] Weicheng Cai, Haiwei Wu, Danwei Cai, and Ming Li, *The DKU replay detection system for the asvspoof 2019 challenge: On data augmentation, feature representation, classification, and fusion*, Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019 (Gernot Kubin and Zdravko Kacic, eds.), ISCA, 2019, pp. 1023–1027. [cytowanie na str. 125, 127]

[CXZ19] Xingliang Cheng, Mingxing Xu, and Thomas Fang Zheng, *Replay detection using cqt-based modified group delay feature and resnewt network in asvspoof 2019*, 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2019, pp. 540–545. [cytowanie na str. 125, 127]

[DG16] Tanima Dutta and Hari Prabhat Gupta, *A robust watermarking framework for high efficiency video coding (hevc) - encoded video with blind extraction process*, Journal of Visual Communication and Image Representation **38** (2016), 29 – 44. [cytowanie na str. 15]

[DGJ11] M. P. Dale, Hiren Galiyawala, and M. A. Joshi, *A new bimodal identification based on hand-geometry and palm-print*, Thinkquest~2010: Proceedings of the First International Conference on Contours of Computing Technology (New Delhi) (S. J. Pise, ed.), Springer India, 2011, pp. 86–91. [cytowanie na str. 131, 152, 153]

[DPMR00] George R. Doddington, Mark A. Przybocki, Alvin F. Martin, and Douglas A. Reynolds, *The nist speaker recognition evaluation - overview, methodology, systems, results, perspective*, Speech Communication **31** (2000), no. 2, 225–254. [cytowanie na str. 98, 101]

[DSBG11] R. Dubolia, R. Singh, S. S. Bhadoria, and R. Gupta, *Digital image watermarking by using discrete wavelet transform and discrete cosine transform and comparison based on psnr*, 2011 International Conference on Communication Systems and Network Technologies, June 2011, pp. 593–596. [cytowanie na str. 14]

[EFG+09] Zekeriya Erkin, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft, *Privacy-preserving face recognition*, pp. 235–253, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. [cytowanie na str. 68]

[Ern71] R. H. Ernst, *Hand id system*, April 27 1971, US Patent 3,576,537. [cytowanie na str. 130]

[FLY14] Yi C. Feng, Meng-Hui Lim, and Pong C. Yuen, *Masquerade attack on transform-based binary-template protection based on perceptron learning*, Pattern Recognition **47** (2014), no. 9, 3019 – 3033. [cytowanie na str. 84]

[FNC+19] A. Fierro-Radilla, M. Nakano-Miyatake, M. Cedillo-Hernandez, L. Cleofas-Sanchez, and H. Perez-Meana, *A Robust Image Zero-watermarking using Convolutional Neural Networks*, 2019 7th International Workshop on Biometrics and Forensics (IWBF), May 2019, pp. 1–5. [cytowanie na str. 15]

[Fri18] Friend MTS, *Comparing subscriber watermarking technologies for premium pay tv content*, Sep 2018. [cytowanie na str. 45]

[FZ14] M. Firas and S. Zainab, *A new features extracted for recognizing a hand geometry using bpnn*, International Journal of Scientific And Engineering Research **5** (2014), no. 9, 232–337. [cytowanie na str. 131]

[FZHK06] Marcos Faundez-Zanuy, Martin Hagmüller, and Gernot Kubin, *Speaker verification security improvement by means of speech watermarking*, Speech Communication **48** (2006), no. 12, 1608–1619, NOLISP 2005. [cytowanie na str. 92]

[GAGLD+21] Alejandro Gomez-Alanis, Jose A. Gonzalez-Lopez, S. Pavankumar Dubagunta, Antonio M. Peinado, and Mathew Magimai.-Doss, *On joint optimization of automatic speaker verification and anti-spoofing in the embedding space*, IEEE Transactions on Information Forensics and Security **16** (2021), 1579–1593. [cytowanie na str. 94]

[GB17] Rachel German and K. Suzanne Barber, *Current Biometric Adoption and Trends*, Tech. Report UT CID Report 18-02, The University of Texas at Austin,Center for Identity, Sep 2017. [cytowanie na str. 67]

[GEF+17] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter, *Audio Set: An Ontology and Human-Labeled Dataset for Audio Events*, Proc. IEEE ICASSP 2017 (New Orleans, LA), 2017. [cytowanie na str. 113]

[GMF+10] Javier Galbally, Chris McCool, Julian Fierrez, Sebastien Marcel, and Javier Ortega-Garcia, *On the vulnerability of face verification*

systems to hill-climbing attacks, Pattern Recognition **43** (2010), no. 3, 1027 – 1038. [cytowanie na str. 68, 92]

[GMF14] J. Galbally, S. Marcel, and J. Fierrez, *Biometric antispoofing methods: A survey in face recognition*, IEEE Access **2** (2014), 1530–1552. [cytowanie na str. 69]

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, *Generative adversarial nets*, Advances in Neural Information Processing Systems 27, 2014, pp. 2672–2680. [cytowanie na str. 10, 46]

[GR12] Akshya Kumar Gupta and Mehul S Raval, *A robust and secure watermarking scheme based on singular values replacement*, Sadhana **37** (2012), no. 4, 425–440. [cytowanie na str. 14]

[GS06] Francesca Gasparini and Raimondo Schettini, *Skin segmentation using multiple thresholding*, Proc. SPIE **6061** (2006), 60610F–60610F–8. [cytowanie na str. 134]

