AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ

Projekt dyplomowy

Rozpoznawanie zmian nowotworowych w obrazach rezonansu magnetycznego mózgu

Recognition of cancer lesions in brain magnetic resonance images

Autor: Kacper Haras
Kierunek studiów: Informatyka Stosowana
Opiekun: dr hab. inż. Szymon Łukasik

Kraków, 2025

Spis treści

1 Wstęp

Obrazowanie rezonansu magnetycznego (z ang. magnetic resonance imaging - MRI) jest jedną z najważniejszych technik diagnostycznych używanych w medycynie do wykrywania guzów mózgu. Zdolność tej metody do dostarczania szczegółowych informacji o tkankach miękkich sprawia, że jest niezwykle użyteczna w rozpoznawaniu zmian nowotworowych. Jest to również metoda bardzo mało inwazyjna, co czyni ją jeszcze bardziej interesującą zarówno pod kątem możliwości jej rozwijania oraz poszukiwania nowych zastosowań.

Nowoczesne metody wykorzystujące algorytmy uczenia maszynowego, odgrywają coraz większą rolę w analizie obrazów MRI. Są używane do takich zadań jak automatyczna klasyfikacja obrazów lub takich zadań jak segmentacja tkanek. Dzięki zastosowaniu takich technologii możliwe jest nie tylko przyspieszenie procesu diagnostycznego, ale także zwiększenie jego precyzji poprzez eliminację błędów ludzkich.

Celem tej pracy inżynierskiej będzie stworzenie rozwiązania opartego na konwolucyjnych sieciach neuronowych, które będzie uniwersalne dla różnych zbiorów danych. Uniwersalność modelu pozwoli na jego adaptację do analizy obrazów pochodzących z różnych urządzeń MRI lub różnych płaszczyzn przekroju zdjęć. Dzięki temu możliwe będzie skuteczniejsze zastosowanie modelu w praktyce, a także jego elastyczność i potencjał do dalszych zastosowań w badaniach naukowych.

Istnieją już prace, które próbowały rozwiązać podobne zagadnienia. Będą one stanowiły punkt odniesienia do uzyskanych wyników. Porównanie uzyskanych rezultatów z osiągnięciami przedstawionymi w literaturze pozwoli na dokładniejszą ocenę skuteczności zaproponowanego rozwiązania. Na tej podstawie będę w stanie określić, w jakim stopniu moje podejście wnosi nową wartość do obszaru analizy obrazów MRI. Na końcu pracy zaprezentowano podsumowanie badań, w którym uwzględnię wnioski płynące z porównań oraz kierunki dalszego rozwoju tego rozwiązania.

2 Wstęp teoretyczny

2.1 Obrazowanie MRI

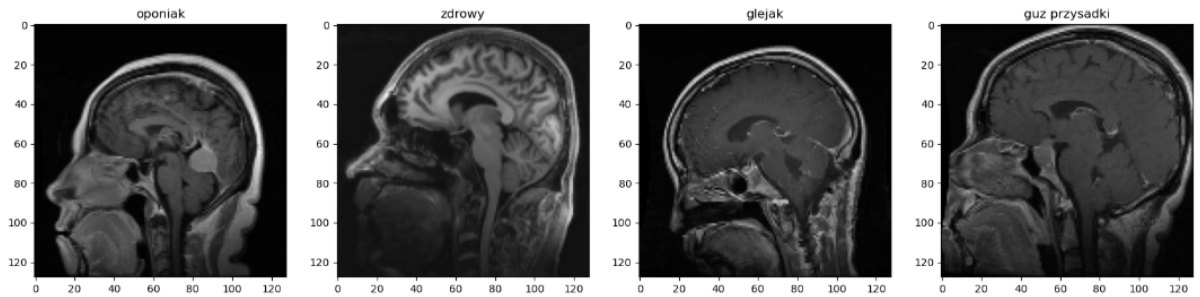
MRI opiera się na zdolności do wzbudzania i wykrywania zmian w zachowaniu protonów zawartych w wodzie, która występuje w tkankach. Metoda ta cechuje się wysoką precyzją, umożliwiając szczegółowe odwzorowanie różnorodnych struktur mózgu. Ma to kluczowe znaczenie w diagnostyce chorób neurologicznych oraz podczas ich późniejszego leczenia. [1]

2.2 Działanie MRI

Silne pole magnetyczne, wytworzone dzięki silnym magnesom, powoduje dopasowanie się osi obrotu protonów do linii pola magnetycznego. Następnie ciało pacjenta zostaje poddane działaniu falom radiowym, które wytrącają protony z równowagi. Po wyłączeniu fal radiowych protony znów ustawiają się zgodnie z polem magnetycznym. Właśnie ten proces powrotu do stanu równowagi w polu magnetycznym uwalnia energię, którą wykrywają czujniki MRI. W zależności od struktury danej tkanki, różni się ilość wyemitowanej energii oraz czas, po którym proton tę energię wydzieli. [1]

2.3 Obrazy w tej pracy

W mojej pracy analizowano dane pochodzące z różnych zbiorów MRI, co umożliwia ocenę działania zaprojektowanego modelu w szerokim spektrum przypadków. Każdy zbiór danych charakteryzuje się swoimi unikalnymi cechami, które mają wpływ na proces uczenia oraz walidacji modelu. Niektóre zbiory zawierają obrazy o wysokiej rozdzielczości, umożliwiające bardziej precyzyjną segmentację tkanek, podczas gdy inne obejmują bardziej różnorodne przypadki kliniczne, co pozwali ocenić zdolność modelu do generalizacji. Uwzględnienie tych różnic jest kluczowe dla osiągnięcia wiarygodnych i użytecznych w praktyce wyników.



Rysunek 1: Przykładowe zdjęcia wraz z etykietami

2.4 Działanie CNN

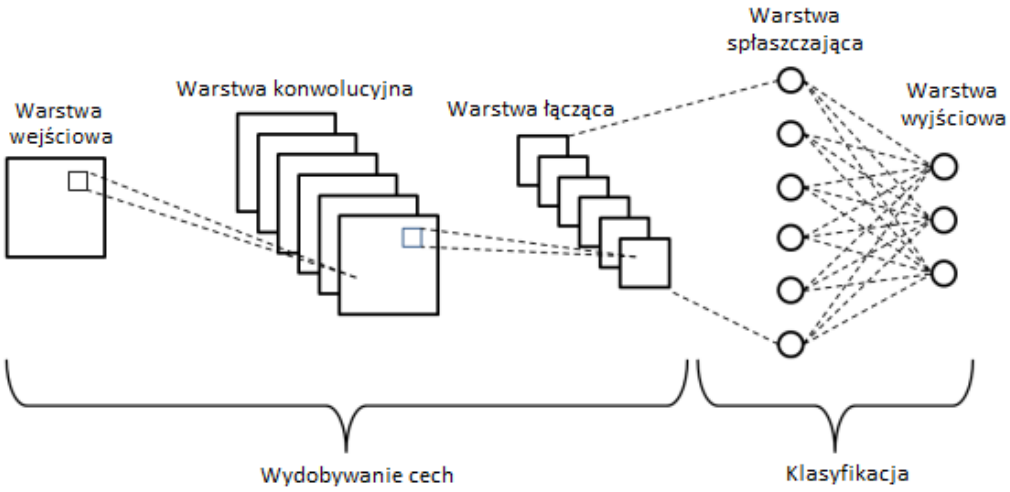
Konwolucyjna sieć neuronowa to jeden z rodzajów sieci neuronowych, zaprojektowanego do przetwarzania danych przestrzennych np. obrazów. Głównym elementem CNN jest wykorzystanie operacji konwolucji, która pozwala na nakładanie filtrów w celu wykrycia lokalnych wzorców w danych takich jak linie, krawędzie czy tekstury w obrazach. Dzięki temu sieć automatycznie uczy się, które cechy obrazu są istotne dla danego zadania.

2.5 Przetwarzania danych wewnątrz CNN

CNN w porównaniu do tradycyjnych sieci w pełni połączonych, redukują liczbę parametrów dzięki zastosowaniu lokalnych połączeń i współdzieleniu wag. Sieć nie łączy każdego piksela obrazu z pojedynczym neuronem, ale dzięki wykorzystaniu lokalnych filtrów analizuje niewielkie fragmenty obrazu. Takie rozwiązanie pozwala na wydobywanie z obrazu wspomnianych wzorców, jednocześnie znacząco zmniejszając złożoność obliczeniową i ilość pamięci potrzebnej do przechowywania parametrów modelu.

2.6 Część klasyfikacyjna

Na końcu sieci mapy cech są "spłaszczane” do jednowymiarowego wektora i podawane na warstwy w pełni połączone. Warstwy te przypominają tradycyjne sieci neuronowe, a każdy neuron jest połączony z każdym elementem wektora wejściowego. Te warstwy działają jak klasyfikator, decydujący na podstawie kombinacji wyodrębnionych wcześniej cech, do jakiej klasy należy dany obraz. Ostatnia warstwa sieci powinna posiadać liczbę neuronów równą liczbie klas.



Rysunek 2: Schemat działania konwolucyjnej sieci neuronowej

2.7 Funkcja straty i optymalizacja

W zależności od zadania dla jakiego planuje się tworzenie model sieci neuronowej, należy użyć odpowiedniej funkcji strat. Jest ona bezpośrednio związana z kierowaniem poprawy wyników walidacyjnych, podczas trenowania modelu. Wybór odpowiedniej funkcji strat pozwoli zmniejszyć błędne predykcje modelu, ale również może zapobiec przypadkom, w których wystąpi przeuczenie modelu.

W przypadku tej pracy najlepszym rozwiązaniem będzie wybór funkcji "Categorical Crossentropy", ponieważ nasz model będzie próbował przewidzieć do jakiej z pośród czterech klas będzie należał testowany obraz. Funkcja ta oparta jest na wykorzystaniu funkcji logarytmicznej, co oznacza, że wraz z wzrostem "pewności"modelu podczas określania do jakiej klasy należy dany obraz rośnie również "kara"w przypadku błędnego przewidywania etykiety. Takie rozwiązanie powoduje, że model uczy się bardziej ostrożnie. [2]

Optymalizator w kontekście sieci neuronowych to algorytm, który dostosowuje wagi modelu w celu minimalizacji funkcji straty podczas treningu.

2.8 Metryki uczenia

Pomimo, że model nie korzysta bezpośrednio z metryk uczenia podczas poprawy wartości funkcji straty to są one dodatkowymi i wartościowymi wskaźnikami. Pomagają w ocenie, jak dobrze model radzi sobie z danym zadaniem. Metryki te nie wpływają na sam proces treningu (np. na obliczanie gradientu), ale są pomocne w monitorowaniu i ocenie postępu modelu. [3] [4]

1. **Dokładność (accuracy)**

   Dokładność określa, jaki procent przewidywań modelu był poprawny w stosunku do wszystkich prób. Definicja matematyczna dokładności określa wzór:

$$\text{Dokładność} = \frac{\text{Liczba poprawnych przewidywań}}{\text{Liczba wszystkich prób}}$$

   Jest to podstawowa metryka oraz łatwa w interpretacji, natomiast gdy zbiory są niezrównoważone, może dawać błędne złudzenie wysokiego wyniku.

2. **Precyzja (precision)**

   Precyzja to procent wszystkich poprawnych przewidywań zakwalifikowanych do danej klasy jest faktycznie tą klasą. W przypadku danych wieloklasowych precyzja mierzona jest dla każdej klasy osobno, a następnie wyliczana jest jej średnia wartość.

