

# Porównanie mechanizmów segmentacji biblioteki MONAI oraz sposobów przetwarzania obrazów TK

Analiza w odniesieniu do nerek i nadnerczy

Mateusz Gołębiewski

29 maja 2025



## Streszczenie

W pracy przedstawiono podejście do automatycznej segmentacji organów wewnętrznych na obrazach tomografii komputerowej (CT) z użyciem sieci neuronowych typu: U-Net oraz SegResNet. Zastosowano podejście slice-by-slice (2D) oraz volume-by-volume (3D) i wykonano porównanie efektywności segmentacji. Zaprezentowano również analizę wpływu wielkości modeli, przedstawiając szczegółowe wyniki uzyskane na wybranych metrykach jakościowych.

# 1 Wstęp

Praca została przygotowana jako projekt indywidualny realizowany w ramach zajęć na Politechnice Warszawskiej. Projekt dotyczy automatycznej segmentacji organów wewnętrznych na obrazach tomografii komputerowej (CT). Celem tego dokumentu jest przedstawienie porównania wybranych metod segmentacji dostępnych w bibliotece MONAI, szczególnie skupiając się na porównaniu podejść przetwarzania 2D i 3D oraz analizie skuteczności sieci neuronowych typu U-Net oraz SegResNet. Jako punkt odniesienia uznawane jest narzędzie TotalSegmentator

Segmentacja medyczna stanowi kluczowe zagadnienie współczesnej diagnostyki obrazowej. Automatyczna ekstrakcja struktur anatomicznych, takich jak organy czy tkanki, jest niezbędna w planowaniu leczenia, diagnostyce chorób oraz monitorowaniu stanu pacjentów. Obrazy TK są jednymi z najczęściej stosowanych źródeł danych do tego typu zadań ze względu na wysoką rozdzielcość przestrzenną i możliwość szczegółowej analizy tkanek miękkich oraz struktur kostnych.

Istniejące narzędzia umożliwiające automatyczną segmentację obrazów medycznych, takie jak TotalSegmentator, biblioteka MONAI oraz Fast.ai, znaczco ułatwiają proces przetwarzania i analizy obrazów diagnostycznych. TotalSegmentator to narzędzie stworzone z myślą o segmentacji szerokiego spektrum struktur anatomicznych z obrazów TK, oferujące gotowe do użycia modele oparte na głębokich sieciach neuronowych. MONAI (Medical Open Network for AI) to biblioteka rozwijana przez społeczność open-source, dostarczająca kompleksowy zestaw narzędzi do szybkiego prototypowania, treningu oraz walidacji modeli AI w zastosowaniach medycznych. Fast.ai oferuje przystępne API pozwalające na szybkie budowanie modeli głębokiego uczenia, także w obszarze medycznym.

W projekcie skoncentrowano się na segmentacji dwóch par organów: nerek oraz nadnerczy. Nerki są często przedmiotem badań diagnostycznych ze względu na ich istotną funkcję w układzie wydalniczym oraz często występujące choroby, takie jak kamica nerkowa, guzy czy przewlekła niewydolność nerek. Nadnercza, mimo mniejszych rozmiarów, odgrywają kluczową rolę hormonalną i ich precyzyjna lokalizacja oraz segmentacja może być ważna w diagnostyce nowotworów czy innych patologii hormonalnych.

Zastosowanie głębokich sieci neuronowych, takich jak U-Net czy SegResNet, pozwala na uzyskanie wysokiej precyzyji automatycznej segmentacji. Sieć U-Net jest jedną z najbardziej popularnych architektur używanych w obrazowaniu medycznym, wykorzystując symetryczną strukturę oraz połączenia pomijające (skip connections), co pozwala na efektywne przekazywanie informacji przestrzennej. Z kolei architektura SegResNet opiera się na blokach resztowych (Residual Blocks), które dzięki swojej strukturze pomagają w stabilizacji i przyspieszeniu treningu oraz często zapewniają lepszą zdolność generalizacji.

W dalszych częściach dokumentu przedstawiono szczegółowe porównanie wybranych podejść oraz analizę uzyskanych wyników w kontekście segmentacji obrazów TK nerek oraz nadnerczy z użyciem wymienionych modeli.

## 2 Postawione zadanie

W ramach projektu zostały postawione następujące cele szczegółowe:

- Utworzenie własnego modułu Python, umożliwiającego automatyczną segmentację wybranych organów na obrazach tomografii komputerowej.
- Przeprowadzenie analizy porównawczej różnych podejść do przetwarzania danych, w tym wykorzystania augmentacji danych oraz podejścia 2D i 3D.
- Porównanie dwóch architektur sieci neuronowych dostępnych w bibliotece MONAI: U-Net oraz SegResNet, wraz z analizą wpływów parametrów tych modeli (m.in. liczby filtrów, głębokości sieci) na skuteczność segmentacji.
- Przygotowanie kompleksowego sprawozdania dokumentującego przebieg badań, wyniki eksperymentów oraz wnioski.
- Przeprowadzenie szczegółowej analizy porównawczej metryk jakościowych oraz wizualnych efektów uzyskiwanych przez modele.

## 3 Zakres porównania

W projekcie uwzględniono następujące aspekty porównawcze:

- Segmentacja w podejściu 2D (slice-by-slice, każdy przekrój przetwarzany osobno) w porównaniu z podejściem 3D (przetwarzanie obrazów w trójwymiarowych blokach: 8x256x256, uwzględniających kontekst przestrzenny danych).
- Porównanie architektur sieci neuronowych U-Net oraz SegResNet, zaimplementowanych w bibliotece MONAI.
- Analiza wpływu złożoności modeli (m.in. liczby filtrów oraz głębokości architektury).
- Porównanie własnych modeli z rozwiązaniem wzorcowym - TotalSegmentator.

## 4 Zbiór danych

W projekcie wykorzystano zbiór danych TotalSegmentator CT — obecnie jeden z największych publicznie dostępnych zestawów danych do segmentacji trójwymiarowych obrazów TK. Zestaw zawiera 1228 obrazów TK, w których ręcznie oznaczono 117 struktur anatomicznych, takich jak nerki, nadnercza, wątroba, serce i wiele innych. Dane pochodzą z różnych ośrodków klinicznych, wykonane przy użyciu różnych urządzeń, co poprawia generalizację modeli.

