Informatyka techniczna i sztuczna inteligencja
Technical informatics and Artificial Intelligence
Engineer of XXI Century'2024

The problem of fundus image segmentation using data augmentation and U-NET architecture

Krzysztof Garlacz [1], Scientific Supervisors: Vasyl Martsenyuk [2], Tomasz Zając [3], Piotr Soszka [4]

[1] *University of Bielsko-Biała, Department of Computer Science and Automatics, Master level student: s55891@student.ubb.edu.pl*
[2] *University of Bielsko-Biała, Department of Computer Science and Automatics, vmartsenyuk@ubb.edu.pl*
[3] *University of Bielsko-Biała, Department of Computer Science and Automatics, tzajac@ubb.edu.pl*
[4] *University of Bielsko-Biała, Department of Computer Science and Automatics, psoszka@ubb.edu.pl*

Abstract: The article describes the architecture of the U-Net convolutional neural network. Next, the process of creating and refining a model using this architecture was presented. A dataset of fundus images along with masks was used to train the network, but due to its small size, data augmentation was necessary. Various variants of geometric data transformations were presented to compare the effects obtained through them. The binary cross-entropy loss function was used during the training process, and later the Dice loss function was tested, as well as their combination by retraining the model with a different loss function and training from scratch using both functions simultaneously. In conclusion, the study demonstrated that various models built on the U-Net architecture and utilizing data augmentation effectively segment fundus images.

Keywords: U-Net; CNN; data augmentation; edge detection methods; image segmentation; machine learning;

Problem segmentacji obrazu dna oka z użyciem augmentacji danych i architektury U-NET

Krzysztof Garlacz [1], Opiekunowie naukowi: Vasyl Martsenyuk [2], Tomasz Zając [3], Piotr Soszka [4]

[1] *Uniwersytet Bielsko-Bialski, Wydział Budowy Maszyn i Informatyki, Informatyka, specjalizacja: techniki tworzenia oprogramowania, s55891@student.ubb.edu.pl*
[2] *Uniwersytet Bielsko-Bialski, Wydział Budowy Maszyn i Informatyki, vmartsenyuk@ubb.edu.pl*
[3] *Uniwersytet Bielsko-Bialski, Wydział Budowy Maszyn i Informatyki, tzajac@ubb.edu.pl*
[4] *Uniwersytet Bielsko-Bialski, Wydział Budowy Maszyn i Informatyki, psoszka@ubb.edu.pl*

Streszczenie: W artykule opisano architekturę konwolucyjnej sieci neuronowej U-Net. Następnie przedstawiony był proces tworzenia i doskonalenia modelu z jej wykorzystaniem. Do treningu sieci użyty został zbiór zdjęć dna oka wraz z maskami o małej liczebności, dzięki czemu konieczna była augmentacja danych. Przedstawione zostały różne warianty geometrycznej transformacji danych w celu porównania efektów dzięki nim uzyskanych. W procesie uczenia wykorzystywana była funkcja strat binarnej entropii krzyżowej, następnie sprawdzona została funkcja strat Dice'a oraz ich połączenia w postaci ponownego trenowania modelu z użyciem innej funkcji strat i uczenia od początku z użyciem dwóch funkcji jednocześnie. Podsumowując, badania wykazały, że różnorodne modele zbudowane na architekturze U-Net i wykorzystujące augmentację danych skutecznie segmentują obrazy dna oka.

Słowa kluczowe: U-Net; CNN; augmentacja danych; metody krawędziowe; segmentacja obrazu; uczenie maszynowe;

DOI: https://doi.org/10.53052/9788367652292.25

1. Wstęp

Ludzki mózg posiada naturalną zdolność do rozpoznawania obiektów na obrazach. Proces ten odbywa się automatycznie, dzięki nauce kształtów i przedmiotów od wczesnych lat dzieciństwa. W przeciwieństwie do człowieka, komputer przetwarza dany obraz jako macierz pikseli i wartości liczbowych, co oznacza, że widzi jedynie dane, a nie konkretne obiekty. W celu nauczenia komputera tej zdolności, konieczne jest zastosowanie algorytmów sztucznej inteligencji, które uczą go rozpoznawania kształtów i obiektów poprzez proces treningu i korekcji.
Segmentacja obrazu jest kluczowym etapem analizy obrazów, polegającym na dzieleniu obrazów na mniejsze elementy w oparciu o kryteria. W kontekście medycyny, na przykład w badaniach dna oka, segmentacja umożliwia w nieinwazyjny sposób wykryć choroby zapalenia błony naczyniowej, jaskry, uszkodzenia siatkówki czy cukrzycy. Dzięki wykorzystaniu sztucznej inteligencji proces ten może zostać zautomatyzowany, co przyspiesza analizę obrazów, redukuje błędy ludzkie oraz wykrywa zmiany, które mogą być pominięte przez specjalistów.

1.1 Segmentacja obrazów

Proces polegający na dzieleniu obrazu na mniejsze, jednorodne fragmenty, które ułatwiają identyfikację i analizę poszczególnych obiektów i struktur. W wyniku segmentacji obraz zostaje uproszczony, dzięki czemu nie zawiera wielu szczegółowych informacji będących w obrazie źródłowym. Przykładowymi kryteriami podziału mogą być poziom szarości, kolor lub tekstura. Dodatkowo segmentacja może być wykorzystana do wykrywania krawędzi na obrazie. Znajduje ona zastosowanie w szeregu wielu dziedzin, w takich jak medycyna, gdzie pozwala na automatyczne wykrywanie zmian patologicznych na obrazach diagnostycznych, przetwarzaniu obrazów i systemach rozpoznawania obiektów.
Klasyfikacja metod segmentacji obrazów opiera się na rodzaju informacji wykorzystywanej podczas procesu segmentacji i obejmuje kilka głównych kategorii. Metody punktowe, takie jak progowanie, polegają na doborze progu na podstawie histogramu obrazu, co skutkuje uzyskaniem obrazu binarnego. Wśród metod klasteryzacyjnych, cechy pikseli są grupowane w obszary na podstawie algorytmów klasteryzacji, co pozwala na identyfikację i wydzielenie homogenicznych regionów. Metody krawędziowe wykorzystują algorytmy wykrywania krawędzi do określenia granic obiektów, co jest kluczowe w analizie kształtów. Z kolei metody obszarowe, takie jak rozrost obszarów, łączenie obszarów, podział obszarów oraz segmentacja wododziałowa, koncentrują się na pracy z większymi regionami w obrazie. Dodatkowo, metody hybrydowe, które łączą dwie lub więcej z powyższych technik, takie jak rozrost obszarów z informacjami o krawędziach, oferują elastyczne podejście do segmentacji, dostosowując się do specyficznych potrzeb analizy obrazów.
Spośród przedstawionych grup metod szczególną uwagę poświęcono metodom krawędziowym. Wybór ten wynika z ich szerokiego zastosowania w praktyce, zwłaszcza w analizie obrazów, gdzie precyzyjne wyznaczanie granic obiektów odgrywa kluczową rolę w kolejnych etapach przetwarzania, takich jak klasyfikacja czy rozpoznawanie. Ponadto metody krawędziowe cechuje stosunkowo prosta implementacja oraz wysoka skuteczność w wykrywaniu struktur na obrazach różnego rodzaju. Szczegółowe omówienie tych metod w dalszej części tekstu pozwala lepiej zrozumieć ich możliwości oraz ograniczenia w kontekście praktycznych zastosowań.

