

Politechnika Warszawska

W Y D Z I A Ł E L E K T R Y C Z N Y



INSTYTUT STEROWANIA I ELEKTRONIKI PRZEMYSŁOWEJ
ZAKŁAD STEROWANIA

Praca dyplomowa inżynierska

na kierunku INFORMATYKA STOSOWANA
w specjalności Inżynieria oprogramowania

Poprawa rozdzielczości zdjęć przy użyciu krzyżowo-skalowej
korelacji cech

Aliaksandr Karolik

nr albumu 295138

promotor
dr inż. Grzegorz Sarwas

Warszawa 2017

Poprawa rozdzielczości zdjęć przy użyciu krzyżowo-skalowej korelacji cech.

Streszczenie

Niniejsza praca porusza problem wykorzystania ceia pecia do zrobienia czegoś wielkiego. W pracy przeanalizowane zostały algorytmy do wykrywania ceia pecia. Wybrane algorytmy zostały zaimplementowane i przebadane. Najlepsze rozwiązania zostały wykorzystane w zaprojektowanej i zbudowanej aplikacji.

Słowa kluczowe: praca dyplomowa, LaTeX, jakość

THESIS TITLE

Abstract

This thesis presents a novel way of using a novel algorithm to solve complex problems of filter design. In the first chapter the fundamentals of filter design are presented. The second chapter describes an original algorithm invented by the authors. It is based on evolution strategy, but uses an original method of filter description similar to artificial neural network. In the third chapter the implementation of the algorithm in C programming language is presented. The fifth chapter contains results of tests which prove high efficiency and enormous accuracy of the program. Finally some possibilities of further development of the invented algorithms are proposed.

Keywords: thesis, LaTeX, quality

Warszawa, 1 lutego 2017

POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY

OŚWIADCZENIE

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa inżynierska pt. Poprawa rozdzielczości zdjęć przy użyciu krzyżowo-skalarowej korelacji cech:

- została napisana przeze mnie samodzielnie,
- nie narusza niczych praw autorskich,
- nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam, że przedłożona do obrony praca dyplomowa nie była wcześniej podstawą postępowania związanego z uzyskaniem dyplomu lub tytułu zawodowego w uczelni wyższej. Jestem świadom, że praca zawiera również wyniki stanowiące własności intelektualne Politechniki Warszawskiej, które nie mogą być udostępniane innym osobom i instytucjom bez zgody Władz Wydziału Elektrycznego.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Aliaksandr Karolik.....

Spis treści

1 Wstęp	1
2 Algorytmy do poprawy rozdzielczości zdjęć	3
2.1 Wstęp teoretyczny	3
2.2 Konwolucyjna sieć neuronowa	4
2.3 Konwolucyjne sieci neuronowe dla super-rozdzielczości	6
2.4 Model CSNLN	7
2.4.1 Moduły uwagi	8
2.4.2 Mutual-Projected Fusion	10
2.5 Model EDSR	12
3 Implementacje	14
3.1 Środowisko programistyczne	14
3.2 Zbiory danych	15
3.3 Model CSNLN	16
3.4 Model EDSR	17
3.5 Metryki porównania jakości modeli	18
3.5.1 Szczytowy stosunek sygnału do szumu	18
3.5.2 Podobieństwo strukturalne	19
4 Testowanie wybranych modeli	21
4.1 Testowanie na zbiorze danych DIV2K	21
4.2 Rozmiary modeli	24
5 Usuwanie szumu z obrazów	25
5.1 Eksperement z użyciem zbioru tylko zasumionych obrazów	25
5.2 Eksperement z użyciem zbioru rozszerzonego	27

Rozdział 1

Wstęp

W dobie dużej popularności cyfrowej rejestracji obrazów przy wykorzystaniu urządzeń mobilnych takich jak kamery, czy tablety za pomocą wbudowanych w nich aparatów fotograficznych jakość/rozdzielcość zarejestrowanych obrazów nie zawsze jest zadowalająca. Zarejestrowane materiały są w różnoraki sposób zakłócone, zniekształcone, co nie pozwala nam na wydruk, w odpowiedniej jakości, tego typu materiału. Ponieważ optyka zainstalowana w średniej półki telefonach komórkowych nie pozwala na uzyskanie wystarczającej jakości fotografii, widać wyraźne zapotrzebowanie na algorytmy poprawiające rozdzielcość i jakość zarejestrowanych obrazów.

Czym jest super-rozdzielcość? Super-rozdzielcość (pisana również jako super resolution, superresolution) jest określeniem zestawu metod zwiększenia skali wideo lub obrazów. Terminy takie jak „skalowanie w górę”, „powiększanie”, „konwersja w górę” i „uprez” również opisują wzrost rozdzielcości w przetwarzaniu obrazu lub edycji wideo.

Większość technik super-rozdzielcości opiera się na tym samym pomyśle: wykorzystanie informacji z kilku różnych obrazów do stworzenia jednego powiększonego obrazu. Algorytmy próbują wyodrębnić szczegóły z każdego obrazu w sekwencji, aby zrekonstruować inne ramki. Obraz w wysokiej rozdzielcości oferuje dużą gęstość pikseli, a tym samym więcej szczegółów na temat oryginalnej sceny.

Potrzeba wysokiej rozdzielcości jest powszechna w wizji komputerowej dla lepszej wydajności w rozpoznawaniu wzorów i analizie obrazów. Wysoka rozdzielcość ma znaczenie w obrazowaniu medycznym dla diagnozy. Wiele aplikacji wymaga powiększenia określonego obszaru w którym niezbędna jest wysoka rozdzielcość, np. aplikacje do nadzoru, kryminalistyki i obrazowania satelitarnego.

Praca ta skupiać się będzie na badaniu rozwiązań algorytmicznych w dziedzinie widzenia komputerowego służących do poprawy rozdzielcości, zwa-

nych również algorytmami super-rozdzielczości (super-resolution).

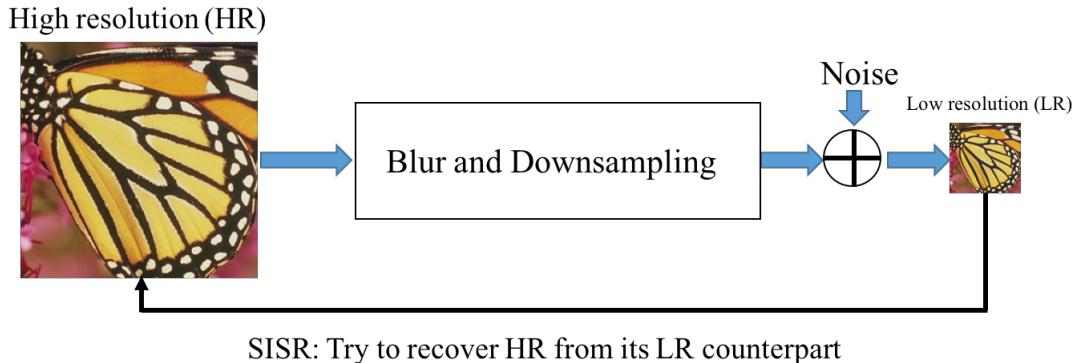
Rozdział 2

Algorytmy do poprawy rozdzielczości zdjęć

W tym rozdziale opisane zostaną podstawowe, jak i obecnie używane architektury sieci neuronowych do poprawy rozdzielczości zdjęć. Przedstawione zostaną teoretyczne podstawy działania oraz główne założenia budowy. Architektury sieci są tak dobrane, aby móc zaprezentować rozwój pomysłów ich twórców.

2.1 Wstęp teoretyczny

Super-rozdzielcość (SR) odnosi się do zadania przywracania obrazów o wysokiej rozdzielczości z jednej lub więcej obserwacji tej samej sceny w niskiej rozdzielczości (LR). Zgodnie z liczbą wejściowych obrazów LR, SR można podzielić na super-rozdzielcość pojedynczego obrazu (SISR) i super-rozdzielcość wielu obrazów (MISR). W porównaniu z MISR SISR jest znacznie bardziej popularną metodą ze względu na wysoką wydajność. Typowa struktura SISR jest zaprezentowana na rysunku 2.1.



Rysunek 2.1: Szkic SISR

Główne algorytmy SISR dzielą się na trzy kategorie: metody oparte na interpolacji, metody oparte na rekonstrukcji oraz metody oparte na uczeniu. Metody SISR oparte na interpolacji, takie jak interpolacja dwusześcienna (bicubic interpolation) i próbkowanie Lanczosa (Lanczos resampling), są bardzo szybkie i proste, ale dość niedokładne.

Metody SR oparte na rekonstrukcji, często przyjmują zaawansowaną wcześniejszą wiedzę w celu ograniczenia możliwej przestrzeni rozwiązań z korzyścią polegającą na generowaniu elastycznych i ostrych szczegółów. Jednak wydajność wielu metod opartych na rekonstrukcji szybko spada, gdy zwiększa się skala, oraz metody te są zwykle czasochłonne.

