

Politechnika Warszawska

W Y D Z I A Ł E L E K T R Y C Z N Y



Instytut Sterowania i Elektroniki Przemysłowej

Praca dyplomowa magisterska

na kierunku Informatyka Stosowana

w specjalności Inżynieria Oprogramowania

Analiza metod prototypowania w problemie rozpoznawania grzybów

inż. Mateusz Gietka

numer albumu 298941

promotor

dr inż. Grzegorz Sarwas

WARSZAWA 2024

Analiza metod prototypowania w problemie rozpoznawania grzybów

Streszczenie

Niniejsza praca porusza problem wykorzystania metody prototypowania w zadaniu rozpoznawania obrazów grzybów. Do badań wykorzystane zostały dwie metody ekstrakcji cech oparte na głębokim uczeniu: autokoder oraz ResNet50. W przeprowadzonych eksperymentach rozpoznawania grzybów wykorzystano algorytm k-Najbliższych Sąsiadów (k-NN) oraz metodę centroidów w różnych konfiguracjach, uwzględniając scenariusze z ograniczoną liczbą próbek treningowych. Dodatkowo, zastosowano algorytm t-SNE do analizy przestrzeni utajonej, co pozwoliło na lepsze zrozumienie reprezentacji cech generowanych przez modele.

Badania te stanowią podstawę do dalszego rozwoju metod rozpoznawania obrazów, szczególnie w scenariuszach wymagających wysokiej precyzji klasyfikacji lub pracy z ograniczonymi danymi. Uzyskane wnioski mogą znaleźć zastosowanie w różnych dziedzinach, takich jak ochrona środowiska, analiza obrazów diagnostycznych czy systemy wspomagania decyzji. Przedstawione podejścia wskazują również na potencjał dalszych badań nad bardziej zaawansowanymi architekturami, które mogą przyczynić się do poprawy efektywności systemów widzenia komputerowego.

Słowa kluczowe: prototypowanie, autoenkoder, CNN, ResNet, algorytmy klasyfikujące, rozpoznawanie obrazów, uczenie głębokie

Analysis of prototyping methods in the problem of mushroom recognition

Abstract

This thesis addresses the problem of using a prototyping method in the task of fungal image recognition. Two feature extraction methods based on deep learning were used for the study: autocoder and ResNet50. Fungal recognition experiments used the k-Nearest Neighbours (kNN) algorithm and the centroid method in different configurations, considering scenarios with a limited number of training samples. In addition, the t-SNE algorithm was used for latent space analysis, allowing a better understanding of the feature representation generated by the models.

This research provides a basis for further development of image recognition methods, especially in scenarios requiring high-precision classification or working with limited data. The findings obtained may find applications in various fields such as environmental protection, diagnostic image analysis or decision support systems. The approaches presented also indicate the potential for further research into more advanced architectures that can improve the efficiency of computer vision systems.

Keywords: prototyping, autoencoder, CNN, ResNet, classification algorithms, image recognition, deep learning

Spis treści

1 Wstęp	9
2 Powiązane prace	13
2.0.1 Zastosowane sieci neuronowe	15
2.0.2 Algorytm k-Najbliższych Sąsiadów (kNN)	20
2.0.3 Centroidy w analizie danych	22
2.0.4 Algorytm t-SNE	23
3 Eksperymenty	27
3.1 Specyfikacja Środowiska Eksperymentalnego i Oprogramowania	27
3.2 Narzędzia i Biblioteki	27
3.3 Eksperymenty z autokoderem i ResNet50	28
3.3.1 Autokoder	29
3.3.2 ResNet50	29
3.4 Wizualizacja t-SNE	29
3.5 Zbiór Danych	29
3.6 Ewaluacja i Metryki	30
3.6.1 Metryki Ewaluacji	30
3.6.2 Cechy dobrego modelu	31
3.6.3 Zalety i ograniczenia zastosowanych metryk	31
4 Wyniki	33
4.1 Wyniki autokodera	33
4.2 Wyniki ResNet50	38
4.3 Problemy Napotkane Podczas Treningu	41
4.4 Wnioski	45
4.4.1 Autokoder jako metoda ekstrakcji cech	45
4.4.2 ResNet50 jako ekstraktor cech	46
4.4.3 Porównanie metod klasyfikacji (kNN i centroidy)	46
4.4.4 Wizualizacje t-SNE	47
4.4.5 Praktyczne zastosowania wyników	47

5 Podsumowanie	49
Bibliografia	51
Spis rysunków	55
Spis tabel	57
Spis załączników	59

Rozdział 1

Wstęp

Rozpoznawanie obrazów to obszar widzenia komputerowego, który skupia się na identyfikacji oraz klasyfikacji obiektów lub cech wizualnych zapisanych na obrazach cyfrowych. Obecnie proces ten wiąże się z opracowywaniem modeli komputerowych, potrafiących rozpoznawać wzorce i struktury w danych wizualnych, pozwalających na ich praktyczne wykorzystanie w rzeczywistych problemach. Obecnie tej dziedziny znacznie wzrosło, co jest efektem rozwoju technologii cyfrowych, rosnącej dostępności dużych zbiorów danych oraz ciągle zwiększającej się mocy obliczeniowej komputerów. Rozpoznawanie obrazów znajduje zastosowanie w wielu obszarach, takich jak:

- Medycyna: Automatyczna analiza obrazów diagnostycznych, takich jak zdjęcia rentgenowskie czy MRI, wspierająca lekarzy w szybszym wykrywaniu chorób, w tym nowotworów oraz schorzeń neurologicznych [26].
- Przemysł: Systemy inspekcji wizualnej stosowane w produkcji umożliwiają identyfikację wad produktów, co przyczynia się do zwiększenia efektywności procesów produkcyjnych [28].
- Rolnictwo: Klasyfikacja roślin, identyfikacja szkodników oraz analizy stanu upraw wspierają optymalizację produkcji rolnej oraz ochronę środowiska [33].
- Rozpoznawanie twarzy: Systemy identyfikacji biometrycznej, wykorzystywane w bezpieczeństwie publicznym, aplikacjach mobilnych oraz kontroli dostępu [9].

W ostatnich latach rozwój technologii głębokiego uczenia maszynowego znacząco wpłynął na postępy w dziedzinie rozpoznawania wzorców. Wprowadzenie splotowych sieci neuronowych (CNN) [11] oraz innowacyjnych architektur, takich jak ResNet, zrewolucjonizowało tę branżę, umożliwiając osiąganie wyników zbliżonych do osiąganych przez ludzi nawet w skomplikowanych zadaniach związanych z widzeniem komputerowym.

Pomimo tych osiągnięć, obszar ten nadal posiada wiele wyzwań:

- Ograniczona ilość danych: W wielu dziedzinach, takich jak diagnostyka medyczna, dostępność oznaczonych danych jest ograniczona, co wymusza efektywne wykorzystanie małych zbiorów danych, jak w przypadkach few-shot i zero-shot learning [8].
- Różnorodność danych: Modele muszą być odporne na zmienność danych spowodowaną różnymi warunkami oświetleniowymi, kątami widzenia oraz zakłóceniami.

- Złożoność obliczeniowa: Głębokie modele wymagają dużych zasobów obliczeniowych, co może stanowić ograniczenie w aplikacjach działających w czasie rzeczywistym lub na urządzeniach o ograniczonej mocy obliczeniowej.

Celem niniejszej pracy było przeprowadzenie badań skuteczności metody prototypowania w zadaniu rozpoznawania obrazów grzybów oraz ocena jej efektywności w generowaniu reprezentacji cech wizualnych. Praca skupia się w szczególności na analizie korzyści uzyskanych z prototypowania, bazując na cechach uzyskanych przy pomocy autokodera i sieci ResNet50, w kontekście klasyfikacji przy użyciu algorytmu k-najbliższych sąsiadów (kNN) oraz metody centroidów. Badania te mają na celu zrozumienie potencjału głębokiego uczenia w rozwiązywaniu problemów klasyfikacyjnych w scenariuszach z ograniczoną liczbą danych.

Autokoder jest rodzajem sieci neuronowej stosowanej w zadaniach uczenia nienadzorowanego, którego celem jest kompresja danych wejściowych do reprezentacji o niższym wymiarze (przestrzeń utajona), a następnie rekonstrukcja danych z tej reprezentacji. Dzięki temu autokodery są często wykorzystywane do redukcji wymiarowości i ekstrakcji kluczowych cech, co czyni je użytecznymi w przetwarzaniu obrazów oraz analizie dużych zbiorów danych [23].

ResNet50 to głęboka sieć neuronowa zbudowana z wykorzystaniem innowacyjnej architektury z blokami rezydualnymi, która umożliwia trenowanie bardzo głębokich sieci bez problemu zanikania gradientu. ResNet50 znajduje szerokie zastosowanie w zadaniach wizji komputerowej, takich jak klasyfikacja obrazów czy wykrywanie obiektów, i jest znana ze swojej efektywności w generowaniu reprezentatywnych cech wizualnych [11].

Algorytm kNN (k-Nearest Neighbors) jest jednym z najprostszych, a zarazem najczęściej stosowanych algorytmów klasyfikacji w uczeniu maszynowym. Jego działanie polega na klasyfikacji nowych danych na podstawie analizy sąsiedztwa w przestrzeni cech. Punkt testowy jest przypisywany do klasy najczęściej występującej wśród k najbliższych sąsiadów w zbiorze treningowym, co czyni ten algorytm intuicyjnym i prostym w implementacji. Dzięki swojej elastyczności kNN znalazł szerokie zastosowanie w problemach takich jak klasyfikacja obrazów, analiza genetyczna czy wykrywanie anomalii [4, 8, 10, 34].