Dzięki tej metryce określana jest trafność przewidywań dla każdej z klas. Definicja matematyczna dokładności określa wzór:

$$\text{Precyzja} = \frac{\text{Liczba prawdziwie pozytywnych}}{\text{Liczba wszystkich przewidywanych jako pozytywne}}$$

W kontekście tej pracy precyzja jest bardzo ważną miarą, ponieważ fałszywe przewidywania obecności zmian nowotworowych może negatywnie wpłynąć na pacjentów. Oczywiście wiedza specjalisty jest obecnie najbardziej miarodajna, ale w przypadku poszukiwania najlepszego rozwiązania należy zwrócić szczególną uwagę na tę metrykę.

3. **Czułość (recall)**

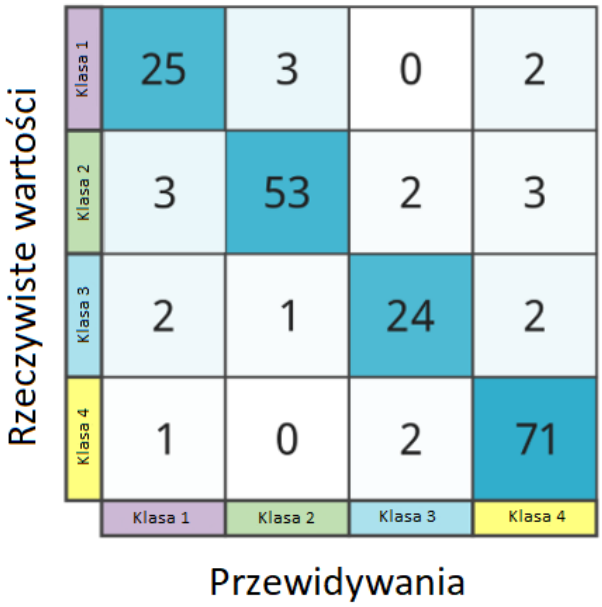
   Czułość to procent wszystkich poprawnie zakwalifikowanych do danej klasy próbek z pośród wszystkich próbek tej klasy. W przypadku klasyfikacji wieloklasowej również wykorzystuje się średnią wartość dla wszystkich klas. Inną nazwą dla czułości spotykaną w literaturze jest odsetek prawdziwie pozytywnych. Definicja matematyczna dokładności określa wzór:

$$\text{Recall} = \frac{\text{Liczba prawdziwie pozytywnych}}{\text{Liczba wszystkich rzeczywistych pozytywnych}}$$

   Czułość to również bardzo ważna metryka w kontekście tej pracy. Przeoczenie zmian nowotworowych podczas ich rzeczywistego występowania może doprowadzić do niewykrycia ich na czas i podjęcia odpowiednich kroków w kierunku leczenia.

2.9 Macierz pomyłek

Wyniki klasyfikatora binarnego można podzielić na cztery kategorie: Prawdziwie pozytywne (true positive – TP), fałszywie pozytywne (false positive – FP), fałszywie negatywne (false negative – FN) oraz prawdziwie negatywne (true negative – TN). W przypadku czterech klas jest to macierz 4*4, gdzie w każdym z wierszy mamy rzeczywiste klasy, a w każdej kolumnie przewidywane. [5]



Rysunek 3: Przykładowa macierz pomyłek

Jeśli model byłby idealny to wszystkie wartości przewidywane odpowiadałyby wartościom prawdziwym. Oznacza to, że w macierzy pomyłek liczby występowały by tylko na przekątnej, a ich suma była równa liczbie próbek ze zbioru testowego. Najczęściej jednak, w przypadku bardziej skomplikowanych zadań, nie da się stworzyć modelu idealnego. Dzięki tablicy pomyłek, możemy w przejrzysty sposób sprawdzić jak model radzi sobie przy testowaniu poszczególnych klas.

3 Zbiory danych

W tej pracy dla porównania wyników modeli wykorzystano trzy zbiory danych, pochodzące z platformy Kaggle. Jest to popularne repozytorium publicznych zbiorów danych czy projektów badawczych z zakresu uczenia maszynowego. Zbiory zostały dobrane tak, by każdy z nich zawierał równą liczbę klas oraz by różniły się zawartością. Poniżej przedstawiono krótki opis każdego ze zbioru danych:

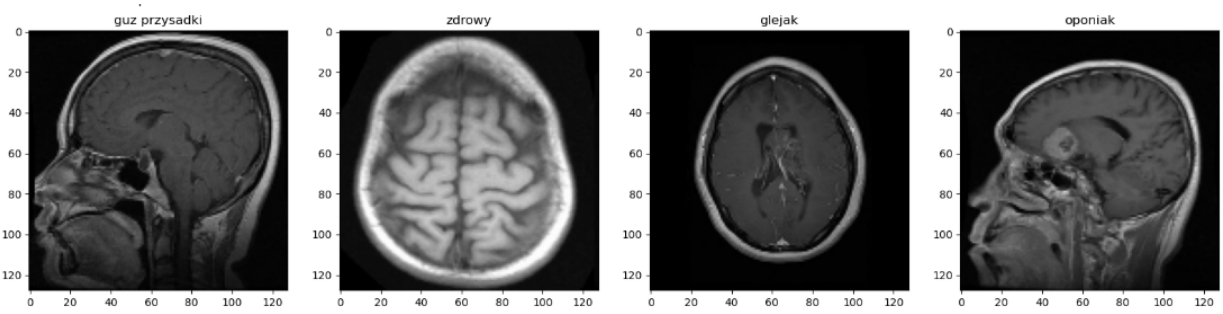
3.1 Brain Tumor MRI Dataset

Zbiór danych został opublikowany przez Masouda Nickparvara na licencji CC0: Public Domain. Ostatnie zmiany w obrazach pochodzą z 24 września 2021. [6]

Dane mają rozmiar 156 MB i są podzielone są na dwie części: uczącą (Training) oraz testującą (Testing), liczące odpowiednio po 5712 oraz 1311 obrazów. Wewnątrz tych folderów znajdują się 4 klasy zawierające trzy rodzaje guzów mózgu: glejak, oponiak, guz przysadki mózgowej oraz przypadek zdrowy.

Training		Testing	
glioma	1321	glioma	300
meningioma	1339	meningioma	306
notumor	1595	notumor	405
pituitary	1457	pituitary	300

Tabela 1: Liczba elementów każdego z podzbiorów zbioru Brain Tumor MRI Dataset



Rysunek 4: Wizualizacja klas dla zbioru Brain Tumor MRI Dataset

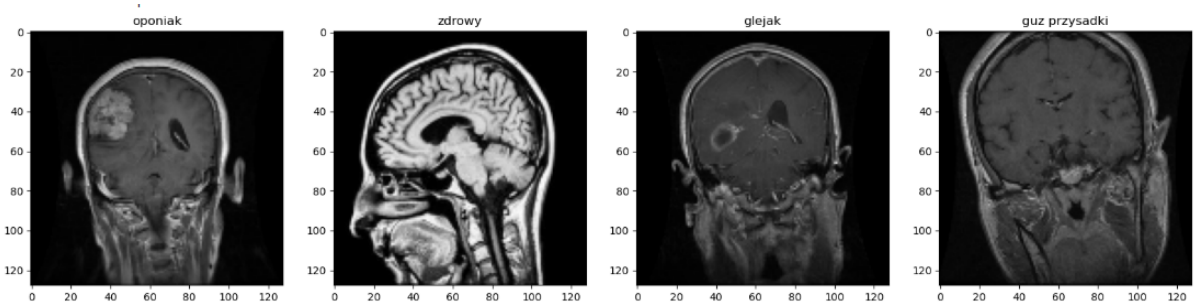
3.2 Brain Tumor Classification (MRI)

Zbiór danych został opublikowany przez Sartaj Bhuvaji na licencji MIT. Ostatnie zmiany w obrazach pochodzą z 24 maja 2020. [7]

Dane mają rozmiar 93 MB i również są podzielone są na dwie części: uczącą (Training) oraz testującą (Testing), liczące odpowiednio po 2870 oraz 394 obrazów. Wewnątrz tych folderów znajdują się 4 klasy zawierające trzy rodzaje guzów mózgu: glejak, oponiak, guz przysadki mózgowej oraz przypadek zdrowy.

Training		Testing	
glioma	826	glioma	100
meningioma	822	meningioma	115
notumor	395	notumor	105
pituitary	827	pituitary	74

Tabela 2: Liczba elementów każdego z podzbiorów zbioru Brain Tumor Classification (MRI)



Rysunek 5: Wizualizacja klas dla zbioru Brain Tumor Classification (MRI)

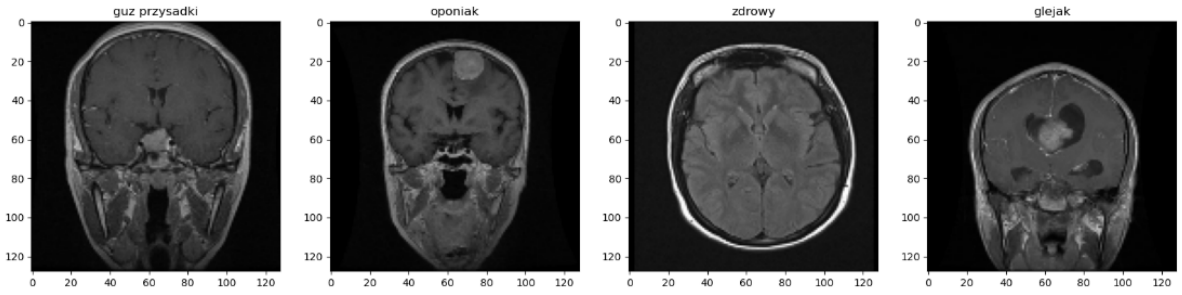
3.3 MRI_Images

Zbiór danych został opublikowany przez Susan Daneshmand na licencji Apache 2.0. Ostatnie zmiany w obrazach pochodzą z 11 grudnia 2023. [8]

Dane mają rozmiar 63 MB i zawierają jeden folder, liczący odpowiednio 3096 obrazów. Wewnątrz folderu znajdują się 4 klasy zawierające trzy rodzaje guzów mózgu: glejak, oponiak, guz przysadki mózgowej oraz przypadek zdrowy.

Data	
glioma	901
meningioma	913
notumor	438
pituitary	844

Tabela 3: Liczba elementów każdego z podzbiorów zbioru MRI_Images



Rysunek 6: Wizualizacja klas dla zbioru MRI_Images

4 Implementacja rozwiązania

W tej części pracy szczegółowo opisano implementacje opracowanego rozwiązania, z podziałem na kolejne etapy projektowania modelu oraz decyzje podjęte podczas pracy nad rozwiązaniem. Opis ten ma na celu wyjaśnić logikę działania algorytmu, strukturę kodu oraz zastosowanych technologii i metod uczenia maszynowego.

Do napisania programu wykorzystano bibliotekę Keras. Jest to wysokopoziomowy interfejs biblioteki TensorFlow, który dzięki prostemu API umożliwia tworzenie, trenowanie oraz testowanie modeli uczenia głębokiego. Dzięki temu, że jest tak intuicyjny, Keras jest jedną z najczęściej wybieranych bibliotek w uczeniu maszynowym.

Implementacja rozwiązania została podzielona na etapy, które odpowiadają poszczególnym krokom schematu postępowania w zadaniach uczenia maszynowego. Każdy z tych etapów odgrywa istotną rolę w odpowiednim wytrenowaniem modelu oraz jego końcowej skuteczności.