1.2 Metody krawędziowe

Podejście wykorzystujące gwałtowną zmianę luminancji do wykrywania krawędzi na danym obrazie. Może to być spowodowane zmianą głębokości, orientacji powierzchni, właściwości materiału oraz różnorodnością oświetlenia scen. Najczęściej stosowane metody to operator Sobela, Prewitta krzyżowy Robersta oraz algorytm Canny'ego, które wykorzystują obliczenia gradientów intensywności. Przykładowo, filtr Robersta oblicza wartość piksela obrazu wynikowego na podstawie macierzy 2x2, co umożliwia uzyskanie gradient ukośny. Następnie pierwiastkuje się oraz sumuje kwadraty różnic elementów po przekątnej, co służy do obliczenia intensywności krawędzi danego piksela. Kolejnym przykładem operatora gradientowego jest operator Sobela, który wykorzystuje dwie macierze o rozmiarze 3x3. Pierwsza macierz, przeznaczona do wykrywania pionowych krawędzi, ma liczby ujemne w pierwszej kolumnie, zera w drugiej oraz dodatnie w trzeciej, co dzieli ją na części o przeciwnych wartościach. Druga macierz, służąca do wykrywania poziomych krawędzi, zawiera zera w środkowej kolumnie oraz przeciwne wartości w pozostałych kolumnach. Po obliczeniu gradientów w procesie konwolucji, wartości są podnoszone do kwadratu, sumowane i pierwiastkowane, co pozwala uzyskać intensywność krawędzi w danym pikselu.

1.3 U-Net

Współczesne zastosowania segmentacji obrazów coraz częściej opierają się na metodach głębokiego uczenia, wśród których szczególnie wyróżnia się sieć U-Net. Ta zaawansowana architektura skutecznie integruje zalety klasycznych metod segmentacji, omówionych w rozdziale 1.1, z możliwościami nowoczesnych technik opartych na sieciach neuronowych.
Architektura U-Net to szczególny typ konwolucyjnej sieci neuronowej, powszechnie stosowanej w zadaniach segmentacji obrazów. Jej nazwa pochodzi od charakterystycznego kształtu przypominającego literę „U", wynikającego z symetrycznego układu warstw. Struktura U-Net dzieli się na dwie główne części: kodującą i dekodującą, które są ze sobą powiązane poprzez warstwy pośrednie. Część kodująca przetwarza dane wejściowe, redukując ich rozdzielczość, ale jednocześnie zwiększając liczbę filtrów konwolucyjnych. Każdy poziom w tej części zawiera pary warstw konwolucyjnych, często uzupełniane o warstwy dropout, a kończy się warstwą pooling, odpowiedzialną za zmniejszenie rozdzielczości. Im niższy poziom kodowania, tym liczba filtrów rośnie, co pozwala na uchwycenie coraz bardziej złożonych cech obrazu. Część dekodująca U-Net, rozpoczynająca się po warstwie podstawowej, ma na celu odwzorowanie oryginalnej rozdzielczości obrazu przy jednoczesnym zachowaniu informacji o cechach wykrytych w części kodującej. Kluczowym elementem jest warstwa upsampling, która zwiększa rozdzielczość obrazu, działając odwrotnie niż warstwa pooling z części kodującej. Następnie stosowana jest warstwa konkatenacji, która łączy odpowiednie poziomy kodujące i dekodujące, umożliwiając przywrócenie informacji przestrzennych utraconych podczas procesu kodowania. Kolejne warstwy konwolucyjne stopniowo przywracają obraz do wyższej rozdzielczości, a końcowy wynik segmentacji uzyskiwany jest dzięki warstwie konwolucyjnej o jednym filtrze o rozmiarze [1], która stanowi ostatni etap działania sieci U-Net.

1.4 Dostępne rozwiązania

Implementację modelu wykorzystującego architekturę U-Net wykonać można korzystając z różnych narzędzi. Podczas badań zastosowana była kombinacja TensorFlow z Keras. Pierwsze z nich, to popularny framework, służący do pracy z uczeniem maszynowym, drugie natomiast jest wysokopoziomowym API, udostępniającym przystępne metody, korzystające z funkcji TensorFlow. Połączenie to dostarcza dużo możliwości dostosowywania modelu do wymagań, zachowując jednocześnie czystość kodu i prostotę użycia.
Alternatywą dla wykorzystanego rozwiązania było użycie PyTorch. Ten framework również jest prosty w użyciu, jednak nie w takim stopniu, jak TensorFlow z Keras. Pozwala on natomiast na większą kontrolę nad procesem uczenia. Trzecim możliwym rozwiązaniem było MMSegmentation. Jest ono mniej popularne od wcześniej wymienionych. Jest to narzędzie stworzone głównie do zadań dotyczących segmentacji obrazów. Posiada wbudowane implementacje wielu popularnych typów modeli.

2. Opis metody

Na podstawie architektury U-Net zaprojektowany został model sieci neuronowej [1 – 4]. Następnie poddany był procesowi uczenia z wykorzystaniem podstawowego zestawu danych w celu wstępnego przetestowania jego możliwości. Kolejne etapy polegały na uczeniu modelu przez określoną liczbę epok i sprawdzaniu wyników oraz przeprowadzaniu augmentacji danych o różnym zakresie w celu uzyskania możliwości porównania pomiędzy różnymi jej wersjami. Wykorzystana augmentacja bazowała na podstawowych operacjach przeprowadzanych na obrazach [5, 6], których przykładami są rotacja zdjęć oraz tworzenie lustrzanych odbić. Oprócz augmentacji zastosowany został także dropout, których wspólnym celem było zredukowanie prawdopodobieństwa wystąpienia problemu niedouczenia modelu oraz nadmiernego jego dopasowania [7].

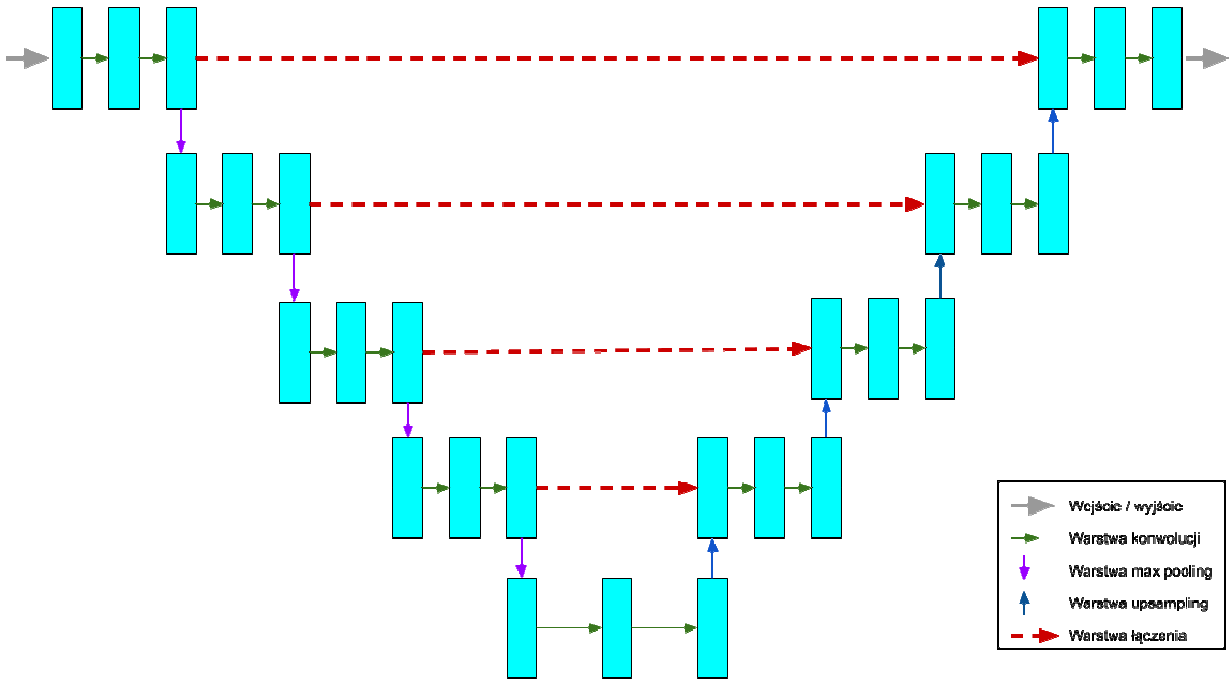
2.1. Opis danych treningowych

Zdjęcia dna oka wraz z odpowiadającymi im maskami, wykorzystane jako dane treningowe, pochodzą z zestawu STARE („Structured Analysis of the Retina") [8] dostępnego na stronie Papers With Code [9]. Składa się on z dwudziestu kolorowych zdjęć o wymiarach 700x605 pikseli oraz takiej samej liczby ręcznie utworzonych masek na ich podstawie.