Centroidy w uczeniu maszynowym to punkty w przestrzeni cech, które reprezentują środek masy próbek należących do danej klasy. Wyznacza się je jako uśrednione wartości cech wszystkich próbek z danej klasy. Centroidy upraszczają proces klasyfikacji, zastępując dużą liczbę próbek jedną reprezentatywną wartością, co zmniejsza złożoność obliczeniową i pozwala na bardziej efektywne przetwarzanie danych.

Prototypy w uczeniu maszynowym oraz ekstrakcji cech to reprezentatywne punkty w przestrzeni cech, które stanowią uogólnione odwzorowanie danych należących do określonych klas. Ich celem jest uproszczenie procesu klasyfikacji poprzez redukcję liczby analizowanych elementów, co pozwala na zmniejszenie złożoności obliczeniowej i zwiększenie interpretowalności wyników. W kontekście klasyfikacji, prototypy mogą być wyznaczane jako centroidy klas, czyli średnie wartości reprezentacji cech dla wszystkich próbek danej klasy, lub inne reprezentacje wynikające z zastosowania zaawansowanych metod uczenia.

Podejście to znajduje szerokie zastosowanie w zadaniach, gdzie kluczowa jest efektywność obliczeniowa, takich jak klasyfikacja w czasie rzeczywistym czy systemy z ograniczoną ilością dostępnych danych. Prototypy są także używane w metodach few-shot learning, gdzie niewielka liczba próbek treningowych wymaga szczególnego podejścia do modelowania reprezentacji cech. Dzięki swojej prostocie i skuteczności, prototypy umożliwiają efektywne odzwierciedlenie struktury danych, wspierając proces klasyfikacji oraz analizy danych wizualnych.

Wyniki pracy mogą mieć potencjalnie szerokie zastosowanie w praktyce. W ekologii i ochronie środowiska zastosowanie takich metod może wspierać monitorowanie populacji gatunków grzybów oraz identyfikację gatunków zagrożonych wyginięciem, co znajduje odzwierciedlenie w badaniach nad automatyczną klasyfikacją gatunków roślin [17]. Podobne systemy mogą być również wykorzystane w medycynie do analizy obrazów diagnostycznych, gdzie wysoka precyzja i interpretowalność wyników są kluczowe [33].

Praca została podzielona na pięć rozdziałów. W rozdziale drugim opisano przegląd literatury, który przedstawia najważniejsze badania dotyczące metod ekstrakcji cech wizualnych oraz prototypowania, prezentuje szczegóły dotyczące zastosowanych metod, w tym architektury autokodera, sieci ResNet50, algorytmów klasyfikacyjnych kNN i centroidów oraz omawia metody ewaluacji i metryki użyte do oceny skuteczności zastosowanych modeli i algorytmów. Wyniki uzyskane w eksperymentach omówiono szczegółowo w rozdziale czwartym. Piąty rozdział zawiera podsumowanie wyników oraz propozycje dalszych kierunków badań.

Rozdział 2

Powiązane prace

Początki sieci neuronowych sięgają lat 40-tych dwudziestego wieku, kiedy to Warren McCulloch i Walter Pitts zaproponowali pierwszy matematyczny model neuronu [27]. Model ten zakładał, że neuron działa jak funkcja progowa, przekształcając sygnały wejściowe w pojedynczy sygnał wyjściowy. Z czasem rozwój teorii sieci neuronowych nabierał tempa, a kolejnym ważnym krokiem było opracowanie perceptronu przez Franka Rosenblatta w 1958 roku. Perceptron był w stanie klasyfikować dane liniowo separowalne, co otworzyło nowe możliwości w uczeniu maszynowym [6].

W latach 80-tych dwudziestego wieku nastąpiła era głębszych sieci neuronowych, kiedy Geoffrey Hinton i jego współpracownicy wprowadzili metodę propagacji wstecznej (backpropagation). Metoda ta pozwalała na efektywne trenowanie wielowarstwowych sieci neuronowych, rozwiązujeąc problem uczenia się złożonych zależności [25]. Kolejne dekady przyniosły rozwój splotowych sieci neuronowych (CNN), które stały się przełomem w zadaniach związanych z przetwarzaniem obrazów. Architektura LeNet, zaproponowana przez Yanna LeCuna w 1998 roku, była jedną z pierwszych CNN stosowanych do rozpoznawania cyfr na czekach bankowych [7].

W 2012 roku AlexNet, zaprojektowany przez Alexa Krizhevsky'ego, zdobył ogromne uznanie, wygrywając konkurs ImageNet dzięki głębokiej architekturze sieci i zastosowaniu GPU do przyspieszenia obliczeń. Wkrótce pojawiły się inne zaawansowane architektury, takie jak VGG, GoogLeNet i ResNet, które wprowadzały nowe mechanizmy, takie jak bloki rezydualne, pozwalające na efektywne trenowanie bardzo głębokich sieci [11]. Obecnie rozwój sieci neuronowych obejmuje modele hybrydowe, takie jak transformery wizyjne i sieci neuronowe oparte na mechanizmach uwagi, które znajdują zastosowanie w zadaniach wizji komputerowej [9].

Przed rozwojem głębokich sieci neuronowych, klasyczne metody widzenia komputerowego opierały się na ręcznej ekstrakcji cech i zastosowaniu algorytmów klasyfikacyjnych. Techniki takie jak wykrywanie krawędzi za pomocą algorytmu Canny'ego czy metoda Harris Corner Detection były szeroko stosowane w latach 80. i 90. XX wieku [27]. Wykorzystanie deskryptorów cech opracowanych na przełomie dwudziestego wieku, takich jak SIFT (Scale-Invariant Feature Transform) i SURF (Speeded-Up Robust Features), pozwalało na efektywne rozpoznawanie obiektów w obrazach o różnej skali i rotacji [15].

Klasyczne metody, takie jak PCA (Principal Component Analysis) i LDA (Linear Discriminant Analysis), były używane do redukcji wymiarowości danych, co pozwalało na skuteczniejsze klasyfikowanie obrazów przy użyciu algorytmów takich jak metoda k Najbliższych Sąsiadów (kNN) czy Maszyna Wektorów Nośnych (SVM) [7]. Istotnym krokiem było również zastosowanie algorytmów segmentacji, takich jak metoda k-średnich (ang. k-means) czy algorytm działu wodnego (ang. watershed), które umożliwiały podział obrazów na istotne regiony [25].

Mimo że klasyczne metody zostały w dużej mierze zastąpione przez sieci neuronowe, nadal znajdują one zastosowanie w zadaniach, gdzie dostępne są ograniczone zasoby obliczeniowe lub niewielka ilość danych. Przykładem są systemy wbudowane czy aplikacje przemysłowe, w których priorytetem jest szybkość działania i prostota implementacji [6].

Obecnie obserwuje się także trend łączenia klasycznych metod z nowoczesnymi podejściami, takimi jak głębokie uczenie (ang. deep learning). Na przykład klasyczne deskryptory cech mogą być stosowane w połączeniu z głębokimi sieciami do ekstrakcji cech wyższego rzędu. Tego rodzaju podejście hybrydowe są szczególnie przydatne w scenariuszach z ograniczoną ilością danych lub w zadaniach zero-shot learning [18, 33].

Przykładem takiego podejścia może być prototypowanie. W latach 70. XX wieku Eleanor Rosch wprowadziła teorię prototypów, sugerując, że niektórzy członkowie kategorii są bardziej centralni niż inni, co wpływa na procesy poznawcze związane z kategoryzacją [22]. W uczeniu maszynowym koncepcja ta znalazła zastosowanie w metodach klasyfikacji, takich jak algorytm k-NN (k najbliższych sąsiadów), gdzie klasyfikacja nowego punktu danych opiera się na podobieństwie do prototypowych przedstawicieli klas [24]. Wprowadzenie metod uczenia z niewielką liczbą przykładów (few-shot learning) oraz uczenia z zerową liczbą przykładów (zero-shot learning) dodatkowo podkreśliło znaczenie prototypów w efektywnym modelowaniu i klasyfikacji danych przy ograniczonej liczbie próbek treningowych [24].

Rozpoznawanie obrazów, będące przedmiotem niniejszej pracy, wymaga skutecznych metod ekstrakcji cech wizualnych oraz klasyfikacji. Jednymi z głównych wyzwań w tej dziedzinie, omówionymi w pierwszym rozdziale, są ograniczona liczba danych treningowych, różnorodność danych oraz złożoność obliczeniowa. W odpowiedzi na te wyzwania, w niniejszej pracy skoncentrowano się na zastosowaniu głębokich sieci neuronowych w połączeniu z klasycznymi metodami klasyfikacji, aby ocenić ich skuteczność w zadaniu klasyfikacji obrazów grzybów. Do realizacji tego celu zaimplementowano i przetestowano dwie różne architektury sieci neuronowych: autorską implementację autokodera oraz wstępnie wytrenowaną sieć splotową ResNet50. Następnie przeanalizowano ich zdolność do generowania reprezentacji cech, które mogłyby być efektywnie wykorzystane w klasyfikacji przy użyciu algorytmu k-Nearest Neighbors (kNN), centroidów oraz t-SNE.

W kolejnych sekcjach szczegółowo opisano każdą z zastosowanych metod oraz kroków przetwarzania danych.

2.0.1 Zastosowane sieci neuronowe

Jak już zostało wspomniane, do ekstrakcji cech wizualnych wykorzystanych do rozpoznawania gryzów zaadaptowane zostały dwa algorytmy: autokoder wariacyjny oraz sieć ResNet50.

Autokoder wariacyjny

Autokoder jest rodzajem sieci neuronowej stosowanej w uczeniu nienadzorowanym. Jego głównym celem jest nauczenie się kompaktowej reprezentacji danych wejściowych w tzw. przestrzeni utajonej (ang. latent space), a następnie zrekonstruowanie oryginalnych danych na podstawie tej reprezentacji. Jest to szczególnie przydatne w redukcji wymiarowości danych oraz w zadaniach związanych z ekstrakcją cech [8, 33].