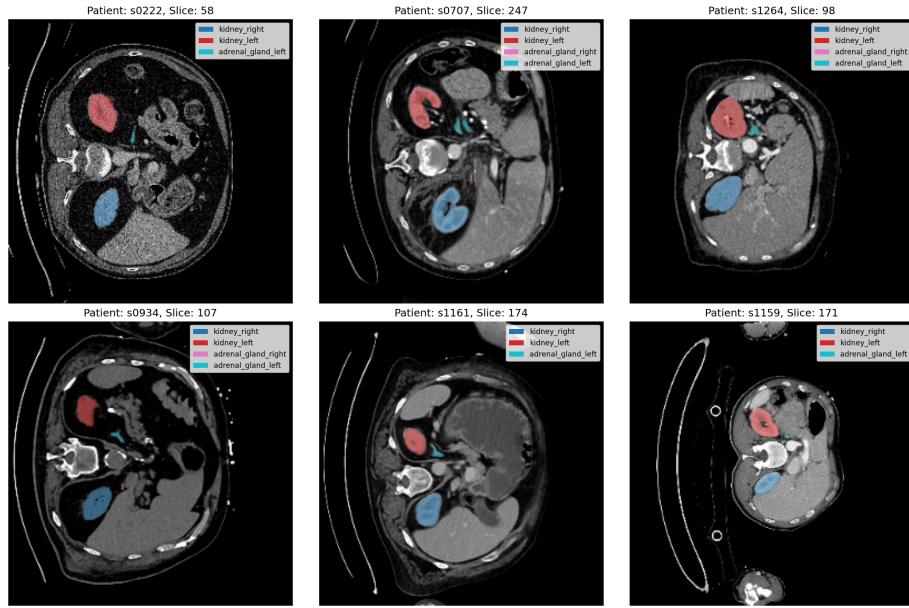
Format plików to **NIfTI (\*.nii.gz)**, powszechnie stosowany w analizie obrazów medycznych. Strukturę zbioru przedstawiono poniżej:

```
dataset/
    patient_001/
        ct.nii.gz
        segmentations/
            kidney_left.nii.gz
            kidney_right.nii.gz
            adrenal_gland_left.nii.gz
            adrenal_gland_right.nii.gz
            ...
    patient_002/
        ...
```

Z uwagi na ograniczenia sprzętowe, takie jak ilość pamięci RAM, VRAM oraz wydajność systemu, związaną z czasem treningu, dla tego projektu ograniczono ilość skanów TK (pacjentów) do 100, co przekłada się na ok. 20 000 przekrojów.

## 4.1 Przykładowe obrazy

Poniżej, na rysunku 1 przedstawiono przykłady pojedynczych przekrojów TK wraz z maskami segmentacji nerek i nadnerczy.



Rysunek 1: Przykładowe przekroje TK wraz z maskami segmentacji nerek oraz nadnerczy.

## 4.2 Podział danych

Dla celów eksperymentalnych dane zostały podzielone zgodnie z następującym schematem:

- **70%** próbek — zbiór treningowy
- **15%** próbek — zbiór walidacyjny
- **15%** próbek — zbiór testowy

Co ważne, dane dzielone są pacjentami a nie przykładami, co odzwierciedla warunki robocze, gdzie model musi rozpoznawać organy nowych, niewidzianych wcześniej pacjentów. Gdyby postąpiono inaczej, model mógłby rozpoznawać budowę już poznanych w trakcie treningu pacjentów, co ułatwiłoby mu zadanie, ponieważ sąsiadujące przekroje mogą być niemal identyczne.

## 5 Ograniczenia sprzętowe

Trening oraz ewaluacja modeli przeprowadzane były na komputerze domowym wyposażonym w:

- **32 GB RAM** – wykorzystywane głównie do przetwarzania i przechowywania danych wejściowych (obrazy TK oraz odpowiadające im maski segmentacji),
- **8 GB VRAM** – dostępna pamięć karty graficznej była głównym czynnikiem ograniczającym możliwą złożoność modelu oraz czas treningu,
- **Karta graficzna NVIDIA RTX 2070 Super** – sprzęt, pozwalający na trenowanie modeli, lecz obarczony dużym obciążeniem i długim czasem przetwarzania dla dużych zbiorów danych.

Zastosowano buforowanie przetworzonych danych wejściowych w formie cache (.pt), co znacznie przyspiesza kolejne uruchomienia modelu bez potrzeby ponownego wczytywania i przetwarzania obrazów źródłowych.

Ostateczny dobór parametrów (wielkość batcha, zakres augmentacji, rozdzielcość wejściowa) był zatem kompromisem pomiędzy jakością trenowanego modelu a dostępnymi zasobami sprzętowymi.

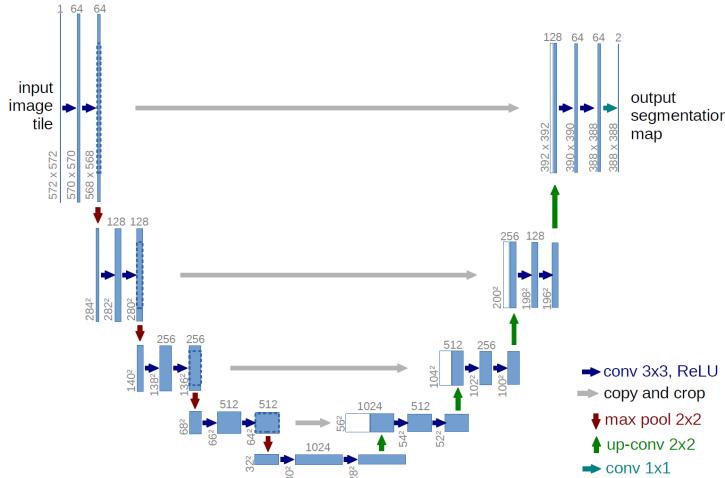
## 6 Wykorzystane modele

### 6.1 U-Net

U-Net to klasyczna architektura sieci konwolucyjnej zaprojektowana do segmentacji obrazów medycznych. Składa się z dwóch głównych części:

- **Koder** – kilkukrotne stosowanie konwolucji  $3 \times 3$  i warstw ReLU, po których następuje pooling w celu redukcji wymiarów i wychwycenia cech kontekstowych,
- **Dekoder** – działania odwrotne: upsampling i konwolucje, przywracające rozdzielcość obrazu.