2.2. Architektura zaproponowanego podejścia U-Net

Model sieci neuronowej utworzony został z wykorzystaniem architektury U-Net [10, 11]. Część kodująca składa się z pięciu bloków. Na każdym z nich umieszczone są warstwy: konwolucyjna, dropout oraz druga konwolucja. Pierwsze cztery bloki posiadają dodatkowo warstwy max pooling i połączone są z odpowiadającymi im blokami części dekodującej.

Część dekodująca składa się z czterech bloków. Każdy z nich zawiera warstwy: upsampling, łączenia, konwolucyjnej, dropout oraz drugiej konwolucyjnej. Wszystkie wymienione warstwy konwolucyjne jako funkcje aktywacji wykorzystują standardowe ReLU [12], a ich filtry mają rozmiar 3x3. Liczba tych filtrów rozpoczyna się od 16 na poziomie pierwszych bloków i wzrasta wykładniczo, podwajając się na każdym z niższych poziomów.
Opisana powyżej architektura przedstawiona została na Rysunku. 1. Warstwy konwolucyjne zaznaczone są na nim zielonymi strzałkami, fioletowe oznaczają max pooling, niebieskie upsampling, a na czerwono przedstawione są połączenia pomiędzy blokami warstwy kodującej, a odpowiadającymi im blokami warstwy dekodującej.



Rysunek 1. Architektura sieci neuronowej

Ostatnim elementem sieci jest dodatkowa warstwa konwolucyjna, która wykorzystuje sigmoidalną funkcję aktywacji. Dzięki temu wartości wynikowe przekształcane są na zakres od 0 do 1, co pozwala na ich interpretację jako wyniki działania sieci neuronowej.
Implementacja opisanej architektury wykonana została na podstawie poniższego pseudokodu, odpowiadającego za utworzenie modelu sieci neuronowej. Uwzględnione w nim zostały przede wszystkim informacje dotyczące warstw znajdujących się w poszczególnych blokach na każdym poziomie.

```
część_kodująca = dla 4 poziomów:
{
    warstwa_konwolucyjna()
    dropout()
    warstwa_konwolucyjna()
    warstwa_max_pooling()
}

część_kodująca += ostatni poziom części kodującej
{
    warstwa_konwolucyjna()
    dropout()
    warstwa_konwolucyjna()
}

część_dekodująca = dla 4 poziomów:
{
    warstwa_upsampling()
    warstwa_łączenia(część_kodująca[obecny_poziom])
    warstwa_konwolucyjna()
    dropout()
```

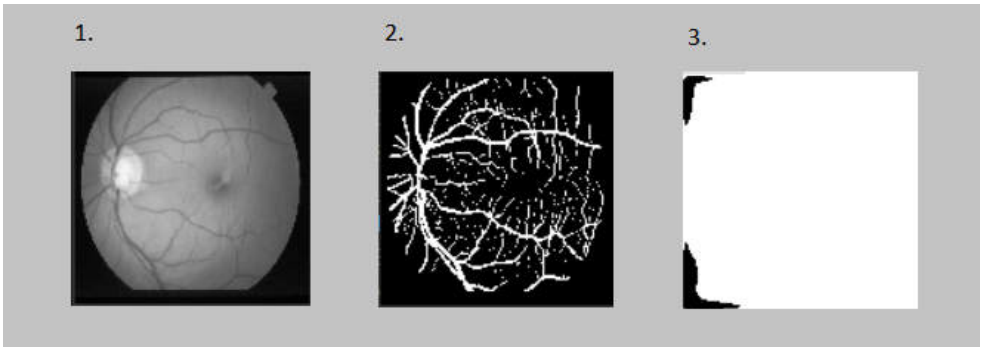
```
    warstwa_konwolucyjna()
}

wynik = warstwa_konwolucyjna(funkcja_aktywacji: sigmoidalna)

model = { część_kodująca, część_dekodująca, wynik }
```

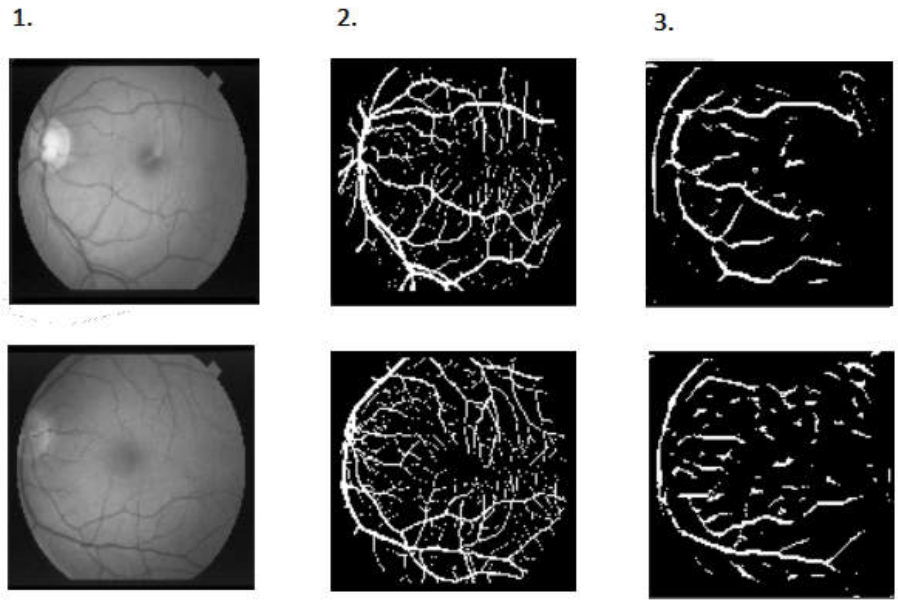
2.3. Proces treningowy

Bazową funkcją strat wykorzystaną w modelu była binarna entropia krzyżowa. W pierwszej kolejności, by przetestować utworzony model, wykorzystany został zestaw 20 zdjęć wraz z ich maskami, a sam proces uczenia obejmował tylko 20 epok. Na rysunku 2 w pierwszej kolejności znajduje się czarnobiałe zdjęcie dna oka. Środkowe przedstawia jego maskę, natomiast ostatnie jest wynikiem zwróconym przez wytrenowany model dla pierwszego zdjęcia.



Rysunek 2. Pierwszy test modelu

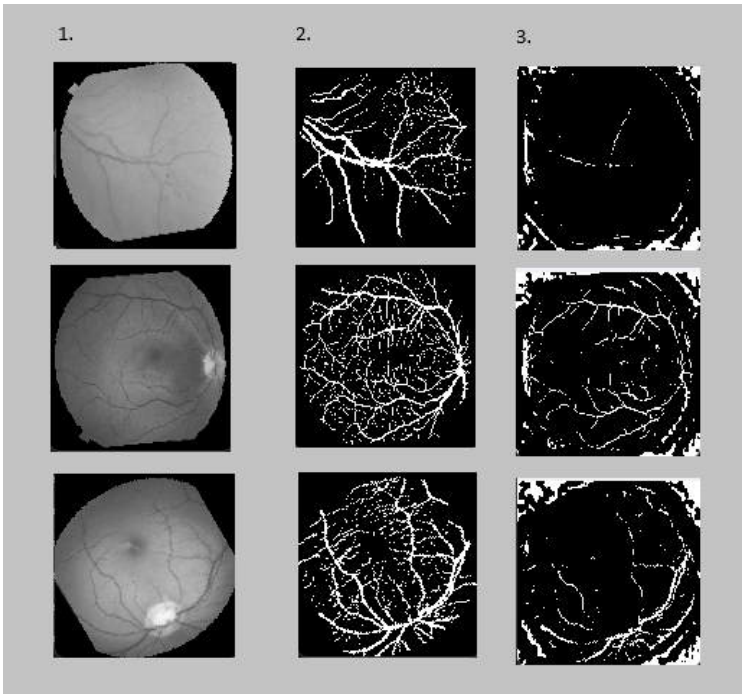
Przy tak małej ilości epok oraz danych treningowych, poprawne wytrenowanie modelu nie jest możliwe, jednak celem tego treningu było przetestowanie działania modelu w najkrótszym możliwym czasie.
Gdy poprawność modelu została potwierdzona, model przetestowany został na większej ilości epok, by sprawdzić możliwości treningu przy znikomej ilości danych testowych. Rysunek 3. przedstawia zdjęcie dna oka, jego maskę oraz wynik przewidywany przez model trenowany 4000 epok.