[GSG16] Puneet Gupta, Saurabh Srivastava, and Phalguni Gupta, *An accurate infrared hand geometry and vein pattern based authentication system*, Knowledge-Based Systems **103** (2016), 143 – 155. [cytowanie na str. 131]

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, International Conference on Learning Representations, 2015. [cytowanie na str. 118]

[GWFM+13] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio, *Maxout networks*, Proceedings of the 30th International Conference on Machine Learning (Atlanta, Georgia, USA) (Sanjoy Dasgupta and David McAllester, eds.), vol. 28, Proceedings of Machine Learning Research, no. 3, PMLR, 17–19 Jun 2013, pp. 1319–1327. [cytowanie na str. 121]

[HCE+17] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron Weiss, and Kevin Wilson, *CNN Architectures for Large-Scale Audio Classification*, International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017. [cytowanie na str. 113]

[HCMB12] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, *Exploiting the circulant structure of tracking-by-detection with kernels*, Computer Vision – ECCV 2012 (Berlin, Heidelberg) (Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, eds.), Springer Berlin Heidelberg, 2012, pp. 702–715. [cytowanie na str. 73]

[HD17] Jamie Hayes and George Danezis, *Generating steganographic images via adversarial training*, Advances in Neural Information Processing Systems (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017, pp. 1954–1963. [cytowanie na str. 14, 46]

[HJZ+12] Rong-Xiang Hu, Wei Jia, David Zhang, Jie Gui, and Liang-Tu Song, *Hand shape recognition based on coherent distance shape contexts*, Pattern Recognition **45** (2012), no. 9, 3348 – 3359. [cytowanie na str. 131, 152, 153]

[HK20] Ippei Hamamoto and Masaki Kawamura, *Neural watermarking method including an attack simulator against rotation and compression attacks*, IEICE Transactions on Information and Systems **E103.D** (2020), 33–41. [cytowanie na str. 15]

[HLP12] C. Y. Hsu, C. S. Lu, and S. C. Pei, *Image feature extraction in encrypted domain with privacy-preserving sift*, IEEE Transactions on Image Processing **21** (2012), no. 11, 4593–4607. [cytowanie na str. 68]

[HRBLM07] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller, *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*, Tech. Report 07-49, University of Massachusetts, Amherst, October 2007. [cytowanie na str. 89]

[HSB] HSBC Customer Service Site, *Your voice is your password*, https://www.us.hsbc.com/customer-service/voice/, ostatnia wizyta: 2023-01-09. [cytowanie na str. 92]

[HT20] Ching-Sheng Hsu and Shu-Fen Tu, *Enhancing the robustness of image watermarking against cropping attacks with dual watermarks*, Multimedia Tools and Applications **79** (2020), no. 17, 11297–11323. [cytowanie na str. 14]

[HZ10] Alain Horé and Djemel Ziou, *Image quality metrics: Psnr vs. ssim*, 2010 20th International Conference on Pattern Recognition, 2010, pp. 2366–2369. [cytowanie na str. 11]

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, *Deep residual learning for image recognition*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. [cytowanie na str. 126, 127]

[IS15] Sergey Ioffe and Christian Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, JMLR.org, 2015, pp. 448–456. [cytowanie na str. 21]

[IVW+18] Tadanobu Inoue, Phongtharin Vinayavekhin, Shiqiang Wang, David Wood, Nancy Greco, and Ryuki Tachibana, *Domestic activities classification based on cnn using shuffling and mixing data augmentation*, Tech. report, DCASE2018 Challange, Sep 2018. [cytowanie na str. 123]

[JHHN12] Huiyun Jing, Xin He, Qi Han, and Xiamu Niu, *Motion vector based information hiding algorithm for h.264/avc against motion vector steganalysis*, Intelligent Information and Database Systems (Berlin, Heidelberg) (Jeng-Shyang Pan, Shyi-Ming Chen, and Ngoc Thanh Nguyen, eds.), Springer Berlin Heidelberg, 2012, pp. 91–98. [cytowanie na str. 15]

[JRP99] Anil K. Jain, Arun Ross, and Sharath Pankanti, *A prototype hand geometry-based verification system*, 2nd International Conference on Audio- and Video-based Biometric Person Authentication (AVBPA), 1999, pp. 166 – 177. [cytowanie na str. 130, 152, 153]

[JTJ20] Lianchao Jin, Fuxiao Tan, and Shengming Jiang, *Generative adversarial network technologies and applications in computer vision*, Computational Intelligence and Neuroscience **2020** (2020), 1459107. [cytowanie na str. 10]

[KB15] Diederik P. Kingma and Jimmy Ba, *Adam: A method for stochastic optimization*, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (Yoshua Bengio and Yann LeCun, eds.), 2015. [cytowanie na str. 30, 51, 120, 124]

[KDE+20] Tomi Kinnunen, Héctor Delgado, Nicholas Evans, Kong Aik Lee, Ville Vestman, Andreas Nautsch, Massimiliano Todisco, Xin Wang, Md Sahidullah, Junichi Yamagishi, and Douglas A. Reynolds, *Tandem assessment of spoofing countermeasures and automatic speaker verification: Fundamentals*, IEEE/ACM Transactions on Audio, Speech, and Language Processing **28** (2020), 2195–2210. [cytowanie na str. 95, 96, 98, 106]