Kluczowym elementem są **połączenia pomijające** – pomiędzy warstwami o tym samym poziomie w koderze i dekoderze, które umożliwiają przekazywanie szczegółowych informacji przestrzennych, niezbędnych do precyzyjnej rekonstrukcji segmentacji.



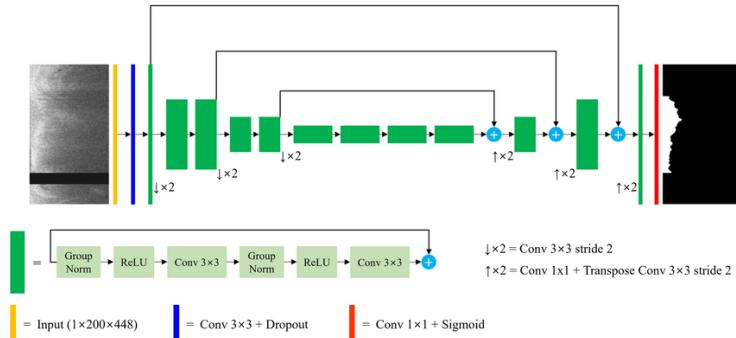
Rysunek 2: Architektura modelu U-Net

**Testowane warianty** W projekcie rozważano różne konfiguracje U-Net:

- U-Net (channels: 64, 128, 256, 512; ResUnits=2);
- U-Net small (channels: 16, 32, 64, 128; ResUnits=2);

## 6.2 SegResNet

SegResNet to architektura typu koder-dekoder bazująca na blokach resztowych (Residual Blocks). Została zaimplementowana w MONAI i przystosowana do segmentacji medycznych danych 2D/3D.



Rysunek 3: Architekтуra modelu SegResNet

### Testowane warianty

- SegResNet (filters=32; blocks\_down=(1,2,2,4); blocks\_up=(1,1,1));
- SegResNet small (filters=16; blocks\_down=(1,1,1,2); blocks\_up=(2,2,2));
- SegResNet 3D (filters=16; blocks\_down=(1,1,1,2); blocks\_up=(1,1,1); spatial\_dims=3);

## 7 Przetwarzanie danych

W projekcie zastosowano dwa odrębne tryby przetwarzania danych obrazowych: **2D (slice-by-slice)** oraz **3D (volume-by-volume)**. Każde z podejść wymagało odmiennych strategii przygotowania danych wejściowych, augmentacji oraz parametrów treningowych.

### 7.1 Podejście 2D (slice-by-slice)

W podejściu 2D każdy trójwymiarowy obraz TK pacjenta został przekształcony na zestaw dwuwymiarowych przekrojów poprzecznych (slices), które traktowano jako niezależne próbki treningowe. To rozwiązanie znacząco zmniejsza zapotrzebowanie na pamięć, umożliwiając trenowanie na kartach graficznych o ograniczonej pojemności VRAM.

- **Fragmentacja danych 2D:** każda próbka to pojedynczy przekrój TK wraz z odpowiadającą mu maską wielokanałową (dla każdego organu osobny kanał).
- **Rozdzielcość wejściowa:** obrazy oraz maski zostały przeskalowane do formatu  $256 \times 256$  pikseli.
- **Wielkość paczki:** ustalona na 16 ze względu na limity VRAM (8 GB).
- **Augmentacja danych:** rotacje, skalowanie, zakłócenia siatki (grid distortion), szum Gaussa. Zrealizowane przy użyciu biblioteki `albumentations`.

## 7.2 Podejście 3D (volume-by-volume)

W trybie 3D przetwarzane są bloki danych zawierające informację przestrzenną. Obraz dzielony jest na wolumetryczne fragmenty ( $256 \times 256 \times 8$ ), co pozwala modelowi na analizę zależności między kolejnymi przekrojami.

- **Fragmentacja danych 3D:** obrazy dzielone na mniejsze woluminy (kawałki) o ustalonej głębokości przestrzennej.
- **Rozdzielcość przestrzenna:** ustalona zgodnie z konfiguracją modelu ( $256 \times 256 \times 8$ ).
- **Wielkość paczki:** zmniejszona względem trybu 2D — 4 wolumeny na raz, ze względu na znacznie wyższe zużycie VRAM.
- **Augmentacje 3D:** wykonywane przestrzennie — m.in. losowe: przesunięcia, skalowania, szum Gaussa, zakłócenia siatki. Realizowane przy użyciu biblioteki MONAI.

## 7.3 Przetwarzanie ogólne

Niezależnie od trybu (2D/3D), wprowadzono dodatkowe mechanizmy filtrowania i przygotowania danych:

- **Odrzucanie próbek ze zbyt małymi maskami:** jeśli liczba aktywnych pikseli dla danego organu nie przekraczała ustalonego progu (np. 50 punktów dla nerek, 30 dla nadnerczy), próbka była pomijana. Pomagało to wyeliminować fałszywe pozytywne klasyfikacje dla przypadków granicznych. W przypadku danych 3D próg było odpowiednio większy i dotyczył całego wolumenu, nie pojedynczych przekrojów. Dodatkowo, rozwiązańe pozwoliło na zawarcie większej ilości wartościowych przykładów w zbiorze uczącym, ograniczając zużycie pamięci.
- **Segmentacja wieloklasowa (multiclass):** dane przygotowywane były jako maski wielokanałowe — każdy kanał odpowiadał jednemu organowi. W fazie ewaluacji model prognozował jednocześnie wszystkie maski, a metryki obliczano niezależnie dla każdego kanału oraz uśredniano.

## 8 Trenowanie modeli

Proces treningu został zrealizowany przy użyciu frameworku PyTorch oraz biblioteki MONAI. Każdy z modeli (zarówno U-Net, jak i SegResNet) był trenowany od zera.