Rysunek 3. Brak augmentacji, 4000 epok

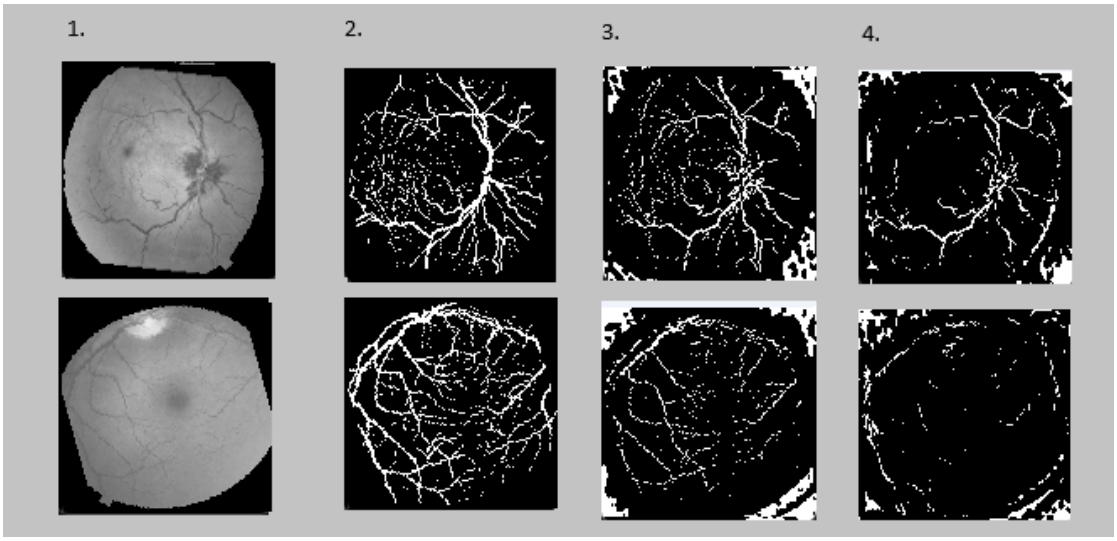
Jak widać na rysunku 3. model był w stanie wykryć parę większych naczyń oraz niewielkie fragmenty średnich. Na zdjęciach znajdują się także fragmenty obrysu samego oka.
W następnym kroku wykorzystana została pierwsza augmentacja. W celu utworzenia większej ilości danych treningowych, na posiadanych zdjęciach oraz ich maskach wykonywane były operacje mające na celu utworzenie odbić lustrzanych względem osi x, y oraz obu jednocześnie. Dzięki temu z 20 początkowych zdjęć udało się uzyskać 80. Same odbicia nie dały jednak zadowalającej liczebności, dlatego każde zdjęcie zostało obrócone o losową wartość z przedziału

od -10 do 10 stopni. Czynność ta była powtarzana jeszcze 9 razy, za każdym razem mnożąc zakresy obrotu przez numer iteracji, dzięki czemu finalnie uzyskanych zostało 880 zdjęć z maskami.
Wykorzystując nowe dane treningowe uzyskane w drodze augmentacji, model trenowany był przez 1000 epok. Rysunek 4. przedstawia wyniki uzyskane przez ten model. Zauważyć można znaczącą różnicę pomiędzy jakością przewidywanych masek różnych zdjęć.



Rysunek 4. Pierwsza augmentacja, 1000 epok

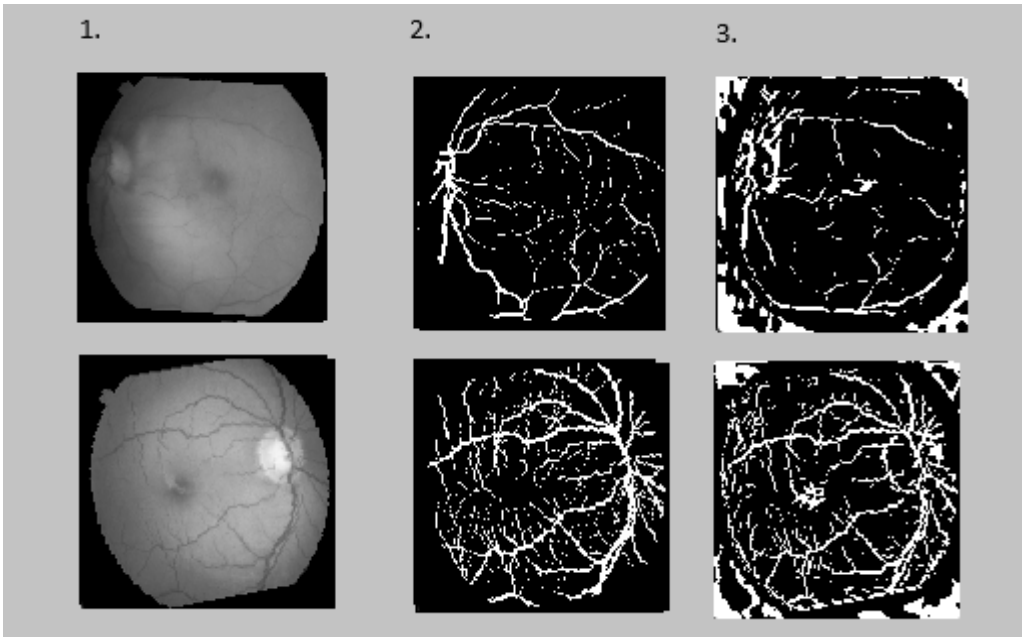
By uzyskać lepszy punkt odniesienia model wytrenowany został jeszcze raz z tymi samymi danymi treningowymi, jednak tym razem nauka trwała 4000 epok. Rysunek 5. przedstawia efekty tej nauki. Dodatkowa kolumna numer 4. zawiera wyniki modelu z 1000 epok. Zauważalne jest większe wykrycie naczyń w przypadku 4000 epok, co oznacza, że trenowanie modelu z taką ich liczbą przynosiło lepsze efekty względem 1000. Minusem tego rozwiązania był jednak mocno wydłużony czas treningu.



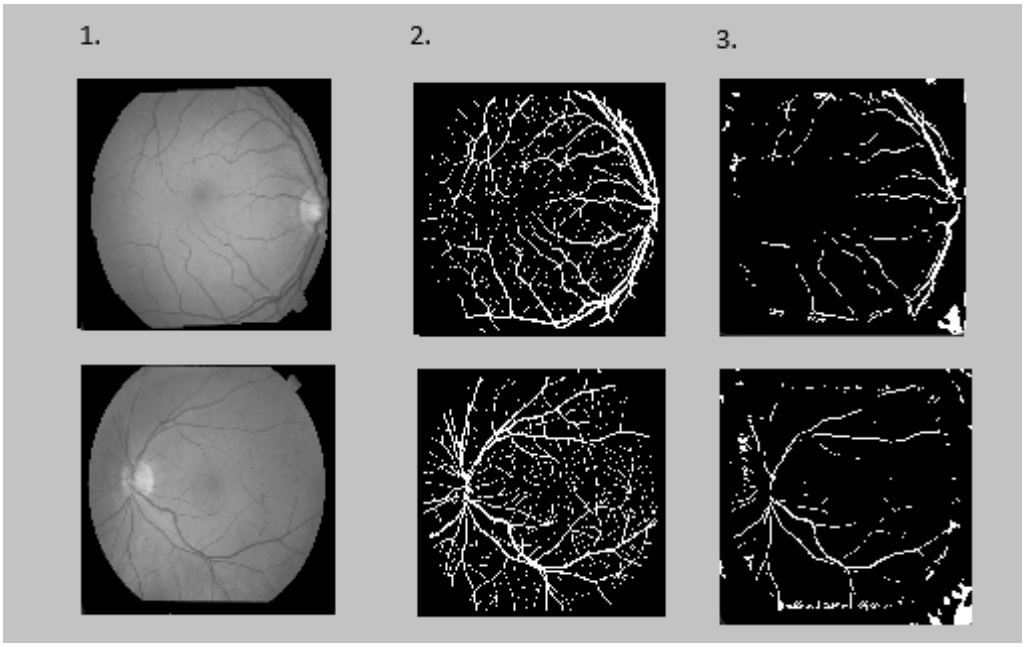
Rysunek 5. Pierwsza augmentacja, porównanie 4000 epok z 1000