[Ker] Douglas A. Kerr, *Chrominance Subsampling in Digital Images*, `http://dougkerr.net/Pumpkin/articles/Subsampling.pdf`, ostatnia wizyta: 2022-12-27. [cytowanie na str. 12, 25]

[KH20] Umair Khan and Javier Hernando, *Unsupervised Training of Siamese Networks for Speaker Verification*, Proc. Interspeech 2020, 2020, pp. 3002–3006. [cytowanie na str. 95]

[KHHW20] Hung-Jui Ko, Cheng-Ta Huang, Gwoboa Horng, and Shiuh-Jeng WANG, *Robust and blind image watermarking in dct domain using inter-block coefficient correlation*, Information Sciences **517** (2020), 128 – 147. [cytowanie na str. 15]

[KL83] JB Kruskal and Mark Liberman, *The symmetric time-warping problem: From continuous to discrete*, Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison (1983). [cytowanie na str. 79]

[KLD+18] Tomi Kinnunen, Kong Aik Lee, Hector Delgado, Nicholas Evans, Massimiliano Todisco, Md Sahidullah, Junichi Yamagishi, and Douglas A. Reynolds, *t-dcf: a detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification*, Proc. Odyssey 2018 The Speaker and Language Recognition Workshop, 2018, pp. 312–319. [cytowanie na str. 93, 94, 95, 96, 98, 106]

[KM19] P. Korus and N. Memon, *Content authentication for neural imaging pipelines: End-to-end optimization of photo provenance in complex distribution channels*, 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8613–8621. [cytowanie na str. 16, 26]

[KMG17] Haribabu Kandi, Deepak Mishra, and Subrahmanyam R.K. Sai Gorthi, *Exploring the learning capabilities of convolutional neural networks for robust image watermarking*, Computers & Security **65** (2017), 247 – 268. [cytowanie na str. 15]

[KP10] B. J. Kang and K. R. Park, *Multimodal biometric method based on vein and geometry of a single finger*, IET Computer Vision **4** (2010), no. 3, 209–217. [cytowanie na str. 131]

[KPS18] Marek Klonowski, Marcin Plata, and Piotr Syga, *User authorization based on hand geometry without special equipment*, Pattern Recognition **73** (2018), 189–201. [cytowanie na str. 6, 7, 8, 129]

[KS14] V. Kazemi and J. Sullivan, *One millisecond face alignment with an ensemble of regression trees*, 2014 IEEE Conference on Computer Vision and Pattern Recognition, June 2014, pp. 1867–1874. [cytowanie na str. 74, 75, 84, 85]

[KSD+17] Tomi Kinnunen, Md. Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee, *The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection*, Proc. Interspeech 2017, 2017, pp. 2–6. [cytowanie na str. 116]

[KSK18] Chandan Kumar, Anuj Kumar Singh, and Priyadarshni Kumar, *Improved wavelet-based image watermarking through spiht*, Multimedia Tools and Applications (2018), 1–14. [cytowanie na str. 14]

[KW14] Diederik P. Kingma and Max Welling, *Auto-Encoding Variational Bayes*, 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014. [cytowanie na str. 108, 117, 118]

[KWSJ03] A. Kumar, D. C. M. Wong, H. C. Shen, and A. K. Jain, *Personal verification using palmprint and hand geometry biometric*, Audio- and Video-Based Biometric Person Authentication **2688** (2003), 668–678. [cytowanie na str. 152, 153]

[KWT+05] Hui Kong, Lei Wang, Eam Khwang Teoh, Jian-Gang Wang, and Ronda Venkateswarlu, *A framework of 2d fisher discriminant analysis: Application to face recognition with small number of training samples*, Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02 (Washington, DC, USA), CVPR '05, IEEE Computer Society, 2005, pp. 1083–1088. [cytowanie na str. 68]

[LAR+19] Cheng-I Lai, Alberto Abad, Korin Richmond, Junichi Yamagishi, Najim Dehak, and Simon King, *Attentive filtering networks*

for audio replay attack detection, 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019. [cytowanie na str. 116]

[LCT11] Khaled Loukhaoukha, Jean-Yves Chouinard, and Mohamed Haj Taieb, *Optimal image watermarking algorithm based on lwt-svd via multi-objective ant colony optimization*, Journal of Information Hiding and Multimedia Signal Processing **2** (2011), no. 4, 303–319. [cytowanie na str. 14]

[LCVD19] Cheng-I Lai, Nanxin Chen, Jesús Villalba, and Najim Dehak, *ASSERT: Anti-Spoofing with Squeeze-Excitation and Residual Networks*, Proc. Interspeech 2019, 2019, pp. 1013–1017. [cytowanie na str. 125, 127]

[LGMn+16] Ruggero Donida Labati, Angelo Genovese, Enrique Muñoz, Vincenzo Piuri, Fabio Scotti, and Gianluca Sforza, *Biometric recognition in automated border control: A survey*, ACM Comput. Surv. **49** (2016), no. 2, 24:1–24:39. [cytowanie na str. 67]

[LHL+19] J. Liu, J. Huang, Y. Luo, L. Cao, S. Yang, D. Wei, and R. Zhou, *An optimized image watermarking method based on hd and svd in dwt domain*, IEEE Access **7** (2019), 80849–80860. [cytowanie na str. 14]