4.1 Wstępne przetwarzanie danych

Funkcja "tf.keras.utils.image_dataset_from_directory"pobrała dane z odpowiednich ścieżek i stworzyła obiekty o typie tf.data.Dataset, który składa się z obrazów i odpowiadającym im etykietom. Dzięki opcji "label_mode=’categorical’"etykiety zostały przekonwertowane do postaci wektora one-hot (pozycja oznaczona jako 1 w wektorze odpowiada klasie, do której należy obraz). Wczytane dane podzielono na serie po 32 obrazy, aby skrócić czas treningu na każdym z etapów uczenia. Dzięki opcji ‘label_mode=’categorical’ etykiety zostały przekonwertowane do postaci wektora one-hot (pozycja oznaczona jako 1 w wektorze odpowiada klasie, do której należy obraz).

```
data_train = tf.keras.utils.image_dataset_from_directory(
    '../data_with_description/Brain_Tumor_MRI_Dataset/Training',
    image_size=(128, 128),
    batch_size=32,
    label_mode='categorical'
)

data_test = tf.keras.utils.image_dataset_from_directory(
    '../data_with_description/Brain_Tumor_MRI_Dataset/Testing',
    image_size=(128, 128),
    batch_size=32,
    label_mode='categorical'
)
```

Rysunek 7: Fragment kodu pobierającego zbiory danych

W funkcji ”preprocess_image” wielkości obrazów zostały ujednolicone dzięki funkcji ”resize” z modułu ”tensorflow.image”, aby każdy obraz posiadał jednakową wielkość, niezależnie od początkowych rozmiarów. Następnie obrazy zostały znormalizowane, Oznacza to, że każdemu pikselowi o wartościach z zakresu od 0 do 255, przypisano wartość z zakresu od 0 do 1. Normalizacja pikselów to standardowy zabieg przyśpieszający proces uczenia modelu. Normalizacja w przypadku obrazów w formacie RGB polega na podzieleniu wartości piksela przez 255. Metoda ”map()” pomaga zastosować funkcję preprocess_image dla każdego z obrazów.

```
def preprocess_image(image, target_size=(128, 128)):
    image = resize(image, target_size)
    image = image / 255.0
    return image
```

Rysunek 8: Funkcja służąca do normalizacji danych

Dane wykorzystane do trenowania oraz testowania tego modelu pochodzą ze zbioru "Brain Tumor MRI Dataset". Zostały wczytane z dwóch podzbiorów (treningowego oraz testowego) zostały podzielone na trzy podzbiory: treningowy – o wielkości 143 serii (80% podzbioru treningowego), walidacyjny – o wielkości 36 serii (20% zbioru podzbioru treningowego) oraz testowy – o wielkości 41 serii (100% podzbioru testowego).

4.2 Projektowanie architektury modelu

Do stworzenia wydajnego i uniwersalnego modelu wykorzystano architekturę opierającą się o "transfer learning"z użyciem istniejącego modelu InceptionV3 jako modelu bazowego. Model ten został załadowany z wagami, które wytrenowane na bardzo dużym zbiorze danych wykorzystywanym do klasyfikacji obrazów - "ImageNet", zawierającym ponad milion obrazów zawartych w 1000 klasach. [9] [10]

Nie załadowano warstwy klasyfikacyjnej dzięki parametrowi ‘include_top=False’. Model ten wykorzystano z uwagi na jego umiejętność do bardzo dobrego uzyskiwania cech z obrazów, dzięki jego odpowiednio zaprojektowanej architekturze. Zdefiniowano również kształt wejściowy obrazów – ”input_shape” jako (128, 128, 3). Odpowiada on wymiarom obrazów, które są poddawane treningowi. Pierwszy wymiar odpowiada za wysokość obrazu liczoną w pikselach a drugi za szerokość obrazu. Trzeci wymiar to liczba kanałów, czyli kolorów obrazu. Dzięki takiemu kształtowi obrazów wejściowych model może przetwarzać jednolite dane.

Do bazowego modelu InceptionV3 dodano kolejne warstwy, które miały na celu umożliwienie modelowi jak najlepsze poradzenie sobie z problemem tej pracy – klasyfikacją obrazów z różnych zbiorów danych.

```
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(128, 128, 3))
base_model.summary()
```

Rysunek 9: Fragment kodu załadowanie modelu bazowego

```
model = Sequential()

model.add(base_model)
model.add(GlobalAveragePooling2D())

model.add(BatchNormalization())
model.add(Dense(256, activation='relu', kernel_regularizer=l2(0.001)))
model.add(Dropout(0.5))

model.add(BatchNormalization())
model.add(Dense(64, activation='relu', kernel_regularizer=l2(0.001)))
model.add(Dropout(0.5))

model.add(Dense(4, activation='softmax'))
```

Rysunek 10: Fragment kodu dodający kolejne warstwy modelu

Pierwszą z warstw dodanych po modelu bazowym jest warstwa ”GlobalAveragePooling2D()”. Przekształca ona trójwymiarowe dane w jednowymiarowy wektor 2048 cech, poprzez wyciągnięcie średniej wartości z wszystkich pikseli przekształcanego obrazu. Kolejne dwa bloki warstw odpowiadają za wydobywanie istotnych cech z danych oraz redukcję nadmiernego dopasowania by zapobiegać przeuczeniu modelu. Warstwa ”BatchNormalization normalizuje dane tak, aby dane miały średnią równą 0 oraz wariancję równą 1. Warstwa ”Dense” tworzy kolejno 256 lub 64 neurony, połączone z warstwami poprzednimi w celu zmniejszenia złożoności modelu. Funkcją aktywacji w tych warstwach jest ”RELU”, a opcja ”kernel_regularizer” służy do zmniejszenia wartości wag, również po to by zmniejszyć ryzyko przeuczenia. Ostatnia z warstw to warstwa odpowiedzialna za klasyfikację. Ma ona 4 neurony odpowiadające każdej z klas, a funkcją aktywacji w tym przypadku jest funkcja ”softmax”. [11] [12]

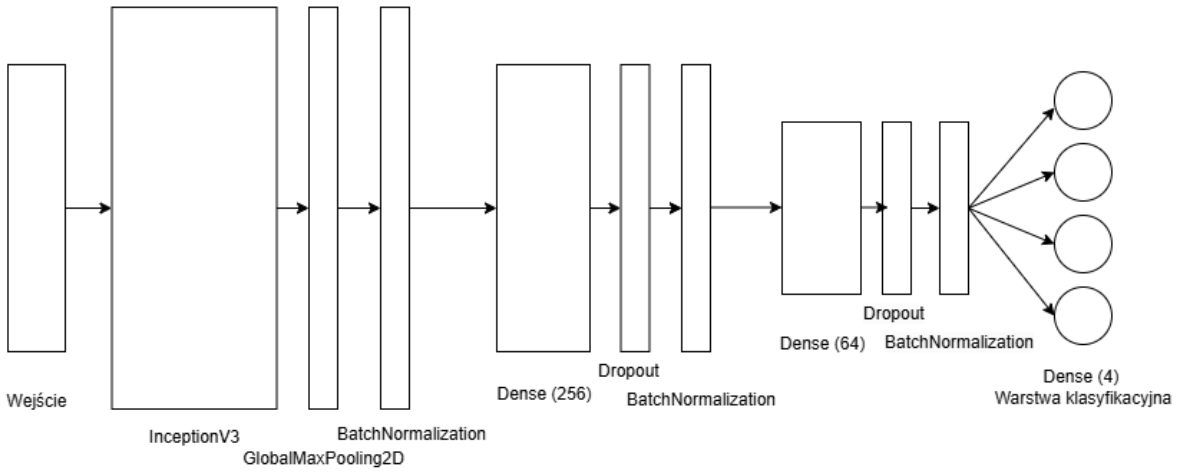
Poniżej przedstawiono podsumowanie struktury modelu wraz z informacjami takimi jak liczba parametrów oraz ich rozmiar.

Model: "sequential"

Layer (type)	Output Shape	Param #
inception_v3 (Functional)	(None, 2, 2, 2048)	21,802,784
global_max_pooling2d (GlobalMaxPooling2D)	(None, 2048)	0
batch_normalization_94 (BatchNormalization)	(None, 2048)	8,192
dense (Dense)	(None, 256)	524,544
dropout (Dropout)	(None, 256)	0
batch_normalization_95 (BatchNormalization)	(None, 256)	1,024
dense_1 (Dense)	(None, 64)	16,448
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 4)	260

Total params: 22,353,252 (85.27 MB)
Trainable params: 22,314,212 (85.12 MB)
Non-trainable params: 39,040 (152.50 KB)

Rysunek 11: Podsumowanie architektury modelu

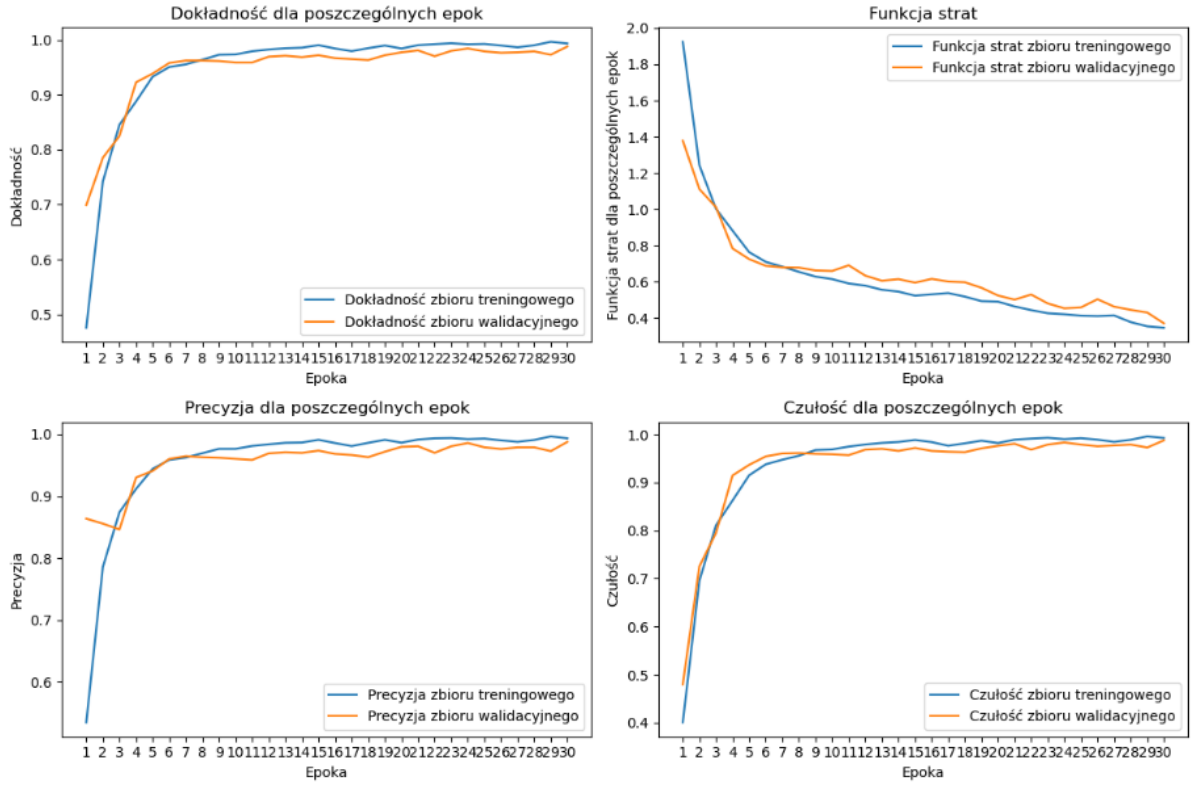


Rysunek 12: Wizualizacja architektury modelu

4.3 Trening modelu

Do wytrenowania modelu użyto zbioru ”Brain_Tumor_MRI_Dataset”. Był to zbiór o największej ilości danych treningowych, co zdecydowano o jego wyborze. Zastosowano opytmalizaotr "Adam"oraz funkcję kosztu "categorical_crossentropy". Jest to najlepsze rozwiązanie, podczas poszukiwania rozwiązania problemu klasyfikacji wieloklasowej.

