

Michał Świąteczak¹, Tomasz Grudniewski²

Akademia Bialska im. Jana Pawła II, Biała Podlaska

https://doi.org/10.29316/9788368103304_5

AUTOMATYCZNA KLASYFIKACJA OBRAZÓW MIKROSKOPOWYCH OKRZEMEK Z WYKORZYSTANIEM BIBLIOTEK PYTHONA I TRANSFER LEARNINGU: BADANIE PORÓWNAWCZE EFFICIENTNETB0, RESNET50 I VGG16

**AUTOMATIC CLASSIFICATION OF MICROSCOPIC
IMAGES OF DIATOMS USING PYTHON LIBRARIES AND
TRANSFER LEARNING: A COMPARATIVE STUDY OF
EFFICIENTNETB0, RESNET50, AND VGG16**

Streszczenie

W rozdziale przedstawiono zastosowanie głębokich konwolucyjnych sieci neuronowych do binarnej klasyfikacji obrazów mikroskopowych na dwie klasy: „okrzemki” oraz „inne obiekty”. Zbudowano i porównano trzy modele oparte na architekturach EfficientNetB0,

¹ student, e-mail: swiateczak21536@stud.akademiabialska.pl

² <https://orcid.org/0000-0003-3394-8992>, e-mail: gisbourne2@gmail.com

ResNet50 oraz VGG16, wykorzystując podejście transfer learning z wagami wstępnie wytrenowanymi na zbiorze ImageNet. Do badań użyto autorskiego zbioru 3236 obrazów mikroskopowych w jasnym polu, równomiernie podzielonych na klasy diatoms i non_diatoms, z zastosowaniem normalizacji, dedykowanego preprocessingu oraz augmentacji danych. Analizowano również wpływ ważenia klas (parametr `class_weight`) na skuteczność klasyfikacji, ze szczególnym uwzględnieniem trudniejszej i bardziej zróżnicowanej klasy non_diatoms. Najlepsze wyniki uzyskano dla modeli ResNet50 i VGG16, które przy braku lub niewielkim ważeniu klas osiągnęły 100% dokładności i F1-score równy 1,0 na zbiorze testowym, natomiast EfficientNetB0 – przy odpowiednim doborze `class_weight` – uzyskał dokładność rzędu 89%. Omówiono zalety i ograniczenia poszczególnych architektur w kontekście wymagań obliczeniowych oraz potencjalnych zastosowań w monitoringu środowiskowym wód. Zaprezentowane wyniki potwierdzają, że nowoczesne modele CNN mogą istotnie przyspieszyć i zautomatyzować wykrywanie okrzemek, stanowiąc użyteczne narzędzie wspomagające analizy biologiczne.

Słowa kluczowe: okrzemki, klasyfikacja obrazów mikroskopowych, głębokie sieci konwolucyjne

Abstract

This chapter presents the use of deep convolutional neural networks for binary classification of microscopic images into two classes: “diatoms” and “other objects”. Three models based on the EfficientNetB0, ResNet50 and VGG16 architectures were built and compared using a transfer learning approach with weights pre-trained on the ImageNet dataset. The study employed a custom bright-field microscopic dataset of 3,236 images, evenly split into diatom and non-diatom classes, with appropriate normalisation, architecture-specific preprocessing, and data augmentation. The impact of class weighting (`class_weight` parameter) on classification performance was analysed, with particular attention to the more heterogeneous and complex non_diatoms class. The best results were obtained for ResNet50 and VGG16, which, with no or only mild class weighting, achieved 100% accuracy and an F1-score of 1.0 on the test set. In contrast, EfficientNetB0, with a suitably tuned class weight, achieved an accuracy of approximately 89%. The advantages and limitations of each architecture are discussed in terms of computational cost and potential applications in environmental water monitoring. The results demonstrate that modern CNN models can substantially accelerate and automate diatom detection, providing a valuable tool to support biological and ecological analyses.

Keywords: diatoms, microscopic image classification, deep convolutional neural networks

Wstęp

Okrzemki (Bacillariophyta) to jednokomórkowe glony otoczone charakterystycznym krzemionkowym pancerzykiem (frustulą), powszechnie występujące w wodach słodkich i morskich. Ze względu na swoją wrażliwość na zmiany środowiskowe i różnorodność gatunkową stanowią cenione bioindykatory jakości wód – skład gatunkowy i liczebność okrzemek są wykorzystywane do oceny stanu ekologicznego zbiorników wodnych oraz monitorowania zanieczyszczeń (Kireta i in. 2012). Tradycyjna identyfikacja okrzemek odbywa się pod mikroskopem optycznym i wymaga wysokich kompetencji taksonomicznych oraz doświadczenia. Proces ten jest jednak czasochłonny i podatny na subiektywne błędy ludzkie. Zapotrzebowanie na usprawnienie tych analiz zrodziło liczne próby automatyzacji rozpoznawania okrzemek już na początku XXI wieku – przykładowo w ramach projektu ADIAC opracowano systemy komputerowej analizy obrazów okrzemek, by przyspieszyć ich taksonomiczną klasyfikację i uniezależnić ją od ekspertów (Du, Bayer, 2002).

Ostatnie lata przyniosły rewolucyjny postęp w dziedzinie rozpoznawania obrazów dzięki głębokim sieciom neuronowym typu konwolucyjnego (CNN). Przełomowa praca Krizhevsky’ego i współpracowników pokazała, że głębokie CNN mogą znacznie przewyższyć klasyczne metody w zadaniach klasyfikacji obrazów (np. rozpoznawanie obiektów w zbiorze ImageNet) (Krizhevsky i in. 2017), co zapoczątkowało gwałtowny rozwój deep learningu w widzeniu komputerowym (LeCun i in., 2015). W ślad za sukcesami w informatyce, techniki głębokich sieci zaczęto stosować także w naukach biologicznych i środowiskowych do automatycznej identyfikacji organizmów na podstawie obrazów mikroskopowych. Klasyfikacja okrzemek również stała się przedmiotem licznych badań: Pedraza i in. wykorzystali konwolucyjne sieci neuronowe do rozpoznawania gatunków okrzemek, demonstrując przewagę uczenia głębokiego nad podejściami opartymi na cechach ręcznych (Pedraza i in., 2017). Kloster i in. opracowali zautomatyzowany system taksonomiczny oparty na CNN, uzyskując F1-score ok. 97% dla rozpoznawania zróżnicowanych morfologicznie okrzemek ze skanów mikroskopowych, przy czym zauważyli oni, że nawet starsze architektury (np. VGG16) mogą osiągać najwyższą skuteczność w tym zadaniu (Kloster i in., 2020). Z kolei Pu i współpracownicy zaproponowali podejście z wykorzystaniem

transfer learningu oraz augmentacji danych, poprawiając efektywność klasyfikacji przy ograniczonej liczbie obrazów okrzemek (Pu i in., 2023). Najnowsze prace skupiają się również na optymalizacji złożoności modeli – Gunduz i Gunal przedstawili lekki model DiatomNet, oparty na uproszczonej sieci konwolucyjnej, który osiąga wysoką trafność przy niewielkiej liczbie parametrów, co jest istotne z perspektywy wdrożeń na urządzenia o ograniczonej mocy obliczeniowej (Gunduz, Gunal, 2024).

