WYDZIAŁ AUTOMATYKI, ELEKTRONIKI I INFORMATYKI

Rozprawa doktorska

Wykorzystanie metod uczenia maszynowego w przetwarzaniu oraz analizie obrazów astronomicznych

Autor: mgr inż. Piotr Jóźwik-Wabik

Promotor: dr hab. inż. Adam Popowicz, prof. Politechniki Śląskiej

Promotor pomocniczy: dr inż. Krzysztof Bernacki

Gliwice, październik 2025

Spis treści

Struktura pracy

Niniejsza praca składa się z ośmiu rozdziałów i jednego dodatku. Poniżej zamieszczono skrócony opis każdego z tych elementów.

W pierwszym rozdziale przedstawiono intensywny rozwój metod uczenia maszynowego, który sprawił, że w ostatnich latach są one coraz częściej wykorzystywane w licznych dziedzinach nauki, w tym w astronomii. Podkreślono rolę małych obserwatoriów w akwizycji danych, a także zwrócono uwagę na zagadnienie dopasowania modelu do konkretnych zadań. Na sam koniec sformułowano tezę niniejszej pracy.

Drugi rozdział opisuje różne typy obserwacji astronomicznych wraz z ich specyfiką, ze szczególnym naciskiem na małe obserwatoria naziemne. Wprowadzono w nim pojęcia kluczowe do zrozumienia dalszej części pracy. Dodatkowo, wymieniono główne aspekty wpływające na jakość pozyskiwanych danych, takie jak zniekształcenie obrazu przez turbulentną atmosferę.

Przeprowadzone badania koncentrują się na dopasowaniu sieci neuronowych do poszczególnych zadań. W trzecim rozdziale wyjaśniono, dlaczego spośród wszystkich metod uczenia maszynowego praca skupia się wyłącznie na sieciach neuronowych. Przybliżono przy tym zasady ich działania, a także podkreślono wyzwania towarzyszące przystosowaniu ich do pracy. Opisano również podstawowe warstwy sieci wykorzystywane w eksperymentach rozważanych w kolejnych rozdziałach.

Najbardziej intuicyjnym podejściem do wytrenowania sieci neuronowych jest zastosowanie uczenia nadzorowanego z wykorzystaniem par obrazów: obraz zaszumiony – obraz pozbawiony szumu. W przypadku obserwacji naziemnych niemożliwym jest pozyskanie danych drugiego typu. W związku z tym, przeprowadzono badania nad innymi podejściami do treningu sieci, w ramach których wykorzystano obrazy monochromatyczne ze zbioru MNIST (ang. *Modified National Institute of Standards and Technology database*).

Bazując na wynikach z rozdziału czwartego, dane syntetyczne zastąpiono docelowymi obrazami nocnego nieba. Wykorzystane zostały obrazy zarejestrowane z czasem ekspozycji 500 ms, w oparciu o które przetestowano wybrane architektury sieci neuronowych. Weryfikacji poddane zostały podstawowe metryki porównujące jakość obrazów, a następnie porównano jakość detekcji gwiazd na przetworzonych obrazach.

Uznanym standardem wykorzystywanym w poprawie obrazów słonecznych jest algorytm MFBD (ang. *Multi-Frame Blind Deconvolution*), który wykorzystuje wiele klatek do estymacji stanu Słońca w danym momencie. Rozdział szósty obejmuje badania nad wykorzystaniem sieci neuronowych jako podejścia alternatywnego, który umożliwia przetwarzanie obrazów niemal w czasie rzeczywistym, podczas gdy metoda MFBD wymaga nieraz wielogodzinnych obliczeń.

W przemysłowych zastosowaniach sieci neuronowych często stosowaną praktyką jest wytrenowanie dużej sieci, a następnie zmniejszenie jej rozmiaru i wymagań, by przystosować ją do działania na mniej wymagającym sprzęcie. Siódmy rozdział opisuje badania nad wpływem takiej optymalizacji na działanie sieci zaproponowanych w rozdziale szóstym.

Ostatni rozdział pracy podsumowuje wyniki przeprowadzonych badań i konfrontuje je z postawioną w pierwszym rozdziale tezą badawczą.

Część wyników opisanych w niniejszej pracy została zaprezentowana w ramach referatów na krajowych i zagranicznych konferencjach, a także opublikowana w formie artykułów naukowych. Dodatek do pracy prezentuje listę wystąpień i publikacji z wyszczególnieniem, której części badań one dotyczą.

1. Wprowadzenie

Wyobraźnię ludzi od tysięcy lat rozpalała idea stworzenia sztucznego, posłusznego bytu, który będzie w stanie sumiennie i precyzyjnie wykonywać powierzone mu zadania. W starożytności marzenie to przybrało postać mitu o Talosie, wykutym przez Hefajstosa olbrzymie z brązu, który miał chronić Kretę, codziennie okrążając ją trzy razy. Przez następne wieki wizja ta pojawiała się w postaci homunkulusa, Golema czy potwora Frankensteina, by na początku XX wieku zaistnieć jako „robot" za sprawą sztuki „Rossumovi Univerzální Roboti" Karela Čapka z 1921 roku. W późniejszych latach określenie to utrwaliło się w języku między innymi za sprawą Isaaca Asimova i jego Trzech Praw Robotyki, dając początek dyscyplinie, jaką jest robotyka.

Same próby stworzenia inteligentnej maszyny nie wychodziły jednak daleko poza ramy literatury fantastycznonaukowej, aż do opublikowania przez Warrena McCullocha i Waltera Pittsa przełomowego artykułu „A Logical Calculus of the Ideas Immanent in Nervous Activity" [1] w 1943 roku. Była to pierwsza formalna próba opisania działania mózgu za pomocą logiki matematycznej i modelu sieci złożonej ze sztucznych neuronów. Chociaż zaproponowany model okazał się z biegiem lat zbyt uproszczony, dał podwaliny na rozwój badań w dziedzinie, którą za sprawą Jacka McCarthy'ego nazwano „sztuczną inteligencją" [2].

Do końca lat 60. zeszłego wieku zainteresowanie sztucznymi sieciami neuronowymi było ogromne i spodziewano się, że już wkrótce pojawi się maszyna zdolna przejść test zaproponowany przez Turinga [3], czyli nie będzie możliwe odróżnienie jej od człowieka jedynie poprzez rozmowę. Stosowane wtedy podejścia okazały się jednak niewystarczające do stawianych im zadań. Ponadto, nawet najprostsze z nich wymagały znaczących zasobów sprzętowych, zazwyczaj przekraczających możliwości ówczesnych komputerów dysponujących małą ilością pamięci i niską mocą obliczeniową. Skutkowało to spadkiem zainteresowania dziedziną i wstrzymaniem finansowania badań na długie lata. Przez następne dekady temat sztucznej inteligencji okresowo wracał do łask, jednakże prawdziwy przełom nastąpił dopiero po dłuższej przerwie.

W 1965 roku Gordon Moore opisał zjawisko podwajania się liczby tranzystorów w układach scalonych w odstępach 18–24 miesięcy przy zachowaniu tej samej ceny [4]. Zjawisko to, nazwane prawem Moore'a, znalazło potwierdzenie w praktyce i w okolicy 2000 roku można było mówić o łatwym dostępie do komputerów wyposażonych we względnie

wydajne procesory i pamięć fizyczną. W tym samym czasie na potrzeby komercyjne prężnie rozwijano jednostki GPU (ang. *Graphics Processing Unit*), pierwotnie wykorzystywane wyłącznie do obsługi grafiki w grach i aplikacjach. Jednym z punktów zwrotnych w zastosowaniu GPU było opracowanie przez firmę NVIDIA w 2006 roku platformy CUDA (ang. *Compute Unified Device Architecture*), która umożliwiła zastosowanie GPU do równoległych obliczeń matematycznych. W rezultacie modele sztucznych sieci neuronowych mogły być „trenowane", czyli dostosowywane do stawianych im zadań, znacznie szybciej i na większą skalę, co ponownie rozbudziło zapał wielu zespołów badawczych.

Za początek trwającego obecnie „renesansu sztucznej inteligencji" najczęściej uznaje się wygranie przez sieć AlexNet [5] konkursu „ImageNet Large Scale Visual Recognition Challenge" w 2012 roku, co pokazało, że sieci mogą być skuteczne w rozpoznawaniu obrazów. Od tego czasu mniej więcej co roku odbywał się swoisty przełom w badaniach nad sieciami neuronowymi. W przetwarzaniu obrazów za największe kamienie milowe można uznać zaprezentowanie sieci generatywnych typu GAN (ang. *Generative Adversarial Network*) [6], wprowadzenie warstw rezydualnych w ResNet [7], poprawę możliwości segmentacji obiektów na obrazie przez Mask R-CNN [8], przetwarzanie obrazów przy pomocy opisu językowego za pomocą CLIP [9] i generowanie obrazów na podstawie tekstu przez DALL·E [10]. Równolegle postępował rozwój w innych poddziedzinach, przy czym najbardziej znaczącym w ostatnich latach okazał się rozwój dużych modeli językowych takich jak BERT [11] czy GPT [12]. W efekcie, rozwiązania oparte na sztucznej inteligencji znajdują zastosowanie w coraz większej liczbie obszarów, spośród których wymienić można wykrywanie nieautoryzowanych transakcji bankowych [13], optymalizacje łańcuchów dostaw [14], wykrywanie nowotworów na cyfrowych skanach biopsji [15] i wiele innych.

Sieci neuronowe odgrywają również coraz większą rolę w licznych pracach badawczych związanych z astronomią. Dziedzina ta charakteryzuje się przetwarzaniem danych, których ilość rośnie wraz z rozwojem teleskopów i instrumentów obserwacyjnych. W tym kontekście sieci neuronowe okazują się bardzo przydatne, oferując możliwości szybkiego przetworzenia i automatycznej analizy informacji, dorównując lub przewyższając algorytmy deterministyczne względem jakości wyników. Do kluczowych zastosowań tej technologii można zaliczyć detekcję obiektów [16][17][18] i ich klasyfikację [19][20][21], które znacząco przyspieszają proces katalogowania i interpretacji wyników, a także mogą prowadzić do wykrycia nietypowych sygnałów. Dodatkowo, sieci neuronowe skutecznie wspomagają

redukcję szumu – potrafią odseparować sygnał rzeczywisty od różnego rodzaju zakłóceń, znacznie poprawiając jakość oraz niezawodność pozyskiwanych danych [22][23][24][25].

Podejście do wykorzystania tych metod w astronomii skupiło się, w głównej mierze, na zastosowaniu ich w teleskopach kosmicznych i teleskopach naziemnych stosowanych przez największe obserwatoria. Do przeprowadzanych eksperymentów najczęściej wykorzystuje się dane ze zbiorów uzyskiwanych w ramach projektów takich jak SDSS (ang. *Sloan Digital Sky Survey*) [26], TESS (ang. *Transiting Exoplanet Survey Satellite*) [27] i ZTF (ang. *Zwicky Transient Facility*) [28]. Rola niewielkich teleskopów wraz z odmiennym charakterem zbieranych przez nie danych jest w tych badaniach najczęściej pomijana, co jest znaczącym niedopatrzeniem.

Małe teleskopy astronomiczne uczestniczą w licznych projektach badawczych, stanowiąc trzon przygotowań do późniejszych, szerzej realizowanych i bardziej szczegółowych badań. Instrumenty te, znacznie liczniej i gęściej rozmieszczone od dużych jednostek obserwacyjnych, mogą synchronicznie dostarczać i od razu weryfikować wiedzę o zjawiskach przejściowych, takich jak tranzyty planet pozasłonecznych. Dzięki większemu polu widzenia są również w stanie na bieżąco monitorować znacznie obszerniejsze części nieba. Ich rola jest szczególnie istotna w kontekście otwarcia Obserwatorium im. Very C. Rubin, które zdaniem wielu astronomów ma rozpocząć nową erę w astronomii. Jednostka ta ma zbierać dane na niespotykaną dotychczas skalę, około 20 terabajtów na dzień. Jednakże nie będzie służyła ona do ciągłej obserwacji zjawisk, lecz tylko do ich wykrywania; do szybkich obserwacji najlepiej nadadzą się właśnie małe teleskopy, które są w stanie szybko zareagować i monitorować interesujące zdarzenia przez kolejne godziny, dni czy tygodnie.

Z tego powodu wykorzystanie sieci neuronowych w przetwarzaniu danych z niewielkich teleskopów naziemnych stanowi istotny obszar badawczy, który wymaga dalszych prac rozwojowych. Rozwiązania stosowane z wykorzystaniem obrazów wysokorozdzielczych z dużych teleskopów nie muszą dawać równie dobrych rezultatów, jeżeli zostaną wykorzystane na danych z małych teleskopów, gdzie rozdzielczość i jakość sygnału są często ograniczone przez sprzęt i warunki obserwacyjne. Zgodnie z twierdzeniem o nieistnieniu darmowych obiadów (ang. *No Free Lunch Theorem*) [29] żaden model nie będzie z założenia działał lepiej od pozostałych; jedynie poprzez ocenę działania każdego z nich z użyciem docelowych danych można przekonać się, który rzeczywiście wykazuje większą skuteczność. W przypadku obrazów niskorozdzielczych oznacza to, że nawet względnie proste sieci neuronowe mogą być w stanie osiągnąć zadowalające wyniki.

Zagadnienie to dodatkowo zyskuje na znaczeniu w sytuacjach, gdy zachodzi potrzeba przetwarzania danych od razu w miejscu przeprowadzania obserwacji. Małe, zdalnie sterowane, teleskopy są często instalowane z dala od miast i źródeł energii elektrycznej. W efekcie, podłączane jest do nich zasilanie akumulatorowe, które powinno zapewnić względnie długi czas pracy w terenie. Należy zatem zadbać również o efektywność energetyczną stosowanych rozwiązań, czyli możliwie zmniejszyć ich złożoność obliczeniową lub zwiększyć ich skuteczność.

Biorąc pod uwagę opisany aspekt badawczy, została sformułowana teza niniejszej pracy: **Przetwarzanie obrazów astronomicznych za pomocą sieci neuronowych może prowadzić do poprawy jakości pomiarowej (detekcji obiektów, pomiaru ich jasności czy położenia) w porównaniu do dotychczas stosowanych algorytmów deterministycznych. Dodatkowo, zastosowanie takich rozwiązań może prowadzić do istotnego spadku zapotrzebowania na zasoby sprzętowe lub do zwiększenia szybkości przetwarzania danych.**

W ramach pracy, zrealizowano prace badawcze, które można podzielić na cztery części, z których każda opisana jest w późniejszych rozdziałach pracy:

I. Do badań wybrano architektury sieci typu autoenkoder. Ich działanie opiera się na kompresji danych wejściowych i ich dekompresji. Zmieniając strukturę sieci, można wpływać na poziom tej kompresji, a co za tym idzie – na ilość traconej informacji. Uznano, że obecny na obrazach losowy szum można potraktować jako element szczegółowy, najtrudniejszy do odtworzenia, a przez to najłatwiejszy do utracenia. Założenie to zweryfikowano z użyciem danych syntetycznych ze zbioru MNIST, który na potrzeby tego eksperymentu uznano za wystarczający. Równocześnie przetestowano wpływ różnych podejść do trenowania sieci na ich wydajność. Doświadczenie to było o tyle istotne, że w przypadku danych z teleskopów naziemnych nie da się uzyskać idealnych obrazów referencyjnych, czyli pozbawionych szumu. W związku z tym należało w pośredni sposób oszacować przydatność innych podejść na tle podejścia tradycyjnego, wykorzystującego dane niezaszumione.
II. Na podstawie wyników eksperymentu na danych syntetycznych przeprowadzono kolejne eksperymenty, tym razem z wykorzystaniem danych z obrazowania nocnego nieba. Do porównania zastosowano powszechnie uznane architektury sieci, które były pierwotnie testowane na obrazach wysokorozdzielczych. Porównano wyniki

detekcji gwiazd, ich położenia oraz jasności na obrazach oryginalnych oraz na obrazach przetworzonych przez sieci neuronowe.

III. Przeprowadzono badania nad przetwarzaniem serii obrazów słonecznych. Celem badań było wyznaczenie wielkości sieci wystarczającej do osiągnięcia rezultatów porównywalnych z wynikami działania deterministycznego algorytmu MFBD, jednakże w znacznie krótszym, quasi-rzeczywistym czasie.

IV. W zastosowaniach przemysłowych często stosuje się rozwiązania polegające na stworzeniu dobrze działającej dużej sieci, a następnie zredukowaniu jej rozmiaru, by mogła zostać uruchomiona na bardziej kompaktowym sprzęcie. Wykorzystując sieci przetestowane w przetwarzaniu obrazów Słońca, zweryfikowano wpływ techniki przycinania (ang. *pruning*) sieci na złożoność struktury modeli i ich wydajność w porównaniu do sieci oryginalnych.

2. Obserwacje astronomiczne w praktyce

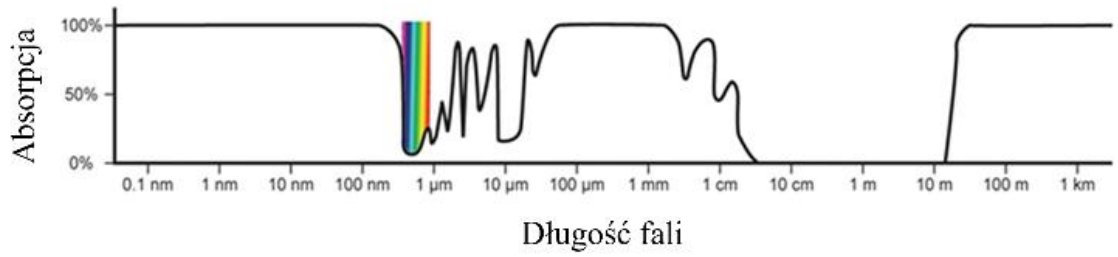
Obserwacje astronomiczne stanowią szczególny obszar nauki, w którym nie ma bezpośredniego dostępu do analizowanych obiektów. W przeciwieństwie do innych dziedzin, astronomia opiera się niemal wyłącznie na analizie sygnałów docierających do nas z odległych ciał niebieskich głównie pod postacią promieniowania elektromagnetycznego. Z tego powodu kluczowe jest zadbanie o prawidłowy odbiór i rejestrację tychże sygnałów. To od jakości i precyzji procesu akwizycji danych zależy, jak dużo jesteśmy w stanie dowiedzieć się o otaczającym nas Wszechświecie.

Proces ten może być realizowany przez różne rodzaje instrumentów obserwacyjnych (teleskopy naziemne, teleskopy kosmiczne, radioteleskopy), których praca opisywana jest z wykorzystaniem specjalistycznej nomenklatury. Zadaniem tego rozdziału jest przybliżyć czytelnikowi dziedzinę astronomii obserwacyjnej w stopniu pozwalającym w pełni zrozumieć stojące przed nią wyzwania, zagadnienia i potrzeby istotne w kontekście dalszej części pracy.

Rozdział podzielony został na trzy główne części. W pierwszej z nich opisane są różne rodzaje obserwacji astronomicznych. Poprzez rodzaje rozumieć należy podział obserwacji ze względu na obserwowany zakres promieniowania, cel obserwacji i użyty sprzęt. Ponownie podkreślona jest rola małych teleskopów, które coraz częściej łączone są w pracujące synchronicznie sieci. Druga część skupia się na opisaniu głównych problemów technicznych związanych z poprawną rejestracją sygnału – wpływem atmosfery, odległością od terenów mieszkalnych i przemysłowych, a także szumami matryc rejestrujących sygnał świetlny. Trzecia część rozdziału przybliża kluczowe pojęcia związane z dziedziną. Bez ich wcześniejszego wyjaśnienia dalsze fragmenty mogłyby okazać się trudne do zrozumienia lub zostać błędnie zinterpretowane – przykładowo: mówiąc o rozdzielczości zarejestrowanych obrazów należy mieć na uwadze, że rozdzielczość optyczna teleskopu jest czymś innym niż rozdzielczość matrycy rejestrującej sygnał. Na końcu rozdziału znajduje się podrozdział, w którym porównano ze sobą dwa obserwatoria astronomiczne. Jednym z nich jest obserwatorium Roque de los Muchachos znajdujące się na Wyspach Kanaryjskich, a drugim są rozproszone obserwatoria grupy naukowej działającej na Politechnice Śląskiej, które zostały wykorzystane do zebrania obrazów wykorzystanych w tej pracy. Porównanie to ma na celu podkreślenie różnic pomiędzy obserwatoriami wykorzystującymi duże oraz małe instrumenty optyczne.

2.1. Rodzaje obserwacji astronomicznych

Życie na Ziemi jest w olbrzymiej mierze możliwe dzięki obecności atmosfery, która utrzymuje odpowiednie ciśnienie i temperaturę planety, wspomaga krążenie wody, ale przede wszystkim, niczym parasol ochronny, absorbuje promieniowanie kosmiczne. Z punktu widzenia astronoma wiąże się to jednak z pewną niedogodnością – atmosfera, skutecznie blokując pewne pasma promieniowania, uniemożliwia ich odbiór na powierzchni Ziemi. Na rysunku 2.1 przedstawiony jest przybliżony wykres absorpcji, czyli nieprzepuszczalności atmosfery, w zależności od długości fali elektromagnetycznej. Jak można zauważyć, promieniowanie gamma i promieniowanie rentgenowskie są całkowicie blokowane i dopiero światło UV jest w stanie w niewielkim stopniu przedostać się przez atmosferę. W dużym stopniu dociera do Ziemi zakres światła widzialnego i podczerwień, przy czym dla podczerwieni absorpcja jest wyraźnie nierównomierna. Bez problemów do powierzchni dochodzą jedynie fale radiowe o długości od kilku centymetrów do kilkunastu metrów, natomiast długie fale radiowe są ponownie całkowicie blokowane.



Rysunek 2.1. Absorpcja promieniowania elektromagnetycznego przez atmosferę.
(*https://commons.wikimedia.org/wiki/File:Atmospheric_electromagnetic_transmittance_or_opacity.jpg*, dostęp: 1.09.2025 r.).

W efekcie, możemy dokonać podziału obserwacji astronomicznych na kilka rodzajów, w zależności od takich czynników jak zakres obserwowanego promieniowania (optyczny, podczerwony, radiowy, etc.) czy miejsce przeprowadzenia obserwacji. To drugie pojęcie należy rozumieć jako „obszar w przestrzeni", z którego prowadzimy obserwacje. Mogą to być obserwacje naziemne, kosmiczne (orbitalne), a także wykonywane za pomocą specjalnie przygotowanych balonów i rakiet, działających krótkoterminowo na granicy atmosfery.

Zasadniczo, większość obserwacji, poza falami radiowymi, najlepiej byłoby przeprowadzać za pomocą teleskopów kosmicznych. Jednakże, koszty związane z budową

takiego sprzętu i umieszczeniem go na orbicie są bardzo wysokie; sam proces jest przy tym czasochłonny, a możliwości naprawy czy korekty takiego sprzętu są niezwykle ograniczone. W związku z tym, olbrzymią rolę w obserwacjach wciąż odgrywają obserwatoria naziemne. Najwięcej uwagi i zainteresowania przyciągają największe z nich, między innymi Europejskie Obserwatorium Południowe, obserwatorium Roque de los Muchachos czy obserwatoria na Mauna Kea.

Budowa i utrzymanie takich jednostek, choć mniej kosztowne niż w przypadku teleskopów pozaziemskich, wymaga pewnych nakładów finansowych i pracy ludzkiej. Ich liczba jest przez to ograniczona, a harmonogram obserwacji zaplanowany jest z wyprzedzeniem na długi czas. Dodatkowo, duże teleskopy mają dosyć ograniczone pole widzenia, przez co są w stanie obserwować jedynie niewielkie fragmenty nieba. Rozwiązaniem tego problemu jest wykorzystanie dużej liczby mniejszych, zdalnie sterowanych teleskopów, które mogą być rozstawione w dogodnych punktach na powierzchni Ziemi, skąd są w stanie monitorować (łącznie) znacznie większe połacie nieba. Podejście to zaproponowane zostało już lata temu i przybrało formę oficjalnych sieci zrzeszających, głównie niewielkie, teleskopy optyczne, takie jak Globalna Sieć Obserwatoriów Hα (ang. *Global Hα Network*) [30] czy GONG (ang. *Global Oscillation Network Group*) [31] w przypadku obserwacji Słońca. Oprócz takich oficjalnych sieci badawczych, wiele projektów naukowych skupia grupy pomniejszych obserwatoriów, które okresowo, w miarę swoich możliwości, włączają się w szerzej zakrojone obserwacje. Wiele z takich projektów stanowi istotne wsparcie dla szerzej zakrojonych badań, czego przykładem może być projekt ExoClock, w którym autor miał przyjemność brać udział [32]. Projekt ten zrzesza dziesiątki obserwatorów, naukowców i amatorów, w celu obserwacji spodziewanych tranzytów egzoplanet, czyli momentów, gdy okrążając swoje gwiazdy, planety te przysłaniają w niewielkim procencie światło docierające do obserwatora na Ziemi. Potwierdzone tranzyty zapisywane są w formie efemeryd zawierających informacje o dokładnych, przewidywanych momentach kolejnych tranzytów. Tablice te następnie zostaną wykorzystane w misji ARIEL (ang. *Atmospheric Remote-sensing Infrared Exoplanet Large-survey*), której głównym celem będzie badanie atmosfer zaobserwowanych planet pozasłonecznych. Rola niewielkich teleskopów w tym i podobnych projektach jest zatem znacząca i będzie decydować o sukcesie przyszłych, kosztownych misji kosmicznych.

Można wobec tego śmiało stwierdzić, że współczesna astronomia w dalszym ciągu opiera się na obserwacjach przeprowadzanych z powierzchni Ziemi. Jednocześnie liczne projekty, takie jak wspomniany ExoClock, udowadniają, że nawet relatywnie niewielkie

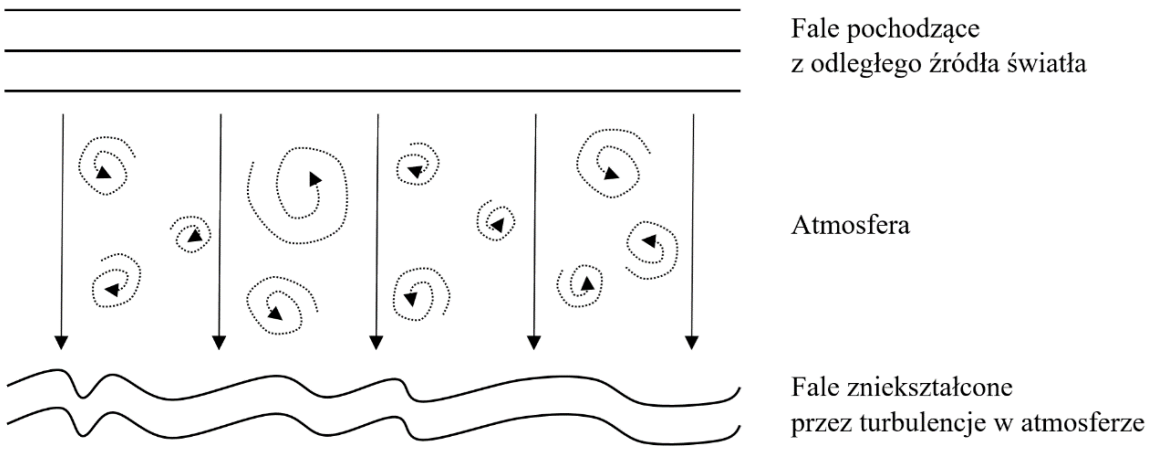
teleskopy są w stanie wnieść znaczny wkład w wiele misji i badań. Możliwość organizowania ich w gęste sieci, które są w stanie prowadzić synchroniczne i ciągłe obserwacje nieba czyni je w wielu przypadkach niezastąpionym elementem współczesnej infrastruktury badawczej. Z tego powodu niezwykle istotne jest podjęcie działań mających na celu jak największą poprawę pozyskiwanych przez nie danych tak, by całkowicie wykorzystać oferowane przez nie możliwości.

2.2. Czynniki wpływające na jakość danych obserwacyjnych

Obserwacje prowadzone z Ziemi borykają się z licznymi problemami wpływającymi na jakość pozyskiwanych danych. Najbardziej oczywistym z nich jest pogoda – przy dużym zachmurzeniu, często połączonym z opadami, obserwacje optyczne w zasadzie nie mają sensu. Dodatkowo, istotne jest zagadnienie doboru odpowiedniego miejsca obserwacji. Chcąc obserwować Słońce, powinniśmy umieścić aparaturę w lokalizacji zapewniającej jak najwyższą liczbę pogodnych dni w roku. Należy przy tym spełnić również inne wymagania związane z poprawną pracą wykorzystywanych instrumentów, w tym zadbać o ich odpowiednie chłodzenie, by zminimalizować możliwe szumy pomiarowe. W kolejnych częściach tego podrozdziału opisane są główne utrudnienia związane z pomiarami optycznymi wykonywanymi przez niewielkie teleskopy, czyli szum atmosferyczny oraz fotonowy (związany między innymi z zanieczyszczeniem nieba światłem). Przybliżono także podstawowe błędy występujące w pracy kamer rejestrujących obraz.

2.2.1. Szum atmosferyczny

Atmosfera ziemska nie jest powłoką jednorodną, lecz składa się z następujących warstw: troposfery, stratosfery, mezosfery, termosfery i egzosfery. Każda z nich cechuje się innymi właściwościami, takimi jak ciśnienie, temperatura czy grubość. Nie są one przy tym statyczne, lecz cechują się dużą dynamiką – ciągle występują w nich ruchy mas powietrza. W rezultacie sygnał przez nie przechodzący podlega licznym zniekształceniom wynikającym z refrakcji światła, jak zaprezentowano na rysunku 2.2. Zniekształcenia te powodują pozorne przesunięcia obiektów na rejestrowanych obrazach oraz losowe pojaśnienia i przyciemnienia pojedynczych pikseli, jak i całych ich grup.

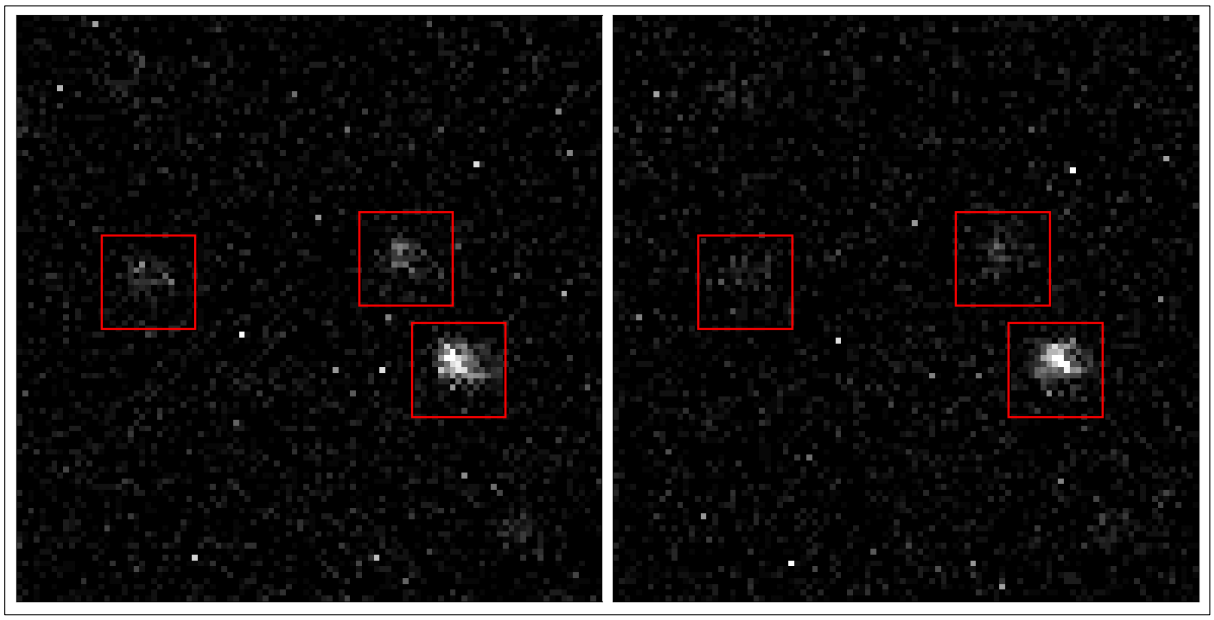


Rysunek 2.2. Wpływ atmosfery na odbierany sygnał.

Co istotne, zniekształcenia te bardzo szybko zmieniają się w czasie (dziesiątki lub nawet setki razy w ciągu sekundy, w zależności od warunków atmosferycznych). Na rysunku 2.3 przedstawiono dwa zdjęcia tego samego fragmentu nieba w odstępie 2 sekund. Czerwonymi ramkami zaznaczono 3 gwiazdy znajdujące się w tym obszarze – jedną świecącą jaśniej i dwie ciemniejsze. Jak można zauważyć, gwiazdy nieco zmieniły swoje położenie, a także zmieniła się jasność poszczególnych pikseli, co jest dużym problemem, gdy chcemy precyzyjnie zmierzyć jasność i pozycję obserwowanych obiektów.

Problem zarejestrowanego szumu atmosferycznego można zniwelować na kilka sposobów. Najprostszym z nich jest wydłużenie czasu obserwacji danego wycinka nieba, co najlepiej zrobić poprzez wykonanie serii wielu zdjęć, z których wylicza się następnie obraz uśredniony o wyższym stosunku sygnału do szumu (dokładnie opisane jest to w: *2.3.6. Technika łączenia klatek (stacking)*). Metoda ta jest jednak zawodna w sytuacjach, gdy chcemy zaobserwować obiekt zmieniający się bardzo dynamicznie, na przykład bliską asteroidę, obiekt satelitarny lub pulsar, którego jasność fluktuuje dziesiątki razy na sekundę. Bardziej wymagającą, a zarazem lepszą, opcją jest montaż teleskopu na dużej wysokości w górach. Atmosfera jest tam zdecydowanie rzadsza, a także bardziej stabilna, co ogranicza wpływ turbulencji na zbierane dane. Sposobem na zmniejszenie względnych, dynamicznych zmian jasności jest również zwiększenie rozmiaru lustra teleskopu, ponieważ wówczas następuje efektywniejsze uśrednianie wiązki światła niż w przypadku niewielkich powierzchni małych teleskopów optycznych.

Osobną grupę rozwiązań stanowią rozwiązania sprzętowe, spośród których najbardziej zaawansowana jest optyka adaptacyjna [33][34][35]. Technologia ta pozwala w czasie rzeczywistym korygować wpływ atmosfery, dzięki czemu pozyskiwane obrazy niewiele ustępują tym uzyskiwanym przez podobnej klasy teleskopy kosmiczne. Jej działanie skupia się na obserwacji gwiazdy odniesienia (naturalnej lub tworzonej sztucznie przy pomocy specjalnego lasera) i pomiarach jej deformacji przez atmosferę przy użyciu oddzielnych sensorów (czujnik czoła falowego). Wyniki tych pomiarów są w czasie rzeczywistym wykorzystywane do odkształcenia tak zwanego lustra deformowalnego, które koryguje sygnał odbierany przez teleskop. Technologia ta jest jednak zbyt droga i zbyt złożona, by znaleźć zastosowanie w przypadku niewielkich teleskopów.



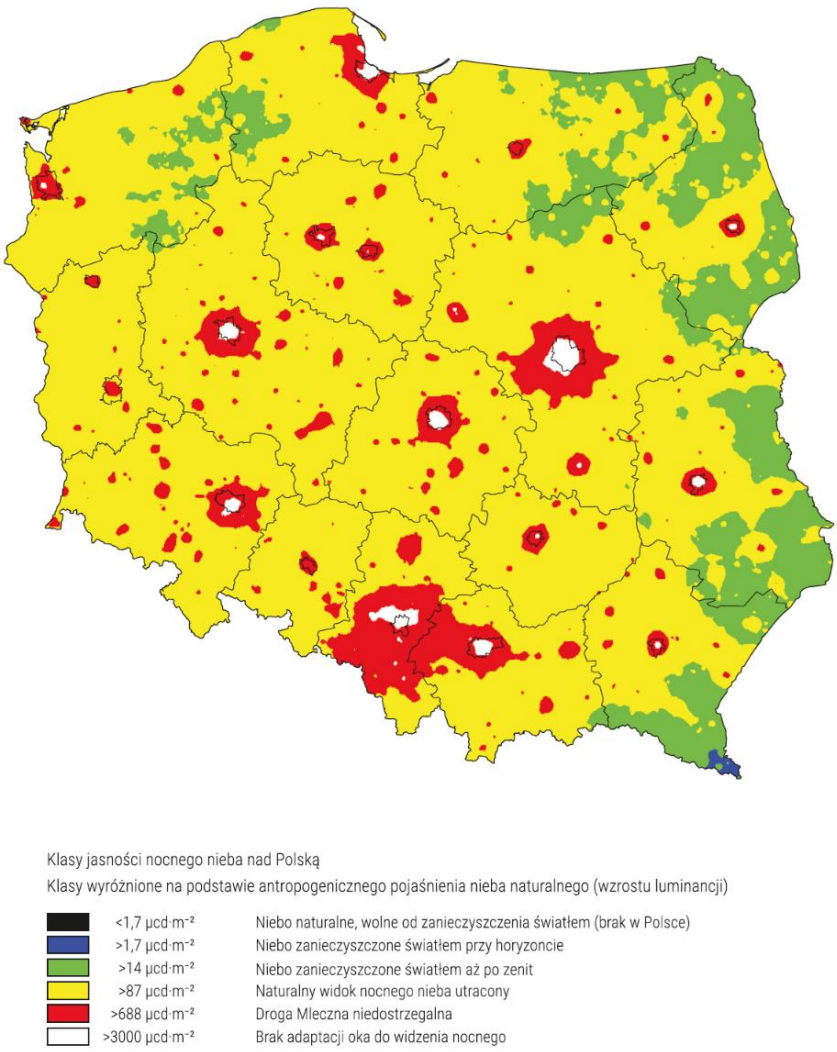
Rysunek 2.3. Zniekształcenie danych obrazowych przez szum atmosferyczny na przykładzie dwóch zdjęć wykonanych w odstępie 2 sekund.

2.2.2. Zanieczyszczenie światłem

Dużym problemem przy obserwacjach astronomicznych jest także zanieczyszczenie światłem. Jako skutek elektryfikacji, pojawiło się mnóstwo źródeł sztucznego światła, takich jak lampy uliczne, reklamy świetlne, czy oświetlenie budynków. Ich niewłaściwe i nadmiarowe użycie sprawia, że w licznych obszarach występuje znaczące rozjaśnienie tła nocnego nieba. Pojaśnienie to określa się przy pomocy luminancji, która jest miarą wartości strumienia światła

emitowanego przez określoną powierzchnię w zadanym kącie bryłowym. W układzie SI jednostką luminancji jest kandela na metr kwadratowy (cd/m^2). Im wyższa jej wartość, tym jaśniejsze niebo, a co za tym idzie – większy udział szumu tła, który skutecznie utrudnia detekcję ciemnych obiektów w obrazach astronomicznych.

W raporcie „Zanieczyszczenie światłem w Polsce. Raport 2023" [36] przedstawiono wyniki pomiarów jasności nocnego nieba nad Polską na przestrzeni 2022 roku. Wynika z niego, że na obszarze całego kraju następuje zjawisko zanieczyszczenia światłem. Na rysunku 2.4 przedstawiono mapę zanieczyszczenia światłem zamieszczoną w oryginalnym raporcie. Jak można zauważyć, najjaśniejsze niebo występuje nad gminami wchodzącymi w skład aglomeracji warszawskiej i konurbacji górnośląskiej, natomiast najciemniejsze są obszary przy wschodniej granicy kraju, w szczególności w Bieszczadach.



Rysunek 2.4. Zanieczyszczenie światłem polskiego nieba [36].

Aby zminimalizować wpływ zanieczyszczenia światłem, obserwacje astronomiczne najlepiej prowadzić z miejsc odludnych i słabo zaludnionych, gdzie panują odpowiednio ciemne warunki sprzyjające precyzyjnym pomiarom. Lokacje te są jednak zazwyczaj trudno dostępne, co ogranicza możliwość bezpośredniego prowadzenia obserwacji. Rozwiązaniem może być montaż niewielkich teleskopów, które mogą być sterowane zdalnie z dogodnego miejsca. Wymaga to jednak zapewnienia niezależnego źródła zasilania, najczęściej w postaci akumulatorów, oraz szczególnego zwrócenia uwagi na efektywność energetyczną stosowanego sprzętu, aby maksymalnie wydłużyć czas jego pracy w terenie.

2.2.3. Szum związany z działaniem aparatury obserwacyjnej

W kamerach astronomicznych występują zarówno matryce CCD (ang. C*harge-Coupled Device*), jak i coraz częściej zastępujące je matryce CMOS (ang. *Complementary Metal-Oxide-Semiconductor*). Matryce te składają się z pikseli, z których każdy działa osobno względem pozostałych. Pod wpływem efektu fotoelektrycznego, fotony padające na piksele powodują generowanie w nich ładunku elektrycznego proporcjonalnego do liczby fotonów. Ładunek ten przekazywany jest do struktury kondensatora, dzięki czemu wytworzone napięcie może być w efekcie wzmocnione i skonwertowane na wartość cyfrową przez przetwornik analogowo-cyfrowy. Proces ten wiąże się z powstawaniem różnego rodzaju szumów, które wpływają na jakość zapisywanych obrazów końcowych.

Podstawowym rodzajem takiego szumu jest szum odczytu związany z działaniem elektroniki odczytowej – tranzystorów, wzmacniaczy i przetworników. Jego wartości są zazwyczaj określone dla danego sprzętu i nie zależą od poziomu odbieranego sygnału [37]. W nowoczesnych matrycach CMOS jest on zazwyczaj niski, jednak w warunkach bardzo słabego oświetlenia może się zdarzyć, że wartość szumu odczytu będzie większa od samego sygnału, który zostanie w efekcie zasłonięty. Zazwyczaj skutkuje to pojawieniem się losowego ziarna i drobnych plamek.

Innym rodzajem szumu jest szum termiczny, zwany także prądem ciemnym. Szum ten nie zależy od ilości padającego światła, lecz od czasu naświetlania matrycy. Wysoka temperatura matrycy powoduje drgania struktury krystalicznej półprzewodników, które mogą dostarczyć energii potrzebnej do samoistnego powstania elektronów w pikselach [38]. W związku z tym, nawet nieoświetlone piksele mogą generować fałszywy sygnał. Otrzymane zdjęcie w swojej strukturze ma piksele o większym (jaśniejsze) oraz o mniejszym (ciemniejsze)

tempie generacji termicznej, tworząc innego rodzaju efekt ziarna. Szum tego typu potocznie bywa określany jako obecność „gorących pikseli", których struktura, pod wpływem obecności atomów obcych lub defektów struktury krystalicznej, wykazuje silną generację termiczną.

Trzecią podstawową składową zakłóceń matrycy CMOS jest szum fotonowy (kwantowy) wynikający z losowej natury procesu zliczania fotonów w krzemie. Pojawia się on nawet w warunkach idealnego, stałego oświetlenia, ponieważ absorpcja fotonów jest procesem probabilistycznym podlegającym rozkładowi Poissona [39]. Chociaż jego wartość wzrasta wraz z poziomem odbieranego sygnału, dzieje się to proporcjonalnie wolniej, w efekcie czego przy większej jasności stosunek sygnału do szumu jest większy. Przykładowo, proces pomiaru strumienia świetlnego, generujący średnio 100 elektronów w ekspozycji trwającej jedną sekundę, będzie generował w kolejnych ekspozycjach ładunek o wartości oczekiwanej 100 elektronów, jednak z rozrzutem w przybliżeniu równym pierwiastkowi ze 100.

Oprócz tego, ważnym źródłem zakłóceń jest niejednorodność czułości pikseli względem padającego na nie światła PRNU (ang. *Photo Response Non-Uniformity*). Wynikające z tego różnice między pikselami są zwykle małe, rzędu 1-2%, jednakże mogą być one zauważalne przy zdjęciach o niskim kontraście, gdy zgromadzone zostaną duże ilości ładunku [40]. Problem PRNU, jako wady fabrycznej, często łączy się też z dwoma innymi czynnikami powodującymi nierównomierne oświetlenie matrycy. Pierwszym z nich są fizyczne zabrudzenia takiej jak kurz osadzający się na matrycy i filtrach kamery. Drugim z nich jest winietowanie, czyli niedoświetlanie brzegów kadru, wynikające z niedoskonałości optyki. Obecność PRNU może być bardziej widoczna w kamerach CMOS niż CCD, bo w ich pikselach wbudowane są dodatkowe układy pomiarowe lub wtórniki, co skutkować może wzrostem niejednorodności czułości obszaru aktywnego optycznie w każdym z pikseli.

Wymienione powyżej źródła szumu można w dużym stopniu zredukować poprzez odpowiednie chłodzenie matrycy, a także przez takie operacje jak łączenie klatek oraz wykorzystanie klatek kalibracyjnych. Więcej informacji na ten temat znajduje się w rozdziałach *2.3.5. Kalibracja surowych danych* i *2.3.6. Technika łączenia klatek (stacking)*.

2.3. Kluczowe pojęcia astronomiczne

W kolejnych rozdziałach niniejszej pracy zostały wykorzystane terminy charakterystyczne dla dziedziny astronomii obserwacyjnej. Ze względu na potencjalną trudność w ich właściwej interpretacji, poniżej przedstawiono i wytłumaczono kluczowe pojęcia.

2.3.1. Czas ekspozycji i kadencja

Czasem ekspozycji nazywamy długość czasu, przez jaki matryca rejestruje światło padające od obserwowanego obiektu. Im dłuższy ten czas, tym więcej sygnału jesteśmy w stanie zgromadzić i tym więcej słabiej świecących obiektów jesteśmy w stanie odróżnić od tła. Czas ten różni się w zależności od przeprowadzanych obserwacji i może wynieść od kilkudziesięciu milisekund do wielu godzin. W przypadku obrazów przetwarzanych w ramach opisanych badań czas ekspozycji wynosił od $^1/_{30}$ do $^1/_2$ sekundy. Czas kadencji opisuje natomiast czas pomiędzy początkiem jednej ekspozycji a początkiem następnej. Jest to suma czasu pojedynczej ekspozycji, czasu odczytu i zapisu danych, a także czasu przestoju, który może być z góry narzuconym interwałem albo zmieniającym się podczas eksperymentu czasem potrzebnym na korektę ustawienia teleskopu.

W przypadku obserwacji szybko zachodzących zjawisk, jak wspomniane tranzyty egzoplanet czy rozbłyski słoneczne, należy uzyskać możliwie krótką kadencję, by dobrze zarejestrować dynamikę zjawiska. Natomiast w przypadku bardziej szczegółowych obserwacji głębokiego nieba najistotniejszy jest długi czas ekspozycji pozwalający dostrzec najsłabiej świecące ciała niebieskie.

2.3.2. Rozdzielczość teleskopu i rozdzielczość matrycy

Omawiając rozdzielczość obrazowania w astronomii, należy podkreślić istotną różnicę między rozdzielczością optyczną teleskopu a rozdzielczością matrycy kamery rejestrującej obraz. Oba te parametry wpływają na jakość końcowego obrazu, jednak ich znaczenie i sposób wyznaczania są zupełnie różne.

Rozdzielczość optyczna teleskopu (oznaczana jako θ) to zdolność instrumentu do rozróżnienia dwóch blisko położonych siebie obiektów jako oddzielne punkty. Rozdzielczość ta zależy od długości fali światła λ i średnicy apertury (lustra lub soczewki) D, tak jak

przedstawiono w równaniu 2.1 (wartość θ we wzorze często określa się granicą Rayleigha). Wynik równania podany jest w radianach, więc by przeliczyć go na sekundy kątowe, należy go pomnożyć przez 206 265. Przyjmując $\lambda \approx 550$ nm (światło zielone) i aperturę o średnicy 200 mm, wielkość ta wyniesie w przybliżeniu 0,69”. Wartość θ maleje proporcjonalnie wraz ze wzrostem średnicy apertury.

$$\theta = 1{,}22\frac{\lambda}{D} \tag{2.1}$$

Rozdzielczość matrycy odpowiada natomiast liczbie w dwóch wymiarach (np_x i np_y) i wielkości pikseli p_x w detektorze, informując o tym, jak szczegółowo kamera może zarejestrować obraz. Przy jej pomocy można wyznaczyć kątową skalę piksela S_{px}, która zależy także od długości ogniskowej f_{ogn} (wzór 2.2). Przykładowo dla kamery o wielkości piksela 2,4 μm i długości ogniskowej teleskopu równej 500 mm, skala piksela wyniesie 0,99”/px. Istotną kwestią jest taki dobór instrumentów, by skala kątowa piksela była mniejsza od granicy Rayleigha. Dzięki temu kamera przeprowadza akwizycję z odpowiednim próbkowaniem obrazu, co pozwala optymalnie wykorzystać możliwości sprzętowe.

$$S_{px} = 206\,265\frac{p_x}{f_{ogn}} \tag{2.2}$$

2.3.3. Pole widzenia

W przypadku astronomii pole widzenia (ang. *Field of View*, FOV) definiujemy jako kątowy obszar nieba, który można zobaczyć przez teleskop w danym momencie. W przypadku teleskopów o długiej ogniskowej, wartość ta wyrażana jest w minutach lub sekundach kątowych, a dla teleskopów szerokokątnych typowe wartości wyraża się zazwyczaj w stopniach kwadratowych. Pole widzenia najczęściej opisuje się w dwóch wymiarach, przeliczając liczbę pikseli w danym rozmiarze przez skalę piksela (wzór 2.3).

$$FOV_x = np_x * S_{px} \qquad FOV_y = np_y * S_{px} \tag{2.3}$$

2.3.4. Filtracja sygnału

W przypadku obserwacji ciał niebieskich bardzo często stosuje się filtry optyczne, by selektywnie rejestrować światła emitowane przez konkretne pierwiastki chemiczne. Podejście to umożliwia analizę właściwości jak skład chemiczny, temperatura czy dynamika gazów wybranych obiektów. Przykładowo, w obserwacjach słonecznych często wykorzystuje się wąskopasmowy filtr Hα przepuszczający światło o długości 656,28 nm w celu dokładnego obrazowania emisji światła ze zjonizowanego wodoru. Dzięki temu zabiegowi możliwe są obserwacje protuberancji, filamentów i granulacji chromosfery Słońca, które w świetle białym są niedostrzegalne. W pewnych zastosowaniach stosuje się także filtry szerokopasmowe do oszacowania całkowitej jasności obiektów. W zastosowaniach komercyjnych szerokości pasma są często określane w angstremach Å, jednostce długości równej 0,1 nm, nazwanej tak dla upamiętnienia szwedzkiego astronoma, Andersa Jönasa Ångströma.

2.3.5. Kalibracja surowych danych

W przypadku obrazowania astronomicznego pojedyncze, surowe zdjęcie zarejestrowane przez kamerę nazywa się klatką. Jak wyjaśniono w podrozdziale *2.2.3. Szum związany z działaniem aparatury obserwacyjnej*, każda z nich oprócz zarejestrowanego sygnału zawiera różnej natury niejednorodności i odchyłki. Ich wpływ na obrazy wynikowe redukowany jest z użyciem specjalnie przygotowanych klatek kalibracyjnych.

W przypadku problemu niejednorodnej czułości pikseli na światło stosuje się klatki płaskie (ang. *flat field frames*). Uzyskuje się je poprzez wykonanie zdjęcia przy jednolitym oświetleniu, najczęściej wykorzystując do tego tzw. flatownice, czyli urządzenia zakrywające teleskop i równomiernie naświetlające całą powierzchnię lustra [40]. Każda z klatek obserwacyjnych zostaje przemnożona przez odpowiednią wartość obliczaną na podstawie takiej klatki płaskiej, dzięki czemu możliwe staje się porównanie jasności obiektów rozmieszczonych w różnych obszarach zdjęcia końcowego. Innymi słowy, niejednorodna transmisja światła do płaszczyzny obrazu zostaje odpowiednio skompensowana przez efekt mnożenia.

Do kalibracji obrazów astronomicznych z matryc CCD/CMOS używa się także klatek ciemnych (ang. *dark frames*). Są to klatki uzyskiwane poprzez rejestracje obrazu bez udziału światła, zazwyczaj poprzez zakrycie teleskopu lub nieodsłonięcie migawki w kamerze. Obrazy

kalibracyjne tego typu powinny być wykonane z takim samym czasem ekspozycji oraz taką samą, stabilną temperaturą matrycy jak w przypadku planowanych obserwacji. Klatki te następnie odejmuje się od wszystkich klatek obserwacyjnych, by zredukować wpływ szumu prądu ciemnego. Z racji tego, że szum ten cechuje się rozkładem Poissona, zaleca się wykonanie wielu klatek ciemnych i ich uśrednienie (metoda opisana w kolejnym podrozdziale: *2.3.6. Technika łączenia klatek (stacking)*), by uzyskać średnią klatkę ciemną. W przypadku różnic czasów ekspozycji zdjęcia świetlnego i klatki ciemnej, wystarczającym okazuje się odpowiednie wymnożenie klatki ciemnej, gdyż przyrost prądu ciemnego w pikselach wraz z czasem ma charakter liniowy.

2.3.6. Technika łączenia klatek (stacking)

Każda pojedyncza klatka K_i zawiera w sobie pochodzący z obiektu sygnał S oraz losowy szum N_i (wzór 2.4). Zakładając, że ten szum ma wartość oczekiwaną równą zero i pewien rozrzut σ, możemy dokonać jego skutecznej redukcji poprzez *stacking*, czyli uśrednienie wielu osobnych klatek. W większości wypadków *stacking* jest równoznaczny jedynie z uśrednieniem zebranych danych, jednak w niektórych sytuacjach nazywany jest tak cały proces wyrównania danych, ich kalibracji, wyboru klatek o najlepszej jakości (*stacking selektywny*) i ich uśrednienia, a także dalszej obróbki.

$$K_i = S + N_i \tag{2.4}$$

Taki obraz uśredniony opisać można jako sumę sygnału i średnią wartość szumu pojawiającego się w trakcie obserwacji (równanie 2.5). W efekcie wariancja szumu w klatce uśrednionej spada wraz ze wzrostem liczby uśrednianych klatek n (wzór 2.6). Oznacza to, że stosunek wartości sygnału do szumu rośnie proporcjonalnie do $\sqrt{n}$.

$$\overline{K}_n = \frac{1}{n}\sum_{i=1}^{n}(S + N_i) = S + \frac{1}{n}\sum_{i=1}^{n} N_i \tag{2.5}$$

$$\mathrm{Var}(\overline{K}_n) = \frac{\sigma^2}{n} \tag{2.6}$$

Nie oznacza to jednak, że można w ten sposób całkowicie wyeliminować szum atmosferyczny; na skutek uśredniania jego wpływ jest redukowany jednak w przypadku atmosfery dochodzą również długozmienne fluktuacje, których nie sposób uśrednić. Dodatkowo, metoda ta sprawdza się najlepiej w redukcji losowych zmian jasności w poszczególnych pikselach, jednak wiąże się to z rozmyciem obrazu w klatce uśrednionej; rozmycie to jest z reguły tym większe, im mniej klatek zostało użytych do uśrednienia. Jest to związane ze zmiennym w czasie zniekształceniem sygnału, które objawia się powstaniem pozornych przesunięć obiektów na rejestrowanych zdjęciach.

2.3.7. Astrometria i fotometria

Obserwując niebo, astronomowie najczęściej zadają dwa pytania: „gdzie znajduje się poszukiwany przez nich obiekt i jak zmienia się jego położenie” oraz „jak jasno świeci i jak zmienia się jego jasność”. Odpowiedziami na te pytania zajmują się odpowiednio astrometria i fotometria.

Astrometria zajmuje się określaniem dokładnej pozycji ciał na niebie, jej zmian względem pozycji innych obiektów, a także służy obliczaniu ich orbit czy odległości od Ziemi. Jest to jedna z najstarszych dziedzin astronomii: już w starożytności ludzie śledzili ruchy gwiazd, by tworzyć efemerydy wykorzystywane do odliczania dni roku, a także nawigacji, zwłaszcza w żegludze morskiej. Współcześnie astrometria korzysta z bardzo precyzyjnych instrumentów, które mierzą położenia gwiazd z dokładnością do mikrosekund kątowych, co umożliwia między innymi tworzenie trójwymiarowych map naszej Galaktyki.