Model trenowano przez maksymalnie 20 epok, a do kontroli treningu użyto techniki wczesnego zatrzymania uczenia (early stopping) – przy każdej epoce sprawdzano wartość funkcji straty dla modelu walidacyjnego, a jeśli wystąpiło 5 epok, w których wartość ta się nie zmniejszała, trening był zatrzymywany. Parametr "restore_best_weights = True"umożliwa była ostateczna konfiguracja modelu w taki sposób, w jaki był ustawiony podczas gdy miał najlepszy wynik w czasie treningu. Dodano również punkt kontrolny odpowiadający za zapisanie najlepszego modelu do pliku ”.keras”.

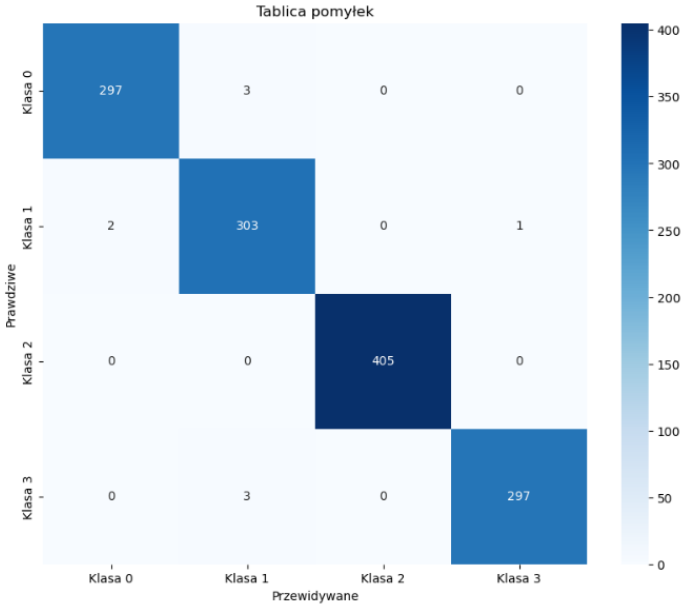


Rysunek 13: Metryki podczas treningu oraz funkcja strat w zależności od danej epoki

4.4 Wyniki testów modelu

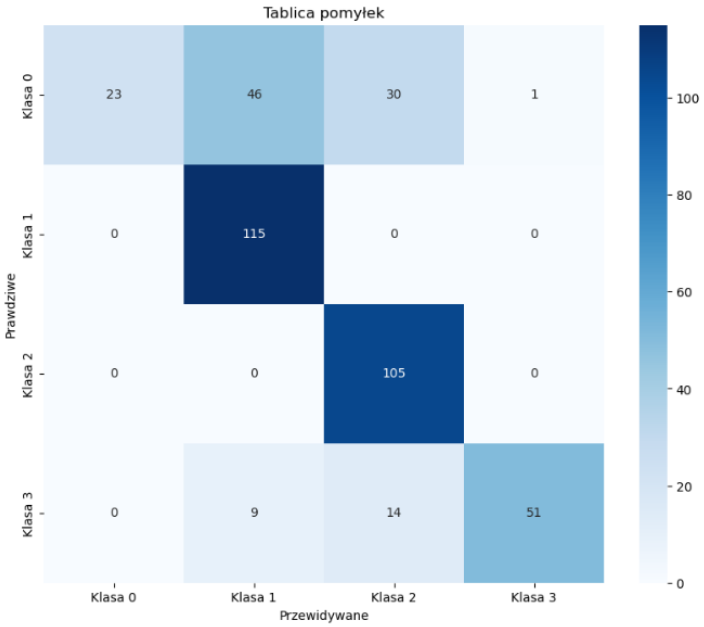
Dokładne wyniki oraz tablice pomyłek dla poszczególnych zbiorów testowych przedstawiono poniżej:

Zbiór testowy (Brain_Tumor_MRI_Dataset):
Dokładność (Accuracy): 0.9931
Precyzja (Precision): 0.9932
Czułość (Recall): 0.9925



Rysunek 14: Tablica pomyłek dla zbioru Brain_Tumor_MRI_Dataset

Zbiór 2 (Brain_Tumor_Classification_(MRI)):
Dokładność (Accuracy): 0.7462
Precyzja (Precision): 0.8233
Czułość (Recall): 0.7298



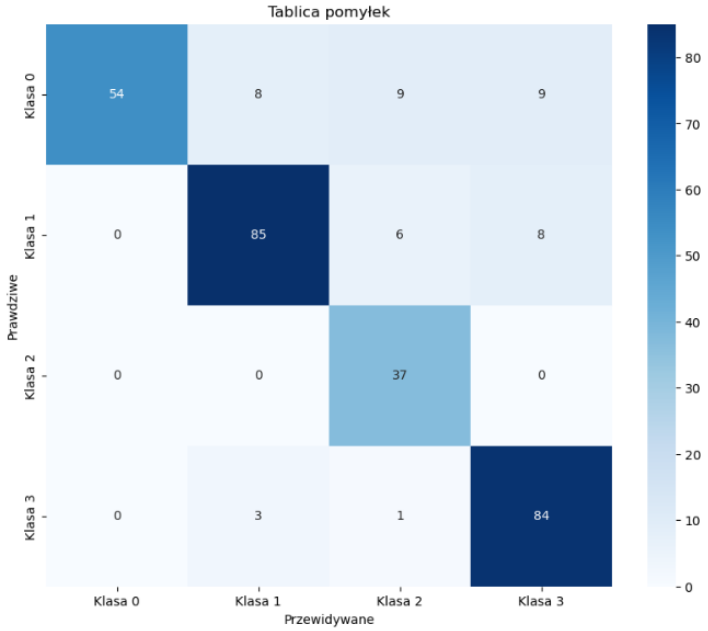
Rysunek 15: Tablica pomyłek dla zbioru Brain_Tumor_Classification_(MRI)

Zbiór 3 (MRI_Images):

Dokładność (Accuracy): 0.8553

Precyzja (Precision): 0.8772

Czułość (Recall): 0.8720



Rysunek 16: Tablica pomyłek dla zbioru MRI_Images

5 Inne implementacje

W tej części pracy przedstawiono trzy różne rozwiązania problemu klasyfikacji obrazów MRI mózgu. Każde z rozwiązań wykorzystuje technikę transfer learningu, opierając się o zastosowanie różnych, wstępnie wytrenowanych architektur sieci neuronowych oraz zbiorów danych. Celem tej części jest przedstawienie kluczowych parametrów każdego z rozwiązań oraz ich wyników dla każdego ze zbiorów. Pozwoli to na porównanie wyników tych trzech implementacji oraz zaprezentowanego przeze mnie w kolejnej części pracy.

W każdej z prac wykonano niezbędne poprawki, dzięki którym usunięto kod nie związany bezpośrednio z modelem oraz jego treningiem. By móc dokonać porównania wszystkich implementacji w każdej z prac zdefiniowano metryki (dokładność, precyzja, czułość) i dodano ich obserwację do treningu modeli. Pozostały kod jest odpowiedzialny za przygotowanie danych, tworzenie modelu oraz proces treningu – został on pozostawiony bez zmian, zgodnie z oryginalnym podejściem autora. W każdej z prac zmodyfikowano również ścieżki do folderów z obrazami. Po treningu każdego z modeli dodano kod odpowiedzialny za testowanie modelu, wyświetlanie tabeli pomyłek oraz raportu klasyfikacji.

5.1 brain-tumor-mri-classification-tensorflow-cnn

Praca została opublikowana pod tytułem ”Brain Tumor MRI Classification: TensorFlow CNN” [13]. Zbiór danych wykorzystany do trenowania modelu to ” Brain_Tumor_ Classification_(MRI)".

Obrazy oraz odpowiadające im etykiety z podzbiorów treningowego i testowego zostały wczytane do dwóch tablic, a zmiar obrazów ustawiono na (150, 150). Następnie obie tablice zostały wymieszane i podzielone na zbiory treningowy (90%) oraz testowy(10%). Etykiety obu zbiorów zostały przekształcone na liczby od 0 do 3, odpowiadające klasom występującym w zbiorze.

Wykorzystano transfer learning używając modelu ” EfficientNetB0” z wagami pochodzącymi ze zbioru ”imagenet”. Pominięto ostatnią warstwę klasyfikacyjną a na jej miejsce dodano kolejne warstwy odpowiadające za: redukcje wielowymiarowych danych do wektora 1280 cech, warstwę wyłączającą połowę neuronów, warstwę gęstą z 4 neuronami oraz funkcją aktywacji ”softmax” – odpowiadającą za klasyfikację.

Model: "sequential"

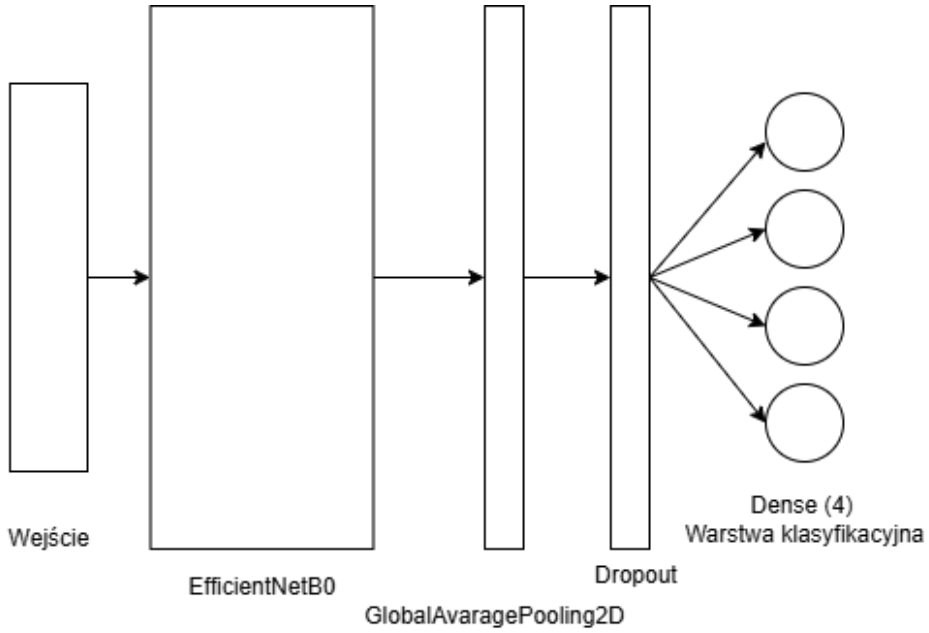
Layer (type)	Output Shape	Param #
EfficientNetB0 (Functional)	(None, 1280, 1280)	4,049,571
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 4)	5,124

Total params: 4,054,695 (15.47 MB)

Trainable params: 4,012,672 (15.31 MB)