[LM14] Gary B. Huang Erik Learned-Miller, *Labeled faces in the wild: Updates and new reporting procedures*, Tech. Report UM-CS-2014-003, University of Massachusetts, Amherst, May 2014. [cytowanie na str. 89]

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick, *Microsoft coco: Common objects in context*, Computer Vision – ECCV 2014 (Cham) (David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, eds.), Springer International Publishing, 2014, pp. 740–755. [cytowanie na str. 30, 51]

[LNM+17] Galina Lavrentyeva, Sergey Novoselov, Egor Malykh, Alexander Kozlov, Oleg Kudashev, and Vadim Shchemelinin, *Audio replay attack detection with deep learning frameworks*, Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017, 2017, pp. 82–86. [cytowanie na str. 112, 121]

[LNT+19] Galina Lavrentyeva, Sergey Novoselov, Andzhukaev Tseren, Marina Volkova, Artem Gorlanov, and Alexandr Kozlov, *STC Antispoofing Systems for the ASVspoof2019 Challenge*, Proc. Interspeech 2019, 2019, pp. 1033–1037. [cytowanie na str. 125, 126, 127]

[Loe03] Claudia Loebbecke, *Digital goods: An economic perspective*, Encyclopedia of Information Systems (Hossein Bidgoli, ed.), Elsevier, New York, 2003, pp. 635–647. [cytowanie na str. 9]

[LSZW20] Jiakang Li, Meng Sun, Xiongwei Zhang, and Yimin Wang, *Joint decision of anti-spoofing and automatic speaker verification by multi-task learning with contrastive loss*, IEEE Access **8** (2020), 7907–7915. [cytowanie na str. 94]

[LT10] Chih-Chin Lai and Cheng-Chih Tsai, *Digital image watermarking using discrete wavelet transform and singular value decomposition*, IEEE Transactions on instrumentation and measurement **59** (2010), no. 11, 3060–3063. [cytowanie na str. 14, 24, 35]

[LZC+20] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar, *Distortion agnostic deep watermarking*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. [cytowanie na str. 15, 20, 21, 22, 32, 35, 36, 52, 54, 55, 56, 62, 64]

[LZL+19] Dong Li, YingNan Zhang, XinChao Li, Ke Niu, XiaoYuan Yang, and YuJuan Sun, *Two-dimensional histogram modification based reversible data hiding using motion vector for h.264*, Multimedia Tools and Applications **78** (2019), 1–15. [cytowanie na str. 15]

[MCYJ19] Guangcan Mai, Kai Cao, Pong C. Yuen, and Anil K. Jain, *On the reconstruction of face images from deep face templates*, IEEE Transactions on Pattern Analysis and Machine Intelligence **41** (2019), no. 5, 1188–1202. [cytowanie na str. 68, 72, 84]

[MDFGOG11] Marcos Martinez-Diaz, Julian Fierrez, Javier Galbally, and Javier Ortega-Garcia, *An evaluation of indirect attacks and countermeasures in fingerprint verification systems*, Pattern Recognition Letters **32** (2011), no. 12, 1643–1651. [cytowanie na str. 92]

[MGS+12] A. Mishra, A. Goel, R. Singh, G. Chetty, and L. Singh, *A novel image watermarking scheme using extreme learning machine*, The 2012 International Joint Conference on Neural Networks (IJCNN), June 2012. [cytowanie na str. 14]

[MHC15] Emanuele Maiorana, Gabriel Emile Hine, and Patrizio Campisi, *Hill-climbing attacks on multibiometrics recognition systems*, IEEE Transactions on Information Forensics and Security **10** (2015), no. 5, 900–915. [cytowanie na str. 68, 92]

[MHP11] J. Määttä, A. Hadid, and M. Pietikäinen, *Face spoofing detection from single images using micro-texture analysis*, 2011 International Joint Conference on Biometrics (IJCB), Oct 2011, pp. 1–7. [cytowanie na str. 69]

[Mil71] R. P. Miller, *Finger dimension comparison identification system*, April 27 1971, US Patent 3,576,538. [cytowanie na str. 130]

[MJ13] Alexis Mignon and Frederic Jurie, *Reconstructing faces from their signatures using rbf regression*, Proceedings of the British Machine Vision Conference, BMVA Press, 2013. [cytowanie na str. 84]

[MK13] Nasrin M Makbol and Bee Ee Khoo, *Robust blind image watermarking scheme based on redundant discrete wavelet transform and singular value decomposition*, AEU-International Journal of Electronics and Communications **67** (2013), no. 2, 102–112. [cytowanie na str. 14]

[MK14] ———, *A new robust and secure digital image watermarking scheme based on the integer wavelet transform and singular value decomposition*, Digital Signal Processing **33** (2014), 134–147. [cytowanie na str. 14]

[MKCK01] T. Mansfield, G. Kelly, D. Chandler, and J. Kane, *Biometric product testing final report*, National Physical Laboratory, Teddington, Middlesex, UK, 2001. [cytowanie na str. 138, 152, 153]

[MNRD12] M. De Marsico, M. Nappi, D. Riccio, and J. L. Dugelay, *Moving face spoofing detection via 3d projective invariants*, 2012 5th IAPR International Conference on Biometrics (ICB), March 2012, pp. 73–78. [cytowanie na str. 69]

[MRLW01] F. Monrose, M.K. Reiter, Qi Li, and S. Wetzel, *Cryptographic key generation from voice*, Proceedings 2001 IEEE Symposium on Security and Privacy. S P 2001, 2001, pp. 202–213. [cytowanie na str. 92]