Autokoder składa się z dwóch głównych komponentów: kodera i dekodera. Schemat autokodera przedstawiono na rysunku 1. Matematycznie działanie autokodera można opisać jako:

$$z = f_{\text{encoder}}(x; \theta_e), \quad (1)$$

gdzie z to reprezentacja utajona, x to dane wejściowe, a θ_e to parametry kodera.

$$\hat{x} = f_{\text{decoder}}(z; \theta_d), \quad (2)$$

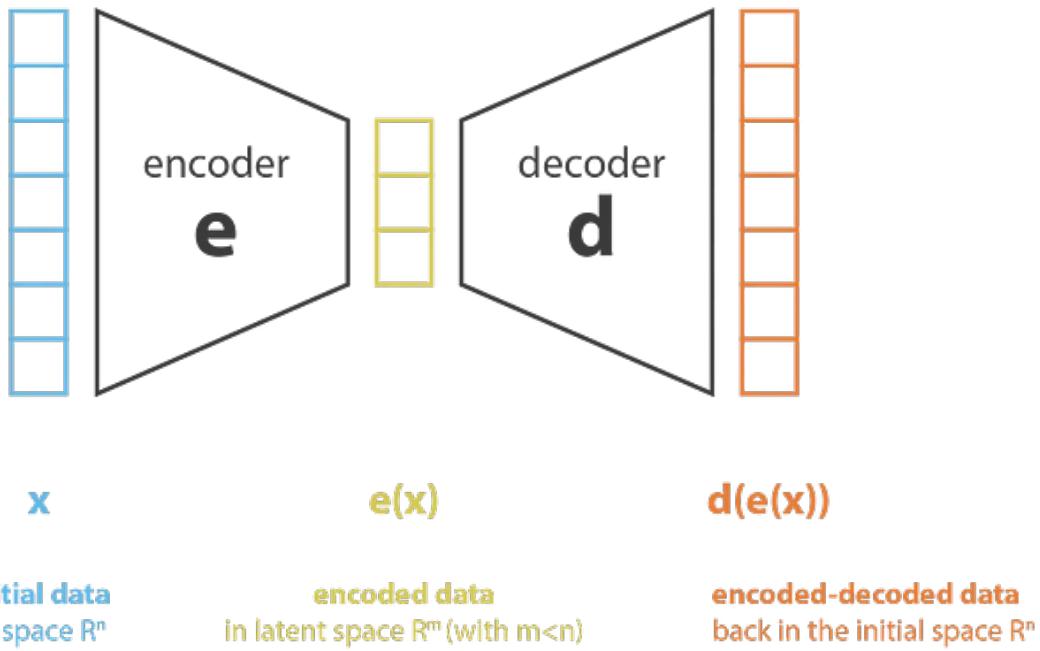
gdzie \hat{x} to zrekonstruowane dane, a θ_d to parametry dekodera.

Klasyczne autokodery, mimo swojej skuteczności, mają ograniczoną zdolność do generowania nowych próbek, co oznacza, że ich przestrzeń utajona może nie być dobrze uogólniona na nowe dane. Aby przezwyciężyć te ograniczenia, wprowadzono autokodery wariacyjne (ang. Variational Autoencoders, VAE). VAE, opracowany przez Kingmę i Wellinga w 2014 roku [14], rozszerza klasyczny model, dodając probabilistyczne podejście do reprezentacji przestrzeni utajonej. Głównym celem VAE jest nauczenie się nie tylko kompresji danych, ale również modelowania rozkładu prawdopodobieństwa danych w przestrzeni cech.

W przeciwieństwie do klasycznego autokodera, który mapuje dane wejściowe na pojedynczy punkt w przestrzeni utajonej, VAE przyjmuje podejście probabilistyczne, reprezentując dane jako rozkład prawdopodobieństwa w przestrzeni utajonej. To pozwala na generowanie nowych próbek danych poprzez próbkowanie z tego rozkładu, co czyni VAE niezwykle przydatnym w zadaniach generatywnych, takich jak synteza obrazów, uzupełnianie brakujących danych czy eksploracja przestrzeni cech.

W przypadku autokodera wariacyjnego (VAE), reprezentacja utajona z jest modelowana jako próbka pochodząca z rozkładu probabilistycznego $q_\phi(z|x)$, który aproksymuje prawdziwy rozkład $p(z|x)$. Przyjmuje się, że $q_\phi(z|x)$ jest rozkładem normalnym z parametrami wyznaczanymi na podstawie danych wejściowych x :

$$q_\phi(z|x) = \mathcal{N}(z; \mu(x), \sigma^2(x)), \quad (3)$$



Rysunek 1. Schemat budowy autokodera [21].

gdzie $\mu(x)$ to średnia, a $\sigma(x)$ to odchylenie standardowe wyznaczane przez kodera. Parametry $\mu(x)$ i $\sigma(x)$ są uzyskiwane jako wyniki dwóch niezależnych warstw w sieci kodera. Dzięki temu $q_\phi(z|x)$ reprezentuje rozkład, z którego próbujemy z , zamiast wyznaczać je deterministycznie.

Drugim kluczowym elementem w VAE jest regularizacja przestrzeni utajonej za pomocą dywergencji Kullbacka-Leiblera (D_{KL}). Jest to miara różnicy pomiędzy rozkładem $q_\phi(z|x)$ (aproksymacja) a rozkładem priorytetowym $p(z)$ (najczęściej przyjmowanym jako rozkład normalny $\mathcal{N}(0, I)$):

$$D_{KL}(q_\phi(z|x) \| p(z)) = \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p(z)} dz. \quad (4)$$

Celem tej regularizacji jest zapewnienie, że rozkład $q_\phi(z|x)$ nie odbiega nadmiernie od priorytetowego $p(z)$, co skutkuje lepiej ustrukturyzowaną przestrzenią utajoną. Dzięki temu próbkowanie z staje się bardziej stabilne i interpretable.

Funkcja kosztu dla VAE łączy dwa komponenty:

$$L = L_{\text{reconstruction}} + D_{KL}. \quad (5)$$

- Komponent rekonstrukcji ($L_{\text{reconstruction}}$) mierzy, jak dobrze dane wejściowe x zostały zrekonstruowane przez model. Zazwyczaj jest to obliczane za pomocą funkcji Binary Cross-Entropy

(BCE) lub Mean Squared Error (MSE):

$$L_{\text{reconstruction}} = \|x - \hat{x}\|^2 \quad \text{lub} \quad - \sum_i x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i),$$

gdzie \hat{x} to dane wyjściowe generowane przez dekoder.

- Komponent regularizacji (D_{KL}) zapewnia, że przestrzeń utajona z jest ustrukturyzowana i bliska priorytetowemu rozkładowi $p(z)$.

Podsumowując, VAE minimalizuje funkcję kosztu, dążąc do jednoczesnej maksymalizacji jakości rekonstrukcji i strukturalizacji przestrzeni utajonej. Dzięki temu jest w stanie generować realistyczne dane nawet przy próbkowaniu z z priorytetowego $p(z)$.

W niniejszych badaniach zastosowano autokoder wariacyjny, który generował reprezentacje utajone obrazów grzybów. Autokoder wariacyjny ze względu na swoją zdolność do modelowania rozkładów prawdopodobieństwa w przestrzeni utajonej, co pozwala na generowanie bardziej uogólnionych reprezentacji cech, zwiększa ich przydatność w zadaniach klasyfikacyjnych i wizualizacji. Reprezentacje te były następnie wykorzystywane w algorytmach klasyfikacyjnych. Dzięki kombinacji funkcji rekonstrukcji i regularyzacji, autokoder umożliwił efektywną ekstrakcję cech wizualnych. Główne różnice pomiędzy klasycznym autokoderem a jego wariacyjną wersją to:

- Probabilistyczna przestrzeń utajona: Zamiast pojedynczego punktu w przestrzeni utajonej, VAE modeluje dane jako próbki z rozkładu prawdopodobieństwa, co pozwala na bardziej uporządkowaną reprezentację danych [29];
- Regularizacja przestrzeni utajonej: Dzięki zastosowaniu Dywergencji Kullbacka-Leiblera (KLD), przestrzeń utajona jest zmuszona do przybliżania rozkładu normalnego, co zwiększa jej interpretowalność oraz zdolności generatywne modelu [32].

W niniejszych badaniach zastosowano funkcję straty składającą się z dwóch komponentów: Binarnej Entropii Krzyżowej (BCE) oraz Dywergencji Kullbacka-Leiblera (KLD).

Binarna Entropia Krzyżowa (BCE) mierzy różnicę między oryginalnymi danymi wejściowymi a ich rekonstrukcją, co pozwala na dokładne odwzorowanie szczegółów danych wizualnych [3]. Jest to kluczowy element funkcji straty w procesie rekonstrukcji [34].

$$\text{BCE} = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)], \quad (6)$$

gdzie y_i to wartość rzeczywista, \hat{y}_i to wartość przewidziana, a N to liczba próbek.