Fotometria jest natomiast dziedziną zajmującą się pomiarami jasności ciał niebieskich w różnych zakresach promieniowania elektromagnetycznego z wykorzystaniem filtrów szeroko i wąskopasmowych. Pozwala ona badać właściwości obserwowanych obiektów, takie jak temperatura, skład chemiczny, a także wykrywać zachodzące w czasie zmiany, na przykład zaćmienia. Jednostką używaną w fotometrii do oznaczenia blasku obiektu jest magnitudo (oznaczane jako mag). Określa ona stosunek natężenia światła rozważanego obiektu I_x do natężenia światła wybranego punktu odniesienia I_{ref} (wzór 2.7), za który zwyczajowo przyjmuje się gwiazdę Vega. Zmiana jasności o 1 magnitudo odpowiada zmianie natężenia światła o czynnik równy w przybliżeniu 2,512, a różnica 5 mag odpowiada stukrotnej zmianie jasności. Im mniejsza wartość magnitudo, tym jaśniejszy jest obiekt; przykładowo, wartość magnitudo dla Słońca wynosi w przybliżeniu -27.

$$\mathrm{mag} = -2{,}5 \log_{10}\left(\frac{I_x}{I_{ref}}\right) \tag{2.7}$$

2.4. Wybrane obserwatoria naziemne

Obserwatoria naziemne można umownie podzielić na dwie klasy – zaawansowane obserwatoria wykorzystujące duże teleskopy oraz jednostki wyposażone w instrumenty o niewielkich aperturach. W tym podrozdziale opisano teleskopy słoneczne stanowiące część dwóch obserwatoriów należących do tych oddzielnych klas. Pierwszym z nich jest Szwedzki Teleskop Słoneczny (STS), jeden z najlepszych naziemnych instrumentów tego typu, a drugim jest niewielki teleskop zarządzany przez działającą na Politechnice Śląskiej grupę naukową SUTO (ang. *Silesian University of Technology Observatories*). Zaprezentowane porównanie ma na celu uwypuklenie różnic pomiędzy obserwatoriami rejestrującymi dane wysokorozdzielcze i dane niskorozdzielcze, które stanowią podstawę niniejszej pracy. Dodatkowo zaprezentowano teleskop obrazujący nocne niebo na potrzeby badań prowadzonych przez SUTO. Jest to drugi, i zarazem ostatni, teleskop, przy pomocy którego zebrano dane wykorzystane w opisanych w niniejszej pracy badaniach.

2.4.1. Obserwatorium Roque de los Muchachos

Obserwatorium Roque de los Muchachos położone na wysokości około 2400 m n.p.m. znajduje się na północnym skraju Caldera de Taburiente, na wyspie La Palma na Wyspach Kanaryjskich. Miejsce to charakteryzuje się ciemnym niebem, małym zanieczyszczeniem światłem, a także nieco rzadszą i stabilniejszą atmosferą. Powszechnie uważane za jedno z najlepszych punktów obserwacyjnych na świecie; na półkuli północnej ustępuje jedynie obserwatoriom na Mauna Kea na Hawajach.

Rysunek 2.5. Szwedzki Teleskop Słoneczny.
(*https://commons.wikimedia.org/wiki/File:Swedish_Solar_Telescope.jpg*, dostęp: 1.09.2025 r.)

Na rysunku 2.5 przedstawiono Szwedzki Teleskop Słoneczny [41], jeden z najnowocześniejszych naziemnych teleskopów słonecznych. Instrument ten posiada zwierciadło o średnicy około 1 metra i jest wyposażony w specjalny korektor pozwalający zniwelować aberrację chromatyczną. Z racji dużego rozmiaru tuby optycznej utrzymywana jest w niej próżnia, by przy długich obserwacjach Słońca nie dochodziło do nagrzania powietrza w środku, a w efekcie pogorszenia jakości rejestrowanych danych pod wpływem wewnętrznych turbulencji. Do obrazowania wykorzystywany jest system optyki adaptacyjnej operującej deformowalnym lustrem, które może się odkształcać do 1000 razy na sekundę, a obrazy końcowe są przetwarzane przez specjalne algorytmy zwiększające ich szczegółowość. Dodatkowo, rozdzielczość optyczna teleskopu wynosi w niebieskim świetle wartość 0,1”, co w przypadku Słońca pozwala na rozróżnienie obiektów znajdujących się około 70 km od siebie. Wymienione parametry tego teleskopu bezsprzecznie stawiają go w grupie najlepszych obecnie dostępnych tego typu instrumentów.

2.4.2. Obserwatoria Politechniki Śląskiej

W zupełnie innej sytuacji znajdują się instrumenty zarządzane przez grupę SUTO, które stanowią bardzo dobry przykład małych, a zarazem względnie dostępnych finansowo sprzętów. Po prawej stronie rysunku 2.6 przedstawiony jest teleskop słoneczny operujący z Kotulina, wsi położonej w powiecie gliwickim. Lokalizacja ta wiąże się ze zdecydowanie gęstszą atmosferą niż w przypadku najlepszych lokalizacji dla tego typu obserwatoriów, a także względnie wysokim stężeniem pyłów w powietrzu, które dodatkowo pogłębiają problem odkształceń sygnału. Rozdzielczość optyczna teleskop w obserwowanej linii Hα wynosi natomiast 3,27”, co jest wartością prawie 33 razy większą od tej w STS. Sprawia to, że wykorzystywany teleskop jest w stanie odróżnić obiekty znajdujące się w odległości około 2 370 km względem siebie w chromosferze słonecznej, co stanowi znaczącą różnicę pod względem jakości obserwacji. Należy do tego jeszcze dodać brak zaawansowanych rozwiązań sprzętowych, by móc stwierdzić, że obserwatoria te rejestrują zdecydowanie inny typ danych.

Po lewej stronie rysunku 2.6 znajduje się natomiast zdjęcie teleskopu obrazującego nocne niebo. Teleskop ten jest umieszczony w hiszpańskiej miejscowości o nazwie Otivar, w specjalnie dopasowanej, automatycznie otwieranej kopule. Stanowi on idealny przykład małego teleskopu, który umieszczony w miejscu cechującym się bardzo dobrymi warunkami obserwacyjnymi, może być sterowany zdalnie. Średnica jego apertury wynosi 30 cm, co pozwala na zbieranie ilości światła potrzebnego nawet do wykrycia odległych obiektów. Posiada przy tym dosyć szerokie pole widzenia, dzięki czemu może spełniać swoją rolę w monitorowaniu dużych obszarów nieba. Nie jest oczywiście w stanie konkurować z większymi teleskopami w obserwacjach odległych galaktyk, ale swą wysoką przydatność wykazuje w takich misjach, jak omawiany wcześniej ExoClock. Tak jak w przypadku innych teleskopów naziemnych, jego największą słabością są krótkie ekspozycje, podczas których liczba zliczeń fotonów pochodzących od obiektów jest porównywalna lub mniejsza niż towarzyszący szum pomiarowy.

Reasumując, w przypadku opisanych małych teleskopów możliwości poprawa wyników może być dokonana głównie poprzez zastosowanie nowych rozwiązań algorytmicznych, podczas gdy możliwości sprzętowe są ograniczone. W tej sytuacji naturalnym podejściem wydaje się być wdrożenie metod uczenia maszynowego w postaci sieci neuronowych, która może dodatkowo poprawić jakość pozyskiwanych przez nie danych.

Rysunek 2.6. Teleskopy wykorzystane do akwizycji danych wykorzystanych w niniejszej pracy.
(*https://www.suto.aei.polsl.pl*, dostęp: 1.09.2025 r.)

3. Sieci neuronowe w przetwarzaniu obrazów

Od przeszło 10 lat sieci neuronowe osiągają coraz lepsze wyniki w przetwarzaniu obrazów, a ich wydajność w odwzorowywaniu złożonych zależności przestrzennych oraz wyodrębnianiu istotnych cech obrazu sprawia, że stosowane są w coraz większej liczbie zadań. Wyniki uzyskiwane przy ich użyciu niejednokrotnie są znacząco lepsze niż te otrzymywane z użyciem opracowanych algorytmów deterministycznych i sprawdzonych heurystyk. Modele poszczególnych sieci operują ogromną liczbą przekształceń i parametrów, co może powodować trudności w interpretacji ich działania – często porównywane są do „czarnych skrzynek", które w niewytłumaczalny sposób zwracają pożądane rezultaty. Z perspektywy ich praktycznego zastosowania potrzeba odpowiedniego zrozumienia zachodzących w nich mechanizmów, by móc je prawidłowo projektować, uruchamiać, a także trafnie diagnozować występujące w nich niedoskonałości i być w stanie je modyfikować. Bez tych umiejętności łatwo o wyciągnięcie błędnych wniosków, a także otrzymanie nieefektywnych rozwiązań.

Z tego powodu, ten rozdział pracy ma na celu przybliżyć najważniejsze informacje dotyczące działania sieci neuronowych, które stanowią podstawę niniejszej pracy doktorskiej. Rozdział został podzielony na sześć oddzielnych części. Pierwsza z nich ma na celu wyjaśnić relację pomiędzy sztuczną inteligencją, uczeniem maszynowym a sieciami neuronowymi – co kryją w sobie te pojęcia i kiedy można ich używać zamiennie. Druga część przybliża ewolucję sieci neuronowych: od pierwszych, prostych modeli perceptronów do współczesnych sieci głębokich. Wyjaśnia też podstawowy mechanizm stojący za ich trenowaniem, czyli propagację wsteczną wraz z optymalizacją funkcji straty. Część trzecia w skrócie opisuje środowisko programistyczne, w którym został napisany kod wykorzystany w pracy. Czwarta część koncentruje się na poszczególnych warstwach sieci – przedstawiono sposób ich działania oraz wpływ parametrów wewnętrznych na funkcjonowanie całości modelu. Każda z warstw odpowiada wykorzystaniu pojedynczej funkcji, dlatego do realizacji bardziej skompilowanych zadań łączy się je w złożone struktury, określane architekturami sieci. W piątej części rozdziału opisano dwie wybrane do badań architektury wraz z uzasadnieniem ich użycia. Ostatnia część rozdziału skupia się na ustawieniach zewnętrznych sieci, które nie są bezpośrednio modyfikowane w trakcie procesu uczenia, lecz mają istotny wpływ na jego przebieg i rezultaty, często stanowiąc decydujący czynnik świadczący o skuteczności wybranego podejścia.

3.1. Uczenie maszynowe a sieci neuronowe

Sztuczną inteligencją nazywamy interdyscyplinarny dział nauki zajmujący się opracowywaniem systemów zdolnych do wykonywania zadań wymagających ludzkiej inteligencji. Zadania te obejmują szerokie spektrum zagadnień, od przetwarzania języka naturalnego przez rozwiązywanie problemów aż po zdolność uczenia się na bazie doświadczenia i adaptacji do możliwych zmian warunków. Kluczowym punktem w rozwoju tej dziedziny była konferencja w Dartmouth [2], na której po raz pierwszy próbowano określić jej ramy i miejsce w naukach technicznych.

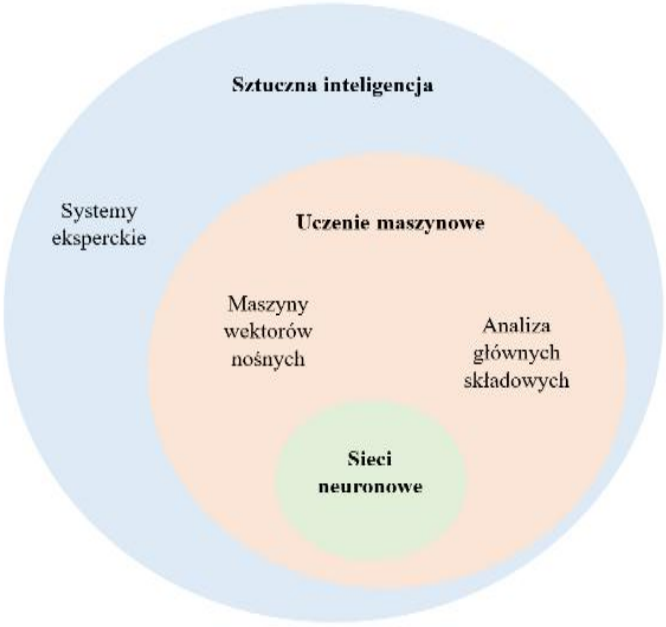
Czasy po tej konferencji zdominowane były przez podejście, które dziś często określa się mianem „starej, dobrej sztucznej inteligencji" (ang. *Good Old-Fashioned Artificial Intelligence*). Opierało się ono na przeświadczeniu, że inteligencja może być przedstawiona przy użyciu reguł logicznych i prostych algorytmów. W efekcie, tworzono rozbudowane systemy oparte na złożonych instrukcjach warunkowych i wiedzy eksperckiej, które znane są dziś głównie pod nazwą systemów eksperckich. Choć w pewnych sytuacjach systemy te charakteryzowały się osiąganymi rezultatami lepszymi od tworzących je ludzi, były one wyspecjalizowane wyłącznie w swoich dziedzinach i nie radziły sobie z generalizacją problemów [42], co wykluczało ich szersze zastosowanie.

W odpowiedzi na te ograniczenia równolegle trwały badania na systemami, które nie muszą bazować na opracowanych regułach, lecz są w stanie same dopasować się do danych bez narzuconych instrukcji; podejście to nazwano uczeniem maszynowym. Pod względem teoretycznym było ono bardzo zbliżone do sposobu, w jaki uczą się ludzie – oparte na przetwarzanych przykładach, a nie na ustalonych z góry regułach. Jednakże takie rozwiązania wymagają wykorzystania dużych zbiorów danych i dostępu do odpowiedniego do ich przetworzenia zasobów sprzętowych. Ograniczenia te sprawiły, że badania nad uczeniem maszynowym pozostawały na dłuższy czas w pewnym sensie zmarginalizowane. Nie przeszkodziło to jednak w opracowaniu takich algorytmów jak drzewa decyzyjne [43], maszyny wektorów nośnych [44] czy algorytm k-najbliższych sąsiadów [45].

Po pewnym czasie nastąpił przełom w kwestii sztucznych sieci neuronowych, które początkowo stanowiły jedną z gałęzi rozwoju systemów eksperckich. Zostały one dopasowane do podejścia uczenia maszynowego, a dzięki opracowaniu algorytmu propagacji wstecznej [46], zaczęto dostrzegać ich potencjał w wielu zastosowaniach praktycznych. Ich prawdziwy rozkwit nastąpił jednak dopiero w ostatnich latach, gdy rozwiązano wiele kwestii związanych

z procesem uczenia. Sieci neuronowe stanowią dzisiaj główny dział uczenia maszynowego, na którym skupiona jest większość uwagi badawczej.

Z takiej perspektywy wyraźnie widać, że sztuczna inteligencja to znacznie szersze zagadnienie niż samo uczenie maszynowe, choć to właśnie ono, a w szczególności sieci neuronowe, stało się jego najskuteczniejszym narzędziem. Na rysunku 3.1 zobrazowano uproszczoną relację między tymi dziedzinami. Warto zauważyć, że do uczenia maszynowego wlicza się także takie algorytmy jak algorytm analizy głównych składowych [47], chociaż powstał on na wiele lat przed pojawieniem się idei uczenia maszynowego. Został on, jak i wiele innych algorytmów, przystosowany do podejścia reprezentowanego przez uczenie maszynowe.



Rysunek 3.1. Sieci neuronowe a sztuczna inteligencja.

3.2. Zasada działania sieci neuronowych

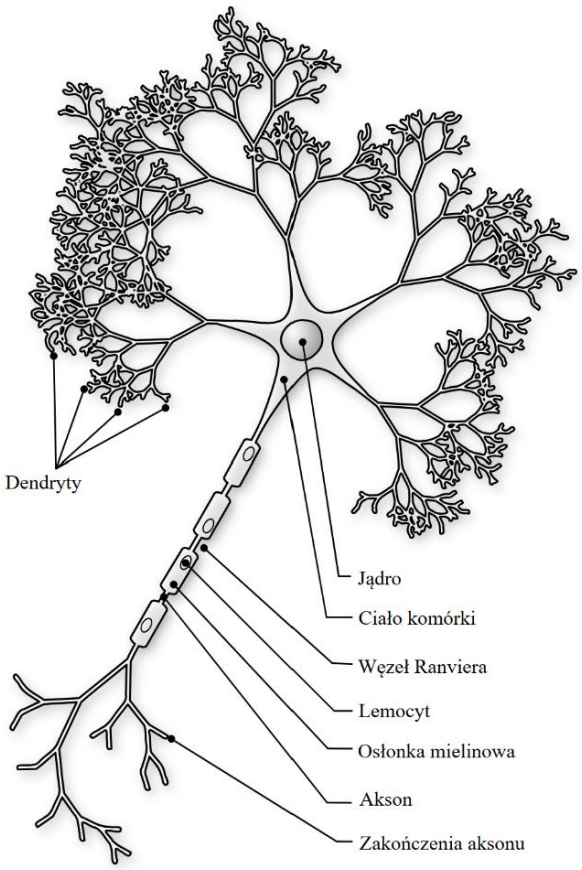
Umiejętne wykorzystanie możliwości sieci neuronowych wymaga znajomości ich struktury, jak i metod zapewniających ich stabilne i przewidywalne działanie. W tym celu przybliżono pokrótce historię ich rozwoju od pojawienia się w połowie zeszłego wieku w formie pojedynczego neuronu do rozwoju w znacznie bardziej złożone struktury. Jednakże sama ich złożoność nie jest warunkiem wystarczającym do ich optymalnej pracy. Najbardziej

istotnym zagadnieniem jest istnienie mechanizmu umożliwiającego im swego rodzaju autokorektę, która wykorzystana w iteracyjnym procesie treningu minimalizuje błędy modelu. Realizowany on jest poprzez algorytm propagacji wstecznej, który od lat 80. stanowi fundament współczesnych sieci neuronowych.

3.2.1. Struktura sieci neuronowych

Jako początek historii sieci neuronowych uznaje się opublikowanie w 1943 roku słynnego artykułu [1] autorstwa neurofizjologa Warrena McCullocha i matematyka Waltera Pittsa. Zaproponowali oni model pierwszej sztucznej sieci neuronowej realizującą obliczenia z wykorzystaniem rachunku zdań. Jej nazwę i sposób działania oparli na podobieństwie do biologicznej komórki nerwowej, neuronu. Na rysunku 3.2 przedstawiony jest schemat takiej komórki wraz z opisem jej elementów. Do najważniejszych z nich należą dendryty, ciało komórki i akson wraz z jego zakończeniami, synapsami. Neurony komunikują się między sobą, generując krótkie impulsy elektryczne, które przechodzą przez akson i wyzwalają w synapsach sygnały chemiczne zwane neuroprzekaźnikami. Sygnały te są odbierane przez dendryty sąsiednich komórek i przekazywane do ciała komórki. Jeżeli neuron otrzyma dostateczną liczbę takich pobudzeń, sam zaczyna generować własne impulsy. Proces ten zachodzi analogicznie we wszystkich innych neuronach, które są ze sobą połączone w złożone, zawierające miliardy elementów, sieci. Na rysunku 3.3 przedstawiony został szkic kory mózgowej człowieka, która jest przykładem takiej wielowarstwowej sieci neuronowej. Na opisanej zasadzie działania opracowano pierwsze sztuczne neurony operujące na wartościach binarnych. Autorzy wspomnianej publikacji udowodnili, że nawet z wykorzystaniem tak uproszczonego modelu można zbudować sieć, która jest w stanie przeprowadzić złożone operacje logiczne.

Trwające w następnych latach badania doprowadziły do zaproponowania przez Franka Rosenblatta architektury perceptronu [48] (rysunek 3.4). Opiera się ona na zmodyfikowanym sztucznym neuronie, zwanym progową jednostką logiczną (ang. *Threshold Logic Unit,* TLU), który nie operuje już na stanach binarnych, lecz na liczbach. Działa ona w taki sposób, że wektorowi n sygnałów wejściowych $x = [x_1, x_2, \ldots, x_n]$ przypisuje określony wektor wag $w = [w_1, w_2, \ldots, w_n]$, wylicza ich sumę ważoną i dodaje człon obciążenia b, zwany także przesunięciem, progiem aktywacji lub potocznie „biasem” (wzór 3.1). Uzyskany wynik z jest następnie przetwarzany z wykorzystaniem wybranej funkcji (wzór 3.2); w perceptronie była to początkowo zazwyczaj funkcja skokowa Heaviside’a (wzór 3.3).



Rysunek 3.2. Schemat budowy biologicznego neuronu (przetłumaczona praca Nicolasa Rougiera, Uznanie autorstwa 3.0, *https://commons.wikimedia.org/wiki/File:Neuron-figure_PL.svg*, dostęp: 1.09.2025 r.).

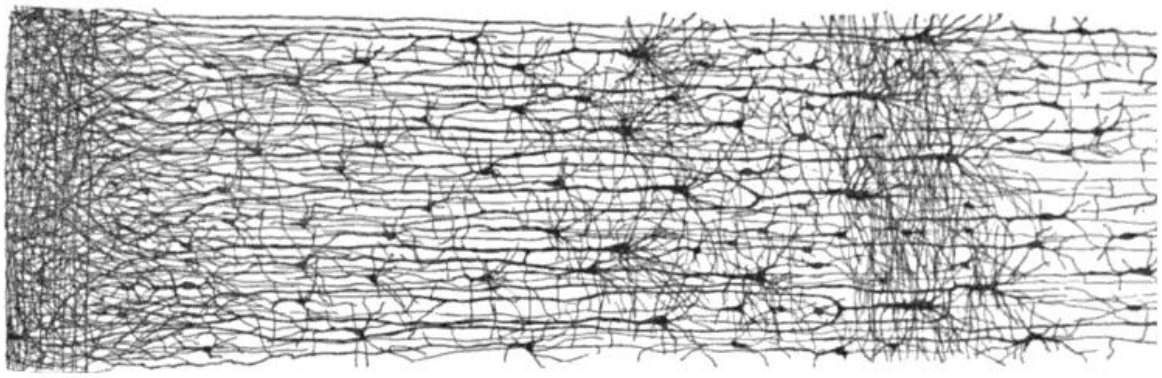
$$z = \sum_{i=0}^{n} w_i x_i + b = w^T x + b \tag{3.1}$$

$$y = f(z) = f(w^T x + b) \tag{3.2}$$

$$Heaviside(z) = \begin{cases} 0, gdy\ z < 0 \\ 1, gdy\ z \geq 0 \end{cases} \tag{3.3}$$

Perceptrony początkowo występowały jako oddzielne jednostki TLU, ale były także organizowane w pojedyncze warstwy jednostek TLU, z których każda była połączona ze wszystkimi wejściami (dlatego też nazwano je warstwami w pełni połączonymi). Początkowo wzbudzały szeroki zachwyt i pokładano w nim nadzieje na dalszy rozwój dziedziny. Jednakże modele te były w dużej mierze ograniczone. W 1969 roku Marvin Minsky i Seymour Papert

wydali monografię „Perceptrons” [49], w której uwypuklili wszystkie wady związane z użyciem perceptronów. Najpoważniejszym zarzutem była zbytnia prostota modelu, który nie był w stanie rozwiązywać bardziej złożonych zdań, a nawet tych bardziej trywialnych jak klasyfikacja alternatywy rozłącznej. Wynikająca z tego ograniczona lista zastosowań modelu, sprawiła, że w dużym stopniu porzucono badania z wykorzystaniem tego typu modeli.

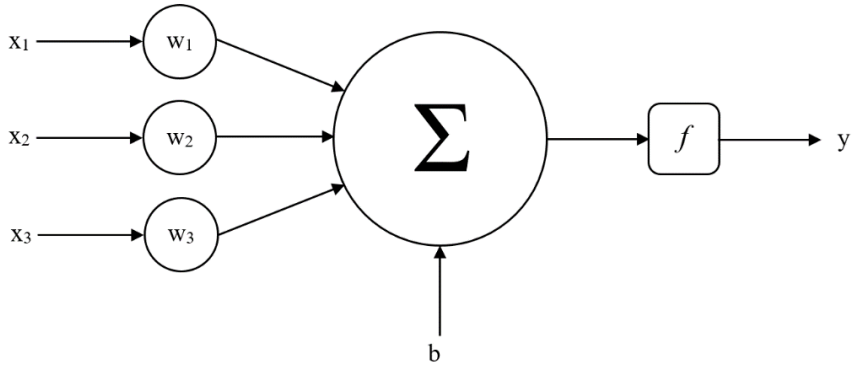


Rysunek 3.3. Uwarstwienie kory mózgowej autorstwa Santiaga Ramóna y Cajala.
(*https://commons.wikimedia.org/wiki/File:Cajal_cortex_drawings.png*, dostęp: 1.09.2025 r.)

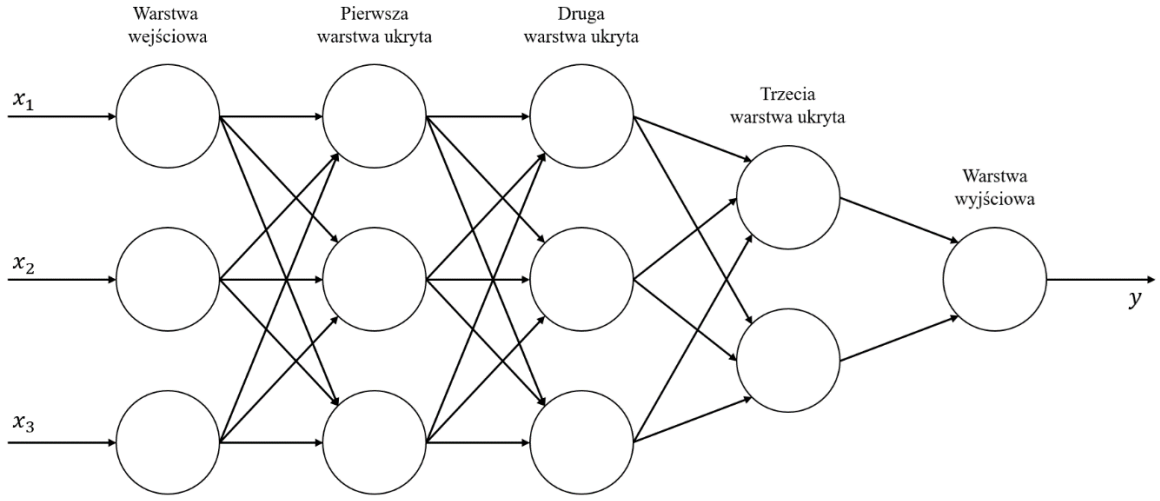
Rozwiązaniem części z tych problemów było zaproponowanie modelu perceptronu wielowarstwowego. Modele te składają się z wielu w pełni połączonych warstw, spośród których rozróżniamy przyjmującą surowy sygnał warstwę wejściową, pewną liczbą warstw ukrytych i warstwę ostatnią, zwaną warstwą wyjściową. Dzięki znacznie większej złożoności były one w stanie sprostać wielu stawianych mu zadaniom, jednakże wiązało się to z problemem poprawnego uczenia takich perceptronów. W przypadku perceptronów jednowarstwowych wagi i obciążenia były inicjowane i aktualizowane z użyciem bardzo prostych algorytmów, które wraz ze wzrostem liczby warstw sprawiały się coraz gorzej. Dopiero pojawienie się znacznie bardziej wyrafinowanych podejść do „trenowania” sieci, poskutkowało pełnym wykorzystaniem potencjału takich sieci (opisane dokładnie w *3.2.2. Trening sieci neuronowych*).

Na rysunku 3.5 przedstawiono schemat przykładowego perceptronu pięciowarstwowego, na wyjściu którego wystawiana jest pojedyncza wartość, a liczba neuronów w warstwach jest podobna. Nie jest to jednak jedyne możliwe rozwiązanie, wyjść perceptronu może być znacznie więcej, natomiast liczba neuronów w warstwach ukrytych może

zarówno rosnąć, maleć, jak i pozostawać bez zmian. Niegdyś, w przypadku sieci neuronowych posiadających więcej niż jedną warstwę używano określenia sieci głębokich, jednak w dzisiejszych czasach powszechne są sieci posiadające setki warstw, wobec czego takie rozgraniczenie bywa nieprecyzyjne.



Rysunek 3.4. Matematyczny model perceptronu złożonego z jednej jednostki TLU.



Rysunek 3.5. Model perceptronu złożonego z wielu warstw gęstych.

3.2.2. Trening sieci neuronowych

Treningiem, czyli procesem uczenia sieci neuronowej, nazywamy realizowaną iteracyjnie poprawę jej działania w celu dopasowania jej do wybranego zadania. W przypadku perceptronów proces ten początkowo przebiegał w oparciu o różne wariacje reguły Hebba [50], według której połączenie między komórkami staje się tym silniejsze, im częściej dochodzi do ich wzajemnego pobudzenia. Stosując takie podejście, można było co prawda uzyskać dobre wyniki w niektórych zadaniach, jednakże w przypadku sieci głębszych nie spełniało ono oczekiwań. Rozpoczęte poszukiwania rozwiązania tego problemu doprowadziły niektórych badaczy do prób aktualizowania parametrów sieci z użyciem algorytmu gradientu prostego.

Algorytm ten wykorzystywany w iteracyjnym treningu polega na optymalizacji ustalonej funkcji kosztu L (w uczeniu maszynowym częściej określana jest jako funkcja straty). W pierwszym kroku inicjalizowane są parametry sieci ρ, a następnie dane wejściowe są przepuszczone przez kolejne warstwy sieci w ramach propagacji w przód. Następnie obliczona jest pochodna funkcji kosztu, czyli gradient, $\nabla_\rho L(\rho)$ dla aktualnych wartości parametrów. Z racji tego, że celem jest minimalizacja funkcji straty, parametry funkcji są aktualizowane poprzez odjęcie od nich wartości tego gradientu przemnożonej przez współczynnik uczenia η, który decyduje o tym, jak duży krok powinien zostać zrobiony w stronę przeciwną do gradientu (wzór 3.4). Krok ten jest powtarzany, aż do spełnienia określonego warunku zatrzymania.

$$\rho = \rho - \eta \nabla_\rho L(\rho) \tag{3.4}$$

Niestety, przy ówczesnym zaawansowaniu komputerów przeprowadzanie takich obliczeń dla złożonych modeli było bardzo czasochłonne, co ograniczało możliwość szerszego stosowania tego podejścia. Kluczowa do rozwiązania tego problemu okazała się praca magisterska Seppo Linnainmaa [51], w której przedstawił on sposób wykorzystania reguły łańcuchowej do automatycznego i efektywnego obliczeniowo wyznaczania wszystkich gradientów modelu w ramach jednego przebiegu w przód i w tył. Algorytm ten, nazywany odwrotnym różniczkowaniem automatycznym, w połączeniu z algorytmem gradientu prostego tworzą algorytm propagacji wstecznej, który stanowi obecnie najczęściej stosowane podejście do trenowania sieci. Stało się to między innymi za sprawą pracy z 1985 roku [52], która przeanalizowała wpływ tego algorytmu na poznawanie skutecznych reprezentacji wewnętrznych przez sieci neuronowe.

3.3. Środowisko testowe

Sieci neuronowe można implementować programowo z wykorzystaniem licznych języków programowania i dostępnych dla nich gotowych modułów zwanych bibliotekami. Dynamiczny rozwój narzędzi do uczenia maszynowego sprawił, że dominującym językiem w tej dziedzinie stał się Python, który został wykorzystany przy tworzeniu niniejszej pracy. Zaletami tego języka są między innymi jego prostota, duża liczba specjalistycznych bibliotek, liczne grupy użytkowników i twórców, a także dostępność do zróżnicowanych *frameworków*[1] skupionych na uczeniu maszynowym. Z najpopularniejszych, a zarazem najbardziej rozbudowanych spośród nich, wymieć można PyTorcha [53], JAXa [54] i TensorFlow [55].

Chociaż wszystkie opisane frameworki mogą być traktowane jako uniwersalne narzędzia, występują pomiędzy nimi istotne różnice, które wpływają na typowe obszary ich zastosowań. Przyjęło się, że TensorFlow wykorzystywany jest przede wszystkim w aplikacjach przemysłowych, natomiast w środowiskach naukowych i badawczych największą popularnością cieszy się PyTorch (w pewnych sytuacjach stosowany jest także JAX). W głównej mierze wynika to z jego dużej elastyczności, która pozwala dynamicznie tworzyć i testować wiele modeli. Z tego względu cały kod wykorzystany w badaniach został oparty na PyTorchu. Jest to informacja o tyle istotna, że używana nomenklatura oraz sposoby implementacji rozwiązań będą zgodne wyłącznie ze specyfiką tego środowiska i mogą się różnić w przypadku wykorzystania innego frameworku.

W przypadku PyTorcha, a także wielu innych podobnych środowisk, dane reprezentowane są przy pomocy tensorów, czyli tablic różnych wymiarów, których przedstawicielami są między innymi wektory (jeden wymiar) i macierze (dwa wymiary). Przy przetwarzaniu obrazów, rozmiar tensora najczęściej opisany jest w czterech wymiarach jako [*B, C, H, W*], gdzie *B* odpowiada rozmiarowi grupy danych (liczbie osobnych próbek), *C* liczbie kanałów obrazu (dla obrazów w skali szarości wartość ta wynosi 1, a dla obrazów kolorowych 3; liczba ta może jednak ulec znacznemu zwiększeniu w trakcie przetwarzania przez sieć – zamiast o kanałach mówimy wtedy o „mapach cech”), natomiast wymiary przestrzenne danych odpowiadają wymiarom *H* i *W*.

[1] W kontekście uczenia sieci neuronowych frameworkiem nazywamy specjalistyczne środowisko programistyczne zapewniające dostęp do gotowych komponentów takich jak funkcje, klasy, moduły i inne narzędzia, które ułatwiają projektowanie, trenowanie, ocenę jakości i wdrażanie modeli sieci neuronowych bez potrzeby tworzenia wszystkiego od podstaw.

3.4. Wybrane warstwy sieci

Sieci neuronowe są złożonymi strukturami, które składają się z wielu elementów zwanych warstwami. Warstwy można podzielić na typy pełniące różne funkcje i przetwarzające dane w unikalny sposób. By móc je poprawnie wdrożyć do wykonywania bardziej złożonych zadań, należy najpierw zrozumieć ich działanie i ich wpływ na proces treningu testowanych sieci. W tym podrozdziale przybliżono działanie tych warstw, które są istotne w kontekście wykorzystywanych w badaniach sieci. Co istotne, warstwy te mogą działać samodzielnie, ale mogą też być łączone w większe jednostki, jak opisano na przykładzie warstw resztkowych (znanych także jako rezydualne).

3.4.1. Warstwa w pełni połączona

Warstwa ta, zwana też warstwą gęstą, określona jest w PyTorch jako torch.nn.Linear z uwagi na liniowe przekształcenie danych wejściowych. Jej parametrami są jedynie rozmiary wejścia i wyjścia oraz opcja wykorzystania wektora obciążeń. Warstwa sama w sobie nie zawiera funkcji aktywacji i trzeba ją dodać w osobnym kroku. Jej struktura opowiada pojedynczej warstwie perceptronu wielowarstwowego z rysunku 3.5. W przetwarzaniu obrazów jej wykorzystanie jest mocno ograniczone przez dużą zależność od wymiarów przestrzennych danych.

3.4.2. Warstwa konwolucyjna

Występujący w nazwie warstwy termin „konwolucja” powinien być tłumaczony na język polski jako splot. Niemniej, ze względu na powszechność użycia, w praktyce inżynierskiej i literaturze przedmiotowej utrwaliła się oryginalna, anglojęzyczna forma. Co ciekawe, określanie realizowanej przez tę warstwę funkcji tym terminem jest o tyle nieprecyzyjne, że warstwa ta w rzeczywistości nie realizuje splotu, lecz zbliżoną do niego korelację krzyżową. Różnica między tymi operacjami sprowadza się do tego, że przed przetworzeniem danych przez splot dochodzi dodatkowo do odwrócenia jądra, co zapewnia tej operacji przemienność. Z punktu widzenia klasycznego przetwarzania sygnałów może to w pewnych sytuacjach mieć duże znaczenie, jednak w przypadku sieci neuronowej różnica ta jest zaniedbywalna, bo wagi jądra są i tak modyfikowane podczas treningu sieci.

Operacja realizowana przez tę warstwę polega na przetworzeniu odpowiednich wycinków danych przez filtr, zwany także jądrem konwolucyjnym, o rozmiarze $k_s \times k_s$ (zazwyczaj maski takich filtrów mają kształt kwadratu) przesuwającego się po obrazie o zadaną wartość kroku. W przypadku kroku większego niż jeden, wiąże się to ze zmniejszeniem rozmiaru obrazu wyjściowego. Najczęściej stosowane są niewielkie filtry o nieparzystej długości: 1×1, 3×3, 5×5 i 7×7, które są najbardziej wydajne obliczeniowo.

W pierwszym kroku umieszcza się jądro w lewym górnym rogu przetwarzanego obrazu. Następnie wymnaża się odpowiadające sobie elementy, a ich sumę wpisuje się jako wartość obrazu wynikowego. Na rysunku 3.6 przedstawiono przetworzenie przykładowego obrazu 5×5 przez jądro 3×3. Na początku filtr dopasowywany jest do lewego górnego rogu (zielona ramka z zaznaczonym na zielono środkiem) i wyliczana jest wartość korelacji krzyżowej:

$$71 * (-4) + 21 * 2 + 34 * 0 + (-9) * (-8) + 39 * 0 + 11 * 4 + (-4) * 5 + 32 * 4 + 0 * 3 = -18$$

Obliczona wartość stanowi pierwszą wartość obrazu wynikowego. Następnie filtr przesuwany jest w rzędzie o zadany krok (w tym wypadku jeden) i obliczana jest kolejna wartość (ramka czerwona):

$$21 * (-4) + 34 * 2 + 23 * 0 + 39 * (-8) + 11 * 0 + 41 * 4 + 32 * 5 + 0 * 4 + 15 * 3 = 41$$

Obraz przetwarzany

71	21	34	23	23
-9	39	11	41	32
-4	32	0	15	34
85	52	93	58	46
8	-5	30	49	21

Jądro splotowe

-4	2	0
-8	0	4
5	4	3

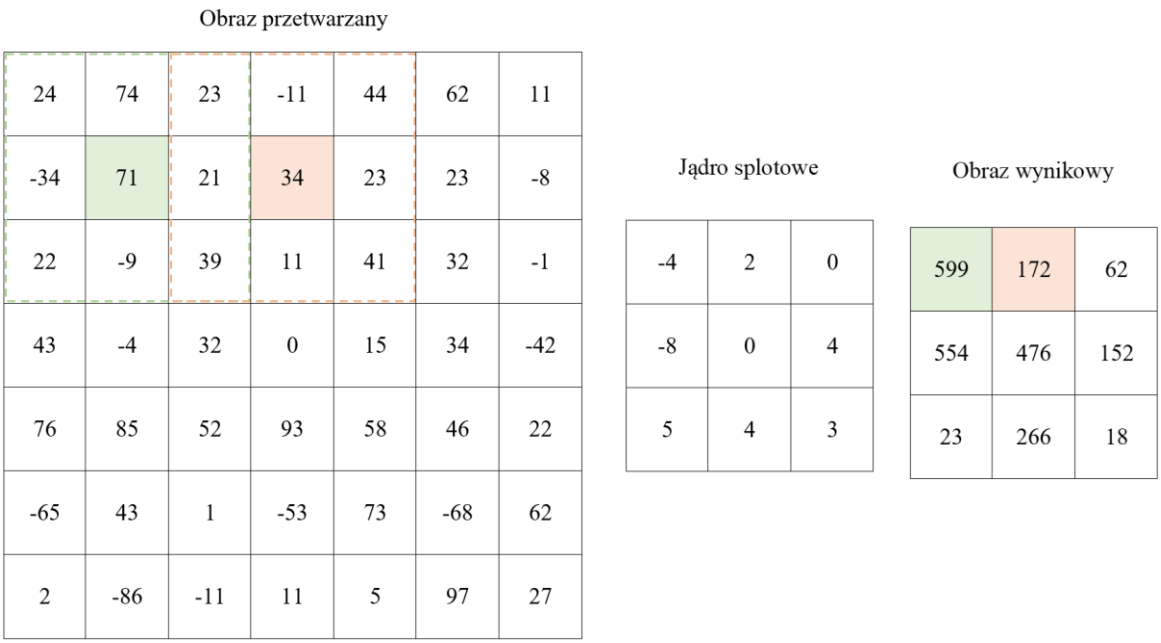
Obraz wynikowy

-18	41	112
1058	476	1009
-118	-70	-121

Rysunek 3.6. Przykład działania splotu (rozmiar jądra: 3×3, krok: 1, bez obciążenia).

Operacja ta jest powtarzana, aż do osiągnięcia prawej krawędzi obrazu. W takiej sytuacji filtr jest przenoszony z powrotem do lewej krawędzi, jednakże przesunięty jest o ustaloną liczbę rzędów w dół – w wykorzystanym przykładzie ten krok również wynosi 1. Zazwyczaj wartość kroku wynosi tyle samo w obydwu kierunkach, jednak niekiedy zdarza się stosować ich różne wartości. Na rysunku 3.7 pokazane jest działanie splotu o kroku 2. Jak można zauważyć, większa długość kroku może posłużyć do redukcji wymiarowości danych i w tym celu jest też często stosowana.

Najważniejszymi parametrami warstw konwolucyjnych jest zatem długość kroku, wielkość jądra, a także liczba kanałów (map cech) wejściowych C_{in} i wejściowych C_{out}. Dla jąder kwadratowych liczba parametrów w danej warstwie wynosi $C_{in} * C_{out} * k_s^2$ i nie jest w żaden sposób związana z rozmiarem przestrzennym obrazu, co jest dużym usprawnieniem w porównaniu do warstw gęstych. Zastosowanie filtrów przestrzennych pozwala dodatkowo uwzględnić informację o pozycji i otoczeniu przetwarzanego piksela, co nie ma miejsca w przypadku warstw w pełni połączonych przetwarzających piksele niezależnie od siebie. Sprawia to, że do przetwarzania obrazów wykorzystuje się, w głównej mierze, warstwy konwolucyjne.



Rysunek 3.7. Redukcja wymiarowości z wykorzystaniem splotu (rozmiar jądra: 3×3, krok: 2, bez obciążenia).

3.4.3. Warstwa uzupełnień

W większości przypadków redukcja rozmiaru obrazu związana z działaniem warstwy konwolucyjnej jest efektem, którego chcemy uniknąć. Aby zapobiec takim przycięciom, stosowane są warstwy uzupełnień (ang. *padding*), które dodają ramkę danych dookoła oryginalnego obrazu. Najprostsza metoda polega na wypełnieniu tej ramki samymi zerami. Jest ona także najszybsza, ale może skutkować pojawieniem się artefaktów na krawędziach, ponieważ wartości zerowe mogą zniekształcić znajdujący się tam sygnał. W związku z tym stosuje się również inne podejścia do uzupełnień, przy czym większość z nich, poza replikacją i odbiciem, jest dosyć rzadko spotykana. Na rysunku 3.8 przedstawiono dopełnienie poprzez lustrzane odbicie wartości względem elementów brzegowych. Uzupełnienie poprzez replikację polega natomiast na powieleniu wartości brzegowych.

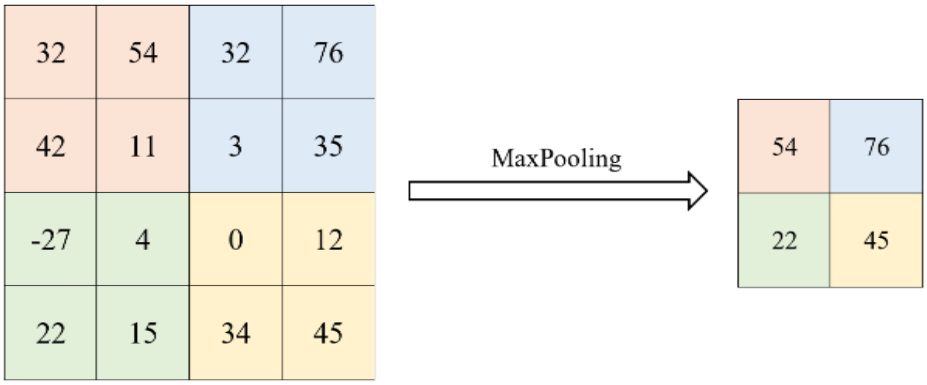
7	-2	7	14	7
0	21	0	5	0
7	-2	7	14	7
8	38	8	-11	8
7	-2	7	14	7

Rysunek 3.8. Przykład dopełnienia przez lustrzane odbicie. Na zielono zaznaczony oryginalny obraz.

3.4.4. Warstwa łącząca

W sieciach neuronowych redukcji wymiaru danych można dokonać z wykorzystaniem warstw konwolucyjnych o kroku większym od jeden, a także z wykorzystaniem specjalnych warstw łączących (także: próbkujących, ang. *pooling*). Warstwy te nie zawierają żadnych parametrów podlegających procesowi uczenia, lecz działają deterministycznie, wykorzystując określoną statystykę. Najpopularniejszymi przedstawicielami tych warstw są warstwa maksymalizująca (ang. *MaxPooling*) i warstwa uśredniająca (ang. *AveragePooling*).

Ich działanie opiera się na podzieleniu przetwarzanych obrazów na fragmenty $k_s \times k_s$ (ustalana przez użytkownika wielkość filtra), a następnie wyznaczeniu z tego obszaru wartości największej (*MaxPooling*) lub obliczeniu wartości średniej (*AveragePooling*). Działanie to pozwala zachować najważniejsze informacje z danego fragmentu, przy jednoczesnym zmniejszeniu jego wymiarowości, co łączy się ze zmniejszeniem liczby parametrów modelu i wymaganych obliczeń. W warstwach tych można także modyfikować wartość kroku przesunięcia filtra, jednak zazwyczaj nie jest to stosowane podejście. Na rysunku 3.9 przedstawiono działanie maksymalizującej warstwy łączącej, która dwukrotnie zmniejsza rozmiar danych w każdym z wymiarów.



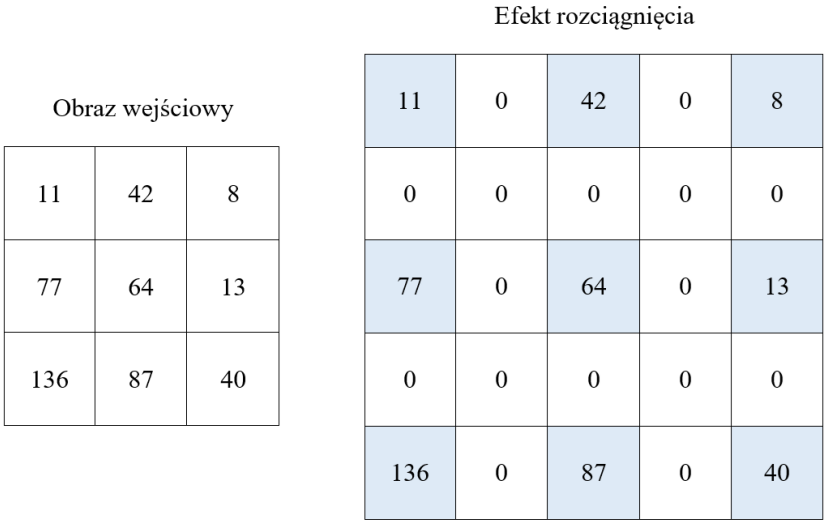
Rysunek 3.9. Przykład działania maksymalizującej warstwy łączącej (rozmiar jądra: 2×2, krok: 1).

3.4.5. Warstwa dekonwolucyjna

W wielu wypadkach oprócz zmniejszania rozmiaru obrazów potrzebne są także techniki jego zwiększania. Jedna z nich opiera się na wykorzystaniu warstwy dekonwolucyjnej, znanej także jako warstwa rozplotowa lub transponowana warstwa konwolucyjna. Analogicznie jak w przypadku warstw konwolucyjnych ich działanie opiera się na korelacji krzyżowej, a nie na splocie, jednak w powszechnym użyciu utrwaliło się właśnie takie określenie.

Działanie takiej warstwy opiera się na rozciągnięciu oryginalnego obrazu (wymaga to ustawienia wartości kroku na wartość > 1) poprzez dodanie rzędów i kolumn zawierających same zera (rysunek 3.10), a następnie przetworzenie takiego obrazu przez jądro konwolucyjne tak jak przedstawiono na rysunku 3.6. Podejście to jednak nie jest zbyt precyzyjne, przez co najczęściej skutkuje pojawieniem się „artefaktów szachownicy”, regularnych zakłóceń

przyjmujących postać siatki zawierającej jaśniejsze i ciemniejsze piksele [56]. By załagodzić ten problem, stosowane są połączenia pomijające [57], które przenoszą bardziej szczegółowe informacje z wcześniejszych warstw, a także zamienienie warstwy dekonwolucyjnej na warstwę zwiększania rozdzielczości wraz ze znajdującą się zaraz po niej warstwą konwolucyjną.



Rysunek 3.10. Zwiększanie rozmiarowości obrazu z wypełnieniem zerami.

3.4.6. Warstwa zwiększania rozdzielczości

Warstwa zwiększania rozdzielczości (ang. *Upsample*) podobnie do warstwy łączącej nie zawiera żadnych podlegających treningowi parametrów, ale opiera się na algorytmach deterministycznych. W odróżnieniu od warstw dekonwolucyjnych, kolumny i wiersze wstawiane przez tę warstwę nie są uzupełniane zerami (jak na rysunku 3.10), lecz wartościami wyliczanymi za pomocą metod interpolacji, takich jak interpolacja najbliższego sąsiada, liniowa, dwuliniowa, dwusześcienna lub trójliniowa.

3.4.7. Warstwy aktywacji

W architekturach sieci neuronowych tworzonych w PyTorchu funkcje aktywacji nie są wbudowane bezpośrednio w inne warstwy, lecz występują jako osobne komponenty, traktowane jak niezależne warstwy. Choć wynika to głównie z przyjętej konwencji

implementacyjnej, w praktyce funkcje aktywacji stanowią odrębne elementy modelu, dlatego zostały opisane w tym rozdziale.

Rola funkcji aktywacji w sieciach neuronowych jest kluczowa, ponieważ to one wprowadzają nieliniowości do modeli, co znacząco zwiększa ich zdolności do przetwarzania złożonych danych. Stosowane początkowo funkcje skokowe, takie jak funkcja Heaviside'a czy funkcja znakowa, zostały w głębszych sieciach zastąpione funkcją sigmoidalną i tangensem hiperbolicznym, które charakteryzowały się płynniejszym przejściem pomiędzy skrajnymi wartościami. W konsekwencji gradienty funkcji strat były skuteczniej przekazywane do głębszych warstw sieci, a efektywność treningu modeli wzrosła. Problemem pozostawała jednak zauważalna niestabilność gradientów – niektóre spośród warstw uczyły się zdecydowanie wolniej od pozostałych. Dopiero w 2010 roku Xavier Glorot i Yoshua Bengio [58] wykazali, że przyczyną tego stanu jest w dużej mierze działanie poszczególnych funkcji aktywacji. Przy bardzo małych lub dużych wartościach wejściowych wspomniane funkcje się nasycają, czyli osiągają wartości skrajne, wobec czego ich gradienty są bliskie zera. Sprawia to, że propagacja wstecz nie jest w stanie zaktualizować parametrów sieci. Zjawisko to nazwano problemem znikającego gradientu.

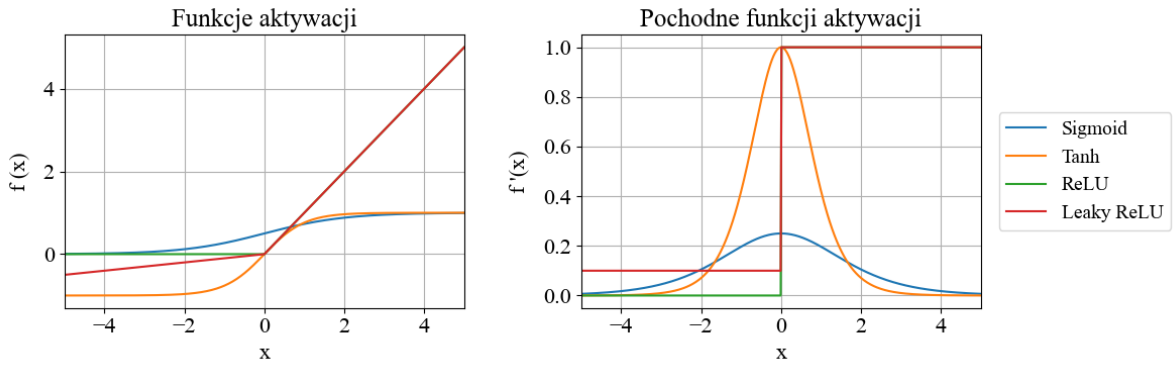
By zaradzić temu problemowi zaczęto prowadzić badania nad funkcjami aktywacji, które nie ulegają nasyceniu. Przełomowe okazało się wykorzystanie funkcji ReLU (prostowana jednostka liniowa, ang. *Rectified Linear Unit*), która nie ma ograniczenia dla wartości dodatnich, a ponadto jest bardzo prosta obliczeniowo (wzór 3.5). To między innymi zastosowaniu tej funkcji przypisuje się sukces sieci AlexNet [5] i do dzisiaj stosuje się ją jako standardową funkcję aktywacji w projektowaniu sieci. Nie jest to jednak aktywacja pozbawiona wad. Przede wszystkim z powodu braku ograniczenia ReLU jest w stanie rosnąć liniowo w nieskończoność, co powoduje wykładniczy wzrost gradientów przy realizacji propagacji wstecznej. Wiąże się to z potrzebą wprowadzenia dodatkowych mechanizmów kontroli, które chronią sieć przed skutkami takiego nieograniczonego wzrostu, zwanego eksplozją gradientu.

Dodatkowo, pochodna ReLU wynosi 0 w przypadku każdej, nawet nieznacznej, wartości ujemnej. W efekcie neuron otrzymujący na wejściu ciągle wartości ujemne lub bliskie zeru nie jest w stanie zaktualizować swoich parametrów, bo jego gradient wynosi zawsze 0. Zjawisko to określano jako śmierć ReLU (ang. *dying ReLUs*). By zaradzić tej niedogodności opracowano funkcje, które przetwarzają wartości ujemne. Ich czołowym przykładem jest „przeciekająca" funkcja ReLU [59] (ang. *LeakyReLU*), która dopuszcza wartości ujemne przemnożone przez niewielki współczynnik kierunkowy α (wzór 3.6) mieszczący się

zazwyczaj w zakresie [$^1/_{100}$; $^1/_{10}$]. Oprócz LeakyReLU występuje wiele innych parametrycznych wersji ReLU [60], oparte na eksponencie funkcje ELU [61] i jej parametryczna odmiana SELU [62], a także bardziej skomplikowane funkcje tak jak GELU [63], Swish [64] oraz Mish [65]. Jednakże mimo ich wielu zalet, w większości eksperymentów w dalszym ciągu dominują aktywacje ReLU wraz z „przeciekającą" wersją. Na rysunku 3.11 przedstawiono wykresy czterech omówionych funkcji oraz ich pochodnych.

$$ReLU(x) = \max(0, x) \tag{3.5}$$

$$LeakyReLU(x) = \begin{cases} x, & \text{gdy } x \geq 0 \\ \alpha x, & \text{gdy } x < 0 \end{cases} \tag{3.6}$$



Rysunek 3.11. Przebiegi wybranych funkcji aktywacji i ich funkcji pochodnych.

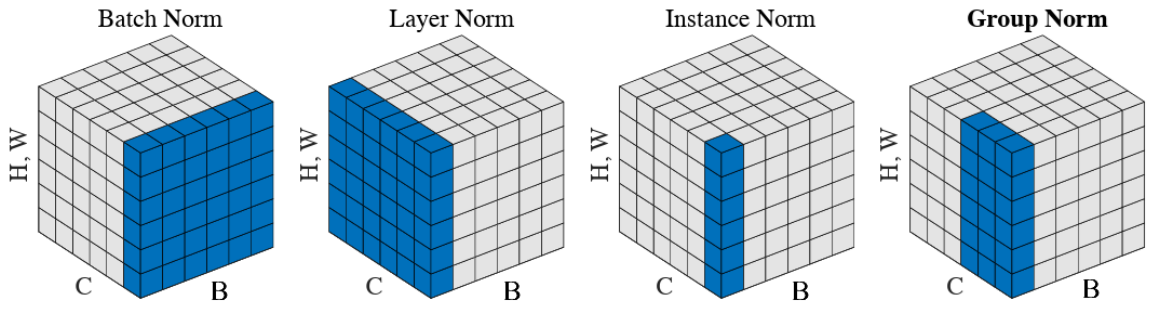
3.4.8. Warstwa normalizacji wsadowej

Uczenie sieci neuronowych opiera się iteracyjnym przetwarzaniu danych treningowych przez sieć i aktualizowaniu jej wag w oparciu o algorytm propagacji wstecznej. Jednokrotne przejście po całym zbiorze danych jest w tym wypadku nazywane epoką. Każda z epok jest zazwyczaj podzielona na wiele iteracji, w czasie których sieć przetwarza losowo wybrane grupy (także: paczki, ang. *batches*) próbek. Jest to w głównej mierze spowodowane możliwościami sprzętowymi – duże zbiory danych w większości przypadków nie są w stanie w całości zmieścić się w pamięci sprzętowej. Sytuacja ta wiąże się jednak z problemem zmiany rozkładu danych wejściowych, który może destabilizować trening.

By przeciwdziałać temu zjawisku, powszechnie używana jest technika normalizacji wsadowej (ang. *batch normalization*) [66], która normalizuje wartości w każdej warstwie względem średniej i wariancji. Co ważne, warstwa normalizacji wsadowej działa nieco inaczej w trakcie treningu i podczas późniejszej pracy. W trakcie treningu przeprowadza ona normalizację z wykorzystaniem statystyk wyznaczonych w obrębie bieżącej grupy danych, aktualizując swoje parametry skalowania i przesunięcia. Natomiast poza treningiem użyte zostają uśrednione wartości średniej oraz wariancji dla całego zbioru danych, a nie tylko poszczególnych paczek. Zastosowanie tego podejścia ułatwia propagację błędów, ogranicza problem znikających gradientów i zmniejsza liczbę epok potrzebną do wyszkolenia sieci (wydłuża przy tym czas trwania każdej epoki, ale efekt końcowy powinien i tak zostać osiągnięty w krótszym czasie).