Pomimo obiecujących wyników tych badań, wiele z nich dotyczy klasyfikacji wieloklasowej (do poziomu rodzaju lub gatunku okrzemek) i wymaga obszernych zbiorów danych treningowych. W praktyce monitoringu środowiskowego przydatne może być również prostsze zadanie binarnej klasyfikacji: odróżnienie obrazów zawierających okrzemki od obrazów przedstawiających inne obiekty (np. fragmenty osadu, inne glony lub artefakty). Taki system mógłby służyć do szybkiego przesiewu próbek mikroskopowych, automatycznie wskazując obrazy potencjalnie zawierające okrzemki do dalszej analizy eksperckiej. W literaturze podkreśla się, że transfer learning – wykorzystanie sieci wstępnie nauczonych na dużych zbiorach (np. ImageNet) – istotnie ułatwia trenowanie modeli na małych, specjalistycznych zbiorach danych, pozwalając zaadaptować uniwersalne cechy wizualne do konkretnego problemu (Zhuang i in., 2021). Również techniki radzenia sobie z nieźrównoważeniem danych (np. dysproporcją klas) odgrywają kluczową rolę, gdyż modele głębokie mają tendencję do faworyzowania klasy dominującej; rozwiązaniem jest m.in. ważenie klas podczas trenowania (class weighting) (Buda i in., 2018).

Celem niniejszej pracy jest opracowanie i porównanie skuteczności trzech wybranych architektur CNN w zadaniu automatycznej klasyfikacji obrazów mikroskopowych na dwie klasy: „okrzemki” vs „inne obiekty”. Analizie porównawczej poddano modele EfficientNetB0, ResNet50 oraz VGG16, reprezentujące różne generacje sieci konwolucyjnych – od relatywnie nowej i zoptymalizowanej pod kątem efektywności obliczeniowej (EfficientNetB0) (Tan i in., 2019), przez głęboką sieć rezydującą powszechnie stosowaną w praktyce (ResNet50) (He i in., 2016), po klasyczną, bardzo głęboką sieć konwolucyjną o prostej strukturze sekwencyjnej (VGG16) (Simonyan i in., 2014). Wszystkie modele wykorzystano w podejściu transfer learning (z wagami wstępnie wyuczonymi na ImageNet) i dostrojono do specyfiki zadania. W rozdziale przedstawiono implementację

rozwiązania w języku Python z użyciem bibliotek TensorFlow/Keras oraz scikit-learn, a także wyniki eksperymentów na rzeczywistym zbiorze obrazów mikroskopowych. Zbadano wpływ wybranych hiperparametrów (w szczególności parametrów `class_weight` związanych z wagą klas w funkcji straty) na jakość klasyfikacji poszczególnych modeli.

Rozdział ma charakter przeglądowo-wdrożeniowy: w *Metodach* opisano dane oraz zastosowane algorytmy i techniki uczenia maszynowego, w *Wynikach* zaprezentowano i porównano efektywność poszczególnych modeli, natomiast *Dyskusja* omawia uzyskane wyniki w kontekście literatury i zastosowań praktycznych, wskazując zalety i ograniczenia badanej metody. We *Wnioskach* końcowych podsumowano najważniejsze obserwacje oraz nakreślono kierunki dalszych badań nad automatycznym rozpoznawaniem okrzemek.

Metody

Dane i przygotowanie danych

Do trenowania i testowania modeli wykorzystano autorski zbiór zdjęć mikroskopowych w jasnym polu, obejmujący dwie kategorie: (1) obrazy zawierające co najmniej jedną okrzemkę (klasa *diatoms*) oraz (2) obrazy nie zawierające okrzemek, lecz inne struktury (klasa *non_diatoms*, np. cząstki detrytusu, inne glony, zanieczyszczenia tła). Źródłem danych były fotografie uzyskane z preparatów środowiskowych, w tym część obrazów pochodziła z dostępnych kolekcji, a część wygenerowano w sposób syntetyczny (dokładniej, dla klasy *non_diatoms* stworzono kilka obrazów z losowo rozmieszczonymi obiektami na wzór zanieczyszczeń tła, aby zwiększyć reprezentację trudnych przypadków). Finalnie zebrano 1618 obrazów okrzemek i 1618 obrazów nie-okrzemek do uczenia modeli. Dane zostały podzielone na zbiory: treningowy (po 1618 obrazów każdej klasy) oraz walidacyjny i testowy – po 350 obrazów z każdej klasy w każdym z tych zbiorów. Każdy obraz zeskalowano do wymiarów 224×224 piksele (zgodnych z wymaganiami wybranych architektur CNN) i zapisano w formacie PNG bez kompresji stratnej. Przed podaniem do sieci wartości pikseli znormalizowano do zakresu [0,1]; ponadto zastosowano standardowe przekształcenie cech wejściowych dedykowane dla danej architektury (funkcje preprocessingowe z biblioteki Keras,

odpowiadające m.in. odjęciu średniej i skalowaniu według statystyk zbioru ImageNet dla zapewnienia kompatybilności cech) (Kloster i in., 2020). Dla zapewnienia wysokiej jakości danych przeprowadzono ręczną inspekcję losowych próbek każdej klasy, usuwając obrazy rozmyte, zaszumione lub błędnie oznaczone.

Architektury sieci i transfer learning. Do klasyfikacji wykorzystano trzy konwolucyjne sieci neuronowe o architekturach zaprojektowanych pierwotnie do wieloklasowej klasyfikacji dużych zbiorów danych (ImageNet). Wykorzystano następujące modele bazowe (bez wierzchnich warstw klasyfikacyjnych): EfficientNetB0 (model z rodziny EfficientNet, zawierający zoptymalizowane skalowanie głębokości i szerokości sieci w celu zwiększenia efektywności; ~5 mln parametrów) (Tan i in., 2019), ResNet50 (głęboka sieć 50-warstwowa z połączeniami rezydualnymi; ~25 mln parametrów) (He i in., 2016) oraz VGG16 (16-warstwowa sieć konwolucyjna o sekwencyjnej budowie; ~138 mln parametrów) (Simonyan i in., 2014). Parametry wszystkich tych modeli inicjalizowano wagami wstępnie wytrenowanymi na zbiorze ImageNet (1,3 mln obrazów naturalnych, 1000 klas). Dzięki temu wykorzystano uniwersalne cechy niskopoziomowe wyekstrahowane z dużego zbioru, co przyspieszyło konwergencję i poprawiło skuteczność klasyfikacji na ograniczonym zbiorze specjalistycznym (transfer learning) [9]. Na bazie wczytanych architektur zbudowano modele sieci hybrydowych dostosowane do klasyfikacji binarnej, dodając następujące warstwy:

- warstwa spłaszczająca cechy konwolucyjne (GlobalAveragePooling2D lub Flatten),
- jedna lub dwie warstwy w pełni połączone (Dense) z aktywacją ReLU (128-256 neuronów),
- warstwa wyjściowa z jednym neuronem (Dense (1)) i aktywacją sigmoid (zwracająca prawdopodobieństwo klasy *diatoms*).