- **Funkcja straty:** zastosowano złożoną funkcję `DiceCELoss`, będącą kombinacją Dice Loss i Cross-Entropy Loss, co umożliwia jednocześnie optymalizację pokrycia segmentacji oraz stabilność klasyfikacyjną.
- **Optymalizator:** Adam ze współczynnikiem uczenia `lr=1e-4` i regularyzacją L2: `weight_decay=1e-4`.
- **Harmonogram:** `ReduceLROnPlateau`, który obniża współczynnik uczenia, gdy metryka walidacyjna przestaje się poprawiać przez kilka epok (5).
- **Early stopping:** zatrzymywanie treningu po 8 epokach bez poprawy Dice Score na zbiorze walidacyjnym.
- **Mixed Precision:** zastosowano `torch.amp` (automatyczne skalowanie gradientów) w celu zwiększenia efektywności obliczeń przy ograniczonej pamięci GPU.
- **Normalizacja:** obrazy TK były znormalizowane do zakresu [0, 1] poprzez okno HU [-100, 300].
- **Logowanie postępów:** po każdej epoce rejestrowano wartości strat treningowych i walidacyjnych oraz metryki (Dice, IoU), które wizualizowano na wykresach.
- **Zapis najlepszego modelu:** checkpoint z najwyższym Dice Score na walidacji był zapisywany w pliku `best_model.pth`.

## 9 Metryki

Do oceny jakości segmentacji wykorzystano zestaw metryk porównujących maski predykcyjne i rzeczywiste. Obliczenia wykonywano osobno dla każdego organu oraz jako uśrednienie ogólne.

- **Dice Score** – miara podobieństwa pomiędzy maskami predykcyjną i rzeczywistą, wrażliwa na nakładanie się pikseli.
- **Intersection over Union (IoU)** – stosunek przecięcia zbiorów do ich sumy (Jaccard Index).
- **Precision** – udział poprawnie przewidzianych pikseli pozytywnych względem wszystkich pozytywnych predykcji.

- **Recall** – udział poprawnie przewidzianych pikseli pozytywnych względem wszystkich faktycznych pikseli pozytywnych.
- **F1-Score** – średnia harmoniczna Precision i Recall, dobrze reprezentuje równowagę pomiędzy nimi.

Obliczenia metryk wykonywano dla każdego przekroju w zbiorze testowym. Następnie wartości były uśredniane po organach i po wszystkich pacjentach. Dodatkowo co 10 batchy zapisywano przykładowe predykcje wizualne (nakładki masek na obraz TK), co ułatwiało diagnostykę błędów modelu.

Wszystkie metryki zapisywano do pliku `evaluation_metrics.txt`, a wyniki prezentowano również graficznie w folderze `eval/`.

## 10 Szczegóły implementacyjne

Projekt został zaimplementowany w języku Python z wykorzystaniem bibliotek PyTorch oraz MONAI. Kod podzielony został na moduły tematyczne zgodnie z zasadą separacji odpowiedzialności, co ułatwia rozwój i testowanie poszczególnych komponentów.

### Główne biblioteki

- **PyTorch** – framework do trenowania modeli głębokiego uczenia, odpowiadający za budowę architektur, optymalizację i obliczenia GPU,
- **MONAI** – biblioteka medyczna oparta na PyTorch, dostarczająca gotowe komponenty (architektury, augmentacje, metryki) dedykowane segmentacji medycznej,
- **Albumentations** – biblioteka do przetwarzania obrazów 2D (augmentacje),
- **Nibabel** – do wczytywania danych medycznych w formacie `.nii.gz`,
- **Matplotlib** – do wizualizacji i wykresów,
- **scikit-learn** – do obliczania metryk.

### Struktura projektu i kluczowe pliki

- `ctseg.py` – główny plik uruchomieniowy; w zależności od parametrów uruchamia trening lub ewaluację.
- `args.py` – parser argumentów linii poleceń, umożliwiający konfigurację treningu (architektura, liczba epok treningu, wielkość paczki, tryb 2D/3D, liczba pacjentów, itp.).
- `train.py` – pełna logika treningu (pętla treningowa, early stopping, scheduler, checkpointy, logowanie historii).

- `eval.py` – ewaluacja modeli na zbiorze testowym: liczenie metryk, zapisywanie wyników i wizualizacji segmentacji.
- `models.py` – definicje architektur U-Net oraz SegResNet 2D (z MONAI), z możliwością parametryzacji (głębokość, liczba kanałów, itp.).
- `data.py` – implementacja klas `Dataset` oraz `DataLoader` dla trybów 2D i 3D, podział zbioru danych, augmentacje, caching, normalizacja.

## **Cache danych i przyspieszenie wczytywania**

Aby przyspieszyć wczytywanie danych i uniknąć ponownej ekstrakcji przekrojów TK, dane zostały buforowane do plików .pt (PyTorch tensors). Pliki cache są zapisywane osobno dla każdej części zbioru (train/val/test) i danego zestawu organów.

## **Obsługa konfiguracji i powtarzalność eksperymentów**

Dzięki modularnej strukturze kodu możliwe było łatwe przeprowadzanie wielu eksperymentów przez zmianę argumentów w wierszu poleceń, np.:

```
python ctseg.py --model segresnet --mode 2d --epoches 30
```

Wszystkie wyniki (checkpointy, wykresy, wizualizacje, metryki) zapisywane były w katalogu `runs/[experiment_name]`, dzięki czemu możliwa była łatwa analiza i porównywanie wielu przebiegów.

## **Powtarzalność wyników**

Kod ustawia domyślne ziarno losowości (np. `seed=42`) zarówno dla NumPy, jak i PyTorch, co umożliwia powtarzalność wyników przy tych samych parametrach.

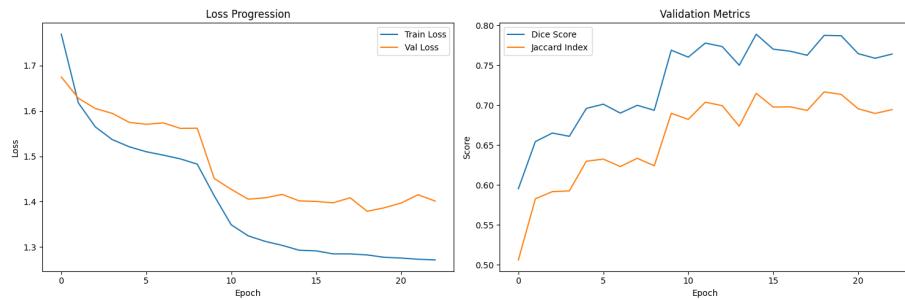
## **Uruchamianie modeli i ewaluacja**

W fazie testowania uruchamiano model w trybie tylko do inferencji, ładując zapisany checkpoint i wykonując predykcje dla zbioru testowego. Dla każdego fragmentu obrazu zapisywano predykcję w formie wizualnej, jak również wyliczano metryki, które zbiorczo zapisywano w `evaluation_metrics.txt`.