Warstwy te jednak mają pewną wadę – w przypadkach bardzo małego rozmiaru grup oszacowanie średniej i wariancji staje się niestabilne, co może prowadzić do gorszych wyników. Wobec tego zaproponowane zostały jeszcze inne podejścia do normalizacji [67], które zostały przedstawione na rysunku 3.12. Przedstawione na nim dane tworzące tensor o wymiarach [*B, C, H, W*] zwizualizowano w trójwymiarowej przestrzeni. Powstały wymiar H,W odpowiada zrzutowanym wymiarom przestrzennym obrazu, a kanały C i B odpowiadają odpowiednio liczbie map cech i wielkości grupy. W przypadku normalizacji grupy (ang. *Batch Norm*) osobno dla każdego kanału C normalizowane są osie H,W i B. Przy normalizacji warstwowej (ang. *Layer Norm*) zachodzi normalizacja osi H,W i C niezależnie dla każdej próbki B. Normalizacja pojedynczego przykładu (ang. *Instance Norm*) wykonywana jest natomiast osobno dla każdego kanału C i próbki B na osi H,W. Ostatni rodzaj normalizacji, normalizacja grupowa (ang. *Group Norm*), jest przypadkiem pośrednim pomiędzy normalizacjami warstwową a pojedynczej próbki – kanały C dzielone są na zadaną ilość grup, z których każda jest normalizowana oddzielnie.

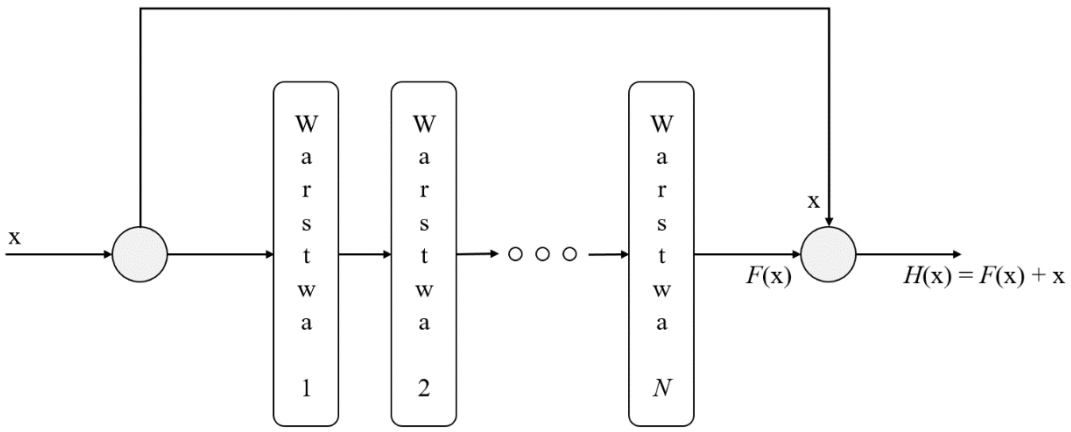
Chociaż inne rodzaje normalizacji rozwiązują problem zbytniej zależności od rozmiaru przetwarzanej paczki danych, nie zdobyły one tak dużej popularności jak algorytm normalizacji wsadowej. Wynika to z faktu, że przy dzisiejszych możliwościach sprzętowych rozmiar paczki nie jest aż tak dużym problemem, a różnice między tymi podejściami są w wielu przypadkach relatywnie niewielkie.



Rysunek 3.12. Metody normalizacji danych [67].

3.4.9. Jednostki rezydualne (resztkowe)

Oprócz prostych warstw realizujących pojedyncze funkcje, w bardziej złożonych sieciach często spotykane są jednostki funkcjonalne, czyli modularne bloki, które odpowiadają strukturą pomniejszym sieciom. Przykładem takiej jednostki może być wprowadzająca połączenia pomijające (ang. *skip connections*) jednostka resztkowa (rysunek 3.13), zastosowana po raz pierwszy we wspomnianej we wstępie sieci ResNet [7].



Rysunek 3.13. Schemat jednostki rezydualnej.

Jednostki zawierające takie połączenia są w stanie w znaczący sposób uprościć mechanizm uczenia sieci, przyspieszyć przejście gradientów przez sieć, a także zapewniają modularność przydatną przy projektowaniu głębszych architektur. Ich działanie sprawdza się szczególnie w przypadku, gdy funkcja $F(x)$ realizowana przez grupę N warstw jest mocno zbliżona do funkcji tożsamościowej. W takiej sytuacji dodanie wejścia x do wyjścia tej grupy

sprawia, że grupa ta jest zmuszona do odwzorowania funkcji $H(x) = F(x) + x$, dla której wyuczenie się $F(x)$ staje się o wiele prostsze.

Tworzenie podobnych (prostszych lub bardziej złożonych) jednostek funkcyjnych jest jedną z podstawowych technik tworzenia architektur sieci, która znalazła zastosowanie w wielu z opisanych w tej pracy eksperymentów. Dla przykładu: opisane tu jednostki rezydualne mają szczególne znaczenie w kontekście sieci wykorzystanych w pracy nad przetwarzaniem obrazów słonecznych (rozdział *6. Redukcja szumu w danych obrazowych Słońca*).

3.5. Architektury sieci

Architekturą sieci neuronowej nazywamy ogólny układ warstw i połączeń w sieci, który określa, jakie warstwy zostały użyte; w jakiej znajdują się one kolejności i konfiguracji; w jaki sposób dane przepływają przez sieć; ile parametrów ma model; a także jakie są jego hiperparametry (rozdział *3.6. Hiperparametryzacja sieci*). Określenie to używane jest zazwyczaj w odniesieniu do teoretycznego modelu matematycznego, w przypadku instancji takiego obiektu najczęściej używa się zamiennie określeń „model” i „sieć”.

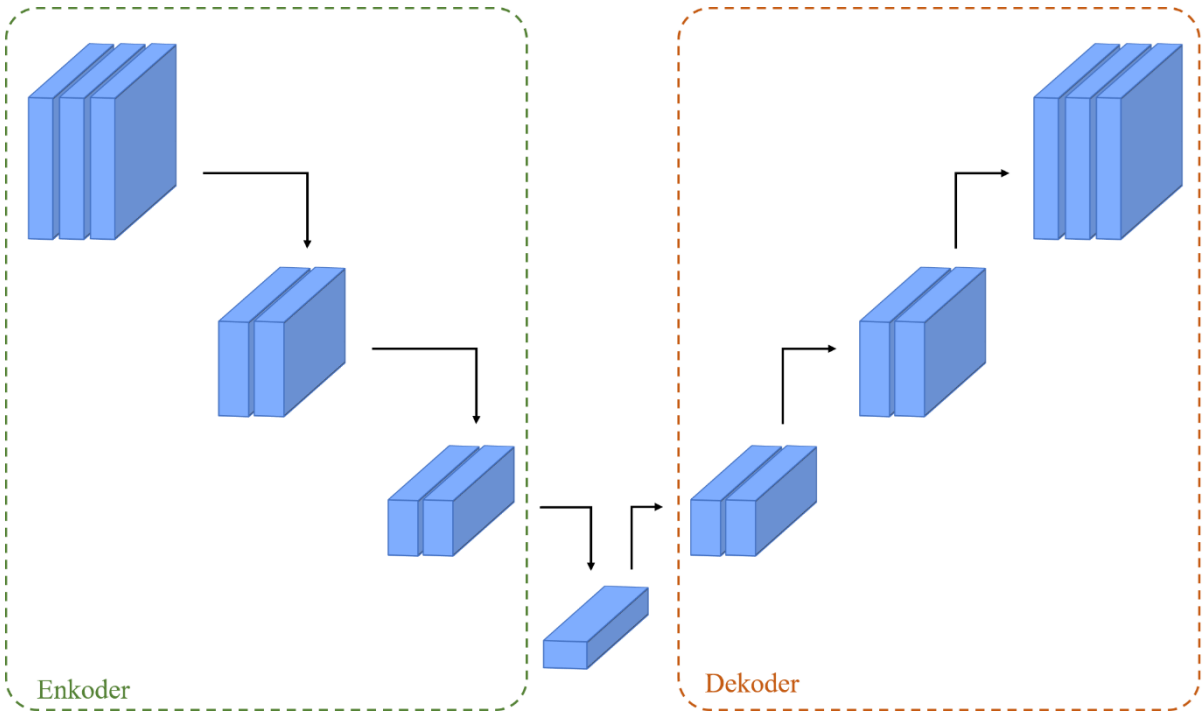
3.5.1. Wybór sieci

W ramach wcześniejszych, niezależnych od niniejszej rozprawy badań, autor zajmował się zagadnieniami związanymi z redukcją wymiarowości danych przy użyciu analizy głównych składowych [68] oraz odwracaniem efektów kompresji stratnej obrazów wynikającej z zastosowania algorytmu JPEG [69]. Obie te metody, przy odpowiednim doborze parametrów, umożliwiają zmniejszenie rozmiaru reprezentacji danych kosztem nieodwracalnej utraty części zawartej w nich informacji. Obserwacja ta posłużyła sformułowaniu hipotezy, że wykorzystanie technik kompresji stratnej, jako filtrów selektywnie przepuszczających jedynie najbardziej istotne informacje wizualne, może usunąć z przetwarzanych obrazów niepożądane elementy takie jak losowy szum. W tym celu autor zwrócił się ku rozwiązaniom oferowanym przez sieci neuronowe, które dzięki zdolnościom modelowania nieliniowych zależności i efektywnemu wydobywaniu kluczowych cech obrazów zostały uznane za szczególnie obiecujące. Najwięcej uwagi poświęcono strukturze dostosowanej do zadań kompresji i dekompresji – architekturze autoenkodera, a także jej rozwinięciu w formie U-Neta.

3.5.2. Autoenkoder

Autoenkoder (AE) to rodzaj sieci neuronowej wykorzystywany do wydajnego kodowania danych wejściowych, zwanych także reprezentacjami ukrytymi (ang. *latent representations*). Kodowania te zazwyczaj mają o wiele mniejszą wymiarowość niż dane oryginalne, dzięki czemu jest on powszechnie wykorzystywany się do kompresji. Stopniem zachodzącej redukcji można sterować poprzez odpowiednie modyfikacje używanej architektury, przede wszystkim przez zmianę rozmiaru docelowego wymiaru kodowania oraz złożoności struktury sieci. W rezultacie autoenkodery znalazły szerokie zastosowanie również w redukcji szumu [70]-[71][73].

Na rysunku 3.14 przedstawiony jest przykładowy schemat autoenkodera. Architektura ta składa się z trzech części: kompresującego dane enkodera, kodowań i dekompresującego je dekodera. Jak można zauważyć na schemacie, poszczególne warstwy podzielone są na oddzielne poziomy – jest to ogólnie przyjęta konwencja w opisywaniu sieci zmieniających rozmiarowość danych. Zgodnie z nią wszystkie warstwy na danym poziomie przetwarzają dane o takich samych rozmiarach przestrzennych H i W, chyba że z opisu schematu jasno wynika coś innego. Często łączy się to oznaczaniem różnych rodzajów warstw z wykorzystaniem odpowiednich barw.

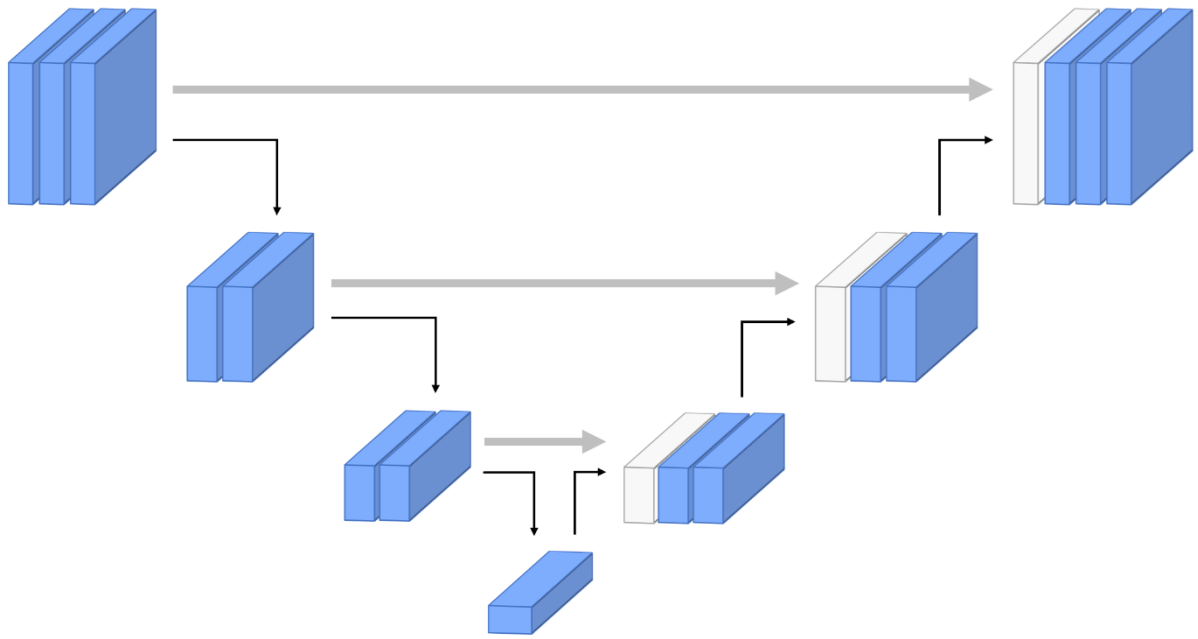


Rysunek 3.14. Architektura autoenkodera.

W większości wypadków enkoder jest strukturą symetryczną do dekodera, jednakże nie jest to z góry narzucona norma. Przykładowo, w sytuacjach, gdy nie potrzebujemy dokładnie odtwarzać całości danych, lecz jedynie ich część, dekoder bywa znacznie mniejszy od enkodera. Ponadto w niektórych implementacjach enkoder działa oddzielnie od dekodera; sieci nie są połączone w jedną strukturę, tylko stanowią dwie współdziałające ze sobą podsieci. Dla rozróżnienia struktura ta nosi nazwę architektury enkoder-dekoder.

3.5.3. U-Net

Architektura U-Net jest modyfikacją symetrycznej, ułożonej w kształt litery U struktury autoenkodera wzbogaconą o połączenia łączące ze sobą poszczególne warstwy, co można zaobserwować na rysunku 3.15 w postaci szarej, pogrubionej strzałki. Połączenia te często przybierają postać połączeń pomijających – na początek wybranej warstwy jest dodawane wyjście z oddalonej warstwy znajdującej się na tym samym poziomie. Oprócz tego popularną techniką jest również łączenie (scalanie) ze sobą tych danych wzdłuż wymiaru map cech C (oznaczone przez dołączenie białego bloku do bloków niebieskich).



Rysunek 3.15. Architektura sieci typu U-Net.

W porównaniu do autoenkodera, U-Net jest w stanie lepiej zachować informacje przestrzenne i tekstury, które są tracone podczas kompresji. Jednak ze względu na połączenia między warstwami wymaga więcej pamięci operacyjnej do przechowywania przetwarzanych danych, co może być problemem w przypadku dużego rozmiaru grup. Chociaż sieci typu U-Net powstały początkowo do celów segmentacji obiektów na obrazach [74], z czasem zaczęto je również stosować w zadaniach redukcji szumu [75][76].

3.6. Hiperparametryzacja sieci

W przypadku sieci neuronowych parametrami nazywamy elementy wewnętrzne modelu, których sieci uczą się w procesie treningu – są nimi wagi i obciążenia. Elementy narzucone, zewnętrzne, które opisują to jak sieć wygląda i jak działa, nazywane są natomiast hiperparametrami (od greckiego „hiper" oznaczającego „nad", „powyżej"). I tak jak ważne jest wyuczenie się przez sieć odpowiednich wartości parametrów wewnętrznych, tak poprawne działanie modelu w głównej mierze zależy od ustawienia poprawnej wartości hiperparametrów. Niestety, nie istnieje żadna metoda ani heurystyka pozwalająca na ich optymalny dobór. Zazwyczaj opiera się on na wyznaczeniu pewnych zalecanych (sprawdzonych w podobnych warunkach) wartości, przetestowaniu działania sieci, a następnie na ich modyfikacji i ponownym porównaniu. W wielu sytuacjach liczba hiperparametrów do strojenia połączona z długim i niejednokrotnie wymagającym procesem uczenia wyklucza możliwość znalezienia modelu optymalnego, czyli gwarantującego najmniejszą możliwą wartość funkcji straty. W przeszłości problem ze znalezieniem minimum globalnego uważano za największy problem uczenia maszynowego, jednakże z biegiem czasu okazało się, że znajdowane minima lokalne sprawowały się na tyle dobrze, że w wielu przypadkach różnice pomiędzy nimi a minimami globalnymi są zaniedbywalnie małe. Jednakże, by sieci były w stanie je osiągnąć, trzeba poprawnie dostroić hiperparametry. W kolejnych podrozdziałach pokrótce omówiono ich działanie i wpływ na skuteczność sieci.

3.6.1. Głębokość sieci

Pierwszym i najbardziej istotnym z hiperparametrów sieci jest jej głębokość, rozumiana jako liczba i konfiguracja warstw. Chociaż udowodniono, że sieć posiadająca zaledwie jedną warstwę ukrytą jest w stanie aproksymować dowolną funkcję ciągłą z dowolną dokładnością

[77], w praktyce tak skrajnie płytkie modele wymagają olbrzymiej liczby neuronów, by skutecznie uchwycić bardziej złożone wzorce. Co więcej, ich zdolność do uogólniania jest często ograniczona, szczególnie w zadaniach o wysokiej złożoności przestrzennej lub semantycznej [78].

Dopiero głębokie sieci neuronowe, składające się z wielu warstw ukrytych, umożliwiają hierarchiczne uczenie reprezentacji. Oznacza to, że każda kolejna warstwa w sieci może przetwarzać dane na innym poziomie; od detekcji prostych cech (na przykład krawędzi) po rozpoznawanie bardziej złożonych struktur (takich jak kształty). Dzięki temu modele te są w stanie uchwycić nieliniowe i wielowymiarowe zależności, co przekłada się na lepszą wydajność w takich zadaniach jak analiza obrazów. Jednak wzrost głębokości sieci wiąże się również z pewnymi wyzwaniami, takimi jak zanikający gradient, trudności w optymalizacji czy zwiększone zapotrzebowanie sprzętowe. Z tego względu wybór odpowiedniej głębokości stanowi wyzwanie, mające na celu znalezienie kompromisu pomiędzy złożonością modelu a jego zdolnością do generalizacji.

3.6.2. Współczynnik uczenia

Przy szkoleniu sieci ważną rolę ogrywa także współczynnik uczenia η. Wybranie zbyt małej wartości może prowadzić do bardzo długiego treningu lub do utknięcia w niemożliwym do opuszczenia minimum lokalnym funkcji straty. Z drugiej strony, duża wartość tego współczynnika może skutkować zbyt dużymi zmianami parametrów, powodującymi destabilizację działania całego modelu. Dodatkowo, minimum globalne może być w takich aktualizacjach przypadkowo pominięte.

By zaradzić temu problemowi opracowano różne strategie do adaptacyjnej zmiany wartości η. Podstawową z nich jest zastosowanie odpowiedniego optymalizatora. Obecnie poza podstawowym algorytmem opisanym wzorem 3.4 najczęściej stosowany jest nieco bardziej rozbudowany algorytm o nazwie Adam (adaptacyjne szacowanie momentów, ang. *Adaptive Moment Estimation*) [79], który zapamiętuje wartości gradientów z poprzednich iteracji i używa ich w celu uzyskania szybszej zbieżności. Uznaniem cieszy się też jego bardziej stabilna, zmodyfikowana wersja, AdamW (Adam z oddzielnym zanikiem wag, ang. *Adam with decoupled Weight Decay*) [80].

Oprócz tych metod można zastosować uczenie z wykorzystaniem harmonogramów, które dbają o zmianę wartości współczynnika uczenia w odpowiednich momentach. Podejście to ma na celu przyspieszenie pierwszej fazy treningu, gdy model musi dokonać największych aktualizacji, a następnie zwolnienie w fazie stabilizacji uczenia, by funkcja strat mogła osiągnąć minimum. Większość podejść opiera się na ustawieniu wysokiej wartości początkowej η, a następnie sukcesywnym zmniejszaniu jej do zera z wykorzystaniem różnych funkcji. Stosowane są także podejścia bazujące na zmianach współczynnika uczenia związany poprzez początkowe „rozgrzanie sieci" [81] lub cykliczne zmiany wartości η [82].

3.6.3. Funkcje straty

Osiągnięcie oczekiwanej jakości wyników w dużej mierze zależy od prawidłowego wyboru funkcji straty. Nie jest to jednak zadanie proste, a stosowane powszechnie normy nie zawsze skutkują zadowalającymi wynikami i to nawet pomimo osiągniętej niskiej wartości błędu [83]. Dwiema z najbardziej popularnych funkcji strat są od lat normy L1 i L2. L1, znana także jako średni błąd bezwzględny (ang. *Mean Absolute Error*, MAE), mierzy średnią wartość bezwzględnych różnic pomiędzy przewidywanymi (y) a rzeczywistymi wartościami pikseli ($\hat{y}$) (wzór 3.7). Dzięki swej odporności na duże odchylenia często prowadzi do zadowalających rezultatów. Jej dużą wadą jest jednak brak pełnej ciągłości (pochodna w punkcie zero gwałtownie przeskakuje z -1 na 1). Natomiast L2, czyli średni błąd kwadratowy (ang. *Mean Squared Error*, MSE), wzmacnia błędy, obliczając ich wartość kwadratową (wzór 3.8), co sprawia, że większe odchylenia mają znacznie większy wpływ na końcowy wynik funkcji. W licznych sytuacjach jest to działanie niepożądane, bo wpływ dużych błędów może zdominować proces uczenia, przez co sieć będzie w mniejszym stopniu skupiać się na poprawnym odtwarzaniu szczegółów.

$$L1(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.7}$$

$$L2(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.8}$$

Aby połączyć zalety obu podejść, Peter J. Huber zaproponował w 1964 roku [84] parametryzowaną funkcję, która dla małych błędów zachowuje się jak MAE, a dla dużych jak MSE (wzór 3.9). Funkcja ta, nazwana od nazwiska twórcy funkcją straty Hubera, wprowadza element progu δ, który zmniejsza wpływ elementów odstających, a zarazem zachowuje lepszą stabilność uczenia poprzez zachowanie lepszej gładkości od MAE.

$$L_\delta(y_i, \hat{y}_i) = \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2, & \text{gdy } |y_i - \hat{y}_i| \leq \delta \\ \delta\left(|y_i - \hat{y}_i| - \frac{1}{2}\delta\right), & \text{gdy } |y_i - \hat{y}_i| > \delta \end{cases} \tag{3.9}$$

W rekonstrukcji obrazów wykorzystuje się obecnie również inne funkcje strat dopasowane do konkretnego zadania, które wykraczają poza tradycyjne podejście oparte na porównywaniu wartości pikseli. Przykładem może być strata częstotliwości ogniskowej [85] (ang. *Focal Frequency Loss*), która z użyciem dyskretnej transformaty Fouriera porównuje obrazy w dziedzinie częstotliwościowej. Skupiając się na najtrudniejszych do odtworzenia częstotliwościach, wpływa na zauważalną poprawę percepcyjną obrazu w rekonstrukcji tekstur, krawędzi i detali.

Optymalny wybór funkcji straty pozostaje wciąż jednak wyzwaniem, ponieważ subtelny wpływ bardziej złożonych funkcji jest trudny do oceny wyłącznie poprzez wskaźniki numeryczne. W praktyce powinno się ją osobno dopasowywać do specyfiki każdego zadania, a następnie zweryfikować wybór poprzez ocenę tak ilościową (przy pomocy wybranych metryk), jak i jakościową (wizualną). Podejście takie jest jednak bardzo czasochłonne, a co za tym idzie, w większości wypadków nieopłacalne, przez co częstym podejściem w badaniach jest wciąż wykorzystanie najprostszych norm L1 i L2.

3.6.4. Długość procesu uczenia

Istotną kwestią mającą wpływ na skuteczność procesu uczenia jest także poprawny wybór liczby epok, przez które odbywa się szkolenie modelu. W przypadku zbyt krótkiego treningu sieć nie jest w stanie nauczyć się odpowiednio uogólniać danych, natomiast przy zbyt długim treningu pojawia się ryzyko nadmiernego dopasowania do danych treningowych, skutkującego ograniczoną zdolnością sieci do uogólniania. W celu ustalenia wystarczającej

długości tego procesu zazwyczaj wybiera się względnie wysoką liczbę iteracji i stosuje się kryterium wczesnego zatrzymania. Kryterium to zazwyczaj opiera się na osiągnięciu określonej wartości funkcji straty przez model lub też osiągnięcie zakładanej liczby epok, podczas których nie nastąpiła poprawa wyników zwracanych przez model. Jednak takie proste podejścia mogą nie być optymalne w wykrywaniu momentu, gdy sieć przestaje poprawiać swoje zdolności uogólniające, dlatego wciąż opracowywane są inne kryteria mające precyzyjniej określić odpowiedni czas zatrzymania treningu [86].

3.6.5. Rozmiar grup danych

Sieć neuronowa w trakcie każdej epoki wykonuje wiele iteracji, podczas których przetwarza tylko pewną część (grupę) losowo wybranych próbek danych. Wielkość tej paczki danych ma znaczący wpływ na przebieg treningu i końcową wydajność modelu. Wykorzystanie małej grupy powoduje wystąpienie większej wariancji w próbce, co pozwala sieci lepiej dostosować się do danych niewystępujących w zbiorze treningowym. Większe paczki umożliwiają natomiast lepsze wykorzystanie zasobów sprzętowych, jednak w praktyce modele trenowane z użyciem dużych paczek często osiągają gorsze wyniki w kontekście generalizacji; ich zdolność do działania na danych testowych bywa ograniczona w porównaniu z modelami uczonymi na mniejszych paczkach [87]. Dobór odpowiedniego rozmiaru grupy jest zatem kompromisem pomiędzy jakością wyników a efektywnością obliczeniową, który w dużej mierze zależy od charakterystyki danych, architektury sieci oraz celu projektu.

3.6.6. Pozostałe hiperparametry

W sieciach neuronowych istnieje jeszcze wiele innych hiperparametrów, które mają znaczenie dla efektywności treningu oraz końcowej jakości działania modelu. Warto przytoczyć tu między innymi różne metody inicjalizacji wag sieci; sposoby i proporcje podziału danych na treningowe, walidacyjne i testowe; podejścia do augmentacji (zwiększeniu różnorodności) wykorzystywanych danych; rodzaje normalizacji danych, a także wiele innych. Każde z możliwych ustawień tych wartości wpływa na to, jak model przetwarza dane, jak szybko i stabilnie się uczy oraz jak dobrze potrafi uogólniać wiedzę na niewidziane wcześniej dane. Ważny jest przy tym fakt, że hiperparametry rzadko działają niezależnie —zazwyczaj są ze

sobą mocno powiązane, co dodatkowo utrudnia znalezienie ich optymalnych wartości. Dalsze zagłębianie się w tematykę ich optymalizacji wykracza jednak poza temat niniejszej pracy.

4. Redukcja szumu w obrazach syntetycznych

W poprzednim rozdziale przedstawiono kluczowe zagadnienia związane z procesem trenowania sieci neuronowych, a szczególny nacisk położono na omówienie wpływu poszczególnych parametrów modeli na ich pracę. Należy jednak podkreślić, że równie istotnym zagadnieniem jest prawidłowy dobór danych, na których przeprowadzany jest trening. Zarówno ich ilość, jak i jakość, mają bezpośredni wpływ na działanie wytrenowanej sieci w warunkach docelowych.

W kontekście zadań związanych z redukcją szumu standardowe podejście opiera się na wykorzystywaniu par składających się z obrazów zawierających szum i tych całkowicie go pozbawionych. Jak jednak wyjaśniono w rozdziale drugim, w przypadku danych pochodzących z teleskopów naziemnych takie podejście jest niemożliwie do zrealizowania, bo wszystkie rzeczywiste obrazy są obarczone szumem. Często stosowanym rozwiązaniem tego problemu jest utworzenie zbioru obrazów zaszumionych, by następnie wyznaczyć obrazy pozbawione szumu, korzystając z uśrednienia serii wielu klatek czy stosując wybrany algorytm deterministyczny służący do redukcji szumu. Ponadto, istnieją różne podejścia do samego procesu uczenia sieci, które nie wymagają posiadania obrazów „czystych". Chociaż wytrenowane w ten sposób sieci mogą dostarczać dobrej jakości wyniki, problemem pozostaje brak danych referencyjnych, które pozwoliłyby obiektywnie ocenić rzeczywistą skuteczność sieci. Z tego względu zdecydowano się na wykorzystanie danych syntetycznych we wstępnych eksperymentach opisanych w niniejszym rozdziale.

Celem badań było przede wszystkim sprawdzenie wpływu złożoności sieci typu autoenkoder na jakość redukcji szumu w przetwarzanych obrazach. Przeanalizowano w jakim stopniu zwiększenie rozmiaru wektora kodowań i liczby elementów w poszczególnych warstwach przekłada się na poprawę wyników. Przetestowano różne strategie uczenia modeli, w tym warianty oparte na technikach uczenia sieci bez dostępu do danych niezaszumionych. Wszystkie eksperymenty przeprowadzono dla trzech odmiennych mocy szumu, które uznano za reprezentatywne dla scenariuszów niskiego, średniego i wysokiego poziomów zakłócenia obrazów. Pozwoliło to określić korelację między stopniem degradacji oryginalnych danych a wymaganą złożonością modeli odszumiających. W celu rzetelnej oceny skuteczności zaproponowanych rozwiązań, wyniki działania sieci neuronowych zostały porównane z wynikami uzyskiwanymi przez najlepsze algorytmy deterministyczne.

Poruszona w tym rozdziale tematyka została podzielona na siedem części. W pierwszej z nich przedstawiono charakterystykę wykorzystanych danych syntetycznych oraz opisano metodę ich sztucznego zaszumiania, prowadzącą do utworzenia trzech niezależnych od siebie zbiorów: treningowego, walidacyjnego i testowego. Następnie zaprezentowano powszechnie stosowane do redukcji szumu algorytmy deterministyczne, które posłużyły jako punkt odniesienia w dalszej analizie. W trzeciej części szczegółowo wyjaśniono, które techniki trenowania sieci wybrano do badań. W kolejnej sekcji skupiono się na opisie zastosowanych architektur autoenkoderów, by zaraz potem przedstawić, w jaki sposób dokonano oceny jakości uzyskiwanych przez nie wyników. Przedostania część obejmuje opis i analizę uzyskanych wyników, natomiast koniec rozdziału poświęcono omówieniu dalszego kierunku badań.

4.1. Wykorzystane dane syntetyczne – zbiór MNIST

Jako zestaw danych syntetycznych wybrano jeden z najbardziej znanych i szeroko wykorzystywanych w dziedzinie uczenia maszynowego zbiorów – MNIST (ang. ***M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology database*) [88]. Jest to kolekcja 70 tysięcy obrazów w skali szarości, które przedstawiają odręcznie napisane cyfry (rysunek 4.1). Mimo że obrazy te są do siebie podobne, charakteryzują się pewną różnorodnością, która zapobiega przetrenowaniu[2] sieci. Zaletą ich stosowania jest także to, że ich jakość jest bardzo łatwo ocenić wizualnie. Z tych powodów zbiór ten stanowi punkt wyjścia do testowania i porównywania ze sobą różnych algorytmów uczenia maszynowego.

Na potrzeby badań zbiór MNIST został podzielony na trzy niezależne części. Przyjęto, że cyfry 4, 7 oraz 8 mają odmienny kształt w porównaniu z pozostałymi, dlatego utworzono z nich zbiór testowy, który wykorzystywano do oceny działania wytrenowanych modeli. Pozostałe dane rozdzielono proporcjonalnie: 85% przeznaczono na zbiór treningowy, a 15% na walidacyjny, przy zachowaniu podziału względem poszczególnych cyfr. Szczegółowy podział zbioru MNIST na podzbiory przedstawiono w tabeli 4.1.

[2] W przypadku treningu sieci możemy mieć do czynienia zarówno z przetrenowaniem (ang. *overfitting*), jak i niedotrenowaniem (ang. *underfitting*). Pierwsze zjawisko polega na tym, że sieć neuronowa uczy się zbyt dokładnie danych treningowych (łącznie z ich szumem), co prowadzi do spadku jej skuteczności na nowych danych. Natomiast niedotrenowanie oznacza, że model nie nauczył się wystarczająco dobrze obecnych w danych zależności i nie jest w stanie osiągać najlepszych wyników. Oba zjawiska świadczą o niepoprawnym dopasowaniu modelu i wymagają dalszego dostosowania jego architektury, parametrów lub danych.

Rysunek 4.1. Przykładowe obrazy ze zbioru MNIST.

Tabela 4.1. Podział zbioru MNIST na podzbiory.

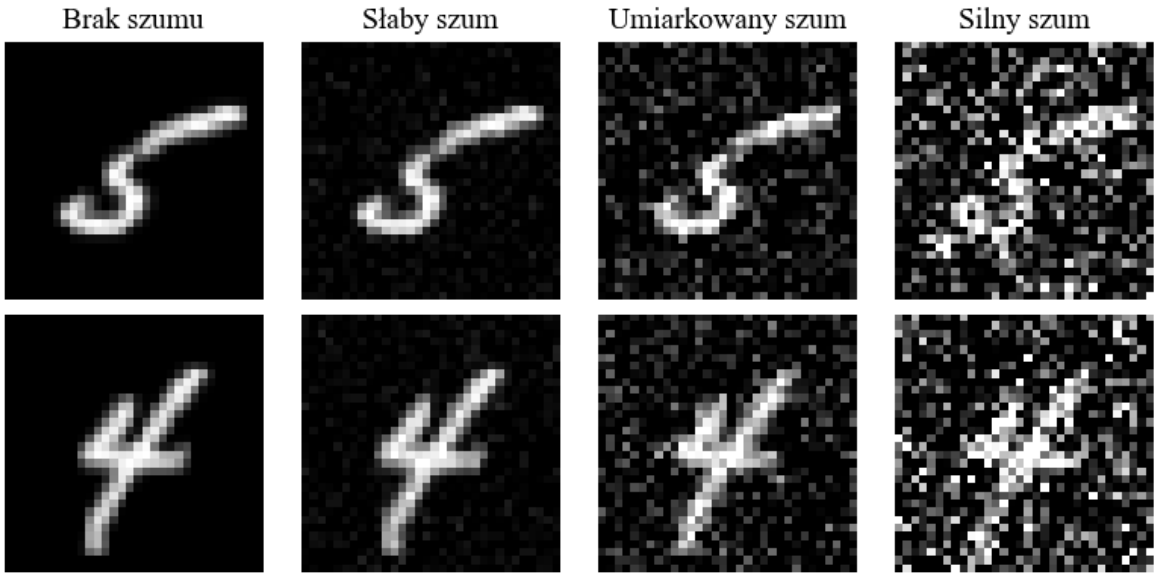
Typ podzbioru danych	**Liczba obrazów z MNIST w podzbiorze**
Treningowy	41 700
Walidacyjny	7 358
Testowy	20 942

Pierwotny rozmiar obrazów ze zbioru MNIST wynosi 28 × 28 pikseli, jednak na cele eksperymentów obrazy zostały przeskalowane do rozmiaru 32 × 32 piksele, by ułatwić zmianę ich rozmiaru przestrzennego przez testowane sieci. Dodatkowo wartości każdego obrazu *im* zostały znormalizowane[3] do zakresu [-1; 1] z wykorzystaniem największych i najmniejszych wartości, jak opisano we wzorze 4.1.

$$im = 2 * \left(\frac{im - \min(im)}{\max(im) - \min(im)} - \frac{1}{2}\right) \tag{4.1}$$

[3] Normalizacja zakresu danych w sieciach neuronowych polega na przekształceniu danych wejściowych tak, aby miały określoną średnią i rozkład (zwykle zerowa średnia i jednostkowe odchylenie standardowe). Dzięki temu proces uczenia staje się szybszy i bardziej stabilny, ponieważ sieć nie musi dostosowywać się do danych o różnych skalach. W praktyce jest często realizowana w sieciach poprzez wykorzystanie warstwy normalizacji wsadowej jako pierwszej warstwy przetwarzającej dane surowe.

W celu przeprowadzenia eksperymentów związanych z redukcją szumu, zdefiniowano trzy poziomy zakłóceń, odpowiadające różnym poziomom szumu gaussowskiego. Moce szumu zostały określone na podstawie wizualnej analizy wpływu szumu na jakość obrazu i sklasyfikowano je jako słabą, umiarkowaną oraz silną. Zaszumione obrazy wygenerowano poprzez dodanie do oryginalnych danych losowego szumu o rozkładzie normalnym, gdzie wartość odchylenia standardowego σ wynosiła odpowiednio 0,1, 0,15 i 0,85 dla każdej mocy szumu. Tak zaszumione obrazy (określane jako **Noise**) miały za zadanie zastąpić pojedyncze klatki obserwacyjne. Równolegle przygotowano zestaw obrazów o zmniejszonym natężeniu szumu (**Clean**), które odpowiadają klatce uśrednionej ze 100 obrazów z serii. Takie uśrednienie powinno dziesięciokrotnie ($\sqrt{100} = 10$) zmniejszyć natężenie szumu, w związku z czym zastosowano odpowiednio mniejsze wartości parametru σ: 0,01, 0,015 oraz 0,085. Na rysunku 4.2 przedstawiono wpływ dodania szumu na jakość danych na przykładzie dwóch losowo dobranych obrazów (cyfry 4 i 5).



Rysunek 4.2. Wpływ mocy szumu na jakość obrazów.

Tak przygotowane dane umożliwiły zastosowanie wielu odmiennych sposobów do uczenia sieci przy zachowaniu kontroli nad poziomem degradacji sygnału. W przypadku danych treningowych zaszumienia te były generowane osobno w każdej epoce, co umożliwiło wielokrotne zwiększenie przykładów w zbiorze treningowym. Natomiast dla zbioru

walidacyjnego i testowego zaszumienie zostało dodane tylko raz, przed rozpoczęciem całego eksperymentu – pozwoliło to obiektywnie ocenić działanie poszczególnych sieci. Zbiory te składały się zatem z par obrazów: obraz zaszumiony **Noise** oraz obraz porównawczy całkowicie pozbawiony szumu, nazwany **Reference**.

4.2. Algorytmy deterministyczne w redukcji szumu

W dziedzinie cyfrowego przetwarzania obrazów stosuje się szeroki wachlarz podejść do redukcji szumu. Najliczniejszą grupą rozwiązań są deterministyczne metody filtracji operujące na pojedynczym, zaszumionym obrazie. Biorąc pod uwagę zakres wykorzystywanej przez nie informacji przestrzennej, można takie algorytmy podzielić na dwie grupy: metody działające lokalnie i nielokalnie.

Lokalne metody filtracji operują w obrębie niewielkiego fragmentu obrazu, tak zwanego okna, otaczającego przetwarzany piksel. Przykładami takich rozwiązań są klasyczne filtry operujące na podstawie lokalnych statystyk, takich jak średnia i wariancja [89], filtry wykorzystujące ważoną wartość mediany [90], filtry Wienera [91] oraz filtry bilateralne [92]. Choć podejścia te cechują się stosunkowo niską złożonością obliczeniową i są łatwe do implementacji, wykazują one istotne ograniczenia, szczególnie w przypadku wysokiego poziomu szumu. Zachodzi wtedy silna degradacja korelacji pomiędzy pikselami w niewielkich sąsiedztwach, co sprawia, że metody te nie są w stanie działać prawidłowo.

W odpowiedzi na niedoskonałości metod lokalnych, zaproponowano algorytmy nielokalne, które analizują powtarzające się wzorce w całym obrazie, niezależnie od ich lokalizacji przestrzennej. Pionierskim rozwiązaniem był algorytm nielokalnych średnich (ang. *Non-Local Means*, NLM) [93], który wykorzystuje średnie ważone wartości pikseli z całego obrazu, bazując na podobieństwie bloków. NLM okazał się znacząco lepszy od pozostałych podejść, a jego dalszy rozwój doprowadził do szeregu ulepszeń [94][95][96], czego zwieńczeniem stało się opracowanie algorytmu BM3D (ang. *Block-Matching and 3D Filtering*) [97]. Poprzez grupowanie podobnych bloków i ich filtrowanie w domenie współrzędnych trójwymiarowych, BM3D był w stanie osiągnąć wyniki znacząco przewyższające te uzyskiwane przez inne metody i stał się w efekcie punktem odniesienia do oceny pozostałych algorytmów redukcji szumu.

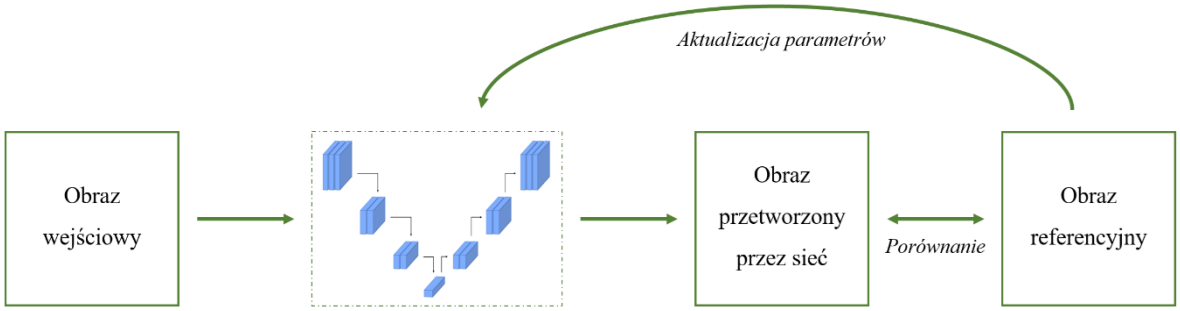
Oprócz wspomnianych metod, na przestrzeni lat rozwinięto również inne techniki, oparte na bardziej złożonych modelach matematycznych i statystycznych. Do najważniejszych z nich zaliczyć można: metodę K-SVD (ang. *K-Singular Value Decomposition*) [98], techniki oparte na adaptacyjnej analizie głównych składowych [99], mieszaniny Gaussa w dziedzinie falkowej [100], probabilistyczne modele mieszanin Gaussowskich [101], a także podejścia Bayesowskie [102][103]. Pomimo zaawansowania teoretycznego wiele z tych rozwiązań okazało w kontekście rzeczywistych danych obrazowych się mniej skuteczne niż BM3D [104], dzięki czemu BM3D pozostaje do dziś standardem referencyjnym w ocenie technik odszumiających. Z tego powodu za algorytmy porównawcze przyjęto właśnie ten algorytm wraz z jego prostszą wersją NLM.

4.3. Sposoby trenowania sieci

Na rysunku 4.3 przedstawiono typowy schemat procesu uczenia sieci, który był szerzej omówiony w poprzednim rozdziale. Chcąc wytrenować sieć, która przyjmuje na wejściu dane zaszumione, a zwraca dane pozbawione szumu, musimy zazwyczaj dysponować parami obrazów: obraz wejściowy (zaszumiony) i obraz referencyjny (pozbawiony szumu). Metoda ta powszechnie jest nazywana trenowaniem Noise2Clean[4], jednakże dla celów prowadzonych badań została ona przemianowana na Noise2Reference. Pary takich obrazów wykorzystuje się do iteracyjnego procesu aktualizacji parametrów modelu, a tym samym do zwiększania jego dokładności.

W rzeczywistych sytuacjach stosunkowo rzadko zdarza się, że mamy dostęp do takich par danych. Zazwyczaj uzyskanie idealnych obrazów pozbawionych szumów (**Reference**) jest albo bardzo trudne w realizacji, albo wręcz niemożliwe. W astronomii czy w obrazowaniu medycznym zaszumienie danych jest zjawiskiem zawsze obecnym podczas rejestracji obrazów, co rodzi poważne problemy w procesie treningu.

[4] Skrócone nazwy sposobów trenowania sieci zapisuje się często w języku angielskim jako X2Y, bo cyfrę „2” wymawia się jak przyimek „to”. Metodę X2Y należy zatem rozumieć jako szkolenie sieci z wykorzystaniem danych typu X, które po przetworzeniu porównuje się **do** danych typu Y.

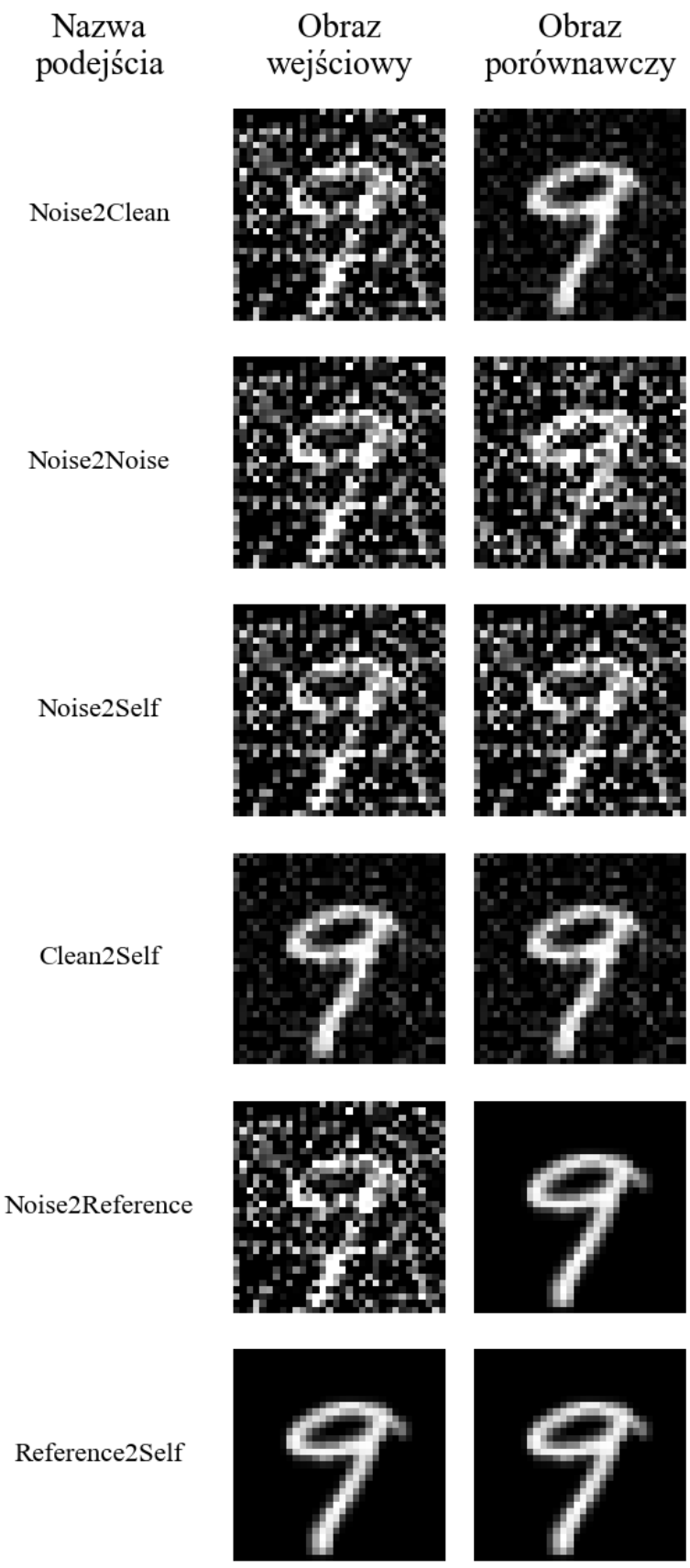


Rysunek 4.3. Schemat trenowania sieci neuronowej.

W ostatnich latach opracowano wiele odrębnych metod uczenia sieci, które operują wyłącznie na danych zaszumionych. Pierwszą z nich jest podejście Noise2Noise [105], które opiera się na wykorzystaniu wielu różnych, niezależnych od siebie realizacji szumu na tym samym obrazie. Osiągane tą metodą wyniki w wielu sytuacjach są w stanie dorównać klasycznemu podejściu Noise2Reference. Rozwinięciem Noise2Noise są podobne podejścia, które wykorzystują maskowanie wybranych wartości obrazu, by móc na tej podstawie ominąć najtrudniejszy element do rekonstrukcji, czyli losowy szum; spośród nich na szczególną uwagę zasługują metody Noise2Void [106][107], Self2Self [108] i Noise2Self [109]. Oprócz nich stosuje się także metody takie jak Noisy-As-Clean [110] czy Noisier2Noise [111], które przyjmują obrazy zaszumione jako referencje, a na wejście sieci przekazują te same obrazy z dodatkowo powiększonym szumem.

Bazując na opisanych metodach, opracowano sześć różnych, reprezentatywnych technik uczenia, które zostały wykorzystane do wytrenowania wybranych modeli. Każda z nich została szczegółowo opisane w kolejnych podpunktach, natomiast na rysunku 4.4 przedstawiono porównanie wykorzystywanych przez nie danych wejściowych sieci i obrazów porównawczych. Należy przy tym podkreślić, że metoda treningu w żaden sposób nie zmienia docelowego sposobu działania sieci, który nadal opiera się on na przetwarzaniu obrazów zaszumionych w celu redukcji szumu.

.



Rysunek 4.4. Zestawienie obrazów wykorzystywanych przez rozważane techniki uczenia sieci w przypadku silnego zaszumienia.

4.3.1. Noise2Clean

W przypadku danych astronomicznych jest to najprostsze podejście do trenowania sieci, gdy nie ma dostępu do pozbawionych szumu danych referencyjnych. Trenując sieć w ten sposób, można sprawdzić, czy da się ominąć potrzebę wielokrotnej akwizycji danych i uzyskać zauważalną redukcję szumu, korzystając jedynie z pojedynczego obrazu przetworzonego przez wybraną sieć.

4.3.2. Noise2Noise

Podejście to opiera się na przetwarzaniu przez sieć obrazu zaszumionego, by następnie porównać go z obrazem zawierającym ten sam sygnał, ale z **inną** realizacją szumu. Jest to pewna zmiana w stosunku do podejścia Noise2Noise opisanego w [105] – w oryginalnej implementacji obrazem porównawczym był obraz z **dowolną** realizacją szumu, czyli mógł to być również obraz wejściowy sieci. W przypadku przeprowadzonych eksperymentów zdecydowano się ograniczyć wybór obraz referencyjnego, by dokonać obiektywnego porównania z wynikami osiąganymi przez Noise2Self.

4.3.3. Noise2Self

Ta technika pozwala przetestować własności kompresyjne stosowanych sieci. Autoenkoder powinien w ramach pracy zapamiętywać jedynie istotną część informacji, między innymi położenie i ogólny kształt struktur, a tracić tę część odnoszącą się do występujących w obrazach szczegółów, jak punktowe zaszumienie. Zastosowanie tego podejścia umożliwiło ocenę stopnia złożoności modelu, dla którego wpływ kompresji zaczyna słabnąć.

4.3.4. Clean2Self

Metoda analogiczna do Noise2Self, przy czym tutaj kompresja jest testowana na danych uśrednionych. Ogólna jakość takich obrazów jest zatem o wiele lepsza od jakości obrazów zaszumionych, co pozwala dodatkowo ocenić wpływ jakości danych wejściowych na działanie wytrenowanych modeli, czyli na stosowaną przez nie kompresję.

4.3.5. Noise2Reference

W sytuacjach rzeczywistych najrozsądniejszym podejściem jest uczenie sieci w trybie Noise2Clean. Jednakże, istotną kwestią jest to, jak Noise2Clean wypada w porównaniu z podejściem domyślnym, Noise2Reference. To między innymi w celu wykonania tej weryfikacji zdecydowano się przeprowadzić opisane w tym rozdziale eksperymenty na danych syntetycznych, gdyż tylko w ich przypadku możliwe jest wykorzystanie obrazów całkowicie pozbawionych szumu.

4.3.6. Reference2Self

Reference2Self jest odpowiednikiem wymienionych wcześniej Noise2Self oraz Clean2Self, zrealizowanym z wykorzystaniem najlepszych jakościowo danych, których nie da się uzyskać w praktycznych zastosowaniach. Została głównie użyta w celu porównania jej wydajności ze wspomnianymi dwoma technikami.

4.4. Testowane architektury

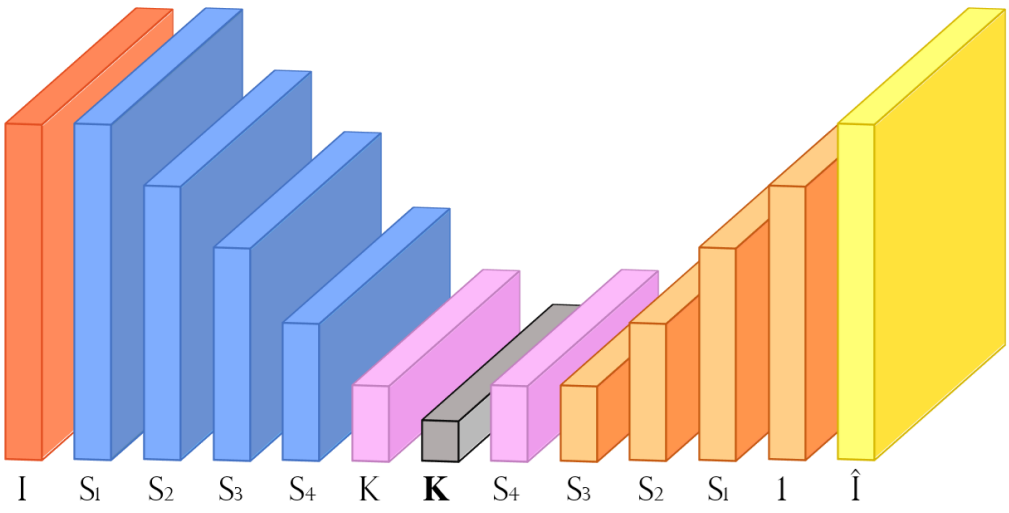
Jednym z najważniejszych elementów praca naukowych poświęconych wykorzystaniu sieci neuronowych jest dokładny opis wykorzystanych architektur. Precyzyjna dokumentacja tych struktur nie tylko zwiększa przejrzystość prowadzonych badań, ale także umożliwia wykorzystanie opisanych modeli przez innych. W większości sytuacji autorzy prac naukowych zamieszczają szczegółowe informacje dotyczące zastosowanych modeli, takie jak liczba, rodzaj i konfiguracja wykorzystanych warstw, liczba map cech, dobór funkcji aktywacji i metod normalizacji czy sposób inicjalizacji wag. Opisom tym często towarzyszą schematy blokowe, wykresy przepływu danych oraz tabele prezentujące kluczowe parametry każdej warstwy, co dodatkowo ułatwia interpretację omawianych rozwiązań.

W przypadku struktur omówionych w tym rozdziale, a także w dalszych częściach pracy, postanowiono przede wszystkim skupić się na odpowiednim zaprezentowaniu sieci w formie graficznej. Każdą z nich przedstawiono przy pomocy kolorowych bloków, które odpowiadają wybranym warstwom lub jednostkom funkcjonalnym, a ich zmieniająca się wielkość świadczy o zmianie wymiarowości przestrzennej obrazów o połowę. Wszystkie schematy są przy tym szczegółowo opisane, co powinno ułatwić zrozumienie ich działania. Na

przestrzeni pracy starano się zachować spójność względem dopasowania poszczególnych kolorów do odpowiednio działających bloków, lecz w przypadku każdej z sieci mogą pojawić się pewne subtelne różnice, na co należy zwrócić uwagę.

Na rysunku 4.5 przedstawiono strukturę sieci autoenkodera, która została wykorzystana do opracowania architektur omówionych w tym rozdziale. Sieć ta składa się z bloków, których kolory odpowiadają wyszczególnionym elementom modelu:

a. blok czerwony oznacza obraz wejściowy sieci I;
b. blok żółty symbolizuje obraz wyjściowy sieci $\hat{I}$;
c. blok niebieski oznacza jednostkę funkcjonalną składającą się z warstwy konwolucyjnej o jądrze 3×3 i kroku równym 2, warstwy normalizacji wsadowej oraz warstwy aktywacji w postaci tangensa hiperbolicznego (w tej kolejności);
d. blok pomarańczowy odpowiada jednostce funkcjonalnej złożonej z warstwy dekonwolucyjnej o jądrze 3×3 i kroku równym 2, warstwy normalizacji wsadowej oraz warstwy aktywacji w postaci tangensa hiperbolicznego (w tej kolejności);
e. blok filetowy oznacza warstwę w pełni połączoną, która przetwarza dwuwymiarowy obraz do jednowymiarowego wektora kodowań (pierwszy fioletowy blok od lewej) lub przetwarza wektor kodowań z powrotem do postaci dwuwymiarowego obrazu (drugi fioletowy blok od lewej);
f. blok ciemnoszary symbolizuje wektor kodowań.



Rysunek 4.5. Struktura testowanych architektur.