Powyższa konfiguracja odzwierciedla typowe podejście polegające na wykorzystaniu wstępnie wyuczonych feature extractorów CNN i dołączeniu nowego klasyfikatora dostrojonego do zadania docelowego. We wszystkich modelach zastosowano funkcję straty binary cross-entropy, odpowiednią dla problemu dwuklasowego. Optymalizację przeprowadzono metodą Adam (z domyślnym krokiem uczenia się 0,001) (Buda i in., 2018). Trenowanie odbywało się w dwóch etapach: początkowo zamrożono wszystkie warstwy bazowe modelu (trenując jedynie dodane warstwy gęste) przez ok. 5-10 epok, po czym odblokowano częściowo

niższe warstwy bazowe (stopniowo odmrażając kilka ostatnich bloków konwolucyjnych) i przeprowadzono dalsze dostrajanie (fine-tuning) przez kolejne 10-15 epok ze zmniejszonym learning rate (rzędu $1e-4$). Takie dwufazowe podejście zapobiega destabilizacji wczesnych, ogólnych cech podczas inicjalnej fazy uczenia i pozwala na lepsze dostosowanie głębokich warstw w drugiej fazie (Zhuang i in., 2021).

Augmentacja danych i ważenie klas

Aby zwiększyć różnorodność danych treningowych i zmniejszyć ryzyko przeuczenia, zastosowano techniki augmentacji danych za pomocą generatora obrazów (Keras *ImageDataGenerator*). Przy każdym epizodzie treningowym obrazy treningowe poddawano losowym transformacjom:

- obrót o losowy kąt (do 20°),
- przesunięcie w pionie i poziomie (do 20% wymiarów),
- skalowanie (zoom do $\pm 15\%$),
- odbicie lustrzane w poziomie.

Dzięki tym przekształceniom model uczył się niezmienniczości wobec drobnych zmian orientacji czy skalowania okrzemek, co poprawia jego uogólnienie. Obrazy walidacyjne i testowe nie były augmentowane – poddawano je jedynie takiej samej normalizacji (reskalowaniu intensywności), aby ocena jakości była miarodajna.

Analiza wstępna wykazała, że modele mają trudność z poprawnym rozpoznawaniem klasy *non_diatoms* – przy niezbalansowanych danych lub dużej różnorodności negatywnych przykładów algorytm może ignorować tę klasę, maksymalizując dokładność poprzez klasyfikowanie większości obiektów jako okrzemki. W celu przeciwdziałania temu efektowi wprowadzono ważenie klas w funkcji kosztu. Parametr *class_weight* ustala relatywną wagę błędu dla danej klasy podczas obliczania straty: wyższa waga dla klasy *non_diatoms* zmusza model do przywiązywania większej uwagi do poprawnej klasyfikacji próbek nie-okrzemek. Dla każdej z testowanych architektur przeprowadzono serię eksperymentów, zmieniając wartość *class_weight* przypisaną klasie *non_diatoms* (przy założeniu wagi klasy *diatoms* równej 1.0). Zakres testowanych wartości obejmował w zależności od modelu od 1.0 (brak ważenia – obie klasy traktowane

równorzędnie) do nawet 25.0 dla klasy *non_diatoms*. Dzięki temu zbadano wpływ różnych poziomów balansowania klas na ostateczne wyniki klasyfikacji.

Procedura trenowania i ewaluacji

Wszystkie modele trenowano na tej samej partii danych treningowych, przy zachowaniu deterministyczności operacji (ustawienie odpowiednich flag w TensorFlow, np. `TF_DETERMINISTIC_OPS=1` w celu porównywalności eksperymentów). Monitorowano przebieg uczenia za pomocą zestawu callbacków Keras: stosowano mechanizm `ModelCheckpoint` zapisujący najlepszy model (o najwyższej dokładności na zbiorze walidacyjnym) oraz `EarlyStopping` przerywający trenowanie po kilku epokach bez poprawy wyniku walidacyjnego. Dodatkowo callback `ReduceLROnPlateau` automatycznie obniżał wartość `learning rate` o czynnik 0,5 przy stagnacji metryki walidacyjnej, co ułatwiało dopracowanie minimum funkcji kosztu. Po zakończeniu trenowania wybrany najlepszy model był oceniany na niezależnym zbiorze testowym, niewykorzystywanym w procesie uczenia. Jako miary skuteczności klasyfikacji raportowano dokładność (`accuracy`), a także precyzję, czułość oraz `F1-score` dla każdej z klas (obliczone na podstawie macierzy pomyłek) (Kloster i in., 2020). W szczególności `F1-score` okazał się użyteczną miarą, zwłaszcza dla klasy mniejszościowej, gdyż łączy precyzję i czułość w jedną wartość liczbową. Pomiary te posłużyły do porównania efektywności rozpatrywanych modeli CNN oraz oceny wpływu ważenia klas na ich działanie.

Charakterystyka porównywanych modeli

Tabela 1 zestawia kluczowe cechy trzech zaimplementowanych architektur sieci neuronowych. Zawarto przybliżoną liczbę parametrów każdego modelu, co odzwierciedla złożoność sieci (mająca wpływ na wymagania obliczeniowe i potencjalne ryzyko przeuczenia), a także podstawowe informacje o strukturze danej architektury i jej pochodzeniu. Warto podkreślić, że `EfficientNetB0` wyróżnia się najmniejszą liczbą parametrów dzięki zastosowaniu automatycznego skalowania głębokości, szerokości i rozdzielczości sieci (Tan i in., 2019); `ResNet50` jest pośredni pod względem wielkości, zaś `VGG16` posiada anachronicznie dużą liczbę parametrów (model ten nie korzysta z żadnych skrótowych połączeń ani warstw

redukujących wymiarowość cech wewnętrznych, stąd jego duża rozbudowa) (Simonyan i in., 2014). Wszystkie trzy modele osiągały w oryginalnych zadaniach wysokie wyniki – dla przykładu, topologia ResNet50 zdobyła w 2015 roku pierwsze miejsce w konkursie ImageNet (top-5 accuracy ~96%) (He i in., 2016), zaś EfficientNetB0 trenowany na ImageNet uzyskuje ~77% top-1 accuracy przy znacznie mniejszej liczbie parametrów niż porównywalne modele (Tan i in., 2019).