Dywergencji Kullbacka-Leiblera (KLD) pełni rolę regulatora, który wymusza na przestrzeni utajonej przybliżenie rozkładu normalnego [23]. W literaturze wskazuje się, że zastosowanie KLD pozwala na poprawę generatywności i uporządkowania przestrzeni utajonej, co jest kluczowe dla wielu zastosowań, takich jak klasyfikacja czy generowanie nowych danych [17, 31].

$$\text{KLD} = -\frac{1}{2} \sum_{j=1}^d \left(1 + \log(\sigma_j^2) - \mu_j^2 - \sigma_j^2 \right), \quad (7)$$

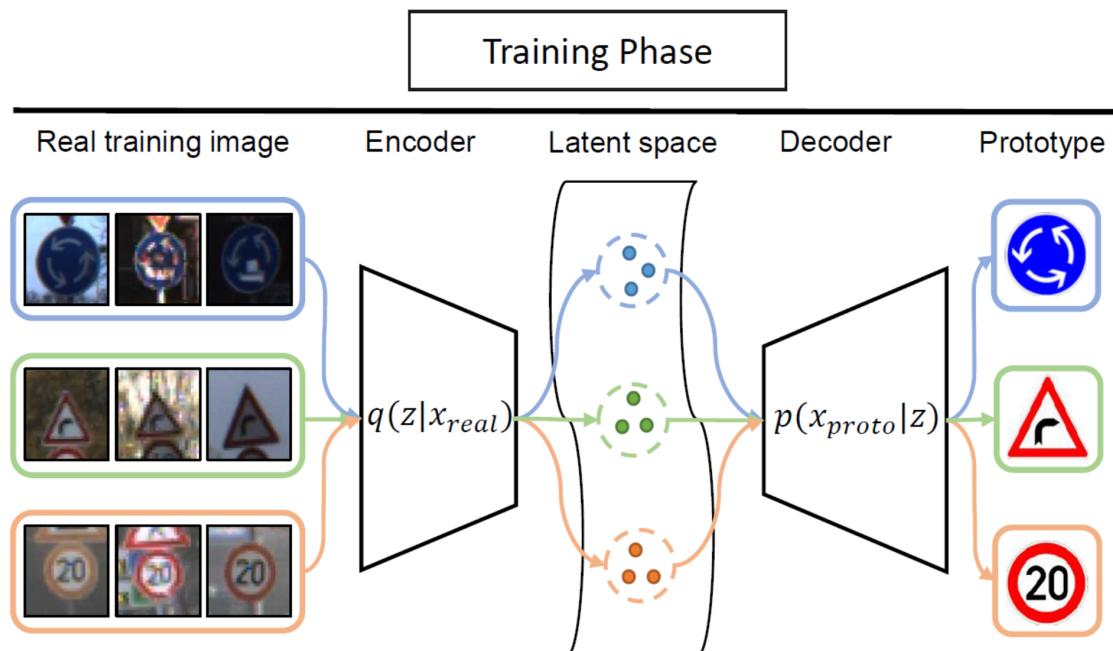
gdzie μ_j i σ_j to odpowiednio średnia i odchylenie standardowe dla wymiaru j w przestrzeni utajonej, a d to liczba wymiarów.

Zastosowanie połączenia funkcji BCE i KLD umożliwiło uzyskanie reprezentacji utajonych, które były zarówno kompaktowe, jak i informatywne. Dzięki temu autokoder wykorzystany w badaniach osiągnął wysoką jakość rekonstrukcji przy jednoczesnym zachowaniu strukturalnej spójności przestrzeni utajonej. Takie podejście było zgodne z wynikami innych badań, które wskazują na zalety probabilistycznej reprezentacji w zadaniach ekstrakcji cech i generowania danych [32, 35].

Podczas treningu wykorzystano następujące parametry dla szkolenia sieci autokodera:

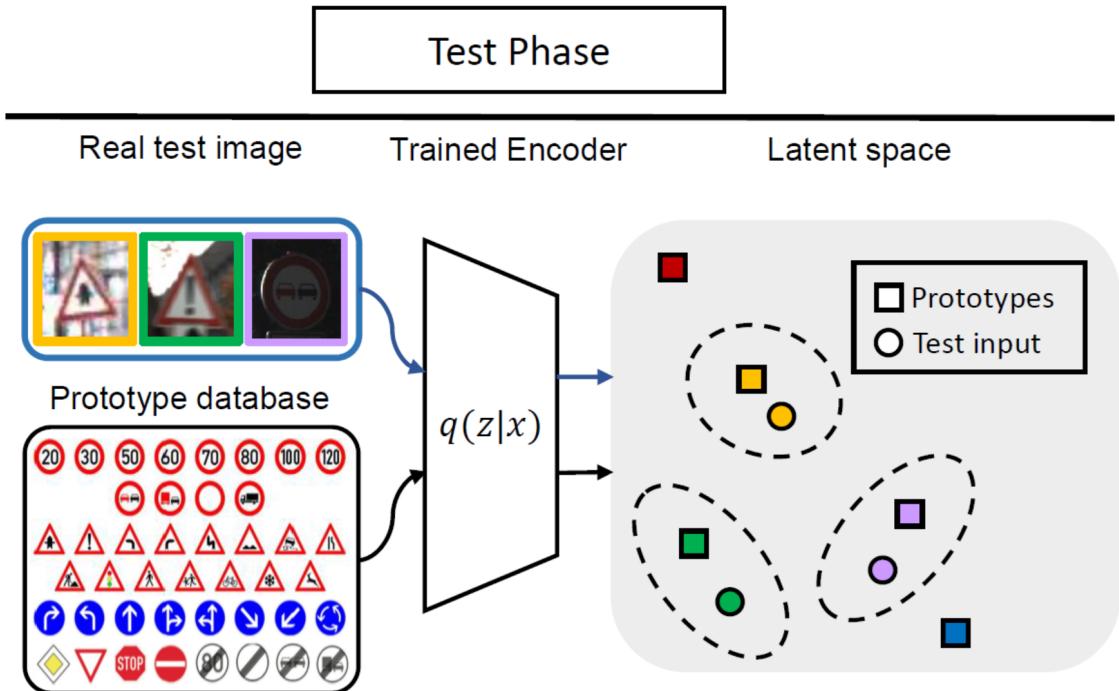
- wymiar wyjściowy cech (2048): wymiar przestrzeni ukrytej został ustalony na 2048, aby zapewnić wystarczającą ilość informacji o obrazach przy jednoczesnej redukcji wymiarowości danych [33],
- współczynnika uczenia (1×10^{-6}): niska wartość współczynnika uczenia zapewnia stabilność treningu, szczególnie przy głębokich sieciach autokodera [18],
- rozmiar partii (ang. batch size) ustalono na 32 jako kompromis pomiędzy czasem obliczeń a jakością gradientów w procesie optymalizacji [8],
- liczba Epok (2000): duża liczba epok umożliwiła pełną zbieżność modelu i uzyskanie stabilnych wyników rekonstrukcji obrazów.

Schematy fazy treningowej oraz testowej autokodera przedstawiono na rysunkach 2 oraz 3.



Rysunek 2. Schemat fazy treningowej autokodera wariacyjnego [13]

Jako obrazy wzorcowe używane podczas trenowania autokodera wykorzystano zdjęcia najbliższe prototypowi danej klasy. Prototypy te zostały wyznaczone przez uśrednienie wyników uzyskanych



Rysunek 3. Schemat fazy testowej autokodera wariacyjnego [13]

przy użyciu sieci ResNet50, a następnie znalezienie obrazu najbliższej odpowiadającego utworzonemu prototypowi w przestrzeni cech.

ResNet50

Sieci splotowe (ang. Convolutional Neural Networks, CNN) są jedną z najważniejszych klas modeli głębokiego uczenia, szeroko stosowaną w zadaniach widzenia komputerowego, takich jak klasyfikacja obrazów, segmentacja czy detekcja obiektów [9]. Podstawowym elementem CNN jest warstwa splotowa, która umożliwia modelowi wyodrębnienie cech obrazu, takich jak krawędzie, tekstury czy wzory. Proces ten jest stosowany iteracyjnie na różnych poziomach hierarchii sieci, co pozwala na modelowanie coraz bardziej złożonych cech wizualnych. Architektury CNN, takie jak AlexNet, VGGNet oraz Inception, stanowiły fundamenty współczesnych rozwiązań, jednak ich rosnąca głębokość wiązała się z problemem zanikającego gradientu, który utrudniał skuteczne trenowanie bardzo głębokich modeli [12].

ResNet (Residual Network) [11] wprowadził istotną innowację w postaci bloków rezydualnych (ang. residual blocks), które rozwiązują problem zanikającego gradientu. Bloki te zawierają połączenia rezydualne, które umożliwiają przekazywanie informacji z wcześniejszych warstw do warstw późniejszych, ułatwiając propagację gradientu w trakcie uczenia. Formalnie, dla wejścia \mathbf{x} , blok rezydualny można opisać jako:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}, \quad (8)$$

gdzie $\mathcal{F}(\mathbf{x}, \{W_i\})$ to funkcja reprezentująca operacje w bloku (np. sploty, normalizacje wsadowe i aktywacje ReLU), a \mathbf{x} to wejście. Połączenie rezydualne dodaje \mathbf{x} do wyjścia bloku, co poprawia przepływ gradientów.

ResNet50 jest trenowany przy użyciu entropii krzyżowej (*Cross-Entropy Loss*) jako funkcji straty, którą dla C klas można zapisać jako:

$$\mathcal{L}_{CE} = - \sum_{i=1}^C y_i \log(\hat{y}_i), \quad (9)$$

gdzie y_i to wartość rzeczywista (1 dla klasy poprawnej, 0 dla innych), a \hat{y}_i to prawdopodobieństwo przewidziane przez model dla klasy i .

Główną zaletą bloków rezydualnych jest możliwość trenowania bardzo głębokich modeli bez degradacji wydajności. Dzięki temu ResNet50 umożliwia skuteczne modelowanie złożonych wzorców wizualnych, co czyni go bardziej precyzyjnym niż płytse wersje, takie jak ResNet18 czy ResNet34 [26].