Jak można zauważyć, sieć ta jest symetryczna – składa się z czterech warstw zmniejszających wymiar, czterech warstw zwiększających wymiar, wektora kodowań oraz dwóch warstw w pełni połączonych, które zajmują się przetwarzaniem danych dwuwymiarowych w jednowymiarowe i odwrotnie. Dodatkowo, pod każdym z bloków znajduje się specjalne oznaczenie. Odpowiada ono liczbie map cech danych **po przetworzeniu** przez daną warstwę (S_1, S_2, S_3, S_4 i 1) lub liczbie elementów w wektorze kodowań (K). Różne liczebności tych map cech wpływają na ogólną złożoności sieci (liczbę jej parametrów), co ma kluczowe znaczenia dla redukcji wymiarowości danych. Bardziej skomplikowane modele powinny być w stanie utworzyć lepsze reprezentacje danych. Zmieniając liczbę map cech, utworzono trzy różne architektury, które opisano w tabeli 4.2. Dodatkowo, przedstawiono w niej liczbę parametrów sieci przy zastosowaniu 64 kodowań dla każdej z architektur.

Tabela 4.2. Liczba wszystkich parametrów sieci i konfiguracja map cech w poszczególnych warstwach.

Architektura	**Zmieniająca się liczba map cech w S_X**				**Liczba parametrów (przy 64 kodowaniach)**
1	$S_1 = 4$	$S_2 = 8$	$S_3 = 4$	$S_4 = 8$	4 035
2	$S_1 = 8$	$S_2 = 12$	$S_3 = 16$	$S_4 = 20$	16 599
3	$S_1 = 48$	$S_2 = 72$	$S_3 = 96$	$S_4 = 120$	427 739

Oprócz złożoności sieci opisanej liczbą map cech ważnym hiperparametrem sieci jest długość wektora kodowań. To on stanowi „wąskie gardło” autoenkoderów i decyduje o tym, jak wiele informacji może być przeniesionych przez sieć. Dla każdej z architektur dobrano wartości K do przetestowania, przy czym dla najbardziej złożonej sieci rozmiary te są znacząco większe niż dla pozostałych. Różnica ta posłużyła dokładniejszemu oszacowaniu wpływu parametru K na pracę całej sieci. W tabeli 4.3 przedstawiono zestawienie długości wektora kodowań dla każdej z architektur.

Co ważne, przy szkoleniu tych sieci zdecydowano się na wykorzystanie nieco bardziej złożonej funkcji straty. Jest ona w tym wypadku sumą trzech oddzielnych funkcji: omówionych już L1 i L2, a także zmodyfikowanej metryce Wassersteina [112], która znalazła pewne zastosowanie w trenowaniu sieci neuronowych [113][114], również w przypadku autoenkoderów [115]. Trening odbył się w 300 epokach, w trakcie których sieci przetwarzały

100 grup złożonych z 64 obrazów. Jako optymalizator wybrano podstawową wersję algorytmu Adam, a współczynnik uczenia określono jako stałą wartość 0,001.

Tabela 4.3. Testowane długości wektora kodowań.

Architektura	Testowane długości wektora kodowań K
1	8, 16, 32, 64
2	8, 16, 32, 64
3	64, 128, 256, 384

4.5. Ocena jakości wyników

W analizie danych obrazowych ocena jakości przetworzonych wyników opiera się w głównej mierze na subiektywnej ocenie wizualnej. Podejście to, mimo że bezpośrednio oddaje percepcyjną jakość obrazu z punktu widzenia człowieka, nie jest wystarczające w kontekście analizy dużych zbiorów danych. Konieczne staje się zastosowanie miar ilościowych, które umożliwiają powtarzalne i obiektywne porównywanie wyników. W tej części pracy omówiono dwie metryki użyte do takiej analizy, a także wyjaśniono statystyczne podejście do prezentowania wyników.

4.5.1. Metryki porównawcze – PSNR i SSIM

Do ilościowej oceny obrazów stosuje się obecnie wiele metryk, spośród których jedną z najprostszych i najczęściej stosowanych jest PSNR (ang. *Peak Signal-to-Noise Ratio*). Wskaźnik ten opiera się na stosunku kwadratu maksymalnej możliwej jasności piksela obrazu MAX_{im} do wartości błędu średniokwadratowego jasności, MSE, obliczonego między rozważanym obrazem *im* a obrazem porównawczym *ref*, tak jak przedstawiono w równaniu 4.2. Wyrażona w decybelach wartość informuje o tym, jak duże różnice zniekształcenia występują pomiędzy obrazami – im wyższa wartość PSNR, tym bardziej obrazy są do siebie zbliżone. Metryka ta jednak uwzględnia jedynie różnice w wartościach poszczególnych pikseli, ignorując przy tym nierzadko znacznie istotniejsze różnice w strukturze obrazów.

$$PSNR(im, ref) = 10 \log_{10} \left(\frac{MAX_{im}^2}{MSE(im, ref)} \right) \tag{4.2}$$

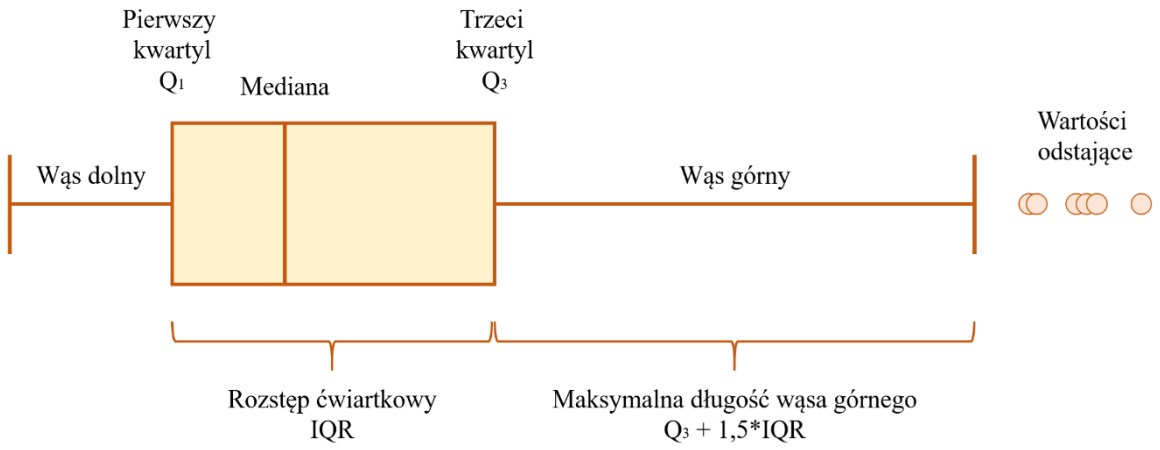
W związku z ograniczeniami metryki PSNR, zastosowano dodatkową, bardziej złożoną metrykę, SSIM (ang. *Structural Similarity Index Measure*) [116], która została zaprojektowana w sposób bardziej odpowiadający ludzkiej ocenie jakości. Metryka ta analizuje obrazy w oknach lokalnych, porównując średnie jasności, wariancje oraz kowariancje, co pozwala lepiej ocenić podobieństwo struktur i podobny poziom kontrastu obrazu. Przyjmowane wartości mieszczą się w zakresie [-1; 1], jednak dla obrazów rzeczywistych częściej jest to zakres [0; 1], przy czym 1 oznacza, że porównywane obrazy są identyczne, a 0 świadczy o całkowitym braku podobieństwa. W wielu sytuacjach trudno wyznaczyć prostą zależność pomiędzy SSIM i PSNR, wobec czego zaleca się stosowanie obydwu metryk równocześnie [117], by móc dokonać dokładniejszego, bardziej holistycznego, porównania obrazów.

4.5.2. Wykresy pudełkowe

W kontekście analizy dużych zbiorów danych liczbowych tradycyjne metody opisu, takie jak średnia czy odchylenie standardowe, mogą nie wystarczyć do pełnego zrozumienia ich struktury. Zazwyczaj potrzebna jest dodatkowo znajomość ich rozkładu, a także wartości ekstremalnych i odstających. Do tego celu wykorzystuje się różne typy wykresów, spośród których najpopularniejszym i najprostszym do analizy jest wykres pudełkowy przedstawiony na rysunku 4.6.

Główna część tego wykresu, czyli prostokąt zwany „pudełkiem", jest obszarem wyznaczonym pomiędzy pierwszym (Q_1) a trzecim (Q_3) kwartylem zbioru wartości analizowanych danych. W jego wnętrzu znajduje się środkowe 50% danych, przy czym ich mediana (zaznaczana osobną belką) niekoniecznie znajduje się pośrodku prostokąta – jej umiejscowienie jest zależne od symetryczności rozkładu opisywanych wartości. Długość boku pudełka, czyli różnica pomiędzy trzecim a pierwszym kwartylem określana jest jako rozstęp międzyćwiartkowy (ang. *interquartile range*, IQR). Przy pomocy wartości IQR wyznacza się maksymalną długość tak zwanych „wąsów". Wąsy określają odległość kwartylu do wartości skrajnej (minimum lub maksimum), jednakże nie sięgają dalej niż 1,5 wartości IQR. Jeżeli jakiś element danych nie mieści się w opisanym zakresie, uznawany jest za wartość odstającą i oznacza się go przy pomocy kropki lub krzyżyka. W zależności od liczby wartości odstających

i charakteru danych, wartości odstających nie wykorzystuje się w dalszej analizie (mogą być wynikiem błędów pomiarowych) lub zwraca się na nie szczególną uwagę (mogą świadczyć o istnieniu dodatkowych zależności między poszczególnymi próbkami danych).



Rysunek 4.6. Schemat wykresu pudełkowego.

Porównując kilka wykresów pudełkowych umieszczonych obok siebie, można szybko zorientować się, w której grupie dane są bardziej rozproszone i jak zmieniają się ich rozkłady. W efekcie wykresy tego typu są istotnym narzędziem przy porównywaniu wyników przetwarzania danych przez różne metody. Pozwalają dobrze zidentyfikować różnice wprowadzane przez testowane algorytmy, dzięki czemu stanowią podstawę opisu statystycznego wyników przedstawionych w niniejszej pracy.

4.6. Analiza i omówienie wyników

W ramach badań wytrenowano sieci oparte na trzech różnych architekturach charakteryzujących się różną liczbą map cech w warstwach konwolucyjnych, a tym samym różną liczbą wszystkich parametrów (rozmiarem sieci). Przeprowadzono badania nad związkiem pomiędzy długości wektora kodowań a pracą tych struktur. Z racji tego, że obrazy wejściowe mają rozmiar 32×32 piksele, dobór liczby kodowań odpowiadał wymienionym stopniom kompresji:

a. 8 kodowań – 0,78 %,
b. 16 kodowań – 1,56 %,

c. 32 kodowań – 3,13 %,
d. 64 kodowań – 6,25 %,
e. 128 kodowań – 12,50 %,
f. 256 kodowań – 25,00 %,
g. 384 kodowań – 37,50 %.

Każdą z sieci wytrenowano z wykorzystaniem różnych stopni zaszumienia, co pozwoliło na wygenerowanie trzech zestawów wykresów porównawczych (rysunki 4.7, 4.8 i 4.9), po jednej dla każdej mocy szumu. Lewa kolumna każdego rysunku przedstawia porównanie wartości PSNR, natomiast w prawej kolumnie znajdują się wartości SSIM. Każdy z rzędów odpowiada jednej z testowanych architektur, co jest też dodatkowo oznaczone. Wyniki na każdym wykresie są pogrupowane względem wykorzystanej liczby kodowań w kolejności rosnącej, przy czym po prawej znajdują się wartości wspomnianych metryk dla danych surowych (oryginalnych, nieprzetworzonych) i wyników działania algorytmów BM3D i NLM. Wyniki dla każdego podejścia są oznaczone różnymi kolorami, a także są uporządkowane w kolejności, w której są wymienione w legendzie każdego rysunku.

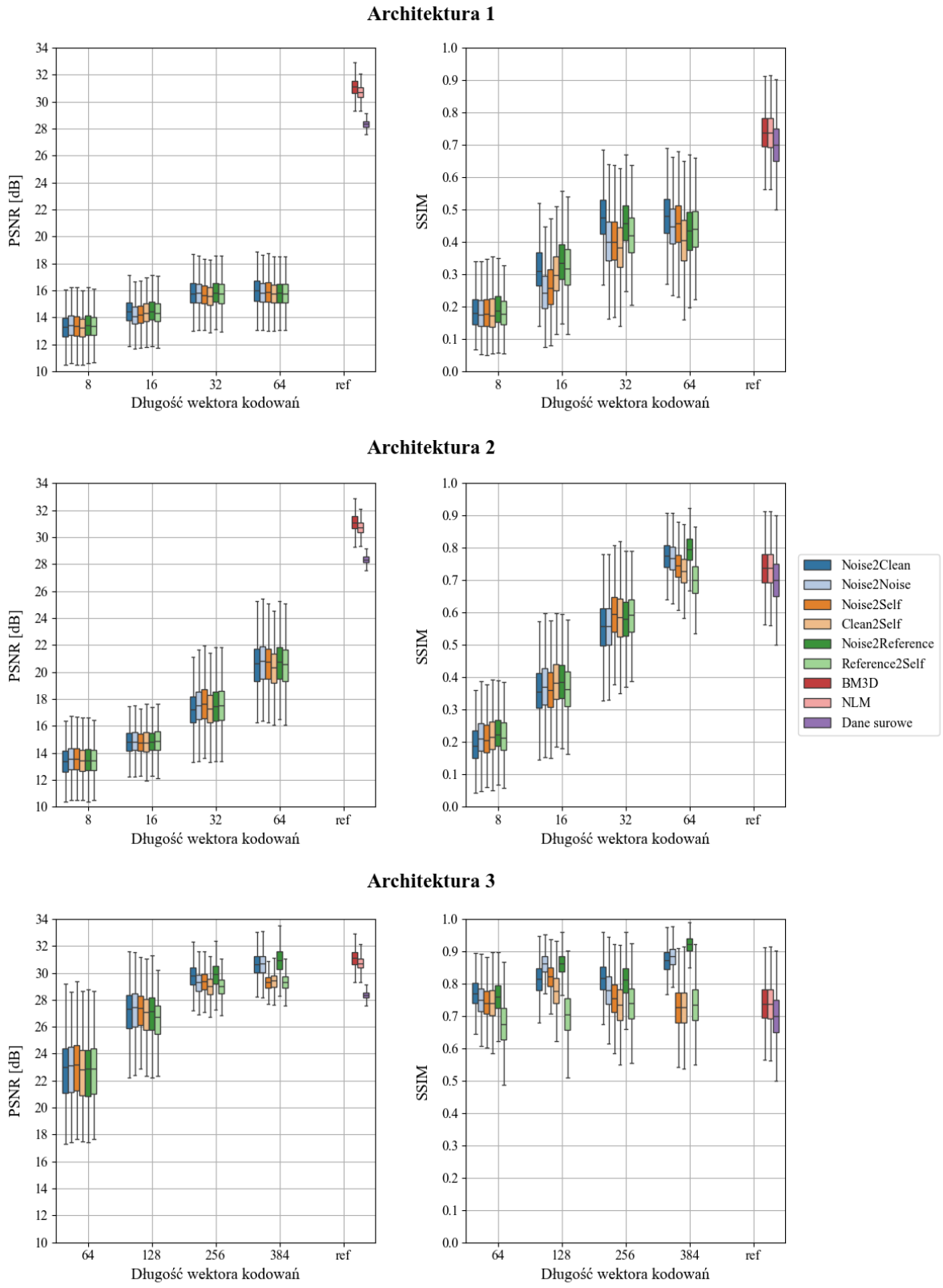
Pierwszym wnioskiem płynącym z analizy wyników jest to, że rosnąca moc szumu zauważalnie pogarsza jakość obrazów, jednak użycie algorytmów deterministycznych statystycznie skutkuje poprawą ich jakości, czyli są to odpowiednio dobrane algorytmy porównawcze. Drugą ważną kwestią są różnice w działaniu architektury pierwszej i drugiej. Wraz ze wzrostem liczby kodowań można zauważyć zauważalną zmianę w jakości wyników uzyskiwanych przez architekturę 2, jednak w przypadku architektury 1 zachodzące zmiany są minimalne. Świadczy to o tym, że sieci tego typu mają niewystarczające możliwości obliczeniowe, by były w stanie skutecznie przetworzyć dane o zadanym rozmiarze. Szczególnie widać to w przypadku słabego i umiarkowanego szumu, gdy zastosowanie tak małych sieci degraduje jakość obrazów wejściowych. Sytuacja wygląda nieco lepiej w przypadku obrazów mocno zaszumionych, dla których poprawiają one wartości PSNR we wszystkich analizowanych przypadkach. Nie jest to jednak znacząca zmiana, wobec czego można się spodziewać, że będzie ona trudna to zauważenia; obserwację tę potwierdzają bardzo niskie wartości SSIM, świadczące o zaniku wielu elementów strukturalnych.

W przypadku drugiej, większej architektury można zauważyć, że jakość sieci w każdym przypadku ulega poprawie. Przy niewielkim zaszumieniu sieci wciąż pogarszają jakość poszczególnych fragmentów obrazu, co można zauważyć po niższych wartościach PSNR, niemniej dla 64 kodowań są w stanie uzyskać wysoką wartość SSIM. Dla porównania

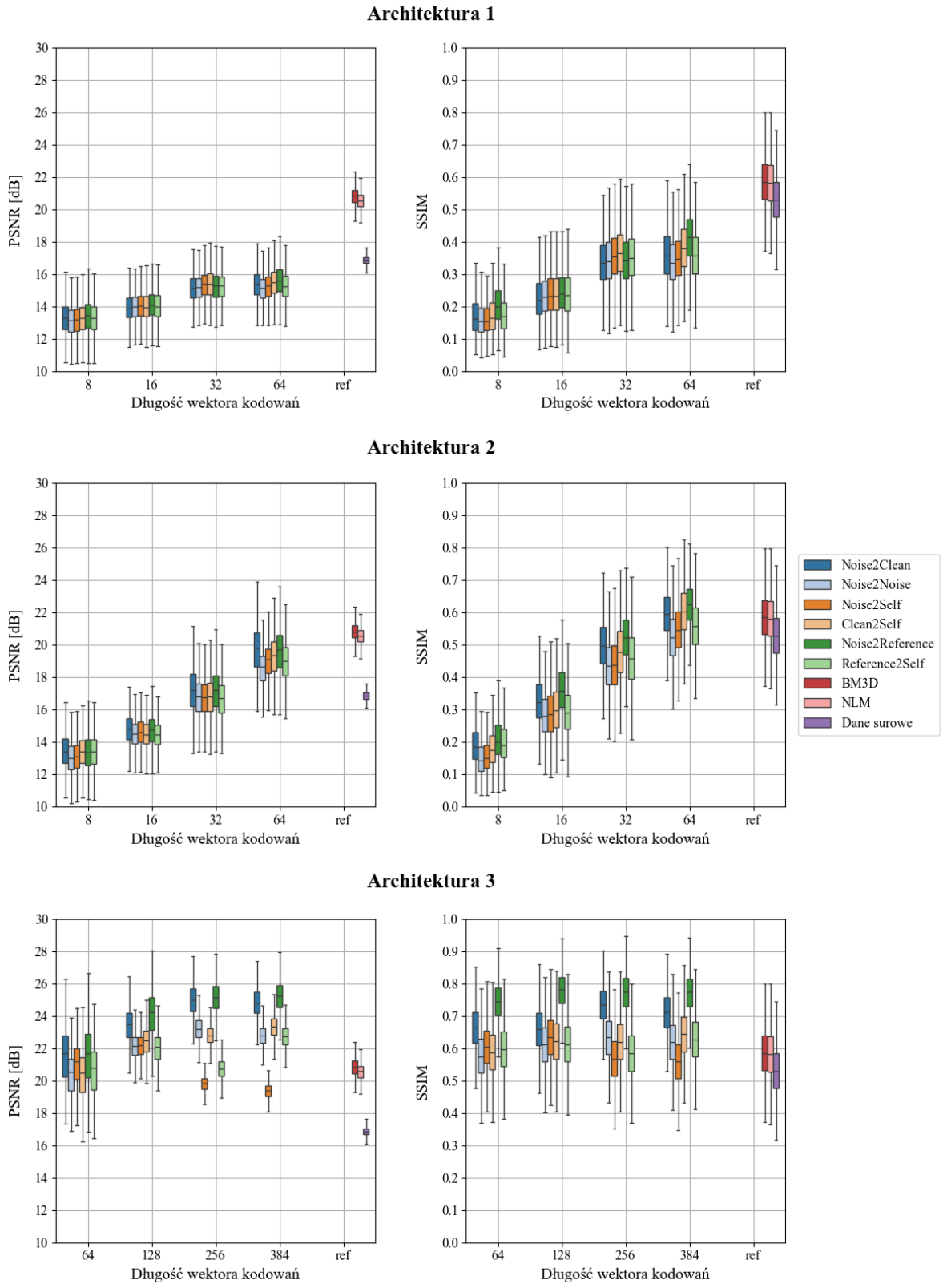
architektura 3 ma przy 64 kodowaniach ponad 25 razy więcej parametrów niż architektura 2, co pozwala jej osiągnąć lepsze wyniki; mimo to poprawa ta nie jest aż tak znacząca, jak można by się spodziewać po takiej różnicy rozmiarów. Ograniczenie to w dużej mierze jest spowodowane niewielką długością wektora kodowań – korzystając z tak małej ilości informacji (jedynie 6,25 % wielkości pierwotnych danych), niezwykle trudno jest dokładnie odtworzyć oryginalny obraz. Jak można zauważyć, dalsze zwiększanie długości tego wektora pozytywnie wpływa na działanie sieci.

Spośród wszystkich punktów analizy najistotniejsza jest jednak ocena wpływu poszczególnych metod uczenia na działanie sieci w rozważanych scenariuszach. Porównanie takie najlepiej jest przeprowadzić na przykładzie wyników architektury 3, ponieważ wyłącznie ona była w stanie poradzić sobie z redukcją szumu dla wszystkich testowanych mocy. Przyglądając się wynikom dla słabego szumu, można zaobserwować, że wszystkie metody osiągają początkowo zbliżone wyniki, jednak wraz ze wzrostem liczby kodowań metody Noise2Clean, Noise2Noise i Noise2Reference zaczynają osiągać znacznie lepsze wyniki niż metody bazujące na uczeniu się prostej kompresji i porównaniu wyniku z obrazem wejściowym. Jest to prawdopodobnie spowodowane tym, że w tych podejściach modele musiały w pewnym stopniu modyfikować obrazy wejściowe, by móc je porównać z obrazami innymi; w ten sposób nauczyły się lepiej uogólniać swoją pracę na niewidziane wcześniej dane.

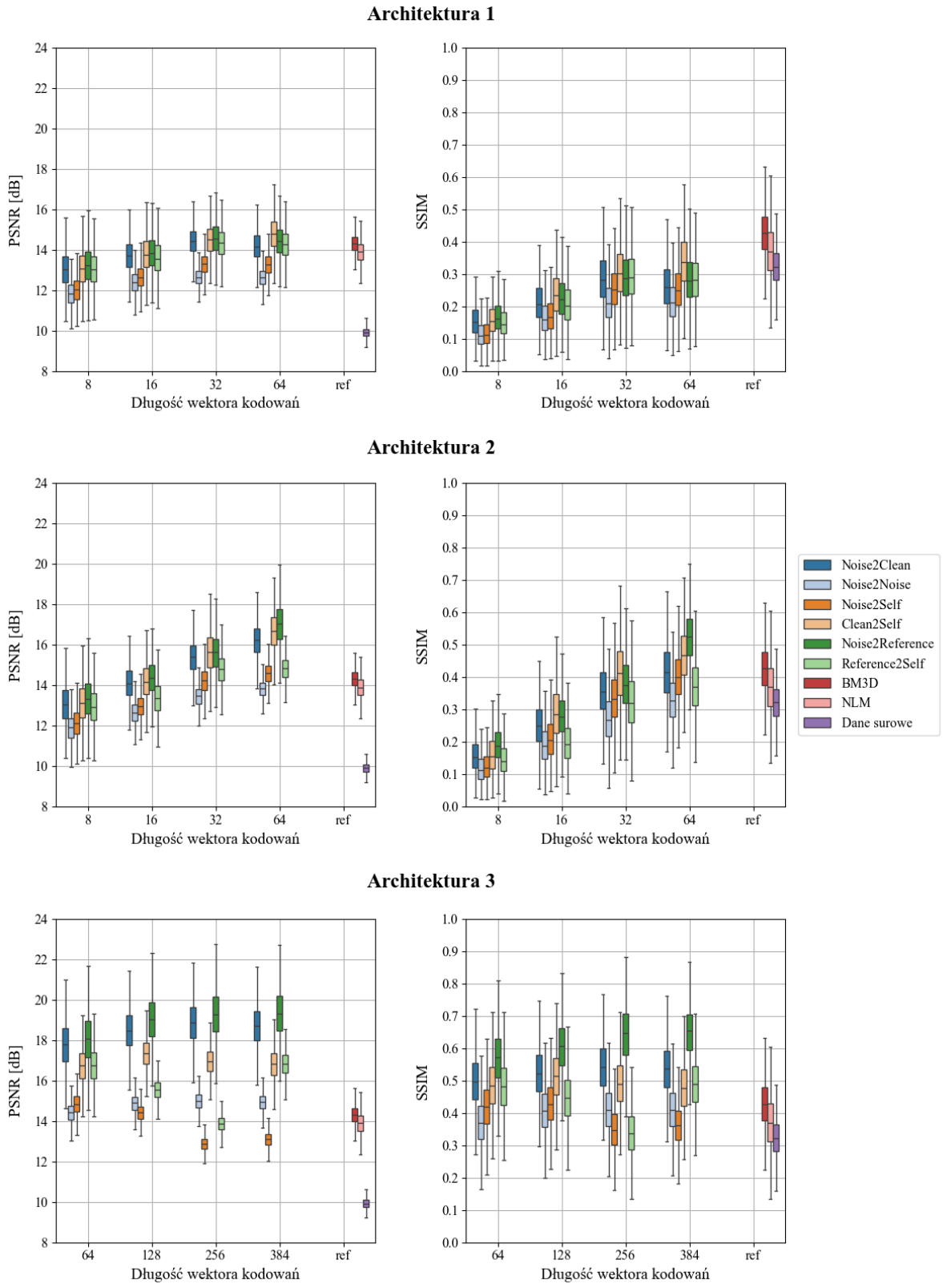
Przy mocniejszych mocach szumu można zaobserwować zjawisko podobne – trening Noise2Reference osiąga bezsprzecznie najlepsze wyniki, co zgadza się z założeniami. Najważniejsze w kontekście reszty pracy jest jednak to, że Noise2Clean uzyskuje wyniki zbliżone do Noise2Reference, co udowadnia, że można stosować to podejście na danych rzeczywistych. Co ciekawe, zaraz za tymi metodami plasują się Clean2Self i Reference2Self. Jak widać, w przypadku większego zaszumienia sieci te są w stanie lepiej przetworzyć ogólną strukturę danych, ignorując przy tym losowy szum. Nieco gorszą metodą jest Noise2Noise, jednak we wszystkich przypadkach podejście to poprawia dane lub przynajmniej nie pogarsza ich jakości. Świadczy to o tym, że na danych astronomicznych można wykorzystać również tę metodę i spodziewać się dobrych jakościowo rezultatów. Najgorzej z testowanych podejść wypadło Noise2Self, które dla większych długości wektora kodowań zaczyna stopniowo tracić na jakości. Jest to zapewne spowodowane zbyt dużą ilością zachowywanej informacji – gdy kompresja staje się słabsza, model zaczyna przenosić szum na obraz wynikowy.



Rysunek 4.7. Porównanie wyników działania algorytmów przy niskim poziomie szumu.



Rysunek 4.8. Porównanie wyników działania algorytmów przy umiarkowanym poziomie szumu.

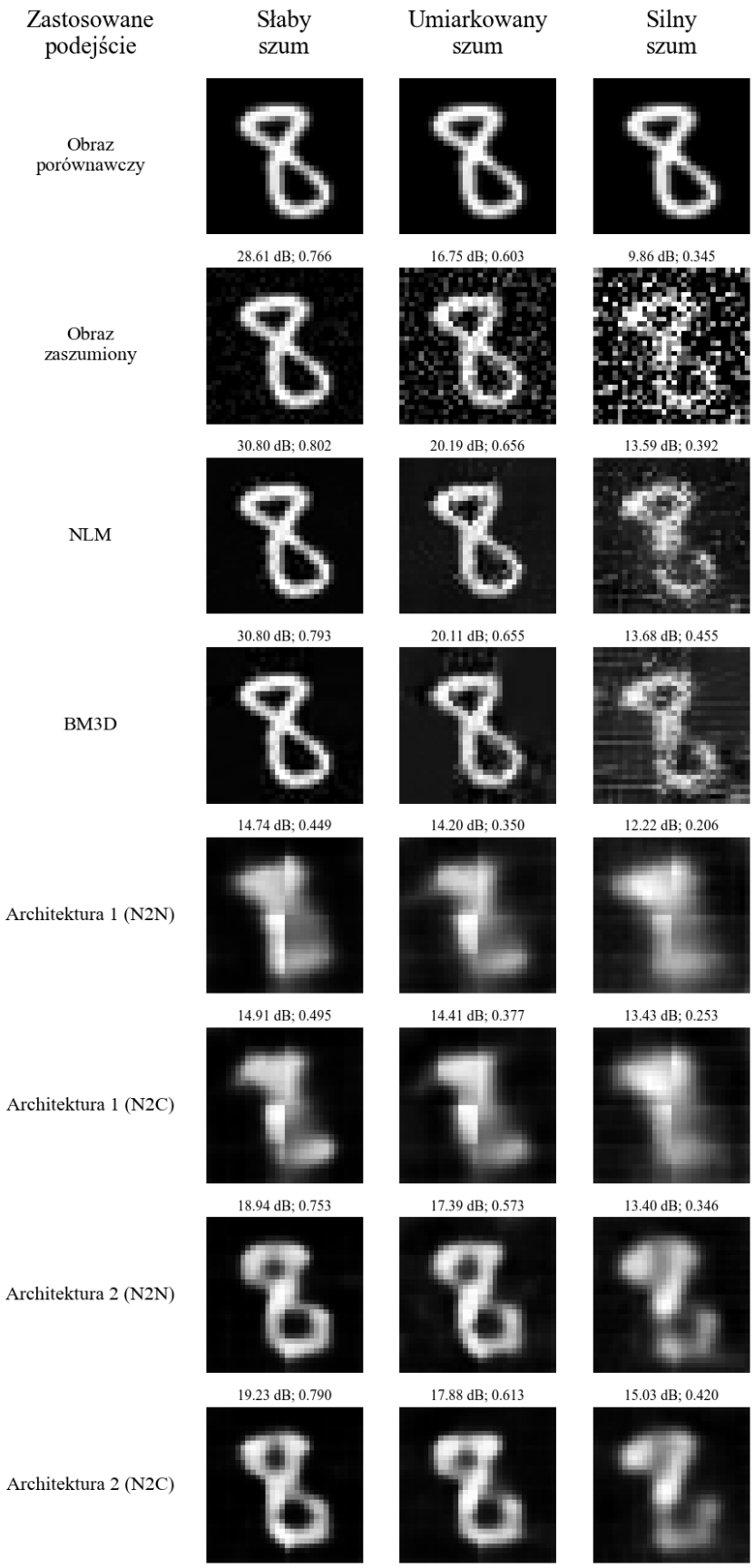


Rysunek 4.9. Porównanie wyników działania algorytmów przy wysokim poziomie szumu.

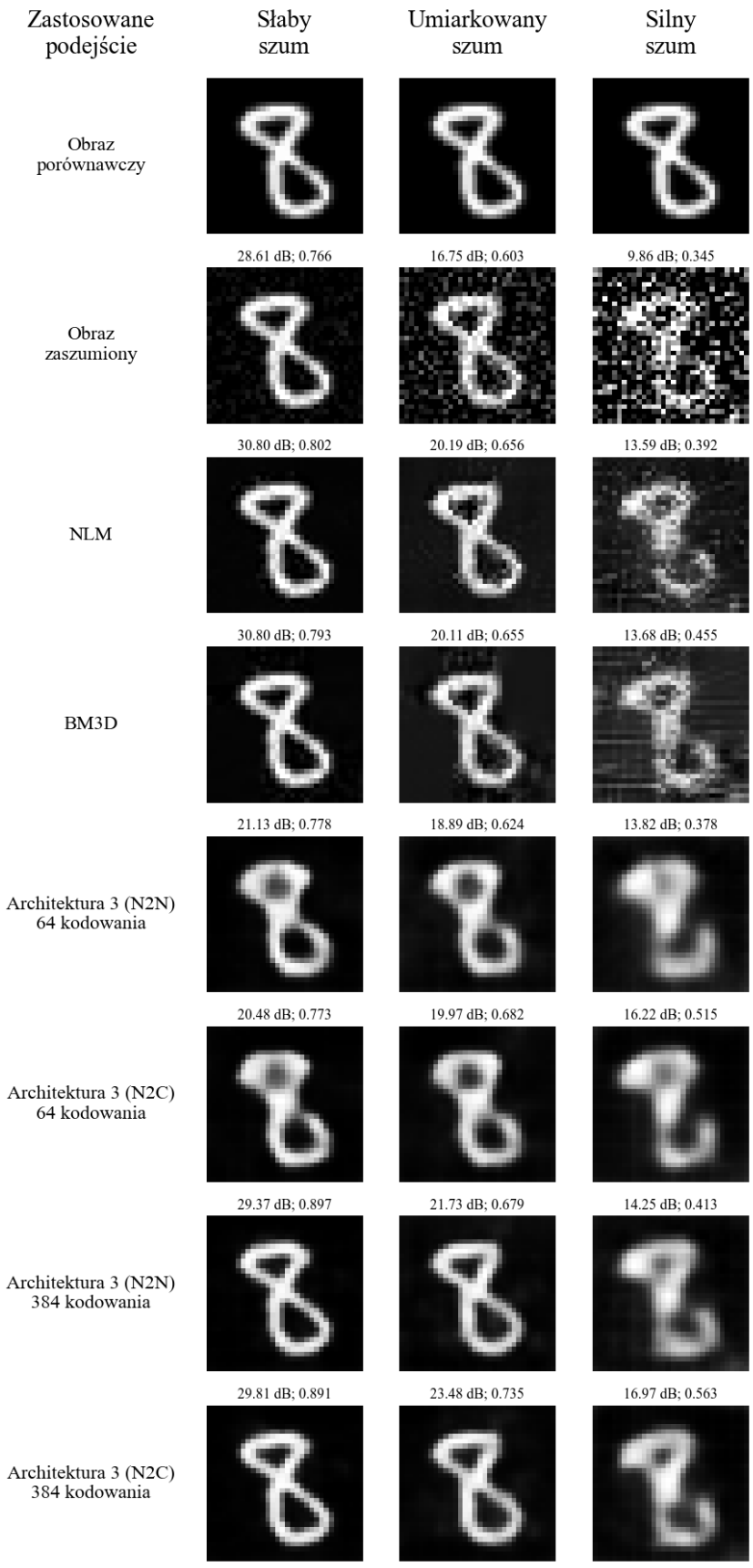
Innym interesujący wniosek wynika z porównania sprawności filtracji z uznanymi metodami deterministycznymi reprezentowanymi przez metody BM3D oraz NLM. Przy niskim poziomie szumu, obie metody są w stanie skutecznie poprawiać jakość obrazu, podczas gdy rozwiązania oparte na sieciach dostarczają danych o poziomie zakłócenia (zniekształcenia) pogarszającym znacząco obraz surowy. Dopiero na poziomie średnich szumów, a zwłaszcza dla szumów silnych, przewaga sieci, szczególnie tych bardziej rozbudowanych, zaczyna być zauważalna. Rezultaty otrzymane w eksperymencie pokazują, że decyzja pomiędzy użyciem rozwiązań opartych na autoenkoderach a algorytmów deterministycznych, może być trudna i wymaga wiedzy o poziomie szumu względem użytecznego sygnału. Dopiero odpowiednie eksperymenty mogą potwierdzić przewagę poszczególnych rozwiązań. Algorytmy BM3D oraz NLM okazały się bezpiecznym rozwiązaniem, zawsze poprawiającym jakość obrazów. Niemniej rozbudowane sieci badane w tym eksperymencie, w przypadku obecności silnego szumu, potrafią wykazywać znacznie większą skuteczność.

By móc potwierdzić powyższą analizę, należy dokonać dodatkowej oceny wizualnej wyników. Na rysunkach 4.10 i 4.11 przedstawiono porównanie wyników działania wybranych sieci i klasycznych algorytmów. Poszczególne kolumny odpowiadają różnym mocom szumu, natomiast każdy wiersz przedstawia wyniki działania dla danej metody z wartościami PSNR oraz SSIM. Jako przetwarzany obraz wybrano przykład zawierający cyfrę 8, ponieważ zawiera dwie pętle wypełnione czarną przestrzenią, co czyni ją jedną z bardziej złożonych struktur spośród wykorzystanych danych syntetycznych.

Na rysunku 4.10 porównano działanie architektur 1 i 2 zawierających po 64 kodowania i uczonych z wykorzystaniem metod Noise2Noise (N2N) i Noise2Clean (N2C). Jak widać, najmniejsze sieci nie są w stanie nauczyć się poprawnego przetwarzania danych, wobec czego ich praca w każdym przypadku kończy się dodatkową degradacją danych wejściowych. Uzyskanie przez nie wysokiej wartości PSNR przy mocnym szumie podkreśla słabości tej metryki, którą w pewnym stopniu niweluje SSIM, którego wartości świadczą o utracie znacznej części informacji strukturalnej. W przypadku bardziej rozbudowanej architektury obraz wynikowy lepiej odpowiada stawianym mu warunkom. Chociaż, te sieci wciąż nie są w stanie odtworzyć dokładnych struktur, można w ich wypadku mówić o zauważalnym podobieństwie do oryginału. Wartości użytych metryk również potwierdzają lepszą jakość danych.



Rysunek 4.10. Wizualne porównanie redukcji szumu przez algorytmy deterministyczne oraz architektury 1 i 2 uczone metodami N2C i N2N dla 64 kodowań.



Rysunek 4.11. Wizualne porównanie redukcji szumu przez algorytmy deterministyczne oraz architektury 3 uczonej metodami N2C i N2N dla 64 i 384 kodowań.

Przedstawione na rysunku 4.11 wyniki działania architektury 3 osiągają jeszcze wyższą dokładność odszumionych obrazów. Szczególnie widoczne jest to przy wykorzystaniu 384 kodowań, gdy sieci okazują się lepsze w działaniu od porównywanych algorytmów deterministycznych. Co szczególnie istotne, w przypadku obrazów przetworzonych przez dowolną z sieci, szum zostaje całkowicie wyeliminowany, a problemem pozostaje poprawne odwzorowanie istotnych obiektów. W przypadku klasycznych algorytmów sytuacja taka nie ma miejsca – wraz ze wzrostem natężenia szumu, pojawiają się artefakty związane z działaniem tych metod. Obserwacja ta stanowi podstawowy wniosek z zaprezentowanych w tym rozdziale rozważań, ponieważ przemawia na korzyść zaproponowanego podejścia do redukcji szumu w obrazach astronomicznych poprzez użycie sieci kompresujących dane.

4.7. Opis dalszego kierunku badań

W rozdziale przeanalizowano wpływ hiperparametrów sieci typu autoenkoder na ich zdolności redukcji szumu w przypadku prostych obrazów monochromatycznych. Jak wykazano, sieci te, odpowiednio dostrojone, są w stanie osiągnąć wyniki o lepszej jakości od tych uzyskiwanych przez klasyczne algorytmy deterministyczne, lecz nie jest to regułą. Co istotne, efekt ten zachodzi nawet w sytuacji, gdy do procesu treningu nie zostaną wykorzystanie dane „czyste", czyli niezawierające szumu. Uzasadnia to wybór tych struktur sieci do dalszych eksperymentów na danych rzeczywistych.

Wykorzystanie autoenkoderów z wektorem kodowania o stałej długości wymaga jednak, by wszystkie dane wejściowe miały jednakowe wymiary przestrzenne, co jest ograniczeniem elastyczności modelu. W wielu sytuacjach wiąże się to z potrzebą przeskalowania lub przycięcia obrazów tak, by odpowiadały danemu rozmiarowi. Dodatkowo warstwy w pełni połączone analizują każdy z pikseli osobno, natomiast w przypadku przetwarzania obrazów bardzo często zależy nam na zachowaniu zależności pomiędzy sąsiednimi pikselami.

Rozwiązaniem tej niedogodności są sieci w pełni konwolucyjne FCN (ang. *Fully Convolutional Network*) [57], które nie zawierają warstw gęstych. Działanie takich sieci nie opiera się na przetwarzaniu danych do jednowymiarowego wektora kodowań, lecz do tensora o zadanej liczbie map cech o nieokreślonych wymiarach przestrzennych (wymiary te są nieokreślone, ponieważ w żaden sposób nie wpływają na pracę danej warstwy, jak wyjaśniono w poprzednim rozdziale). Dzięki takiemu podejściu, sieci typu FCN są w stanie przetwarzać

obrazy o dowolnych[5] rozmiarach. W związku z tym, dalsze badania zostały skierowane na wykorzystanie sieci FCN. Pozwoliło to między innymi znacząco uprościć proces uczenia i zbierania danych wymaganych do przeprowadzenia eksperymentów, co jest szczególnie istotne w kontekście wykorzystanych danych słonecznych, które zostaną omówione w rozdziale 6.

[5] Rozmiar ten może być dowolny jednak jedynie w pewnym stopniu. Sieci FCN zazwyczaj wielokrotnie zmieniają wymiarowość obrazu podczas jego przetwarzania. Dla czterokrotnego zmniejszenia rozmiaru przestrzennego o połowę, obraz w najgłębszej warstwie będzie 16-krotnie zmniejszony względem oryginału. By uniknąć możliwej zmiany rozmiaru obrazu wyjściowego, należy zadbać, by obraz wejściowy mógł bez przeszkód zostać poddany takim operacjom (w opisanej sytuacji jego długość i szerokość muszą być opisane wartością podzielną przez 16).

5. Redukcja szumu w danych obrazowych nocnego nieba

W poprzednim rozdziale opisano wpływ złożoności sieci i zastosowanej strategii treningowej na możliwości redukcji szumu. Wyciągnięte z przeprowadzonych badań wnioski umożliwiły przeprowadzenie dalszych badań z wykorzystaniem typowych danych astronomicznych. Zdecydowano się przetestować wpływ rozmiaru sieci i metod treningowych Noise2Noise oraz Noise2Clean na jakość przetworzonych danych rzeczywistych.

Zbiór danych utworzono z wielu serii obserwacyjnych, każdej złożonej z pięćdziesięciu klatek wykonanych jedna po drugiej przy czasie ekspozycji równym 500 ms. Stosując uśrednianie serii, otrzymano obrazy referencyjne typu **Clean**. Oczekiwanym rezultatem działania sieci było uzyskanie obrazów o jakości lepszej od pojedynczych klatek serii **Noise**, docelowo porównywalnych z **Clean**. Analiza wyników nie ograniczyła się w tym wypadku jedynie do zestawienia wartości wybranych metryk i oceny wizualnej. Zastosowano dodatkowe miary porównawcze, które umożliwiły weryfikację kluczowych cech danych astronomicznych. Główny nacisk położono na możliwość poprawnej detekcji gwiazd, które w pojedynczych klatkach surowych mogą być w dużej mierze „przysłonięte” przez szum. Pozwoliło to porównać znacznie trudniejsze w opisie charakterystyki kształtu obiektów gwiezdnych, które mogą nie być dobrze odzwierciedlone przez standardowe wskaźniki. Przeanalizowano również, jak duże zmiany zachodzą w zmierzonych położeniach i jasnościach gwiazd, które mogły być wykryte w obrazie.

Do eksperymentów wykorzystano struktury autoenkoderów o różnym stopniu złożoności, a rozwiązania wzbogacono dodatkowo o sieci typu U-Net. Pierwsza z nich powstała jako rozwinięcie wykorzystanego autoenkodera, a za drugą przyjęto jedno ze sprawdzonych w literaturze rozwiązań. Z algorytmów deterministycznych zdecydowano się ponownie na wykorzystanie BM3D, a zrezygnowano z NLM, który każdorazowo zwracał bardzo zbliżone, acz nieco gorsze wyniki. Zastąpiono go znacznie prostszym, a zarazem szybszym, filtrem Wienera, co pozwoliło lepiej ocenić stopień złożoności zagadnienia redukcji szumu w obrazach rzeczywistych w kontekście zastosowania najprostszych rozwiązań.

Przeprowadzone badania opracowano w formie niniejszego rozdziału, na który składa się sześć oddzielnych części. W pierwszej z nich przybliżono charakter wykorzystanych danych, by w kolejnej przedstawić metody oceny ich jakości: wybrane metryki, technikę oceny poprawności detekcji, a także podejście do analizy astrometrycznej i fotometrycznej. Trzecia z sekcji obejmuje spis najnowszych sieci neuronowych, które są obecnie stosowane w redukcji

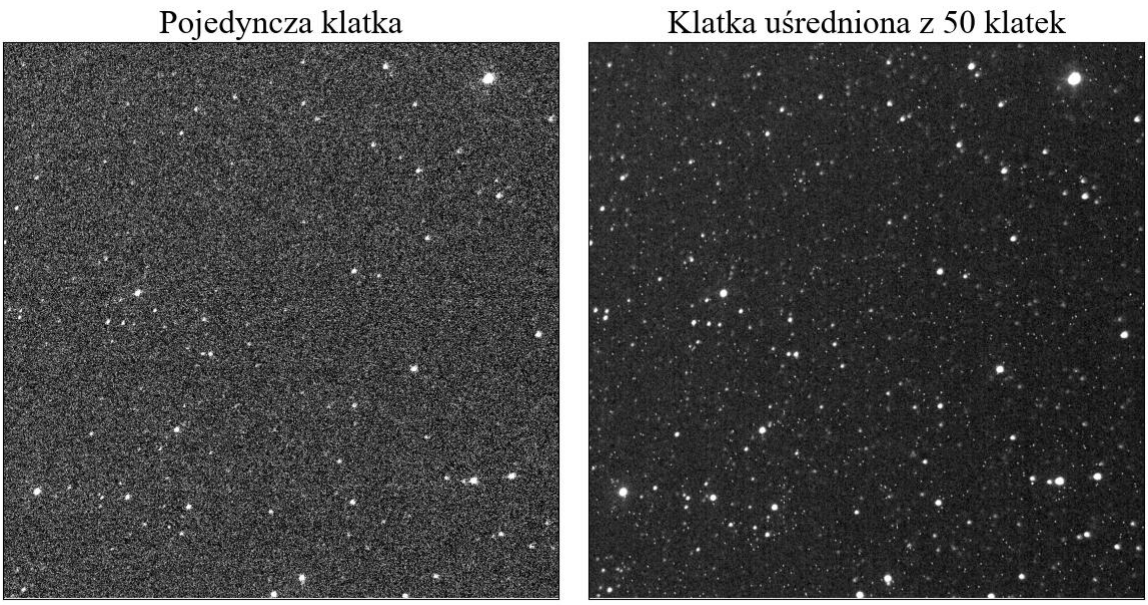
szumu na obrazach astronomicznych. Czwartą część poświęcono dokładnemu omówieniu testowanych sieci. Przedostatni podrozdział poświęcony jest analizie otrzymanych wyników, a w ostatnim poruszono temat dalszego kierunku badań.

5.1. Wykorzystane dane

Do treningu i testów użyto zestawu danych zebranych podczas jednej nocy obserwacji, na który składają się serie zdjęć różnych obszarów nieba w seriach złożonych z 50 klatek o czasie ekspozycji równym 500 ms. Do akwizycji wykorzystano zaprezentowany na rysunku 2.6 teleskop Newtona o średnicy lustra głównego 30 cm wyprodukowany przez firmę ASA. Teleskop ten wyposażono w kamerę chłodzoną CMOS ASI ZWO 1600MM. Pozyskane tą aparaturą surowe zdjęcia mają rozdzielczość 3520×4656 pikseli i skalę kątową piksela równą 0,636 ”/piksel. Zestaw tego typu można uznać za reprezentatywny przykład instrumentu niewielkiego obserwatorium astronomicznego.

Z tak utworzonego zbioru wybrano dwie serie, których obrazy przycięto do rozmiaru 1024×1024, centrując je na obszarach z największą liczbą jasnych gwiazd; posłużyły one walidacji i testom. Centralny fragment obrazu gwarantuje również brak istotnych wad optycznych, które, szczególnie w konstrukcji teleskopu Newtona, pojawiają się wraz z oddalaniem się od osi optycznej. W trakcie treningu wykorzystano pozostałe serie, z których w każdej epoce wybierano losowe, pokrywające się fragmenty o rozmiarze 128×128 pikseli. Każda z grup tych wycinków zostawała następnie z pewnym prawdopodobieństwem poddawana procesom rotacji oraz symetrycznym odbiciom względem osi pionowej, poziomej lub głównej przekątnej obrazu. Pozwoliło to dodatkowo zwiększyć rozmiar zbioru treningowego.

Na rysunku 5.1 przedstawiono porównanie nieprzetworzonej klatki ze zbioru testowego z wynikiem uśrednienia całej serii. Jak można zauważyć, na klatce uśrednionej szum został w znacznym stopniu zredukowany, dzięki czemu można wyszczególnić więcej gwiazd niż na obrazie surowym. Odpowiada to przyjętym założeniom i pozwala porównać wyniki działania testowanych rozwiązań.



Rysunek 5.1. Porównanie pojedynczej klatki surowej i klatki uśrednionej z całej serii (czas ekspozycji dla pojedynczej klatki wynosi 500 ms).

5.2. Ocena jakości przetworzonych obrazów

Wyniki działania sieci zostały ponownie opisane ilościowo za pomocą wykresów pudełkowych, które pozwoliły ocenić statystyczny wpływ poszczególnych metod na jakość przetwarzanych obrazów. Do analizy wykorzystano metryki stosowane do oceny obrazów astronomicznych, zweryfikowano możliwość detekcji gwiazd, a także obliczono ich odchyłki położenia i jasności.

5.2.1. Metryki obrazowe – PSNR i FSIM

Początkowo planowano ponownie wykorzystać metryki PSNR i SSIM. W trakcie analizy okazało się jednak, że wartości SSIM we wszystkich przypadkach wynoszą więcej niż 0,999, a różnice występują dopiero na czwartym miejscu po przecinku. Chociaż może to świadczyć o braku istotnych artefaktów wprowadzanych przez testowane metody, ogranicza to możliwość rzetelnej oceny wyników z wykorzystaniem tej metryki. Biorąc pod uwagę, że wykrywalne gwiazdy zajmują tylko niewielką część powierzchni obrazów, a większość stanowi jednorodne tło, trudno wyciągnąć z takiej analizy wnioski dotyczące zmian lokalnych w obrębie obiektów zainteresowania jakim są gwiazdy.

W związku z tym metryka SSIM została zastąpiona przez wskaźnik podobieństwa cech FSIM [118] (ang. *Feature Similarity Index*) będący jej rozwinięciem. Chociaż miara ta jest bardziej złożona obliczeniowo, stosuje się ją w szczegółowych analizach, bo lepiej opisuje cechy istotne dla ludzkiej percepcji. Wartości FSIM mieszczą się w zakresie [0; 1], przy czym 1 oznacza obraz identyczny, a 0 całkowity brak podobieństwa.

5.2.2. Detekcja gwiazd

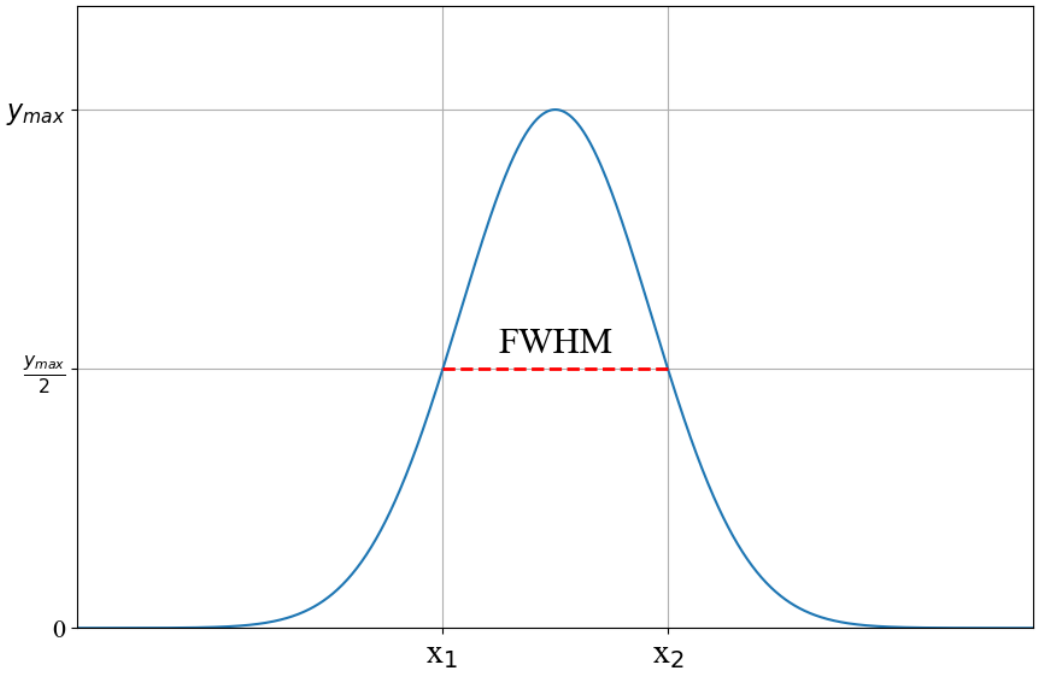
Do detekcji gwiazd na obrazach wykorzystano algorytm DAOFIND [119] zaimplementowany w bibliotece Photutils [120] jako DAOStarFinder. Jest to powszechnie stosowane rozwiązanie w technikach przetwarzania danych astronomicznych z teleskopów optycznych. Działanie tego algorytmu można opisać w sześciu punktach:

1) Na początku wyznaczone zostają podstawowe statystyki analizowanego obrazu, czyli wartości mediany i odchylenia standardowego σ.
2) Od każdego piksela obrazu odejmowana jest mediana, by tło nie dominowało w procesie detekcji, a źródła punktowe bardziej się wyróżniały.
3) Użytkownik ustawia wartości parametrów detektora. Pierwszym z nich jest wartość progu detekcji p_d, który określa, od jakiej wartości poszczególne piksele mogą być zakwalifikowane jako część potencjalnego źródła światła. Parametr ten jest najczęściej określany jako wielokrotność σ. Zazwyczaj p_d mieści się w zakresie od 3σ do 7σ, a zalecaną wartością jest 5σ, która zachowuje kompromis pomiędzy wykrywaniem gwiazd a pomijaniem szumu.
4) Następnie dopasowywana jest wartość szerokości połówkowej FWHM (ang. *Full Width at Half Maximum*) profilu gwiazdy w obrazie. Jest to wielkość liczbowa równa odległości pomiędzy dwoma punktami x_1 i x_2, dla których rozważana funkcja, profil gwiazdy, osiąga połowę swojej maksymalnej wartości (rysunek 5.2). W praktyce astronomicznej zakłada się, że FWHM dla obiektów punktowych w umiarkowanych warunkach stabilności atmosfery wynosi 3”, wobec czego dla wykorzystanych zdjęć ustalono tę wartość na 5 pikseli (5 pikseli pomnożone przez skalę kątową piksela, 0,636 ”/piksel, wynosi w przybliżeniu 3”).
5) Trzecim parametrem jest minimalna odległość w pikselach pomiędzy obiektami, które podlegają działaniu algorytmu. Dla wszystkich testów została ona ustawiona na

5 pikseli, co odpowiada sytuacji w której dwie sąsiednie gwiazdy muszą być oddalone co najmniej o 3 sekundy kątowe.

6) Następnie do każdego punktowego źródła dopasowywany jest dwuwymiarowy model Gaussa, który reprezentuje idealny kształt gwiazdy. Wszystkie punkty spełniające określone kryteria zostają sklasyfikowane przez detektor jako gwiazdy.

Opisany powyżej wybór parametrów w procesie automatycznej detekcji skonsultowano z astronomami oraz osobami zajmującymi się od wielu lat profesjonalnymi pomiarami astronomicznymi.

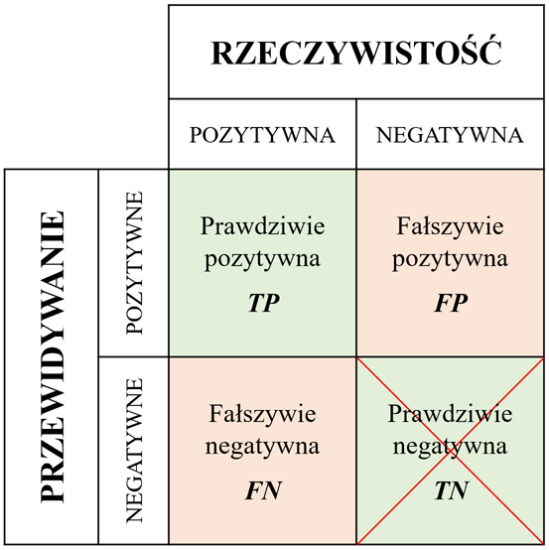


Rysunek 5.2. Przedstawienie szerokości połówkowej (FWHM zaznaczono czerwoną linią przerywaną).