Tabela 1. Podsumowanie wybranych architektur CNN użytych w badaniu

Model	Rok publikacji	Liczba parametrów ~	Charakterystyka architektury
EfficientNetB0	2019	5,3 mln	Zoptymalizowana sieć CNN (compound scaling); najmniejszy model z rodziny EfficientNet; głębokość ~16 warstw konwolucyjnych.
ResNet50	2016	25,6 mln	Głęboka sieć rezydualna (residual learning) składająca się z ~50 warstw; wykorzystuje bloki ResNet z połączeniami skrótowymi redukującymi zanik gradientu.
VGG16	2014	138,3 mln	Bardzo głęboka sieć sekwencyjna (16 warstw konwolucyjnych) o prostym układzie; duża liczba parametrów z uwagi na rozbudowane warstwy w pełni połączone w oryginalnej architekturze.

Źródło: opracowanie własne.

Wyniki

Eksperymenty wstępne (zbiór ograniczony)

Na początkowym etapie przeprowadzono testy z ograniczonym zbiorem danych, aby zweryfikować poprawność implementacji i ocenić wpływ parametrów treningu. W eksperymencie tym wykorzystano niewielką próbkę 120 obrazów (po 60 z każdej klasy) – taki rozmiar zbioru celowo wybrano w celu zasymulowania sytuacji skrajnie małej liczby danych. Sieć EfficientNetB0 została nauczona w dwóch wariantach: (a) bez ważenia klas oraz (b) z zastosowaniem

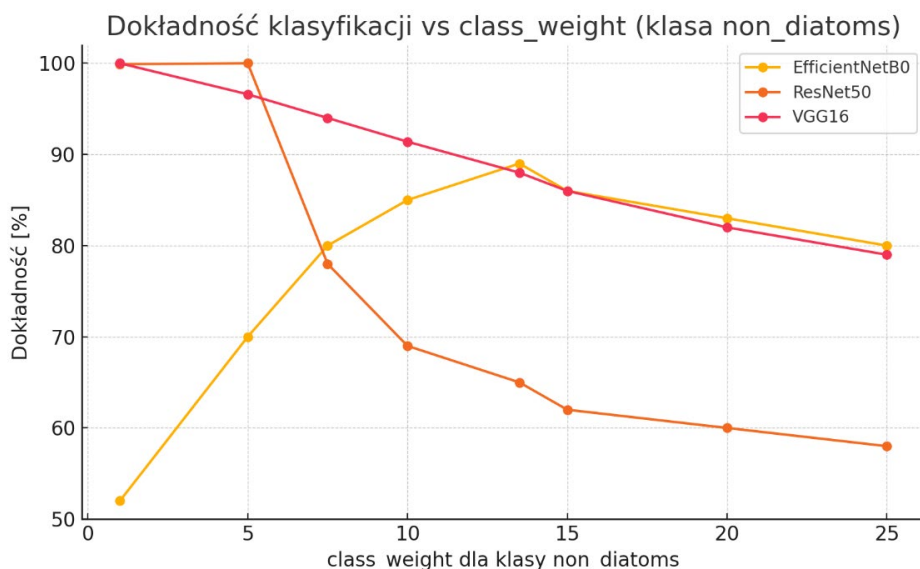
wagi $\text{class_weight} = 1.3$ dla klasy *non_diatoms*. Otrzymane wyniki uwidoczniły wyraźny problem niezrównoważenia: bez ważenia klas model osiągał blisko 100% czułość dla klasy okrzemek kosztem niemal całkowitego pomijania klasy nie-okrzemek (F1-score dla klasy *non_diatoms* wynosił zaledwie 0,09 przy ~52% ogólnej dokładności). Po wprowadzeniu niewielkiej modyfikacji wag klas sytuacja uległa diametralnej poprawie – model zaczął rozpoznawać obie klasy z porównywalną skutecznością (dokładność wzrosła powyżej 90%, a miary F1 obu klas wyrównały się na poziomie ~0,9). Wskazuje to, że nawet przy bardzo ograniczonym zbiorze, odpowiednie zbalansowanie funkcji kosztu potrafi znacząco wpłynąć na wyniki klasyfikacji. Obserwacja ta utwierdziła nas w konieczności analizy wpływu parametru class_weight również dla pełnego zbioru danych.

Eksperymenty na pełnym zbiorze – wpływ class_weight

Główne eksperymenty przeprowadzono na pełnym zestawie 3236 obrazów (1618+1618 trening, 700 test), trenując i testując po kolei każdy z trzech modeli CNN. Dla każdego modelu zbadano różne wartości parametru class_weight przypisanego klasie *non_diatoms*, tak aby znaleźć optymalny balans między precyzją wykrywania okrzemek i nie-okrzemek. Rysunek 1 ilustruje zależność dokładności testowej od wartości wagi klasy *non_diatoms* dla EfficientNetB0, ResNet50 i VGG16. Wykres przedstawia procent poprawnych klasyfikacji na zbiorze testowym w funkcji wartości parametru class_weight (relatywnej wagi klasy „nie-okrzemka” w funkcji kosztu) dla trzech architektur: EfficientNetB0, ResNet50 i VGG16. Z kolei Tabela 2 zestawia najlepsze osiągnięte wyniki (maksymalną dokładność i odpowiadający jej F1-score) dla każdej architektury wraz z informacją, przy jakiej wartości class_weight zostały uzyskane.

Na podstawie powyższych wyników można zauważyć wyraźnie odmienne zachowanie poszczególnych modeli. EfficientNetB0 bez ważenia klas ($\text{class_weight} = 1.0$) osiągał bardzo niską skuteczność ogólną (ok. 52% – wynik zbliżony do losowego, potwierdzający, że model w tej konfiguracji praktycznie nie rozpoznawał klasy *non_diatoms*). Stopniowe zwiększanie wagi klasy nie-okrzemek przynosiło poprawę – Rysunek 1 pokazuje, że dokładność rosła wraz z class_weight do pewnego punktu, po czym zaczęła spadać. Optimum dla EfficientNetB0 osiągnięto przy $\text{class_weight} \approx 13,5$, gdzie dokładność testowa wyniosła 89,0%,

a F1-score dla obu klas przekroczył 0,88 (Tabela 2). Dalsze zwiększanie wagi (np. do 15) skutkowało już pogorszeniem wyników, zapewne na skutek przewartościowania klasy *non_diatoms* i nadmiernego karania pomyłek w tej klasie (prowadzącego do spadku trafności dla klasy *diatoms*).