Metody SISR oparte na uczeniu, znane również jako metody oparte na przykładach, najczęściej używane ze względu na ich szybkie obliczenia i wyjątkową wydajność. Metody te zwykle wykorzystują algorytmy uczenia maszynowego do analizy związków statystycznych między LR i odpowiadającym mu odpowiednikiem HR z istotnych przykładów szkoleniowych.

Technika MISR wykorzystuje jako wejście zestaw obrazów niskiej rozdzielczości do budowy obrazu HR, ale jak już wcześniej było wspomniane, SISR jest popularniejsza ze względu na wysoką wydajność.

2.2 Konwolucyjna sieć neuronowa

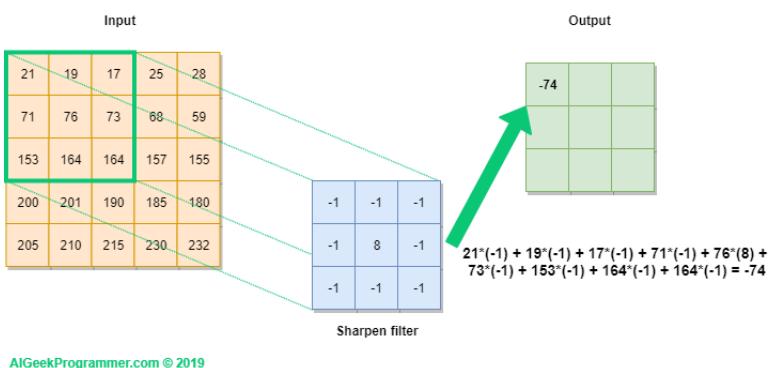
Konwolucyjne sieci neuronowe (CNN) są prawie wszędzie. Jest to prawdopodobnie najbardziej popularna architektura głębokiego uczenia. Niedawny wzrost zainteresowania głębokim uczeniem wynika z ogromnej popularności i skuteczności konwolucyjnych sieci neuronowych. Zainteresowanie CNN rozpoczęło się od AlexNet [5] w 2012 roku i od tego czasu rosło wykładniczo.

W ciągu zaledwie trzech lat naukowcy przeszli z 8-warstwowej sieci AlexNet [5] do 152-warstwowej sieci ResNet [4].

CNN jest obecnie modelem go-to dla każdego problemu związanego z obrazem. CNN również stosowane w systemach rekomendacji, przetwarzania języka naturalnego i nie tylko. Główną zaletą sieci CNN w porównaniu z jej poprzednikami jest to, że automatycznie wykrywa ona ważne cechy bez żadnego nadzoru człowieka. Na przykład, biorąc pod uwagę wiele zdjęć kotów i psów, sieć sama uczy się cech charakterystycznych dla każdej klasy.

Podstawowym narzędziem sieci jest warstwa konwolucyjna. Warstwa konwolucyjna składa się z zestawu filtrów, zadaniem, której jest wykrycie poszczególnych cech ze zdjęcia.

Mnożenie splotowe lub konwolucja to operacja matematyczna polegająca na połączeniu dwóch zestawów informacji. W naszym przypadku konwolucja jest stosowana na danych wejściowych oraz filtru. W wyniku powstaje nowa macierz, która jest nazywana mapą cech, wartości mapy są wynikiem kombinacji liniowej poszczególnych pikseli obrazu wejściowego i przesuwającego się filtra. Poniżej na rysunku 2.2 zaprezentowany jest schemat mnożenia splotowego lub konwolucji:

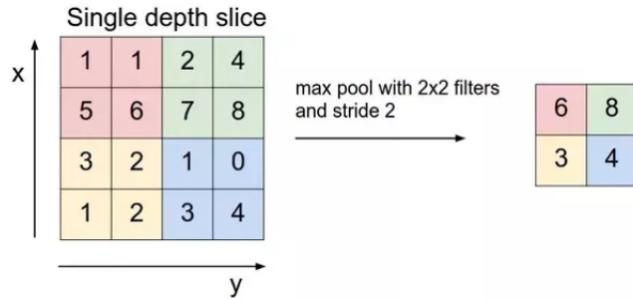


Rysunek 2.2: Schemat mnożenia splotowego (konwolucji)

Tak samo, jak w zwykłej sieci, po warstwie konwolucyjnej występuje warstwa aktywacji (najczęściej używana jest funkcja ReLU), zadaniem, której jest wprowadzenie nieliniowości do sieci.

Drugim podstawowym elementem sieci konwolucyjnej jest warstwa łączącą (pooling layer). Zadaniem jej jest zmniejszenie wymiarów mapy cech, wyznaczonej w warstwie konwolucyjnej, przy zachowaniu jej kluczowych elementów. Warstwa ta również odpowiada za redukcję szumu. Najczęściej używaną metodą jest „Max pooling”.

Algorytm działania metody Max pooling jest następujący: definiowany jest filtr oraz krok przesunięcia. Kolejne wartości macierzy wyjściowej są maksymalną wartością objętą filtrem. Na rysunku 2.3 zaprezentowany jest schemat działania metody Max pooling:



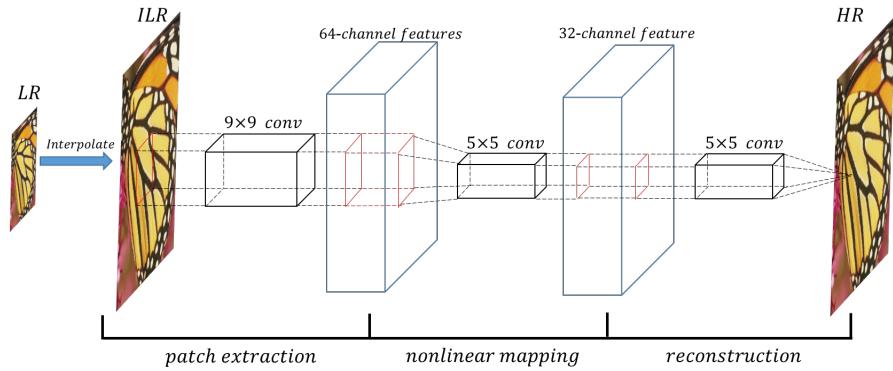
Rysunek 2.3: Schemat metody Max pooling

2.3 Konwolucyjne sieci neuronowe dla super-rodziczości

Pierwszą zaproponowaną architekturę używającą CNN do mapowania obrazów niskiej rozdzielczości do wysokiej jest SRCNN [2]. Architekturę SRCNN zaprezentowana jest na rysunku 2.4. SRCNN jest trójwarstwowym CNN, poszczególne warstwy której składają się z filtrów o wymiarach $64 \times 1 \times 9 \times 9$, $1 \times 32 \times 5 \times 5$ i $1 \times 32 \times 5 \times 5$. Każda warstwa odpowiada za następujące czynności:

1. Wyodrębnienie i reprezentacja. Obraz jest przepuszczany przez zestaw filtrów. Zadaniem, których jest wyodrębnienie cech specyficznych.
2. Mapowanie nieliniowe. Dla każdej mapy cech wyprodukowanych w poprzedniej warstwie przyporządkowywany jest wektor cech o wysokiej rozdzielczości.
3. Rekonstrukcja. Ostatnia warstwa konwulacyjna układa wektory cech uzyskane w poprzedniej warstwie w jeden obraz wysokiej rozdzielczości.

Autorzy proponujące architekturę SRCNN przewidywali, że zwiększenie ilości warstw konwulacyjnych korzystnie wpłynie na wyniki. Niedługo po ich publikacji zaczęły pojawiać się rozwiązania o głębszych architekturach. Kim i in. zaproponowali bardzo głęboki model VDSR z ponad 16 warstwami



Rysunek 2.4: Architektura SRCNN

konwolucyjnymi korzystającymi ze skutecznego uczenia rezydualnego (reszt-kowego,residual learning). Aby w pełni wykorzystać moc głębokich CNN, Lim i in. zintegrowali bloki rezydualne w framework SR, w wyniku powstały modeli EDSR i MDSR.

2.4 Model CSNLN

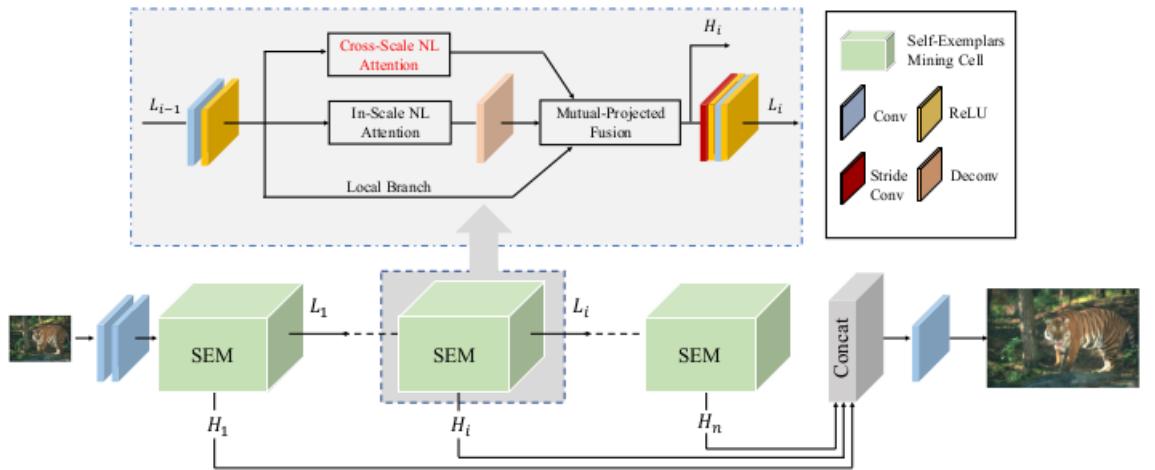
Modeli do poprawy rozdzielczości pojedynczego obrazu oparte na dużej ilości warstw konwolucyjnych wykorzystują korzyści płynące z dużych zewnętrznych zasobów obrazu do lokalnej odbudowy, jednak w większości istniejących prac pominięto dalekosiężne podobieństwa cech charakterystycznych. Niektóre z ostatnich prac z powodzeniem wykorzystały te wewnętrzne korelacje cech, używając nielokalne moduły uwagi.