[Naj17] E. Najfi, *A robust embedding and blind extraction of image watermarking based on discrete wavelet transform*, Mathematical Sciences, vol. 11, Dec 2017, pp. 307–318. [cytowanie na str. 14]

[NCZ17] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, *VoxCeleb: A Large-Scale Speaker Identification Dataset*, Proc. Interspeech 2017, 2017, pp. 2616–2620. [cytowanie na str. 89]

[Net] Netflix's Support Site, *Downloading TV shows and movies on Netflix*, `https://help.netflix.com/en/node/54816`, ostatnia wizyta: 2022-12-27. [cytowanie na str. 10]

[NK19] Jakub Nalepa and Michal Kawulok, *Selecting training sets for support vector machines: a review*, Artificial Intelligence Review **52** (2019). [cytowanie na str. 81]

[NL19] E. Najafi and K. Loukhaoukha, *Hybrid secure and robust image watermarking scheme based on svd and sharp frequency localized contourlet transform*, Journal of Information Security and Applications **44** (2019), 144 – 156. [cytowanie na str. 14]

[NNKJ10] Abhishek Nagar, Karthik Nandakumar, and Anil K. Jain, *Biometric template transformation: A security analysis*, Proceedings of SPIE - The International Society for Optical Engineering, vol. 7541, 02 2010, p. 75410. [cytowanie na str. 72]

[NNS17] S. Natu, P. Natu, and T. Sarode, *Improved robust digital image watermarking with svd and hybrid transform*, 2017 International Conference on Intelligent Communication and Computational Techniques (ICCT), Dec 2017, pp. 177–181. [cytowanie na str. 14]

[NPK20] Truc Nguyen, Franz Pernkopf, and Michal Kosmider, *Acoustic scene classification for mismatched recording devices using heated-up softmax and spectrum correction*, ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 126–130. [cytowanie na str. 110]

[NWE+21] Andreas Nautsch, Xin Wang, Nicholas Evans, Tomi H. Kinnunen, Ville Vestman, Massimiliano Todisco, Héctor Delgado, Md Sahidullah, Junichi Yamagishi, and Kong Aik Lee, *Asvspoof 2019: Spoofing countermeasures for the detection of synthesized, converted and replayed speech*, IEEE Transactions on Biometrics, Behavior, and Identity Science **3** (2021), no. 2, 252–265. [cytowanie na str. 125]

[OCJL03] Michael Goh Kah Ong, Tee Connie, Andrew Teoh Beng Jin, and David Ngo Chek Ling, *A single-sensor hand geometry and palmprint verification system*, Proceedings of the 2003 ACM SIGMM Workshop on Biometrics Methods and Applications

(New York, NY, USA), WBMA '03, ACM, 2003, pp. 100–106. [cytowanie na str. 131]

[OEB03] Cenker Oden, Aytul Ercil, and Burak Buke, *Combining implicit polynomials and geometric features for hand recognition*, Pattern Recogn. Lett. **24** (2003), no. 13, 2145–2152. [cytowanie na str. 131]

[OH16] M. O. Oloyede and G. P. Hancke, *Unimodal and multimodal biometric sensing systems: A review*, IEEE Access **4** (2016), 7532–7555. [cytowanie na str. 68, 131]

[Par17] Michael Parker, *Chapter 25 - image and video compression fundamentals*, Digital Signal Processing 101 (Second Edition) (Michael Parker, ed.), Newnes, second edition ed., 2017, pp. 329 – 346. [cytowanie na str. 28]

[PHC05] V. M. Potdar, S. Han, and E. Chang, *A survey of digital image watermarking techniques*, INDIN '05. 2005 3rd IEEE International Conference on Industrial Informatics, 2005., Aug 2005, pp. 709–716. [cytowanie na str. 11]

[PK13] GiTae Park and Soowon Kim, *Hand biometric recognition based on fused hand geometry and vascular patterns*, Sensors **13** (2013), no. 3, 2895. [cytowanie na str. 131]

[Pla19] Marcin Plata, *Deep neural networks with supported clusters preclassification procedure for acoustic scene recognition*, Tech. report, DCASE2019 Challange, June 2019. [cytowanie na str. 110]

[PM92] William B. Pennebaker and Joan L. Mitchell, *Jpeg still image data compression standard*, 1st ed., Kluwer Academic Publishers, USA, 1992. [cytowanie na str. 26, 28]

[PPB10] Jagdish C Patra, Jiliang E Phua, and Cedric Bornand, *A novel dct domain crt-based watermarking scheme for image authentication surviving jpeg compression*, Digital Signal Processing **20** (2010), no. 6. [cytowanie na str. 14]

[PRC15] V. M. Patel, N. K. Ratha, and R. Chellappa, *Cancelable biometrics: A review*, IEEE Signal Processing Magazine **32** (2015), no. 5, 54–65. [cytowanie na str. 68, 87]

[PS20] M. Plata and P. Syga, *Robust spatial-spread deep neural image watermarking*, 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2020, pp. 62–70. [cytowanie na str. 5, 7]

[PS23] Marcin Plata and Piotr Syga, *Ddd-wm: Robust deep neural network–based image watermarking with double detector-discriminator scheme*, arXiv:2006.03921. [cytowanie na str. 7]