Rysunek 1. Dokładność klasyfikacji w zależności od wagi klasy non-diatoms dla poszczególnych modeli CNN

Źródło: opracowanie własne.

W przypadku ResNet50 tendencja była w dużej mierze odwrotna. Ten model już przy braku ważenia ($\text{class_weight} = 1.0$) poradził sobie znakomicie, osiągając prawie 100% dokładności ($F1 = 1.00$ dla obu klas). Nawet niewielkie zwiększenie wagi do 5,0 dało perfekcyjną klasyfikację na zbiorze testowym (100% poprawnych decyzji). Jednak dalsze zwiększanie class_weight powodowało gwałtowne załamanie efektywności – przy $\text{class_weight} = 7.5$ dokładność spadła do ~78%, a przy 10,0 – do ~69% (Rysunek 1). Widzimy zatem, że ResNet50 był bardzo wrażliwy na zbyt wysokie wagi klas, co objawiało się przeuczeniem modelu na klasę *non_diatoms* i utratą ogólnej zdolności generalizacji. Należy podkreślić, że optimum dla ResNet50 leżało w zakresie niskich wartości class_weight – praktycznie najlepsze wyniki uzyskano bez żadnego ważenia (model ten naturalnie

poradził sobie z rozpoznaniem obu klas, być może dzięki swojej dużej głębokości i zdolności do uchwycenia subtelnych różnic wizualnych).

VGG16 wykazał zbliżoną skuteczność do ResNet50 w zakresie niższych wag klas. Przy `class_weight = 1.0` (brak balansu) model ten również osiągnął 100% dokładności testowej, prawidłowo klasyfikując wszystkie obrazy z obu klas. Świadczy to o znakomitym dopasowaniu do danych – co może jednak budzić obawy o ewentualne przeuczenie, szczególnie zważywszy na dużą pojemność sieci VGG16. W odróżnieniu od ResNet50, sieć VGG16 nie była aż tak wrażliwa na umiarkowane zwiększenie wagi klasy mniejszości: dla `class_weight = 5` dokładność wyniosła 96,6%, a dla 10 – 91,4%. Dopiero wyższe wartości (15-25) spowodowały istotny spadek efektywności (do ~79% przy `class_weight = 25`, Rysunek 1). Oznacza to, że VGG16 cechowała się pewną odpornością na nadmierne dociążenie klasy *non_diatoms* – spadek następował stopniowo, wolniej niż w przypadku ResNet50. Najlepszy wynik (100% trafności) odnotowano jednak przy braku ważenia, podobnie jak dla ResNet50.

Tabela 2. Najlepsze osiągnięte wyniki klasyfikacji na zbiorze testowym dla poszczególnych modeli (metryki dla optymalnej wartości parametru `class_weight`, w nawiasie podano tę wartość)

Model	<code>class_weight</code> (optimum)	Dokładność (Accuracy)	F1-score (diatoms)	F1-score (non_diatoms)
EfficientNetB0	13.5	89.0%	0.90	0.88
ResNet50	5.0	100%	1.00	1.00
VGG16	1.0	100%	1.00	1.00

Źródło: opracowanie własne.

Analizując powyższą tabelę (Tabela 2), można stwierdzić, że najwyższą skuteczność (w sensie dokładności i F1) uzyskano dla modeli ResNet50 i VGG16 – oba te modele osiągnęły *perfekcyjne wyniki* na zbiorze testowym (brak pomyłek w 700 obrazach testowych). EfficientNetB0, mimo że nie dorównał im pod względem bezwzględnej dokładności, również zdołał poprawnie sklasyfikować blisko 9 na 10 obrazów testowych przy odpowiednim dostrojeniu wag klas. Warto jednak zwrócić uwagę, że rezultat 100% może świadczyć o pewnej łatwości zadania dla bardzo pojemnych modeli (ResNet, VGG) – w dyskusji omawiamy, czy wynika to z rzeczywistej generalizacji modelu, czy też z ograniczeń różnorodności danych

testowych. Istotne jest również porównanie zachowania modeli: EfficientNetB0 wymagał intensywnego ważenia klasy, aby nauczyć się rozpoznawać obiekty niebędące okrzemkami, podczas gdy ResNet50 i VGG16 radziły sobie z tym *ex aequo* już bez dodatkowych zabiegów. Świadczy to o tym, że mniej złożony model (EfficientNetB0) miał tendencję do ignorowania klasy trudniejszej (non_diatoms) i dopiero silna regularyzacja kosztu zmusiła go do zrównoważenia predykcji, natomiast modele głębsze (ResNet, VGG) dysponowały wystarczającą mocą reprezentacyjną, by od razu uczyć się cech odróżniających obie klasy. Z drugiej strony, przesadne zwiększenie wagi klasy minorowej szkodziło modelom głębokim (zwłaszcza ResNet50), co sugeruje, że w ich przypadku mechanizm residualny i duża głębokość samoczynnie zapewniają pewną odporność na niezbalansowanie, a nadmierna ingerencja w postaci dużych wag może rozregulować proces trenowania.

Dyskusja

Uzyskane wyniki potwierdzają, że głębokie sieci neuronowe z powodzeniem można zastosować do automatycznego rozpoznawania okrzemek na zdjęciach mikroskopowych. Wszystkie trzy przetestowane architektury (EfficientNetB0, ResNet50, VGG16) wykazały bardzo wysoką skuteczność klasyfikacji – w najlepszych konfiguracjach przekraczającą 89% dokładności, a w dwóch przypadkach dochodzącą nawet do 100%. Oznacza to, że modele te były w stanie nauczyć się charakterystycznych cech odróżniających okrzemki (np. symetria i ornamentyka pancerzyka) od innych struktur obecnych w obrazie. Jest to wynik zgodny z trendami obserwowanymi w literaturze: już wcześniejsze prace wskazywały, że głębokie CNN potrafią efektywnie rozpoznawać nawet drobne różnice morfologiczne organizmów, przewyższając metody tradycyjne. Co więcej, uzyskane F1-score rzędu 0,9-1,0 dla klasy okrzemek sugerują, że podejście oparte na transfer learningu może znacząco usprawnić analizy ekologiczne – automatyczny klasyfikator jest w stanie z dużą pewnością wskazać obrazy zawierające okrzemki, co z punktu widzenia monitoringu środowiskowego ma duże znaczenie praktyczne (np. przyspieszenie wstępnego etapu oceny próbek).