Jednakże istniejące podejścia do odtwarzania obrazów używały jedynie podobieństwa cech w tej samej skali, ignorując liczne wewnętrzne wzorce LR-HR w różnych skalach, co prowadziło do stosunkowo niskiej wydajności. Wiadomo, że wewnętrzne korelacje HR zawierają bardziej istotne informacje o wysokich częstotliwościach. W tym celu, Yiqun Mei i in. w swoim artykule [9] proponują pierwszy moduł uwagi Cross-Scale Non-Local (CS-NL) z integracją do rekurencyjnej sieci neuronowej.

Proponowana architektura sieci przedstawiona jest na rysunku 2.5. Jest to w zasadzie rekurencyjna sieć neuronowa, w której każda komórka zwana Self-Exemplars Mining (SEM) w pełni integruje uczenie oparte o lokalne czynniki, nielokalne czynniki w tej samej skali obrazów oraz w skali zmiennej używając nowo proponowany moduł CS-NL.

Powtarzające się komórki SEM są osadzone w sekwencyjną strukturę, jak

pokazano na rysunku 2.5. Każda komórka SEM produkuje H_i przekształconą mapę cech powstającą z połączenia wyników dwóch modułów uwagi oraz mapy cech otrzymanej ze zwykłej warstwy konwolucyjnej. Wydobyte mapy cech H_i łączone jedną warstwą konwulacyjną w wyniku łączenia powstaje obraz wysokiej rozdzielczości.



Rysunek 2.5: Architektura CSNLN

2.4.1 Moduły uwagi

Jak było już wspomniano wcześniej wybrany algorytm wyszukuje nielokalne korelacje w obrazach w tej samej skali oraz w skali zmiennej. Korelacje poszukiwane są za pomocą dwóch modułów uwagi nielokalnej o nazwach In-Scale Non-Local Attention (IS-NL) oraz Cross-Scale Non-Local Attention (CS-NL).

Nielokalna uwaga może eksplorować próbki poprzez podsumowanie cech charakterystycznych ze zbioru obrazów wejściowych. Autorzy publikacji używając artykułu [10] zdefiniowali nielokalną uwagę dla wybranej X mapy cech obrazu przy pomocy wzoru 2.1.

$$Z_{i,j} = \sum_{g,h} \frac{\exp(\phi(X_{i,j}, X_{g,h}))}{\sum_{u,v} \exp(\phi(X_{i,j}, X_{u,v}))} \psi(X_{g,h}), \quad (2.1)$$

gdzie:

- $(i, j), (g, h)$ oraz (u, v) pary kordynat mapy X

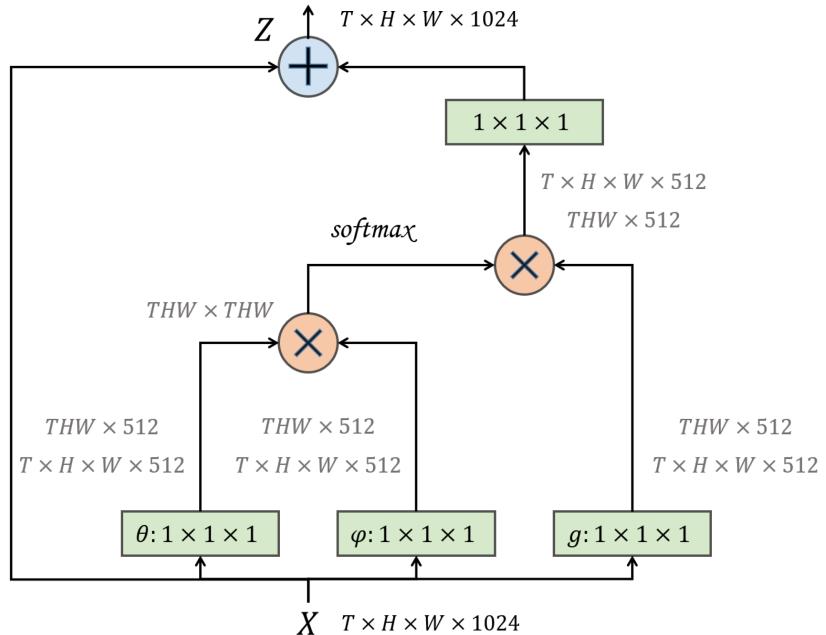
- $\phi(\cdot)$ funkcja transformacji cech
- $\psi(\cdot, \cdot)$ funkcją korelacji do pomiaru podobieństwa. Definiowana w sposób 2.2

$$\psi(X_{i,j}, X_{g,h}) = \theta(X_{i,j})^T \delta(X_{g,h}), \quad (2.2)$$

gdzie:

- $\theta(\cdot), \delta(\cdot)$ funkcje transformacji cech

W oparciu o wzór 2.1 autorzy zaimplementowali moduł IS-NL poniżej na rysunku 2.6 jest graficzna reprezentacja.

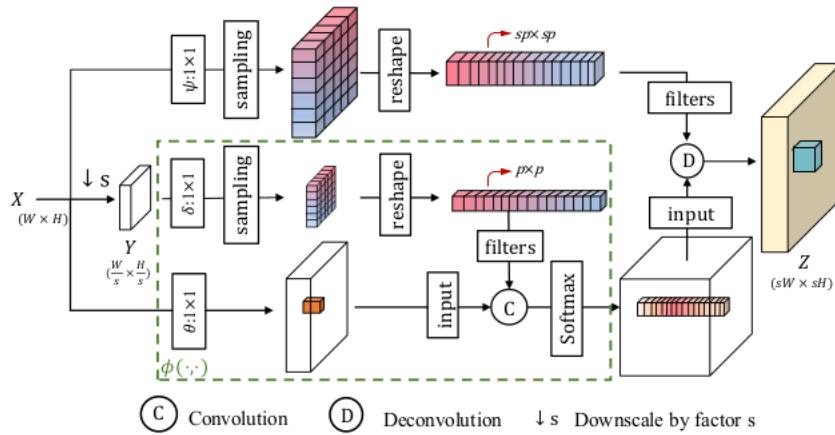


Rysunek 2.6: Moduł IS-NL

Powyższe sformułowanie 2.1 zostało rozszerzone do wersji używającej krzyżowo-skalową korelację cech. Zamiast pomiaru wzajemnej korelacji pikselowej jak jest robione w modułu IS-NL, proponowany nowy moduł uwagi ma na celu pomiar korelacji pomiędzy pikselami o niskiej rozdzielczości a plastrami większej skali obrazu LR.

Na rysunku 2.7 zaprezentowana została architektura modułu uwagi wykorzystującą krzyżowo-skalową korelację cech zaprojektowana w oparciu o 2.1 przez twórców artykułu. Moduł CS-NL działa w następujący sposób wejściowa mapa cech X o wymiarach $W \times H$ przekształcana jest przy pomocy

interpolacji dwuliniowej do Y o wymiarach $\frac{W}{s} \times \frac{H}{s}$. Następnie plastry $p \times p$ z mapy X są porównywane do $p \times p$ w mapie Y aby uzyskać wynik dopasowania softmax. Na samym końcu jest używana warstwa dekonwolucyjna na wynikach softmaxa oraz wagach plastra o wymiarze $sp \times sp$ wydobywana z mapy X . W wyniku powstaje mapa Z o wymiarach $sW \times sH$, która będzie s razy bardziej określona niż X .