Do wyznaczenia punktu odniesienia wykorzystano uśredniony obraz z serii testowej, *ref*. Dokonano na nim detekcji z uznaną za standard wartością progu p_d równą 5σ i przeprowadzono analizę wizualną, która potwierdziła, że DAOStarFinder poprawnie wyznaczył większość dostrzegalnych na obrazie gwiazd. Wyznaczone pozycje posłużyły do porównania działania testowanych metod. Porównywanie to zostało sprowadzone do zadania klasyfikacji, które oceniono z wykorzystaniem macierzy pomyłek przedstawionej na rysunku 5.3. Pozwoliło to uzyskać statystyki opisujące poprawność detekcji na obrazach *im* w porównaniu do detekcji na obrazie *ref*.

a. *TP* (prawdziwie pozytywna, ang. *True Positive*) – liczba gwiazd wykrytych na *im*, które pokrywają się z detekcjami na *ref*;
b. *FN* (fałszywie negatywna, ang. *False Negative*) – liczba gwiazd, które wykryto na *ref*, a pozostały niewykryte na *im*;
c. *FP* (fałszywie pozytywna, ang. *False Positive*) – liczba obiektów wykrytych na *im*, których nie wykryto na *ref*.

Pozostała statystyka *TN* (prawdziwie negatywna, ang. *True Negative*) oznaczałaby w danym eksperymencie „poprawnie niewykryte" gwiazdy, czyli oznaczenie przez detektor, że analizowany punkt nie jest gwiazdą. Wartość ta nie ma odpowiednika w uzyskanych wynikach, wobec czego została pominięta w analizie.



Rysunek 5.3. Macierz pomyłek z odrzuconą statystyką *TN*.

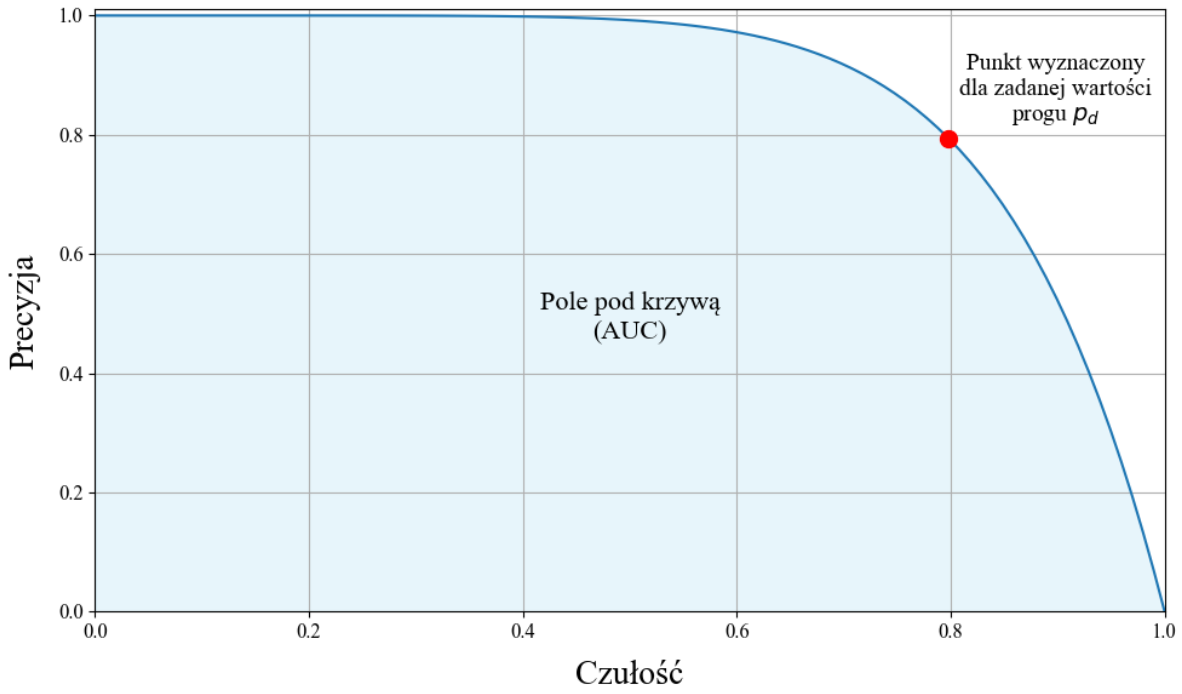
Bazując na wartościach *TP*, *FN* i *TP* wyznaczono wartości precyzji (równanie 5.1) i czułości (równanie 5.2). Pierwsza z tych metryk określa, jaka część obiektów wykrytych na *im* to rzeczywiście gwiazdy, natomiast druga opisuje, jaka część gwiazd z *ref* została wykryta na *im*. W praktyce dąży się do optymalizacji obu tych parametrów jednocześnie, jednak jest to zazwyczaj niemożliwe, wobec czego wyznaczana jest wartość najlepiej dostosowana do konkretnego zastosowania. By pomóc w znalezieniu kompromisu pomiędzy tymi statystykami, stosuje się krzywą PR (ang. *Precision-Recall*), która przedstawia precyzję w funkcji czułości

dla różnych ustawień algorytmu detekcyjnego. W przypadku przeprowadzonych eksperymentów omawianym, regulowanym ustawieniem jest wartość progu detekcji p_d.

$$precyzja = \frac{TP}{TP + FP} \tag{5.1}$$

$$czułość = \frac{TP}{TP + FN} \tag{5.2}$$

Na rysunku 5.4. przedstawiono przykładowy przebieg takiej krzywej i zaznaczono punkt, który odpowiada parze wartości (czułość, precyzja) uzyskanych przy zadanej wartości progu p_d. Z racji trudności w precyzyjnej ocenie zmian krzywych PR, powszechną praktyką umożliwiającą porównanie metod jest wyliczanie wartości pola pod krzywą, AUC (ang. *Area Under the Curve*). W przypadku przeprowadzonych badań, im wyższa wartość AUC, tym skuteczniejszy proces detekcji na analizowanych obrazach.



Rysunek 5.4. Przykładowa krzywa PR wraz z zaznaczonym polem pod krzywą AUC.

5.2.3. Zmiana położenia centroidu gwiazdy

Obecność szumu sprawia, że wykrywane na poszczególnych obrazach *im* gwiazdy mogą być oddalone o Δp pikseli od swoich pozycji wyznaczonych na *ref*. W przeprowadzonej analizie zdecydowano się określić maksymalną akceptowalną wartość Δp równą pięciu pikselom, co jest równoważne z minimalną liczbą pikseli pomiędzy wykrywanymi gwiazdami. Jeżeli przesunięcie było większe, detekcja była uznawana za nieprawidłową, ponieważ dochodzi przy niej do znaczącej (wynoszącej co najmniej 3") zmiany pozycji obiektu.

Oprócz samej detekcji zdecydowano się także zestawić ze sobą wartości przesunięć Δp dla wszystkich metod, by ocenić ich działanie również w tym aspekcie. Co istotne, pozycje gwiazd nie są określane przy pomocy indeksów pikseli, lecz oznacza się je jako pozycje centroidów, które są wyznaczane z dokładnością subpikselową, czyli uwzględniającą część ułamkową. Analizowane wartości mają przez to charakter ciągły, a nie dyskretny.

5.2.4. Zmiana jasności gwiazd

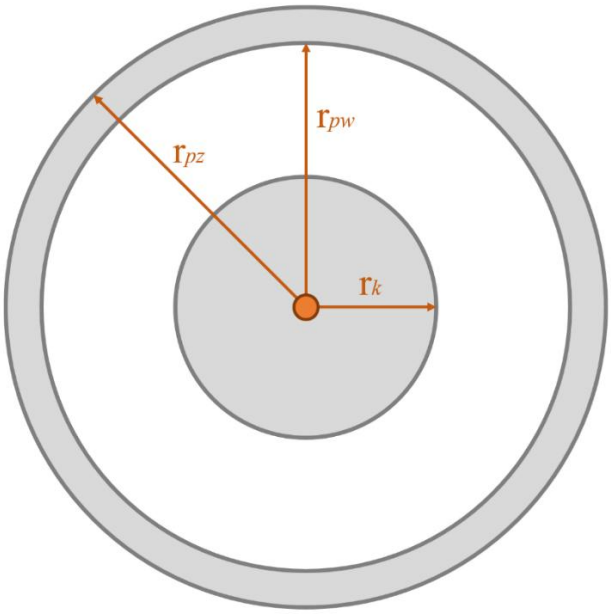
Kolejnym podlegającym ocenie zagadnieniem była zmiana jasności obiektów wykrywanych na *im* względem ich jasności określonej na *ref*. Pomiar tej wielkości zachodzi w opisany poniżej sposób, z wykorzystaniem parametrów ustalanych odgórnie przez osobę dokonującą analizy (na rysunku 5.5 przedstawiono schemat pomocniczy). Proces ten nazywany jest fotometrią aperturową.

1) Dla każdej wykrytej gwiazdy ustala się promień r_k opisanego na niej koła. Wartość ta wynosi zazwyczaj dwukrotność wartości FWHM wykorzystanego w procesie detekcji. W tym wypadku została zatem określona jako 10 pikseli.
2) Ustawiane są również wartości promieni r_{pw} i r_{pz}, które opisują otaczający gwiazdę pierścień pomiaru tła. W przypadku analizowanych obrazów przyjęto $r_{pw} = 20$ pikseli i $r_{pz} = 22$ piksele.
3) Wyznaczana jest mediana wartości pikseli wchodzących w skład pierścienia tła. Wartość ta opisuje typową wartość tła w otoczeniu gwiazdy. Zamiennie do mediany stosuje się czasem wartość średnią pierścienia, ale może być ona obciążona obecnością pobliskich gwiazd.
4) Sumowane są wartości wszystkich pikseli wyznaczonego koła wewnętrznego, a od wyniku odejmowana jest wartość mediany tła pomnożona przez pole koła wyrażone

w pikselach. W efekcie uzyskuje się wartość strumienia światła *f* (ang. *flux*) pochodzącego od badanej gwiazdy.

5) Wartość *f* jest przedstawiana w jednostkach magnitudo, obliczanych zgodnie ze wzorem 5.3. W przypadku porównywania jasności obiektów znajdujących się na danych pochodzących z odmiennych instrumentów, obliczana jest wartość punktu zerowego *ZP* (ang. *zero point*), która określa wymagane przesunięcie dla każdej jasności. W przypadku prowadzonych badań wszystkie dane zostały zebrane przez ten sam instrument, wobec czego pominięto wyznaczanie wartości *ZP* i porównywano jedynie tak zwane magnitudo instrumentalne.

$$mag = -2{,}5 \log_{10}(f) + ZP \tag{5.3}$$



Rysunek 5.5. Wyznaczanie jasności analizowanego obiektu.

5.3. Znane metody przetwarzania obrazów astronomicznych

W ostatnich latach można było zaobserwować duże zainteresowanie zastosowaniem sieci neuronowych w przetwarzaniu obrazów astronomicznych nocnego nieba. Przyczyną tego stanu jest dynamicznie zwiększająca się ilość danych do przetworzenia i związana z nią potrzeba automatyzacji tego procesu. Rozważając stosowane podejścia pod względem ich

zastosowania w detekcji obiektów astronomicznych, można wyodrębnić dwie grupy algorytmów.

Pierwsza z nich obejmuje metody, które, ucząc się odporności na zakłócenia, pomijają etap wstępnego przetwarzania obrazu i dokonują detekcji bezpośrednio na surowych danych [121][122], często łącząc to z klasyfikacją wykrytych obiektów [123][124] lub z pomiarem ich jasności [125]. Taki sposób działania jest niewątpliwie korzystny z punktu widzenia automatyzacji, jednak wiążą się z nim pewne trudności. Podstawową z nich jest złożoność implementacyjna – do poprawnego treningu wyniki działania sieci muszą być porównywane z rzeczywistymi pozycjami obiektów, co wymaga przeprowadzenia czasochłonnego procesu oznaczania danych i weryfikacji jego poprawności. Nadrzędnym problemem jest jednak konieczność posiadania danych z instrumentów co najmniej o klasę lepszych (w kontekście precyzji pomiarowej), najlepiej pracujących w przestrzeni kosmicznej i rejestrujących światło w tym samym zakresie spektralnym. Niestety, w takim przypadku pozycje gwiazd mogą się różnić od tych rejestrowanych z poziomu Ziemi, głównie z powodu braku obecności refrakcji atmosferycznej.

Alternatywą okazują się być rozwiązania z drugiej grupy, które skupiają się wyłącznie na przeprowadzeniu pojedynczego etapu przetwarzania lub analizy danych (na przykład na samej fotometrii [126]). Działanie takich metod łatwiej ocenić i zinterpretować, zwłaszcza wizualnie, a ich trening nie wymaga tak skomplikowanego przygotowania danych. Dodatkowo zwiększają one modularność całego procesu, bo można zamiennie używać algorytmów skupiających się wyłącznie na jednym aspekcie. Z tego powodu uwagę zwrócono głównie na te sieci, które zajmują się samą redukcją szumu.

Spośród spotykanych rozwiązań najwięcej wymienić można tych bazujących na architekturze U-Net [127]-[132], ale wykorzystywano również autoenkodery [129][133][134] oraz inne sieci złożone głównie z warstw konwolucyjnych [135][136], w tym jedną sieć złożoną z warstw gęstych [137]. Większość z wymienionych rozwiązań była testowana na danych pochodzących z teleskopów wysokorozdzielczych i/lub danych syntetycznych. Zasadne jest pytanie, czy inne, mniejsze, sieci nie okażą się lepsze w przypadku obciążonych większym szumem obrazów niskorozdzielczych uzyskiwanych w małych obserwatoriach. By to ocenić, wybrano sieć Astro U-Net [127] jako punkt odniesienia, a jej pracę porównano z mniej złożonymi modelami opisanymi w kolejnym podrozdziale.

5.4. Testowane architektury

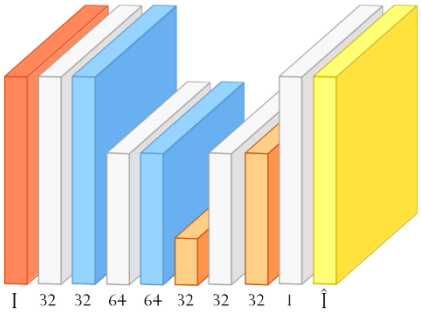
Do badań wykorzystano łącznie 4 różne struktury sieci: dwa autoenkodery oraz dwie sieci typu U-Net. Autoenkodery rozróżniono względem rozmiaru jako Płytki AE (rysunek 5.6) oraz nieco bardziej złożony Głęboki AE (rysunek 5.7). Na bazie drugiego z autoenkoderów stworzono model nazwany Prostym U-Netem (rysunek 5.8) – różnica pomiędzy nim a siecią bazową polega głównie na dodaniu operacji, które odpowiadają za obsługę połączeń pomijających. Ostatnim z modeli jest Astro U-Net (rysunek 5.9), który posłużył do oceny prostszych rozwiązań jako sprawdzona doświadczalnie sieć [127].

Strukturę sieci ponownie przedstawiono przy pomocy schematów blokowych. Każdy z użytych kolorów oznacza inny typ warstwy, jak wyszczególniono poniżej:

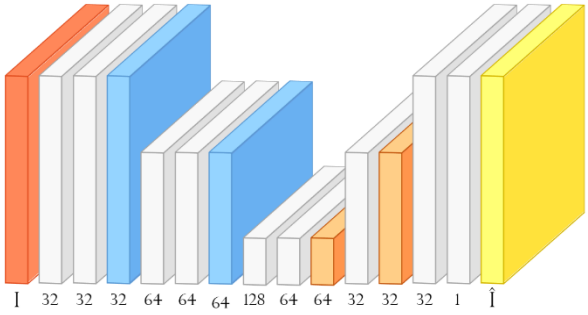
a. blok czerwony oznacza obraz wejściowy sieci I;
b. blok żółty symbolizuje obraz wyjściowy sieci $\hat{I}$;
c. blok szary symbolizuje jednostkę funkcjonalną złożoną z warstwy uzupełnień zerami, warstwy konwolucyjnej o jądrze 3×3 i kroku równym 1 oraz warstwy aktywacji wykorzystującej funkcję LeakyReLU (wyjątkiem są ostatnie bloki tego typu w Płytkim AE, Głębokim AE i Prostym U-Necie, w których funkcją aktywacji, ostatnią w sieci, jest funkcja sigmoidalna);
d. blok niebieski oznacza maksymalizującą warstwę łączącą o jądrze 3×3 i kroku równym 1;
e. blok pomarańczowy odpowiada jednostce funkcjonalnej złożonej z warstwy dekonwolucyjnej o jądrze 2×2 i kroku równym 2 oraz warstwy aktywacji w postaci LeakyReLU;
f. blok fioletowy oznacza operację połączenia ze sobą dwóch tensorów według wymiaru map cech – łączone są ze sobą wyjście warstwy poprzedzającej ten blok i wyjście ostatniego szarego bloku znajdującego się na tym samym „poziomie” sieci (dla ułatwienia analizy jest on oznaczony fioletowym konturem i deseniem szachownicy na jednym boku);
g. obecny jedynie w Astro U-Necie biały blok z czarnym konturem odpowiada pojedynczej warstwie konwolucyjnej o jądrze 3×3 i kroku równym 1 bez zastosowania funkcji aktywacji.

Do treningu wykorzystano funkcję straty Hubera z wartością progu δ równą 1, a za optymalizator przyjęto podstawową wersję algorytmu Adam. Modele trenowano przez 2 000 epok, przy czym każda epoka składała się ze 128 iteracji złożonych z grup zawierających po 64 obrazy, co dało w sumie 8 192 próbek przetwarzanych w każdej epoce. Łącznie trening objął około 16,4 miliona obrazów. Do uczenia wykorzystano harmonogram, który przez pierwsze 200 epok treningu „rozgrzewał" sieć do zadanego współczynnika uczenia, a następnie zmniejszał jego wartość do zera, wykorzystując metodę wygaszania kosinusowego. Wartość tego współczynnika wynosiła 0,001 dla wszystkich sieci z wyjątkiem sieci Astro U-Net, dla której musiała ona zostać stukrotnie zmniejszona, by zapewnić stabilność treningu.

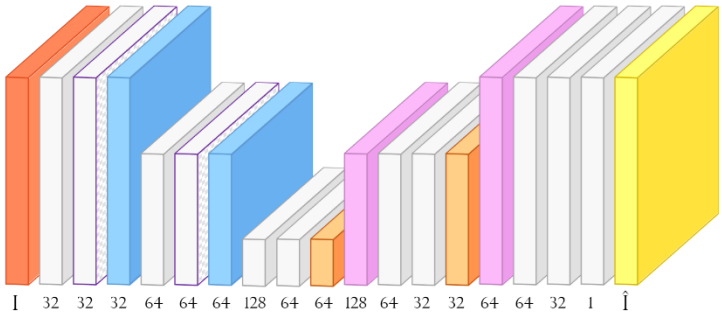
Każda z sieci została wytrenowana z wykorzystaniem dwóch odmiennych podejść, Noise2Noise (N2N) i Noise2Clean (N2C). Po zakończeniu tego procesu zweryfikowano, że wszystkie modele przestały poprawiać swoją wydajność pomiędzy 1500 a 1700 epoką, gdy funkcje straty osiągnęły swoją najniższą wartość.



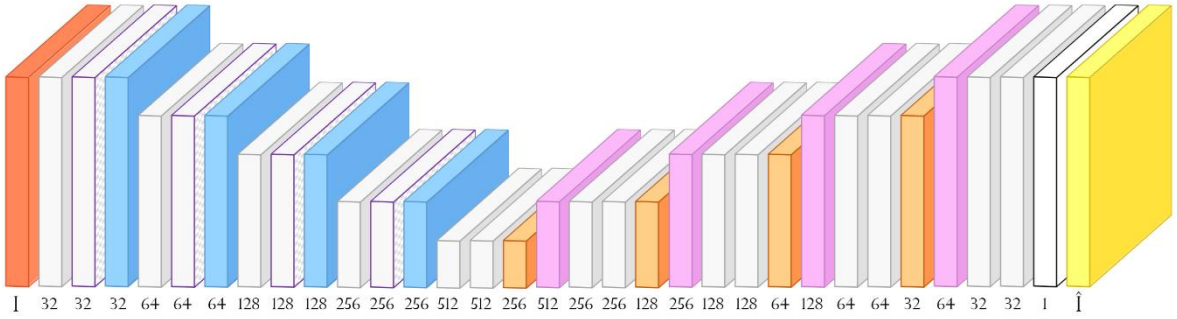
Rysunek 5.6. Płytki AE – schemat blokowy.



Rysunek 5.7. Głęboki AE – schemat blokowy.



Rysunek 5.8. Prosty U-Net – schemat blokowy.



Rysunek 5.9. Astro U-Net – schemat blokowy.

5.5. Opis i analiza wyników

Wyniki przeprowadzonych eksperymentów zostały uporządkowane w sześciu sekcjach. Pierwsza z nich poświęcona jest analizie zależności między rozmiarem przestrzennym danych wejściowych a wielkością modelu oraz związanym z tym zapotrzebowaniem na zasoby obliczeniowe. Druga część przedstawia porównanie wartości wybranych miar jakości obrazu: PSNR i FSIM. Trzecia sekcja obejmuje ocenę wpływu zastosowanych metod na skuteczność detekcji gwiazd, natomiast w czwartej przeanalizowano ich wpływ na rejestrowaną jasność wykrytych obiektów. Piąta część poświęcona jest analizie zmian położenia gwiazd spowodowanych działaniem poszczególnych metod, a ostatnia zawiera ocenę wizualną obrazów wynikowych.

5.5.1. Wpływ wymiarowości danych na rozmiar sieci

Wykorzystane w eksperymentach sieci są modelami w pełni konwolucyjnymi, dzięki czemu mogą przyjąć na wejściu obrazy o niemal dowolnym rozmiarze, jak wyjaśniono w poprzednim rozdziale. W tabeli 5.1 przedstawiono liczbę parametrów sieci i szacowany rozmiar pamięci potrzebny do przetworzenia przez nie jednego obrazu o wymiarach 128×128 pikseli. Jak można zauważyć, wraz ze stopniem złożoności strukturalnej sieci rośnie liczba ich parametrów oraz wymagany rozmiar pamięci. Co istotne, ta druga wartość nie jest liniowo zależna od liczby parametrów, co można zobaczyć porównując ze sobą Płytki AE i Głęboki AE – ponad pięciokrotnie większa liczba parametrów skutkuje jedynie minimalnie większym zapotrzebowaniem na pamięć operacyjną. Podobną relację opisuje zestawienie Prostego U-Neta z Astro U-Netem.

Tabela 5.1. Złożoność sieci przy wykorzystaniu obrazów 128x128.

Typ sieci	**Liczba parametrów sieci**	**Zapotrzebowanie sieci na pamięć operacyjną [MB]**
Płytki AE	46 817	7,15
Głęboki AE	261 217	7,47
Prosty U-Net	353 473	22,74
Astro U-Net	7 759 521	55,93

W tabeli 5.2 przedstawione są analogiczne wartości dla przetwarzanych obrazów o rozmiarze 1024×1024 pikseli. Przy porównaniu tych tabel najważniejszą obserwacją jest to, że liczba parametrów nie ulega żadnej zmianie. Wynika z tego, że sieci można początkowo wyszkolić na niewielkich obrazach, a po treningu użyć ich z powodzeniem na większych. Podejście to znajduje powszechne zastosowanie w praktyce.

Kolejnym wartym podkreślenia wnioskiem jest to, że zapotrzebowanie sieci na pamięć nie musi rosnąć w takiej same skali jak wymiarowość danych. Przykładowo, 64-krotne zwiększenie rozmiaru obrazu (ze 128×128 na 1024×1024 piksele) skutkuje jedynie około 30-krotnie większym zapotrzebowaniem Astro U-Neta na zasoby sprzętowe.

Tabela 5.2. Złożoność sieci przy wykorzystaniu obrazów 1024x1024.

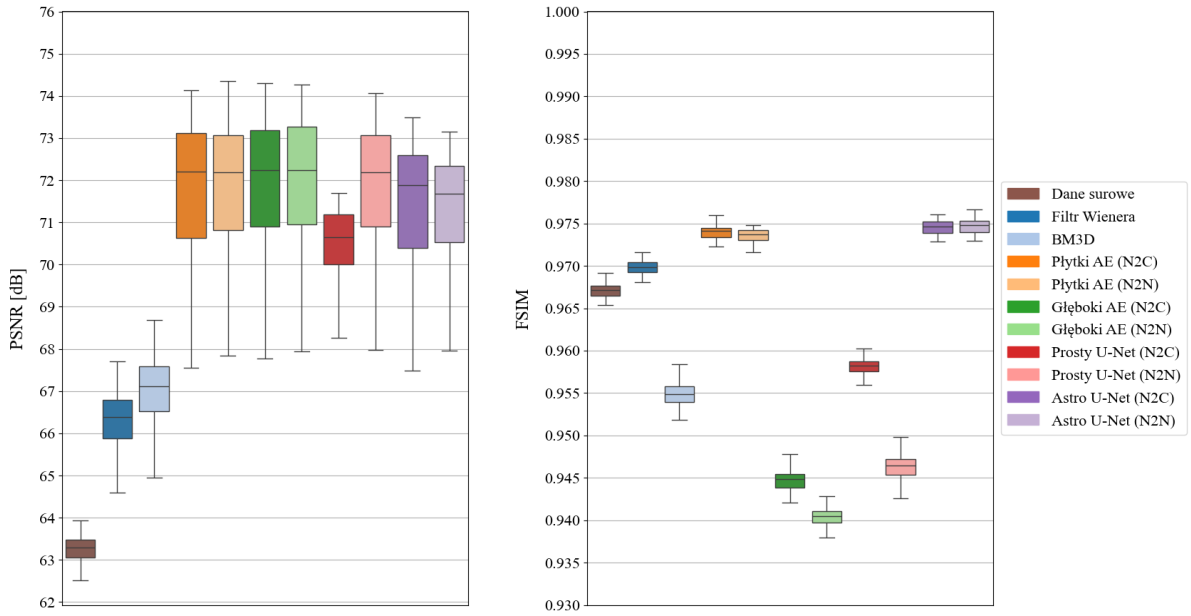
Typ sieci	**Liczba parametrów sieci**	**Zapotrzebowanie sieci na pamięć operacyjną [MB]**
Płytki AE	46 817	411,50
Głęboki AE	261 217	442,68
Prosty U-Net	353 473	1 357,35
Astro U-Net	7 759 521	1 701,60

5.5.2. Porównanie wartości metryk PSNR i FSIM

Każdą z klatek zbioru testowego przetworzono badanymi metodami, a wyniki porównano z klatką uśrednioną. Na rysunku 5.10 przedstawiono wykresy pudełkowe, które opisują wartości metryk PSNR (po lewej) i FSIM (po prawej). Rozpatrując wyniki PSNR, można zauważyć, że zastosowanie dowolnej metody bezsprzecznie poprawiło jakość obrazu. Co najbardziej interesujące, wszystkie sieci, bez wyjątku, okazały się lepsze od testowanych algorytmów deterministycznych. Należy jednak pamiętać, że PSNR jest mocno zależny od większościowej w obrazach liczby piksel tła. Zatem dowolny algorytm redukujący poziom szumu w pikselach tła, będzie skutkował ostatecznie istotnym wzrostem wartości tej metryki.

W przypadku wskaźnika FSIM można mówić o znacznie większym rozrzucie wyników. Jedynie zastosowanie filtru Wienera oraz Płytkiego AE i Astro U-Netu skutkuje nieznaczną poprawą otrzymywanych wyników. Dla pozostałych rozwiązań widoczny jest natomiast spadek wartości tej metryki.

Analizując omawiane wykresy, należy jednak wziąć pod uwagę, że wszystkie uzyskane wartości są relatywnie wysokie, co może wynikać z dużego podobieństwa danych surowych do danych porównawczych, a nie ze sposobu przetwarzania danych. W tej sytuacji stosowanie takich metod analizy jest niewystarczające do pełnej oceny testowanych metod, a nawet może prowadzić do błędnych ocen. Wymagana jest zatem analiza efektów przetwarzania danych w postaci pomiarów astrometrycznych i fotometrycznych poprzedzonych detekcją źródeł.

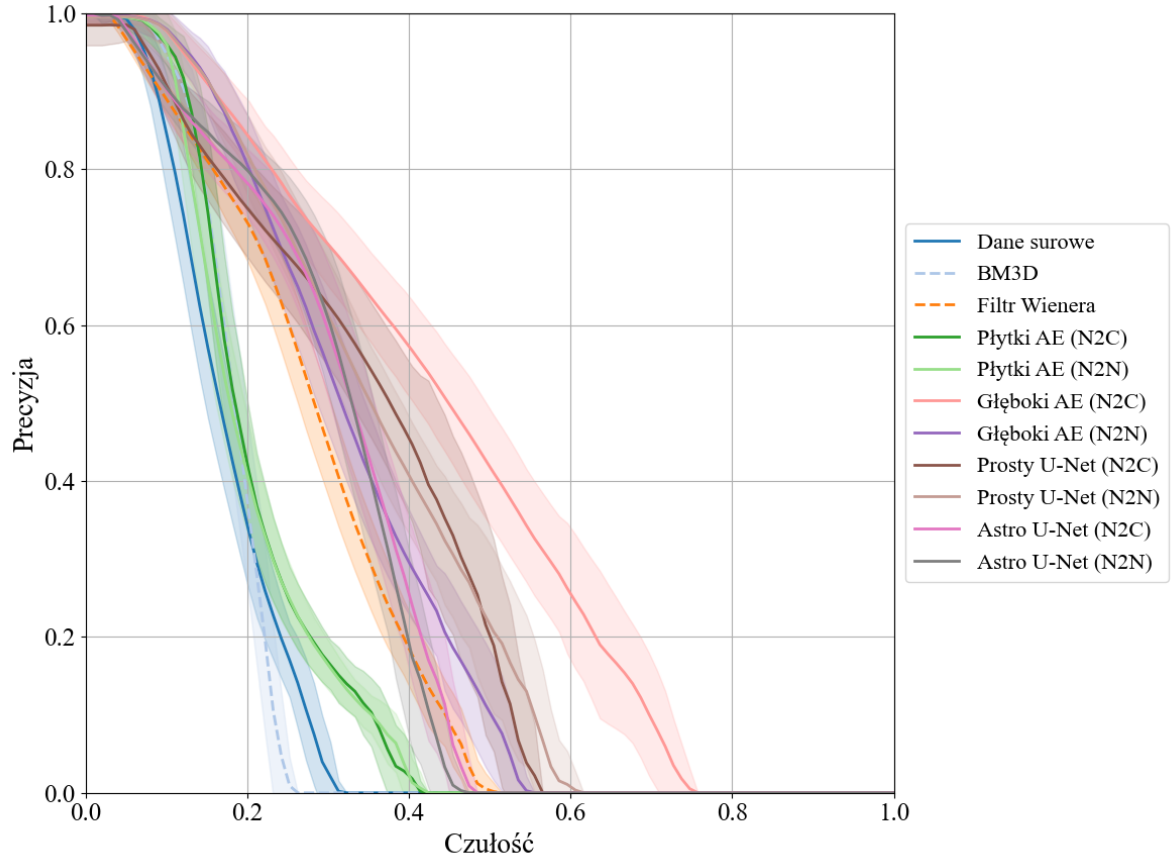


Rysunek 5.10. Wartości PSNR i FSIM dla badanych obrazów.

5.5.3. Wpływ testowanych rozwiązań na detekcję obiektów

Każda z 50 klatek serii testowej została przetworzona z wykorzystaniem wszystkich testowanych metod. Następnie na uzyskanych obrazach wynikowych przeprowadzono detekcję gwiazd, modyfikując wartość progu detekcji p_d w celu wyznaczenia par punktów (czułość, precyzja) niezbędnych do skonstruowania krzywych PR. Wyniki tej procedury przedstawiono na rysunku 5.11. Każdej z metod przypisano krzywą w odrębnym kolorze, reprezentującą uśrednioną skuteczność detekcji na każdej z analizowanych klatek. Obszar otaczający każdą z nich, oznaczony zredukowaną przezroczystością i odpowiadającym zabarwieniem, ilustruje rozrzut wyników (krzywych PR) równy jednemu odchyleniu standardowemu.

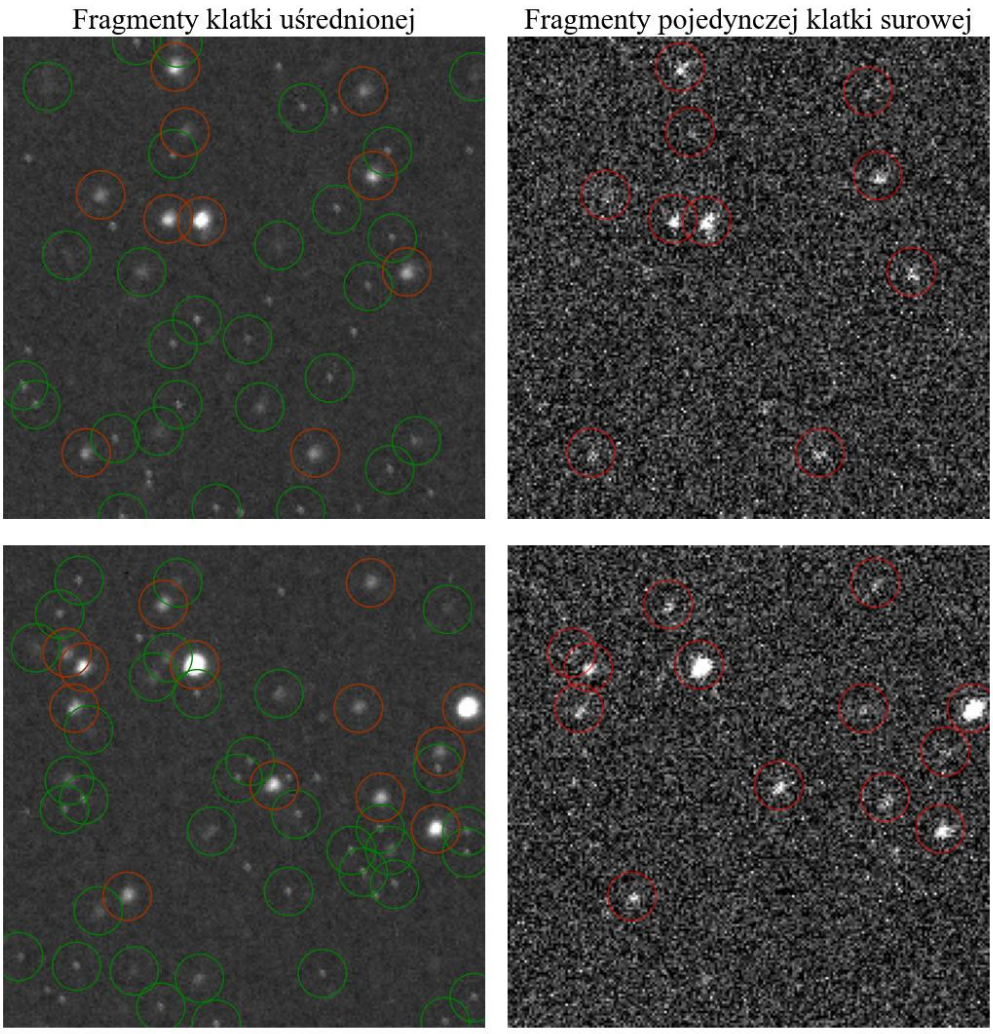
Jak można zauważyć, przebieg przedstawionych krzywych zdecydowanie różni się od przykładu z rysunku 5.4. W żadnym z analizowanych przypadków nie doszło do detekcji wszystkich gwiazd wyznaczonych na obrazie referencyjnym, natomiast krzywe cechują się silną tendencją spadkową. Jest to efektem wyraźnych różnic pomiędzy pojedynczą klatką serii a klatką uśrednioną – dla obrazów surowych czułość osiąga najwyższą wartość wynoszącą w przybliżeniu zaledwie 0,3, co potwierdza znaczące obciążenie szumem pojedynczych klatek i wynikający z tego brak widoczności najciemniejszych obiektów.



Rysunek 5.11. Krzywa PR dla wyników detekcji porównanych z pełnym zbiorem.

By lepiej ocenić efektywność detekcji obiektów, które mogły być wykryte w obrazach odszumionych, zdecydowano się nieco ograniczyć liczbę gwiazd. Obrazy surowe poddano analizie wizualnej w odniesieniu do obrazu referencyjnego i spośród pierwotnych detekcji wybrano jedynie te, które były możliwe do dostrzeżenia w pojedynczej, zaszumionej klatce. Na rysunku 5.12 przedstawiono ograniczenie tego zbioru dla dwóch przykładowych wycinków o wymiarach 200×200. Zielonymi okręgami zaznaczono detekcje, które odrzucono z dalszej analizy, a czerwonymi pozostałe gwiazdy przydzielone do ograniczonego zbioru.

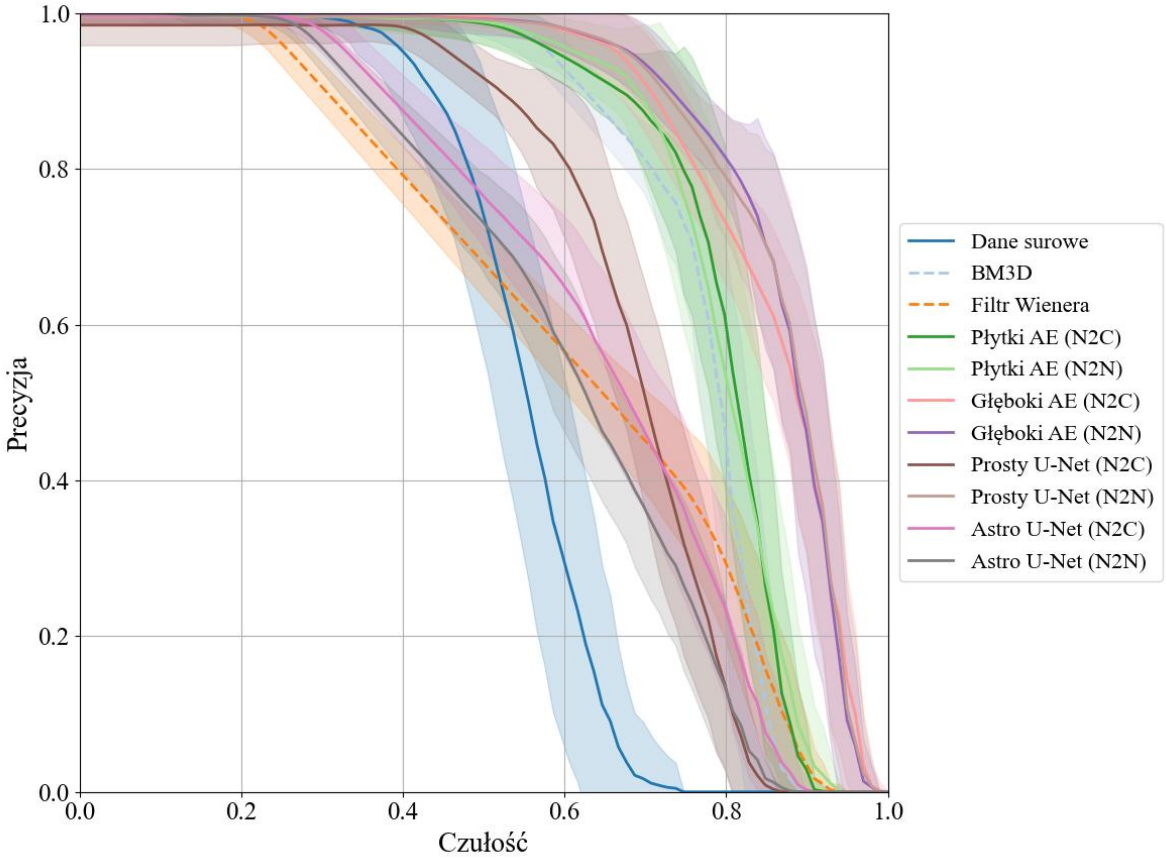
Po wprowadzeniu tej modyfikacji ponownie przeprowadzono detekcję, a następnie wyznaczono zaktualizowane krzywe, które przedstawiono na rysunku 5.13. Nie występuje już dla nich tak wyraźna tendencja spadkowa, a ich kształt jest bardziej zgodny z oczekiwanym przebiegiem krzywej PR. W celu pogłębienia analizy dla obydwu przypadków detekcji obliczono pole pod krzywymi, a wyniki zaprezentowano w tabeli 5.3. Porównanie wartości tabelarycznych z profilami krzywych PR z rysunków 5.11 i 5.13 pozwoliło sformułować kilka istotnych wniosków.



Rysunek 5.12. Ograniczenie zbioru detekcji na przykładzie dwóch fragmentów analizowanych klatek: referencyjnej (po lewej) i surowej (po prawej). Na zielono zaznaczono gwiazdy wykluczone z analizy.

Przede wszystkim działanie każdej z sieci neuronowych pozytywnie wpłynęło na możliwości detekcji gwiazd. Wykorzystanie metod klasycznych również zaowocowało poprawą, chociaż BM3D uzyskał niespodziewanie niski wynik dla zbioru wszystkich detekcji, powodując pogorszenie efektów detekcji względem procesu detekcji wykonanym na nieprzetworzonych obrazach. Uwzględniając relatywnie dobry wynik dla ograniczonego zbioru, można wnioskować, że algorytm ten prawdopodobnie nie jest w stanie dobrze oddać charakteru słabiej świecących gwiazd lub są one mocno tłumione. Pod tym względem lepszy jest filtr Wienera, jednak analizując odpowiadającą mu krzywą na rysunku 5.13, można zauważyć, że wzrost czułości detektora na przetworzonych przez niego obrazach jest mocno

skorelowany ze spadkiem precyzji. Oznacza to, że filtr ten dopasowuje do profilu gwiazd także losowy szum, co w pewnym stopniu niweluje jego użyteczność.



Rysunek 5.13. Krzywa PR dla wyników detekcji porównanych z ograniczonym zbiorem.

W przypadku badanych sieci neuronowych można wyznaczyć zależność pomiędzy ich złożonością a osiąganymi rezultatami. Najmniej złożona sieć, Płytki AE (kolor zielony), jest w stanie osiągnąć dobre wyniki na ograniczonym zbiorze, ale w przypadku pełnego zbioru różnica względem danych surowych jest niewielka. Sugeruje to, że do przetworzenia ciemniejszych gwiazd wymagane są bardziej złożone sieci, co potwierdzają zauważalnie lepsze wyniki Głębokiego AE. Co interesujące, użycie większych, bardziej złożonych sieci typu U-Net nie skutkuje istotną poprawą względem prostszego Głębokiego AE. Oznacza to, że dla analizowanych obrazów niskorozdzielczych nie są potrzebne duże i skomplikowane modele, by móc osiągnąć bardzo dobry wynik.

Tabela 5.3. Wartości pola pod krzywą. Pogrubieniem zaznaczono najlepsze wyniki w obu przypadkach, a podkreśleniem drugie najlepsze wartości.

Sposób przetwarzania	**Pole pod krzywą**	
	Przy pełnym zbiorze detekcji	**Przy ograniczonym zbiorze detekcji**
Dane surowe	0,1734	0,5490
BM3D	0,1806	0,7664
Filtr Wienera	0,2767	0,6214
Płytki AE (N2C)	0,2087	0,7907
Płytki AE (N2N)	0,2033	0,7913
Głęboki AE (N2C)	**0,4349**	0,8538
Głęboki AE (N2N)	0,3217	**0,8644**
Prosty U-Net (N2C)	0,3420	0,6785
Prosty U-Net (N2N)	<u>0,3551</u>	<u>0,8635</u>
Astro U-Net (N2C)	0,3040	0,6449
Astro U-Net (N2N)	0,3004	0,6083

5.5.4. Analiza wpływu rozważanych metod na jasność gwiazd

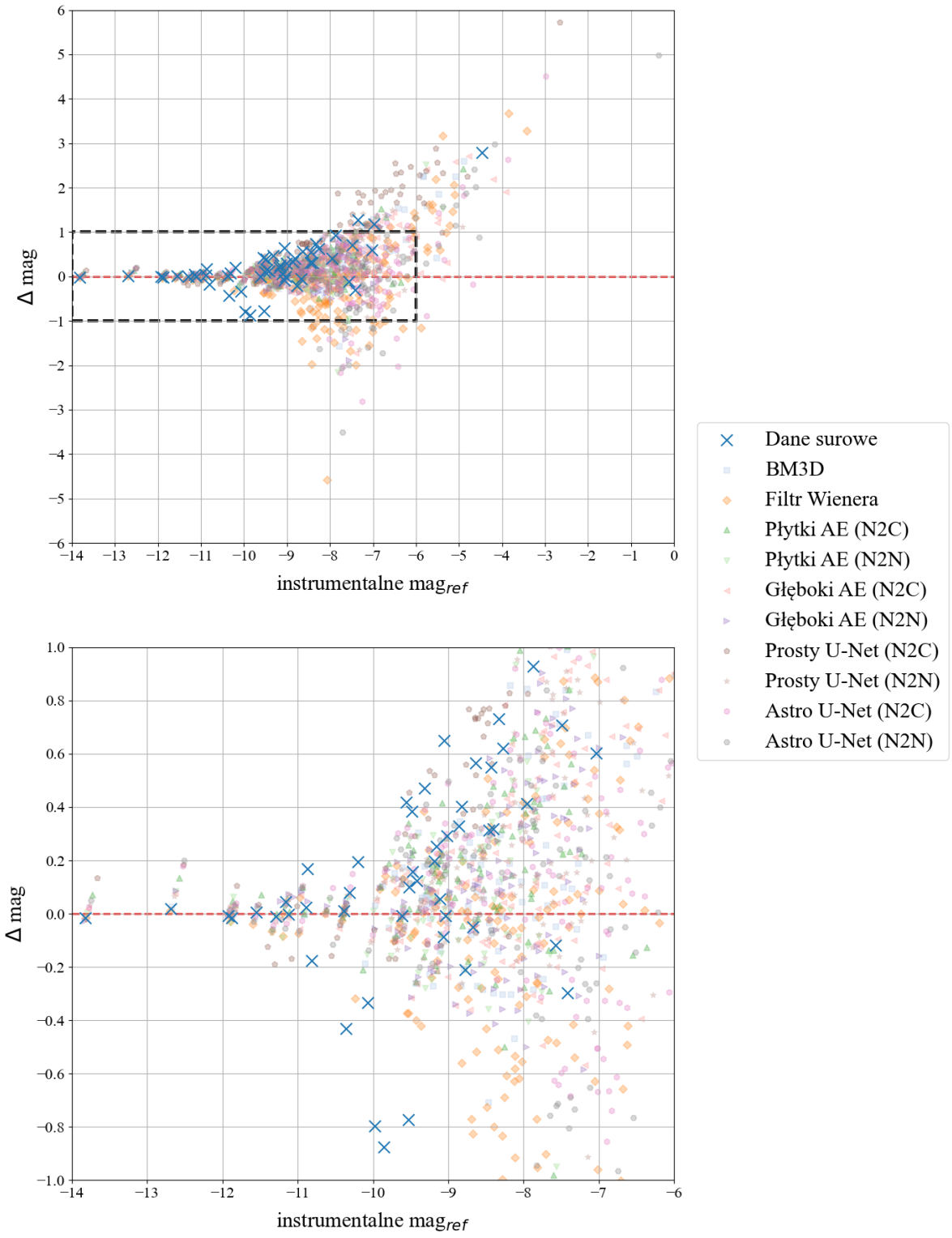
Dla każdej wykrytej gwiazdy wykonano pomiar jasności, by ocenić jak duże odchylenia tych wartości wprowadzają testowane techniki. W górnej części rysunku 5.14 przedstawiono zmiany jasności wykrytych gwiazd względem jasności odpowiadającym im gwiazd na obrazie referencyjnym. Wartości dodatnie świadczą o przyciemnieniu obiektu, natomiast ujemne o jego pojaśnieniu. Dla zwiększenia czytelności wyników przedstawiono je jedynie dla analizy jednej, losowo wybranej klatki.

W trakcie analizy można zauważyć, że dla najjaśniejszych gwiazd różnice dla klatki surowej są bardzo małe i rosną wraz ze spadkiem jasności gwiazd porównawczych. Zjawisko to ma uzasadnienie fizyczne – w przypadku gwiazd, których sygnał w niewielkim stopniu przekracza poziom tła, pojawiający się szum może mocno zaburzyć wynik pomiaru, co nie ma aż takiego znaczenia przy jaśniejszych gwiazdach. Warto także zwrócić uwagę na strukturę „skrzydła" pojawiającą się dla wartości dodatnich. Z jej obecności wynika, że metody mają

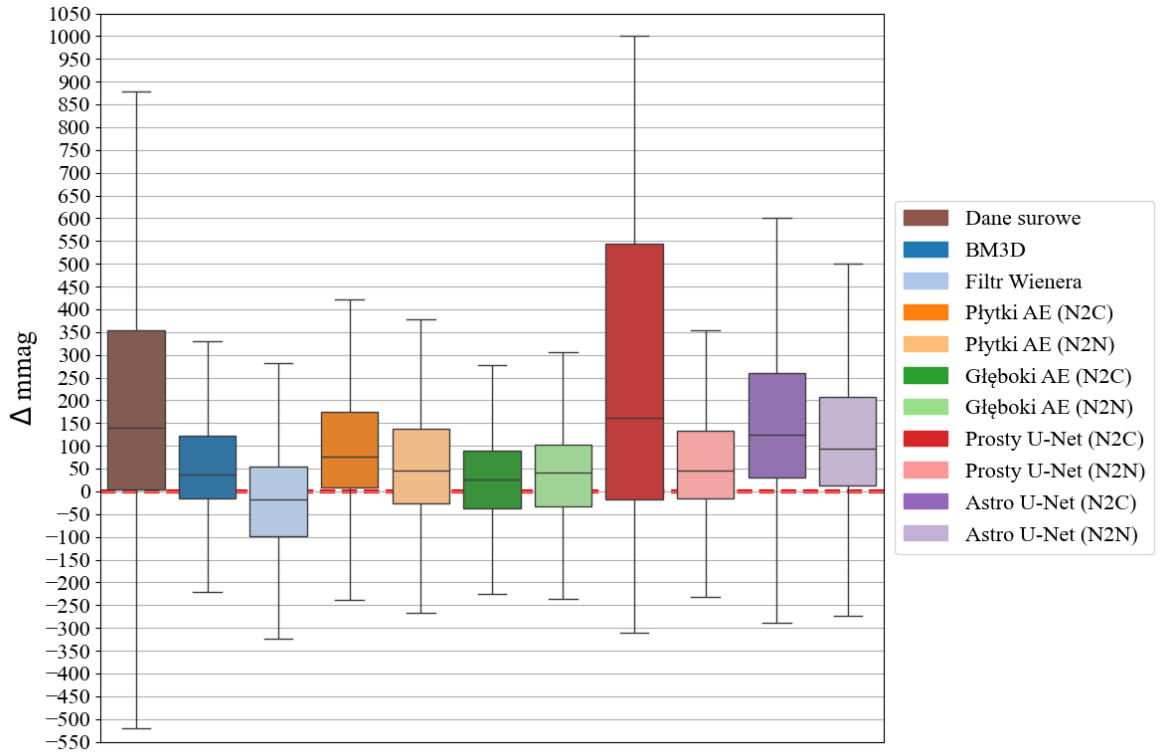
tendencję do tłumienia słabych sygnałów. Jest to o tyle interesująca kwestia, że pomimo tego efektu detektor jest nadal w stanie poprawnie zakwalifikować przytłumiony sygnał jako gwiazdę.

Dolna część rysunku 5.14 prezentuje przybliżenie na zaznaczony czarnym prostokątem fragment całego wykresu. Pozwala to dostrzec, że stosowane metody zarówno zmniejszają, jak i zwiększają jasność obiektów. By zgłębić to zagadnienie, zagregowano dane uzyskane dla wszystkich klatek testowych i przedstawiono je w formie wykresów pudełkowych na rysunkach 5.15 i 5.16. Rysunek 5.15 obrazuje zmiany jasności gwiazd na obrazach przetworzonych względem obrazu referencyjnego. Poza wynikami uzyskanymi z użyciem filtra Wienera, wszystkie metody cechują się przyciemnieniem analizowanych gwiazd, natomiast wyniki dla Prostego U-Neta (N2C) jako jedyne obejmują większy zakres przyciemnienia.

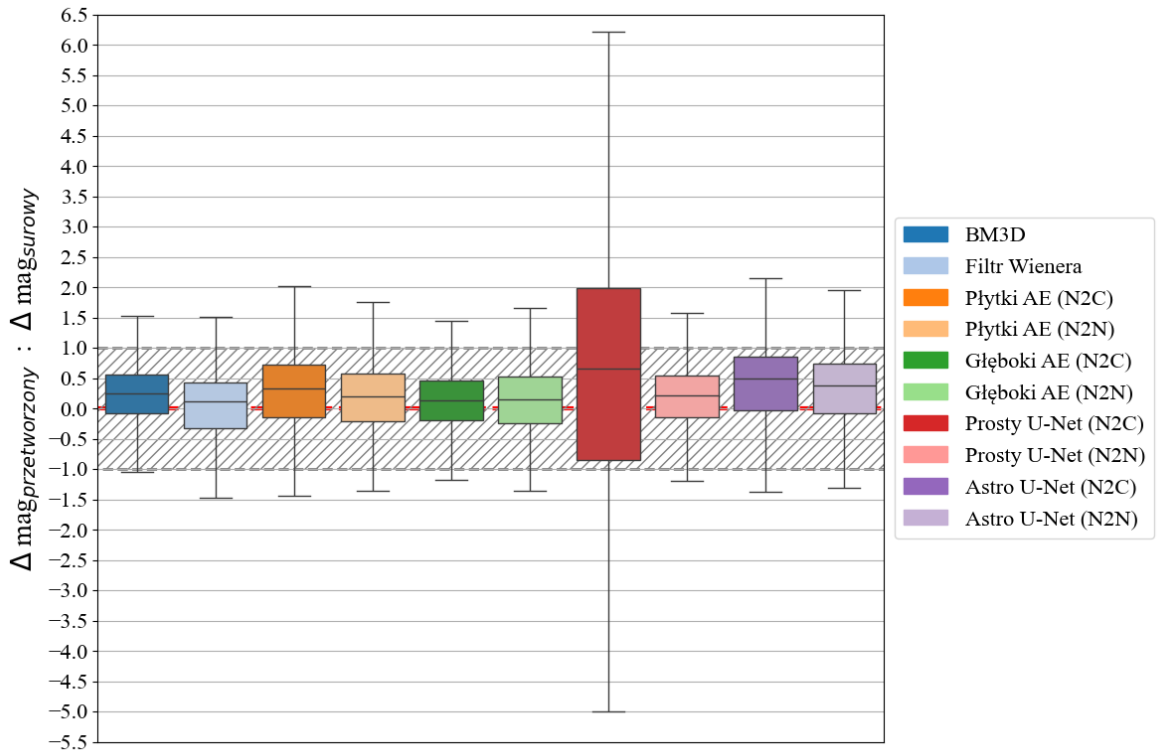
Na rysunku 5.16 przedstawiono odchyłki jasności gwiazd wykrytych na obrazach przetworzonych w stosunku do tychże odchyłek na obrazach surowych. Czerwoną linią oznaczono wartość 0, która świadczy o tym, że nie występują żadne różnice w porównaniu z obrazem referencyjnym, a wyniki znajdujące się w zaznaczonym na szaro obszarze od -1 do 1 odpowiadają zmniejszeniu się wartości bezwzględnej różnicy jasności. Jak można zauważyć, dla niemal wszystkich metod zachodzi istotne zmniejszenie błędu fotometrycznego, a jedynym wyjątkiem jest Prosty U-Net (N2C), którego wyniki prowadzą do błędów wykraczających poza błędy pojawiające się w fotometrii klatek surowych.



Rysunek 5.14. Zmiany jasności wykrywanych gwiazd na przykładowej klatce serii. Na górze porównanie wszystkich wyników, a poniżej zbliżenie na najbardziej interesujący fragment.