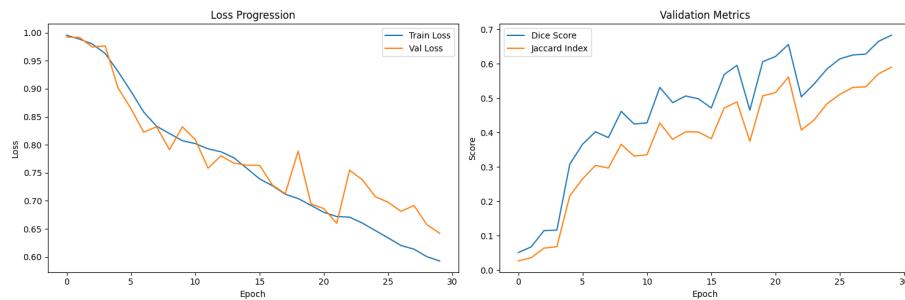
## 11 Wyniki

### 11.1 Porównanie przebiegów treningu

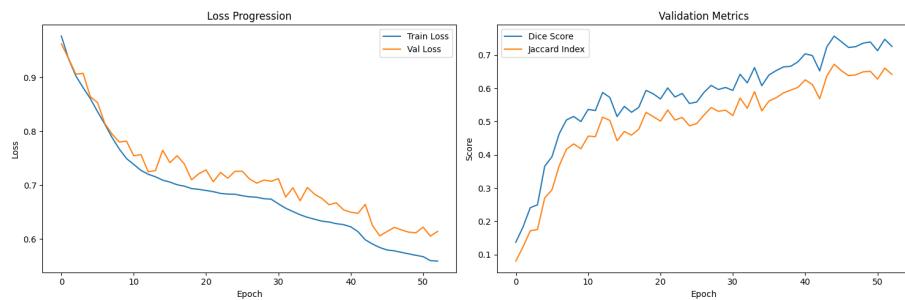
Na rysunkach 4, 5, 6, 7, 8 przedstawiono przebiegi treningowe modeli.



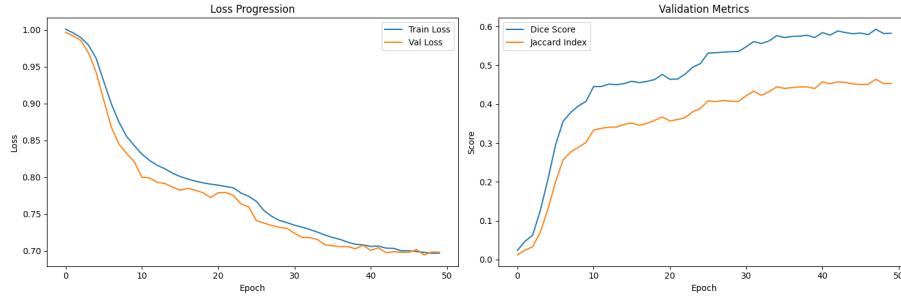
Rysunek 4: Przebieg treningowy - SegResNet



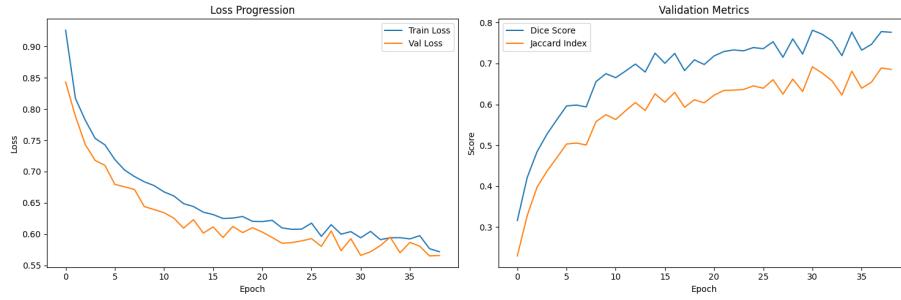
Rysunek 5: Przebieg treningowy - U-Net



Rysunek 6: Przebieg treningowy - SegResNet small



Rysunek 7: Przebieg treningowy - U-Net small



Rysunek 8: Przebieg treningowy - SegResNet 3D

Z powyższych przebiegów wynika, że modele SegResNet znacznie szybciej zyskiwały na jakości wraz z epokami, lecz później hamowała. U-Net natomiast rozwijał się miarowo na przestrzeni większej ilości epok.

## 11.2 Porównanie metryk końcowych

Tabela 1 prezentuje uogólnione, końcowe wyniki metryk.

Model	Dice	IoU	Precision	Recall
U-Net	0,6732	0,5763	0,7193	0,7685
U-Net small	0,5807	0,4452	0,5170	0,8127
SegResNet	0,8456	0,7703	0,8470	0,8871
SegResNet small	0,6814	0,5941	0,6961	0,7330
SegResNet 3D	0,7804	0,7015	0,8136	0,8121

Tabela 1: Wyniki ogólne dla badanych modeli

Tabela 2 prezentuje szczegółowe wyniki końcowe metryk dla poszczególnych masek organów.

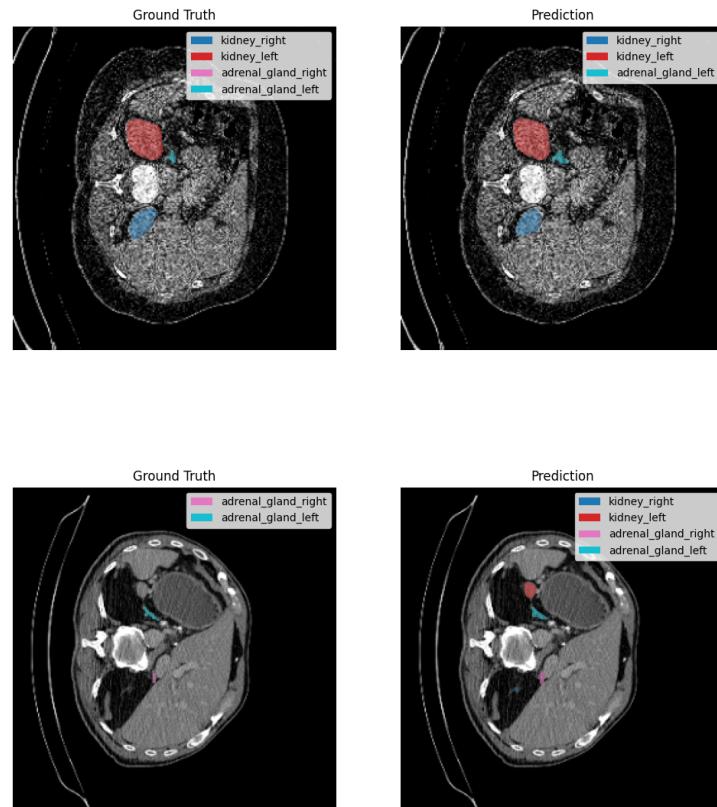
### 11.3 Szczegółowe porównanie metryk końcowych