W kontekście niniejszych badań ResNet50 został zaadaptowany jako ekstraktor cech wizualnych poprzez usunięcie końcowej warstwy klasyfikacyjnej (*fully connected layer*). Tak przygotowany model umożliwił generowanie wektorów cech reprezentujących obrazy grzybów w przestrzeni cech:

$$\mathbf{z} = \mathcal{F}_{\text{ResNet50}}(\mathbf{x}), \quad (10)$$

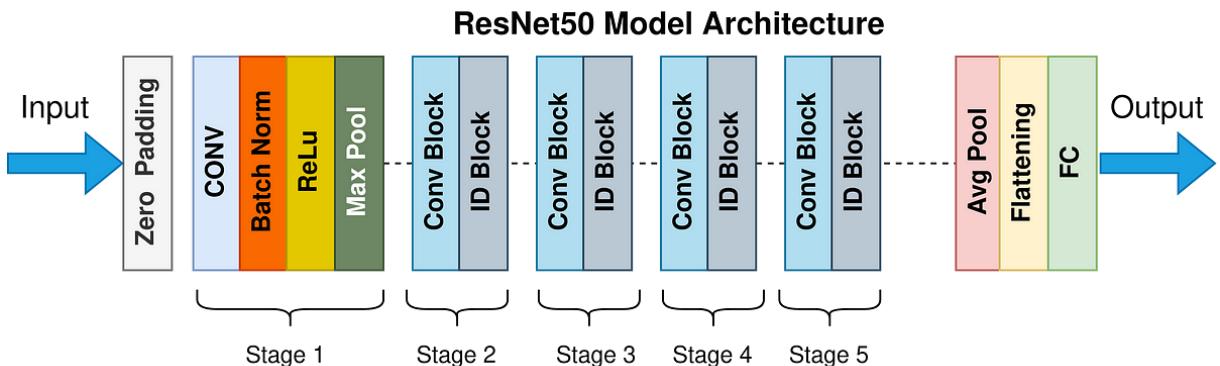
gdzie \mathbf{x} to wejściowy obraz, a \mathbf{z} to wektor cech o wymiarze 2048. Wektorów tych użyto następnie w algorytmie k-Nearest Neighbors (kNN) oraz metodzie centroidów do klasyfikacji obrazów.

Podczas treningu wykorzystano następujące parametry:

- wymiar wyjściowy cech (2048): odpowiada wyjściu ostatniego bloku rezydualnego sieci [33],
- współczynnik uczenia (1×10^{-6}): zapewnia stabilność w procesie dostosowywania wcześniej wytrenowanych wag sieci [18],
- rozmiar partii (32): optymalny kompromis między czasem obliczeń a wykorzystaniem pamięci GPU [8],
- liczba epok (2000): umożliwiła stabilne dostosowanie wstępnie wyuczonych wag do zbioru danych grzybów.

2.0.2 Algorytm k-Najbliższych Sąsiadów (kNN)

Algorytm kNN jest jednym z najprostszych, a zarazem najczęściej stosowanych algorytmów klasyfikacji w uczeniu maszynowym. Bazuje na zasadzie sąsiedztwa, gdzie klasyfikacja nowych danych odbywa się poprzez analizę najbliższych punktów w przestrzeni cech. Ze względu na swoją intuicyjność i skuteczność, kNN znajduje szerokie zastosowanie w różnorodnych problemach, takich jak klasyfikacja obrazów, wykrywanie anomalii czy analiza genetyczna [4, 8, 10, 34].



Rysunek 4. Schemat działania sieci ResNet50 [20]

Algorytm kNN klasyfikuje dane na podstawie większościowej klasy ich najbliższych sąsiadów w przestrzeni cech. Dla nowego punktu testowego x , kNN oblicza odległości do wszystkich punktów w zbiorze treningowym i wybiera k najbliższych sąsiadów. Klasa punktu x jest określana przez najczęściej występującą klasę wśród tych sąsiadów. Proces ten można formalnie zapisać jako:

$$\text{Klasa}(x) = \arg \max_y \sum_{i=1}^k \mathbb{F}(y_i = y), \quad (11)$$

gdzie y_i to klasa i -tego sąsiada, y to potencjalna klasa, a \mathbb{F} to funkcja wskaźnikowa, która przyjmuje wartość 1, gdy warunek jest spełniony, w przeciwnym razie 0.

W niniejszej pracy zastosowano odległość euklidesową jako metrykę do obliczania odległości między punktami, co jest standardowym wyborem w przypadku analizy danych wysokowymiarowych:

$$d(x, y) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}, \quad (12)$$

gdzie x i y to punkty w przestrzeni cech, a d to wymiar tej przestrzeni.

Algorytm kNN charakteryzuje się wieloma zaletami:

- prostota implementacji: brak potrzeby trenowania modelu czyni kNN łatwym do wdrożenia,
- skalowalność: algorytm może być stosowany do różnych problemów klasyfikacyjnych i regresyjnych,
- efektywność przy małych zbiorach danych: brak potrzeby optymalizacji parametrów czyni go idealnym dla niewielkich zbiorów danych.

Jednak kNN ma również pewne ograniczenia:

- wysoka złożoność obliczeniowa dla dużych zbiorów danych, wynikająca z konieczności obliczania odległości od każdego punktu ze zbioru treningowego,

- wrażliwość na szum w danych i nierównomierne rozmieszczenie próbek w przestrzeni cech,
- wymaga przechowywania całego zbioru treningowego, co zwiększa wymagania pamięciowe [10].

W ramach niniejszej pracy kNN został zastosowany do klasyfikacji obrazów grzybów, wykorzystując cechy wyekstrahowane za pomocą autokodera i ResNet50. Eksperymenty obejmowały różne wartości parametru k , aby ocenić wpływ liczby sąsiadów na skuteczność klasyfikacji. Proces klasyfikacji składał się z następujących kroków:

- ekstrakcja cech wizualnych za pomocą autokodera oraz ResNet50,
- obliczenie odległości euklidesowej między punktami testowymi a wszystkimi próbками w zbiorze treningowym,
- określenie klasy punktu testowego na podstawie większościowej klasy wśród k najbliższych sąsiadów.

Zastosowanie kNN w niniejszych badaniach umożliwiło ocenę zdolności generalizacyjnych modeli ekstrakcji cech, takich jak autokoder i ResNet50. Algorytm ten pozwolił na:

- przeprowadzenie bezpośredniego porównania różnych metod ekstrakcji cech,
- zbadanie wpływu różnych parametrów klasyfikacji na skuteczność modeli,
- identyfikację potencjalnych problemów, takich jak nachodzenie się klas w przestrzeni cech, co mogło wpływać na skuteczność klasyfikacji [4].

Wyniki uzyskane z klasyfikacji za pomocą kNN dostarczyły cennych informacji na temat efektywności badanych modeli oraz ich przydatności w zadaniach klasyfikacyjnych opartych na cechach wizualnych. Analiza ta pozwoliła również na lepsze zrozumienie rozdzielności klas w przestrzeni cech, co stanowiło istotny element oceny porównawczej między autokoderem a siecią ResNet50.

2.0.3 Centroidy w analizie danych

Centroid to punkt geometryczny reprezentujący środek ciężkości zbioru punktów w przestrzeni. W kontekście analizy danych centroidy odgrywają kluczową rolę jako reprezentatywne punkty klas w przestrzeniach wielowymiarowych. Ich zastosowanie jest szczególnie popularne w algorytmach klasteryzacji, takich jak k-średnich, gdzie centroid stanowi środek każdego klastra [2]. W niniejszej pracy centroidy są wykorzystywane do oceny strukturalnej spójności klas w przestrzeni cech oraz do uproszczonej klasyfikacji.

Centroidy są obliczane jako średnia arytmetyczna współrzędnych punktów w przestrzeni. Dla zbioru N punktów x_1, x_2, \dots, x_N w przestrzeni d -wymiarowej, centroid \bar{x} wyznacza się według wzoru:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i, \quad (13)$$

gdzie x_i to wektory punktów, a N to liczba punktów w zbiorze. Dzięki temu centroid odzwierciedla średnią pozycję punktów w przestrzeni, co pozwala na uproszczenie reprezentacji całej klasy.

Zastosowanie centroidów w analizie danych ma wiele zalet:

- upraszczają analizę danych poprzez reprezentację całych klas jednym punktem,
- umożliwiają ocenę strukturalnej spójności klas w przestrzeni cech,
- redukują złożoność pamięciową procesu klasyfikacji, ponieważ nie wymagają przechowywania wszystkich próbek.

Jednakże centroidy mogą być podatne na zakłócenia wynikające z obecności szumów w danych lub nierównomiernego rozmieszczenia punktów w obrębie klasy [1].

W niniejszej pracy centroidy zostały zastosowane jako uproszczona metoda klasyfikacji obrazów grzybów oraz do analizy struktury przestrzeni cech wygenerowanych przez autokoder i ResNet50. Proces ten obejmował następujące kroki:

- ekstrakcję cech wizualnych za pomocą autokodera i ResNet50,
- obliczenie centroidów dla każdej klasy w zbiorze treningowym, uwzględniając wszystkie próbki danej klasy,
- klasyfikację obrazów testowych na podstawie najbliższego centroidu, wykorzystując metrykę odległości euklidesowej.

Zastosowanie centroidów w analizie danych i klasyfikacji obrazów ma kilka kluczowych zalet:

- umożliwia analizę globalnej struktury przestrzeni cech oraz ocenę rozdzielczości klas,
- upraszcza proces klasyfikacji, eliminując potrzebę przechowywania wszystkich próbek treningowych,
- zwiększa odporność na szum danych poprzez uśrednianie reprezentacji punktów [2].

Wyniki uzyskane z klasyfikacji opartej na centroidach pozwoliły na porównanie ich skuteczności z algorytmem kNN. Analiza ta dostarczyła dodatkowych informacji o zdolnościach modeli do uogólniania wiedzy oraz ich potencjału w uproszczonych scenariuszach klasyfikacyjnych.