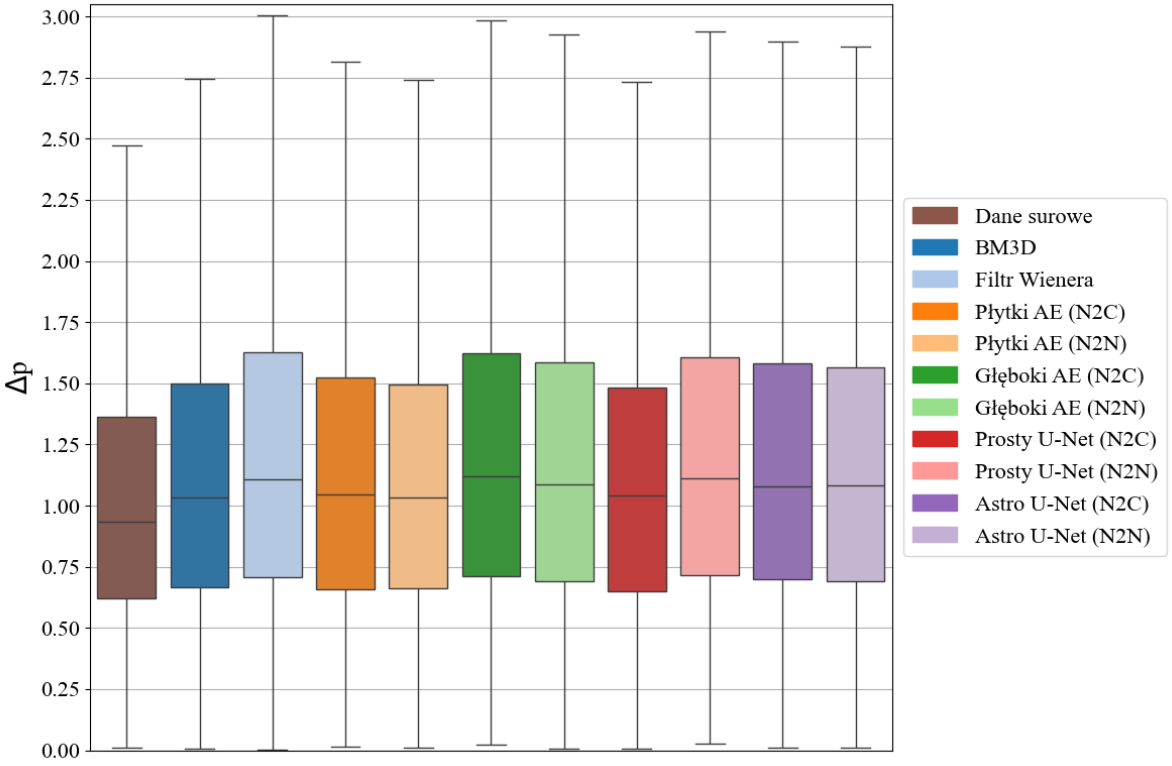
Rysunek 5.15. Różnice jasności wykrytych gwiazd względem odpowiadających im jasności na obrazie referencyjnym.



Rysunek 5.16. Stosunek zmian jasności gwiazd wykrytych na obrazach przetworzonych względem zmian jasności odpowiadających im gwiazd na obrazach surowych.

5.5.5. Porównanie zmian położenia gwiazd

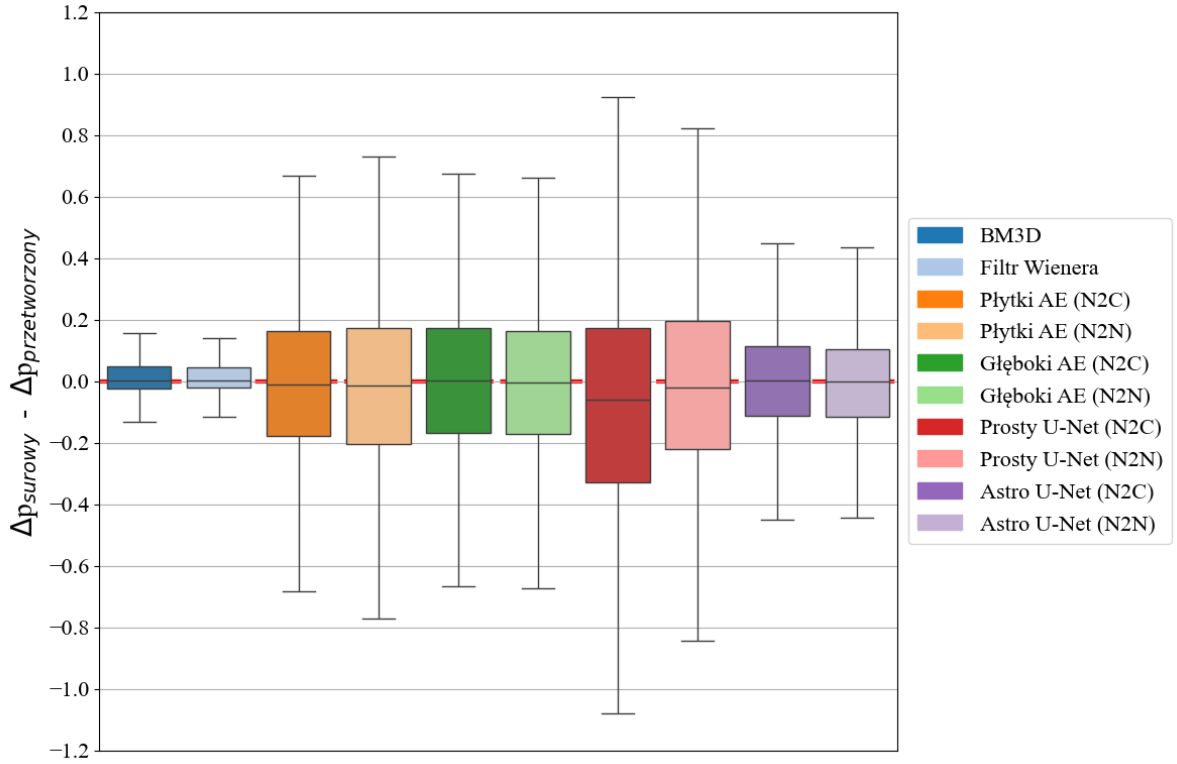
Analogicznej analizie poddano zmianę pozycji wykrywanych gwiazd. Na rysunku 5.17 przedstawiono wartości przesunięć względem pozycji gwiazd na obrazie referencyjnym (przesunięcia wyrażono w pikselach). Jak można zaobserwować, są one statystycznie nieco większe niż w przypadku obrazów surowych, lecz nigdy nie wynoszą więcej niż 3 piksele, podczas gdy dla obrazów surowych jest to maksymalnie około 2,5 piksela.



Rysunek 5.17. Różnice położenia wykrytych gwiazd względem gwiazd na obrazie referencyjnym.

W kolejnym kroku dokonano weryfikacji zmian położenia centroidów gwiazd na obrazach przetworzonych względem pozycji na obrazach surowych. W ten sposób sprawdzono, jak każda z metod zniekształca dane wejściowe, a wyniki przedstawiono na rysunku 5.18. Wynika z niego, że zmiany zachodzą symetrycznie zarówno w kierunku położenia gwiazd na *ref*, jak i w kierunku przeciwnym. Najwyższą jakość zachowują przy tym klasyczne algorytmy, BM3D oraz filtr Wienera, które nie wnoszą aż tak dużych przesunięć jak sieci neuronowe. Reasumując, najważniejszym faktem jest jednak to, iż obserwowane odchyłki położeń są na

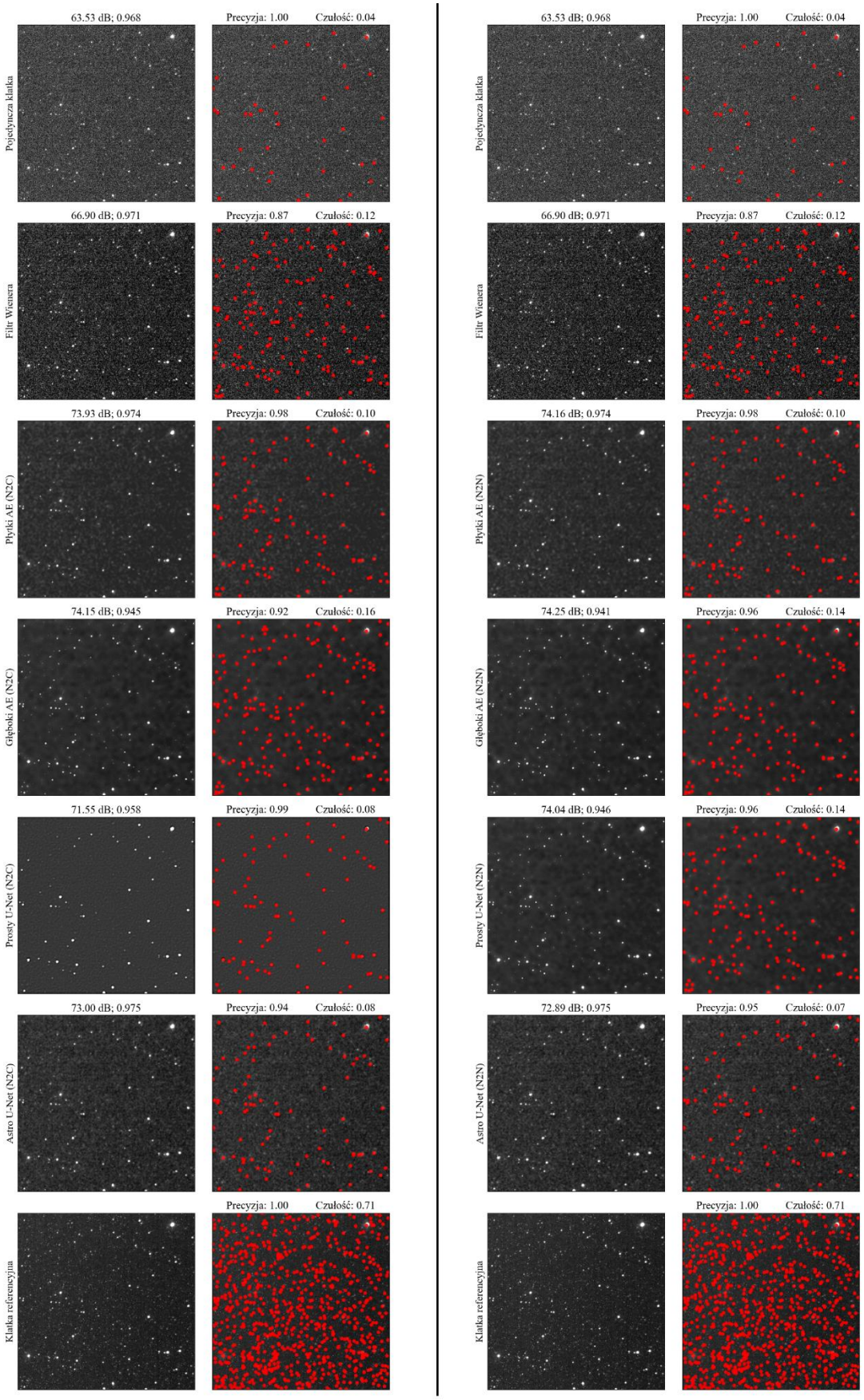
poziomie subpikselowym, zazwyczaj znacznie mniejszym niż 0,2 piksela, co świadczy o praktycznie niezauważalnych problemach astrometrii na zdjęciach przetworzonych.



Rysunek 5.18. Zmiana położenia wykrytych gwiazd względem odpowiadających im gwiazd na obrazie surowym.

5.5.6. Ocena wizualna wyników

Na koniec dokonano oceny wizualnej działania testowanych metod. Oceniono przy tym tak jakość obrazów końcowych, jak i poprawność detekcji przeprowadzonej dla progu $p_d = 7$. Wybór takiej wartości tego parametru pozwolił na ocenę rozwiązań z uwzględnieniem jedynie najpewniejszych, najbardziej rygorystycznych detekcji. Wyniki tej analizy przedstawiono na rysunku 5.19.

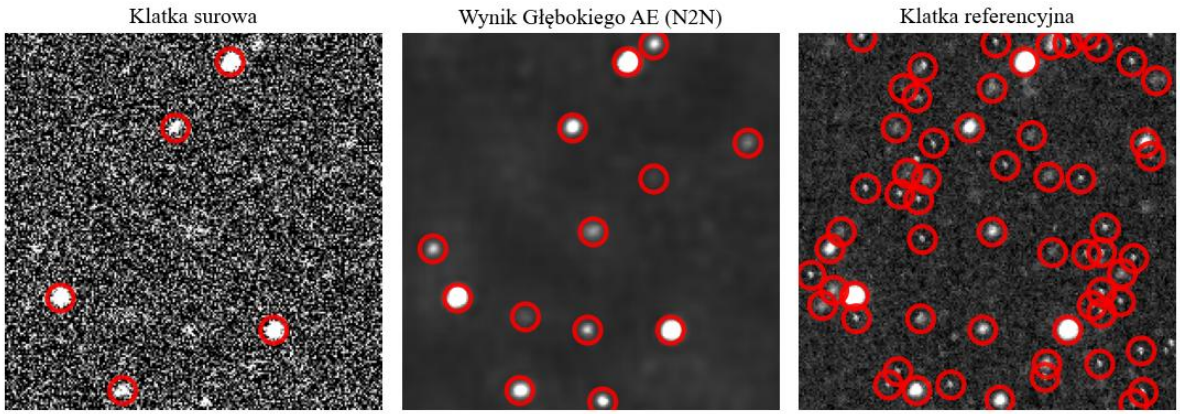


Rysunek 5.19. Porównanie wizualne wpływu zastosowania filtra Wienera i sieci neuronowych na detekcję gwiazd ($p_d = 7\sigma$). Po lewej przedstawiono wyniki dla trenowania sieci strategią N2C, a po prawej N2N.

Rysunek 5.19 podzielony jest na 2 dwie części – w lewej umieszczono wyniki trenowania sieci techniką N2C, a w prawej techniką N2N. Z algorytmów klasycznych porównano jedynie filtr Wienera, który uzyskiwał średnio lepsze wyniki niż BM3D. Obie części rysunku zostały podzielone na dwie kolumny – w lewej przedstawiono rozważane obrazy z obliczonymi wskaźnikami PRNR i FSIM, a w prawej zaznaczono detekcje i wyznaczono wartości precyzji i czułości.

Po przetworzeniu obrazu surowego filtrem Wienera wciąż pozostaje w nim obecny łatwo dostrzegalny szum. W przypadku działania sieci szum ten jest zdecydowanie bardziej przytłumiony, a wartość metryk jest wyższa, przy czym po raz kolejny występuje wyjątek w postaci Prostego U-Neta (N2C), który silnie wyrównuje sygnał tła. Porównując wyniki działania sieci z klatką surową, można zauważyć, że zastosowanie dowolnej metody skutkuje spadkiem precyzji, ale wiąże się z większą czułością. Zgodnie z wcześniejszymi wnioskami, najwyższą czułość połączoną z dobrą precyzją utrzymuje Głęboki AE.

Na rysunku 5.20 przedstawiono działanie tej sieci trenowanej sposobem N2N na mniejszym fragmencie surowej klatki. Jak można zobaczyć na zbliżeniu, model ten był w stanie zauważalnie wyrównać tło i zarazem zredukować szum. Doprowadziło to odsłonięcia i detekcji dodatkowych gwiazd, które pierwotnie były w dużej mierze nierozróżnialne od szumu.



Rysunek 5.20. Wpływ działania Głębokiego AE (N2N) na jakość obrazu surowego i możliwości detekcji przedstawione na wybranym fragmencie.

5.6. Wnioski z opisem dalszego kierunku badań

W niniejszym rozdziale dokonano walidacji proponowanych rozwiązań z użyciem danych obrazowych nocnego nieba, wykorzystując klasyczne metryki oparte na pomiarach astrometrycznych oraz fotometrycznych. Potwierdzono, że modele dobrze radzące sobie z redukcją szumu można wytrenować z wykorzystaniem zarówno techniki Noise2Noise, jak i Noise2Clean. Należy przy tym podkreślić, że widoczna w przypadku danych syntetycznych różnica jakości wyników pomiędzy tymi podejściami nie jest aż tak znacząca w przypadku danych rzeczywistych. Ponadto, sieci neuronowe w zadanych warunkach są w stanie osiągać wyniki na podobnym, a często wyższym, poziomie niż algorytmy deterministyczne. Szczególnie optymistycznym wnioskiem z badań jest fakt, iż redukcja szumu w obrazach astronomicznych prowadzi do zdecydowanie lepszej detekcji oraz precyzyjniejszej fotometrii przy zachowaniu informacji o położeniu obiektów. Jest to o tyle istotne, że metody redukcji szumu oparte czy to na metodach deterministycznych, czy wykorzystujących sieci neuronowe, nie są obecnie powszechnie stosowane w astronomii obserwacyjnej, pozostając w obszarze badań i eksperymentów.

W pewnym stopniu poruszona została także kwestia zależności między stopniem złożoności modelu a jego zapotrzebowaniem na pamięć operacyjną. W większości sytuacji wytrenowana sieć powinna działać szybko i precyzyjnie, a także być możliwa do uruchomienia na jak najprostszym sprzęcie. Jak wykazano w tym dziale, czasem nawet prostsze modele, takie jak Głęboki AE, są w stanie osiągać wyniki lepsze niż znacznie bardziej złożone architektury, jak Astro U-Net.

Tematyka złożoności i wydajności sieci neuronowych zostanie rozwinięta w kolejnych rozdziałach na przykładzie danych pochodzących z obrazowania Słońca. W odróżnieniu od stosowanego dotąd podejścia „jeden do jednego”, w którym pojedynczy obraz wejściowy przetwarzany jest do odpowiadającego mu obrazu wyjściowego, zastosowana zostanie strategia „wiele do jednego”. Oznacza to, że model będzie trenowany na sekwencjach wielu obrazów wejściowych, by na ich podstawie generować pojedynczy wynik. To podejście umożliwi testowanym sieciom uzyskać więcej informacji o zmieniającej się w czasie strukturze szumu, co powinno skutkować otrzymaniem wysokiej jakości wyników. Przeprowadzona zostanie analiza szybkości działania rozpatrywanych architektur w ramach określenia alternatywy do algorytmu deterministycznego, który jest powszechnie stosowanym w tym zadaniu.

6. Redukcja szumu atmosferycznego w seriach danych obrazowych Słońca

W dotychczasowych eksperymentach uwagę zwrócono na zagadnienie przetwarzania pojedynczych obrazów astronomicznych w celu redukcji obecnego na nich szumu. Jak wykazano na przykładzie obrazów nocnego nieba, zastosowanie sieci neuronowych jest w stanie znacząco poprawić ich parametry. Jednakże zmienne warunki atmosferyczne sprawiają, że w przypadku analizowanych serii obserwacyjnych widoczność szczegółów może się znacząco różnić pomiędzy kolejnymi klatkami. Kolejne obrazy w serii mogą być nieco zniekształcane pod wpływem turbulencji atmosferycznych. Efektywność stosowanych rozwiązań jest przez to w dużej mierze zależna od jakości obrazu wejściowego.

W związku z tym postanowiono przeprowadzić badania nad wykorzystaniem danych wejściowych złożonych z sekwencji wielu klatek wykonanych jedna po drugiej. Taka technika pozwala na wykorzystanie informacji o zmienności zakłóceń w czasie, co powinno skutkować odzyskaniem (rekonstrukcją) analizowanych struktur, które na poszczególnych klatkach jawią się nieco inaczej.

Do eksperymentów wykorzystano dane pochodzące z obserwacji słonecznych, które stanowią materiał innego typu niż obserwacje gwiazd, rozszerzając zakres niniejszej pracy. Słońce jako jasne źródło światła o dużej dynamice, ma znacznie bogatszą strukturę powierzchniową, a dodatkowo znajduje się w niewielkiej odległości od Ziemi, co umożliwia obserwację tych struktur przy użyciu teleskopów niskorozdzielczych. Widoczne na nim liczne obszary aktywne, filamenty i protuberancje dają pole do eksperymentów nad wydajnością różnych podejść do rekonstrukcji obrazu idealnego.

Niniejszy rozdział, podobnie jak wcześniejsze, został podzielony na odrębne części odpowiadające poszczególnym zagadnieniom. W pierwszej z nich omówiono metody służące poprawie uzyskanych serii obrazów, obejmujące zarówno algorytmy deterministyczne, jak i proponowane w ostatnich latach rozwiązania oparte na sieciach neuronowych. Następnie opisano wykorzystany zbiór danych, który utworzono przy użyciu teleskopu opisanego w rozdziale *2.4.2. Obserwatoria Politechniki Śląskiej*. Trzecia i czwarta sekcja skupiają się na opisie testowanych architektur sieci oraz wskaźników jakości. Przedostatnia część opisuje analizę otrzymanych wyników, a ostatnia stanowi podsumowanie przeprowadzonych eksperymentów i wskazuje na kolejny problem badawczy poruszony w dalszej części rozprawy.

6.1. Rozwiązania stosowane w obrazowaniu słonecznym

W przypadku obserwacji słonecznych wykorzystywane są dwa algorytmy do osłabienia wpływu szumu atmosferycznego na dane obrazowe. Pierwszym z nich jest interferometria plamkowa [138][139], która wykorzystuje serie setek (lub więcej) obrazów rejestrowanych przy czasach ekspozycji rzędu kilku milisekund dla pojedynczej klatki. Poprzez zastosowanie na takich seriach transformaty Fouriera możliwe jest odtworzenie niezakłóconej informacji o strukturze obserwowanego obiektu. Praktyczne wykorzystanie tej metody jest jednak znacząco ograniczone ze względu braku dostępu do dynamicznie zmieniającego się wzorca funkcji rozmywającej obraz, który w przypadku obserwacji nocnych może być wygenerowany przy pomocy laserowej gwiazdy odniesienia lub zarejestrowany dla pobliskiej gwiazdy.

Drugim rozwiązaniem jest metoda ślepej dekonwolucji wieloklatkowej, MFBD [140][141][142] (ang. *Multi-Frame Blind Deconvolution*). W porównaniu do interferometrii plamkowej jest ona bardziej odporna na zmienność warunków atmosferycznych, co umożliwia jej stosowanie na obrazach o dłuższym czasie ekspozycji. Nie jest również wymagana znajomość postaci zmieniającej się w czasie funkcji rozmywającej obraz, gdyż ona jest właśnie iteracyjne oszacowywana przez algorytm. Ze względu na problemy z oceną funkcji zniekształcenia obrazu, MFBD jest częściej stosowane w praktyce obserwacji Słońca; między innymi stanowi istotny etap przetwarzania danych rejestrowanych przez teleskop STS. Znaczącą wadą tego podejścia są duże wymagania sprzętowe i bardzo długi czas przetwarzania danych, który wielokrotnie przekracza czas ich akwizycji. W badaniach przeprowadzonych w niniejszej rozprawie skorzystano z implementacji MFBD przygotowanej przez Instytut Fizyki Słońca Uniwersytetu Sztokholmu[6]. Wszelkie parametry pracy programu konsultowano z twórcą algorytmu oraz autorem prac dotyczących algorytmów MFBD, Matsem Löfdahlem.

Spośród prób zastosowania sieci neuronowych na uwagę niewątpliwie zasługuje praca [143], w której opisano badania nad opracowaniem modelu enkoder-dekoder oraz sieci rekurencyjnej, których celem było przetworzenie serii n obrazów w taki sposób, by obraz wynikowy miał jakość porównywalną z wynikiem działania MFBD. Sieci te były przez autorów dalej rozwijane w [144][145][146] i wykorzystano je jako punkt odniesienia do oceny podobnych rozwiązań [147][148]. Oczywiście oprócz tych podejść podjęto również próby wykorzystanie w tym celu odmiennych architektur i technik ich treningu [149][150][151].

[6] Implementacja dostępna jest na portalu GitHub pod adresem: *https://github.com/ISP-SST*.

W ramach przeprowadzonych w niniejszej pracy eksperymentów postanowiono wykorzystać omówioną strukturę enkoder-dekoder [143], która odpowiadała przyjętym wcześniej założeniom dotyczącym wykorzystania sieci kompresujących dane. Potraktowano tę architekturę jako przykład dużej sieci i porównano jej wydajność z mniejszymi modelami. Tak jak w wymienionym artykule, celem było uzyskanie wyników porównywalnych z działaniem algorytmu MFBD.

6.2. Opis danych SUTO Solar

W opisanych w poprzedniej części podejściach opartych na sieciach neuronowych wykorzystywano głównie dane syntetyczne albo pochodzące z najnowocześniejszych teleskopów słonecznych, takich jak STS. Instrumenty tego typu umożliwiają obserwację bardzo małych fragmentów Słońca z dużą dokładnością. Jakość pozyskiwanych przez nie obrazów jest dodatkowo poprawiana poprzez użycie rozwiązań technicznych, do których wlicza się między innymi optyka adaptacyjna. Charakterystyka danych tego typu zdecydowanie różni się od tych pozyskiwanych przez małe obserwatoria, których główną zaletą jest możliwość monitorowania całej tarczy słonecznej z wielu lokalizacji w ramach ciągłego „patrolu słonecznego".

Z tego względu w przeprowadzonych badaniach wykorzystano zbiór SUTO-Solar [152], który zawiera wielomiesięczne dane obserwacyjne zebrane przez mały teleskop słoneczny zarządzany przez grupę SUTO [153]. Chociaż sprzęt ten jest w stanie obserwować całą tarczę Słońca, na utworzony zbiór składają się jedynie fragmenty 100×100 lub 200×200 pikseli, na których uwidocznione są obszary aktywne na Słońcu. Zbiór ten zawiera jedynie interesujące struktury, a dodatkowo jest on w miarę kompaktowy – pojedynczy wycinek 100×100 pikseli wymaga ponad 500 razy mniej miejsca niż cała klatka. Podejście to jest szczególnie istotne w przypadku zapisu serii obserwacyjnych, na które składają się setki klatek. Na rysunku 6.1 przedstawiono standardowy obraz pełnego dysku słonecznego, pozyskiwany przez mały teleskop słoneczny, ze zbliżeniem na trzy przykładowe fragmenty o rozmiarze 100×100 pikseli (zaznaczone jako A, B i C).

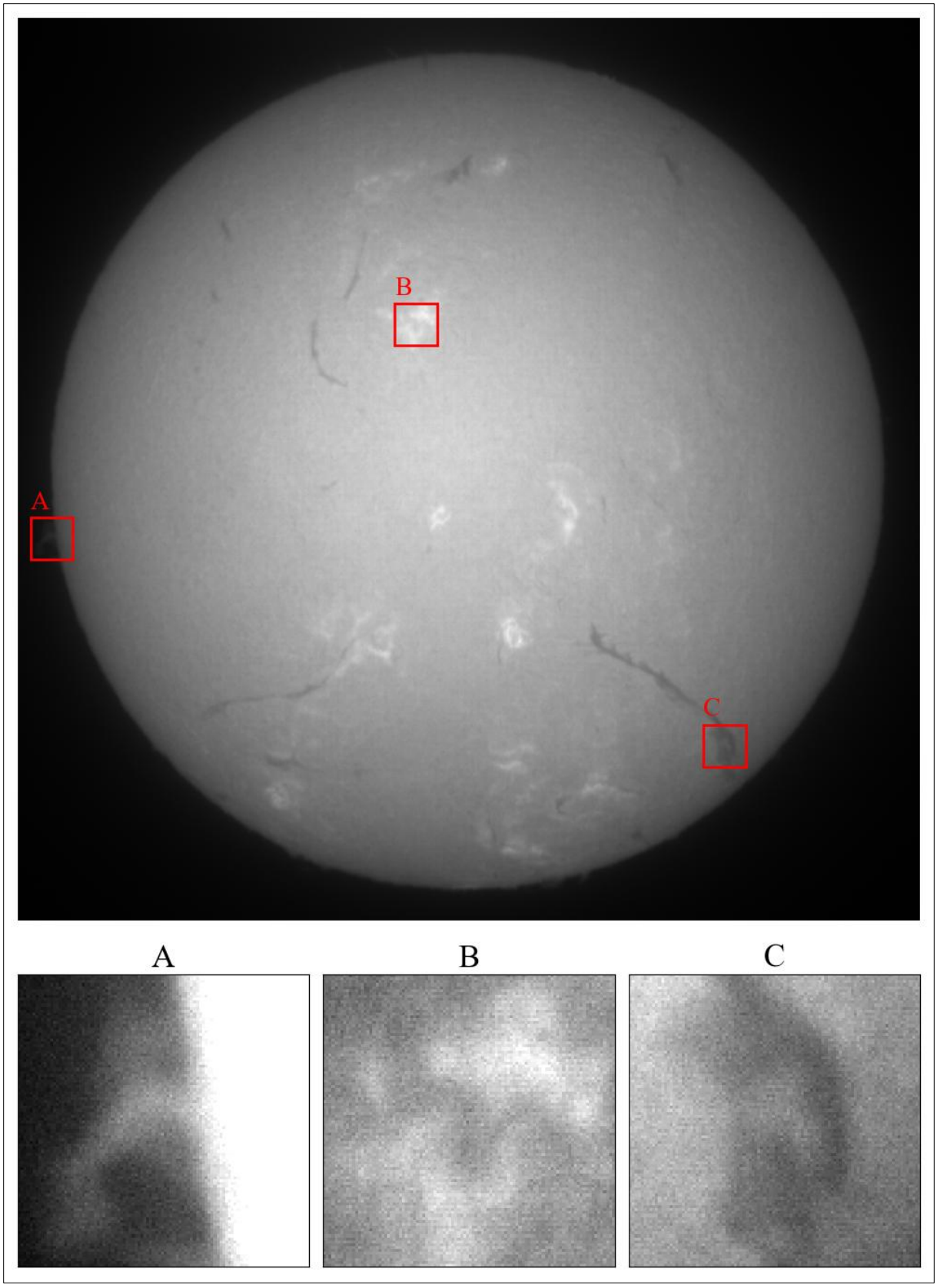
Istotną cechą użytego zbioru jest także jego różnorodność. Utworzono go podczas obserwacji w czasie ponad jednego roku, co jest rzadko spotykane w tego typu badaniach. Zazwyczaj używa się danych zarejestrowanych w ciągu jednego lub kilku dni, co może powodować, że analiza jest w dużym stopniu obarczona problemami związanymi ze stanem

atmosfery w momencie obserwacji (na przykład poprzez wyjątkowo sprzyjające lub szczególnie niekorzystne warunki).

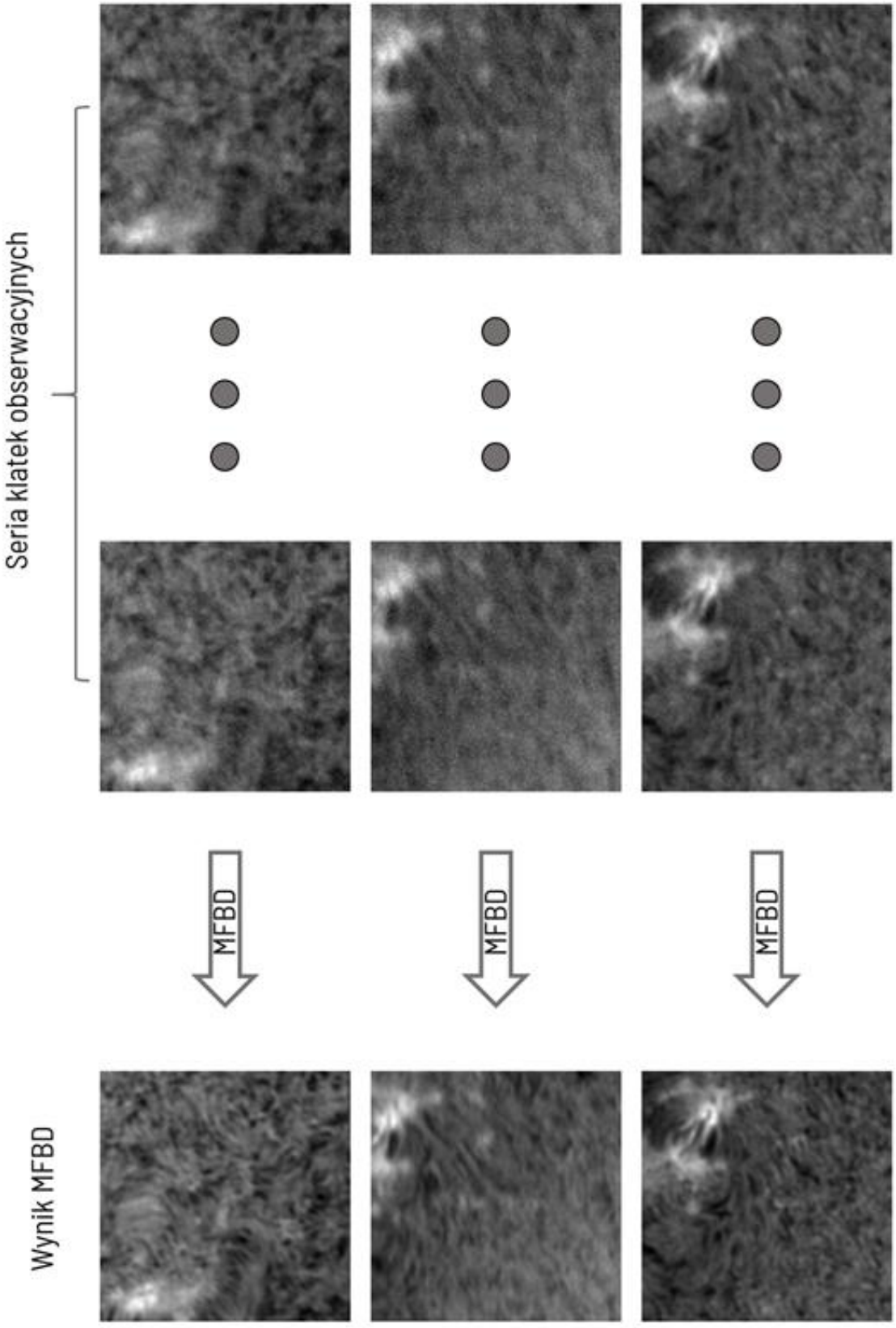
Na wybrany zbiór składa się kilka tysięcy serii liczących 100 lub 200 takich pojedynczych klatek. Rejestrowano je z częstotliwością około 30 klatek na sekundę, co odpowiada czasowi obserwacji od 3 do 6 sekund. W teleskopach o wysokiej rozdzielczości tak długi czas pozwoliłby zaobserwować już pewne zmiany w chromosferze, jednak niska rozdzielczość kątowa użytego sprzętu wyklucza wykrywalność takich zmian. Fakt niezmienności obserwowanego fragmentu Słońca umożliwia zatem efektywne wykorzystanie algorytmu MFBD do przetwarzania serii obrazów, którego przykładowe efekty działania przedstawiono na rysunku 6.2. U góry rysunku znajdują się fragmenty 100×100 pikseli będące losowo wybranymi klatkami serii, a poniżej wyniki przetworzenia ich przez algorytm MFBD.

Oryginalny zbiór składa się z ponad 3800 serii liczących po 100 klatek o rozmiarze 100×100 pikseli i odpowiadających im wyników działania MFBD mających rozmiar 70×70 pikseli. Różnica wymiarów spowodowana jest działaniem algorytmu, który odrzuca część informacji znajdującą się przy krawędzi obrazów surowych, zakładając przesunięcia obrazu jako efekt turbulencji. Aby porównać działanie sieci z wynikami tej metody, wszystkie klatki surowe zostały wyrównane względem odpowiednich klatek MFBD i przycięte do tego samego rozmiaru. W kolejnym kroku poddano je dodatkowemu przycięciu do rozmiaru 64×64, by uniknąć zmian ich wymiarów związanych z działaniem sieci. Przetworzone dane podzielono w stosunku 4:1 na zbiór treningowy i walidacyjny, co zaprezentowano w tabeli 6.1.

Do sprawdzenia wydajności sieci utworzono osobny zbiór testowy, który nie wchodził pierwotnie w skład zestawu SUTO-Solar. Początkowo składał się on z 405 serii obserwacji liczących po 200 klatek każda (200×200 pikseli) i odpowiadającym im wyników MFBD o rozmiarze 177×177 pikseli. Po wyrównaniu i przycięciu odpowiadających sobie klatek uzyskano zbiór obrazów o rozmiarze 176×176 pikseli. Dla tego zbioru wyznaczono jednak znacznie więcej wyników przetwarzania surowych serii przez algorytm MFBD. Mianowicie, uzyskano wyniki przetworzenia pierwszych 10, 20, 50, 100 i 200 surowych klatek, oznaczonych odpowiednio jako $MFBD_{10}$, $MFBD_{20}$, $MFBD_{50}$, $MFBD_{100}$ i $MFBD_{200}$. Przyjęto, że wykorzystanie większej liczby klatek skutkuje otrzymaniem wyników MFBD o wyższej jakości, wobec czego potraktowano obrazy $MFBD_{200}$ jako główne dane porównawcze. Pozostałe obrazy MFBD zastosowano jako pomocnicze punkty odniesienia, które umożliwiły weryfikację tego, czy sieci mogą przewyższyć MFBD w przypadku dostępu do ograniczonej liczby klatek.



Rysunek 6.1. Dane z obrazowania Słońca. U góry widok całej tarczy Słońca (2304×2304 piksele), a na dole zbliżenie na poszczególne fragmenty 100×100 ze zmienionym kontrastem.



Rysunek 6.2. Wpływ algorytmu MFBD na jakość przetwarzanych obrazów.

Tabela 6.1. Podział zbioru SUTO-Solar na podzbiory.

Typ podzbioru danych	**Liczba serii obserwacyjnych**	**Liczba klatek przypadająca na serię**	**Rozmiar pojedynczej klatki**
Treningowy	3 081	100	64×64
Walidacyjny	771	100	64×64
Testowy	405	200	176×176

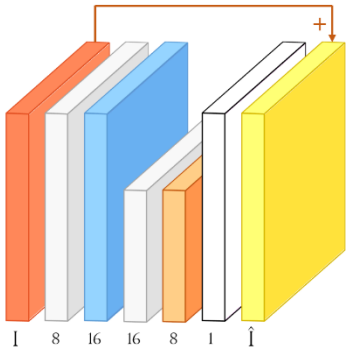
6.3. Wykorzystane architektury

Do badań wykorzystano 3 różne struktury sieci. Pierwszą z nich jest model zaproponowany w [143], a dwie pozostałe są jego odpowiednikami o zmniejszonym rozmiarze. Dla rozróżnienia sieci te określono jako małą (rysunek 6.3), średnią (rysunek 6.4) i dużą (rysunek 6.5). Podział ten oparty jest na liczbie warstw sieci i całkowitej liczbie ich parametrów, która wynosi odpowiednio około 12 000, 150 000 i 3 200 000 parametrów przy 100 klatkach wejściowych. Należy przy tym podkreślić, że chociaż w oryginalnej publikacji bazowa architektura została określona jako enkoder-dekoder, jest to w rzeczywistości U-Net, ponieważ posiada ona połączenia pomijające, które wprowadzają dodatkowy przepływ danych w sieci.

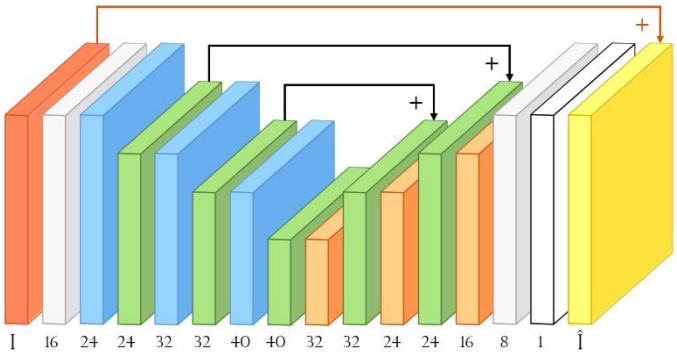
Strukturę sieci ponownie przedstawiono w postaci kolorowych schematów blokowych, jednak w tym wypadku występują znaczące różnice pomiędzy sieciami małą i średnią a siecią dużą. W związku z tym, w pierwszej kolejności opisano bloki dla dwóch pierwszych modeli, a następnie wyszczególniono zmiany występujące w najbardziej złożonym modelu. Zmiany wielkości bloków ponownie odpowiadają dwukrotnej zmianie rozmiaru przestrzennego danych, liczby pod blokami odpowiadają wyjściowej liczbie map cech, a kolorowe bloki odpowiadają następującym operacjom:

a. blok czerwony oznacza serię obrazów wejściowych sieci I złożoną z n następujących po sobie klatek (odpowiednio: 10, 20, 50 lub 100);
b. blok żółty symbolizuje pojedynczy obraz wyjściowy sieci $\hat{I}$;
c. blok szary symbolizuje jednostkę funkcjonalną złożoną z warstwy uzupełnień przez odbicie lustrzane, warstwy konwolucyjnej o jądrze 3×3 i kroku równym 1, warstwy normalizacji wsadowej oraz warstwy aktywacji LeakyReLU (w tej kolejności);
d. blok niebieski obejmuje te same operacje co blok szary, zmieniona jest jedynie długość kroku, która wynosi 2;

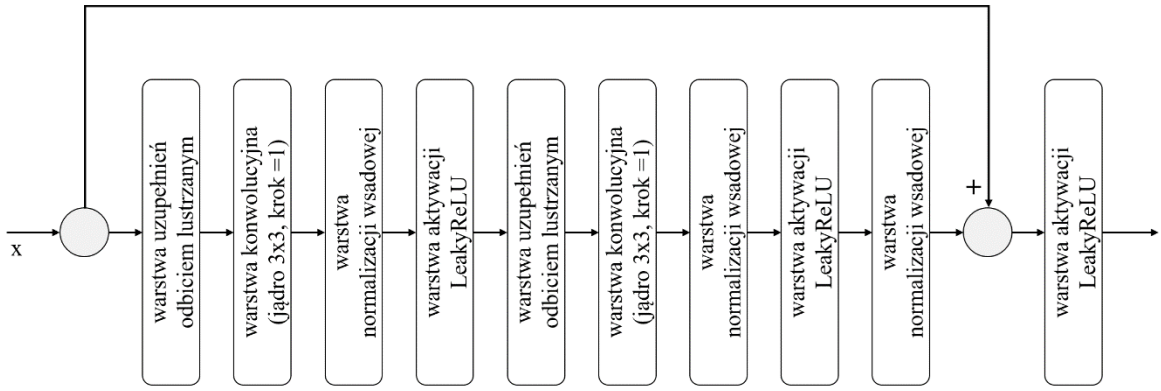
e. blok pomarańczowy odpowiada zastosowaniu warstwy zwiększenia rozdzielczości wykorzystującej interpolację metodą najbliższego sąsiada i przetworzeniu wyniku przez blok szary;
f. blok zielony oznacza jednostkę rezydualną, która przetwarza sygnał w sposób przedstawiony na rysunku 6.5;
g. biały blok z czarnym konturem odpowiada pojedynczej warstwie konwolucyjnej o jądrze 3×3 i kroku równym 1 bez zastosowania funkcji aktywacji.



Rysunek 6.3. Mała sieć.

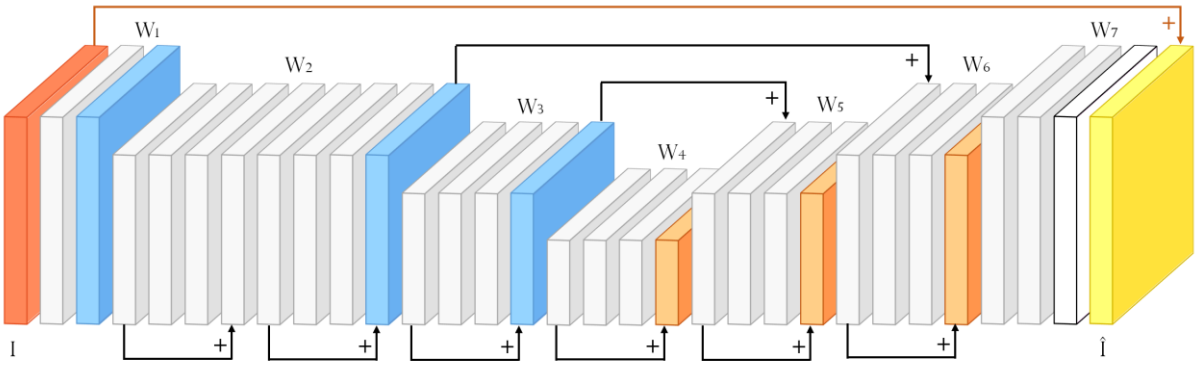


Rysunek 6.4. Średnia sieć.



Rysunek 6.5. Schemat jednostek rezydualnych (zielone bloki na schemacie średniej sieci).

W przypadku dużej sieci ma miejsce pewna istotna zmiana. Działanie każdego bloku pozostaje takie samo, ale zmienia się kolejność operacji w podstawowym bloku, zaznaczonym kolorem szarym. W sieci dużej blok szary odpowiada następującej kolejności warstw: warstwa normalizacji wsadowej, warstwa aktywacji ReLU, warstwa uzupełnień przez odbicie lustrzane i na końcu warstwa konwolucyjna o jądrze 3×3 i kroku równym 1. Szczegółowy opis parametrów poszczególnych warstw dla tej sieci zamieszczono w tabeli 6.2. Grupy (poziomy) poszczególnych warstw sieci opisano jako $W_{i,j}$, gdzie i oznacza numer grupy, a j numer warstwy w i-tej grupie.



Rysunek 6.6. Duża sieć.

Tabela 6.2. Opis parametrów poszczególnych warstw dużej sieci.

Warstwa	Rozmiar jądra	Krok	Liczba map cech wyjściowych
$W_{1,0}$	7×7	1	32
$W_{1,1}$	3×3	2	64
$W_{2,1} - W_{2,7}$	3×3	1	64
$W_{2,8}$	3×3	2	128
$W_{3,1} - W_{3,3}$	3×3	1	128
$W_{3,4}$	3×3	2	256
$W_{4,1} - W_{4,3}$	3×3	1	256
$W_{4,4}$	3×3	1	128
$W_{5,1} - W_{5,3}$	3×3	1	128
$W_{5,4}$	3×3	1	64
$W_{6,1} - W_{6,3}$	3×3	1	64
$W_{6,4}$	3×3	1	64
$W_{7,1}$	3×3	1	64
$W_{7,2}$	3×3	1	16
$W_{7,3}$	1×1	1	1

Na zmieszczonych schematach przepływ danych został przedstawiony za pomocą strzałek. W przypadku małej i średniej sieci, czarne strzałki wskazują na operację dodawania **wyjściowych** tensorów wybranej warstwy do tensorów wejściowych warstwy docelowej. W dużej sieci strzałki te oznaczają natomiast dodanie **wejściowych** tensorów jednej warstwy do tensorów wejściowych wskazywanej warstwy.

Znacznie istotniejsza jest operacja oznaczona poprzez strzałki brązowe. Odpowiada ona dodaniu pewnej porcji informacji wejściowej do wyjścia sieci. Według bazowego podejścia [143] do obrazu wyjściowego dodawany był pojedynczy, pierwszy obraz przetwarzanej serii. Wobec tego zadaniem sieci było znalezienie maski, która po zsumowaniu z surowym pierwszym obrazem wejściowym spowoduje zredukowanie obecnego na nim szumu. Biorąc pod uwagę losowość miejscowych zakłóceń, zadanie to uznać można za wymagające. Wobec tego zaproponowano zmianę pierwszego obrazu serii na obraz uśredniony. Jak wykazano w poprzednich rozdziałach, obecny na takim obrazie szum jest w znacznej mierze zredukowany, a detale są lepiej widoczne. Zadaniem sieci byłoby zatem zredukowanie

powstałych rozmyć i wyostrzenie widocznych struktur, co uznano za zadanie znacznie prostsze, bardziej logiczne w koncepcji, a przez to mogące skutkować lepszymi wynikami. W opisie wyników część uwagi poświęcono analizie tego podejścia w oparciu o wyniki uzyskiwane przez dużą sieć.

6.4. Ocena efektywności rozwiązań

Wykorzystany zbiór danych zawiera obrazy przedstawiające złożone struktury chromosfery słonecznej, w związku z czym postanowiono ponownie wykorzystać wskaźnik FSIM. Miarę PSNR uznano w tym wypadku za niewystarczającą i zastąpiono ją metryką „wierności informacji wizualnej" VIF [154] (ang. *Visual Information Fidelity),* która nie porównuje bezpośrednio wartości pikseli, ale ocenia zachowaną ilość informacji w obrazie. Jej wartość mieści się zazwyczaj w przedziale [0; 1], ale może przekroczyć 1 w sytuacjach, gdy na porównywanym obrazie występuje wyższy kontrast.

Zdecydowano się również na wprowadzenie dodatkowej miary: podobieństwa gradientów po filtracji medianowej MFGS [155] (ang. *Median Filter-Gradient Similarity*). Wskaźnik ten opiera się na porównaniu gradientów w obrazie przed i po przetworzeniu go przez filtr medianowy, nie wymagając przy tym obrazu referencyjnego. MFGS jest w stanie bardzo dobrze opisać jakość istotnych struktur, wobec czego jest uznanym standardem w analizie obrazów Słońca [156]. Wartość tej metryki zawsze mieści się w przedziale [0; 1], a wysokie wartości świadczą o dobrej jakości rozważanego obrazu.

Oprócz tego dokonano porównania szybkości działania poszczególnych rozwiązań. Oszacowano średnie czasy przetwarzania pojedynczej serii obrazów testowych przez każdą z sieci i zestawiono je z czasami obliczeń algorytmu MFBD.

6.5. Analiza wyników

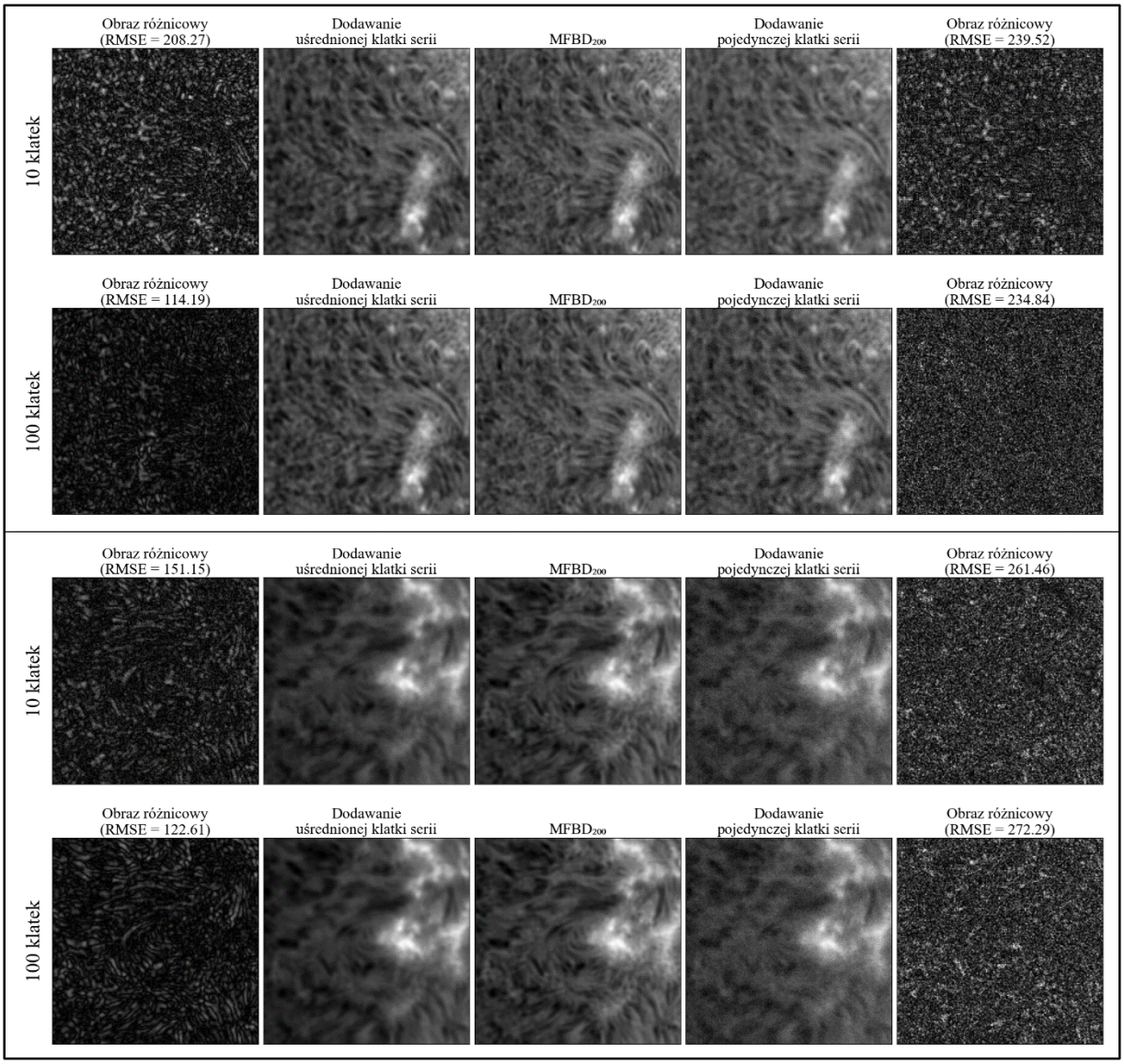
Sieci zostały wytrenowane i przetestowane na sprzęcie wyposażonym w 16-rdzeniowy procesor AMD Ryzen 9 5950X (3,40 GHz) oraz kartę graficzną GeForce RTX 3090Ti 24 GB RAM. Wyniki MFBD uzyskano natomiast za pomocą specjalnie skonfigurowanego oprogramowania działającego na sprzęcie wyposażonym w 20-rdzeniowy procesor Intel Xeon E5-2680 (2,80 GHz). Do procesu trenowania użyto standardowego optymalizatora Adam oraz

stałego współczynnika uczenia równego 0,00001. Wagi każdej warstwy zainicjalizowano za pomocą algorytmu He [157], a sieci trenowano przez 1000 epok na obrazach o rozmiarze 64×64. Wielkość zbioru treningowego została ponownie zwiększona z wykorzystaniem losowo dobieranych operacji rotacji oraz odbić w pionie i w poziomie. W kolejnych podrozdziałach dokonano analizy jakości obrazów wynikowych względem wykorzystanej metody, typu obrazu dodawanego na wyjście sieci, a także szybkości przetwarzania danych.

6.5.1. Zamiana obrazu pojedynczego na uśredniony

Testy rozpoczęto od oceny typu maskowania dokonywanego przez sieć. Na rysunku 6.7 przedstawiono wyniki przetwarzania dwóch różnych serii obrazów składających się z 10 i 100 klatek, jak oznaczono po lewej stronie każdego rzędu. Powstała macierz obrazów zawiera pięć kolumn. Punktem wyjścia jest środkowa kolumna, w której umieszczono obrazy referencyjne, $MFBD_{200}$. W kolumnie po jej lewej stronie znajduje się wynik działania sieci, która maskowała klatkę uśrednioną, a w skrajnej lewej kolumnie przedstawiono obraz różnicowy tego obrazu i $MFBD_{200}$. Kolumny po prawej przedstawiają analogiczne obrazy uzyskane dla sieci maskującej klatkę pojedynczą. Dodatkowo, nad obrazami różnicowymi przedstawiono wartości RMSE, które odpowiadają pierwiastkowi z błędu średniokwadratowego (ang. *Root Mean Square Error*). Im wyższa jest wartość tej miary, tym większe są różnice jasności pikseli pomiędzy porównywanymi obrazami.

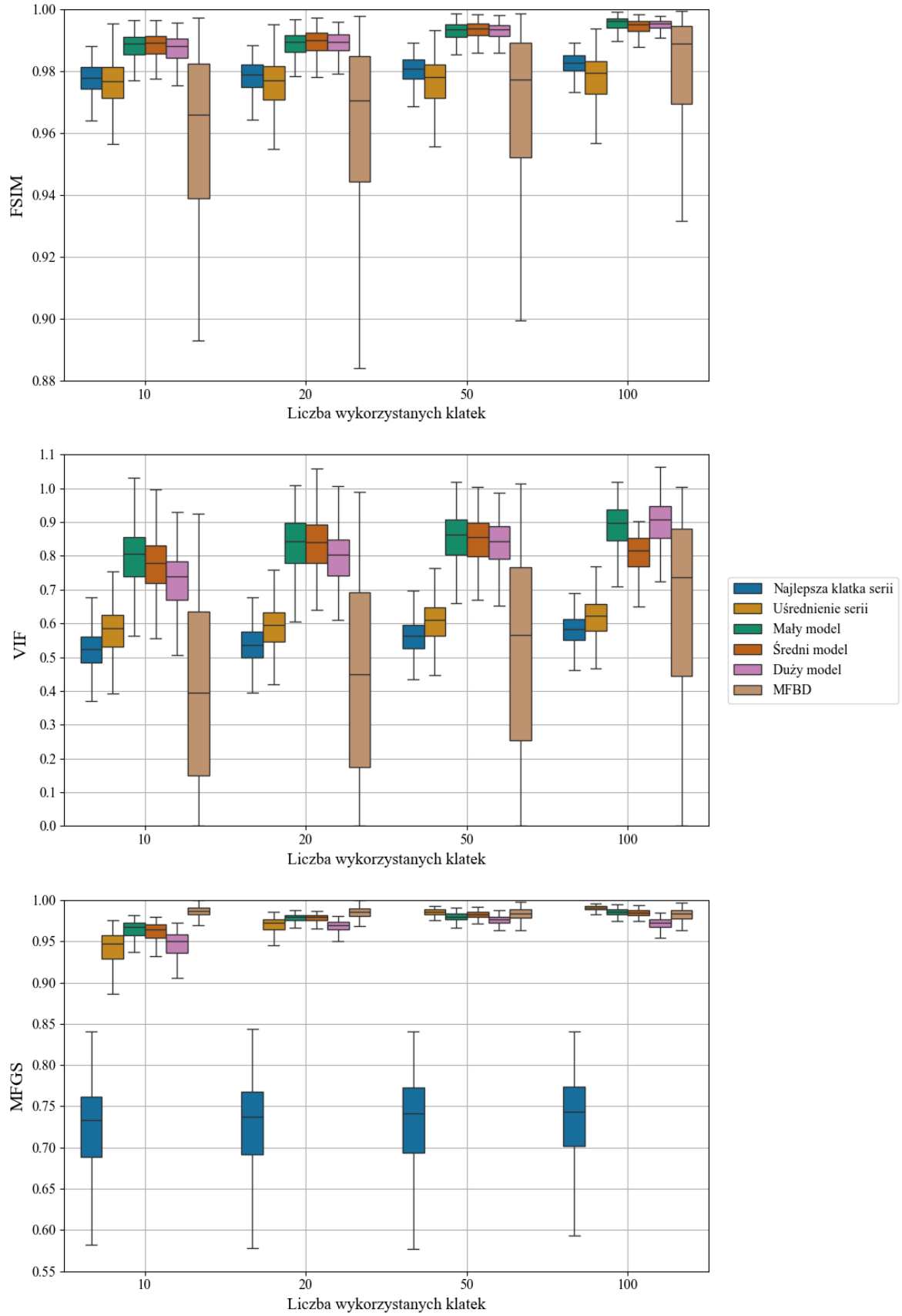
Analizując przedstawione wyniki, można zauważyć, że maskowanie klatki uśrednionej skutkuje pojawieniem się różnic w jasności pośrodku poszczególnych struktur, co jest oznaczone obecnością białych plam. Krawędzie są przy tym dobrze rozróżnialne. Przy wykorzystaniu 10 klatek można zauważyć, że występują punktowe zmiany, które świadczą o tym, że sieć nie była w stanie usunąć całości szumu. Jednak po zwiększeniu liczby klatek wejściowych do 100 zmiany te niemal całkowicie znikają. Świadczy to o tym, że sieci nie uczą się dobrze maskować szumu, przez co duże znaczenie ma uśrednienie odpowiedniej liczby klatek wejściowych. W przypadku sieci dodających do wyjścia pierwszy obraz serii zaszumienie przetworzonych danych sieci jest znaczące. Punktowe różnice są łatwe do zauważenia i w znaczący sposób degradują ogólną jakość danych, co potwierdzają wyższe wartości RMSE. Obserwacja ta potwierdza przyjęte założenie, że maskowanie klatki uśrednionej jest podejściem skuteczniejszym w poprawy jakości obrazu od maskowania pojedynczej klatki serii.