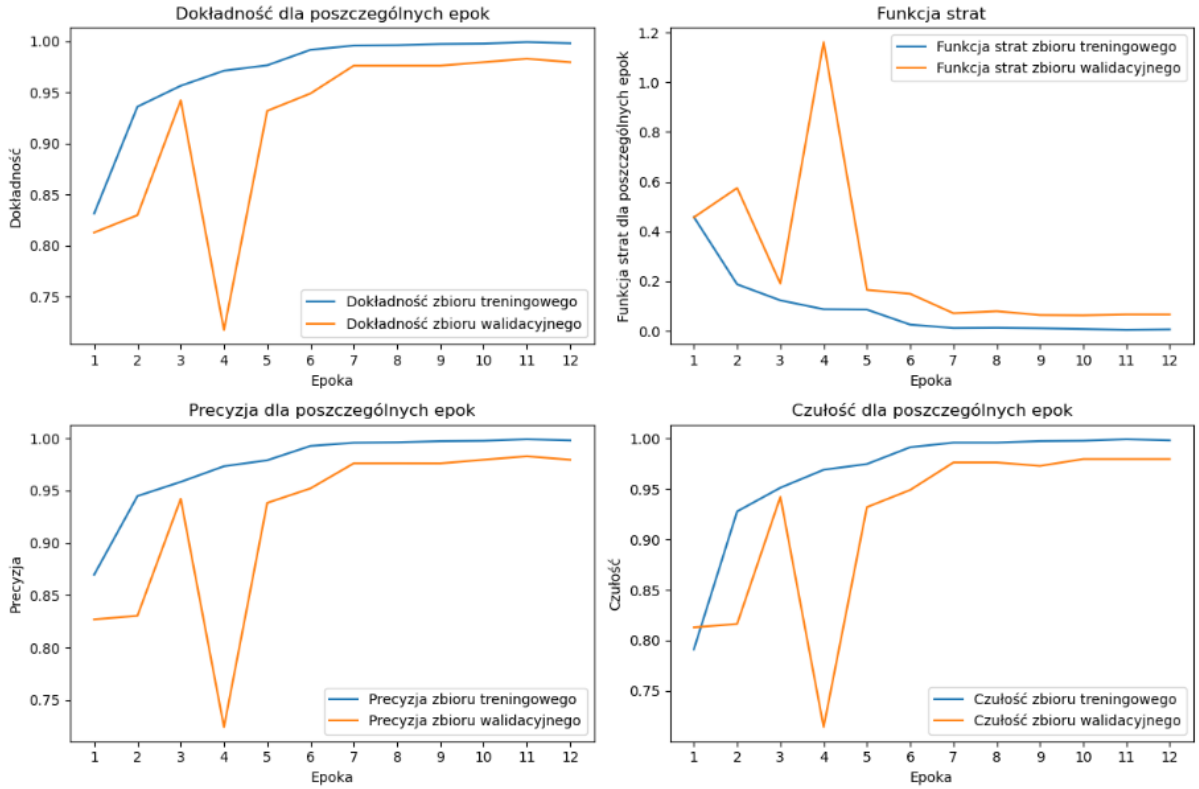
Non-trainable params: 42,023 (164.16 KB)

Rysunek 17: Podsumowanie modelu w pracy Brain Tumor MRI Classification: TensorFlow CNN



Rysunek 18: Wizualizacja modelu w pracy Brain Tumor MRI Classification: TensorFlow CNN

Model skompilowano z użyciem optymalizatora ”Adam”, funkcji straty ” categorical_crossentropy” oraz opisanych wcześniej metryk. Model trenowano przez 12 epok, ze zbiorem walidacyjnym o wielkości 10% całości zbioru treningowego, a dane przekazywano seriami po 32 obrazy. Stworzono punkt kontrolny do monitorowania i zapisywania najlepszej wersji modelu podczas treningu na podstawie najlepszego wyniku dokładności na zbiorze walidacyjnym. Ustawiono dynamiczne zmniejszanie tempa uczenia, gdy model przestaje poprawiać wynik dokładności na zbiorze walidacyjnym przez 2 epoki z rzędu.



Rysunek 19: Metryki podczas treningu modelu oraz funkcja strat w zależności od danej epoki w pracy Brain Tumor MRI Classification: TensorFlow CNN

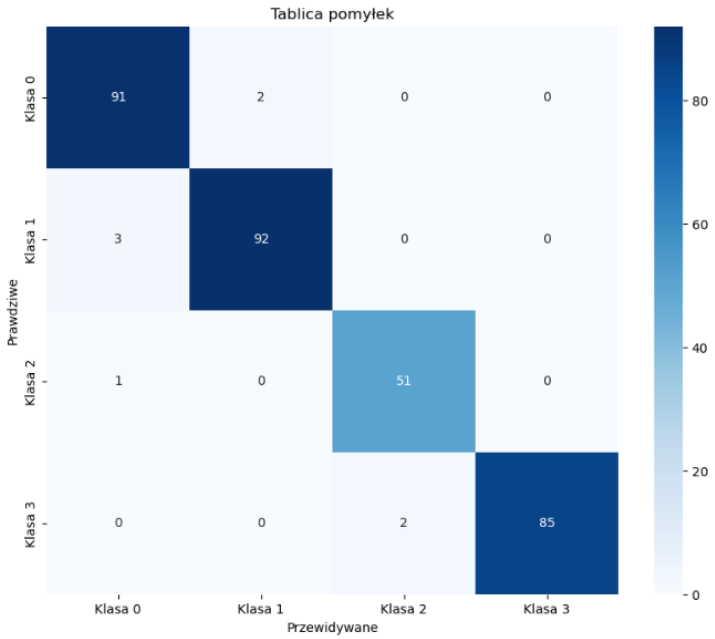
Dokładne wyniki oraz tablice pomyłek dla poszczególnych zbiorów testowych przedstawiono poniżej:

Zbiór testowy (Brain_Tumor_Classification_(MRI)):

Dokładność (Accuracy): 0.9755

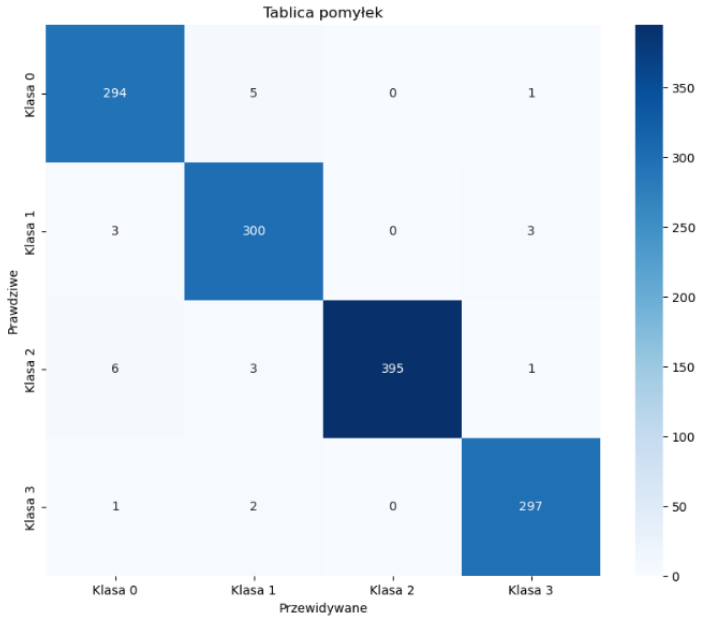
Precyzja (Precision): 0.9758

Czułość (Recall): 0.9762



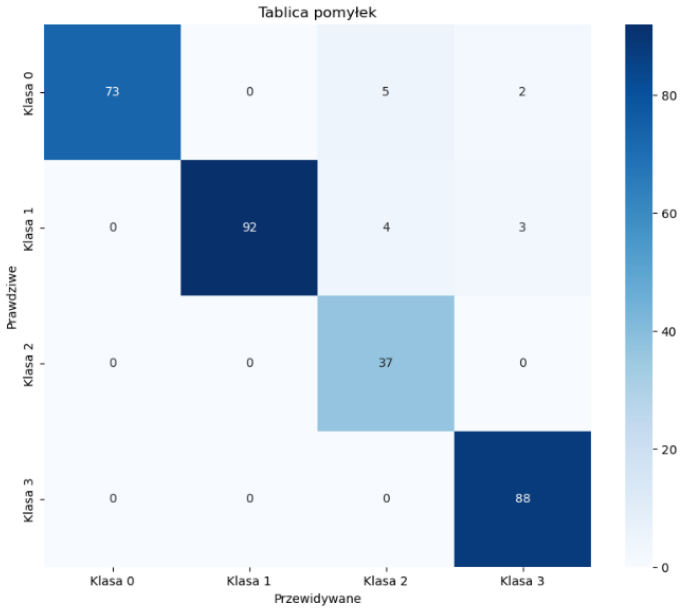
Rysunek 20: Tablica pomyłek dla zbioru Brain_Tumor_Classification_(MRI)

Zbiór 2 (Brain_Tumor_MRI_Dataset):
Dokładność (Accuracy): 0.9809
Precyzja (Precision): 0.9812
Czułość (Recall): 0.9814



Rysunek 21: Tablica pomyłek dla zbioru Brain_Tumor_MRI_Dataset

Zbiór 3 (MRI_Images):
Dokładność (Accuracy): 0.9539
Precyzja (Precision): 0.9606
Czułość (Recall): 0.9604



Rysunek 22: Tablica pomyłek dla zbioru MRI_Images

5.2 braintumormri-using-mobilenetv2

Praca została opublikowana pod tytułem ”BrainTumorMRI using MobileNetV2” [14]. Zbiór danych wykorzystany do trenowania modelu to ”Brain_Tumor_MRI_Dataset”.

Dla podzbioru treningowego i testowego zostały wczytane obrazy oraz etykiety za pomocą biblioteki Pandas. Następnie połączono obrazy z odpowiadającymi im etykietami tworząc zbiór testowy i treningowy. Połowę danych ze zbioru testowego wykorzystano do stworzenia zbioru walidacyjnego. Wielkość serii ustawiono na 16, a kształt danych ustawiono na (224,224,3) – odpowiednio dla wysokości obrazu, szerokości obrazu i ilości kanałów.

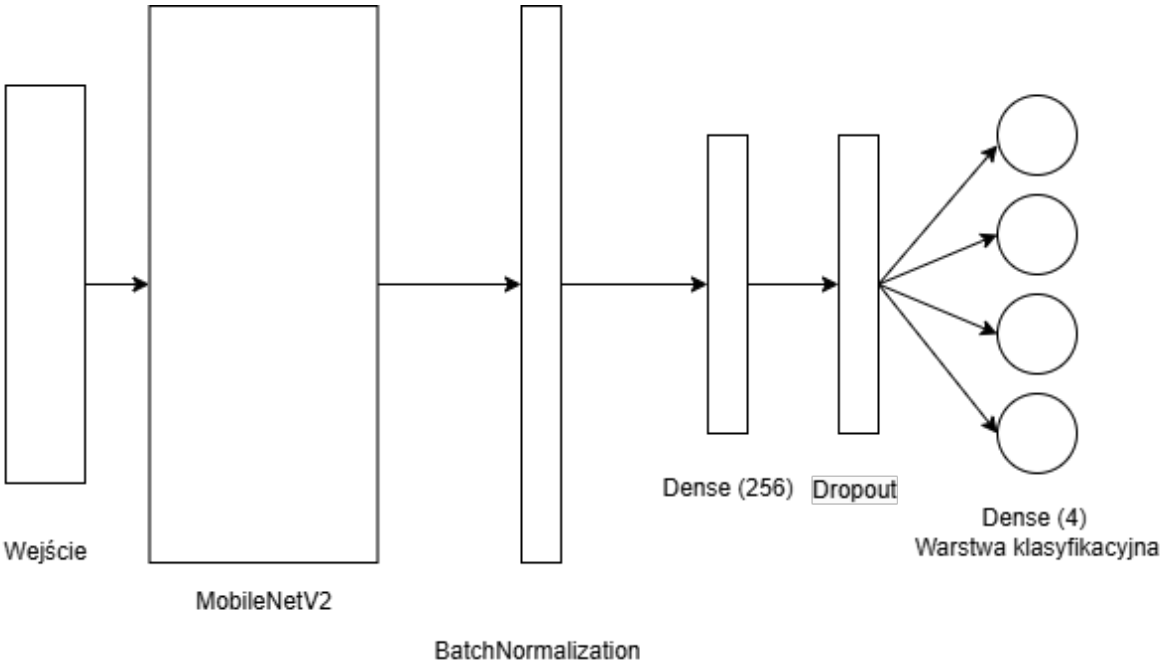
Model bazowy, który został użyty w tej pracy to MobileNetV2. Został on załadowany z wagami wytrenowanymi na zbiorze obrazów ”imagenet”. Ostatnią warstwę konwolucyjną modelu bazowego zredukowano do jednego wektora zawierającego 1280 cechy. Pominięto warstwę klasyfikacyjną bazowego modelu, a dodano cztery warstwy odpowiadające kolejno za: normalizację danych, warstwę gęstą z 256 neuronami oraz funkcją aktywacji ”RELU”, warstwę wyłączającą 45% neuronów oraz ostatnią warstwę gęstą z 4 neuronami i funkcją aktywacji ”softmax” – odpowiadającą za klasyfikację.