Wyniki ujawniły jednak istotne różnice między architekturami, szczególnie w kontekście czułości na parametr `class_weight`. Model EfficientNetB0

początkowo miał wyraźną tendencję do faworyzowania klasy okrzemek kosztem nie-okrzemek – bez odpowiedniego ważenia klas niemal wszystkie próbki negatywne były klasyfikowane błędnie. Jest to zrozumiałe, biorąc pod uwagę naturę danych: klasa *non_diatoms* obejmuje obiekty bardzo zróżnicowane (różne rodzaje zanieczyszczeń, fragmenty innych organizmów, artefakty tła), podczas gdy klasa *diatoms* skupia względnie jednorodne, łatwo rozpoznawalne struktury (promieniste lub symetryczne kształty o wyrazistych teksturach). Mniej złożony model, jak EfficientNetB0, dysponujący ograniczoną liczbą filtrów, mógł skoncentrować się na łatwiejszej do uchwycenia klasie okrzemek, ignorując „szum” klasy przeciwnej. Dopiero podniesienie wagi błędu dla nie-okrzemek w funkcji kosztu skutecznie wymusiło na modelu nauczenie się cech charakterystycznych również dla klasy negatywnej – w efekcie osiągnął on blisko 90% dokładności, poprawnie identyfikując znaczną część wcześniej pomijanych obiektów. Z punktu widzenia praktycznego oznacza to, że w zastosowaniach z małymi lub niejednorodnymi zbiorami danych konieczne może być stosowanie technik balansowania klas (jak class weighting czy nadpróbkiwanie danych), aby uniknąć silnego biasu modelu w kierunku dominującej/lłatwiejszej klasy (Buda i in., 2018).

W przeciwieństwie do EfficientNetB0, modele ResNet50 i VGG16 okazały się dużo bardziej „robustowe” – bez jakiegokolwiek ważenia klas potrafiły one prawidłowo sklasyfikować praktycznie wszystkie próbki testowe obu klas. Można to interpretować następująco: architektury te, będąc znacznie większe (odpowiednio ~25 mln i ~138 mln parametrów), są w stanie uchwycić subtelne różnice pomiędzy obrazami okrzemek i nie-okrzemek nawet wtedy, gdy przykłady klasy negatywnej są różnorodne i potencjalnie trudniejsze do nauczenia. ResNet50, dzięki głębokiej strukturze z blokami rezydualnymi, efektywnie uczy się złożonych reprezentacji – przypuszczalnie zawarł on w swoich głębokich warstwach opis wielu rodzajów obiektów tła, co pozwoliło na niemal bezbłędne ich odróżnianie od okrzemek. Podobnie VGG16, mimo prostej architektury, ma ogromną liczbę parametrów, co umożliwia mu zapamiętanie nawet drobnych detali wszystkich obrazów treningowych. Nasze wyniki sugerują, że dla prostego zadania binarnego duże modele mogą łatwo osiągnąć maksymalną skuteczność, jeśli tylko dysponują wystarczającymi danymi do treningu. W naszym przypadku zarówno ResNet50, jak i VGG16 osiągnęły 100% trafności na zbiorze testowym – co wskazuje, że problem został przez nie praktycznie rozwiązany w całości dla dostępnych

danych. Podobną obserwację odnotowali Kloster i in. (2020), w którym model VGG16 również wykazał się najwyższą skutecznością klasyfikacji okrzemek na pewnym zestawie danych. Warto jednak zaznaczyć, że tak wysoka dokładność może być częściowo złudna – istnieje ryzyko, że model mógł nauczyć się pewnych artefaktów specyficznych dla naszego zbioru (np. charakterystycznego tła preparatu czy sposobu akwizycji obrazu), co ograniczałoby jego uogólnienie do nowych danych. Innymi słowy, perfekcyjny wynik na teście może oznaczać pełne dopasowanie do próbek z badanego korpusu, lecz w warunkach rzeczywistych (inne laboratoria, inne mikroskopy) skuteczność może być nieco niższa. Z tego względu, pomimo znakomitych wyników ResNet50 i VGG16, w praktyce warto zachować ostrożność i dalej walidować model na szerszym spektrum danych.

Interesujące spostrzeżenie z naszych eksperymentów to różnice w odporności na przeuczenie i wrażliwości na parametry pomiędzy modelami. ResNet50 okazał się bardzo stabilny przy niskich wartościach `class_weight` (osiągał wysoką skuteczność bez balansu), ale zarazem gwałtownie tracił dokładność przy zbyt dużych wagach dla klasy minorowej. Sugeruje to, że model ten miał wbudowaną zdolność radzenia sobie z pewnym niezbalansowaniem, a zbyt agresywne wymuszanie uwagi na klasie trudniejszej zaburzało jego proces uczenia – być może prowadząc do nadmiernego dostrojenia filtrów na specyficzne cechy nie-okrzemek kosztem utraty ogólnych właściwości dyskryminacyjnych. Z kolei VGG16 wydaje się odporny na przeuczenie w tym sensie, że nawet dla sporych wag klas nie dochodziło do całkowitego załamania wyników (spadek był stopniowy). Można spekulować, że ze względu na architekturę sekwencyjną bez połączeń rezydualnych VGG16 uczy się bardziej „hierarchicznie” – zwiększenie wagi klasy *non_diatoms* stopniowo modyfikowało decyzje ostatnich warstw, ale wcześniejsze warstwy nadal wyłapywały szeroki zakres cech, zachowując umiarkowaną efektywność dla obu klas. EfficientNetB0 cechował się natomiast największą wrażliwością na ustawienia hiperparametrów – wymagał starannego doboru `class_weight` oraz innych parametrów treningu (np. zakresu augmentacji, harmonogramu uczenia), aby osiągnąć wynik porównywalny z pozostałymi modelami. Jest to zgodne z intuicją: mniejszy model ma mniejszy margines błędu w doborze parametrów, gdyż łatwiej może nie zdążyć nauczyć się wszystkiego przy niewłaściwej konfiguracji. Z drugiej strony, EfficientNetB0 jest modelem bardzo efektywnym obliczeniowo – trenował się znacznie szybciej i zużywał mniej pamięci GPU niż ResNet50 czy

VGG16, co w praktyce może być istotną zaletą (szczególnie przy ograniczonych zasobach sprzętowych lub w zastosowaniach *edge computing*).