[PSK19] M. Plata, P. Syga, and M. Klonowski, *How to save your face: a facial recognition method robust against image reconstruction*, 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS), 2019, pp. 1–10. [cytowanie na str. 6, 7, 8]

[PSWL07] G. Pan, L. Sun, Z. Wu, and S. Lao, *Eyeblink-based anti-spoofing in face recognition from a generic webcamera*, 2007 IEEE 11th International Conference on Computer Vision, Oct 2007, pp. 1–8. [cytowanie na str. 69]

[Pun06] C. Pun, *A novel dft-based digital watermarking system for images*, 2006 8th international Conference on Signal Processing, vol. 2, Nov 2006. [cytowanie na str. 14]

[RCB01] N. K. Ratha, J. H. Connell, and R. M. Bolle, *Enhancing security and privacy in biometrics-based authentication systems*, IBM Systems Journal **40** (2001), no. 3, 614–634. [cytowanie na str. 68]

[RDPM00] Douglas A. Reynolds, George R. Doddington, Mark A. Przybocki, and Alvin F. Martin, *The nist speaker recognition evaluation - overview methodology, systems, results, perspective*, Speech Commun. **31** (2000), no. 2?3, 225?254. [cytowanie na str. 101]

[RH13] Michael Raggo and Chet Hosmer, *Chapter 1 - history of secret writing*, Data Hiding (Michael Raggo and Chet Hosmer, eds.), Syngress, Boston, 2013, pp. 1–17. [cytowanie na str. 9]

[Ric03] Iain E. G. Richardson, *H.264 and mpeg-4 video compression: video coding for next generation multimedia*, Chichester; Hoboken, NJ: Wiley, [2003] ©2003, [2003], Includes bibliographical references (page [277]) and index. [cytowanie na str. 42, 52]

[RMB17] A. Rajpal, A. Mishra, and R. Bala, *Fast digital watermarking of uncompressed colored images using bidirectional extreme learning machine*, 2017 International Joint Conference on Neural Networks (IJCNN), May 2017, pp. 1361–1366. [cytowanie na str. 14]

[RPCP13] Y. Rahulamathavan, R. C. W. Phan, J. A. Chambers, and D. J. Parish, *Facial expression recognition in the encrypted domain based on local fisher discriminant analysis*, IEEE Transactions on Affective Computing **4** (2013), no. 1, 83–92. [cytowanie na str. 69]

[RRM17] A. Tirupathi Rao, N. Pattabhi Ramaiah, and C. Krishna Mohan, *Palmprint recognition based on minutiae quadruplets*, Proceedings of International Conference on Computer Vision and Image Processing: CVIP 2016, Volume 2 (Singapore) (Balasubramanian Raman, Sanjeev Kumar, Partha Pratim Roy, and Debashis Sen, eds.), Springer Singapore, 2017, pp. 117–126. [cytowanie na str. 131]

[Sam] Samsung Newsroom, *Samsung Electronics Expands the Smart Features of the Family Hub Refrigerator*, https://news.samsung.com/global/samsung-electronics-expands-the-smart-features-of-the-family-hub-refrigerator, ostatnia wizyta: 2023-01-09. [cytowanie na str. 100]

[SAT+16] Christos Sagonas, Epameinondas Antonakos, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic, *300 faces in-the-wild challenge: database and results*, Image and vision computing **47** (2016), 3–18. [cytowanie na str. 75]

[SBD15] J. Svoboda, M. M. Bronstein, and M. Drahansky, *Contactless biometric hand geometry recognition using a low-cost 3d camera*, 2015 International Conference on Biometrics (ICB), May 2015, pp. 452–457. [cytowanie na str. 131]

[SC07] Stan Salvador and Philip Chan, *Toward accurate dynamic time warping in linear time and space*, Intell. Data Anal. **11** (2007), no. 5, 561–580. [cytowanie na str. 79]

[SDNC15] R. P. Singh, N. Dabas, Nagendra, and V. Chaudhary, *Weighted extreme learning machine for digital watermarking in dwt domain*, 2015 International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Sep. 2015, pp. 393–396. [cytowanie na str. 14]

[SJH+18] Hye-Jin Shim, Jee-Weon Jung, Hee-Soo Heo, Sung-Hyun Yoon, and Ha-Jin Yu, *Replay spoofing detection system for automatic speaker verification using multi-task learning of noise classes*, 2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI), 2018, pp. 172–176. [cytowanie na str. 112, 123]

[SKK+15] Aleksandr Sizov, Elie Khoury, Tomi Kinnunen, Zhizheng Wu, and Sébastien Marcel, *Joint speaker verification and antispoofing in the i -vector space*, IEEE Transactions on Information Forensics and Security **10** (2015), no. 4, 821–832. [cytowanie na str. 94]

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin, *Facenet: A unified embedding for face recognition and clustering*, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823. [cytowanie na str. 113]

[SLZ+13] Guanghua Song, Zhitang Li, Juan Zhao, J. Hu, and Hao Tu, *A reversible video steganography algorithm for mvc based on motion vector*, Multimedia Tools and Applications **74** (2013), 3759–3782. [cytowanie na str. 15]

[Soc10] Social, Behavioral and Economic Sciences Working Group United States of America, *Introduction to biometrics*, National Science and Technology Council, Washington, DC, USA, 2010. [cytowanie na str. 150]