Do uczenia kolejnego modelu zmieniona została maksymalna wartość obrotu zdjęć dna oka. Dotychczasowy mnożnik został zmieniony tak, by numer iteracji zwiększony o jeden był pomnożony przez dwa, zamiast dziesięciu. Zabieg ten miał zapobiec zniekształcaniu zdjęć oraz masek podczas rotacji. Następujące po sobie rysunki 6 oraz 7 przedstawiają kolejno wynik modelu uczonego przez 1000 epok oraz 4000 epok.

DOI: https://doi.org/10.53052/9788367652292.25



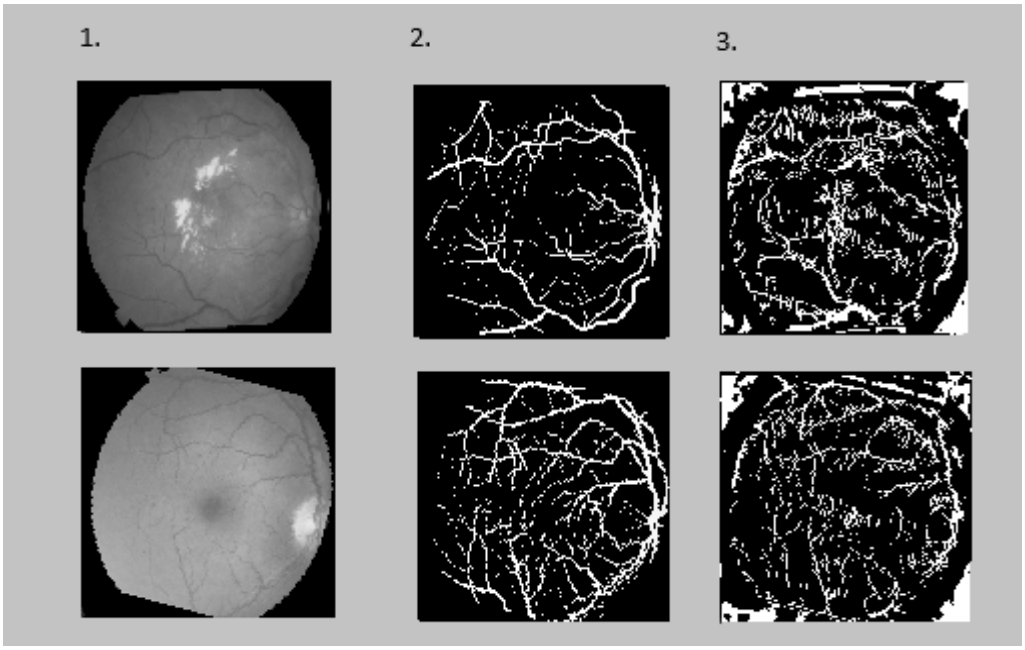
Rysunek 6. Augmentacja z obrotem o małe wartości, 1000 epok



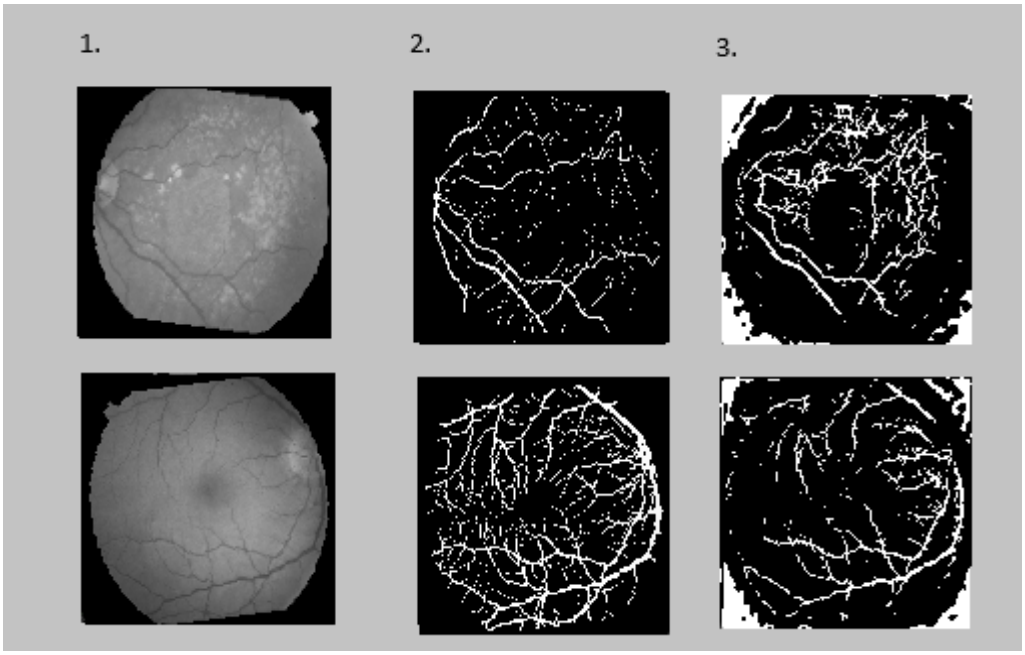
Rysunek 7. Augmentacja z obrotem o małe wartości, 4000 epok

Po porównaniu ze sobą różnic w efektach modelu uczonego na 1000 epok i 4000, dłużej uczony model był bardziej ostrożny przy oznaczaniu naczyń, przez co zaznaczył ich mniej oraz widać więcej przerw w zaznaczonych liniach, co było przeciwieństwem krócej uczonego modelu – zaznaczał on więcej elementów, jednak w dużej mierze był podatny na szum.

Kolejna zmiana w augmentacji miała na celu uzyskanie większej liczby zdjęć o jakości poprzedniej wersji. By tego dokonać, zmieniony został zakres, w którym rotowane były zdjęcia. W nowej wersji za każdym razem, gdy zdjęcie miało zostać obrócone, było obracane tylko w jedną stronę, a następnie drugi raz w przeciwną. Dzięki temu uzyskane zostało dwa razy tyle zdjęć. Rysunek 8 przedstawia wynik uczenia modelu przez 1000 epok korzystając z nowych danych treningowych, a rysunek 9 efekt wersji z 4000 epok.