Model: "sequential"

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, 1280)	2,257,984
batch_normalization (BatchNormalization)	(None, 1280)	5,120
dense (Dense)	(None, 256)	327,936
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 4)	1,028

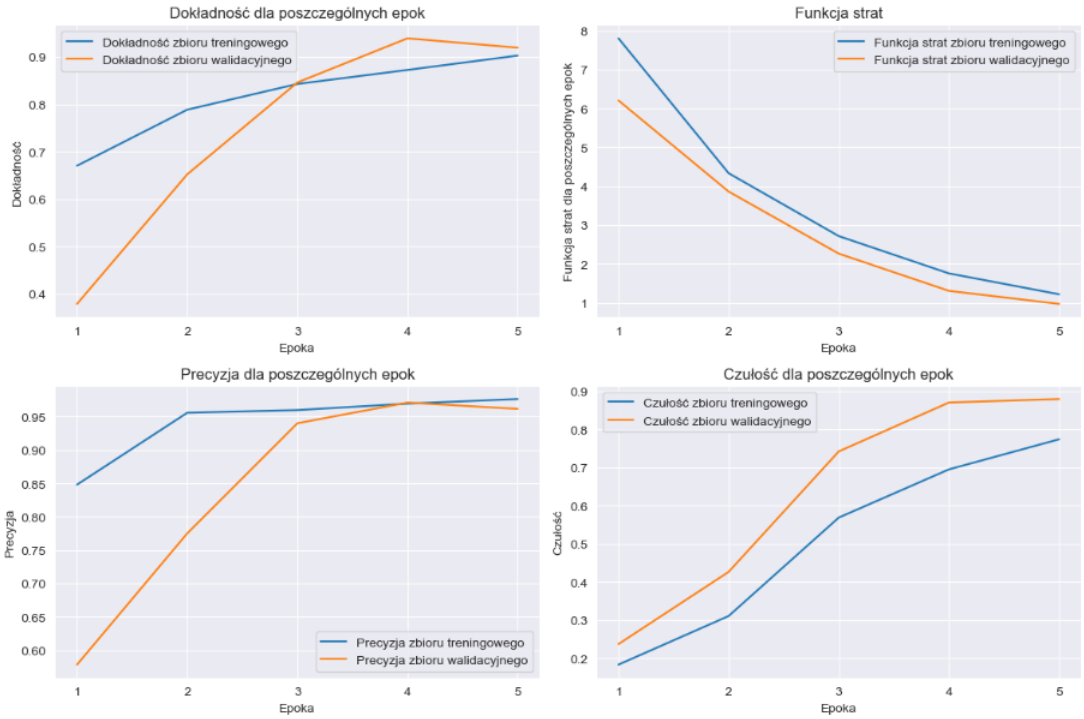
Total params: 2,592,068 (9.89 MB)
Trainable params: 2,555,396 (9.75 MB)
Non-trainable params: 36,672 (143.25 KB)

Rysunek 23: Podsumowanie modelu w pracy Brain-TumorMRI using MobileNetV2



Rysunek 24: Wizualizacja modelu w pracy Brain-TumorMRI using MobileNetV2

Model został skompilowany z użyciem optymalizatora ”Adamx”, funkcji straty ”categorical_crossentropy”, a trenowano przez 5 epok obserwując wartości wspomnianych wcześniej metryk.



Rysunek 25: Metryki podczas treningu modelu oraz funkcja strat w zależności od danej epoki w pracy Brain-TumorMRI using MobileNetV2

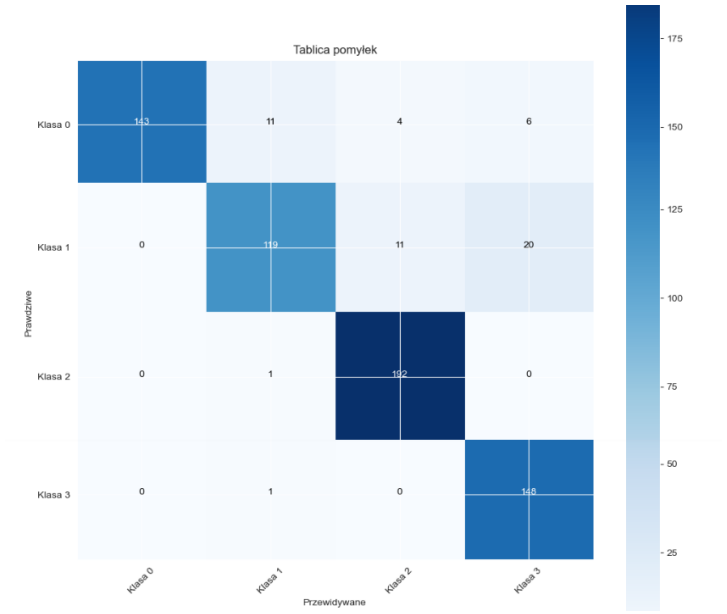
Dokładne wyniki oraz tablice pomyłek dla poszczególnych zbiorów testowych przedstawiono poniżej:

Zbiór testowy (Brain_Tumor_MRI_Dataset):

Dokładność (Accuracy): 0.9177

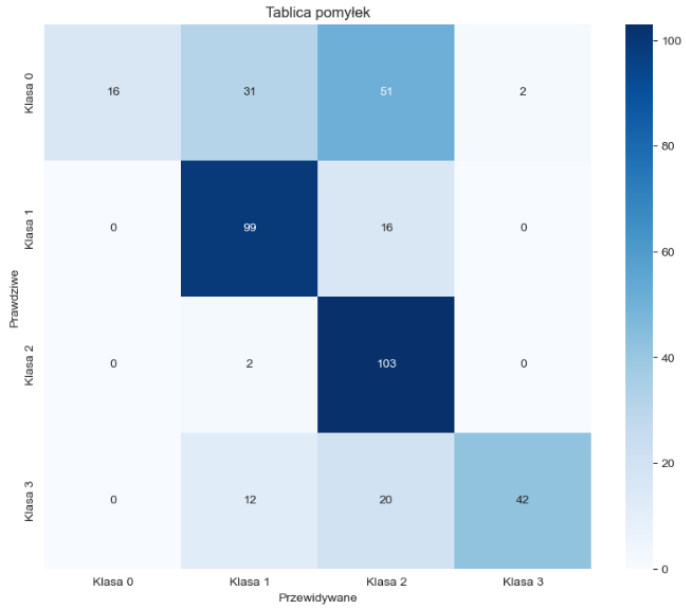
Precyzja (Precision): 0.9222

Czułość (Recall): 0.9133



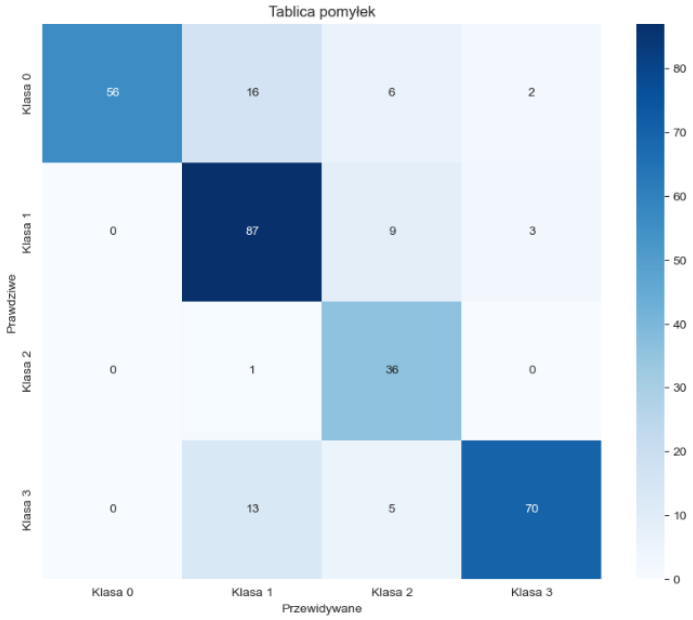
Rysunek 26: Tablica pomyłek dla zbioru Brain_Tumor_MRI_Dataset

Zbiór 2 (Brain_Tumor_Classification_(MRI)):
Dokładność (Accuracy): 0.6599
Precyzja (Precision): 0.7782
Czułość (Recall): 0.6423



Rysunek 27: Tablica pomyłek dla zbioru Brain_Tumor_Classification_(MRI)

Zbiór 3 (MRI_Images):
Dokładność (Accuracy): 0.8191
Precyzja (Precision): 0.8537
Czułość (Recall): 0.8368



Rysunek 28: Tablica pomyłek dla zbioru MRI_Images

5.3 mri-brain-tumor-classification

Praca o nazwie podanej powyżej została opublikowana pod tytułem "Enhanced MRI Brain Tumor Classification"[15]. Zbiór danych wykorzystany do trenowania modelu w tej pracy to ”MRI_Images”.

Dane wczytano seriami po 16 obrazów oraz z wymiarami (256, 256). Podzielono je na zbiory treningowy (80% wszystkich danych), walidacyjny (10%) i testowy (10%). Etykiety przekształcono na liczby od 0 do 3, odpowiadające kolejnym klasom.

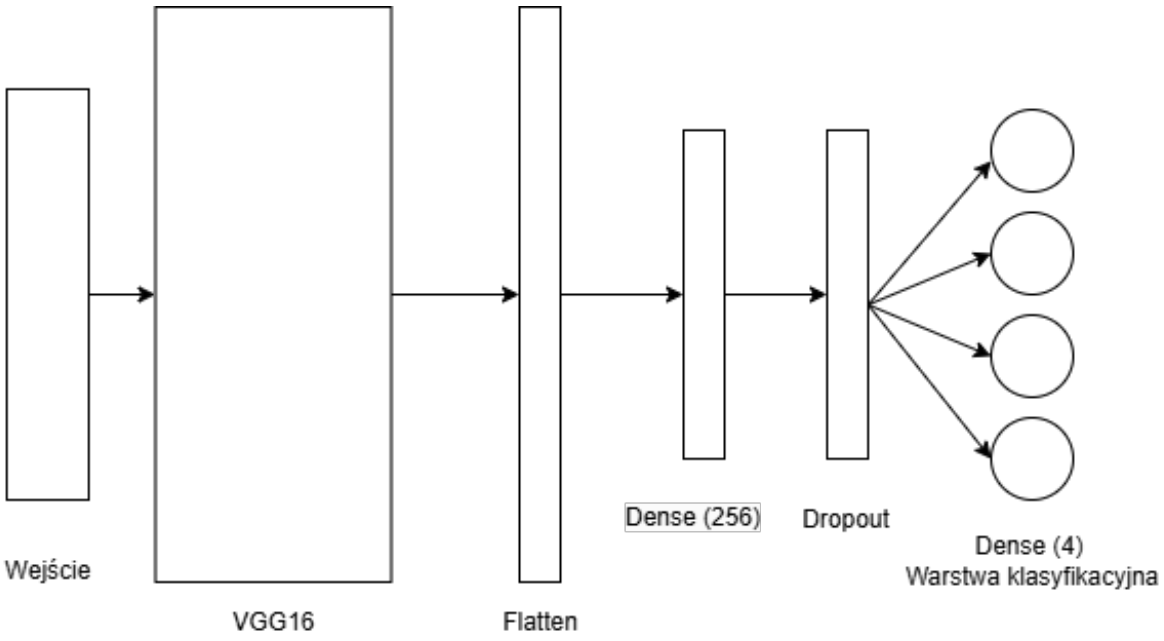
Jako model bazowy wybrano VGG16, którego załadowano z wagami wytrenowanymi na zbiorze obrazów ”imagenet”. Usunięto ostatnią warstwę odpowiadającą za klasyfikację, a dodano warstwy: spłaszczającą dane do postaci jednowymiarowego wektora 32768 cech, gęstą z 256 neuronami i funkcją aktywacji ”RELU”, warstwę usuwającą 50% neuronów oraz warstwę gęstą z 4 neuronami i funkcją aktywacji ”softmax” – odpowiadającą za klasyfikację.