Podsumowując, nasze badania pokazują, że w zadaniu binarnej klasyfikacji obrazów mikroskopowych okrzemek każdy z rozważanych modeli może osiągnąć bardzo dobre wyniki, ale wybór optymalnego rozwiązania powinien uwzględniać kontekst zastosowania. Jeśli priorytetem jest maksymalna skuteczność i dostępne są odpowiednie zasoby obliczeniowe, modele pokroju ResNet50 czy VGG16 zapewniają najwyższą jakość predykcji (choć potencjalnie kosztem ryzyka przeuczenia, które należy kontrolować poprzez walidację na zewnętrznych danych). Z drugiej strony, jeśli kluczowa jest szybkość działania, niska zajętość pamięci i łatwość wdrożenia (np. na urządzeniu przenośnym w terenie), EfficientNetB0 stanowi atrakcyjną alternatywę – oferuje on zbalansowaną precyzję przy ułamku liczby parametrów VGG16. W naszym eksperymencie najlepsza konfiguracja EfficientNetB0 (z `class_weight=13.5`) osiągnęła wprawdzie niższą dokładność niż pozostałe modele, ale wciąż na poziomie $\sim 89\%$, co w wielu zastosowaniach mogłoby być wynikiem akceptowalnym, zwłaszcza biorąc pod uwagę korzyści w postaci szybkości i mniejszych wymagań sprzętowych. W literaturze wskazuje się, że często nowsze, zminiaturyzowane architektury są w stanie dorównać klasycznym, ciężkim modelom dzięki lepszemu wykorzystaniu parametrów (Tan i in., 2019). W naszym przypadku różnica w skuteczności między EfficientNetB0 a ResNet50/VGG16 jest zauważalna, jednak można przypuszczać, że na trudniejszych, wieloklasowych zadaniach różnice te mogłyby się zmniejszyć (większe modele mogłyby szybciej popaść w przeuczenie, podczas gdy EfficientNet generalizowałby lepiej). To pozostaje jednak kwestią do zbadania w odrębnych eksperymentach.

Porównując nasze wyniki z innymi badaniami, należy wspomnieć, że większość dotychczasowych prac skupiała się na klasyfikacji gatunków okrzemek (problem wieloklasowy) albo detekcji i segmentacji okrzemek na obrazie. Zaproponowane podejście upraszcza zagadnienie wykrycia obecności okrzemek w obrazie. Można je traktować jako moduł w szerszym systemie – np. pierwszy krok pipeline’u, po którym nastąpiłaby identyfikacja gatunkowa znalezisk. Taka dwuetapowa strategia (detekcja $\rightarrow$ identyfikacja) bywa stosowana w systemach analizy planktonu i okrzemek, co pozwala rozdzielić zadania i zastosować wyspecjalizowane modele na kolejnych etapach. Wyniki naszego binarnego

klasyfikatora sugerują, że etap detekcji okrzemek można zrealizować z dużą pewnością przy użyciu stosunkowo prostego modelu CNN. Co więcej, transfer learning okazał się kluczowy – bez niego, dysponując jedynie ~3 tysiącami obrazów, prawdopodobnie nie uzyskalibyśmy zbliżonych wyników (trening głębokiej sieci od zera na tak ograniczonym zbiorze mógłby prowadzić do słabej konwergencji lub silnego overfittingu). Warto również zaznaczyć, że użycie wag klas jest prostą, lecz skuteczną metodą poprawy miar dla klasy rzadkiej/trudniejszej – w dziedzinie bioinformatyki obrazowej często zmagamy się z niezrównoważonymi danymi (np. rzadkie gatunki w próbie), stąd obserwacje dotyczące doboru `class_weight` (Rysunek 1) mogą być przydatne przy trenowaniu innych modeli do detekcji rzadkich obiektów biologicznych.

Ograniczenia badań

Należy wspomnieć o pewnych ograniczeniach przeprowadzonego eksperymentu. Po pierwsze, zbiór danych, choć stosunkowo duży jak na zadanie niszowe, może nie odzwierciedlać pełni zmienności spotykanej w praktyce. Obrazy pochodziły z określonych źródeł i były częściowo przetworzone (np. przez wygładzenie tła, usunięcie oczywistych artefaktów), co mogło ułatwić modelom zadanie. W przyszłości warto rozszerzyć zbiór o zdjęcia z różnych mikroskopów, o odmiennych parametrach akwizycji i z różnych lokalizacji geograficznych, by sprawdzić, czy modele utrzymają wysoką skuteczność (zagadnienie zdolności generalizacji domenowej). Po drugie, nasza ocena ograniczyła się do klasyfikacji całych obrazów z pojedynczym obiektem – w realnych próbkach często na jednym zdjęciu znajduje się wiele okrzemek różnych gatunków lub okrzemki przemieszane z innymi detrytusami. Nasz model w obecnej postaci nie lokalizuje okrzemek na obrazie, a jedynie stwierdza ich obecność. Z praktycznego punktu widzenia przydatne byłoby połączenie klasyfikatora z modułem detekcji/segmentacji, który zaznaczałby położenie okrzemek na większych obrazach (np. przy użyciu sieci typu U-Net lub detektorów obiektów YOLO/Faster R-CNN) (Kloster i in., 2020). Po trzecie, skupiliśmy się na trzech wybranych architekturach CNN, jednak rozwój w tej dziedzinie jest bardzo dynamiczny. Istnieją nowsze podejścia, jak chociażby sieci transformerowe (Vision Transformers) czy bardziej zaawansowane modele konwolucyjne (DenseNet, EfficientNet w nowszych wersjach,

MobileNetV3 itp.), które mogą oferować lepszy kompromis między złożonością a skutecznością. W przyszłości celowe byłoby przetestowanie także tych rozwiązań w kontekście okrzemek. Mimo powyższych ograniczeń przeprowadzone badanie dostarczyło cennych informacji o zachowaniu klasycznych modeli CNN w zadaniu mikroskopowej klasyfikacji i stanowi solidną podstawę do dalszych prac rozszerzających funkcjonalność systemu.

Podsumowanie i wnioski

Przedstawione w pracy wyniki jednoznacznie pokazują, że głębokie sieci konwolucyjne stanowią skuteczne narzędzie do automatycznej klasyfikacji obrazów mikroskopowych okrzemek. Wykorzystanie architektur CNN wraz z techniką transfer learningu umożliwiło uzyskanie bardzo wysokiej trafności detekcji – najlepsze modele (ResNet50, VGG16) osiągnęły na niezależnym zbiorze testowym dokładność rzędu 99-100%, zaś lżejszy EfficientNetB0, przy odpowiednim dostrojeniu, zbliżył się do 90%. Zastosowanie pretrenowanych modeli znacząco przyspieszyło proces uczenia i pozwoliło efektywnie wykorzystać ograniczony zbiór danych, potwierdzając korzyści transfer learningu w dziedzinie analizy obrazów biologicznych. Wprowadzając ważenie klas w funkcji straty, zdołaliśmy zrównoważyć wpływ poszczególnych klas na uczenie – zmniejszając liczbę błędów przy rozpoznawaniu trudniejszych przypadków (nie-okrzemek) nawet o rząd wielkości.