Rysunek 2.7: Moduł CS-NL

2.4.2 Mutual-Projected Fusion

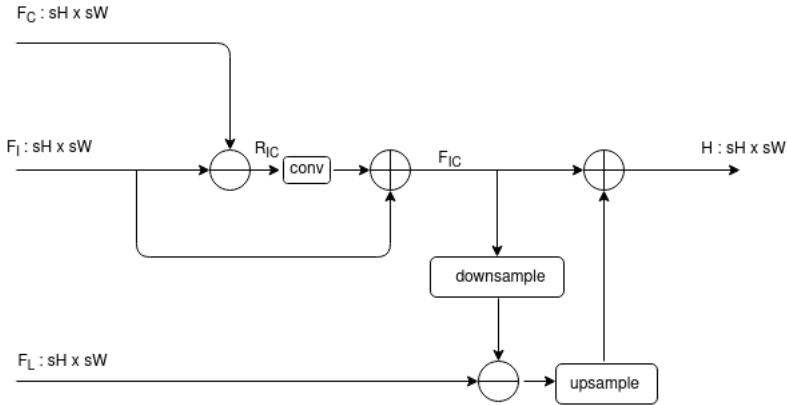
Jak wcześniej było wspomniane, każda komórka SEM zawiera trójbranżową strukturę, przy pomocy której generuje trzy mapy cech poprzez niezależne wykorzystanie każdego ze źródeł informacji z obrazów LR, niejasne pozostaje, jak połączyć te oddzielne tensory w kompleksową mapę funkcji. Autorzy zaproponowali własny algorytm do stopniowego łączenia cech. Procedura algorytmu została przedstawiona na rysunku 2.8.

Aby pozwolić sieci skupić się na bardziej informacyjnych cechach najpierw jest obliczana różnica R_{IC} pomiędzy dwoma mapami cech F_I i F_C z modułów IS-NL oraz CS-NL odpowiednio. Następnie wynikowa mapa R_{IC} jest przepuszczana przez jedną warstwę konwolucyjną, później z powrotem jest dodawana mapa F_I jest to robione do przewrócenia straconych informacji przy operacji odejmowania.

$$R_{IC} = F_I - F_C \quad (2.3)$$

$$F_{IC} = conv(R_{IC}) + F_I \quad (2.4)$$

Pozostająca cecha R_{IC} reprezentuje szczegóły istniejące w jednym źródle, a brakujące w drugim. Taka projekcja pozwala sieci skupić się tylko na odrebnich informacjach pomiędzy źródłami, omijając przy tym powszechną wiedzę, co zwiększa jej zdolność dyskryminacyjną.



Rysunek 2.8: Algorytm łączenia map cech

Wzorując się artykułem DBPN [3], autorzy zaadoptowali podejście back-projection w celu włączenia lokalnych informacji, aby dodać regularyzację cech i skorygować błędy rekonstrukcji. Wynikowa mapa H jest obliczana w następujący sposób.

$$e = F_L - \text{downsample}(F_{IC}), \quad (2.5)$$

$$H = \text{upsample}(e) + F_{IC} \quad (2.6)$$

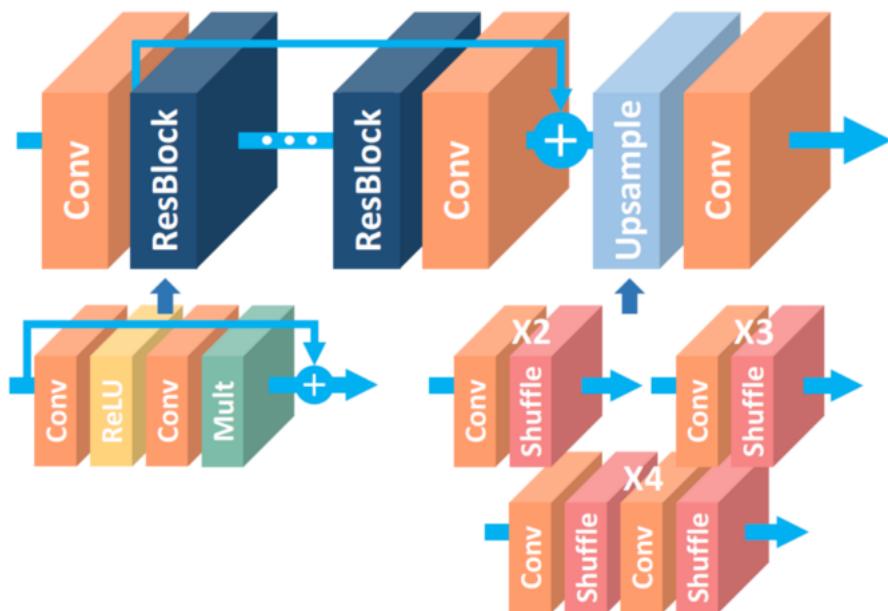
gdzie F_L jest mapą cech z gałęzi lokalnej (Local branch), *downsample* jest dokonywany przy pomocy warstwy konwolucyjnej do zmniejszenia wymiaru oraz *upsample* odpowiednio warstwa konwolucyjna do przewrócenia wymiarów do $sH \times sW$.

Zaproponowany algorytm gwarantuje uczenie resztkowe (residual learning) przy jednoczesnym łączeniu różnych źródeł cech, co umożliwia bardziej dyskryminacyjne uczenie w porównaniu do zwykłego dodawania lub łączenia.

2.5 Model EDSR

Aby zweryfikować skuteczność modelu CSNLN, postanowiono porównać otrzymane wyniki do metody state-of-the-art o nazwie EDSR [7] (ang. Enhanced Deep Residual Networks for Single Image Super-Resolution). Wybór takiego modelu był podyktowany częstym odwoływaniem się do wspomnianego modelu w wielu artykułach naukowych napotkanych w trakcie przeglądania literatury związanej z dziedziną problemu.

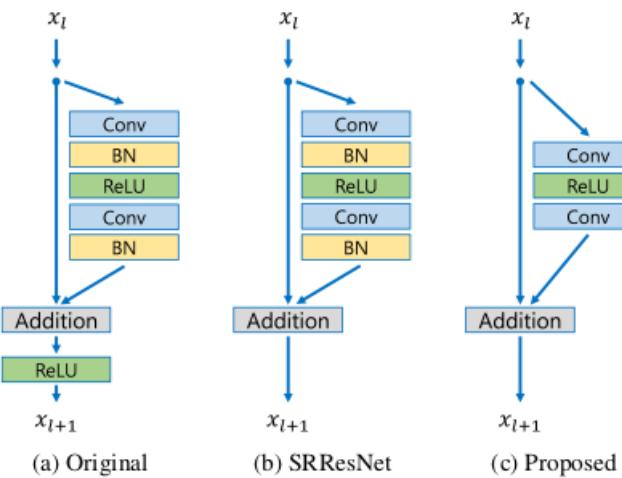
Architektura modelu EDSR zaprezentowana na rysunku 2.9. Struktura modelu jest podobna do sieci SRResNet [6], ale zaproponowany model nie posiada warstw aktywacji poza blokami rezydualnymi. Dodatkowo w modelu EDSR zostały zmodyfikowane bloki rezydualne.



Rysunek 2.9: Architektura modelu EDSR

Rysunku 2.10 zawiera nowo proponowany blok rezydualny wraz z porównaniem go z blokiem z sieci ResNet oraz SRResNet. Nowo proponowany blok składa się tylko z dwóch warstw konwolucyjnych oraz warstwy aktywacji. Jak widać z rysunku 2.10 warstwy normalizacji wsadowej (ang. Batch Normalization) zostały usunięte. Taka decyzja była podjęta, ponieważ warstwy normalizacji wsadowej, pozbywają się elastyczności sieci poprzez normalizację cech obrazu wejściowego.

Taka konfiguracja modelu bez warstwy normalizacji pozwoliła zaoszczędzić około 40% zużycia pamięci podczas szkolenia, w porównaniu z SRResNet [6].



Rysunek 2.10: Architektura modelu EDSR

Rozdział 3

Implementacje

W tym rozdziale zostaną przedstawione implementacje wcześniej opisanego modelu oraz benchmarku do poprawy rozdzielczości zdjęć. Modeli były przygotowywane do rozwiązania problemu dwukrotnego oraz trzechkrotnego powiększania rozdzielczości zaszumionych obrazów.

3.1 Środowisko programistyczne

Rozpoczynając pracę nad modelami, używany był własny komputer Lenovo Yoga 500 z kartą graficzną Nvidia Geforce 940M oraz systemem operacyjnym Linux 20.04.1 LTS. Po przeanalizowaniu implementacji wybranych metod stało się jasne, że nie uda się przeprowadzić proces uczenia na maszynie ze względu na zbyt małą ilość pamięci graficznej. Rozwiązaniem tego problemu było udostępnienie mnie karty graficznej **Nvidia Geforce RTX 2080** na serwerze politechniki warszawskiej. Lokalne środowisko zostało przeniesione na serwer.

Język programowania został wybrany **Python**. Ze względu na wcześniejszą znajomość języka, duży zbiór narzędzi do wizualizacji i przekształcania obrazów oraz ogólną popularność w zastosowaniach związanych ze sztuczną inteligencją.

Jako biblioteka do uczenia maszynowego została wybrana **PyTorch**. Główne wybór był spowodowany istniejącymi implementacjami modeli oraz chęcią poznania tej biblioteki jako narzędzia, które będzie używane w przyszłości.

Do tworzenia własnych datasetów wykorzystano biblioteki **Pillow** i **OpenCV**. Są to zaawansowane biblioteki języka programowania Python, udostępniające zestawy modułów służących do obróbki grafiki.