Rysunek 6.7. Wpływ proponowanej modyfikacji na wydajność sieci na przykładzie dwóch obrazów.

6.5.2. Porównanie działania sieci względem $MFBD_{200}$

Działanie każdej z metod oceniono z użyciem wykresów pudełkowych, na których przedstawiono wyniki pogrupowane względem wykorzystanej liczby klatek wejściowych (rysunek 6.8). Oprócz obrazów wyjściowych sieci, do porównania wykorzystano: wyniki MFBD uzyskane przy mniejszej liczbie klatek, klatki uśrednione, a także klatki wejściowe, które w danej serii miały najwyższą wartość wskaźnika FSIM. Wybór ostatniego typu porównywanych obrazów opowiada zastosowaniu techniki „szczęśliwego obrazowania" (ang. *lucky imaging*) polegającej na selekcji klatek najmniej zniekształconych przez atmosferę, które są wykorzystywane w dalszych badaniach.



Rysunek 6.8. Statystyczne porównanie wartości wykorzystanych metryk.

Jak można zauważyć, dla wyników MFBD wartości metryk FSIM i VIF odstają od pozostałych, wskazując na pogorszenie jakości przetworzonych przez ten algorytm obrazów. Jedynie wartości MFGS odpowiadają oczekiwanym, wysokim wartościom. W związku z tym postanowiono dokonać analizy wizualnej, by wyjaśnić przyczynę tego nieoczekiwanego zjawiska. Na rysunku 6.9 przedstawiono zestawienie wyników MFBD, najlepszych klatek serii, klatek uśrednionych i $MFBD_{200}$ dla wybranej serii obserwacji. Każdy z rzędów odpowiada innej liczbie rozważanych klatek serii, a nad każdym obrazem wypisano wartości wskaźników FSIM, VIF i MFGS obliczonych względem $MFBD_{200}$.

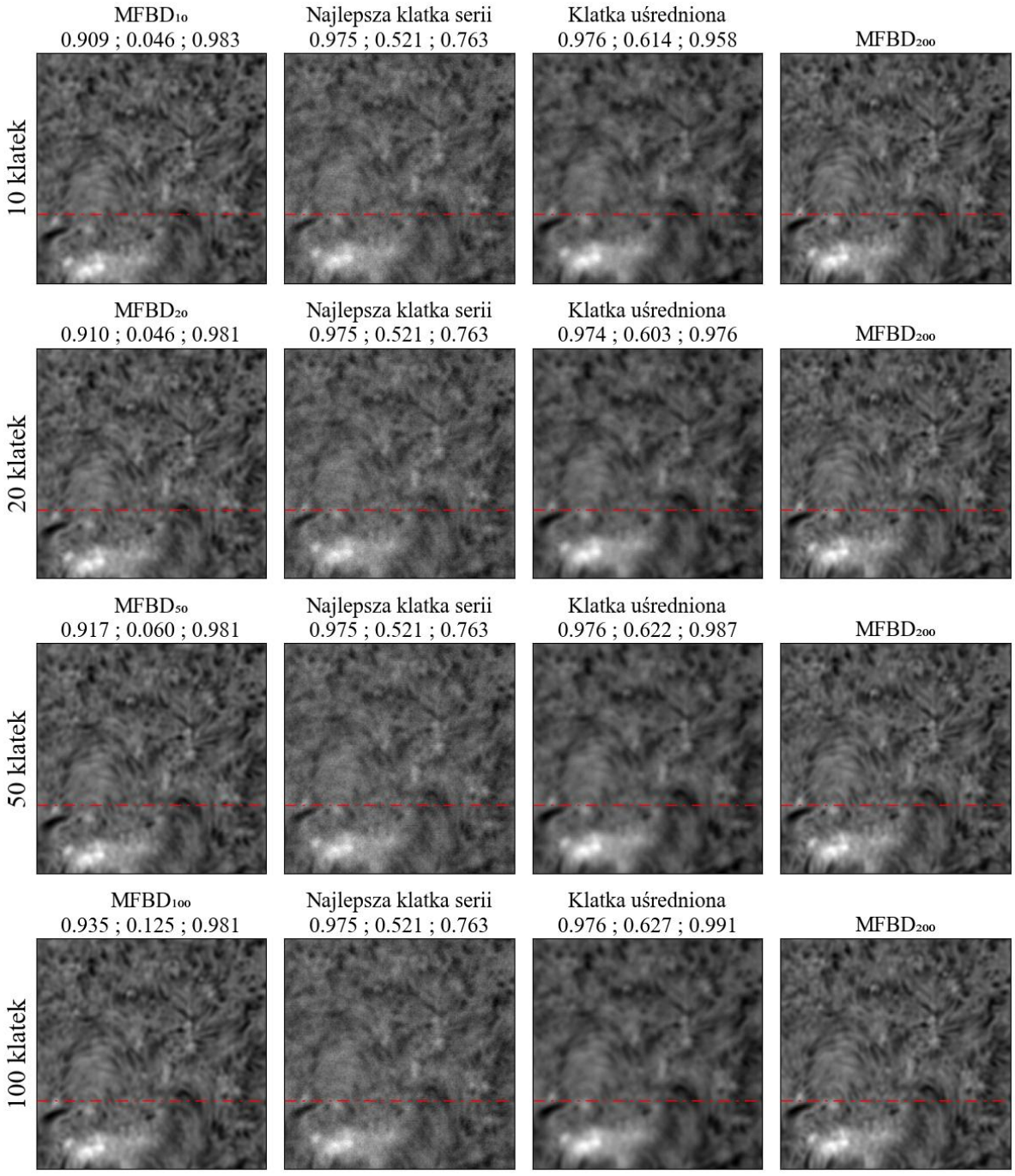
Porównując ze sobą poszczególne obrazy, można stwierdzić, że najlepsza klatka jest w dużym stopniu obciążona szumem, a na klatce uśrednionej wiele szczegółów jest rozmytych. Pomimo tego obrazy te w każdym przypadku uzyskały wyższe wartości FSIM i VIF niż wyniki MFBD. Spowodowane jest to obecnym na obrazach MFBD przesunięciu, które można zauważyć, porównując położenie struktur względem czerwonej przerywanej linii na każdym z obrazów.

Przesunięcie to spowodowane jest działaniem samego algorytmu MFBD, który dokonuje przycięcia krawędzi oryginalnego obrazu tak, by uzyskać wynik o najlepszej jakości. Wobec zmiennej widoczności struktur na przestrzeni serii, przycięcie to jest wykonywane w sposób zależny od długości rozważanej serii. Sprawia to, że ocena ilościowa jest w tym wypadku niewystarczająca i należy przede wszystkim polegać na ocenia wizualnej, która potwierdza wysoką jakość wyników MFBD. W przypadku obrazów testowych efekt ten nie występuje, ponieważ zostały one wyrównane i przycięte względem $MFBD_{200}$, Wobec tego można porównywać je z wykorzystaniem wybranych metryk.

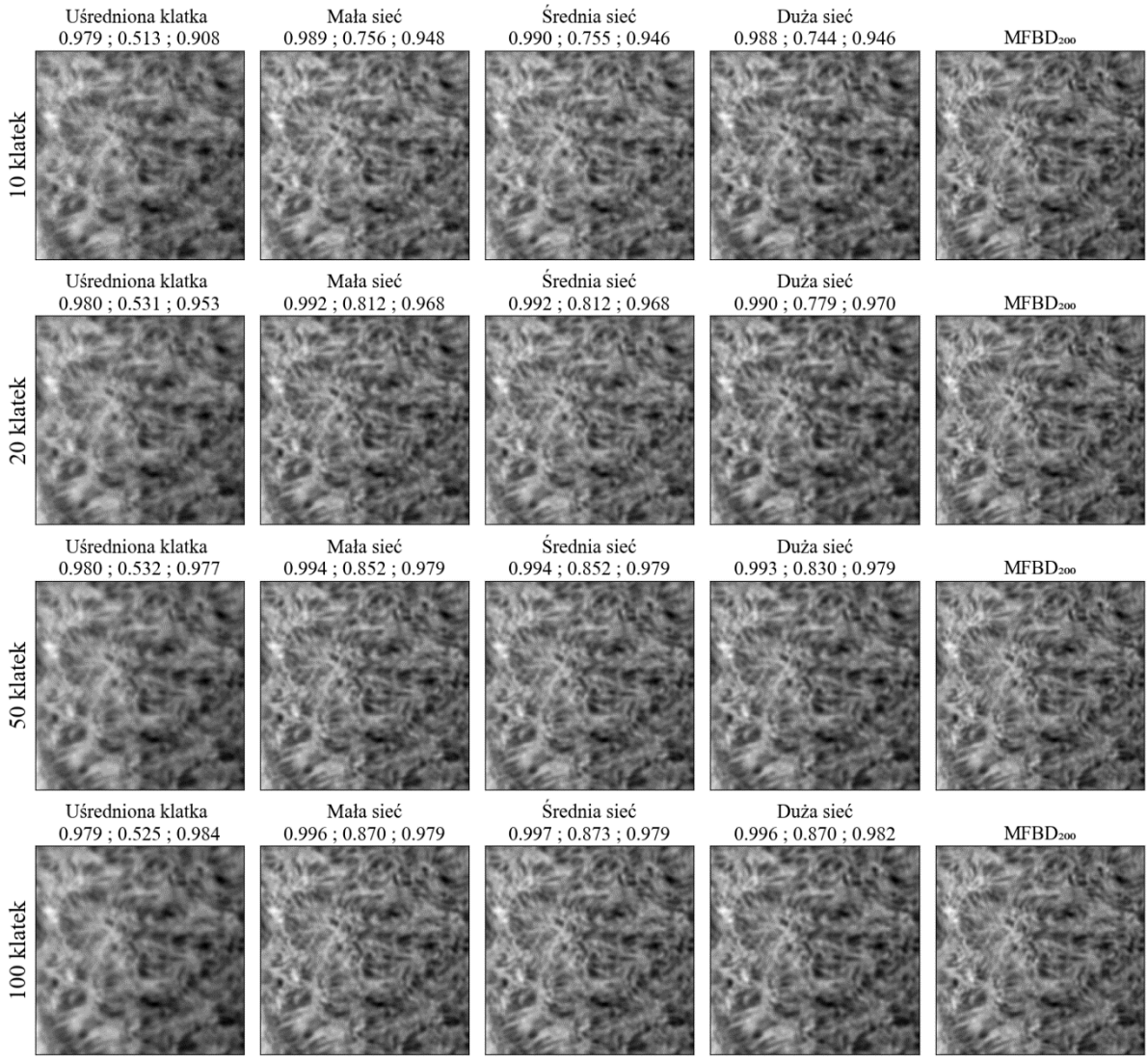
Na rysunku 6.10 przedstawiono, analogiczne do rysunku 6.9, zestawienie wyników działania poszczególnych sieci z klatką uśrednioną i $MFBD_{200}$. Wykorzystanie dowolnej sieci skutkuje uzyskaniem niemal identycznego obrazu, a różnice w wartościach wskaźników są niewielkie. Obserwacja ta znajduje potwierdzenie w przedstawionych na rysunku 6.8 różnicach w rozrzutach tych wartości, które dla małej i średniej sieci są w większości przypadków zaniedbywalnie małe, a dla sieci dużej różnice są zauważalne, ale względnie niewielkie.

Dodatkowo, rysunek 6.10 wskazuje na to, że do oceny rozmycia najbardziej nadaje się miara VIF, która pozwala łatwo odróżnić klatkę uśrednioną od reszty obrazów. Analiza tego rysunku potwierdza również słuszność postawionej tezy badawczej. Jak można zaobserwować,

w przypadku przeprowadzonego eksperymentu nawet relatywnie małe sieci były w stanie uzyskać rezultaty porównywalne z tymi osiąganymi przez sieci znacznie większe.



Rysunek 6.9. Porównanie wyników dla danych surowych i MFBD wyznaczanego dla zadanej liczby klatek. Liczby nad poszczególnymi obrazami oznaczają wartości obliczonych wartości wskaźników FSIM, VIF i MFGS. Czerwone linie przerywane wykreślono na tej samej wysokości w każdym obrazie, by oszacować występujące na obrazach przesunięcia.



Rysunek 6.10. Porównanie wyników działania poszczególnych sieci. Liczby nad poszczególnymi obrazami oznaczają wartości obliczonych wartości wskaźników FSIM, VIF i MFGS

6.5.3. Szybkość przetwarzania danych

W tabeli 6.3 przedstawiono średni czas wymagany do przetworzenia pojedynczej serii obrazów o rozmiarze 176×176 pikseli z wykorzystaniem jednostki GPU. W tabeli 6.4 przedstawiono natomiast analogiczne wyniki dla sieci uruchomionych na CPU (ang. *Central Processing Unit*), czyli na procesorze. Algorytm MFBD jest uruchamiany jako program na specjalnie przystosowanym sprzęcie, wobec czego nie zmierzono czasu działania samego algorytmu, ale całego oprogramowania. Podany czas jest w związku z tym wartością obarczoną pewną niepewnością. W przypadku sieci neuronowych zmierzono jedynie czas potrzebny do

wczytania i przetworzenia danych, wobec czego jest on podany ze znacznie większą dokładnością.

Tabela 6.3. Porównanie czasu przetwarzania pojedynczej serii złożonej z obrazów o rozmiarze 176×176 pikseli z wykorzystaniem GPU.

Liczba klatek wejściowych	**Czas przetwarzania danych [ms]**			
	Mała sieć	**Średnia sieć**	**Duża sieć**	**MFBD**
10	2	5	7	27 000
20	2	5	8	28 000
50	3	6	8	35 000
100	4	8	9	38 000

Tabela 6.4. Porównanie czasu przetwarzania pojedynczej serii złożonej z obrazów o rozmiarze 176×176 pikseli z wykorzystaniem CPU.

Liczba klatek wejściowych	**Czas przetwarzania danych [ms]**			
	Mała sieć	**Średnia sieć**	**Duża sieć**	**MFBD**
10	2	7	25	27 000
20	3	8	26	28 000
50	5	9	28	35 000
100	7	12	31	38 000

Jak można zauważyć, różnica pomiędzy szybkością testowanych modeli a MFBD jest ogromna i wynosi 3 rzędy wielkości na korzyść sieci neuronowych. Różnica pomiędzy szybkością sieci na GPU a CPU jest przy tym zauważalna, ale w przypadku tego eksperymentu nie jest znacząca. Można stwierdzić, że sieci są w stanie we wszystkich analizowanych sytuacjach przetwarzać dane w czasie rzeczywistym, który jest zdecydowanie krótszy od czasu zarejestrowania pojedynczej klatki.

Algorytm MFBD wymaga natomiast stosowania wydajnych jednostek obliczeniowych wyposażonych w wielordzeniowe procesory, lecz mimo to nie jest w stanie zapewnić oczekiwanej szybkości przetwarzanych danych. Problem ten zyskuje na znaczeniu

w przypadku danych o wysokiej rozdzielczości przestrzennej, odpowiadających obserwacjom całej tarczy Słońca. Przykładowo, przetworzenie serii 200 obrazów pełnego dysku słonecznego, czyli 200 macierzy 2304x2304 pikseli, przez algorytm MFBD zajmuje około 2 godzin. Całkowicie eliminuje to możliwość wykorzystania tej metody do poprawy obrazów w sensownym, użytecznym czasie. W sytuacji, gdy zjawiska zachodzące na słońcu wymagają pilnej reakcji, tak duże opóźnienia są nieakceptowalne.

6.6. Ograniczenia stosowanych rozwiązań

W niniejszym rozdziale opisano przeprowadzone badania, które miały na celu przetestowanie wpływu ilości danych wejściowych na działanie sieci. Jak udowodniono, wykorzystanie sieci neuronowych stanowi atrakcyjną alternatywę wobec najlepszego algorytmu deterministycznego, MFBD. Otrzymywane rezultaty cechują się porównywalną jakością, przy jednocześnie krótszym o rzędy wielkości czasie potrzebnym na wykonanie obliczeń. Co istotne, w przypadku obrazów niskorozdzielczych dobre rezultaty można uzyskać także przy użyciu stosunkowo niewielkich architektur.

Należy jednak podkreślić, że przeprowadzone testy bazowały na obrazach o wymiarach 176×176 pikseli. Natomiast w zastosowaniu docelowym sieć powinna być w stanie przetwarzać pełne obrazy tarczy słonecznej o znacznie wyższej rozdzielczości przestrzennej. W przypadku wykorzystania obrazów o rozmiarze 2304×2304 piksele, tak jak przedstawiono na rysunku 6.1, wielkość samych danych wejściowych wzrosłaby ponad 170-krotnie. Wiązałoby się to również z większym zapotrzebowaniem sieci na pamięć operacyjną, co jest pewnym ograniczeniem w kontekście możliwości stosowania proponowanych rozwiązań. Chociaż nie jest ono tak drastyczne w przypadku użytego sprzętu, może stanowić zasadniczy problem przy próbach uruchomienia sieci na bardziej kompaktowym sprzęcie, który można by zamontować tuż przy zdalnie sterowanym teleskopie słonecznym, obrazującym cały dysk Słońca.

W kolejnym, ostatnim, rozdziale pracy podjęta zostanie tematyka kompresji sieci neuronowych. Jest to obecnie jedno z najintensywniej rozwijanych zagadnień w dziedzinie uczenia maszynowego, co wynika z rosnącej złożoności i rozmiarów współczesnych modeli. Coraz częściej analizowane są sposoby ich efektywnego uruchamiania na urządzeniach takich jak mikrokomputery, telefony komórkowe czy mikrokontrolery. Stosowane metody kompresji umożliwiają znaczną redukcję zapotrzebowania na pamięć i moc obliczeniową przy zachowaniu wysokiej jakości działania modeli.

7. Kompresja sieci na przykładzie danych słonecznych

Sieci neuronowe cechują się obecnie dużą złożonością oraz wysokimi wymaganiami sprzętowymi, co czasami stanowi wyzwanie w ich praktycznym zastosowaniu. W związku z tym, można zaobserwować rosnące zainteresowanie metodami kompresji modeli, które umożliwiają zmniejszenie ich rozmiaru przy zachowaniu akceptowalnej jakości uzyskiwanych przez nie wyników.

W wielu sytuacjach techniki te można stosować zarówno na wytrenowanych wcześniej sieciach, jak i wdrożyć je do treningu od jego początku. Wymagają one jednak odpowiedniego dostrojenia, co może znacząco wydłużyć i utrudnić proces uczenia. Podejściem alternatywnym jest wyszkolenie sieci o rozmiarze dostosowanym bezpośrednio do charakterystyki rozważanego problemu. Strategia ta może skutkować uzyskaniem rozwiązania o oczekiwanej efektywności, lecz z pominięciem wspomnianej optymalizacji.

W ostatniej części niniejszej rozprawy postanowiono zweryfikować, które z dwóch wymienionych podejść może skutkować lepszymi rezultatami. W tym celu przeprowadzono eksperymenty z wykorzystaniem omówionego w poprzednim rozdziale zbioru SUTO-Solar i wytrenowanych na nim sieci. Uwagę skoncentrowano na przycinaniu sieci, jednej z najpopularniejszych metod kompresji, która zmienia strukturę przetwarzanych modeli. Pozwoliło to ocenić wpływ kompresji na działanie testowanych sieci pod względem zapotrzebowania na pamięć operacyjną, szybkość przetwarzania danych i ich końcową jakość. Dodatkowo sprawdzono możliwości uruchomienia testowanych modeli na mikrokomputerze Raspberry Pi 5, który uznano za dobry przykład urządzenia spełniającego wymogi przenośności, kompaktowości i niewielkiego poboru mocy.

Treść niniejszego rozdziału została podzielona na cztery części. W pierwszej z nich omówiono najpopularniejsze techniki kompresji sieci neuronowych. Sekcja druga skupia się na opisie metody przycinania (ang. *pruning*) sieci, czyli usuwania z niej wybranych elementów. W kolejnej części poddano analizie uzyskane wyniki, natomiast w ostatniej podsumowano wyniki badań.

7.1. Metody kompresji sieci

Możliwości implementacji sieci neuronowych są w wielu wypadkach znacząco ograniczone przez ich duże wymagania sprzętowe. By zaradzić temu problemowi, opracowano wiele metod kompresji sieci neuronowych, które mają na celu redukcję liczby parametrów, złożoności obliczeniowej i zapotrzebowania na pamięć przy jednoczesnym zachowaniu wysokiej wydajności. Do najważniejszych z tych metod należą przycinanie (ang. *pruning*), destylacja wiedzy (ang. *knowledge distillation*) oraz kwantyzacja (ang. *quantization*).

Przycinanie sieci neuronowych polega na usuwaniu wybranych elementów modelu w celu uzyskania rzadszej struktury. Jedną z pionierskich prac w tym obszarze była [158], w której autorzy przedstawili metodę usuwania wag o niewielkim wpływie na funkcję kosztu. Metoda ta została w późniejszych latach rozwinięta o iteracyjne przycinanie połączone z ponownym treningiem [159], a następnie o automatyzację tego procesu [160]. Ideę przycinania rozszerzono również o „hipotezę biletu na loterię” [161][162], według której w dużych modelach istnieją podzbiory małych i efektywnych sieci, które można „wylosować”, czyli uzyskać, poprzez odpowiednie zastosowanie przycinania i inicjalizacji.

Kwantyzacja polega natomiast na redukcji precyzji reprezentowanych wag i aktywacji. Zamiast 32-bitowych liczb zmiennoprzecinkowych, model może działać na liczbach 8-bitowych, a w skrajnych sytuacjach nawet na binarnych. Znacząco zmniejsza to zapotrzebowanie na pamięć i umożliwia wydajne korzystanie z akceleratorów sprzętowych przystosowanych do tego typu danych.

Destylacja wiedzy oznacza transfer wiedzy z dużego modelu, nauczyciela, do mniejszego modelu, ucznia. Koncepcja ta opiera się na trenowaniu ucznia tak, aby odtwarzał wyniki generowane przez nauczyciela, a tym samym nauczył się bardziej złożonych zależności, mając mniej parametrów niż nauczyciel [163]. Jednakże, takie podejście nie zawsze skutkuje uzyskaniem wyników lepszych od tych otrzymywanych w sytuacji, gdy uczeń trenowany jest na zbiorze uczącym z pominięciem nauczyciela [164]. Metoda ta jest z tego powodu często łączona z innymi technikami [165], co pozwala zwiększyć jej efektywność.

Spośród łączonych metod kompresji najpopularniejsze jest przycinanie stosowane razem z kwantyzacją [166][167]. Takie podejście może w zależności od charakterystyki zadania zredukować złożoność obliczeniową i zapewnić lepszą kompaktowość sieci [168]. Niemniej, w przypadku prowadzonych w tym rozdziale rozważań, wykorzystana została jedynie technika

przycinania sieci, ponieważ celem badań była analiza wpływu złożoności sieci na ich wydajność, a nie ich optymalizacja.

7.2. Przycinanie sieci

Przycinanie sieci neuronowych stanowi jedną z bardziej złożonych metod kompresji, ponieważ w znaczący sposób ingeruje w strukturę sieci. Polega ona na wyborze najmniej znaczących wag, neuronów lub nawet całych warstw i ich usunięciu. By zachować jak najwyższą wydajność przetwarzanych modeli, proces ten jest zazwyczaj realizowany iteracyjnie. W każdej z iteracji usuwa się określoną liczbę elementów, a następnie dostraja się sieć, przeprowadzając ponowny trening. Z uwagi na przeprowadzony wcześniej proces uczenia, trwa on jednak znacznie krócej od głównego treningu; najczęściej przyjmuje się liczbę epok odpowiadającą 5-10% liczby epok głównego treningu.

Metody przycinania sieci można podzielić według kryteriów określających ich sposób funkcjonowania. Poniżej przedstawiono trzy najważniejsze z nich.

1) Podział względem **zasięgu działania**
   a) Przycinanie globalne – skupia się na analizie wszystkich elementów sieci jednocześnie, dzięki czemu można precyzyjnie wskazać te najmniej istotne. To podejście może skutkować uzyskaniem lepszych wyników, jednak jest ono znacznie bardziej złożonym zagadnieniem. W wielu przypadkach wiąże się z potrzebą wprowadzenia pewnych ograniczeń, które chronią model przed rozregulowaniem. Przykładowo, w sieciach typu U-Net należy zadbać o odpowiadające sobie wymiary danych w połączeniach pomijających, by model mógł działać prawidłowo.
   b) Przycinanie lokalne – polega na niezależnej analizie elementów do usunięcia w każdej warstwie osobno. Metoda ta ułatwia implementację, daje większą kontrolę nad strukturą modelu, a także korzystnie wpływa na zachowanie stabilności procesu uczenia.

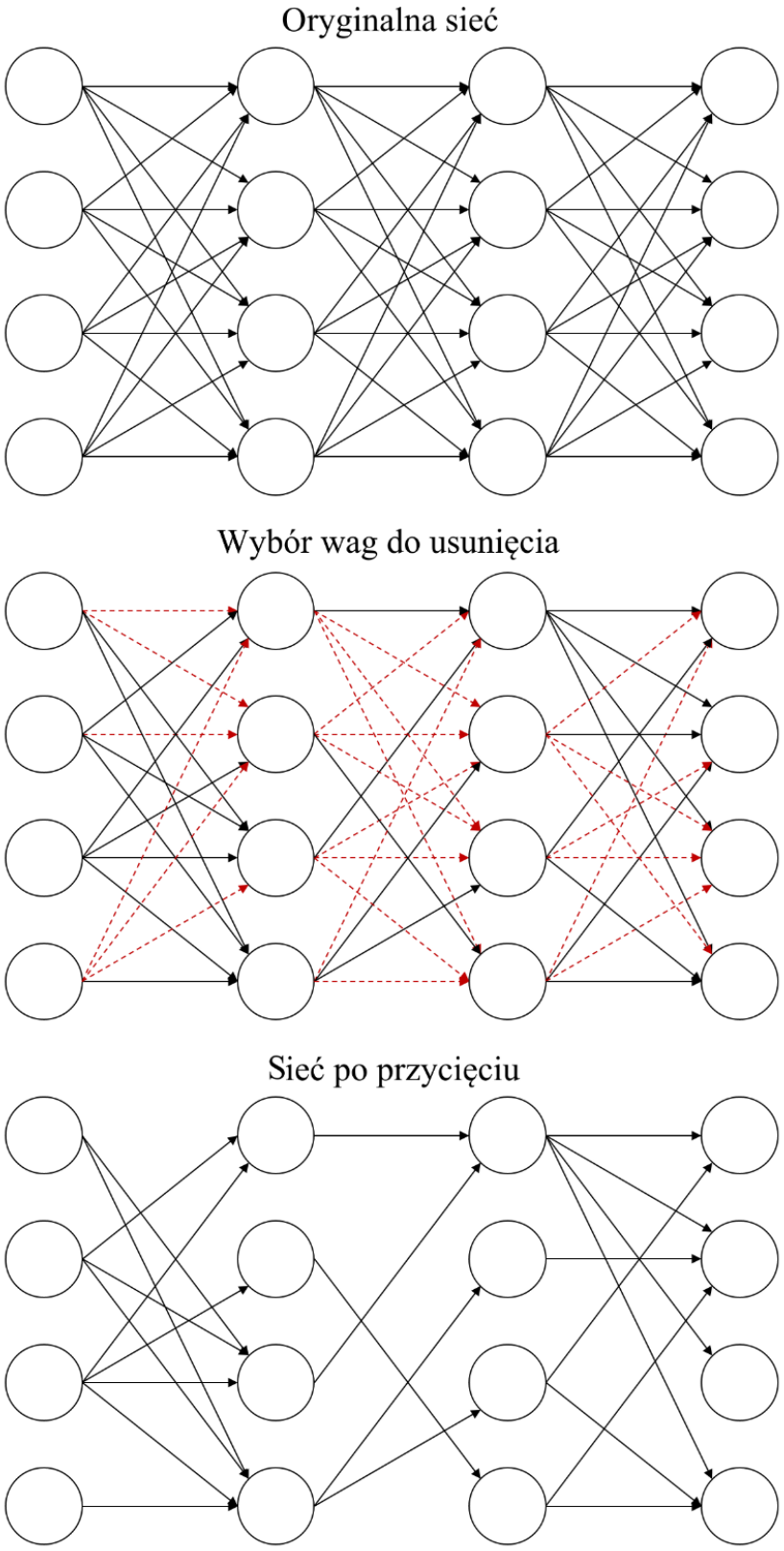
2) Podział względem **struktury usuwanych elementów**
   a) Przycinanie niestrukturalne – polega na wyzerowaniu wartości poszczególnych wag sieci, co przy odpowiednio silnym przycięciu może doprowadzić do zmiany macierzy wag w macierze rzadkie. Nie jest to jednak równoznaczne z usunięciem tych elementów, przez co rozmiar modelu nie ulega zmniejszeniu. Szybkość przetwarzania danych

również nie ulega poprawie, jeśli wykorzystany sprzęt i oprogramowanie nie wspierają przetwarzania macierzy rzadkich.

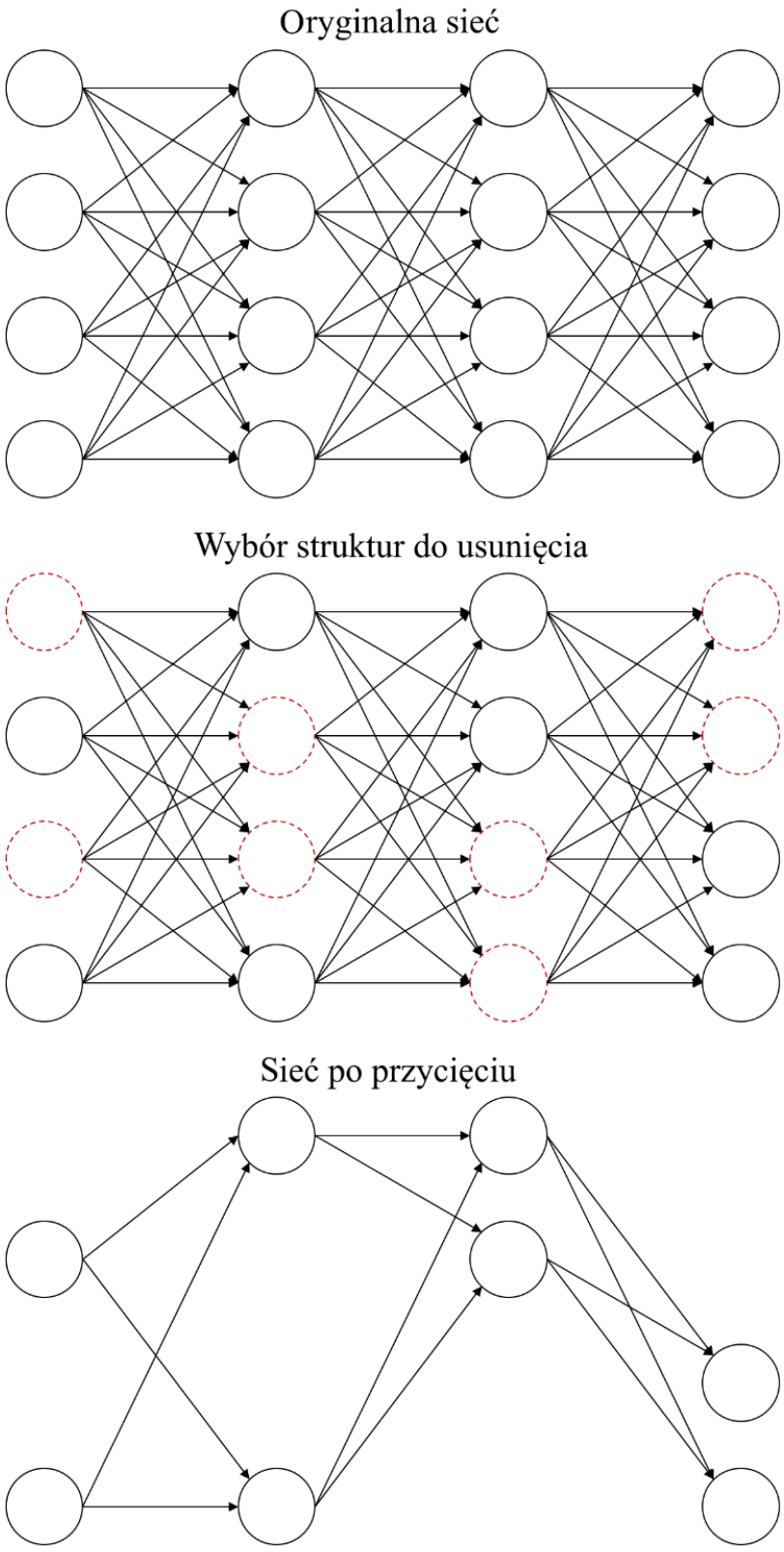
b) Przycinanie strukturalne – opiera się na usuwaniu całych struktur, takich jak neurony w warstwach gęstych czy mapy cech w warstwach konwolucyjnych. W efekcie zawsze skutkuje zmniejszeniem zapotrzebowania sprzętowego.

3) Podział względem **metody wyboru usuwanych elementów**
   a) Podejście oparte na wartości wag – polega na usunięciu elementów o najmniejszej wartości bezwzględnej wag, operując na założeniu, że wagi bliskie zeru mają niewielki wpływ na pracę sieci. W odróżnieniu od pozostałych metod nie wymaga przetworzenia danych przez sieć.
   b) Podejścia oparte na wartości funkcji straty – polegają na usuwaniu elementów, których zniknięcie ma najmniejszy wpływ na wydajność sieci określoną przez wartość funkcji straty.
   c) Podejście oparte na wartości aktywacji elementu – polega na usunięciu elementów o najmniejszej wartości funkcji aktywacji. Ta metoda może nie działać w przypadku globalnego przycinania sieci, w których występuje wiele różnych funkcji aktywacji.

Na rysunku 7.1 przedstawiono niestrukturalne przycinanie globalne na przykładzie warstw gęstych złożonych z 4 neuronów. Z pierwotnych 48 połączeń wybrane zostały 24 (50%) najmniej znaczące, które zaznaczono czerwonymi przerywanymi liniami. Następnie wagi tych połączeń zostały wyzerowane. Jak można zauważyć, liczba połączeń w każdej warstwie jest w tym przypadku inna, co może mieć miejsce jedynie przy przycięciu globalnym. Na rysunku 7.2 przedstawiono analogiczny schemat działania strukturalnego przycinania lokalnego. W każdej warstwie wybrano 50% (2 z 4) najmniej istotnych neuronów, które zostały usunięte.



Rysunek 7.1. Niestrukturalne przycinanie globalne.



Rysunek 7.2. Strukturalne przycinanie lokalne.

7.3. Analiza wyników

Wszystkie z omawianych w poprzednim rozdziale sieci zostały poddane strukturalnemu przycinaniu lokalnemu, które przeprowadzono dla dwóch różnych kryteriów wyboru map cech do usunięcia – wartości wag i wartości aktywacji. Przeprowadzone zostało ono iteracyjnie tak, że w każdym kroku najpierw usuwano 5% początkowej liczby map cech we wszystkich warstwach, a następnie dostrajano przycięte sieci w ponownym treningu, który trwał 50 epok. Do celów porównawczych wykorzystano sieci przycięte o 10%, 25%, 50% i 80%.

7.3.1. Zapotrzebowanie sprzętowe i szybkość działania

Parametry porównywanych sieci przedstawiono w postaci tabel 7.1, 7.2 i 7.3, z których każda opisuje jeden z rodzajów przycinanych sieci; odpowiednio są to sieci: mała, średnia i duża. Pierwsza z kolumn opisuje liczbę klatek wejściowych, które przetwarza dany wariant sieci. W drugiej kolumnie przedstawiono stopień przycięcia sieci, a w trzeciej liczbę wszystkich parametrów. Następnie przedstawione zostały wymagania każdej z sieci względem dostępnej pamięci operacyjnej, których wartości oszacowano na podstawie pomiaru zużycia pamięci układu GPU. Jako punkt odniesienia do tych wyników należy nadmienić, że inicjalizacja sieci małej, średniej i dużej wymagała odpowiednio 0,06 MB, 0,62 MB i 12,18 MB wolnej pamięci. Wczytanie serii 100 obrazów o rozmiarze 176×176 pikseli wiązało się z zalokowaniem 12 MB pamięci, natomiast dla serii 100 obrazów o rozmiarze 2304×2304 pikseli wartość ta wyniosła 2 026 MB. W ostatnich kolumnach zapisano oszacowany pomiar czasu przetwarzania pojedynczej serii danych. Szarym kolorem zaznaczono wiersze zawierające informacje o oryginalnych, nieprzyciętych sieciach.

Jak można zauważyć, zależność między stopniem przycięcia, a liczbą wszystkich parametrów sieci nie zmienia się liniowo. Jest to spowodowane zależnościami pomiędzy usuwanymi strukturami – zmniejszanie liczby map cech w każdej z warstw powoduje dodatkowe zmniejszenie wymiarowości danych przetwarzanych przez następną warstwę. W efekcie przycięcie sieci o 10% elementów powoduje nieproporcjonalnie dużą zmianę w liczbie parametrów.

W przypadku pamięci operacyjnej zmiany zachodzą w mniejszym stopniu i jedynie dla dużej sieci można zaobserwować systematyczny spadek zapotrzebowania na zasoby sprzętowe. W przypadku sieci średniej kompresja modelu przetwarzającego 100 klatek nie skutkuje

znaczącym obniżeniem wymagań. Natomiast dla małej sieci podobne ograniczenie zaczyna się już przy przetwarzaniu 50 klatek. Co istotne, duża sieć poddana 80% przycinaniu ma parametry zbliżone do nieprzetworzonej sieci średniej, co pozwala precyzyjnie porównać ich wydajność.

Tabela 7.1. Parametry porównywanych wariantów małych sieci.

Liczba klatek wejściowych	**Procent przycięcia**	**Liczba parametrów**	**Wymagana pamięć operacyjna [MB]**		**Czas przetwarzania pojedynczej serii [ms]**	
			176×176	**2304×2304**	**176×176**	**2304×2304**
10	–	5 545	8,05	1 368,47	2	108
	10%	4 348	7,22	1 226,65	3	107
	25%	3 295	6,20	1 056,03	1	106
	50%	1 621	4,53	770,80	1	107
	80%	418	2,64	445,87	1	102
20	–	6 265	9,24	1 570,97	2	232
	10%	4 987	8,40	1 430,16	2	234
	25%	3 835	7,39	1 258,95	2	228
	50%	1 981	5,71	976,02	1	221
	80%	598	5,03	852,52	2	217
50	–	8 425	13,12	2 219,18	3	487
	10%	6 868	13,00	2 199,17	2	479
	25%	5 455	12,87	2 179,17	2	481
	50%	3 061	12,63	2 139,16	3	473
	80%	1 138	12,38	2 097,65	2	472
100	–	12 025	39,09	6 245,12	4	861
	10%	10 018	37,90	6 387,11	3	863
	25%	8 155	37,66	6 367,11	4	861
	50%	4 861	37,17	6 165,08	3	864
	80%	2 038	36,67	6 205,04	3	852

Czas przetwarzania danych o wymiarze 2304×2304 pikseli (cały dysk słoneczny) w głównej mierze zależy od liczby klatek w seriach wejściowych. Jej zmiana powoduje

zauważalną zmianę szybkości sieci i to o wiele większą niż ta pojawiająca się na skutek kompresji. W przypadku mniejszego rozmiaru przestrzennego danych wspomniany czas zmienia się minimalnie, ale zawsze wynosi mniej niż 10 ms. Pozwala to stwierdzić, że testowane sieci stanowią doskonałą alternatywę dla MFBD, ponieważ w badanych przypadkach są w stanie uzyskać dobrej jakości wyniki w czasie wyraźnie krótszym niż czas akwizycji serii danych.

Tabela 7.2. Parametry porównywanych wariantów średniej sieci.

Liczba klatek wejściowych	**Przycięcie**	**Liczba parametrów**	**Wymagana pamięć operacyjna [MB]**		**Czas przetwarzania pojedynczego obrazu [ms]**	
			176×176	**2304×2304**	**176×176**	**2304×2304**
10	–	134 633	15,38	2 528,61	5	144
	10%	110 677	14,00	2 309,98	5	138
	25%	76 351	11,80	1 959,47	4	138
	50%	34 385	8,28	1 381,73	5	124
	80%	5 719	3,98	666,15	4	116
20	–	136 073	17,10	2 733,11	5	239
	10%	111 937	15,95	2 514,11	4	238
	25%	77 431	13,28	2 162,18	5	240
	50%	35 205	9,47	1 587,02	5	241
	80%	5 989	5,23	872,84	4	241
50	–	140 393	20,13	3 338,63	6	453
	10%	115 717	18,75	3 119,38	7	449
	25%	80 671	16,54	2 767,37	6	449
	50%	37 365	13,28	2 218,34	5	437
	80%	6 799	12,58	2 117,98	5	432
100	–	147 593	40,66	6 407,71	8	894
	10%	122 017	40,53	6 391,62	7	896
	25%	86 071	40,39	6 327,47	7	889
	50%	40 965	39,26	6 245,29	6	876
	80%	8 149	36,99	6 225,37	6	872

Tabela 7.3. Parametry porównywanych wariantów dużej sieci.

Liczba klatek wejściowych	**Przycięcie**	**Liczba parametrów**	**Wymagana pamięć operacyjna [MB]**		**Czas przetwarzania pojedynczego obrazu [ms]**	
			176×176	**2304×2304**	**176×176**	**2304×2304**
10	–	3 152 469	84,33	11 884,37	7	303
	10%	2 552 246	72,52	10 480,44	7	293
	25%	1 775 493	58,75	8 723,95	6	259
	50%	791 093	37,44	5 880,15	7	200
	80%	129 435	15,17	2 500,65	6	159
20	–	3 155 369	87,32	12 089,51	8	391
	10%	2 554 876	72,97	10 683,57	7	383
	25%	1 777 673	59,90	8 925,54	8	353
	50%	792 553	39,16	6 084,65	6	295
	80%	129 995	16,36	2 703,32	7	262
50	–	3 164 069	88,37	12 693,54	8	655
	10%	2 562 766	76,14	11 289,35	7	643
	25%	1 784 213	62,91	9 532,14	7	608
	50%	796 933	43,04	6 692,73	7	545
	80%	131 675	19,91	3 306,63	6	469
100	–	3 178 569	96,17	13 707,72	9	1 018
	10%	2 575 916	82,89	12 302,15	8	1 012
	25%	1 795 113	69,74	10 541,29	9	1 008
	50%	804 233	48,98	7 701,26	8	958
	80%	134 475	38,21	6 367,16	8	905

7.3.2. Ocena jakości przetworzonych obrazów

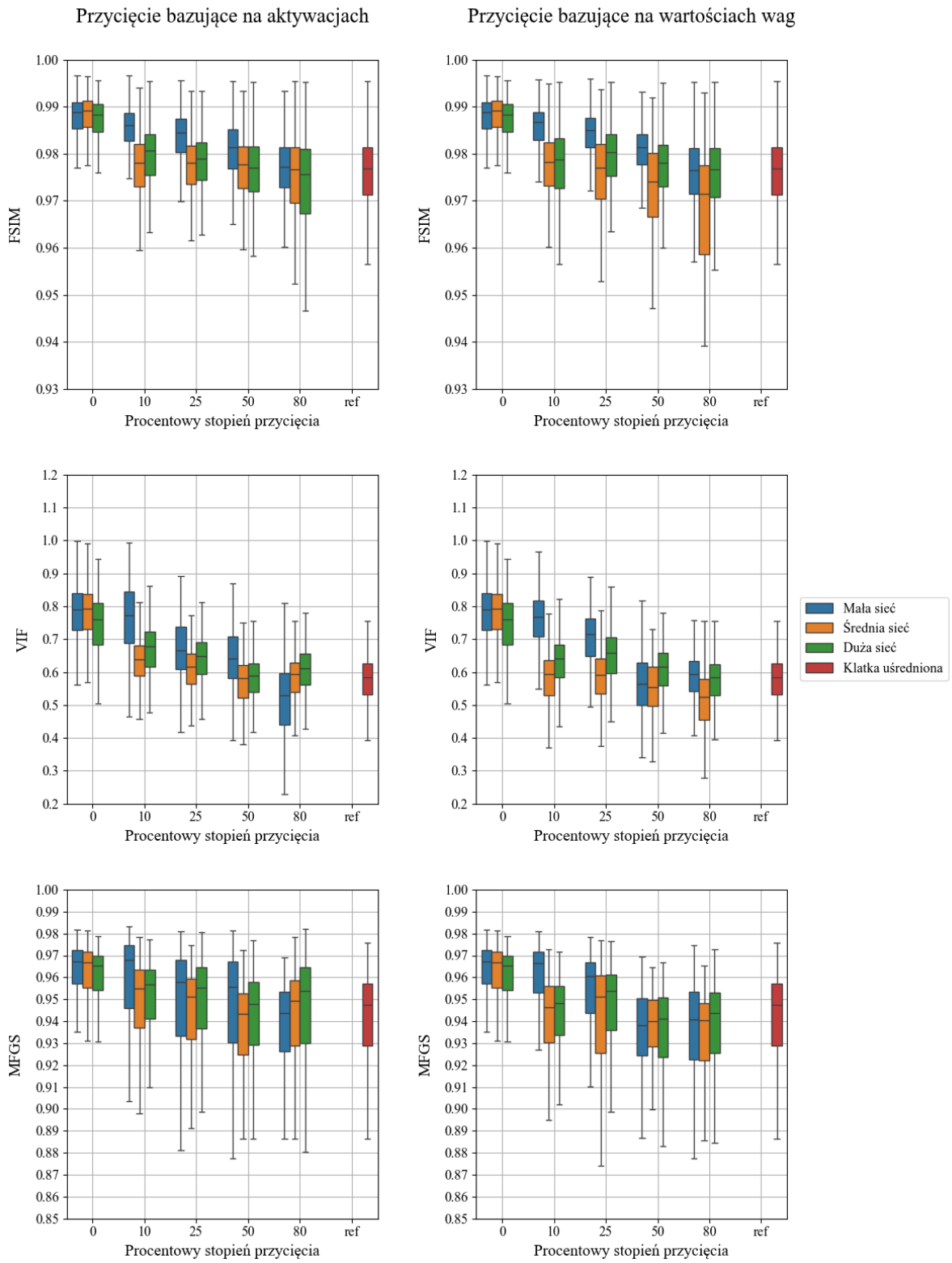
Każdą z rozważanych sieci wykorzystano do przetworzenia zbioru testowego, a obrazy wynikowe zostały ponownie porównane z $MFBD_{200}$. Do oceny ilościowej użyto omówionych wcześniej metryk, a także wykorzystano klatki uśrednione jako dodatkowy punkt odniesienia.

Powstałe w efekcie zestawienia wyników przedstawiono na wykresach 7.3 (dla 10 klatek) i 7.4 (dla 100 klatek). Lewe kolumny przedstawiają wyniki uzyskane dla przycinania opartego o wartości aktywacji, a prawe – opartego na wartościach wag.

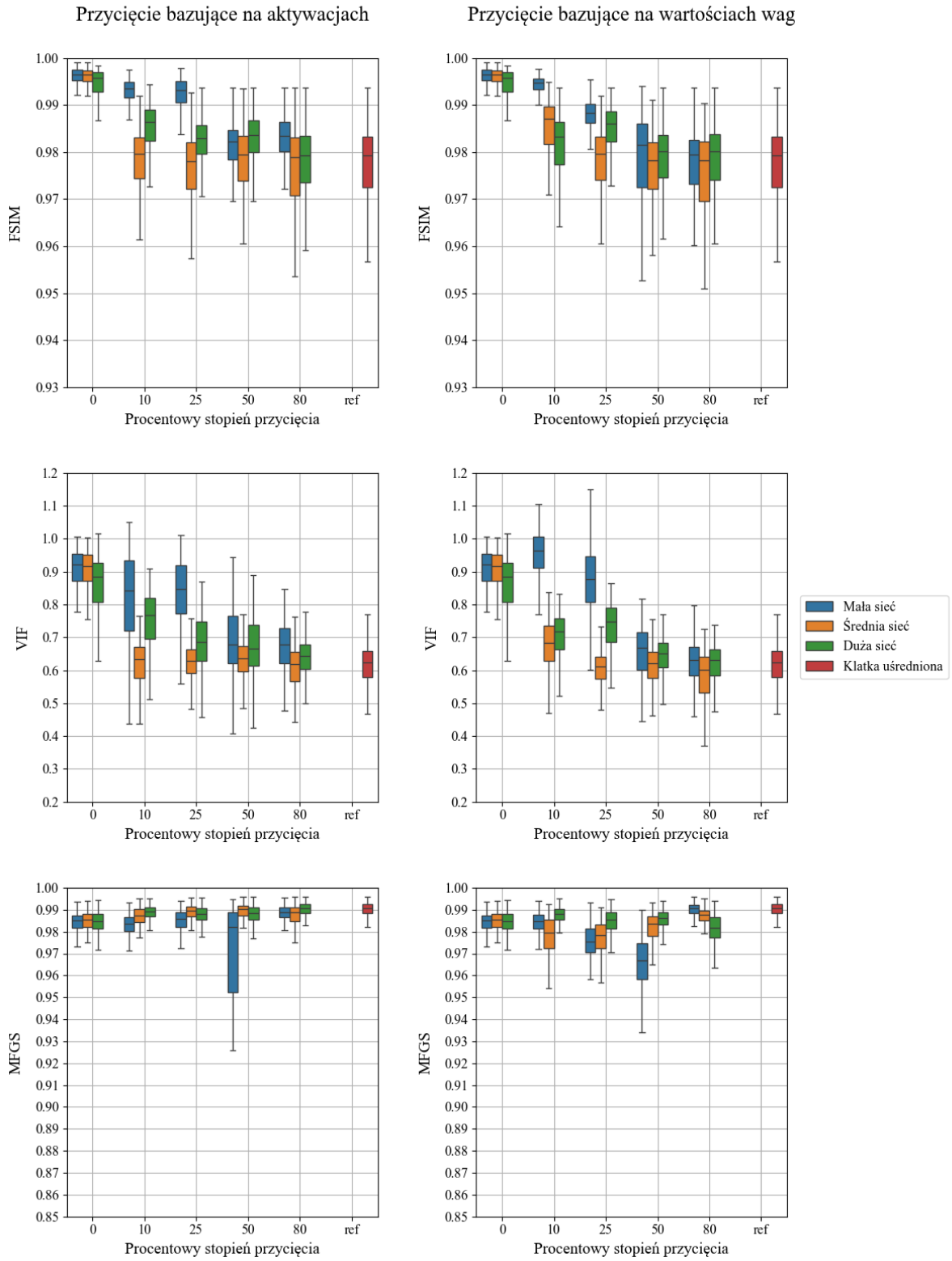
Pierwszym spostrzeżeniem jest to, że wybór kryterium usuwania elementów ma znikomy wpływ na kompresję. W obydwu przypadkach dostrzegalny jest podobny trend spadkowy, a występujące między tymi podejściami różnice sprowadzają się głównie do niewielkich zmian wydajności małego modelu. Natomiast porównując ze sobą wyniki modeli przetwarzających 10 i 100 klatek, można zauważyć, że więcej informacji zostaje zachowane w bardziej złożonej sieci.

Analizując wartości poszczególnych metryk, można stwierdzić, że już przy 10% przycięciu sieci radzą sobie gorzej z odtwarzaniem informacji, co widać na zestawieniu wyników dla metryk FSIM i MFGS. W przypadku miary VIF zmiany zachodzą nieco wolniej, niemniej sieci zaczynają zbliżać się jakością wyników do jakości obrazów uśrednionych. Świadczy to tym, że modele przestają działać poprawnie i nie realizują oczekiwanej rekonstrukcji obrazu.

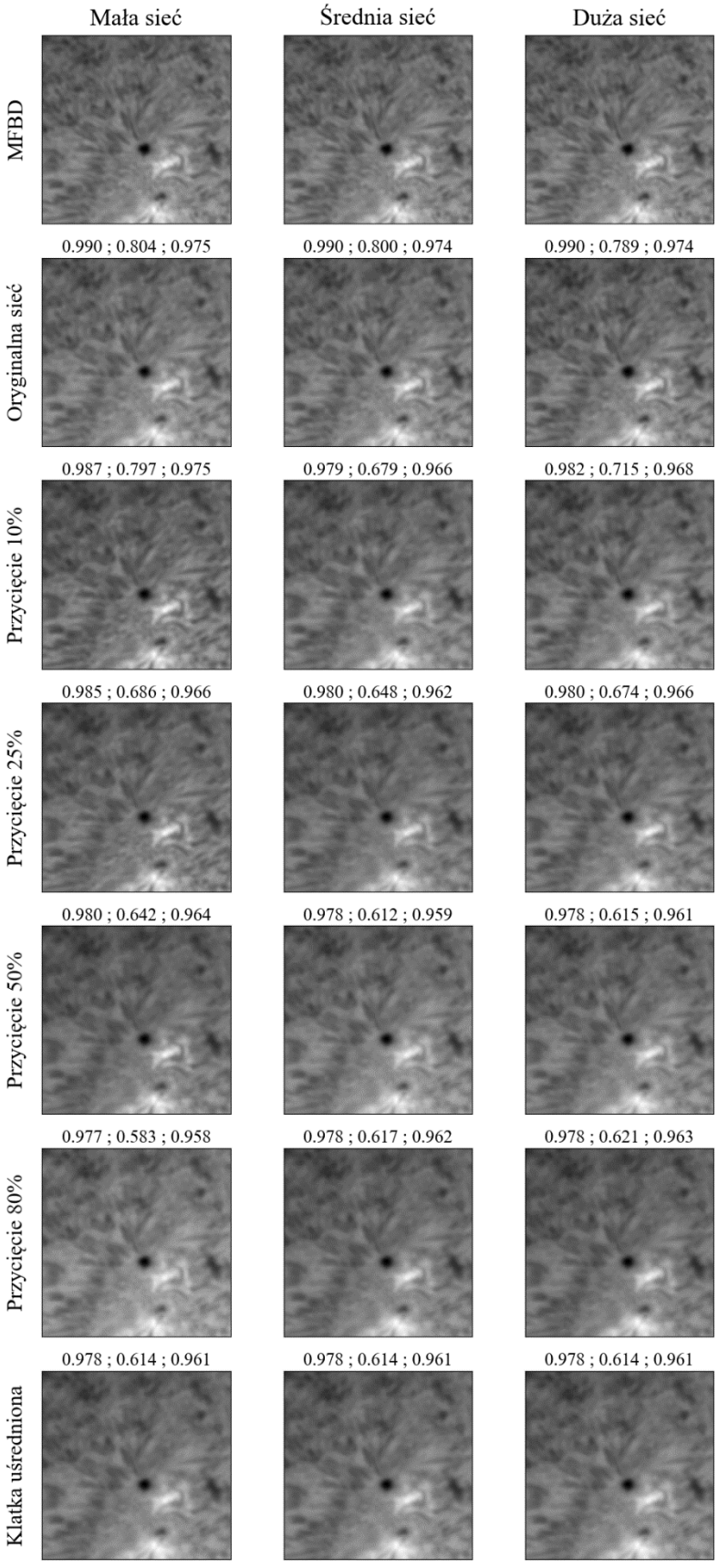
Aby lepiej ocenić to zagadnienie, opracowano zestawienie wyników działania poszczególnych algorytmów dla wybranej serii danych. Na rysunku 7.5 przedstawiono wyniki działania sieci wykorzystujących 10 klatek wejściowych, a na 7.6 analogiczne rezultaty uzyskane dla 100 klatek. Jak widać, oryginalne sieci są w stanie tak przetworzyć dane uśrednione, by otrzymać obrazy zbliżone do wyników MFBD. Wprowadzanie przycinania redukuje zmiany wprowadzane przez sieci w takim stopniu, że przy przycięciu 80% trudno odróżnić wynik działania sieci od danych wejściowych. Świadczy to o tym, że zmiany wprowadzane przez sieć mające odpowiednio uwydatnić (zrekonstruować) drobne struktury chromosfery słonecznej przestają mieć miejsce.