[SRSAGM00] Raul Sanchez-Reillo, Carmen Sanchez-Avila, and Ana Gonzalez-Marcos, *Biometric identification through hand geometry measurements*, IEEE Trans. Pattern Anal. Mach. Intell. **22** (2000), no. 10, 1168–1171. [cytowanie na str. 130]

[SS17] J. L. Divya Shivani and Ranjan K. Senapati, *Robust image embedded watermarking using dct and listless spiht*, Future Internet **9** (2017), 33. [cytowanie na str. 14]

[STKB14] Aditya Pratap Singh, Ranjan Kumar Thakur, Arabind Kumar, and Ram Baksh, *User authentication using hand images*, International Journal of Science and Research **3** (2014), no. 3, 317–322. [cytowanie na str. 131, 152]

[TDE17] Massimiliano Todisco, Héctor Delgado, and Nicholas Evans, *Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification*, Computer Speech & Language **45** (2017), 516 – 535. [cytowanie na str. 112]

[TLT+19] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang, *Cnn-based adversarial embedding for image steganography*, IEEE Transactions on Information Forensics and Security **14** (2019), no. 8, 2074–2087. [cytowanie na str. 15]

[TMN20] Matthew Tancik, Ben Mildenhall, and Ren Ng, *Stegastamp: Invisible hyperlinks in physical photographs*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. [cytowanie na str. 15]

[TWV+19] Massimiliano Todisco, Xin Wang, Ville Vestman, Md. Sahidullah, Héctor Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi H. Kinnunen, and Kong Aik Lee, *ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection*, Proc. Interspeech 2019, 2019, pp. 1008–1012. [cytowanie na str. 109, 127]

[USA14] J.A. Unar, Woo Chaw Seng, and Almas Abbasi, *A review of biometric technology along with trends and prospects*, Pattern Recognition **47** (2014), no. 8, 2673–2688. [cytowanie na str. 129]

[VCF18] V. Vukotić, V. Chappelier, and T. Furon, *Are deep neural networks good for blind image watermarking?*, 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 2018, pp. 1–7. [cytowanie na str. 15]

[vdMH08] Laurens van der Maaten and Geoffrey Hinton, *Visualizing data using t-sne*, Journal of Machine Learning Research **9** (2008), no. 86, 2579–2605. [cytowanie na str. 114, 115]

[VJ01] P. Viola and M. Jones, *Rapid object detection using a boosted cascade of simple features*, Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, vol. 1, 2001, pp. I–511–I–518 vol.1. [cytowanie na str. 73]

[VL07] P. Varchol and D. Levicky, *Using of hand geometry in biometric security systems*, RADIOENGINEERING **16** (2007), no. 4, 82–86. [cytowanie na str. 131, 138, 152]

[VLB+19] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio, *Manifold mixup: Better representations by interpolating hidden states*, Proceedings of the 36th International Conference on Machine Learning (Kamalika Chaudhuri and Ruslan Salakhutdinov, eds.), Proceedings of Machine Learning Research, vol. 97, PMLR, 09–15 Jun 2019, pp. 6438–6447. [cytowanie na str. 123]

[WA19] Bingyang Wen and Sergul Aydore, *ROMark: A Robust Watermarking System Using Adversarial Training*, arXiv e-prints (2019), arXiv:1910.01221. [cytowanie na str. 15, 20, 22, 25, 32, 36]

[WCD+17] Joanne Woodage, Rahul Chatterjee, Yevgeniy Dodis, Ari Juels, and Thomas Ristenpart, *A new distribution-sensitive secure*

sketch and popularity-proportional hashing, Advances in Cryptology – CRYPTO 2017 (Cham) (Jonathan Katz and Hovav Shacham, eds.), Springer International Publishing, 2017, pp. 682–710. [cytowanie na str. 92]

[WD19] Eric Wengrowski and Kristin Dana, *Light field messaging with deep photographic steganography*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019. [cytowanie na str. 15]

[WEK+15] Zhizheng Wu, Nicholas Evans, Tomi Kinnunen, Junichi Yamagishi, Federico Alegre, and Haizhou Li, *Spoofing and countermeasures for speaker verification: A survey*, Speech Communication **66** (2015), 130–153. [cytowanie na str. 90, 91]

[WHST18] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan, *A Light CNN for Deep Face Representation with Noisy Labels*, IEEE Transactions on Information Forensics and Security **13** (2018), no. 11, 2884–2896. [cytowanie na str. 108, 121]

[WI20] A.G. Wilson and P. Izmailov, *Bayesian deep learning and a probabilistic perspective of generalization*, vol. 2020-December, Neural information processing systems foundation, 2020, cited By 3 (English). [cytowanie na str. 124]

[WKZ+17] Marcin Witkowski, Stanislaw Kacprzak, Piotr Zelasko, Konrad Kowalczyk, and Jakub Galka, *Audio replay attack detection using high-frequency features*, Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017, 2017, pp. 27–31. [cytowanie na str. 112]

[WLCM19] Xinyu Weng, Yongzhi Li, Lu Chi, and Yadong Mu, *High-capacity convolutional video steganography with temporal residual modeling*, Proceedings of the 2019 on International Conference on Multimedia Retrieval (New York, NY, USA), ICMR '19, Association for Computing Machinery, 2019, pp. 87–95. [cytowanie na str. 15]

[WLW04] Ting-Fan Wu, Chih-Jen Lin, and Ruby C. Weng, *Probability estimates for multi-class classification by pairwise coupling*, J. Mach. Learn. Res. **5** (2004), 975–1005. [cytowanie na str. 80]