3.2 Zbiory danych

Do uczenia modeli zostało wykorzystano trzy zbiory danych:

1. zbiór danych DIV2K,
2. własny zbiór danych zawierający tylko zaszumione obrazy,
3. własny zbiór danych zawierający obrazy zaszumione jak i nie zaszumione.

Pierwszym zbiorem danych był zbiór o nazwie **DIV2K** [1]. Jest to zbiór 1000 kolorowych obrazów RGB, autorzy tworzące ten zbiór zwracali szczególną uwagę na jakość obrazu, różnorodność źródeł (stron i kamer). Obrazy niskiej rozdzielczości uzyskiwane przez pomniejszenie zdjęć ze zbioru za pomocą interpolacji dwusześciennej. Zbiór był używany do wyzwań NTIRE (2017, 2018) oraz PIRM 2018. Zbiór DIV2K składa się:

- zbioru uczącego. Zbiór zawiera 800 par obrazów o wysokiej i niskiej rozdzielczości.
- zbioru validacyjnego. Zbiór zawiera 100 par obrazów o wysokiej i niskiej rozdzielczości.
- zbioru testowego. Zbiór zawiera 100 par obrazów o wysokiej i niskiej rozdzielczości.

Biorąc pod uwagę jakość obrazów zbioru DIV2K ze stanem faktycznym zdjęć robionych przy pomocy telefonu komórkowego, postanowiono stworzyć własne zbiory danych. Do tego wybrano 150 obrazów, zrobionych za pomocą własnego telefonu komórkowego Huawei P20 rozdzielcość obrazów wynosiła 4160×3120 . Zbiór zawierał zdjęcia natury, ludzi, krajobrazów oraz zwierząt.

Drugi zbiór danych wykorzystywany do trenowania i walidacji modeli został utworzony wykorzystując wcześniej opisane zdjęcia. Tworzony on był z myślą o sprawdzenie jak modeli będą radzić z zadaniem odszumiania zdjęć. Aby to sprawdzić każde zdjęcie zostało zinterpolowane za pomocą interpolacji dwuliniowej oraz po zmniejszeniu każdy obraz był powielony czterokrotnie z dodaniem różnych szumów. W tym przypadku zostały wykorzystane następujące szумy:

- szum Gausowski,
- szum Gausowski z lokalne odchylenia w każdym punkcie obrazu,
- szum typu sól,
- szum typu piepsz.

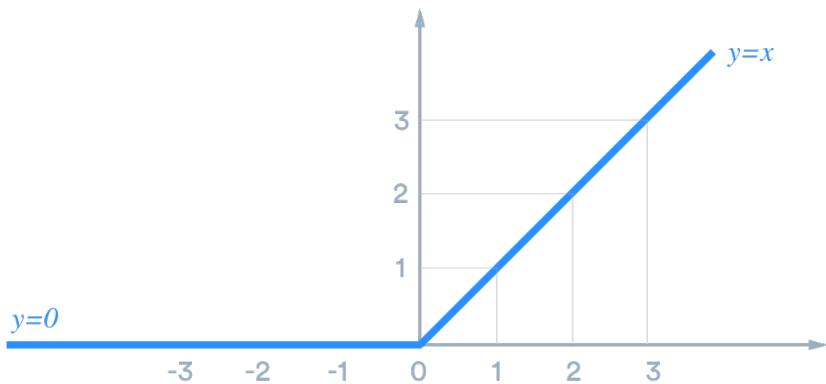
Ostatni zbiór danych wejściowych był zrobiony jako kombinacja zdjęć zaszumionych i zinterpolowanych. Taki zbiór zapewniał dużą zdolność generalizacji, bo w przypadku modeli uczonych tylko na zdjęciach zaszumionych nie były one w stanie zapewnić dobrą jakość obrazu w przypadku gdy na wejście trafiały obrazy niezaszumione.

Podstawowy zbiór zawierający 150 obrazów był powielany pięciokrotnie. Każde zdjęcie również było poddawane interpolacji dwuliniowej, ale tylko dwa zdjęcia z pięciu były poddawane zaszumieniu. Pozostałe trzy zdjęcia były niezmieniane. Używane były następujące szумy:

- szum Gausowski,
- połączenie szumów sól i piepsz.

3.3 Model CSNLN

Pracując nad przygotowaniem modeli CSNLN do poprawy rozdzielczości zdjęć, wykorzystano artykuł [9] z 2020 roku oraz oficjalną implementację autorów artykułu, która jest ogólnie dostępna na GitHub. Zastosowano architekturę przedstawioną w tej publikacji, która była wcześniej opisana w sekcji 2.4. Model składał się z 12 komórek SEM. W każdej z warstw modelu używano funkcji aktywacji ReLU. Wykres tej funkcji jest zaprezentowany na rysunku 3.1.

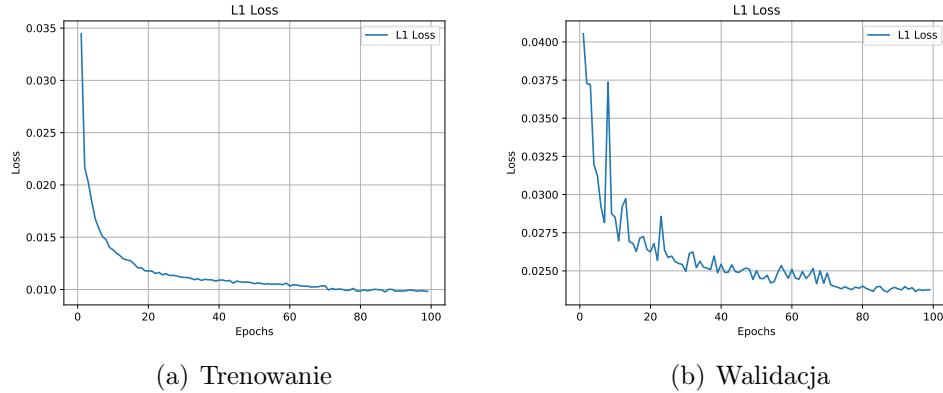


Rysunek 3.1: Wykres funkcji ReLU

Model był trenowany przy pomocy normy L^1 , jednak była podjęta próba udoskonalenie modelu przez zmianę funkcji strat na L^2 . Do optymalizacji wag modeli użyto algorytmu Adam o współczynnikach $\beta_1 = 0.9$, $\beta_2 = 0.999$ oraz $\epsilon = 1e - 8$.

Zgongie z publikacją rozmiar wejściowy obrazów (ang. patch) wynosił 48×48 , jednak został on zmniejszony do 34×34 dla dwukrotnego oraz 36×36 trzechkrotnego powiększenia. Wybór takich był uwarunkowany dostępnej ilością pamięci graficznej. Ewaluacja modeli następowała po przerobieniu 16 zdjęć (ang. batch). Początkowy krok uczenia wynosił $1e - 4$.

Rysunek 3.2 przedstawia wykresy uczenia oraz ewaluacji modeli.



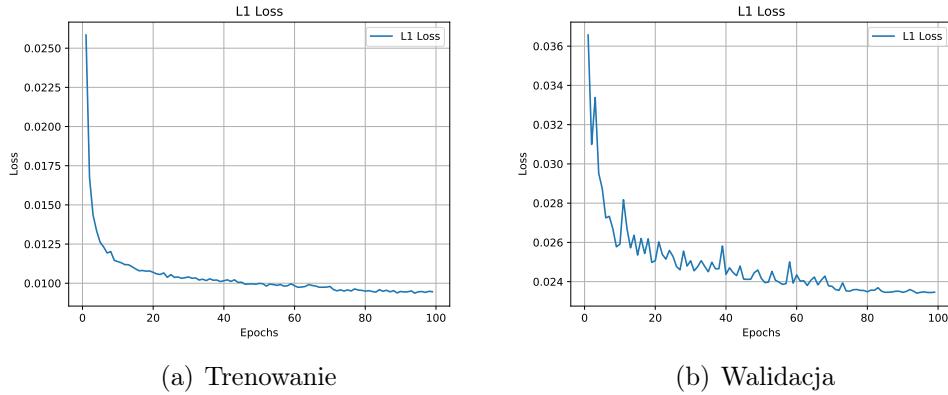
Rysunek 3.2: Wykresy funkcji strat w procesie uczenia modelu CSNLN

3.4 Model EDSR

Badania zostały wykonane na podstawie oficjalnej implementacji, która jest ogólnie dostępna na GitHub. Kod został zaktualizowany do najnowszych dostępnych bibliotek na podstawie artykułu [7]. Konfiguracja obrazów wejściowych była zostawiona taka sama jak w przypadku modelu CSNLN opisanego w sekcji 3.3.

Zgodnie z artykułem zostały wykorzystane 32 bloki rezydualne oraz liczba kanałów była ustalona na 256. Podczas treningu sieci, do optymalizacji jej parametrów wykorzystano algorytm Adam o współczynniku zdefiniowanych w artykule, czyli $\beta_1 = 0.9$, $\beta_2 = 0.999$ oraz $\epsilon = 1e - 8$. Jako funkcję straty przyjęto normę L^1 . Początkowy krok uczenia był ustawiony $1e - 4$. Ewaluacja modelu odbywała się po przejściu 16 obrazów.