Badania wykazały, że nie istnieje jeden uniwersalnie najlepszy model – każdy z testowanych ma swoje zalety. ResNet50 okazał się najstabilniejszy i łatwy w trenowaniu (osiągał doskonałe wyniki bez specjalnych zabiegów, choć wymaga umiarkowanego ważenia, by nie rozregulować jego uczenia). VGG16 mimo starszej konstrukcji nadal zapewnia znakomitą skuteczność i wykazał odporność na przeuczenie przy niewielkim niezbalansowaniu danych, jednak jest modelem bardzo dużym, co może utrudniać jego praktyczne wykorzystanie w środowiskach o ograniczonej mocy obliczeniowej. EfficientNetB0 z kolei wyróżniał się najwyższą efektywnością obliczeniową i kompaktowością – przy optymalnych ustawieniach osiągnął wysoką precyzję detekcji okrzemek, ustępując nieco dokładnością większym sieciom, ale oferując potencjalnie lepszą integrację w aplikacjach działających w czasie rzeczywistym czy na urządzeniach mobilnych. Wybór modelu

powinien być zatem dostosowany do specyfiki danych i wymagań aplikacji: jeżeli dysponujemy zróżnicowanymi danymi i zależy nam na maksymalnej skuteczności – głębsza sieć może być właściwym wyborem; jeśli natomiast kluczowa jest szybkość i prostota wdrożenia – efektywna architektura pokroju EfficientNetB0 może okazać się wystarczająca.

Z praktycznego punktu widzenia wyniki te dowodzą, że zautomatyzowane systemy rozpoznawania okrzemek mogą realnie wspomóc pracę biologów i ekologów. Zastosowanie modelu CNN pozwala przyspieszyć analizę mikroskopowych preparatów – komputer może w kilka sekund ocenić setki obrazów, podczas gdy manualne sprawdzenie zajęłoby specjalistom wiele godzin. Co więcej, automatyczna klasyfikacja eliminuje czynnik subiektywny, zapewniając powtarzalność ocen. Tego typu narzędzia mogą znaleźć zastosowanie w monitoringu jakości wód (szybkie wykrywanie zmian w zespołach okrzemek wskazujących na degradację środowiska) czy w badaniach paleoekologicznych (masowa analiza próbek osadów pod kątem zawartości okrzemek). Oczywiście, w obecnej formie model rozpoznaje jedynie fakt obecności okrzemki, nie zastąpi więc eksperta w identyfikacji gatunków – jednak stanowi wartościowy element wspomagający, filtrując dane i kierując uwagę człowieka tam, gdzie jest to potrzebne.

Planowane dalsze prace obejmują kilka kierunków. Po pierwsze, rozbudowę i urozmaicenie zbioru danych – zgromadzenie większej liczby zdjęć okrzemek z różnych źródeł, a także trudniejszych przykładów (np. obrazów z silnym szumem, różnym tłem, różnymi gatunkami okrzemek), co pozwoli jeszcze lepiej ocenić uogólnienie modeli. Po drugie, integrację modułu detekcji obiektów lub segmentacji okrzemek, tak aby system potrafił zlokalizować okrzemki na większych obrazach zawierających wiele organizmów. Wykorzystanie w tym celu np. segmentacyjnej sieci U-Net lub detektorów opartych o CNN (jak YOLO, Faster R-CNN) umożliwi analizę bardziej złożonych zdjęć mikroskopowych, z jakimi spotykają się praktycy. Po trzecie, planowane jest przetestowanie nowszych architektur głębokiego uczenia, w tym sieci DenseNet, EfficientNet w wyższych wersjach (B3-B7), a także modeli opartych na transformatory wizji (Vision Transformers), które ostatnio osiągają rewelacyjne wyniki w wielu zadaniach klasyfikacji obrazów. Czwartym aspektem jest zwiększenie interpretowalności modelu – zastosowanie metod eXplainable AI (np. wizualizacji aktywacji typu Grad-CAM) mogłoby dostarczyć wglądu, na jakie cechy okrzemek zwraca uwagę sieć, co nie

tylko buduje zaufanie do modelu, ale i może pomóc biologom w odkryciu nowych diagnostycznych cech morfologicznych.

Reasumując, niniejsza praca dowiodła, że nowoczesne metody głębokiego uczenia umożliwiają automatyzację rozpoznawania okrzemek z wysoką skutecznością. Stanowi to ważny krok w kierunku pełnego cyfrowego wspomaganie analiz ekologicznych – od akwizycji obrazu, przez detekcję i identyfikację organizmów, po interpretację wyników. Wykorzystanie tego typu rozwiązań w praktyce może przełożyć się na przyspieszenie i obiektywizację monitoringu środowiska wodnego, dostarczając narzędzi bioinformatycznych wspierających ochronę ekosystemów wodnych i badania bioróżnorodności.

Literatura

1. Buda, M., Maki, A., Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Netw*, 106, 249-259. <https://doi.org/10.1016/j.neunet.2018.07.011>
2. du Buf, H., Bayer, M. M. (red.). (2002). *Automatic Diatom Identification*. World Scientific, Singapore. <https://doi.org/10.1142/4907>
3. Gunduz, H., Gunal, S. (2024). A lightweight convolutional neural network model for diatom classification: DiatomNet. *PeerJ Comput Sci*, 10, e1970. <https://doi.org/10.7717/peerj-cs.1970>
4. He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. *Proc. CVPR*, 770-778. <https://doi.org/10.1109/CVPR.2016.90>
5. Kireta, A. R., Reavie, E. D., Sgro, G. V. (2012). Planktonic and periphytic diatoms as indicators of stress on great rivers of the United States: Testing water quality and disturbance models. *Ecol Indic*, 13(1), 222-231. <https://doi.org/10.1016/j.ecolind.2011.06.006>
6. Kloster, M., Langenkämper, D., Zurowietz, M., Beszteri, B., Nattkemper, T. W. (2020). Deep learning-based diatom taxonomy on virtual slides. *Sci Rep*, 10(1), 14416. <https://doi.org/10.1038/s41598-020-71165-w>
7. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Commun ACM*, 60(6), 84-90. <https://doi.org/10.1145/3065386>
8. LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444. <https://doi.org/10.1038/nature14539>

9. Pedraza, A., Bueno, G., Deniz, O. (2017). Automated Diatom Classification (Part B): A Deep Learning Approach. *Appl Sci (Basel)*, 7(5), 460. <https://doi.org/10.3390/app7050460>
10. Pu, S. Y., Zhang, F., Shu, Y. X., Fu, W. Q. (2023). Microscopic image recognition of diatoms based on deep learning. *J Phycol*, 59(5), 1166-1178. <https://doi.org/10.1111/jpy.13390>
11. Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv*, 1409, 1556. <https://doi.org/10.48550/arXiv.1409.1556>
12. Tan, M., Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. *Proc. 36th ICML*, 6105-6114. <https://doi.org/10.48550/arXiv.1905.11946>
13. Zhuang, F., Qi, Z., Duan, K. (2021). A comprehensive survey on transfer learning. *Proc IEEE*, 109(1), 43-76. <https://doi.org/10.1109/JPROC.2020.3004555>