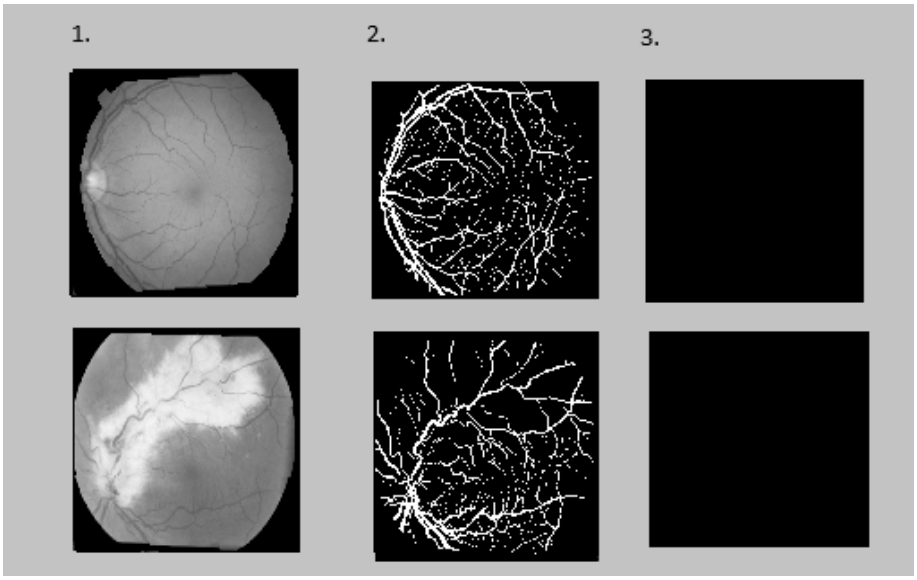
Rysunek 8. Największa liczba zdjęć, 1000 epok



Rysunek 9. Największa liczba zdjęć, 4000 epok

Wersja przedstawiona na rysunku 9 radzi sobie najlepiej z wykrywaniem naczynek obecnych na zdjęciach dna oka. Wersja ta korzystała z najbardziej rozbudowanej bazy zdjęć danych treningowych oraz uczenie trwało przez największą liczbę epok, dzięki czemu model radził sobie najlepiej. Nie jest to jednak wersja całkiem pozbawiona wad – obszary, gdzie na zdjęciu nie znajduje się żaden element oka zaznaczone zostały naczynia.
W celu eliminacji zaznaczania rogów obrazu jako naczyń obecnych w oku, do trenowania modelu wykorzystana została funkcja strat Dice'a. Niestety model nie był w stanie poradzić sobie ze swoim zadaniem, co przedstawione zostało na rysunku 10.

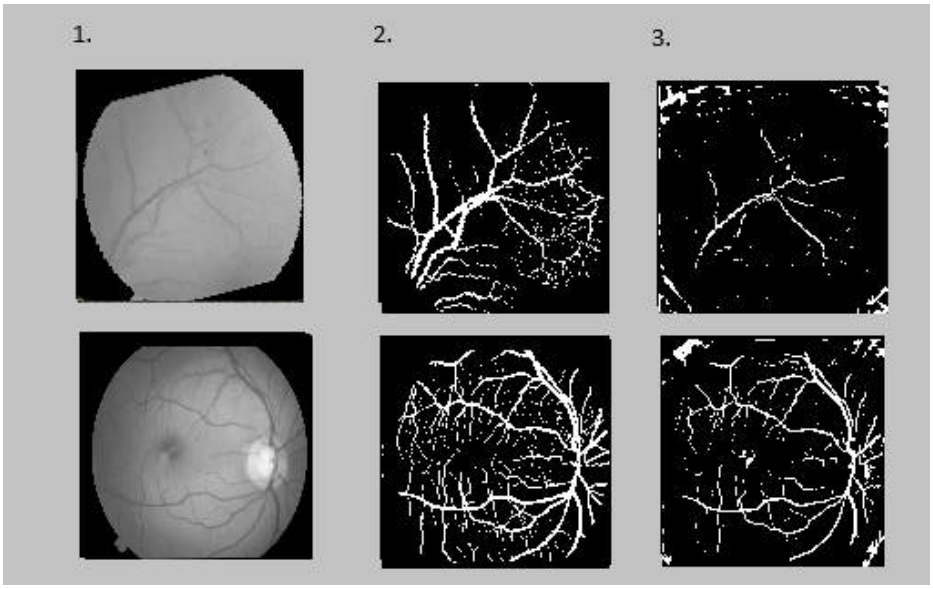
DOI: https://doi.org/10.53052/9788367652292.25



Rysunek 10. Funkcja strat Dice'a

Niepowodzenie związane z wykorzystaniem funkcji strat Dice'a spowodowane było w dużej mierze rygorystyczną naturą samej funkcji oraz jakością danych treningowych. Gdy obrazy dna oka wraz z ich maskami skalowane były do rozmiarów kwadratów o boku 128 pikseli, na maskach obecne były pozostałości po mniejszych naczynkach, które zostały ledwo widoczne. Oznacza to, że pomimo bycia funkcją utworzoną specjalnie na potrzeby segmentacji obrazu, nie była ona jednak w stanie poradzić sobie w przypadku małej powierzchni elementów docelowych oraz dużej ilości szumu na maskach zdjęć.

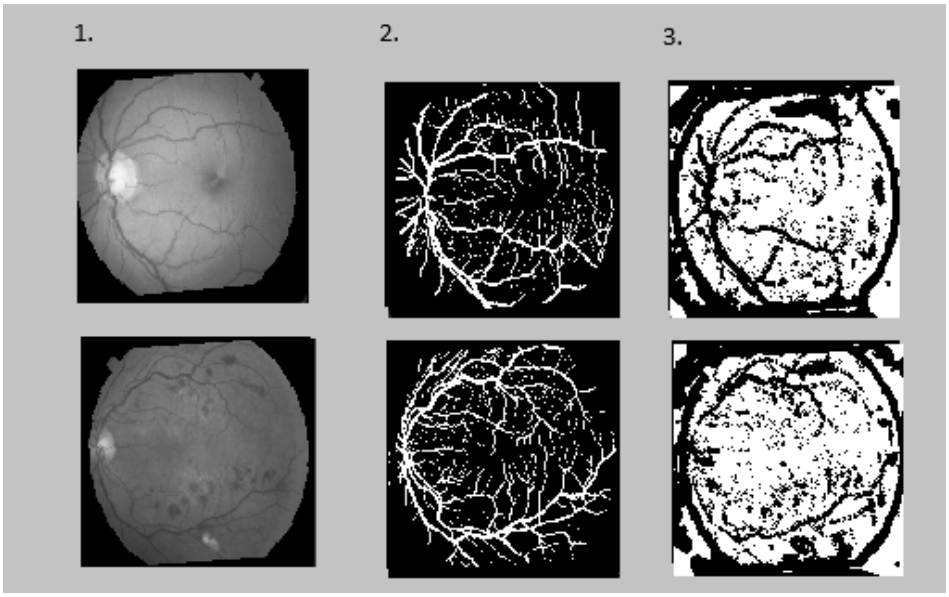
Problem wykorzystania funkcji strat Dice'a nie pozwalał na naukę modelu, ze względu na jego zbyt małą tolerancję błędu przy problemie segmentacji obrazu dna oka, dlatego wykorzystany został nauczony już wcześniej model z najnowszą wersją augmentacji danych oraz 4000 epok, jako baza, która następnie została ponownie poddana uczeniu, tym razem z wykorzystaniem funkcji strat Dice'a i dodatkowego 1000 epok. Na rysunku 11, gdzie przedstawione zostały efekty dodatkowego treningu z nową funkcją strat, zauważalne są braki w zaznaczeniach obecnych wcześniej w narożnikach zdjęć, co oznacza, że dodatkowy tysiąc epok z funkcją strat Dice'a wpłynął na nie korzystnie.



Rysunek 11. Ponownie trenowany model z funkcją strat Dice'a

Ostatnie podejście do problemu polegało na wykorzystaniu jednocześnie dwóch funkcji strat do nauki nowego modelu. Wykorzystana została binarna entropia krzyżowa, której wynik był odpowiedzialny za 60% całego wyniku wspólnej funkcji strat, a do obliczania pozostałej części wykorzystana była funkcja Dice'a. Przedstawiony poniżej rysunek 12, jest wynikiem uczenia modelu dwoma funkcjami strat jednocześnie. Model ten nieoczekiwanie rozpoznawał głównie

obszar przeciwny do oczekiwanego, jednak po zanegowaniu wyniku widać, że do grupy z naczyniami oka zakwalifikował się również obrys samego oka.



Rysunek 12. Połączona funkcja strat

W zamieszczonych poniżej tabelach od 1 do 3 przedstawione zostały wyniki uzyskane przez dające wizualnie najlepsze wyniki modele. Zostały one utworzone w celu matematycznego porównania efektów treningu przy zastosowaniu różnych parametrów.