Przebieg uczenia oraz ewaluacji modeli zaprezentowano na Rysunku 3.3.



Rysunek 3.3: Wykresy funkcji strat w procesie uczenia modelu EDSR

3.5 Metryki porównania jakości modeli

Do porównania wyników działania algorytmów zostały wykorzystane następujące metryki:

- Szczytowy stosunek sygnału do szumu (PSNR, ang. peak signal-to-noise ratio)
- Podobieństwo strukturalne (SSIM, ang. structure similarity)

3.5.1 Szczytowy stosunek sygnału do szumu

Szczytowy stosunek sygnału do szumu, rzadziej nazywany jako stosunek sygnału szczytowego do szumu (PSNR, ang. peak signal-to-noise ratio) – stosunek maksymalnej mocy sygnału do mocy szumu zakłócającego ten sygnał. Ze względu na szeroki zakres wartości PSNR wyrażany jest w decybelach.

Najczęściej PSNR stosowany jest do oceny jakości kodków wykorzystujących strażną kompresję obrazów. W takim przypadku sygnałem są nieskompresowane dane źródłowe, a szumem – artefakty (zniekształcenia) spowodowane zastosowaniem kompresji strażnej.

W celu wyznaczenia PSNR należy najpierw obliczyć współczynnik MSE (ang. mean squared error) bazujący na obu porównywanych obrazach, wykorzystując wzór 3.1. A później używając współczynnika MSE obliczyć szczytowy stosunek sygnału do szumu za pomocą wzoru 3.2.

$$MSE = \frac{1}{M * N} \sum_{i=1}^N \sum_{j=1}^M ([f(i, j) - f'(i, j)]^2) \quad (3.1)$$

Gdzie:

- N, M - wymiary obrazu w pikselach,
- $f(i, j)$ - wartość piksela o współrzędnych (i, j) obrzu oryginalnego,
- $f'(i, j)$ - wartość piksela o wpółrzędnych (i, j) obrazu skompresowanego.

$$PSNR = 10 \log_{10} \frac{[\max(f(i, j))]^2}{MSE} \quad (3.2)$$

Gdzie:

- $\max(f(i, j))$ – wartość maksymalna danego sygnału; w przypadku obrazów zwykle jest to wartość stała, np. dla obrazów monochromatycznych o reprezentacji 8-bitowej wynosi 255.

3.5.2 Podobieństwo strukturalne

Podobieństwo strukturalne (SSIM) jest metodą przewidywania postrzeganej jakości telewizji cyfrowej i obrazów filmowych, a także innych rodzajów zdjęć i filmów cyfrowych. Indeks SSIM jest metodą pełnego porównania, innymi słowy, mierzy jakość w oparciu o oryginalny obraz (nie skompresowany lub bez zniekształceń). Wskaźnik SSIM jest rozwinięciem tradycyjnych metod, takich jak PSNR (peak signal-to-noise ratio) i standardowej metody średniego błędu kwadratowego (MSE), które okazały się niezgodne z fizjologią ludzkiego postrzegania.

Indeks SSIM zawiera się w przedziale od -1 do $+1$. Wartość $+1$ jest osiągana tylko wtedy, gdy próbki są w pełni autentyczne. Indeks SSIM jest obliczany w różnych oknach obrazu. Miara pomiędzy dwoma oknami x i y o wspólnym rozmiarze $N \times N$ obliczana jest za pomocą wzoru 3.3.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (3.3)$$

Gdzie:

- μ_x wartość średnia z x
- μ_y wartość średnia z y
- σ_x^2 wariancja x
- σ_y^2 wariancja y
- σ_{xy} kowariancja x i y

- $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ dwie zmienne stabilizujące podział ze słabym mianownikiem
- L zakres dynamiki wartości pikseli
- $k_1 = 0.01$ i $k_2 = 0.03$

Rozdział 4

Testowanie wybranych modeli

W tym rozdziale zostały zaprezentowane oraz opisane wyniki przeprowadzonych badań wytrenowanych modelach na pierwszym zbiorze danych opisanym w sekcji 3.2. Zamieszczone zostały metryki porównawcze działania algorytmów dla zdjęć pomniejszonych. Również znajdzie się w tym rozdziale porównanie rozmiarów wytrenowanych modeli dla dwukrotnego oraz trzechkrotnego powiększenia obrazu.

4.1 Testowanie na zbiorze danych DIV2K

W pierwszej części testów postanowiono sprawdzenie wyników modelu przedstawione w publikacji. Jednak proces uczenia nie był dokładnie powtórzony, jak opisuje artykuł. Proces trenowania był przerywany po przejściu 100 epok oraz, jak już było wspomniane, wejściowe obrazy miały mniejsze wymiary.

W tablice 4.1 są zaprezentowane uzyskane wyniki na zbiorze testowym Set5 [8]. Również tabela 4.1 zawiera wyniki powiększenia obrazów przy pomocy interpolacji. Decyzja zamieszczenia tych wyników była kierowana chęcią pokazania efektów działania najprostszych metod powiększenia obrazów w porównaniu do głębszych sieci neuronowych.

Wśród interpolacji wyniki są tym lepsze, im złożonej jest metoda interpolacji. Jak i było przewidywano, wyniki uzyskane przy pomocy sieci neuronowych są lepsze w stosunku do interpolacji. Jednak różnią się one od wyników pokazanych w publikacji, chociaż i nie wiele. Najprawdopodobniej jest to spowodowane zmniejszeniem wymiarów wejściowych obrazu oraz krótszym czasem trenowania.

Drugim czynnikiem wpływającym na uzyskane wartości metryk mogła być zmiana przestrzeni barw, na podstawie której były pozyskiwane wartości

Algorytm	Skala	PSNR	SSIM
Interpolacja dwuliniowa	x2	30.422	0.901
Interpolacja dwusześcienna	x2	32.399	0.928
EDSR	x2	35.841	0.943
CSNLN (L^1)	x2	36.026	0.955
CSNLN (L^2)	x2	35.721	0.933
Interpolacja dwuliniowa	x3	26.396	0.808
Interpolacja dwusześcienna	x3	26.913	0.825
EDSR	x3	32.532	0.903
CSNLN (L^1)	x3	33.169	0.932
CSNLN (L^2)	x3	32.321	0.912

Tablica 4.1: Wyniki metod uzyskane na zbiorze testowym

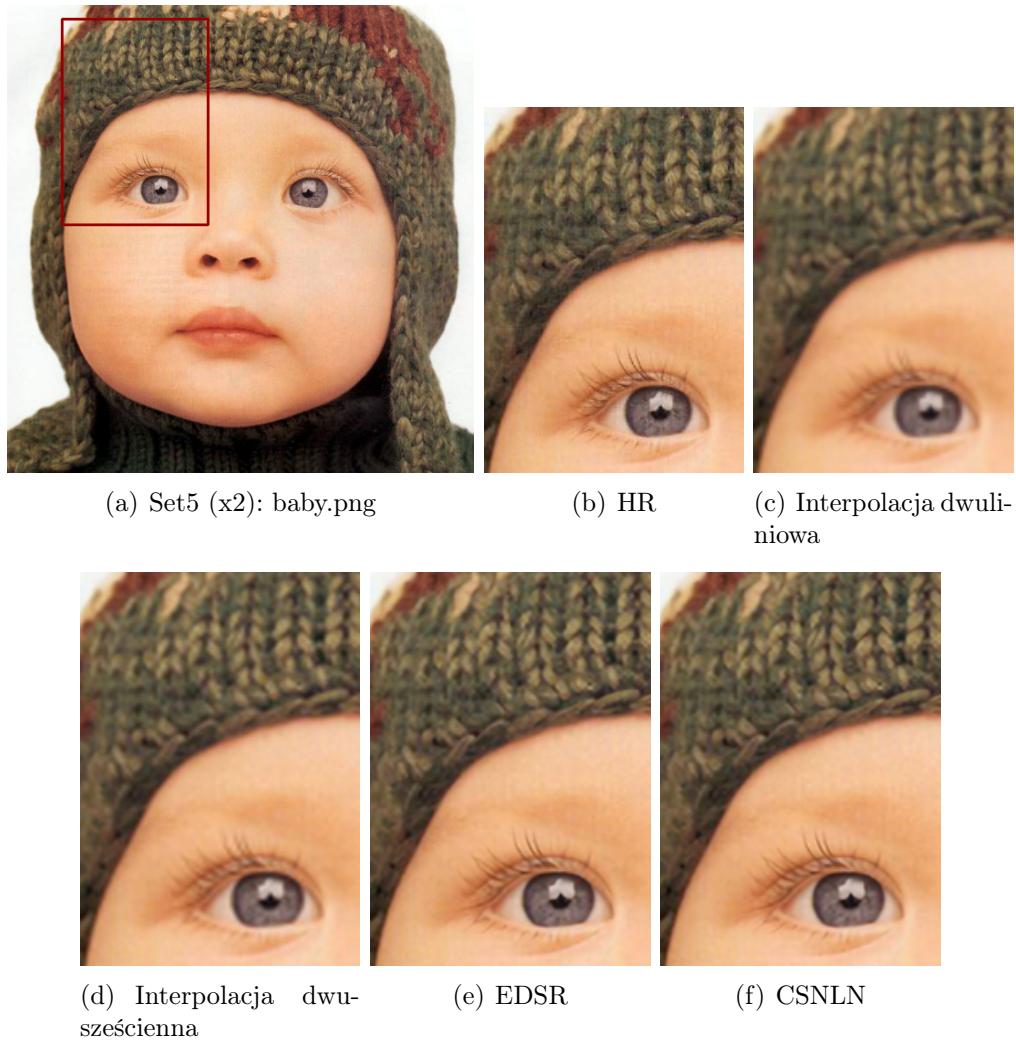
PSNR i SSIM. W publikacji PSNR i SSIM obliczane dla składowej Y w przestrzeni barw YCbCr natomiast wartości pokazane w tabeli 4.1 były obliczane w przestrzeni RGB.