Organ	Metryka	U-Net	U-Net Small	SegResNet	SegResNet Small	SegResNet 3D
Prawa nerka	Dice	0.7282	0.6086	0.9181	0.7440	0,8833
	IoU	0.6399	0.4698	0.8622	0.6726	0,8192
	Precision	0.7647	0.5012	0.8946	0.7617	0,8715
	Recall	0.8317	0.8892	0.9576	0.7742	0,9329
Lewa nerka	Dice	0.7180	0.6186	0.9249	0.7605	0,8906
	IoU	0.6350	0.4780	0.8728	0.6854	0,8369
	Precision	0.7673	0.5055	0.8933	0.7657	0,9042
	Recall	0.7671	0.8874	0.9704	0.7832	0,9035
Prawe nadnercze	Dice	0.5516	0.5060	0.6319	0.5375	0,5290
	IoU	0.4227	0.3776	0.4920	0.4039	0,4038
	Precision	0.5726	0.5029	0.6242	0.5210	0,6336
	Recall	0.7242	0.6578	0.7221	0.6605	0,5704
Lewe nadnercze	Dice	0.5338	0.4742	0.6513	0.4369	0,5745
	IoU	0.4094	0.3547	0.5252	0.3173	0,4559
	Precision	0.6199	0.6016	0.8061	0.4963	0,6739
	Recall	0.6514	0.5533	0.6451	0.5577	0,5598

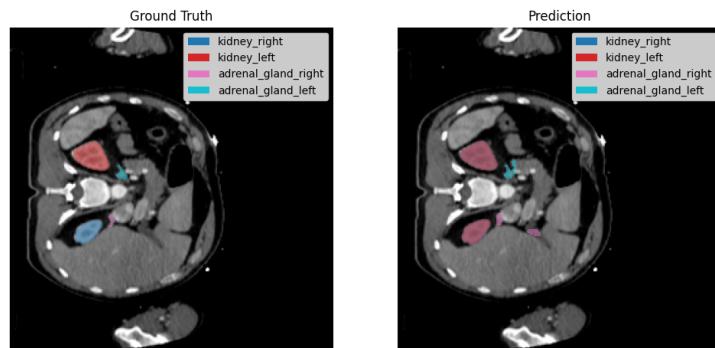
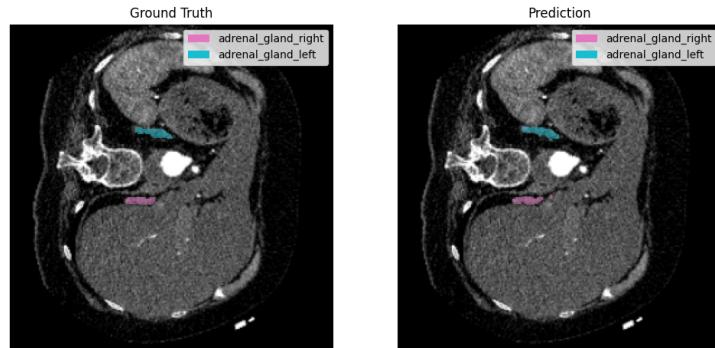
Tabela 2: Porównanie szczegółowych metryk dla narządów i modeli

## 11.4 Wizualizacje predykcji

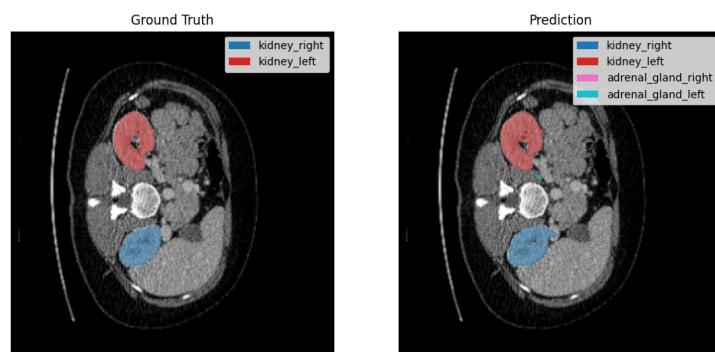
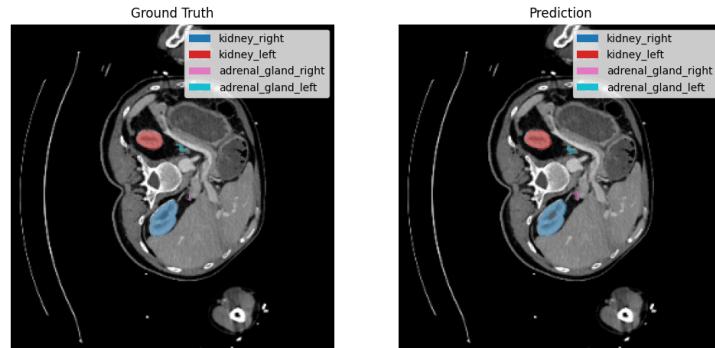
Na rysunkach 9, 10, 11, 12, 13 przedstawiono przykładowe wizualizacje wyników segmentacji.