Model: "sequential"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 8, 8, 512)	14,714,688
flatten (Flatten)	(None, 32768)	0
dense (Dense)	(None, 256)	8,388,864
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 4)	1,028

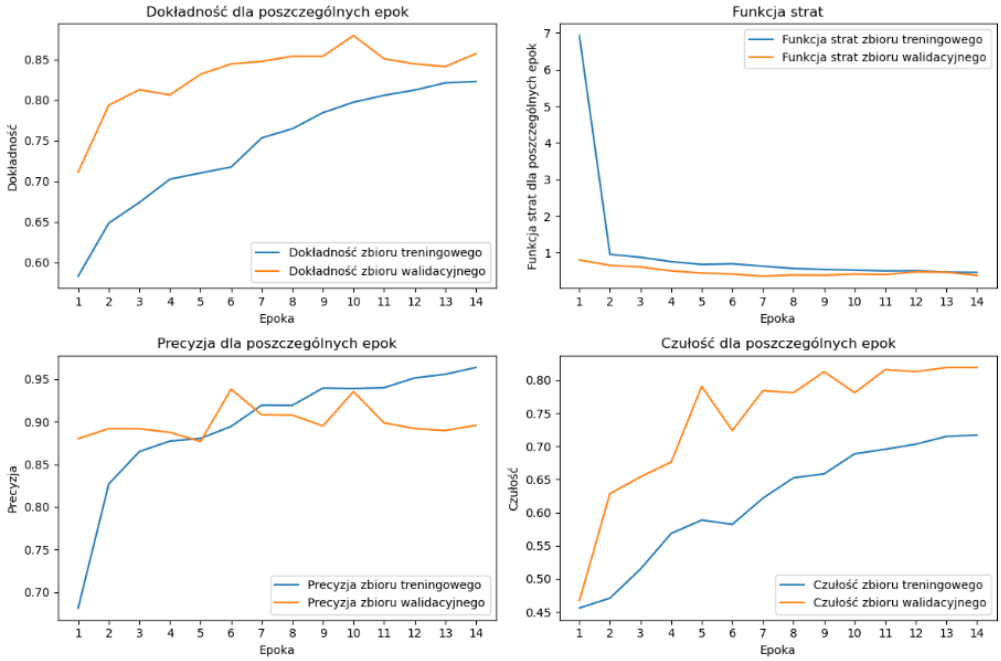
Total params: 23,104,580 (88.14 MB)
Trainable params: 8,389,892 (32.00 MB)
Non-trainable params: 14,714,688 (56.13 MB)

Rysunek 29: Podsumowanie modelu w pracy Enhanced MRI Brain Tumor Classification



Rysunek 30: Wizualizacja modelu w pracy Enhanced MRI Brain Tumor Classification

Model skompilowano używając optymalizatora "Adam", funkcję straty "categorical_crossentropy". Trening ustawiono na maksymalnie 100 epok, ale zastosowano opcje wczesnego zatrzymania treningu, gdy wartość funkcji strat nie zmniejszała się przez 7 epok z rzędu. Dodatkowo zastosowano zmniejszanie tempa uczenia, jeśli model nie zmniejsza wartości funkcji strat zbioru walidacyjnego przez 2 epoki pod rząd.



Rysunek 31: Metryki podczas treningu modelu oraz funkcja strat w zależności od danej epoki w pracy Enhanced MRI Brain Tumor Classification

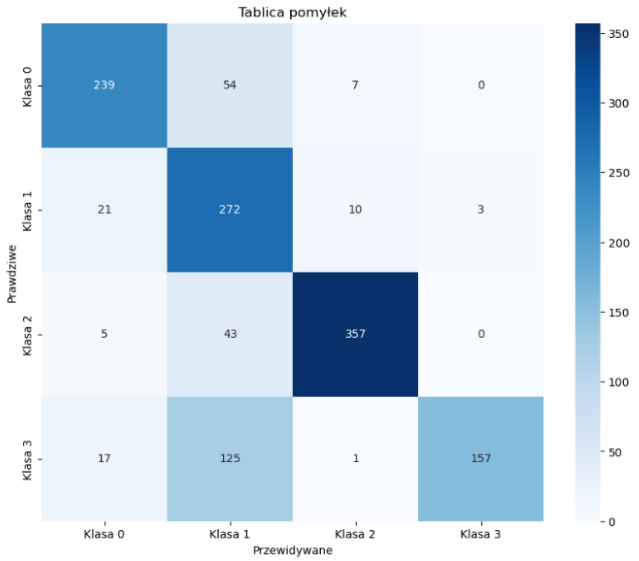
Dokładne wyniki oraz tablice pomyłek dla poszczególnych zbiorów testowych przedstawiono poniżej:

Zbiór testowy (MRI_Images):

Dokładność (Accuracy): 0.9046

Precyzja (Precision): 0.9088

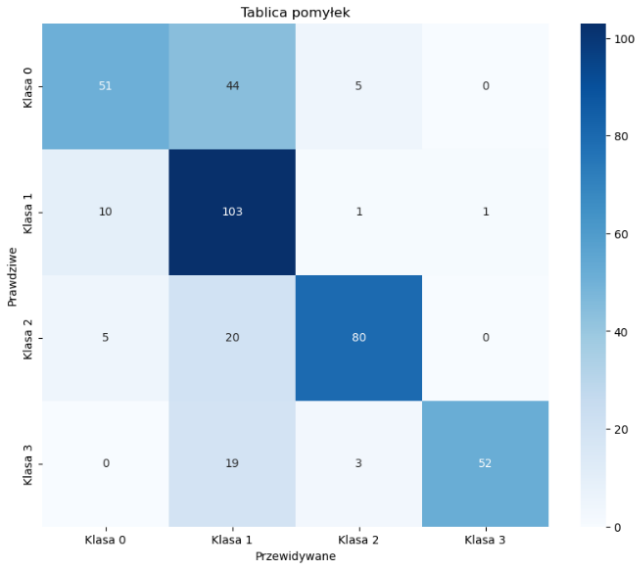
Czułość (Recall): 0.8997



Rysunek 32: Tablica pomyłek dla zbioru MRI_Images

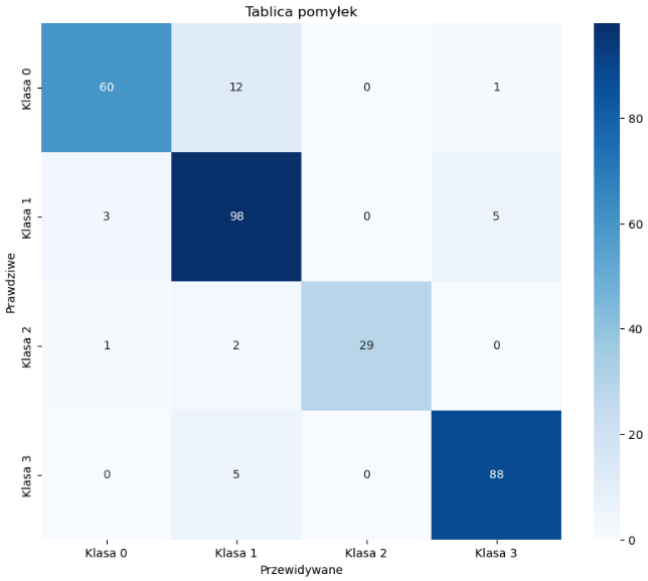
Zbiór 2 (Brain_Tumor_MRI_Dataset):

Dokładność (Accuracy): 0.7818
Precyzja (Precision): 0.8411
Czułość (Recall): 0.7726



Rysunek 33: Tablica pomyłek dla zbioru Brain_Tumor_MRI_Dataset

Zbiór 3 (Brain_Tumor_Classification_(MRI)):
Dokładność (Accuracy): 0.7259
Precyzja (Precision): 0.7816
Czułość (Recall): 0.7176



Rysunek 34: Tablica pomyłek dla zbioru Brain_Tumor_Classification_(MRI)

6 Podsumowanie wyników i wnioski

W tej części pracy podsumowano wyniki uzyskane podczas trenowania każdego z modeli na różnych zbiorach danych. Poddano również analizie i ocenie zdolności modelów do klasyfikacji obrazów ze zbioru, na którym były trenowane oraz na danych testowych z innych źródeł. Analiza pozwoli zrozumieć jak modele poradziły sobie z problemem generalizacji oraz przedstawić zaobserwowane trendy w wynikach.

Dla uproszczenia zapisu w dalszej części pracy przyjmowane będą poniższe skróty:

Model 1 - brain-tumor-mri-classification-tensorflow-cnn

Model 2 - braintumormri-using-mobilenetv2

Model 3 - mri-brain-tumor-classification

Model 4 – model przedstawiony w tej pracy

Zbiór 1 - Brain Tumor MRI Dataset

Zbiór 2 - Brain Tumor Classification (MRI)

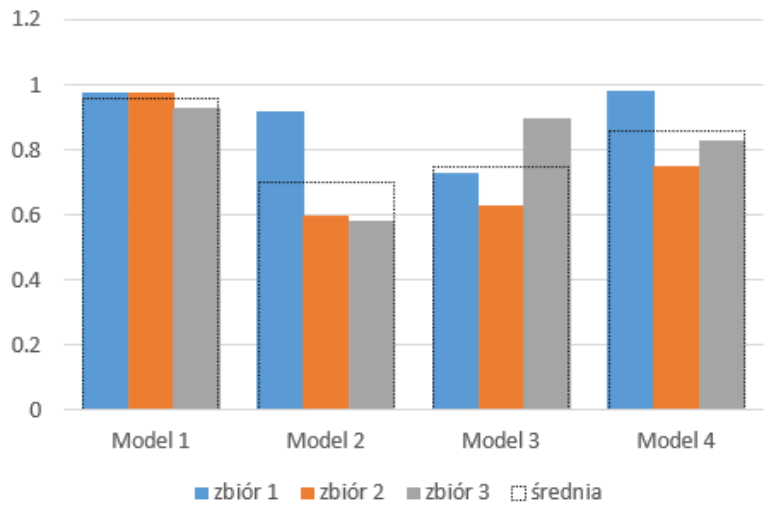
Zbiór 3 – MRI_Images

6.1 Wyniki i ich wizualizacja

Tabele wyników oraz wykresy dla zbiorów testowych dla poszczególnych metryk:

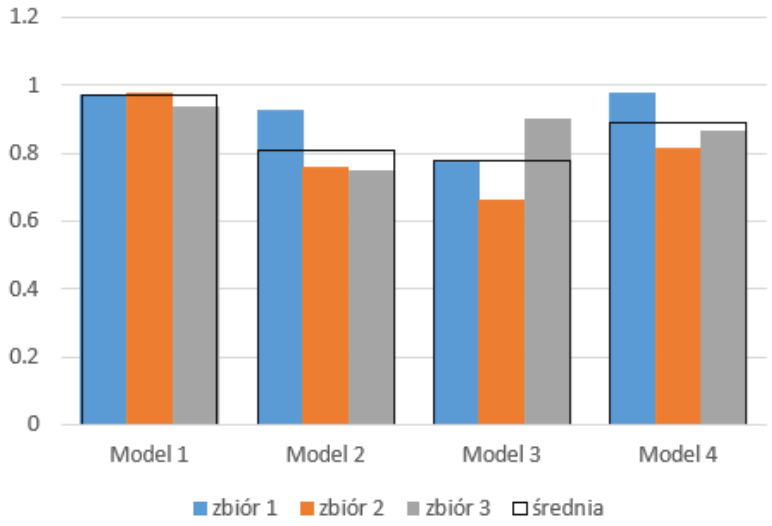
Dokładność:

	zbiór 1	zbiór 2	zbiór 3	średnia	odchylenie std.
Model 1	0.9809	0.9755	0.9539	0.97	0.01
Model 2	0.9177	0.6599	0.8191	0.8	0.13
Model 3	0.7818	0.7259	0.9046	0.8	0.09
Model 4	0.9931	0.7462	0.8553	0.86	0.12



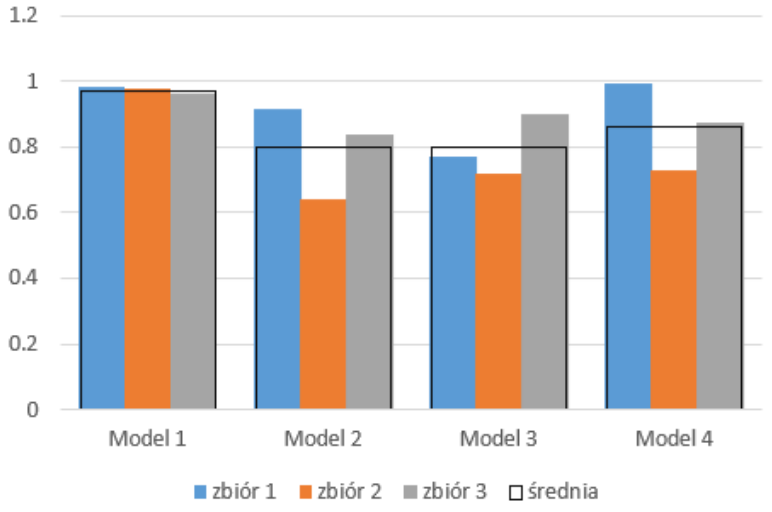
Precyzja:

	zbiór 1	zbiór 2	zbiór 3	średnia	odchylenie std.
Model 1	0.9812	0.9758	0.9606	0.97	0.01
Model 2	0.9222	0.7782	0.8537	0.85	0.07
Model 3	0.8411	0.7816	0.9088	0.84	0.06
Model 4	0.9932	0.8233	0.8772	0.9	0.09

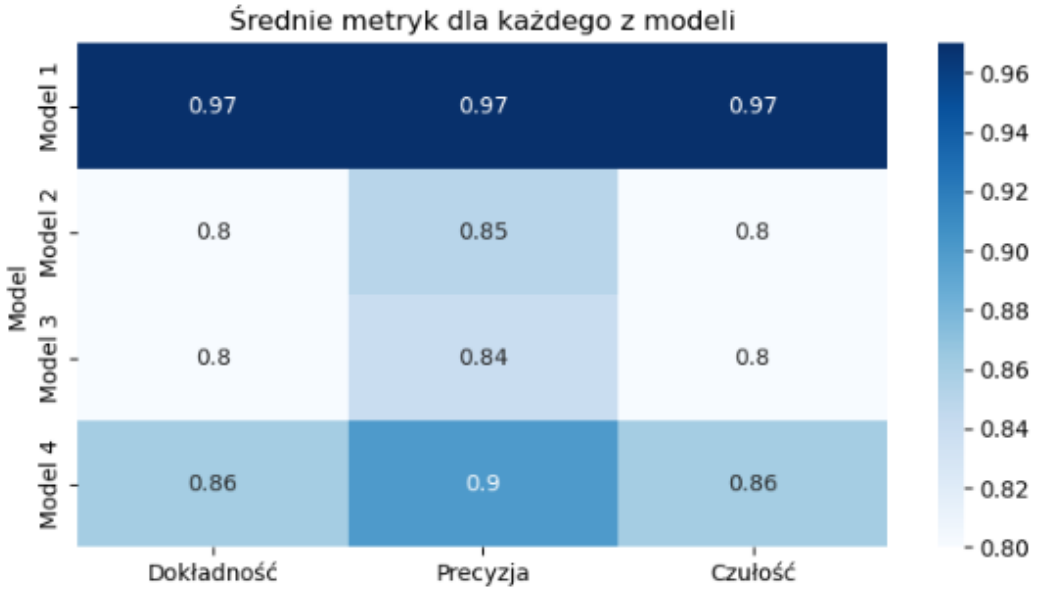


Czułość:

	zbiór 1	zbiór 2	zbiór 3	średnia	odchylenie std.
Model 1	0.9814	0.9762	0.9604	0.97	0.01
Model 2	0.9133	0.6423	0.8368	0.8	0.07
Model 3	0.7726	0.7176	0.8997	0.8	0.06
Model 4	0.9925	0.7298	0.872	0.86	0.09



Poniżej przedstawiono wizualizację - w skali kolorów - średnich metryk dla poszczególnych modeli by lepiej zobrazować wyniki:



6.2 Analiza wyników i wydajności modelów

Na podstawie uzyskanych wyników można wyciągnąć kilka wniosków dotyczących wydajności każdego z modeli oraz zbiorów danych.

Wydajność modeli na podstawie metryk:

- **Model 1**

  Najlepszym ze wszystkich modeli okazał sie model 1. Osiągnął on najyższe wyniki w każdej z badanych metryk. Wysokie wyniki uzyskał nie tylko dla zbioru testowego pochodzącego z tego samego źródła, ale również dla innych zbiorów danych.

- **Model 2**

  Ten model ma najniższą średnią dokładność wynoszącą 0.8. Pomimo uzyskania całkiem dobrych wyników dla zbioru pochodzącego z tego samego źródła co zbiór treningowy, model poradził sobie gorzej z klasyfikacją na pozostałych zbiorach, a w szczególności na zbiorze 2.

- **Model 3**

  Model 3 osiągnął niemal identyczne średnie wyniki jak model 2. Ten model również uzyskał zadowalające wyniki dla zbioru testowego pochodzącego z tego samego źródła co treningowy, wszystkie wynoszące blisko 0.9. Wyniki na pozostałych zbiorach są niewiele gorsze - nie przekraczają 0.8.

- **Model 4**

  Ten model ma wyższe średnie wyniki niż model 2 i 3, natomiast nie poradził sobie tak dobrze jak model 1. Średnie wyniki dla każdej z metryk są bliskie 0.9, jednak dla zbiorów pochodzących z innego źródła wypada gorzej niż model 1. Warto jednak zauważyć, że jego wyniki każdej z metryk dla zbioru testowego ze źródła, na któym był testowany jest wyższa nawet od modelu 1.

6.3 Podsumowanie

Model 4 osiągnął wyniki na zadowalającym poziomie. Każda z metryk uzyskała wynosiła blisko 0.9, co stanowi wysoki wynik przy zadaniu o takiej złożoności. Model ten uzyskał rezultaty lepsze niż model 2 i 3, co daje mu drugie miejsce, zaraz po modelu 1, który uzyskał najwyższe wyniki. Model 4 okazał siejednak lepszy od modelu 1 jeśli chodzi o wyniki każdej z metryk dla zbioru testowego pochodzącego z tego samego źródła co zbiór treningowy. Niestety w kontekscie tej pracy oznacza to przetrenowanie modelu, co nie jest korzystne dla klasyfikacji obrazów pochodzących z nowych źródeł.

Lepsza skuteczność modelu 1 może być wynikiem zastosowanego dynamicznego zmniejszania współczynnika uczenia. Jest to sposób, dzięki któremu model może bardziej dokładnie dopasować parametry modelu w trakcie uczenia. Użycie tego sposobu w modelu 4 mogłoby jeszcze bardziej zwiększyć wyniki jego metryk nawet na zbiorach pochodzących z innych źródeł. By jeszcze bardziej poprawić działanie zaproponowanego modelu na rzeczywistych obrazach można przeprowadzić proces treningu modelu na zbiorze danych zawierającym wszystkie omawiane zbiory. Takie rozwiązanie zwiększyłoby różnorodność danych treningowych prowadząć do lepszej generalizacji działania moddelu na nowych danych. Innym rozwiązaniem, które mogłoby poprawić skuteczność modelu jest podział obrazów na grupy, w zależności od płaszczyzny przekroju w jakiej wykonano zdjęcie. Powstałyby wtedy trzy różne modele, ale taki zabieg mógłby znacząco poprawić wyniki.

Bibliografia

[1] National Institute of Biomedical Imaging and Bioengineering (NIBIB), *Magnetic Resonance Imaging (MRI)*, dostęp online: 15.12.2024. Dostępne na: https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri

[2] Chris Hughes - Medium, *A Brief Overview of Cross Entropy Loss | by Chris Hughes | Medium*, dostęp online: 15.12.2024.Dostępne na: https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset

[3] Google for Developers, *Classification: Accuracy, recall, precision, and related metrics*, dostęp online: 28.12.2024.

Dostępne na: [4]https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

[4] Google for Developers, *Thresholds and the confusion matrix*, dostęp online: 28.12.2024.

Dostępne na: https://developers.google.com/machine-learning/crash-course/classification/thresholding#confusion_matrix

[5] Scikit-learn, *sklearn.metrics*, dostęp online: 31.12.2024. Dostępne na: https://scikit-learn.org/1.5/api/sklearn.metrics.html

[6] Masoud Nickparvar - Kaggle, *Brain Tumor MRI Dataset*, dostęp online: 28.12.2024.

Dostępne na: https://www.kaggle.com/datasets/masoudnickparvar

[7] Sartaj Bhuvaji - Kaggle, *Brain Tumor Classification (MRI)*, dostęp online: 28.12.2024.

Dostępne na: https://www.kaggle.com/datasets/sartajbhuvaji

[8] Susan Daneshmand - Kaggle, *MRI_ Images*, dostęp online: 28.12.2024. Dostępne na: https://www.kaggle.com/datasets/susandaneshmand/mri-images/data

[9] Sparsh Gupta - Kaggle, *Inception V3 Model* , dostęp online: 23.12.2024. Dostępne na: https://www.kaggle.com/datasets/imsparsh/inception-v3-model

[10] Stanford Vision Lab, *About ImageNet*, dostęp online: 23.12.2024. Dostępne na: https://www.image-net.org/about.php

[11] Jay Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Polo Chau, *CNN Explainer*, dostęp online: 31.12.2024. Dostępne na: https://poloclub.github.io/cnn-explainer/#article-pooling

[12] TensorFlow, *Module: tf.keras.layers*, dostęp online: 31.12.2024. Dostępne na: https://www.tensorflow.org/api_docs/python/tf/keras/layers/

[13] jaykumar1607 - Kaggle, *Brain Tumor MRI Classification: TensorFlow CNN*, do-

stęp online: 31.12.2024. Dostępne na: `https://www.kaggle.com/code/jaykumar1607/brain-tumor-mri-classification-tensorflow-cnn`

[14] abdullahsaida011 - Kaggle, *BrainTumorMRI using MobileNetV2*, dostęp online: 31.12.2024.

Dostępne na: https://www.kaggle.com/code/abdullahsaida011/braintumormri-using-mobilenetv2

[15] farzadnekouei - Kaggle, *Enhanced MRI Brain Tumor Classification* , dostęp online: 31.12.2024.

Dostępne na: https://www.kaggle.com/code/farzadnekouei/enhanced-mri-brain-tumor-classification