Na rysunku 4.1 zaprezentowane zostały wyniki powiększenia zdjęcia ze zbioru testowego Set5 [8]. Wyniki ilustrują działanie dwóch wariantów interpolacji oraz dwóch głębokich sieci neuronowych.

Zdecydowanie najgorszy wynik był uzyskany przy użyciu interpolacji dwuliniowej, zdjęcie jest bardzo rozmyte. Nieco lepiej wygląda zdjęcie uzyskane przy pomocy interpolacji dwusześciennej, kolory wyglądają bardzo naturalnie. Punkty, w których występują skupienia takich samych pikseli, są mniej rozmyte. Jednak wynik uzyskany przy pomocy interpolacji jest nie do zaakceptowania, ponieważ nadal rozmycie jest zauważalne.

Znacznie lepiej prezentują się wyniki uzyskane przy pomocy głębokich sieci neuronowych. Pod względem ostrości obie sieci radzą skuteczniej w porównaniu do wyników uzyskanych za pomocą interpolacji. Trudno jednak stwierdzić która sieć działa lepiej wizualnie, bo różnice widoczny tylko przy dogłębnych przejrzeniu.

Tablica 4.1 zawiera wyniki próby udoskonalenie modelu przez zmianę funkcji strat na L^2 . Eksperymenty wskazują, że trenowanie z użyciem normy L^1 osiąga lepsze wyniki.



Rysunek 4.1: Porównanie wizualne dla skali x2 na zbiorze danych Set5

4.2 Rozmiary modeli

W tabeli 4.2 zaraportowane rozmiary modeli oraz ich skuteczność. Łatwo zauważać, że model CSNLN jest lżejszy niż EDSR. Potrzebuje tylko 7% rozmiaru modeli EDSR dla dwukrotnego oraz 13% dla trzechkrotnego powiększenia. Jednak do otrzymania lepszych wyników model CSNLN ma w dwa razy dłuższy czas treningu niż EDSR.

	Skala	EDSR	CSNLN
Rozmiar modelu	x2	43.68M	3.06M
Czas trenowania	x2	20 godzin	46 godzin
PSNR	x2	35.841	36.026
Rozmiar modelu	x3	43.68M	6.01M
Czas trenowania	x3	20 godzin	46 godzin
PSNR	x3	32.532	33.169

Tablica 4.2: Rozmiar modelu i porównanie wydajności dla zbioru Set5

Rozdział 5

Usuwanie szumu z obrazów

W tym rozdziale zostały zaprezentowane oraz opisane wyniki przeprowadzonych badań zdolności modeli opisanych w sekcjach 3.3, 3.4 usuwać szum z obrazów przy jednoczesnym ich powiększeniu.

5.1 Eksperiment z użyciem zbioru tylko zasumiętych obrazów

Eksperyment polegał na sprawdzeniu który z modeli lepiej rozwiązuje problem otrzymania zdjęć przy jednoczesnym ich powiększeniu. Wytrenowano sieci EDSR oraz CSNLN przy użyciu drugiego zbioru danych opisanego w sekcji 3.2. Każdy z modeli został następnie przetestowany w działaniu na zdjęciach. Wyniki tych testów demonstruje tablica 5.1.

Algorytm	Skala	PSNR	SSIM
EDSR	x2	31.808	0.845
CSNLN (L^1)	x2	31.112	0.832
CSNLN (L^2)	x2	31.423	0.838
EDSR	x3	31.681	0.860
CSNLN (L^1)	x3	31.453	0.853
CSNLN (L^2)	x3	31.723	0.867

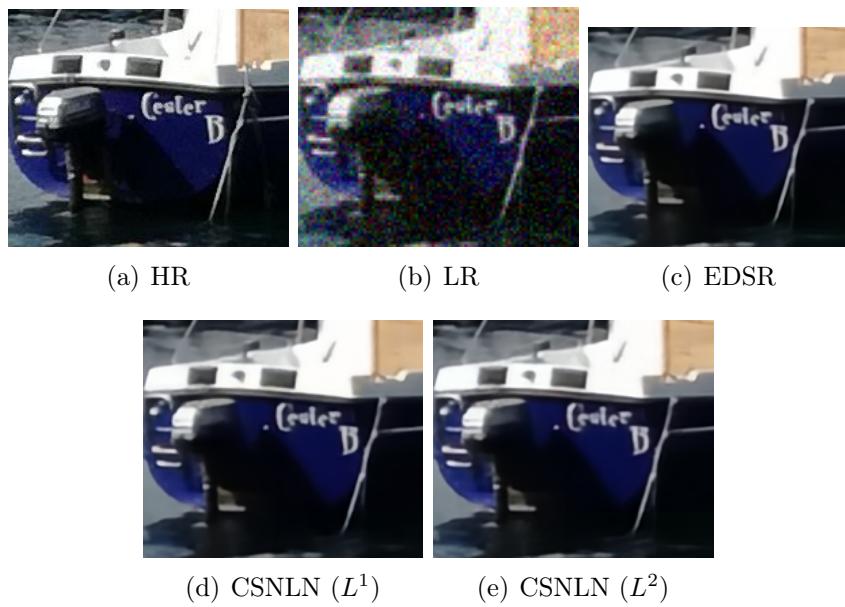
Tablica 5.1: Wyniki uzyskane na zbiorze testowym

Dla dwukrotnego powiększenia najlepsze wyniki zostały uzyskane przy pomocy modeli EDSR. Uzyskanie takich wyników może być związane z tym, że architektura EDSR ma globalne połączenie, omijające bloki rezydualne, w wyniku sieć uczy się głównie różnice pomiędzy zdjęciami HR i LR. Różnica

pomiędzy wynikami modeli EDSR i CSNLN jest niewielka. Zmiana funkcji straty korzystnie wpłynęła na rezultaty modelu CSNLN.

W przypadku trzechkrotnego powiększenia najlepsze wyniki uzyskano przy pomocy modelu CSNLN, chociaż i nie wiele lepsze niż wynik uzyskany przez model EDSR. Jak i w przypadku dwukrotnego powiększenia obrazu zmiana funkcji straty wpłynęła pozytywnie na wyniki.

Na rysunku 5.1 zostały umieszczone fragmenty z zasumionego obrazu wejściowego oraz wynikowymi obrazami, które udało się uzyskać. Całkowity obraz z którego został zrobiony wycinek dla rysunku 5.1 jest przedstawiony na rysunku 5.2. Z rysunku widać, że obie sieci dobrze poradzili z problemem usuwania szumu, jednak wyjściowym zdjęciom widocznie brakowało ostrości. Po przeanalizowaniu wyników stwierdzono, że problem mógł być związany z używanym zbiorem danych, modeli dostawały tylko zaszumione zdjęcia. Aby rozwiązać ten problem, został wykreowany rozszerzony zbiór danych o zdjęcia tylko pomniejszone opisany w sekcji 3.2. Wyniki eksperymentów na takim zbiorze zaprezentowane zostały w sekcji 5.2.



Rysunek 5.1: Porównanie wizualne wycinka ze zdjęcia



Rysunek 5.2: Całkowity obraz testowy

5.2 Eksperiment z użyciem zbioru rozszerzonego

Celem eksperymentu było wytrenowanie modeli na zbiorze zawierającym zdjęcia zaszumione i niezaszumione. W rezultacie takiego treningu oczekiwany jest model, który będzie miał wysoką zdolność do generalizacji swojej wiedzy na nowe przypadki. Przewidywano, że model musi działać poprawnie dla zdjęć zaszumionych również, jak i niezaszumiony.

Do sprawdzenia wytrenowanych modeli był użyty własny zbiór danych zawierający zdjęcia zaszumione i niezaszumione oraz wcześniej wspomniany zbiór Set5 [8]. Wyniki uzyskane przez modeli zostały zamieszczone w tabelach 5.2, 5.3.