2.0.4 Algorytm t-SNE

Algorytm t-SNE (*t-Distributed Stochastic Neighbor Embedding*) to technika redukcji wymiarowości, która umożliwia wizualizację danych wysokowymiarowych w przestrzeni dwuwymiarowej lub trójwymiarowej [19]. t-SNE jest szczególnie użyteczny w analizie struktur danych, takich jak klastry, co czyni go odpowiednim narzędziem do oceny rozdzielczości cech generowanych przez modele ekstrakcji cech, takie jak autokoder i ResNet50.

t-SNE opiera się na dwóch głównych etapach:

- modelowaniu zależności lokalnych w przestrzeni wysokowymiarowej: Algorytm oblicza prawdopodobieństwo sąsiedztwa dla każdej pary punktów, gdzie prawdopodobieństwo p_{ij} jest proporcjonalne do odległości euklidesowej między punktami x_i i x_j :

$$p_{ij} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k \neq l} \exp(-\|x_k - x_l\|^2/2\sigma_k^2)}, \quad (14)$$

gdzie σ_i to parametr określający lokalną skalę sąsiedztwa dla punktu x_i . Punkty bliskie w przestrzeni wysokowymiarowej mają wysokie prawdopodobieństwo sąsiedztwa.

- rekonstrukcji lokalnych zależności w przestrzeni niskowymiarowej: Algorytm t-SNE stara się znaleźć układ punktów w przestrzeni niskowymiarowej, który jak najlepiej odzwierciedla prawdopodobieństwa sąsiedztwa p_{ij} . W tym celu oblicza podobieństwo q_{ij} w przestrzeni niskowymiarowej według wzoru:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}, \quad (15)$$

gdzie y_i i y_j to współrzędne punktów w przestrzeni niskowymiarowej.

Aby znaleźć optymalny układ punktów w przestrzeni niskowymiarowej, t-SNE minimalizuje rozbieżność Kullbacka-Leiblera (KLD) między rozkładami p_{ij} i q_{ij} :

$$C = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}. \quad (16)$$

Minimalizacja funkcji kosztu C jest realizowana za pomocą algorytmów gradientowych, co pozwala na znalezienie układu punktów o wysokiej zgodności lokalnych zależności między punktami w obu przestrzeniach.

Algorytm t-SNE jest szeroko stosowany ze względu na swoje zalety:

- skuteczność w wizualizacji danych wysokowymiarowych,
- możliwość identyfikacji lokalnych struktur danych, takich jak klastry,
- dobra intuicyjność i interpretowalność wyników.

Mimo tych zalet t-SNE ma również pewne ograniczenia, takie jak wysoka złożoność obliczeniowa oraz wrażliwość na dobór hiperparametrów, takich jak *perplexity* i szybkość uczenia (*learning rate*) [16].

W niniejszej pracy t-SNE został zastosowany w celu wizualizacji przestrzeni cech wygenerowanych przez autokoder oraz ResNet50. Algorytm umożliwił analizę strukturalnej spójności przestrzeni utajonej oraz ocenę separowalności klas w tej przestrzeni. Przeprowadzono następujące eksperymenty:

- wizualizacja t-SNE dla zbioru treningowego, aby zbadać, jak poszczególne klasy są reprezentowane w przestrzeni cech,

-
- wizualizacja t-SNE dla pełnego zbioru danych, w tym obrazów niewidzianych podczas treningu, co pozwoliło na ocenę zdolności modeli do generalizacji,
 - wizualizacja prototypów klas dla zbioru treningowego, wyznaczonych jako średnie wartości reprezentacji cech, w celu analizy rozkładu danych treningowych oraz oceny ich spójności w przestrzeni utajonej,
 - wizualizacja prototypów klas dla pełnego zbioru danych, wyznaczonych jako średnie wartości reprezentacji cech, mająca na celu ocenę globalnej struktury przestrzeni oraz jej zdolności do odzwierciedlenia różnorodności wszystkich danych.

Wyniki wizualizacji t-SNE dostarczyły cennych informacji na temat zdolności modeli do generowania reprezentacji cech. Ułatwiły również zidentyfikowanie problemów, takich jak nachodzenie się reprezentacji różnych klas w przestrzeni cech, co może wpływać na skuteczność klasyfikacji. Analiza ta stanowiła istotny element oceny porównawczej pomiędzy autokoderem a siecią ResNet50, wspierając wnioski dotyczące ich efektywności w zadaniu rozpoznawania obrazów grzybów.

Rozdział 3

Eksperymenty

W celu oceny skuteczności wybranych metod ekstrakcji cech oraz klasyfikacji przeprowadzono szereg eksperymentów, które obejmowały trening modeli, ewaluację ich wydajności oraz analizę zdolności generalizacyjnych. Poniżej przedstawiono szczegóły przeprowadzonych eksperymentów.

Przed przystąpieniem do treningu przeprowadzono następujące operacje wstępnego przetwarzania:

- skalowanie obrazów do rozmiaru 256×256 pikseli,
- normalizację wartości pikseli do zakresu $[0, 1]$,
- losowe przetasowanie próbek w zestawach treningowych i walidacyjnych,
- zastosowanie losowych transformacji do zbioru treningowego:
 - losowe poziome odbicie,
 - losowa rotacja w zakresie ± 10 stopni.

3.1 Specyfikacja Środowiska Eksperimentalnego i Oprogramowania

Eksperymenty zostały przeprowadzone na serwerze z procesorem AMD EPYC 7F72 24-Core (48 wątków) o maksymalnej częstotliwości 3,2 GHz oraz akceleracją GPU realizowaną przez karty NVIDIA RTX 5000 Ada Generation. Środowisko obliczeniowe obejmowało system operacyjny Linux oraz wykorzystanie frameworka PyTorch w wersji zoptymalizowanej pod kątem obliczeń równoległych na GPU.

3.2 Narzędzia i Biblioteki

Do realizacji badań nad rozpoznawaniem obrazów grzybów wykorzystano nowoczesne narzędzia i biblioteki, które umożliwiły implementację, trenowanie modeli oraz analizę wyników. Poniżej przedstawiono szczegóły dotyczące użytych technologii:

Python jest jednym z najpopularniejszych języków programowania w dziedzinie uczenia maszynowego i wizji komputerowej. Jego wszechstronność oraz dostępność szerokiego ekosystemu bibliotek

sprawiają, że jest idealnym narzędziem do realizacji złożonych projektów badawczych. W ramach niniejszej pracy Python został wykorzystany jako główny język programowania do implementacji algorytmów i analizy danych.

PyTorch [18] to otwartoźródłowa biblioteka do tworzenia i trenowania sieci neuronowych. Jej elastyczność, wsparcie dla dynamicznych obliczeń oraz intuicyjny interfejs czynią ją jedną z najczęściej używanych platform w badaniach naukowych. W pracy wykorzystano PyTorch do:

- Implementacji i trenowania autokodera,
- Modyfikacji i zastosowania ResNet50 jako ekstraktora cech.

scikit-learn (sklearn) [34] to biblioteka oferująca szeroki zakres algorytmów uczenia maszynowego. W niniejszej pracy została wykorzystana do implementacji algorytmu k-Nearest Neighbors (kNN) oraz analizy wyników klasyfikacji. Jej prostota i wydajność pozwoliły na łatwe dostosowanie parametrów klasyfikatora, takich jak liczba sąsiadów czy metryka odległości.

NumPy [8] i Matplotlib [33] to podstawowe biblioteki do obliczeń numerycznych oraz wizualizacji danych w Pythonie. NumPy został wykorzystany do przetwarzania danych wejściowych oraz zarządzania macierzami cech. Matplotlib umożliwił tworzenie wykresów przedstawiających wyniki eksperymentów, co ułatwiło analizę uzyskanych rezultatów.

Dzięki zastosowaniu powyższych narzędzi i bibliotek możliwe było skuteczne przeprowadzenie eksperymentów oraz szczegółowa analiza wyników, co przyczyniło się do osiągnięcia założonych celów badawczych.

3.3 Eksperymenty z autokoderem i ResNet50

Dla obu metod ekstrakcji cech – autokodera oraz ResNet50 – zastosowano te same techniki klasyfikacji w celu zapewnienia porównywalności wyników. Przeprowadzono eksperymenty z wykorzystaniem algorytmu kNN oraz centroidów, przy czym dla obu metod użyto następujących konfiguracji:

- Algorytm kNN:
 - $k = 5$,
 - $k = 3$,
 - prototypy klas (uśrednione wartości wektorów cech ze zbioru treningowego) jako reprezentacja klas w przypadku $k = 1$.
- Metoda centroidów:
 - centroidy utworzone na podstawie całego zbioru treningowego (wszystkie wektory cech treningowych dla danej klasy),
 - centroidy utworzone na podstawie prototypów klas (uśrednione wartości wektorów cech treningowych).

3.3.1 Autokoder

Autokoder został przeszkolony na zbiorze danych grzybów w celu uzyskania reprezentacji cech w przestrzeni utajonej o wymiarze 2048. Proces treningowy obejmował 2000 epok z następującymi parametrami:

- liczba kanałów wejściowych: 3 (obrazy w formacie RGB),
- wymiar przestrzeni utajonej: 2048,
- współczynnik uczenia: 1×10^{-6} ,
- rozmiar partii: 32.

Po zakończeniu treningu wyekstrahowano cechy ze wszystkich zdjęć w zbiorze danych i przeprowadzono klasyfikację przy użyciu algorytmu kNN oraz centroidów zgodnie z opisanymi wyżej konfiguracjami.

3.3.2 ResNet50

Model ResNet50 został zaadaptowany jako ekstraktor cech poprzez usunięcie końcowej warstwy klasyfikacyjnej. Wyekstrahowane cechy były następnie wykorzystywane w klasyfikacji z użyciem algorytmu kNN oraz centroidów. Sieć wytrenowana została z wykorzystaniem następujących parametrów:

- liczba kanałów wejściowych: 3 (obrazy w formacie RGB),
- wymiar przestrzeni cech: 2048,
- współczynnik uczenia: 1×10^{-6} ,
- rozmiar partii: 32.

3.4 Wizualizacja t-SNE

Dla obu modeli przeprowadzono również wizualizację przestrzeni cech z wykorzystaniem algorytmu t-SNE. Wizualizacje obejmowały:

- t-SNE dla zbioru treningowego,
- t-SNE dla całego zbioru danych,
- t-SNE dla prototypów klas utworzonych na podstawie zbioru treningowego,
- t-SNE dla prototypów klas utworzonych na podstawie całego zbioru danych.