[WP07] Y. Wang and K. N. Plataniotis, *Face based biometric authentication with changeable and privacy preservable templates*, 2007 Biometrics Symposium, Sept 2007, pp. 1–6. [cytowanie na str. 68]

[WS02] L. N. Wong and P. Shi, *Peg-free hand geometry recognition using hierarchical geometry and shape matching*, IAPR Workshop on Machine Vision Applications (Nara), 2002, pp. 281–284. [cytowanie na str. 130, 152]

[WVB+18] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse, *Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches*, International Conference on Learning Representations, 2018. [cytowanie na str. 118, 119, 124]

[WYT+20] Xin Wang, Junichi Yamagishi, Massimiliano Todisco, Héctor Delgado, Andreas Nautsch, Nicholas Evans, Md Sahidullah, Ville Vestman, Tomi Kinnunen, Kong Aik Lee, Lauri Juvela, Paavo Alku, Yu-Huai Peng, Hsin-Te Hwang, Yu Tsao, Hsin-Min Wang, Sébastien Le Maguer, Markus Becker, Fergus Henderson, Rob Clark, Yu Zhang, Quan Wang, Ye Jia, Kai Onuma, Koji Mushika, Takashi Kaneda, Yuan Jiang, Li-Juan Liu, Yi-Chiao Wu, Wen-Chin Huang, Tomoki Toda, Kou Tanaka, Hirokazu Kameoka, Ingmar Steiner, Driss Matrouf, Jean-François Bonastre, Avashna Govender, Srikanth Ronanki, Jing-Xuan Zhang, and Zhen-Hua Ling, *Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech*, Computer Speech And Language **64** (2020), 101114. [cytowanie na str. 109, 111, 112]

[WZW+17] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin, *Deep Metric Learning with Angular Loss*, IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, 2017, pp. 2612–2620. [cytowanie na str. 113]

[XWW11] Dawen Xu, Rangding Wang, and Jicheng Wang, *A novel watermarking scheme for h.264/avc video authentication*, Signal Processing: Image Communication **26** (2011), no. 6, 267 – 279. [cytowanie na str. 15]

[XXO05] Wei Xiong, Changsheng Xu, and Sim Heng Ong, *Peg-free human hand shape analysis and recognition*, Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005., vol. 2, March 2005, pp. 77–80. [cytowanie na str. 131]

[Yin19] Xue Ying, *An overview of overfitting and its solutions*, Journal of Physics: Conference Series **1168** (2019), 022022. [cytowanie na str. 109]

[YKSD06] E. Yoruk, E. Konukoglu, B. Sankur, and J. Darbon, *Shape-based hand recognition*, IEEE Transactions on Image Processing **15** (2006), no. 7, 1803–1815. [cytowanie na str. 151, 155]

[YWZK22] Jinkun You, Yuan-Gen Wang, Guopu Zhu, and Sam Kwong, *Truncated robust natural watermarking with hungarian optimization*, IEEE Transactions on Circuits and Systems for Video Technology **32** (2022), no. 2, 483–495. [cytowanie na str. 14]

[YZZ21] Weike You, Hong Zhang, and Xianfeng Zhao, *A siamese cnn for image steganalysis*, IEEE Transactions on Information Forensics and Security **16** (2021), 291–306. [cytowanie na str. 15]

[ZBK+20] Chaoning Zhang, Philipp Benz, Adil Karjauv, Geng Sun, and In Kweon, *Udh: Universal deep hiding for steganography, watermarking, and light field messaging*, Advances in Neural Information Processing Systems 33, 2020. [cytowanie na str. 15]

[ZCDLP18] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz, *mixup: Beyond empirical risk minimization*, International Conference on Learning Representations, 2018. [cytowanie na str. 123]

[ZGG16] David Zhang, Zhenhua Guo, and Yazhuo Gong, *Empirical study of light source selection for palmprint recognition*, Multispectral Biometrics: Systems and Applications (Cham), Springer International Publishing, 2016, pp. 139–151. [cytowanie na str. 131]

[ZHMS20] X. Zhong, P. Huang, S. Mastorakis, and F. Y. Shih, *An automated and robust image watermarking scheme based on deep neural networks*, IEEE Transactions on Multimedia (2020). [cytowanie na str. 15]

[ZKJFF18] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei, *HiDDeN: Hiding Data with Deep Networks*, The European Conference on Computer Vision (ECCV), Sep 2018. [cytowanie na str. 15, 20, 21, 22, 24, 25, 32, 33, 35, 36, 48, 52, 54, 55, 56, 62, 64]

[ZS19] Xin Zhong and Frank Y. Shih, *A Robust Image Watermarking System Based on Deep Neural Networks*, arXiv e-prints (2019), arXiv:1908.11331. [cytowanie na str. 15]

[ZXCIV19] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni, *Robust invisible video watermarking with attention*, arXiv:1909.01285. [cytowanie na str. 15, 52, 56]

[ZXZ16] David Zhang, Yong Xu, and Wangmeng Zuo, *Multifeature palmprint authentication*, Discriminative Learning in Biometrics (Singapore), Springer Singapore, 2016, pp. 147–164. [cytowanie na str. 131]

[ZYL+12] Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, *A face antispoofing database with diverse attacks*, 2012 5th IAPR International Conference on Biometrics (ICB), March 2012, pp. 26–31. [cytowanie na str. 69]