Dla dwukrotnego powiększenia zdjęć z własnego zbioru testowego najlepsze wyniki uzyskano przy pomocy modelu EDSR. W przypadku modelu CSNLN zmiana funkcji straty nie wniosła zauważalnego rezultatu. Jednak dla trzykrotnego powiększenia model CSNLN przy wykorzystywaniu normy L^1 osiągał najlepsze wyniki na zbiorze testowym. Jak i w przypadku dwukrotnego powiększenia zmiana funkcji straty na L^2 nie doprowadziła do usprawnienia działania modelu.

Algorytm	Skala	PSNR	SSIM
EDSR	x2	34.937	0.918
CSNLN (L^1)	x2	34.661	0.916
CSNLN (L^2)	x2	34.615	0.915
EDSR	x3	29.321	0.831
CSNLN (L^1)	x3	29.418	0.834
CSNLN (L^2)	x3	29.185	0.829

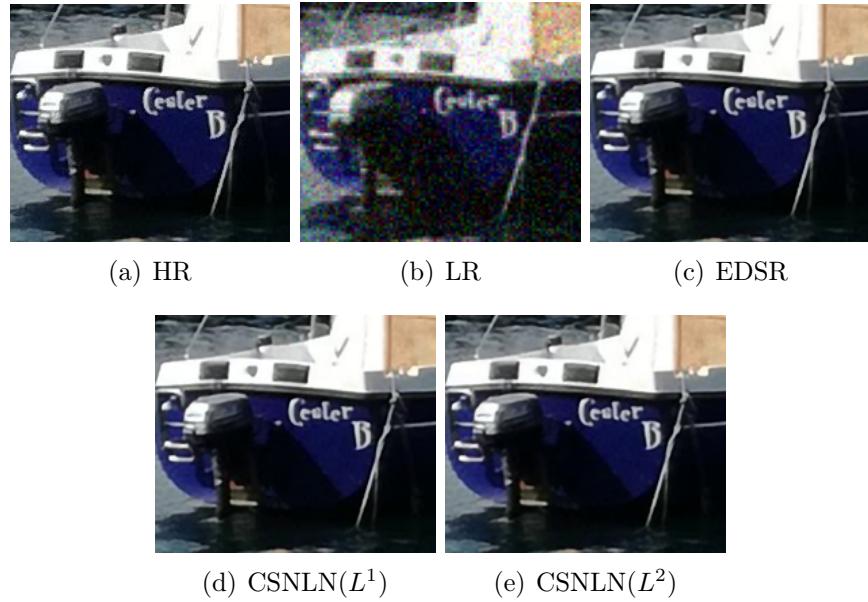
Tablica 5.2: Wyniki uzyskane na zbiorze testowym

Algorytm	Skala	PSNR	SSIM
EDSR	x2	31.899	0.938
CSNLN (L^1)	x2	32.083	0.939
CSNLN (L^2)	x2	31.997	0.937
EDSR	x3	27.478	0.840
CSNLN (L^1)	x3	27.138	0.854
CSNLN (L^2)	x3	27.503	0.858

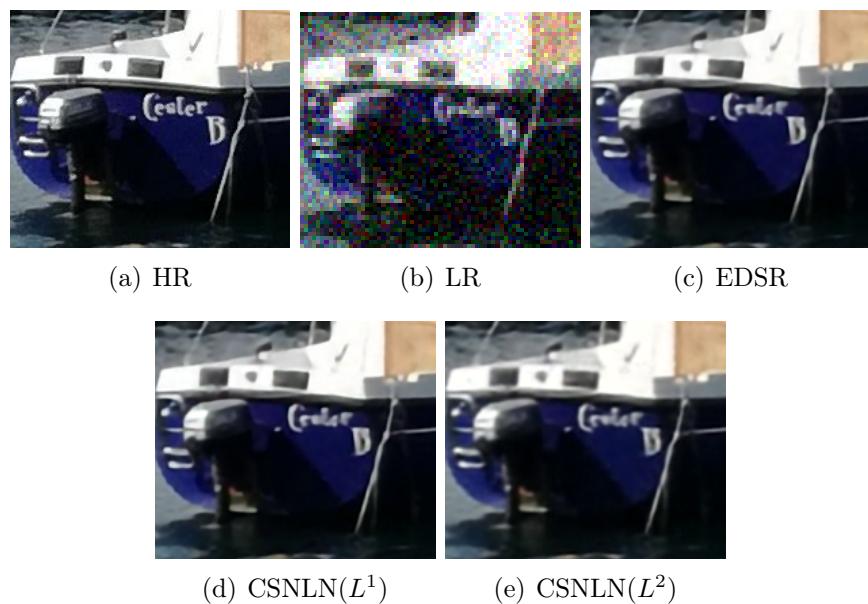
Tablica 5.3: Wyniki uzyskane na zbiorze testowym Set5

W przypadku zbioru danych Set5 [8] zadziwiające rezultaty pokazał model CSNLN uzyskano przez niego najlepsze rezultaty dla dwukrotnego oraz trzykrotnego powiększenia. Względem tabeli 4.1 z sekcji 4.1 wyniki uzyskane przez modeli wydają bardzo słabe, jednak trzeba mieć na uwadze to, że modeli były trenowane na zdjęciach nie idealnie wykonanych w przeciwieństwie do zbioru DIV2K [1]. Najprawdopodobniej jakość zdjęć wchodzących do zbioru uczącego była powodem uzyskania aż tak niskich rezultatów na zbiorze Set5 [8].

Na Rysunkach 5.3, 5.4 zaprezentowane zostały wyniki powiększania oraz oryginał zdjęcia ze zbioru testowego. Zaprezentowany został tylko wycinek ze zdjęcia bo nie miało sensu zamieszczenie całego obrazu ze względu na wyjściowy wymiar. Z obu rysunków widać potencjał oraz skuteczność stosowania głębokiego uczenia do poprawy rozdzielczości zdjęć. Jak i w przypadku modeli trenowanych na zbiorze DIV2K [1] model CSNLN najlepsze wyniki uzyskał dla większej skali powiększania co jest zgodne z artykułem naukowym [9].



Rysunek 5.3: Porównanie wizualne wycinka ze zdjęcia. Dwukrotne powiększenie.



Rysunek 5.4: Porównanie wizualne wycinka ze zdjęcia. Trzykrotne powiększenie.

Założenia że rozszerzenie zbioru uczącego o zdjęcia nie zaszumione zwiększą zdolność modelu do generalizacji zostały potwierdzone. Rysunek 5.5 zawiera wizualne porównanie potwierdzające korzyści wpływające z rozszerzenia zbioru uczącego. Otrzymany obraz jest bardzo wyraźny, precyzyjny w porównaniu do obrazu otrzymanego przy pomocy modelu uczącego się tylko na zdjęciach zasumionych



(a) $\text{CSNLN}(L^1)$ drugi zbiór danych (b) $\text{CSNLN}(L^1)$ trzeci zbiór danych

Rysunek 5.5: Porównanie wizualne zdjęcia ze zbioru Set5

Rozdział 6

Podsumowanie

W pracy udało się dogłębnie przeanalizować rozwiązania służące poprawie rozdzielczości zdjęć, szczególnie to oparte na krzyżowo-skalowej korelacji cech. Cel pracy został spełniony. Zaimplementowane zostały dwa rozwiązania bazujące na sieciach splotowych – EDSR oraz CSNLN. W ostatnim z nich została wprowadzona modyfikacja w postaci zmiany funkcji aktywacji. Sieci zostały następnie przetestowane i porównane między sobą. O ile standardowe metryki jakości obrazów wygenerowanych przez sieci były podobne do zamieszczanych w artykułach naukowych, o tyle przetestowanie modeli w bardziej naturalnych warunkach przyniosło zaskakujące rezultaty.

Wyniki działania zaimplementowanych metod na oryginalnych zdjęciach, drastycznie różnią się od tych na sztucznie pomniejszonych. Przeprowadzone eksperymenty pokazały kluczowe znaczenie zbioru uczącego. Okazało się, że najlepiej z poprawą rozdzielczości prawdziwych zdjęć radzi sobie sieć trenowana na ziorze zawierającym obrazy zaszumione oraz niezaszumione.

Przedstawione prace badawcze można kontynuować w kierunku udoskonalenia zbioru uczącego oraz poszukiwania rozwiązań lepiej radzących sobie z prawdziwymi danymi. Udoskonalenie opisanych algorytmów może odbyć się przede wszystkim przez zmianę wymiaru zdjęć wejściowych zwiększenia ilości warst oraz rozbudowanie zbioru uczącego o nowe znikształcenia.

Bibliografia

- [1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.
- [2] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015.
- [3] Muhammad Haris, Greg Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. *CoRR*, abs/1803.02735, 2018.
- [4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.
- [5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.
- [6] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016.
- [7] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. *CoRR*, abs/1707.02921, 2017.
- [8] Christine Guillemot Marco Bevilacqua, Aline Roumy and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on

- nonnegative neighbor embedding. In *Proceedings of the British Machine Vision Conference*, 2012.
- [9] Yiqun Mei, Yuchen Fan, Yuqian Zhou, Lichao Huang, Thomas S. Huang, and Humphrey Shi. Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining, 2020.
 - [10] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *CoRR*, abs/1711.07971, 2017.