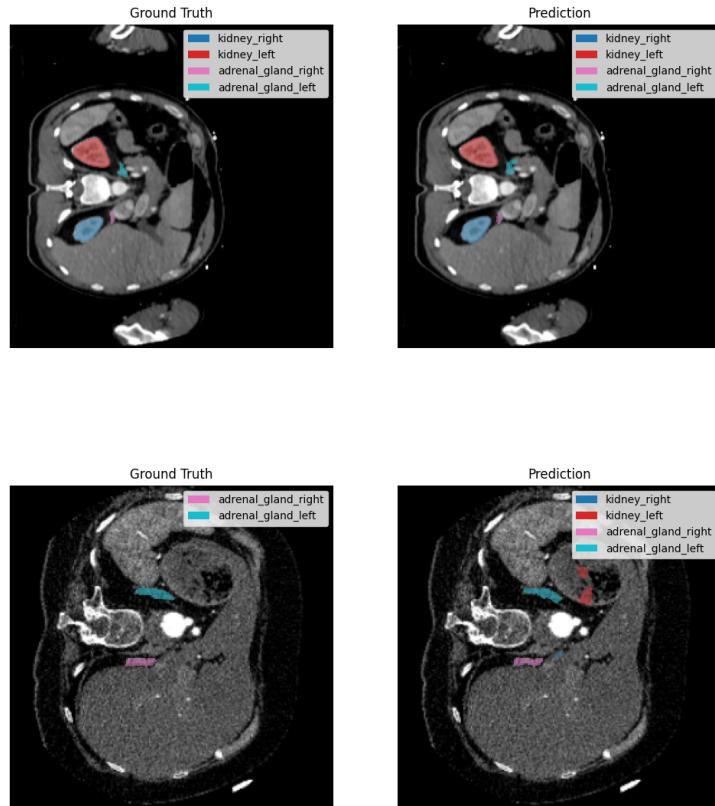
Rysunek 9: Wizualizacja segmentacji - U-Net (dobra i słaba predykcja)



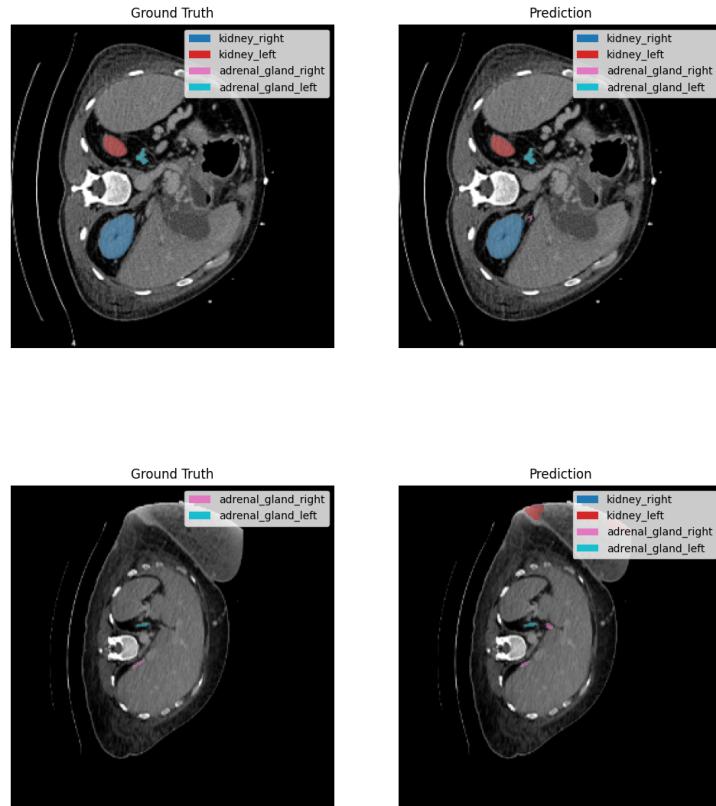
Rysunek 10: Wizualizacja segmentacji - U-Net small (dobra i słaba predykcja)



Rysunek 11: Wizualizacja segmentacji - SegResNet (dobra i słaba predykcja)



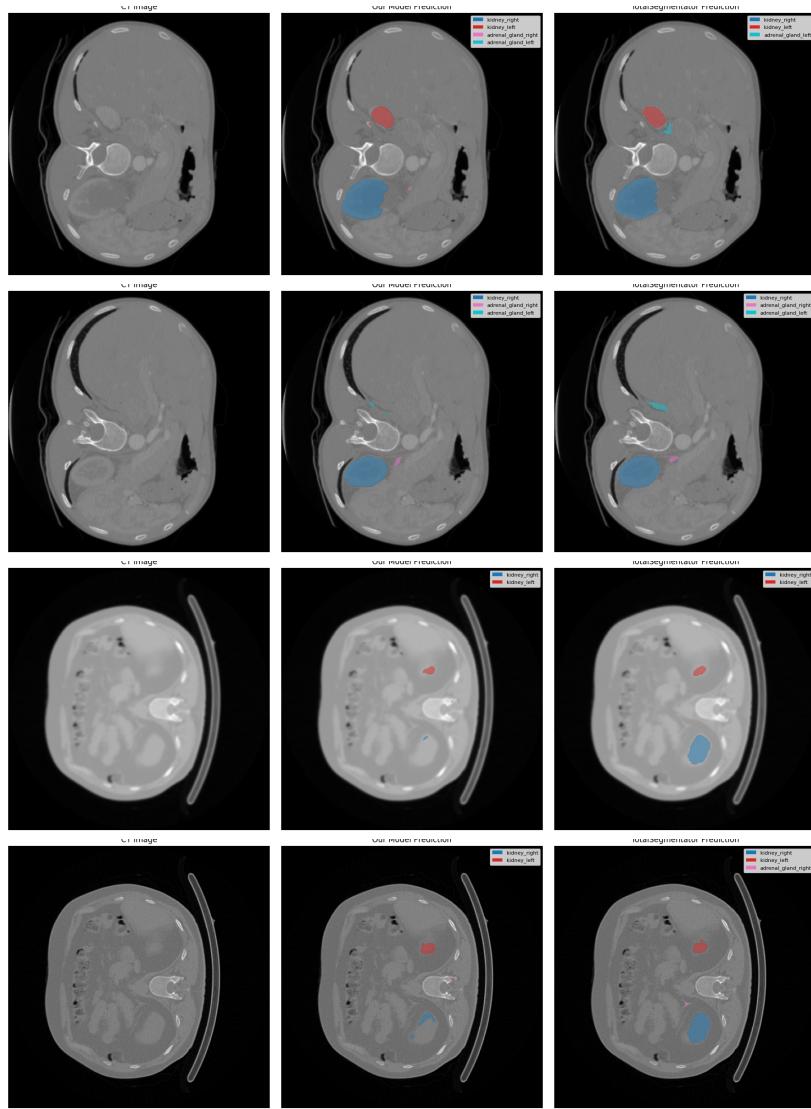
Rysunek 12: Wizualizacja segmentacji - SegResNet small (dobra i słaba predykcja)



Rysunek 13: Wizualizacja segmentacji - SegResNet 3D (dobra i słaba predykcja)

## 11.5 Porównanie z rozwiązaniem wzorcowym (TotalSegmentator)

Na potrzeby tego etapu skorzystano ze zbioru danych, dostarczonego przez dr Roszczyka na potrzeby projektu, tak aby TotalSegmentator nie był testowany na swoim zbiorze treningowym. Porównanie dokonywane jest z użyciem najlepszego modelu własnego - SegResNet (0,84 Dice Score).