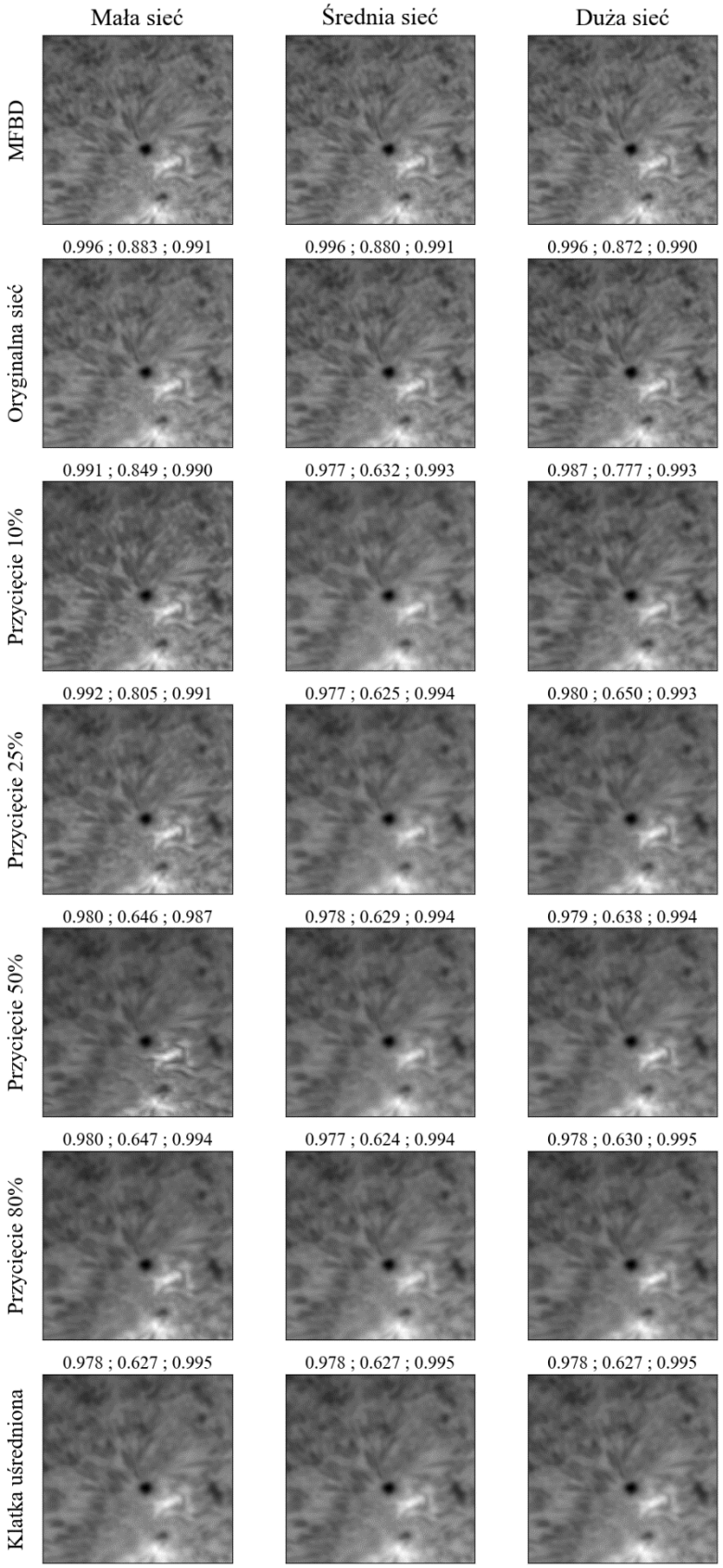
Rysunek 7.3. Statystyczne porównanie wybranych metryk po przycięciu sieci wykorzystujących 10 klatek wejściowych.



Rysunek 7.4. Statystyczne porównanie wybranych metryk po przycięciu sieci wykorzystujących 100 klatek wejściowych.



Rysunek 7.5. Porównanie wizualne obrazów po przycięciu sieci wykorzystujących 10 klatek wejściowych.



Rysunek 7.6. Porównanie wizualne obrazów po przycięciu sieci wykorzystujących 100 klatek wejściowych.

7.3.3. Uruchomienie na Raspberry Pi 5

Podczas dotychczasowych badań korzystano wyłącznie z komputera opisanego w poprzednim rozdziale. Chociaż w wielu sytuacjach wykorzystanie takiego sprzętu może okazać się wystarczające, pożądanym efektem prac jest uruchomienie testowanych sieci na znacznie mniej wymagającym sprzęcie. Taki sprzęt można podłączyć bezpośrednio do teleskopu, co pozwala przetwarzać dane od razu na miejscu obserwacji. Jest to zagadnienie szczególnie istotne przy akwizycji dużej ilości danych, które mogą zapełnić dostępną pamięć, zanim zostaną przeniesione do docelowego archiwum. Autorowi znany jest przykład takiej sytuacji z teleskopu kosmicznego Gaia, który w pewnym momencie rejestrował dane szybciej, niż je przesyłał na Ziemię, co wymusiło usunięcie pewnej ich części.

Postawiono zatem przetestować możliwości uruchomienia testowanych sieci na minikomputerze Raspberry Pi 5 wyposażonym w 8 GB RAM. Sprzęt ten obsługuje język Python wraz z frameworkiem PyTorch, co pozwoliło bezproblemowo przenieść na niego te rozwiązania. Należy przy tym podkreślić, że nie jest to norma, bo takie urządzenia wspierają zazwyczaj jedynie specjalistyczne formaty sieci, które wymuszają stosowanie konwersji. W przypadku PyTorcha największą elastyczność zapewnia konwersja modeli do formatu ONNX (ang. *Open Neural Network Exchange*).

W ramach badań na początku uruchomiono oryginalne sieci i przetestowano ich działanie na obrazach o wymiarach 176×176. Wyniki zaprezentowano w tabeli 7.4. Jak można zauważyć, nawet na sprzęcie tej klasy czas przetwarzania danych przez modele jest o wiele krótszy od czasu ich rejestrowania. Jedynym wyjątkiem jest duża sieć działająca na 10 klatkach, dla której czasy te są do siebie zbliżone.

Tabela 7.4. Czas przetwarzania danych przez sieci na Raspberry Pi 5 (176×176).

Liczba klatek wejściowych	**Czas przetwarzania danych [ms]**		
	Mała sieć	**Średnia sieć**	**Duża sieć**
10	17	67	324
20	19	70	327
50	47	76	339
100	75	87	357

Podjęto również próby uruchomienia tych sieci z wykorzystaniem obrazów o rozmiarze 2304×2304 piksele (tabela 7.5). Spośród dużych sieci, testom podlegały jedynie wersje skompresowane w 80% , bo inne wymagały zbyt dużo pamięci. Mimo że modele te powinny teoretycznie działać bez problemu, porównania można było dokonać jedynie dla sieci przetwarzające serie 10 i 20 klatek wejściowych. Jest to spowodowane tym, że spośród 8 GB pamięci na Raspberry Pi 5 część jest rezerwowana przez procesy systemowe, a także na wczytanie danych z pliku. Szybkość działania jest przy tym mniejsza niż w przypadku obrazów o wymiarach 176×176. Wartości te można jeszcze zmniejszyć poprzez wprowadzenie dalszych metod kompresji, takich jak kwantyzacja wag. Nie było to jednak celem pracy, więc eksperymenty zakończono na tym etapie.

Tabela 7.5. Czas przetwarzania danych przez sieci na Raspberry Pi 5 (2304×2304).

Liczba klatek wejściowych	**Czas przetwarzania danych [ms]**		
	Mała sieć (oryginalny rozmiar)	**Średnia sieć (oryginalny rozmiar)**	**Duża sieć (przycięta w 80%)**
10	3 438	12 170	9 615
20	3 767	12 919	10 220

7.4. Omówienie wyników

Do przeprowadzenia eksperymentów wykorzystano trzy sieci omówione w poprzednim rozdziale. Mimo że występują pomiędzy nimi znaczące różnice w złożoności, wszystkie są w stanie osiągnąć rezultaty zbliżone do wyników algorytmu MFBD. Sugeruje to, że ilość informacji zawarta w wykorzystanych danych jest wystarczająca nawet dla małej sieci. W konsekwencji większe architektury okazują się być zbędne, gdyż zawierają znacznie więcej parametrów, niż jest konieczne do osiągnięcia wysokiej wydajności. Najbardziej efektywnym podejściem jest zatem projektowanie mniejszych modeli, które są odpowiednio dopasowane do realizowanego zadania.

Z praktycznego punktu widzenia zadanie to nie jest proste, ponieważ trudno z góry oszacować optymalny stopień złożoności modeli. Pod tym względem bardziej bezpiecznym rozwiązaniem wydaje się być wstępne wytrenowanie dużej sieci neuronowej, a następnie

iteracyjne zmniejszanie jej wielkości połączone z obserwacją rezultatów. Podejście to wiąże się z potrzebą precyzyjnego dostrojenia algorytmów kompresji i ponownego treningu sieci. Jak jednak wykazano w eksperymencie, skompresowane modele w przypadku tego konkretnego zadania rekonstrukcji obrazów słonecznych osiągnęły rezultaty wyraźnie gorsze niż nieskompresowane, mniejsze modele.

W związku z tym, można stwierdzić, że w zadaniach operujących na danych niskorozdzielczych większe architektury nie zawsze przynoszą korzyści, a ich trenowanie oraz stosowana następnie kompresja mogą być skrajnie nieefektywne. Jednocześnie należy podkreślić, że wnioski te odnoszą się do konkretnego typu danych i rozważanego problemu, więc niekoniecznie muszą być uniwersalne. Niemniej, zaobserwowany fakt należy traktować jako przesłankę do każdorazowej próby poszukiwania alternatywnych, mniejszych, architektur sieci zamiast względnie bezpiecznego podejścia wykorzystującego kompresję większych struktur.

8. Podsumowanie pracy

W ostatnich latach astronomia obserwacyjna charakteryzuje się rosnącym zapotrzebowaniem na metody efektywnego przetwarzania rejestrowanych danych. Wiele uwagi poświęcono opracowaniu technik uczenia maszynowego służących poprawie ich ogólnej jakości i analizie. Większość prac skupiła się jednak wyłącznie na wykorzystaniu danych pochodzących z największych teleskopów naziemnych. Pominięto przy tym ich mniejsze odpowiedniki, które stanowią trzon wielu projektów badawczych. Celem niniejszej dysertacji było rozszerzenie badań na te zagadnienie poprzez ocenę możliwości sieci neuronowych w redukcji szumu niskorozdzielczych obrazów astronomicznych. Analizę tę podzielono na cztery oddzielne części, z których każda jest rozwinięciem poprzedniej.

Na początku określono grupę obiecujących rozwiązań w postaci sieci neuronowych, które przetwarzają dane, stosując przy tym redukcję ich wymiarowości. Założono, że taka procedura, dostosowana do narzuconych wymagań, powinna odpowiadać kompresji stratnej przetwarzanych danych, w przypadku której usunięta zostanie głównie informacja o losowym szumie. Jako testowane architektury wybrano zatem sieci typu autoenkoder, których wydajność zweryfikowano na zbiorze obrazów syntetycznych. Pozwoliło to ocenić wybrane techniki trenowania sieci w sytuacji, gdy nie ma dostępu do danych niezawierających szumu. Otrzymane wyniki zostały porównane względem rezultatów uzyskiwanych przez najlepsze algorytmy deterministyczne.

Przeprowadzone eksperymenty udowodniły wyższość rozwiązań bazujących na uczeniu maszynowym nad klasycznymi metodami, co wstępnie potwierdziło słuszność głównej tezy pracy. Wyniki wskazywały przy tym na możliwość skutecznego wykorzystania badanych strategii uczenia na obrazach rzeczywistych. Wnioski dotyczące tej części pracy skupiły się na wskazaniu ograniczenia rozważanych architektur, którym był ich brak elastyczności względem wejściowych danych. Pierwotnie struktura modeli pozwalała jedynie na przetwarzanie obrazów o stałych rozmiarach przestrzennych. By zaradzić temu problemowi, w dalszych etapach badań zdecydowano się wykorzystać wyłącznie sieci w pełni konwolucyjne.

Druga część badań skupiła się na walidacji wybranych technik uczenia i architektur sieci z użyciem danych obrazowych nocnego nieba. Analizowane modele wzbogacono o sieci typu U-Net, które są rozbudowaną formą symetrycznych autoenkoderów. Jakość przetworzonych obrazów została oceniona z wykorzystaniem pomiarów położenia, jasności i możliwości

detekcji gwiazd. Poruszono przy tym także tematykę wpływu złożoności sieci na jej efektywność.

Wykazano, że nawet relatywnie niewielkie sieci neuronowe są w stanie przewyższyć algorytmy deterministyczne w redukcji szumu nocnego nieba. Mimo że testowane rozwiązania mają tendencję do wyrównywania jasności tła, czyli przyciemniania słabiej świecących gwiazd, dobrze zachowują ich pozycję i zauważalnie zwiększają ich wykrywalność. Najistotniejszym wnioskiem było jednak to, że najlepsze wyniki nie zostały osiągnięte przez najbardziej złożone architektury, co może świadczyć o mniejszych wymaganiach danych niskorozdzielczych względem złożoności sieci. Tematykę tę rozwinięto w dalszej pracy.

Testowane do tego momentu sieci działały wyłącznie w oparciu o przetwarzanie pojedynczych obrazów. Jednak rozważane dane astronomiczne składają się zazwyczaj z serii wielu klatek wykonanych sekwencyjnie. W trzeciej części badań zweryfikowano możliwości sieci w redukcji szumu obecnego w takich seriach poprzez jednoczesne przetwarzanie informacji z wybranej liczby klatek. Celem było wyznaczenie odpowiedniej wielkości sieci, która pozwala uzyskać wyniki zbliżone do otrzymywanych przez najlepszy algorytm deterministyczny MFBD, lecz w znacznie krótszym czasie.

Jak zaobserwowano, wykorzystane architektury mogą stanowić atrakcyjną alternatywę wobec MFBD, ponieważ uzyskiwane wyniki cechują się porównywalną jakością, a obliczenia są wykonywane w czasie krótszym o rzędy wielkości. W przypadku wykorzystanego sprzętu można nawet stwierdzić, że obliczenia wykonywane są w czasie rzeczywistym – przetworzenie serii 100 klatek zachodzi w czasie krótszym od czasu ekspozycji pojedynczej klatki. Co istotne, również i w tym eksperymencie niewielkie sieci były w stanie osiągnąć bardzo dobry wynik, co ponownie potwierdza słuszność tezy.

Ostatnia część pracy skupia się na praktycznym podejściu do wykorzystania sieci. Jak przedstawiono, małe sieci mogą uzyskiwać lepsze wyniki od sieci większych, a ich trening jest zazwyczaj o wiele mniej wymagający sprzętowo. Niestety, nie da się z góry stwierdzić, jaki rozmiar sieci będzie najlepiej dopasowany do potrzeb. Bardziej przystępnym rozwiązaniem wydaje się być w takiej sytuacji wytrenowanie dużego modelu, a następnie zmniejszenie jego wielkości do odpowiedniego rozmiaru.

Wobec tego w ostatniej części pracy poruszono zagadnienie kompresji sieci neuronowych. Spośród wszystkich metod wybrano tę, która zmienia strukturę architektury, co pozwoliło ocenić wpływ wielkości modelu na wyniki działania. Jak zweryfikowano, podejście

to wymaga precyzyjnego dostrojenia algorytmów kompresji i ponownego treningu sieci, aby mogła ona konkurować z mniejszymi sieciami. Większe architektury mogą zatem okazać się w pewnych sytuacjach nieefektywne.

Podsumowując, wszystkie założone prace badawcze zostały z powodzeniem zrealizowane. Otrzymane wyniki wskazują, że niewielkie sieci neuronowe mogą osiągać wysoką skuteczność w redukcji szumu na obrazach astronomicznych, przewyższając pod tym względem zarówno algorytmy deterministyczne, jak i bardziej rozbudowane modele. Są przy tym w stanie wykonać tę pracę przy niższych wymaganiach sprzętowych i w krótszym czasie. Czyni to z nich sensowną grupę rozwiązań, a wyciągnięte z eksperymentów wnioski mogą posłużyć jako punkt wyjścia do przyszłych badań rozwijających ten temat. Na koniec należy zaznaczyć, że zastosowane rozwiązania nie były poddane pełnej optymalizacji, gdyż nie stanowiło to głównego celu badań.

Dodatek: Spis publikacji i wystąpień konferencyjnych

Niektóre z fragmentów niniejszej pracy zostały opublikowane w czasopismach naukowych oraz przedstawione w ramach wystąpień konferencyjnych. Poniżej zamieszczono chronologiczne zestawienie tych opracowań wraz z informacją o wykorzystanych fragmentach pracy.

1. Część wyników obejmująca wpływ różnych sposobów uczenia sieci neuronowych na ich efektywność na przykładzie zbioru MNIST została przedstawiona w ramach wystąpienia konferencyjnego na „Ogólnopolskiej Konferencji Młodych Naukowców na temat: Inżynieria – Spojrzenie Młodych Naukowców", a następnie opublikowana w formie artykułu „Wykorzystanie metod uczenia maszynowego w przetwarzaniu i poprawie jakości obrazów astronomicznych obserwatoriów Politechniki Śląskiej" [169] w opracowaniu zbiorczym „Zagadnienia aktualnie poruszane przez młodych naukowców".
2. Wyniki opisane w punkcie pierwszym zostały także w szerszej formie opublikowane jako artykuł „Comparison of training strategies for autoencoder-based monochromatic image denoising" [170] w czasopiśmie Sensors.
3. Wstępne wyniki prac nad wpływem doboru parametrów sieci neuronowej na jakość detekcji w przetworzonych obrazach zostały zaprezentowane na ogólnopolskiej konferencji „Analiza Zagadnienia, Analiza Wyników – Wystąpienie Młodego Naukowca – Edycja V".
4. Wyniki prac nad redukcją szumu astronomicznego na obrazach nocnego nieba i Słońca zostały w okrojonej formie zaprezentowane na „Górnośląskiej Kosmicznej Konferencji Naukowej GeKKoN", konferencji popularnonaukowej skierowanej do studentów i uczniów szkół średnich.
5. Część wyników prac nad zastosowaniem sieci neuronowych jako alternatywy metody MFBD w przetwarzaniu obrazów Słońca została przedstawiona w ramach referatu na międzynarodowej konferencji „The 38th Annual European Simulation and Modelling Conference (ESM 2024)", a następnie opublikowana w materiałach konferencyjnych jako „Increasing the observation capabilities of small solar telescopes using neural networks" [171].

6. Problematyka optymalizacji sieci neuronowych pod względem wykorzystania zasobów sprzętowych została poruszona w ramach referatu na „IX Międzynarodowej Interdyscyplinarnej Konferencji Uczelni Technicznych InterTechDOC'24".
7. Wyniki opisane w punkcie piątym zostały w poszerzonej formie (odpowiadającej większości Rozdziału 6) opublikowane jako artykuł „Fully convolutional neural networks for processing observational data from small remote solar telescopes" [172] w czasopiśmie Scientific Reports.
8. Wyniki prac nad porównaniem wpływu sieci neuronowych na detekcję gwiazd w przetworzonych obrazach nocnego nieba zostały przedstawione w ramach referatu na międzynarodowej konferencji „The 39th Annual European Simulation and Modelling Conference (ESM 2025)", a następnie opublikowane w materiałach konferencyjnych jako „Comparison of neural network training approaches in limited data scenarios" [173].
9. Obecnie trwają także prace nad poszerzeniem opisanych w rozdziale 5 eksperymentów na dane o różnym czasie ekspozycji, a następnie opracowanie artykułu naukowego, który zostanie zgłoszony do publikacji w czasopiśmie Astronomy & Astrophysics.

Bibliografia

[1] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4), 115-133

[2] McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (2006). A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. AI magazine, 27(4), 12-12

[3] Turing, A. M. (1950). Computing machinery and intelligence. Mind, 59(236), 433–460.

[4] Moore, G.E. (1965) Cramming More Components onto Integrated Circuits. Electronics, 38, 114-117

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

[6] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[8] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[9] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020.

[10] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., & Sutskever, I. (2021). Zero-shot text-to-image generation. arXiv preprint arXiv:2102.12092.

[11] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers) (pp. 4171-4186).

[12] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

[13] Nguyen, T. T., Tahir, H., Abdelrazek, M., & Babar, A. (2020). Deep learning methods for credit card fraud detection. arXiv preprint arXiv:2012.03754.

[14] Li, X., Luo, W., Yuan, M., Wang, J., Lu, J., Wang, J., ... & Zeng, J. (2021, April). Learning to optimize industry-scale dynamic pickup and delivery problems. In 2021 IEEE 37th international conference on data engineering (ICDE) (pp. 2511-2522). IEEE.

[15] Pantanowitz, L., Quiroga-Garza, G. M., Bien, L., Heled, R., Laifenfeld, D., Linhart, C., ... & Dhir, R. (2020). An artificial intelligence algorithm for prostate cancer diagnosis in whole slide images of core needle biopsies: a blinded clinical validation and deployment study. The Lancet Digital Health, 2(8), e407-e416.

[16] Vida, K., Bódi, A., Szklenár, T., & Seli, B. (2021). Finding flares in Kepler and TESS data with recurrent deep neural networks. Astronomy & Astrophysics, 652, A107.

[17] Grishin, K., Mei, S., & Ilić, S. (2023). YOLO–CL: Galaxy cluster detection in the SDSS with deep machine learning. Astronomy & Astrophysics, 677, A101.

[18] Sun, R., Jia, P., Sun, Y., Yang, Z., Liu, Q., & Wei, H. (2023). PNet—A Deep Learning Based Photometry and Astrometry Bayesian Framework. The Astronomical Journal, 166(6), 235.

[19] Gómez, C., Neira, M., Hernández Hoyos, M., Arbeláez, P., & Forero-Romero, J. E. (2020). Classifying image sequences of astronomical transients with deep neural networks. Monthly Notices of the Royal Astronomical Society, 499(3), 3130-3138.

[20] Bairouk, A., Chaumont, M., Fouchez, D., Paquet, J., Comby, F., & Bautista, J. (2023). Astronomical image time series classification using CONVolutional attENTION (ConvEntion). Astronomy & Astrophysics, 673, A141.

[21] Chaini, S., Bagul, A., Deshpande, A., Gondkar, R., Sharma, K., Vivek, M., & Kembhavi, A. (2023). Photometric identification of compact galaxies, stars, and quasars using multiple neural networks. Monthly Notices of the Royal Astronomical Society, 518(2), 3123-3136.

[22] Baso, C. D., de la Cruz Rodriguez, J., & Danilovic, S. (2019). Solar image denoising with convolutional neural networks. Astronomy & Astrophysics, 629, A99.

[23] Park, E., Moon, Y. J., Lim, D., & Lee, H. (2020). De-noising SDO/HMI solar magnetograms by image translation method based on deep learning. The Astrophysical Journal Letters, 891(1), L4.

[24] Vojtekova, A., Lieu, M., Valtchanov, I., Altieri, B., Old, L., Chen, Q., & Hroch, F. (2021). Learning to denoise astronomical images with U-nets. Monthly Notices of the Royal Astronomical Society, 503(3), 3204-3215.

[25] Navarro, F., Hall, D., Budavari, T., & Sukurdeep, Y. (2023). Learning the night sky with deep generative priors. arXiv preprint arXiv:2302.02030.

[26] York, D. G., Adelman, J., Anderson Jr, J. E., Anderson, S. F., Annis, J., Bahcall, N. A., ... & Yasuda, N. (2000). The sloan digital sky survey: Technical summary. The Astronomical Journal, 120(3), 1579.

[27] Ricker, G. R., Winn, J. N., Vanderspek, R., Latham, D. W., Bakos, G. Á., Bean, J. L., ... & Villasenor, J. (2015). Transiting exoplanet survey satellite. Journal of Astronomical Telescopes, Instruments, and Systems, 1(1), 014003-014003.

[28] Masci, F. J., Laher, R. R., Rusholme, B., Shupe, D. L., Groom, S., Surace, J., ... & Kulkarni, S. R. (2018). The zwicky transient facility: Data processing, products, and archive. Publications of the Astronomical Society of the Pacific, 131(995), 018003.

[29] Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. Neural computation, 8(7), 1341-1390.

[30] Steinegger, M., Hanslmeier, A., Otruba, W., Freislich, H., Denker, C., Goode, P. R., ... & Zhang, Q. (2000). An overview of the new global high-resolution H-alpha network. Hvar Observatory Bulletin, Volume 24, Issue 1, p. 179, 24, 179.

[31] Harvey, J. W., Hill, F., Hubbard, R. P., Kennedy, J. R., Leibacher, J. W., Pintar, J. A., ... & Yasukawa, E. (1996). The global oscillation network group (GONG) project. Science, 272(5266), 1284-1286.

[32] Kokori, A., Tsiaras, A., Edwards, B., Jones, A., Pantelidou, G., Tinetti, G., ... & Naves, R. (2023). ExoClock Project. III. 450 new exoplanet ephemerides from ground and space observations. The Astrophysical Journal Supplement Series, 265(1), 4.

[33] Babcock, H. W. (1953). The possibility of compensating astronomical seeing. Publications of the Astronomical Society of the Pacific, 65(386), 229-236.

[34] Schmidt, D., Gorceix, N., Goode, P. R., Marino, J., Rimmele, T., Berkefeld, T., ... & Von Der Lühe, O. (2017). Clear widens the field for observations of the Sun with multi-conjugate adaptive optics. Astronomy & Astrophysics, 597, L8.

[35] Rao, C., Gu, N., Rao, X., Li, C., Zhang, L., Huang, J., ... & Ma, W. (2020). First light of the 1.8-m solar telescope–CLST. Science China. Physics, Mechanics & Astronomy, 63(10), 109631.

[36] Light Pollution Think Tank (2023). Zanieczyszczenie światłem w Polsce. Raport 2023. Kotarba, A. Z. (red.) Wydawnictwo Centrum Badań Kosmicznych PAN, Warszawa, www.lptt.org.pl

[37] Guidash, M., Ma, J., Vogelsang, T., & Endsley, J. (2016). Reduction of CMOS image sensor read noise to enable photon counting. Sensors, 16(4), 517.

[38] Hennel J. (2003). Podstawy elektroniki półprzewodnikowej. Wydawnictwo Naukowo-Techniczne.

[39] Kitamura, J., Katsuma, H., & Nishimura, T. (2009). A Reduction Method of Photon Shot Noise on CMOS Image Sensor Based on Local Brightness Distribution. IEEJ Transactions on Electronics, Information and Systems, 129(6), 1147-1155.

[40] McLean, I. S. (2008). Electronic imaging in astronomy: detectors and instrumentation. Berlin, Heidelberg: Springer Berlin Heidelberg.

[41] Scharmer, G. B., Bjelksjo, K., Korhonen, T. K., Lindberg, B., & Petterson, B. (2003, February). The 1-m Swedish solar telescope. In Innovative telescopes and instrumentation for solar astrophysics (Vol. 4853, pp. 341-350). SPIE.

[42] Feigenbaum, E. A., Buchanan, B. G., & Lederberg, J. (1970). On generality and problem solving: A case study using the DENDRAL program (No. NASA-CR-123182).

[43] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

[44] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

[45] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE transactions on information theory, 13(1), 21-27.

[46] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. nature, 323(6088), 533-536.

[47] Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin philosophical magazine and journal of science, 2(11), 559-572.

[48] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6), 386.

[49] Minsky, M., & Papert, S. (1969). An introduction to computational geometry. Cambridge tiass., HIT, 479(480), 104.

[50] Hebb, D. O. (1949). The organization of behavior. A neuropsychological theory. Wiley.

[51] Linnainmaa, S. (1970). The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors (Praca magisterska, Univ. Helsinki).

[52] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation (No. ICS8506).

[53] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32.

[54] Frostig, R., Johnson, M. J., & Leary, C. (2019, March). Compiling machine learning programs via high-level tracing. In SysML conference 2018.

[55] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Zheng, X. (2016). {TensorFlow}: a system for {Large-Scale} machine learning. In 12th USENIX symposium on operating systems design and implementation (OSDI 16) (pp. 265-283).

[56] Odena, A., Dumoulin, V., & Olah, C. (2016). Deconvolution and checkerboard artifacts. Distill, 1(10), e3.

[57] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[58] Glorot, X., & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256). JMLR Workshop and Conference Proceedings.

[59] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In Proc. icml (Vol. 30, No. 1, p. 3).

[60] Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853.

[61] Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289, 4(5), 11.

[62] Klambauer, G., Unterthiner, T., Mayr, A., & Hochreiter, S. (2017). Self-normalizing neural networks. Advances in neural information processing systems, 30.

[63] Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415.

[64] Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941.

[65] Misra, D. (2019). Mish: A self regularized non-monotonic activation function. arXiv preprint arXiv:1908.08681.

[66] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning (pp. 448-456). pmlr.

[67] Wu, Y., & He, K. (2018). Group normalization. In Proceedings of the European conference on computer vision (ECCV) (pp. 3-19).

[68] Jóźwik-Wabik, P. (2021). Batch effect identification methods in high-throughput omics experiments (Praca magisterska, Politechnika Śląska).

[69] Jóźwik-Wabik, P., Gładys, B., Hermansa, M., Macha, D., Kalisz, S., Strzoda, T., Foszner, P., Popowicz, A., & Marczyk, M. (2021). Removing compression artifacts on whole slide HE-stained histopathological images. In S. Bajkacz & Z. Ostrowski (Editors), Recent advances in computational oncology and personalized medicine. Silesian University of Technology.

[70] Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008, July). Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning (pp. 1096-1103).

[71] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. A., & Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of machine learning research, 11(12).

[72] Xie, J., Xu, L., & Chen, E. (2012). Image denoising and inpainting with deep neural networks. Advances in neural information processing systems, 25.

[73] Gondara, L. (2016, December). Medical image denoising using convolutional denoising autoencoders. In 2016 IEEE 16th international conference on data mining workshops (ICDMW) (pp. 241-246). IEEE.

[74] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Cham: Springer international publishing.

[75] Heinrich, M. P., Stille, M., & Buzug, T. M. (2018). Residual U-net convolutional neural network architecture for low-dose CT denoising. Current Directions in Biomedical Engineering, 4(1), 297-300.

[76] Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502.

[77] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4), 303-314.

[78] Bengio, Y. (2009). Learning deep architectures for AI. Foundations and trends® in Machine Learning, 2(1), 1-127.

[79] Kingma, D. P. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[80] Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101.

[81] Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.

[82] Smith, L. N. (2017, March). Cyclical learning rates for training neural networks. In 2017 IEEE winter conference on applications of computer vision (WACV) (pp. 464-472

[83] Zhao, H., Gallo, O., Frosio, I. & Kautz, J. (2016). Loss functions for image restoration with neural networks. IEEE Transactions on Computational Imaging, 3, 47–57.

[84] Huber, P. J. (1964). Robust Estimation of a Location Parameter. Ann. Math. Statist., 35(4), 73-101.

[85] Jiang, L., Dai, B., Wu, W., & Loy, C. C. (2021). Focal frequency loss for image reconstruction and synthesis. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 13919-13929).

[86] Miseta, T., Fodor, A., & Vathy-Fogarassy, Á. (2024). Surpassing early stopping: A novel correlation-based stopping criterion for neural networks. Neurocomputing, 567, 127028.

[87] Masters, D., & Luschi, C. (2018). Revisiting small batch training for deep neural networks. arXiv preprint arXiv:1804.07612.

[88] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (2002). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[89] Lee, J. S. (1981). Refined filtering of image noise using local statistics. Computer graphics and image processing, 15(4), 380-389.

[90] Yang, R., Yin, L., Gabbouj, M., Astola, J., & Neuvo, Y. (1995). Optimal weighted median filtering under structural constraints. IEEE transactions on signal processing, 43(3), 591-604.

[91] Benesty, J., Chen, J., & Huang, Y. (2010, March). Study of the widely linear Wiener filter for noise reduction. In 2010 IEEE international conference on acoustics, speech and signal processing (pp. 205-208). IEEE.

[92] Tomasi, C., & Manduchi, R. (1998, January). Bilateral filtering for gray and color images. In Sixth international conference on computer vision (IEEE Cat. No. 98CH36271) (pp. 839-846). IEEE.

[93] Buades, A., Coll, B., & Morel, J. M. (2005, June). A non-local algorithm for image denoising. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) (Vol. 2, pp. 60-65). Ieee.

[94] Wang, J., Guo, Y., Ying, Y., Liu, Y., & Peng, Q. (2006, October). Fast non-local algorithm for image denoising. In 2006 International Conference on Image Processing (pp. 1429-1432). IEEE.

[95] Thaipanich, T., Oh, B. T., Wu, P. H., Xu, D., & Kuo, C. C. J. (2010). Improved image denoising with adaptive nonlocal means (ANL-means) algorithm. IEEE Transactions on Consumer Electronics, 56(4), 2623-2630.

[96] Sutour, C., Deledalle, C. A., & Aujol, J. F. (2014). Adaptive regularization of the NL-means: Application to image and video denoising. IEEE Transactions on image processing, 23(8), 3506-3521.

[97] Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007). Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Transactions on image processing, 16(8), 2080-2095.

[98] Aharon, M., Elad, M., & Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on signal processing, 54(11), 4311-4322/

[99] Muresan, D. D., & Parks, T. W. (2003, September). Adaptive principal components and image denoising. In Proceedings 2003 international conference on image processing (Cat. No. 03CH37429) (Vol. 1, pp. I-101). IEEE.

[100] Portilla, J., Strela, V., Wainwright, M. J., & Simoncelli, E. P. (2003). Image denoising using scale mixtures of Gaussians in the wavelet domain. IEEE Transactions on Image processing, 12(11), 1338-1351.

[101] Zoran, D., & Weiss, Y. (2011, November). From learning models of natural image patches to whole image restoration. In 2011 international conference on computer vision (pp. 479-486). IEEE.

[102] Elad, M., & Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image processing, 15(12), 3736-3745.

[103] Moldovan, T. M., Roth, S., & Black, M. J. (2006, October). Denoising archival films using a learned Bayesian model. In 2006 International Conference on Image Processing (pp. 2641-2644). IEEE.

[104] Plotz, T., & Roth, S. (2017). Benchmarking denoising algorithms with real photographs. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1586-1595).

[105] Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., & Aila, T. (2018). Noise2Noise: Learning image restoration without clean data. arXiv preprint arXiv:1803.04189.

[106] Krull, A., Buchholz, T. O., & Jug, F. (2019). Noise2void-learning denoising from single noisy images. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 2129-2137.

[107] Krull, A., Vičar, T., Prakash, M., Lalit, M., & Jug, F. (2020). Probabilistic noise2void: Unsupervised content-aware denoising. Frontiers in Computer Science, 2, 5.

[108] Quan, Y., Chen, M., Pang, T., & Ji, H. (2020). Self2self with dropout: Learning self-supervised denoising from single image. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 1890-1898).

[109] Batson, J., & Royer, L. (2019, May). Noise2self: Blind denoising by self-supervision. In International conference on machine learning (pp. 524-533). PMLR.

[110] Xu, J., Huang, Y., Cheng, M. M., Liu, L., Zhu, F., Xu, Z., & Shao, L. (2020). Noisy-as-clean: Learning self-supervised denoising from corrupted image. IEEE Transactions on Image Processing, 29, 9316-9329.

[111] Moran, N., Schmidt, D., Zhong, Y., & Coady, P. (2020). Noisier2noise: Learning to denoise from unpaired noisy data. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 12064-12072).

[112] Bonneel, N., Rabin, J., Peyré, G., & Pfister, H. (2015). Sliced and radon wasserstein barycenters of measures. Journal of Mathematical Imaging and Vision, 51(1), 22-45.

[113] Deshpande, I., Hu, Y. T., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., ... & Schwing, A. G. (2019). Max-sliced wasserstein distance and its use for gans. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10648-10656).

[114] Deshpande, I., Zhang, Z., & Schwing, A. G. (2018). Generative modeling using the sliced wasserstein distance. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3483-3491).

[115] Kolouri, S., Pope, P. E., Martin, C. E., & Rohde, G. K. (2019, May). Sliced Wasserstein Auto-Encoders. In ICLR (Poster).

[116] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4), 600-612.

[117] Hore, A., & Ziou, D. (2010, August). Image quality metrics: PSNR vs. SSIM. In 2010 20th international conference on pattern recognition (pp. 2366-2369). IEEE.

[118] Zhang, L., Zhang, L., Mou, X., & Zhang, D. (2011). FSIM: A feature similarity index for image quality assessment. IEEE transactions on Image Processing, 20(8), 2378-2386.

[119] Stetson, P. B. (1987). DAOPHOT: A computer program for crowded-field stellar photometry. Publications of the Astronomical Society of the Pacific, 99(613), 191.

[120] Bradley, L., Sipocz, B., Robitaille, T., Tollerud, E., Deil, C., Vinícius, Z., ... & Weaver, B. A. (2016). Photutils: Photometry tools. Astrophysics Source Code Library, ascl-1609

[121] Jia, P., Zheng, Y., Wang, M., & Yang, Z. (2023). A deep learning based astronomical target detection framework for multi-colour photometry sky survey projects. Astronomy and Computing, 42, 100687.

[122] Parisot, O., & Jaziri, M. (2024). Deep sky objects detection with deep learning for electronically assisted astronomy. Astronomy, 3(2), 122-138.

[123] Jia, P., Liu, Q., & Sun, Y. (2020). Detection and classification of astronomical targets with deep neural networks in wide-field small aperture telescopes. The Astronomical Journal, 159(5), 212.

[124] Shi, J. H., Qiu, B., Luo, A. L., He, Z. D., Kong, X., & Jiang, X. (2022). A photometry pipeline for SDSS images based on convolutional neural networks. Monthly Notices of the Royal Astronomical Society, 516(1), 264-278.

[125] Sun, R., Jia, P., Sun, Y., Yang, Z., Liu, Q., & Wei, H. (2023). PNet—A Deep Learning Based Photometry and Astrometry Bayesian Framework. The Astronomical Journal, 166(6), 235.

[126] Yang, Z., Liu, M., Yuan, H., Bu, Y., Yi, Z., Kong, X., ... & Zhang, R. (2023). Star Photometry for DECam Legacy Survey and Sloan Digital Sky Survey Images Based on Convolutional Neural Networks. The Astronomical Journal, 166(5), 210.

[127] Vojtekova, A., Lieu, M., Valtchanov, I., Altieri, B., Old, L., Chen, Q., & Hroch, F. (2021). Learning to denoise astronomical images with U-nets. Monthly Notices of the Royal Astronomical Society, 503(3), 3204-3215.

[128] Elhakiem, A. A., Ghoniemy, T. E., & Salama, G. I. (2021, December). Astronomical image denoising based on Convolutional Neural Network. In 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS) (pp. 51-56). IEEE.

[129] Bernardi, R. L., Berdja, A., Guzmán, C. D., Torres-Torriti, M., & Roth, M. M. (2022). Restoration of images with a spatially varying PSF of the T80-S telescope optical model using neural networks. Monthly Notices of the Royal Astronomical Society, 510(3), 4284-4294.

[130] Li, Y., Niu, Z., Sun, Q., Xiao, H., & Li, H. (2022). Bsc-net: background suppression algorithm for stray lights in star images. Remote Sensing, 14(19), 4852.

[131] Zhang, Y., Nord, B., Pagul, A., & Lepori, M. (2022). Noise2astro: Astronomical image denoising with self-supervised neural networks. Research Notes of the AAS, 6(9), 187.

[132] Bernardi, R. L., Berdja, A., Guzmán, C. D., Torres-Torriti, M., & Roth, M. M. (2023). Restoration of T80-S telescope's images using neural networks. Monthly Notices of the Royal Astronomical Society, 524(2), 3068-3082.

[133] Jia, P., Li, X., Li, Z., Wang, W., & Cai, D. (2020). Point spread function modelling for wide-field small-aperture telescopes with a denoising autoencoder. Monthly Notices of the Royal Astronomical Society, 493(1), 651-660.

[134] Bartlett, O. J., Benoit, D. M., Pimbblet, K. A., Simmons, B., & Hunt, L. (2023). Noise reduction in single-shot images using an auto-encoder. Monthly Notices of the Royal Astronomical Society, 521(4), 6318-6329.

[135] Jia, P., Wu, X., Li, Z., Li, B., Wang, W., Liu, Q., ... & Cai, D. (2021). Point spread function estimation for wide field small aperture telescopes with deep neural networks and calibration data. Monthly Notices of the Royal Astronomical Society, 505(4), 4717-4725.

[136] Liu, T., Quan, Y., Su, Y., Guo, Y., Liu, S., Ji, H., Hao, Q., Gao, Y., Liu, Y., Wang, Y., Sun, W., & Ding, M. L. (2025). Astronomical image denoising by self-supervised deep learning and restoration processes. Nature Astronomy, 9(2), 123–135.

[137] Rodriguez-Alvarez, N., Jao, J. S., Munoz-Martin, J. F., Lee, C. G., & Oudrhiri, K. (2022). Feed-forward neural network denoising applied to goldstone solar system radar images. Remote Sensing, 14(7), 1643.

[138] Labeyrie, A. (1970). Attainment of diffraction limited resolution in large telescopes by Fourier analysing speckle patterns in star images. Astronomy and Astrophysics, Vol. 6, p. 85 (1970), 6, 85.

[139] Von der Lühe, O. (1993). Speckle imaging of solar small scale structure. I-Methods. Astronomy and Astrophysics (ISSN 0004-6361), vol. 268, no. 1, p. 374-390., 268, 374-390.

[140] Löfdahl, M. G. (2002, December). Multiframe blind deconvolution with linear equality constraints. In Image reconstruction from incomplete data II (Vol. 4792, pp. 146-155). SPIE.

[141] Van Noort, M., Der Voort, L. R. V., & Löfdahl, M. G. (2005). Solar image restoration by use of multi-frame blind de-convolution with multiple objects and phase diversity. Solar Physics, 228(1), 191-215.

[142] Löfdahl, M. G., & Hillberg, T. (2022). Multi-frame blind deconvolution and phase diversity with statistical inclusion of uncorrected high-order modes. Astronomy & Astrophysics, 668, A129.

[143] Ramos, A. A., de la Cruz Rodríguez, J., & Yabar, A. P. (2018). Real-time, multiframe, blind deconvolution of solar images. Astronomy & Astrophysics, 620, A73.

[144] Baso, C. D., de la Cruz Rodriguez, J., & Danilovic, S. (2019). Solar image denoising with convolutional neural networks. Astronomy & Astrophysics, 629, A99.

[145] Ramos, A. A., & Olspert, N. (2021). Learning to do multiframe wavefront sensing unsupervised: Applications to blind deconvolution. Astronomy & Astrophysics, 646, A100.

[146] Asensio Ramos, A., Esteban Pozuelo, S., & Kuckein, C. (2023). Accelerating multiframe blind deconvolution via deep learning. Solar Physics, 298(7), 91.

[147] Wang, S., Chen, Q., He, C., Zhang, C., Zhong, L., Bao, H., & Rao, C. (2021). Blind restoration of solar images via the Channel Sharing Spatio-temporal Network. Astronomy & Astrophysics, 652, A50.

[148] Zhang, C., Wang, S., Zhong, L., Chen, Q., & Rao, C. (2023). Cascaded Temporal and Spatial Attention Network for solar adaptive optics image restoration. Astronomy & Astrophysics, 674, A126.

[149] Armstrong, J. A., & Fletcher, L. (2021). A machine-learning approach to correcting atmospheric seeing in solar flare observations. Monthly Notices of the Royal Astronomical Society, 501(2), 2647-2658.

[150] Shu, J., Xie, C., & Gao, Z. (2022). Blind restoration of atmospheric turbulence-degraded images based on curriculum learning. Remote Sensing, 14(19), 4797.

[151] Cai, Z., Zhong, Z., & Zhang, B. (2023). High-resolution restoration of solar images degraded by atmospheric turbulence effect using improved CycleGAN. New Astronomy, 101, 102018.

[152] Popowicz, A., & Orlov, V. (2022). Suto-solar through-turbulence open image dataset. Sensors, 22(20), 7902.

[153] Popowicz, A., Swoboda, F., Lasota, S., Fiolka, J., Orlov, V., Bernacki, K., & Rudawy, P. (2023). Solar patrol observatory at the Silesian University of Technology. Journal of Astronomical Telescopes, Instruments, and Systems, 9(2), 027001-027001.

[154] Sheikh, H. R., & Bovik, A. C. (2006). Image information and visual quality. IEEE Transactions on image processing, 15(2), 430-444.

[155] Deng, H., Zhang, D., Wang, T., Ji, K., Wang, F., Liu, Z., ... & Cao, W. (2015). Objective image-quality assessment for high-resolution photospheric images by median filter-gradient similarity. Solar Physics, 290(5), 1479-1489.

[156] Popowicz, A., Radlak, K., Bernacki, K., & Orlov, V. (2017). Review of image quality measures for solar imaging. Solar Physics, 292(12), 187.

[157] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision (pp. 1026-1034).

[158] LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

[159] Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

[160] Manessi, F., Rozza, A., Bianco, S., Napoletano, P., & Schettini, R. (2018, August). Automated pruning for deep neural network compression. In 2018 24th International conference on pattern recognition (ICPR) (pp. 657-664). IEEE.

[161] Frankle, J., & Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv preprint arXiv:1803.03635.

[162] Malach, E., Yehudai, G., Shalev-Schwartz, S., & Shamir, O. (2020, November). Proving the lottery ticket hypothesis: Pruning is all you need. In International Conference on Machine Learning (pp. 6682-6691). PMLR.

[163] Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. International journal of computer vision, 129(6), 1789-1819.

[164] Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A. A., & Wilson, A. G. (2021). Does knowledge distillation really work?. Advances in neural information processing systems, 34, 6906-6919.

[165] Kim, J., Chang, S., & Kwak, N. (2021). PQK: model compression via pruning, quantization, and knowledge distillation. arXiv preprint arXiv:2106.14681.

[166] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149.

[167] Hu, P., Peng, X., Zhu, H., Aly, M. M. S., & Lin, J. (2021, May). Opq: Compressing deep neural networks with one-shot pruning-quantization. In Proceedings of the AAAI conference on artificial intelligence (Vol. 35, No. 9, pp. 7780-7788).

[168] Kuzmin, A., Nagel, M., Van Baalen, M., Behboodi, A., & Blankevoort, T. (2023). Pruning vs quantization: Which is better?. Advances in neural information processing systems, 36, 62414-62427.

[169] Jóźwik-Wabik, P. (2022). Wykorzystanie metod uczenia maszynowego w przetwarzaniu i poprawie jakości obrazów astronomicznych obserwatoriów Politechniki Śląskiej. In K. Piech (editor), Zagadnienia aktualnie poruszane przez młodych naukowców 21. [Dokument elektroniczny]. Creativetime.

[170] Jóźwik-Wabik, P., Bernacki, K., & Popowicz, A. (2023). Comparison of training strategies for autoencoder-based monochromatic image denoising. Sensors, 23, Article 12.

[171] Jóźwik-Wabik, P., & Popowicz, A. (2024). Increasing the observation capabilities of small solar telescopes using neural networks. In M. Graña & J. D. Nuñez-Gonzalez (Editors), Modelling and simulation 2024. The 2024 European Simulation and Modelling Conference.

[172] Jóźwik-Wabik, P., & Popowicz, A. (2025). Fully convolutional neural networks for processing observational data from small remote solar telescopes. Scientific Reports, 15, Article 1.

[173] Jóźwik-Wabik, P., & Popowicz, A. (2025). Comparison of neural network training approaches in limited data scenarios. In S. S. Bhonsale, M. E. Polanska, & J. F. M. van Impe (Editors), Modelling and simulation 2025. The 2025 European Simulation and Modelling Conference. ESM′2025, October 22-24, 2025, Ghent, Belgium.

Spis oznaczeń i skrótów

AE	sieć neuronowa typu autoenkoder	[str. 53]
AUC	pole pod krzywą (ang. *Area Under the Curve*)	[str. 93]
B	liczba próbek w minigrupie	[str. 41]
BM3D	najlepszy algorytm deterministyczny będący rozwinięciem NLM; punkt odniesienia przy ocenie działania testowanych sieci (ang. *Block-Matching and 3D Filtering*)	[str. 65]
b	człon obciążenia/przesunięcia (ang. *bias*)	[str. 36]
C	liczba kanałów (map cech) obrazu	[str. 41]
C_{in}	liczba map cech wejściowych warstwy	[str. 44]
C_{out}	liczba map cech wyjściowych warstwy	[str. 44]
CCD	rodzaj matryc wykorzystywanych w kamerach (ang. *Charge-Coupled Device*)	[str. 22]
CMOS	rodzaj matryc wykorzystywanych w kamerach (ang. *Complementary Metal-Oxide-Semiconductor*)	[str. 22]
CPU	jednostka centralna, procesor (ang. *Central Processing Unit*)	[str. 131]
D	średnica apertury	[str. 24]
DAOStarFinder	zastosowany w pracy algorytm detekcji gwiazd	[str. 90]
f	natężenie światła zmierzone dla badanego obiektu	[str. 95]
f_{ogn}	długość ogniskowej teleskopu	[str. 25]
FOV	pole widzenia; najczęściej określane osobno w dwóch wymiarach (ang. *Field of View*)	[str. 25]
FCN	w pełni konwolucyjna sieć pozbawiona warstw gęstych (ang. *Fully Convolutional Network*)	[str. 84]
FN	oznaczenie klasyfikacji fałszywie negatywnej (ang. *False Negative*)	[str. 92]
FP	oznaczenie klasyfikacji fałszywie pozytywnej (ang. *False Positive*)	[str. 92]
FSIM	wskaźnik podobieństwa cech (ang. *Feature Similarity Index*)	[str. 90]
FWHM	szerokość połówkowa (ang. *Full Width at Half Maximum*)	[str. 90]

GPU	procesor graficzny, zwykle zoptymalizowany pod wykorzystanie uczenia maszynowego (ang. *Graphics Processing Unit*)	[str. 10]
H	wysokość przestrzenna obrazu	[str. 41]
I	dane wejściowe sieci (1 lub więcej obrazów)	[str. 71]
I_x	natężenie światła rozważanego obiektu	[str. 28]
I_{ref}	natężenie światła obiektu referencyjnego	[str. 28]
$\hat{I}$	wynik przetworzenia danych wejściowych przez sieć (pojedynczy obraz)	[str. 71]
im	standardowe oznaczenie omawianego obrazu	[str. 63]
K	rozmiar wektora kodowań	[str. 72]
K_i	pojedyncza, *i*-ta klatka obserwacyjna	[str. 27]
k_s	wielkość jądra konwolucyjnego	[str. 43]
L	funkcja straty/kosztu	[str. 40]
LeakyReLU	zmodyfikowany („przeciekający") wariant funkcji ReLU	[str. 48]
MAE	średni błąd bezwzględny, norma L1 (ang. *Mean Absolute Error*)	[str. 57]
mag	magnitudo; jednostka wyrażona w skali logarytmicznej służąca określeniu jasności ciał niebieskich	[str. 28]
MAX_{im}	maksymalna możliwa wartość obrazu (255 dla obrazów zapisanych jako uint8, 65535 dla uint16, 1 dla obrazów znormalizowanych do przedziału [0; 1])	[str. 73]
MFBD	algorytm ślepej dekonwolucji wieloklatkowej wykorzystywany do poprawy jakości rejestrowanych obrazów Słońca (ang. *Multi-Frame Blind Deconvolution)*	[str. 7]
$MFBD_n$	wynik działania MFBD wyznaczony dla *n* klatek	[str. 118]
MFGS	miara podobieństwa gradientów po filtracji medianowej (ang. *Median Filter-Gradient Similarity*)	[str. 125]
MSE	średni błąd kwadratowy, norma L2 (ang. *Mean Squared Error*)	[str. 57]
MNIST	zbiór obrazów odręcznie pisanych cyfr wykorzystany do badań wykorzystujących dane syntetyczne (ang. *Modified National Institute of Standards and Technology database*)	[str. 6]

N2C	skrócony zapis techniki uczenia sieci Noise2Clean	[str. 81]
N2N	skrócony zapis techniki uczenia sieci Noise2Noise	[str. 81]
N	liczba warstw	[str. 51]
N_i	szum występujący na i-tej klatce	[str. 27]
NLM	algorytm nielokalnych średnich służący do redukcji szumu (ang. *Non-Local Means*)	[str. 65]
n	liczba klatek	[str. 27]
np_i	liczba pikseli w i-tym wymiarze matrycy	[str. 25]
p_d	próg detekcji algorytmu DAOStarFinder	[str. 90]
p_x	fizyczna wielkość piksela, najczęściej wyrażana w mikrometrach	[str. 25]
PR	krzywa opisująca precyzję w funkcji czułości (ang. *Precision-Recall*)	[str. 92]
PRNU	niejednorodność czułości pikseli względem padającego na nie światła (ang. *Photo Response Non-Uniformity*)	[str. 23]
PSNR	podstawowa metryka służąca porównaniu względnej jakości dwóch obrazów w oparciu o błąd średniokwadratowy (ang. *Peak Signal-to-Noise Ratio*)	[str. 73]
ref	standardowe oznaczenie obrazu referencyjnego	[str. 73]
r_k	promień koła wykorzystywanego w pomiarze jasności obiektu	[str. 94]
r_{pw}	promień wewnętrzny pierścienia wykorzystywanego w pomiarze jasności obiektu	[str. 94]
r_{pz}	promień zewnętrzny pierścienia wykorzystywanego w pomiarze jasności obiektu	[str. 94]
ReLU	prostowana jednostka liniowa; jedna z najbardziej popularnych funkcji aktywacji (ang. *Rectified Linear Unit*)	[str. 48]
RMSE	pierwiastek z błędu średniokwadratowego (ang. *Root Mean Square Error*)	[str. 126]
S	rzeczywisty (niezaszumiony) sygnał	[str. 27]
S_{px}	skala kątowa piksela wyrażana w sekundach kątowych na piksel	[str. 25]

SSIM	metryka porównawcza obrazów uwzględniająca podobieństwo strukturalne (ang. *Structural Similarity Index Measure*)	[str. 74]
STS	Szwedzki Teleskop Słoneczny znajdujący się na Wyspach Kanaryjskich	[str. 29]
SUTO	działająca na Politechnice Śląskiej grupa badawcza zajmująca się astronomią (ang. *Silesian University of Technology Observatories*)	[str. 29]
TN	oznaczenie klasyfikacji prawdziwie negatywnej (ang. *True Negative*)	[str. 92]
TP	oznaczenie klasyfikacji prawdziwie pozytywnej (ang. *True Positive*)	[str. 92]
TLU	progowa jednostka logiczna (ang. *Threshold Logic Unit*)	[str. 36]
VIF	metryka „wierności informacji wizualnej” (ang. *Visual Information Fidelity*)	[str. 125]
W	szerokość przestrzenna obrazu	[str. 41]
$W_{i,j}$	oznaczenie j-tej warstwy w i-tej grupie warstw sieci	[str. 123]
w	wektor wag	[str. 36]
x	wektor sygnałów wejściowych	[str. 36]
x_i	oznaczenie punktów, dla których funkcja osiąga połowę wartości maksymalnej (w opisie FWHM)	[str. 90]
y	wynik działania sieci	[str. 57]
$\hat{y}$	rzeczywista, oczekiwana, docelowa wartość porównywana z wynikiem działania sieci	[str. 57]
z	suma ważona sygnałów wejściowych z dodanym obciążeniem	[str. 36]
ZP	punkt zerowy (ang. *zero point*), który określa wymagane przesunięcie wartości obliczonych w magnitudo jasności obiektów, by można dokonać porównania pomiędzy obrazami uzyskanymi przez różne instrumenty	[str. 95]

α	współczynnik kierunkowy funkcji LeakyReLU określony dla wartości ujemnych	[str. 48]
Δp	zmiana pozycji gwiazdy na rozważanym obrazie *im* obliczona względem pozycji na obrazie referencyjnym *ref*	[str. 94]
δ	wartość progu w funkcji straty Hubera	[str. 58]
η	współczynnik uczenia	[str. 40]
θ	rozdzielczość optyczna teleskopu	[str. 24]
λ	długość fali światła	[str. 24]
σ	rozrzut wartości szumu; także: odchylenie standardowe	[str. 27]
ρ	zbiór parametrów sieci	[str. 40]

Spis rysunków

Spis tabel