Wyniki wizualizacji umożliwiły ocenę zdolności modeli do separacji klas w przestrzeni cech oraz analizę ich zdolności uogólniania.

3.5 Zbiór Danych

Do przeprowadzenia badań wykorzystano zbiór danych Mushroom species dataset [5], dostępny w serwisie Kaggle. Zbiór ten zawiera ponad 50 000 zdjęć przedstawiających 100 różnych gatunków

grzybów, które zostały zarejestrowane w Rosji. Zdjęcia w zbiorze charakteryzują się wysoką jakością i różnorodnością, co czyni je odpowiednimi do zadań klasyfikacji wizualnej.

Główne cechy zbioru danych:

- Ponad 50 000 zdjęć,
- Ilość klas: 100 gatunków grzybów,
- Rozdzielcość zdjęć: 256×256 pikseli (po wstępny przetwarzaniu),
- Źródło: zdjęcia wykonane w naturalnym środowisku w Rosji.

Zdjęcia zostały przetworzone w celu zapewnienia spójności danych wejściowych. Proces przetwarzania obejmował:

- skalowanie do rozdzielcości 256×256 pikseli,
- normalizację wartości pikseli do zakresu $[0, 1]$,
- podział na zestaw treningowy (80% danych) i walidacyjny (20% danych).
- zastosowanie losowych transformacji do zbioru treningowego:
 - losowe poziome odbicie,
 - losowa rotacja w zakresie ± 10 stopni.

Do treningu modeli wykorzystano jedynie 60 zdjęć z każdej klasy, zgodnie z podziałem na 80% zestawu treningowego i 20% walidacyjnego. W przypadku algorytmu k-Nearest Neighbors (kNN), wytrenowane modele przepuszczono przez cały zbiór danych, co pozwoliło na ocenę ich zdolności do generalizacji na większej liczbie próbek.

Wykorzystanie Mushroom species dataset umożliwiło przeprowadzenie kompleksowej analizy metod ekstrakcji cech oraz algorytmów klasyfikacyjnych. Zróżnicowanie klas grzybów pozwoliło na ocenę zdolności uogólniania zastosowanych modeli w rzeczywistych warunkach, gdzie obrazy mogą być podobne wizualnie, ale należeć do różnych kategorii.

3.6 Ewaluacja i Metryki

W niniejszej sekcji przedstawiono szczegółowy opis kryteriów i metod ewaluacji zastosowanych w pracy. Omówiono oczekiwane cechy dobrych modeli oraz wskaźniki, które pozwalają na ocenę ich skuteczności i zdolności generalizacyjnych. Przeanalizowano również wymagania wobec poszczególnych metod: autokodera, ResNet50, kNN, centroidów oraz wizualizacji t-SNE.

3.6.1 Metryki Ewaluacji

Główna metryką używaną do oceny modeli była precyza, która mierzy stosunek liczby poprawnych klasyfikacji do całkowitej liczby próbek w zestawie danych. Precyzję obliczano według wzoru:

$$\text{Precyza} = \frac{\text{Liczba poprawnych predykcji}}{\text{Całkowita liczba próbek}} \times 100\% \quad (17)$$

Dokładność mierzono dla następujących przypadków:

- Algorytmu kNN z wartościami $k = 5$, $k = 3$ oraz dla prototypów klas.

- Klasyfikacji na podstawie centroidów oraz dla prototypów klas.

We wszystkich przypadkach początkowe wartości punktów w przestrzeni obliczane były za pomocą zbioru treningowego, a następnie klasyfikacji dokonywano na pełnym zbiorze danych. Dodatkowo zastosowano analizę wizualną przestrzeni cech za pomocą t-SNE, aby ocenić separację klas oraz strukturę przestrzeni utajonej.

3.6.2 Cechy dobrego modelu

W kontekście niniejszej pracy każda z zastosowanych metod ewaluacji miała swoje unikalne kryteria jakości i cechy określające jej efektywność. Poniżej przedstawiono kryteria sukcesu dla każdej metody.

Autokoder powinien spełniać następujące cechy:

- wysoka jakość rekonstrukcji danych wejściowych,
- niska wartość funkcji straty rekonstrukcji,
- spójna i uporządkowana przestrzeń utajona,
- efektywna redukcja wymiarowości, przy jednoczesnym zachowaniu istotnych cech obrazu.

ResNet50, jako zaawansowana sieć splotowa, powinna:

- generować bogate i reprezentatywne cechy wizualne w przestrzeni utajonej,
- zapewniać dobre oddzielenie klas w przestrzeni cech,
- być stabilna podczas treningu i nie przejawiać oznak przeuczenia.

Algorytm kNN powinien:

- wykazywać wysoką dokładność klasyfikacji,
- być wrażliwy na jakość reprezentacji cech,
- w przypadku zastosowania prototypów klas dokładność powinna być zbliżona do tej uzyskanej przy użyciu pełnego zbioru danych treningowych.

Centroidy powinny:

- zapewniać intuicyjną interpretację, gdzie każdy centroid reprezentuje średnie cechy danej klasy,
- wykazywać dobrą zgodność z wynikami uzyskanymi przy użyciu pełnych zbiorów danych,
- prezentować stabilność wyników.

Wizualizacja za pomocą t-SNE powinna spełniać następujące kryteria:

- wyraźne i dobrze oddzielone skupiska punktów reprezentujących różne klasy,
- niskie nakładanie się klas,
- intuicyjne odwzorowanie lokalnych zależności w przestrzeni cech.

3.6.3 Zalety i ograniczenia zastosowanych metryk

Każda z zastosowanych metryk i metod ewaluacji ma swoje zalety i ograniczenia:

- dokładność: prosta i intuicyjna, ale nie uwzględnia przypadków, w których klasy są nierównomiernie reprezentowane,
- kNN: skuteczny w ocenie lokalnych zależności w przestrzeni cech, ale czas obliczeń rośnie wraz z wielkością zbioru danych,

- centroidy: intuicyjna metoda uproszczenia reprezentacji klas, ale podatna na zakłócenia przez szum w danych,
- t-SNE: umożliwia wizualną analizę przestrzeni cech, ale jej wyniki są trudniejsze do interpretacji ilościowej i zależne od dobranych hiperparametrów.

Dzięki zastosowaniu różnorodnych metryk i metod ewaluacji możliwe było uzyskanie wszechstronnej oceny efektywności zaproponowanych metod, co dostarczyło cennych informacji na temat ich zdolności do ekstrakcji cech i klasyfikacji obrazów grzybów.

Rozdział 4

Wyniki

W niniejszym rozdziale przedstawiono szczegółowe wyniki uzyskane podczas przeprowadzonych eksperymentów. Wyniki zostały zorganizowane w podsekcje, które szczegółowo omawiają efektywność metod KNN oraz Centroidów dla Autokodera oraz sieci ResNet50. Dodatkowo zaprezentowano wizualizacje procesów treningowych oraz analizy rozkładów cech w przestrzeniach utajonych za pomocą algorytmu t-SNE.

4.1 Wyniki autokodera

Autokoder został zastosowany jako metoda ekstrakcji cech wizualnych grzybów. Wyniki uzyskane w poszczególnych eksperymentach zostały podsumowane w Tabelach 1 oraz 2. Wyniki te stanowią średnią z pięciu powtórzeń każdego eksperymentu, co pozwoliło na zwiększenie wiarygodności pomiarów i ograniczenie wpływu losowych fluktuacji.

Tabela 1. Średnie wyniki dokładności kNN dla autokodera

Iteracja	kNN k=5	kNN k=3	kNN z prototypami
1	31,33%	29,31%	31,96%
2	31,45%	29,44%	32,02%
3	31,78%	29,38%	32,12%
4	31,52%	29,32%	31,97%
5	31,46%	29,42%	31,76%
Średnia ± Niepewność	31,51% ± 0,17%	29,37% ± 0,06%	31,97% ± 0,13%

Tabela 2. Średnie wyniki dokładności metody centroidów dla autokodera

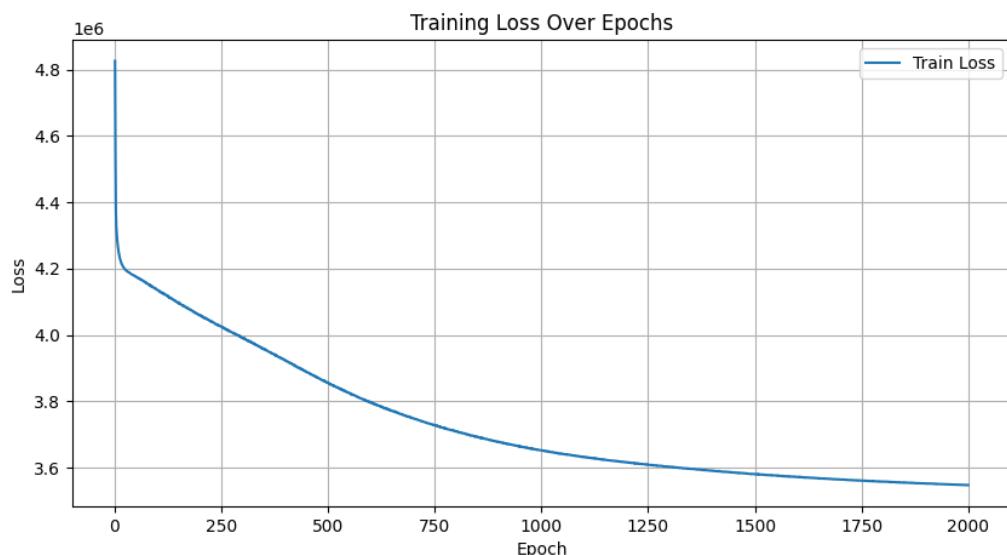
Iteracja	Centroid	Centroid z prototypami
1	26,43%	28,94%
2	26,40%	28,95%
3	26,49%	28,88%
4	26,50%	28,86%
5	26,53%	28,83%
Średnia ± Niepewność	26,47% ± 0,05%	28,89% ± 0,05%

Wyniki eksperymentów wskazują, że autokoder osiągnął najwyższą skuteczność w metodzie prototypowania wykorzystując metodę k-najbliższych sąsiadów, gdzie średnie wartości cech ze zbioru treningowego dla każdej klasy zostały wykorzystane jako reprezentatywne prototypy. Należy jednak zauważyć, że różnice w skuteczności między konfiguracjami kNN i metodą centroidów są stosunkowo niewielkie.