Tabela 1. Pierwsze porównanie najlepszych modeli

Model	Precyzja	Czułość	Wynik F1	Procent wykrytych	Procent zanegowanych
Pierwsza augmentacja, 4000 epok	0,40	0,45	0,42	44,64	88,12
Druga augmentacja, 4000 epok	0,68	0,28	0,39	27,60	97,64
Trzecia augmentacja, 4000 epok	0,37	0,47	0,41	46,71	85,55
Trzecia augmentacja, ponowny trening z funkcją Dice'a, 1000 epok	0,54	0,57	0,55	56,80	91,19
Trzecia augmentacja, połączone funkcje strat, 2000 epok	0,12	0,49	0,19	48,86	33,06

Tabela 2. Drugie porównanie najlepszych modeli

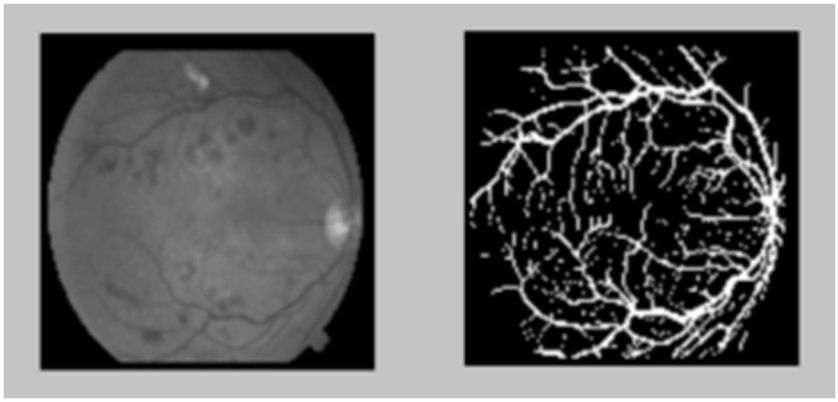
Model	Precyzja	Czułość	Wynik F1	Procent wykrytych	Procent zanegowanych
Pierwsza augmentacja, 4000 epok	0,44	0,50	0,46	49,52	90,31
Druga augmentacja, 4000 epok	0,78	0,35	0,48	35,10	98,48
Trzecia augmentacja, 4000 epok	0,39	0,47	0,43	47,48	88,59
Trzecia augmentacja, ponowny trening z funkcją Dice'a, 1000 epok	0,63	0,59	0,61	58,52	94,72
Trzecia augmentacja, połączone funkcje strat, 2000 epok	0,09	0,42	0,14	41,94	31,95

Tabela 3. Trzecie porównanie najlepszych modeli

Model	Precyzja	Czułość	Wynik F1	Procent wykrytych	Procent zanegowanych
Pierwsza augmentacja, 4000 epok	0,21	0,38	0,27	37,80	90,85
Druga augmentacja, 4000 epok	0,38	0,21	0,27	20,56	97,81
Trzecia augmentacja, 4000 epok	0,19	0,35	0,24	34,88	90,12
Trzecia augmentacja, ponowny trening z funkcją Dice'a, 1000 epok	0,41	0,54	0,47	54,13	95,01
Trzecia augmentacja, połączone funkcje strat, 2000 epok	0,06	0,58	0,10	57,76	36,17

3. Rezultaty

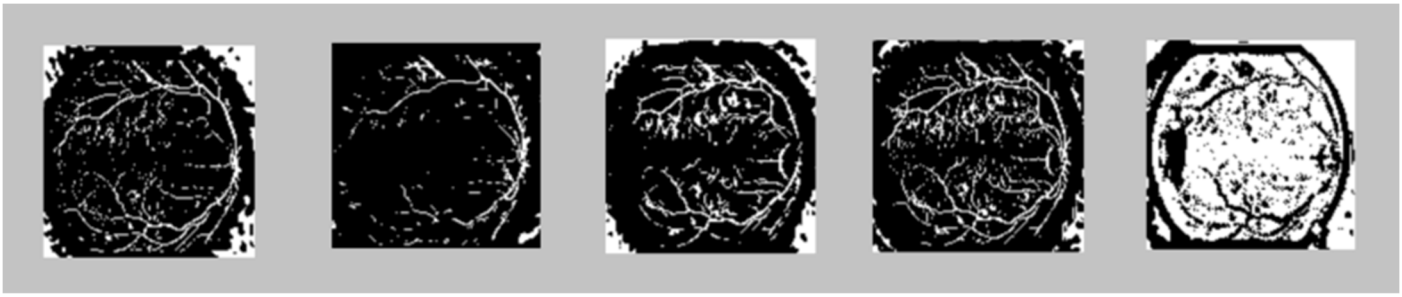
Efekty uzyskane z użyciem wybranych modeli powstałych podczas badań zostały zestawione ze sobą w trzech próbach. Celem takiego zestawienia było umożliwienie porównania wyniku oraz jakości poszczególnych modeli na podstawie segmentacji tych samych obrazów. Rysunek 13 przedstawia obraz oraz przypisaną mu maskę, które losowo wybrane zostały do pierwszego testu, natomiast rysunek 14 jest wynikiem tego porównania.



Rysunek 13. Obraz i maska pierwszego porównania

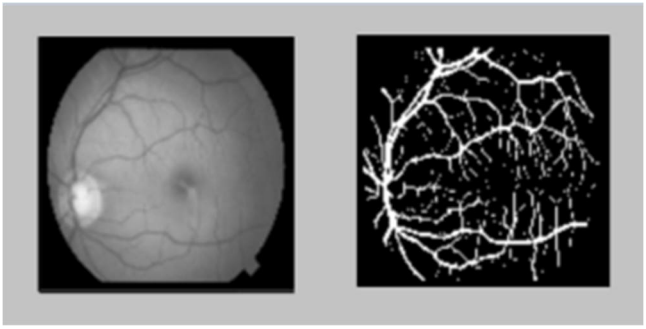
Widoczne na rysunku 14 efekty segmentacji dokonane są kolejno przez modele:

- Pierwsza augmentacja, 4000 epok
- Druga augmentacja (zmniejszenie zakresu rotacji zdjęć), 4000 epok
- Trzecia augmentacja (zwiększenie liczby zdjęć), 4000 epok
- Trzecia augmentacja, ponownie uczony model z wykorzystaniem funkcji strat Dice'a, 1000 epok
- Trzecia augmentacja, połączone funkcje strat, 2000 epok



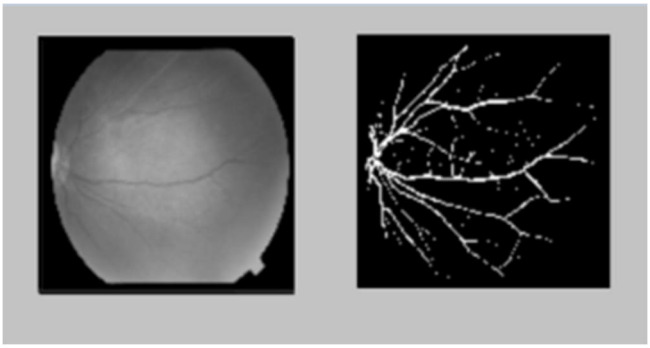
Rysunek 14 Pierwsze porównanie modeli

Druga próba przedstawiona została na rysunkach 15 i 16 zachowując tą samą kolejność modeli. Ten sam algorytm użyty został przy zamieszczonych poniżej rysunkach 17 oraz 18.



Rysunek 15. Obraz i maska drugiego porównania

Rysunek 16. Wynik drugiego porównania



Rysunek 17. Obraz i maska trzeciego porównania

Rysunek 18. Wynik trzeciego porównania

Powyższe zestawienie ukazało, że wyniki segmentacji uzyskane z użyciem modelu trenowanego przez 4000 epok były jednymi z najdokładniejszych. W rogach zdjęć zaznaczone były obszary nie należące do oka, a w środku znajdował się szum, jednak wynik ogólny był bardzo dobry.
Wprowadzenie drugiej wersji augmentacji spowodowało usunięcie większości zaznaczonego obszaru w rogach obrazu, jednak kosztem wykrycia mniejszej ilości naczyń. Ilość szumu też była mocno zmniejszona, jednak pojawiały się krawędzie oka.