Rysunek 14: Od lewej: dane wejściowe, predykcja SegResNet, predykcja Total-Segmentator

## 12 Wnioski końcowe

Przeprowadzone w ramach projektu badania pozwoliły na wszechstronne porównanie wybranych metod segmentacji obrazów tomografii komputerowej (CT) nerek i nadnerczy, z wykorzystaniem biblioteki MONAI. Główne cele projektu, obejmujące utworzenie modułu do automatycznej segmentacji, analizę podejść 2D i 3D oraz porównanie architektur U-Net i SegResNet, zostały zrealizowane.

Kluczowe wnioski płynące z przeprowadzonych eksperymentów są następujące:

- **Wydajność modeli:** Architektura **SegResNet** w wariantie 2D okazała się najskuteczniejsza, osiągając najwyższe uśrednione wyniki metryki Dice Score (0,8456) oraz IoU (0,7703). Model U-Net, choć popularny w segmentacji medycznej, uzyskał niższe wyniki (Dice 0,6732).
- **Wpływ wielkości modelu:** Zaobserwowano, że większy model SegResNet (z większą liczbą filtrów i bloków) znacznie przewyższał swoją mniejszą wersję (SegResNet small). W przypadku U-Net zaobserwowano podobną zależność – większy model (U-Net) osiągnął lepsze wyniki niż jego mniejsza wersja (U-Net small), co sugeruje, że większa pojemność modeli (w pewnych granicach) jest korzystna dla tego zadania.
- **Podejście 2D vs 3D:** W ramach przeprowadzonych testów, podejście 2D (slice-by-slice) dla modelu SegResNet dało lepsze wyniki metryczne niż jego odpowiednik 3D (volume-by-volume). Należy jednak podkreślić, że podejście 3D było silnie ograniczone przez dostępną pamięć VRAM (8 GB), co wymusiło zastosowanie znacznie mniejszej wielkości paczki (batch size = 4) w porównaniu do trybu 2D (batch size = 16). Mimo to, model 3D uzyskał obiecujące wyniki (Dice 0,7804), szczególnie w segmentacji nerek, co wskazuje na jego potencjał przy dostępności mocniejszych zasobów sprzętowych. Podejście 3D ma teoretyczną przewagę dzięki uwzględnieniu kontekstu przestrzennego.
- **Trudność segmentacji organów:** Zgodnie z oczekiwaniemi, segmentacja **nerek** była znacznie łatwiejsza i dokładniejsza (najlepszy Dice Score  $> 0,92$  dla SegResNet) niż segmentacja **nadnerczy** (najlepszy Dice Score  $\sim 0,65$  dla SegResNet). Wynika to przede wszystkim z małych rozmiarów nadnerczy i ich mniejszego kontrastu w obrazach TK.
- **Porównanie z TotalSegmentator:** Wizualna ocena wyników na niezależnym zbiorze danych wskazuje, że najlepszy wytrenowany model (SegResNet 2D) oferuje jakość segmentacji w najlepszym przypadku porównywalną, lecz najczęściej gorszą niż referencyjne narzędzie TotalSegmentator. Biorąc jednak pod uwagę ograniczoną ilość przykładów uczących, można uznać osiągnięty stan za satysfakcyjny. Model uczyony od zera miał problemy z przekrojami, gdzie nie występują szukane organy - generował fałszywie prawdziwe predykcje. Ponadto, źle interpretował przy-

kłady, gdzie obrazy TK różniły się od tych ze zbioru uczącego - o innym wyglądzie (inne metody oraz urządzenia obrazujące).

- **Rola narzędzi:** Biblioteka MONAI udowodniła swoją użyteczność, dostarczając solidnych, gotowych do użycia komponentów (modele, funkcje straty, transformacje), co znaczco przyspieszyło proces implementacji i testowania. Zastosowanie buforowania danych (.pt) oraz zautomatyzowana struktura projektu umożliwiły efektywne zarządzanie eksperymentami i zapewniły powtarzalność wyników.

Podsumowując, projekt wykazał, że możliwe jest osiągnięcie wysokiej jakości automatycznej segmentacji nerek i nadnerczy przy użyciu sieci SegResNet dostępnych w MONAI, nawet przy ograniczonych zasobach sprzętowych, stosując podejście 2D. Uzyskane wyniki są konkurencyjne w stosunku do istniejących rozwiązań. Dalsze prace mogłyby koncentrować się na optymalizacji modeli 3D z wykorzystaniem większych zasobów obliczeniowych, eksploracji innych architektur oraz technik augmentacji, a także na rozszerzeniu badań na większy zbiór danych w celu dalszej poprawy generalizacji i dokładności, szczególnie w kontekście trudniejszych do segmentacji nadnerczy.

## Literatura

- [1] Ronneberger, O., Fischer, P., & Brox, T. “U-Net: Convolutional Networks for Biomedical Image Segmentation.”, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015.
- [2] Wasserthal, J., Meyer, M., Hanns, N., Isensee, F., & Maier-Hein, K. H. “TotalSegmentator: Robust Segmentation of 104 Anatomical Structures in CT Images.”, *Radiology: Artificial Intelligence*, 5(5), 2023.
- [3] Cardoso, M. J. et al. “MONAI: An open-source framework for deep learning in healthcare.”, *arxiv pre-print*, 2022. [Online]. Dostęp: 29 Maj 2025.
- [4] MONAI Project. “MONAI - Medical Open Network for AI”. [Online]. Dostęp: 29 Maj 2025.
- [5] Wasserthal, J. et al. “TotalSegmentator: Robust segmentation of 117 anatomical structures in CT images (Dataset)”. Zenodo, 2023. [Online]. Dostęp: 29 Maj 2025.