Zwiększenie liczby zdjęć spowodowało powrót zaznaczenia w rogach zdjęć oraz zwiększenie ilości wykrytych naczyń. Zniknął też obrys obecny w wersji z mniejszym zestawem danych treningowych. Wyniki tego modelu były ciężkie do porównania z pierwszą wersją augmentacji, ponieważ w zależności od zdjęcia wykrywał więcej naczynek, lub dostarczał więcej szumu.
Ponowne trenowanie modelu z wykorzystaniem funkcji strat Dice'a poprawiło efekty otrzymane przez model bazowy. W rogach obrazu zaznaczone zostało mniejsze pole, pojawiło się natomiast więcej szumu, jednak ilość wykrytych naczyń została zwiększona.
Ostatni model trenowany był najkrócej ze wszystkich, jednak był w stanie wykryć elementy znajdujące się na zdjęciach. Działał on w sposób odwrotny do reszty modeli, co dało efekt zanegowania ich efektów. Poza wykrytymi naczyniami, zaznaczony został także obrys oka oraz zakłócenia w formie plam.

4. Kierunki dalszych badań

Ze względu na ograniczone możliwości obliczeniowe i wymagane trenowanie dużej liczby modeli o różnych parametrach, dane wejściowe skalowane były do minimalnych rozmiarów. W dalszych badaniach, korzystając z wiedzy zdobytej podczas porównywania efektów wytrenowanych dotychczas modeli, przetestowane zostałyby modele uczone z wykorzystaniem zdjęć o większej rozdzielczości, zbliżonej do oryginalnej, dzięki czemu funkcja strat Dice'a mogłaby być wykorzystana z większą szansą na powodzenie. Następnie wyniki byłyby porównane z uzyskiwanymi dotychczas w celu ustalenia, czy koniecznym jest wykorzystywanie największych możliwych rozmiarów zdjęć w celu uzyskania najlepszych efektów segmentacji obrazu dna oka. Dodatkowo zgodnie z sugestią zawartą w artykule „Empirical Evaluation of Rectified Activations in Convolution Network" [12] przetestowane zostałyby także inne funkcje aktywacji z rodziny ReLU.

5. Podsumowanie

Podczas badań przetestowane zostały różne podejścia do treningu modelu o architekturze U-Net. Początkowy model nie korzystający z żadnej wersji augmentacji danych potrafił po 4000 epok w pewnym zakresie wykryć naczynia obecne na zdjęciu dna oka oraz zawierał stosunkowo mało zaznaczonego niewłaściwego obszaru.
Dodając początkową augmentację danych, która dopuszczała rotację zdjęć w największym testowanym zakresie udało się uzyskać precyzyjniejsze zaznaczenia obszaru, jednak pojawił się na nich większy mylnie zaklasyfikowany obszar w rogach zdjęć oraz szum, który obecny był również na samych maskach po skalowaniu. Pomimo wymienionych wad, wyniki dzięki niemu uzyskane należały do najlepszych.
Zmniejszając zakres maksymalnej rotacji zdjęć i ustawiając liczbę epok na 4000, podczas augmentacji powstał model klasyfikujący w bardziej ostrożny sposób. Zaznaczanego było mniej obszaru, mniej wykrytych naczyń oraz ilość obecnego szumu też była znacznie zmniejszona względem poprzednich modeli. Za wady tego modelu mogła być odpowiedzialna zbyt mała różnorodność zdjęć.
Ostatnia wersja augmentacji, która zwiększyła tylko liczbę danych w zestawie wpłynęła korzystnie na modele. W przeciwieństwie do poprzednich, korzystających z mniej licznej bazy zdjęć, nowe modele wykrywały dużą liczbę naczyń obecnych na obrazach dna oka, niewiele różniąc się pod tym względem od modelu z pierwszej augmentacji. W rogach ponownie pojawiło się mylne oznaczenie, co było jedynym minusem względem wersji z mniejszą ilością danych.
Funkcja strat Dice'a użyta do uczenia modelu nie przyniosła żadnego wyniku, gdy model uczony był tylko z jej użyciem. Pożądany efekt przynosić zaczęła dopiero, gdy wykorzystana została do ponownego trenowania modelu z ostatniej augmentacji. Ponowne uczenie trwało tylko 1000 epok, jednak widoczna była poprawa w wynikach dzięki niemu uzyskanych. Wykrytych zostało więcej naczyń, pojawiło się więcej zakłóceń niż w modelu bazowym, jednak ogólny wynik był lepszy dzięki zauważalnemu początku zaniku niewłaściwych zaznaczeń w rogach obrazu. Model ten przedstawił najlepsze wyniki.
Drugą działającą wersją korzystającą z funkcji strat Dice'a był model trenowany jednocześnie z jej wykorzystaniem oraz binarnej entropii krzyżowej. Jego przewidywania były przeciwne do oczekiwanych, jednak po odwróceniu wartości wyników zaznaczone były naczynia, krawędzie oka, szum jak w przypadku wcześniejszych modeli oraz całe plamy niewłaściwie zaznaczonego obszaru. Model ten był uczony przez tylko 2000 epok, co znaczy, że jego wyniki mogą być gorsze również z powodu niedouczenia modelu.
Podsumowując, podczas wykonywania badań utworzonych zostało wiele modeli różniących się od siebie liczbą epok oraz liczebnością zestawu danych wejściowych. Wyniki uzyskane z ich wykorzystaniem potwierdzają, że zastosowanie architektury U-Net oraz augmentacji danych, umożliwia wytrenowanie modeli zdolnych do przeprowadzania segmentacji obrazów dna oka. Dodatkowo, analiza wyników wykazała, że zwiększenie liczby epok oraz różnorodności

danych wejściowych znacząco wpływa na poprawę dokładności segmentacji, co wskazuje na potencjał dalszego doskonalenia modeli w przyszłych badaniach. Otrzymane rezultaty stanowią solidną podstawę do zastosowania tego podejścia w praktycznych scenariuszach klinicznych.

Reference

1. N. Siddique, S. Paheding, C. P. Elkin and V. Devabhaktuni, "U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications". IEEE Access 2021, vol. 9, pp. 82031 – 82057.
2. Krithika Alias AnbuDevi, M., and K. Suganthi. „Review of semantic segmentation of medical images using modified architectures of UNET". Diagnostics 2022, vol. 12
3. Williams, Christopher, et al. „A unified framework for U-Net design and analysis". Advances in Neural Information Processing Systems 2023, vol. 36 27745-27782.
4. Liu, Fangyu, and Linbing Wang. „UNet-based model for crack detection integrating visual explanations". Construction and Building Materials 2022, vol. 322 126265
5. Shorten, C., Khoshgoftaar, T.M., „A survey on Image Data Augmentation for Deep Learning". Journal of Big Data 2019, vol. 6.
6. Francisco López de la Rosa, et al. „Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a ResNet50 convolutional neural network". Expert Systems with Applications 2022, vol. 206, 117731
7. Wang J., Perez L., „The Effectiveness of Data Augmentation in Image Classification using Deep Learning".
8. Papers With Code STARE. Available online: https://paperswithcode.com/dataset/stare (dostęp 20.10.2024).
9. Papers With Code. Available online: https://paperswithcode.com (dostęp 20.10.2024).
10. Zhihao Chen. „Medical Image Segmentation Based on U-Net". Journal of Physics: Conference Series 2023
11. Wagner, Fabien H., et al. „Using the U-net convolutional network to map forest types and disturbance in the Atlantic rainforest with very high resolution images". Remote Sensing in Ecology and Conservation 5.4 (2019): 360-375.
12. Xu Bing, Wang Naiyan, Chen Tianqi, Li Mu. „Empirical Evaluation of Rectified Activations in Convolution Network". arXiv preprint arXiv:1505.00853 (2015)