Rysunek 5 przedstawia zmienność funkcji straty w trakcie 2000 epok treningu autokodera. Wykres pokazuje wyraźny trend malejący, co wskazuje na skuteczne dopasowywanie modelu do danych treningowych. Początkowa wartość straty wynosiła około $4,8 \times 10^6$, a na końcu procesu treningowego osiągnęła wartość $3,6 \times 10^6$.

Tendencja spadkowa jest szczególnie widoczna w początkowych epokach, co sugeruje szybkie dopasowywanie się modelu do danych. Po około 1000 epokach tempo spadku maleje, co może wskazywać na zbliżanie się do minimum lokalnego funkcji straty.

Uzyskane wyniki potwierdzają stabilność procesu treningowego i skuteczność zastosowanego podejścia optymalizacji, które pozwoliło na stopniowe poprawianie jakości rekonstrukcji generowanych przez autokoder. Jednakowoż wartość końcowej straty sugeruje, iż istnieje potencjał do dalszego ulepszania modelu poprzez modyfikację architektury lub hiperparametrów treningowych.



Rysunek 5. Funkcja straty podczas treningu autokodera

Autokoder umożliwia generowanie zrekonstruowanych obrazów na podstawie przestrzeni utajonej. W Tabelach 3 oraz 4 przedstawiono przykłady obrazów dla wybranych klas grzybów, zawierające zdjęcia wzorcowe, wejściowe oraz obrazy odtworzone przez model.

Tabela 3. Rekonstrukcje obrazów widzianych podczas treningu

Klasa Grzyba	Zdjęcie Wzorcowe	Zdjęcie Wejściowe	Obraz Odtworzony
<i>Amanita citrina</i>			
<i>Calocera viscosa</i>			
<i>Flammulina velutipes</i>			
<i>Paxillus involutus</i>			
<i>Flammulina velutipes</i>			

Rekonstrukcje obrazów widzianych przez autokoder podczas treningu wykazują wysoką zgodność między obrazem wejściowym a odtworzonym, co wskazuje na zdolność modelu do zachowania kluczowych cech wizualnych. Obrazy odtworzone przez autokoder z danych, które były używane w procesie treningowym, są niemal identyczne do obrazów wzorcowych, co świadczy o skuteczności modelu w kompresji i dekompresji informacji wizualnej.

Rozdział 4. Wyniki

Ten wysoki poziom zgodności sugeruje, że autokoder nauczył się efektywnego odwzorowywania struktury i szczegółów obrazu w przestrzeni utajonej, zachowując zarówno globalne, jak i lokalne cechy charakterystyczne. Na przykład elementy takie jak kształty, tekstury czy wzory specyficzne dla poszczególnych klas grzybów zostały wiernie zachowane w procesie rekonstrukcji.

Tabela 4. Rekonstrukcje obrazów niewidzianych podczas treningu

Klasa Grzyba	Zdjęcie Wzorcowe	Zdjęcie Wejściowe	Obraz Odtworzony
<i>Amanita citrina</i>			
<i>Calocera viscosa</i>			
<i>Flammulina velutipes</i>			
<i>Paxillus involutus</i>			
<i>Flammulina velutipes</i>			

Tabela z Rekonstrukcjami Niewidzianych Obrazów Rekonstrukcja obrazów niewidzianych bezpośrednio podczas treningu, ale należących do klas znanych modelowi, ujawnia ograniczoną zdolność autokodera do generalizacji. Model, choć widział obrazy tych klas podczas procesu uczenia, nie miał dostępu do konkretnych próbek użytych w tej części eksperymentu. Wyniki pokazują, że rekonstrukcje tych obrazów są zauważalnie mniej precyzyjne w porównaniu do rekonstrukcji danych widzianych podczas treningu. W szczególności detale charakterystyczne dla określonych gatunków grzybów, takie jak faktura kapelusza czy specyficzne wzory na powierzchni, nie zawsze były wiernie odtworzone.

Niższa jakość rekonstrukcji może wynikać z trudności modelu w uchwyceniu pełnej różnorodności danych wewnętrz każdej klasy, co jest typowym ograniczeniem modeli uczących się z ograniczonej liczby próbek. Dodatkowo, wprowadzenie elementów zakłócających, takich jak różnorodność tła (np. ręce, liście, gałęzie) czy zmienność oświetlenia, mogło wpływać negatywnie na zdolność modelu do odseparowania istotnych cech wizualnych od informacji mniej istotnych.

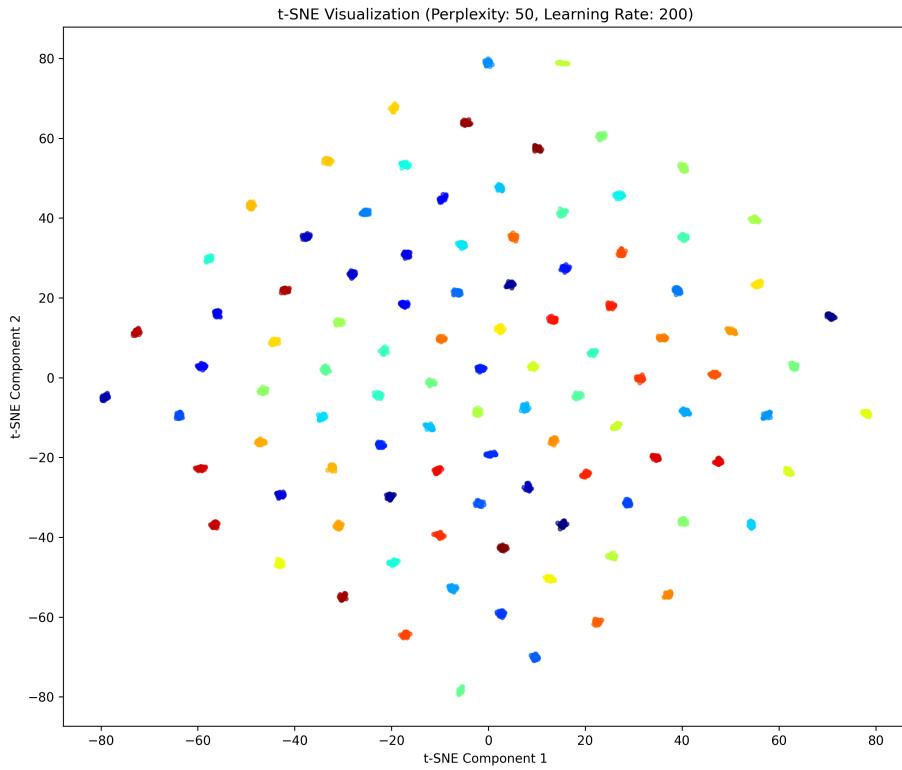
Wizualizacje t-SNE dla autokodera Aby lepiej zrozumieć strukturę przestrzeni utajonej, zastosowano algorytm t-SNE, który umożliwia redukcję wymiarowości cech i ich wizualizację w przestrzeni dwuwymiarowej. Przeprowadzono cztery eksperymenty wizualizacyjne, które pozwalają na analizę skupisk reprezentacji cech oraz ich strukturalnej spójności:

- Wizualizacja t-SNE dla zbioru treningowego (Rysunek 6),
- Wizualizacja t-SNE dla całego zbioru danych (Rysunek 7),
- Wizualizacja t-SNE dla prototypów klas wyznaczonych na podstawie wyłącznie zdjęć treningowych (Rysunek 8),
- Wizualizacja t-SNE dla prototypów klas wyznaczonych na podstawie całego zbioru danych (Rysunek 9).

Na Rysunku 6 przedstawiono rozmieszczenie próbek treningowych w przestrzeni utajonej, co ukazuje wyraźne, choć częściowo nachodzące na siebie skupiska reprezentacji cech dla poszczególnych klas. Analogicznie, na Rysunku 7 zaprezentowano układ reprezentacji cech dla całego zbioru danych, który pokazuje większą różnorodność danych oraz wyższy stopień pokrywania się skupisk klas.

W kolejnych eksperymentach, przedstawionych na Rysunku 8 oraz Rysunku 9, zastosowano po-dejście oparte na prototypach klas. Prototypy zostały wyznaczone jako średnie wartości reprezentacji cech dla każdej klasy, co pozwala na uproszczenie analizy oraz ocenę ogólnej struktury przestrzeni utajonej. Wyniki pokazują, że prototypy klas są bardziej odseparowane w przestrzeni dwuwymiarowej niż pojedyncze próbki, co podkreśla ich reprezentatywność i potencjał w zadaniach klasyfikacyjnych.

Wnioski z wizualizacji Wyniki wskazują, że autokoder jest w stanie skutecznie rozdzielać klasy w przestrzeni utajonej, jednak dla pełnego zbioru danych widoczny jest większy stopień nachodzenia na siebie reprezentacji klas. Wprowadzenie prototypów znacząco poprawia separację klas, co potwierdza ich przydatność w zadaniach klasyfikacyjnych. Szczególnie widać to na Rysunku 8 i Rysunku 9, gdzie



Rysunek 6. Wykres t-SNE dla autokodera - zbiór treningowy

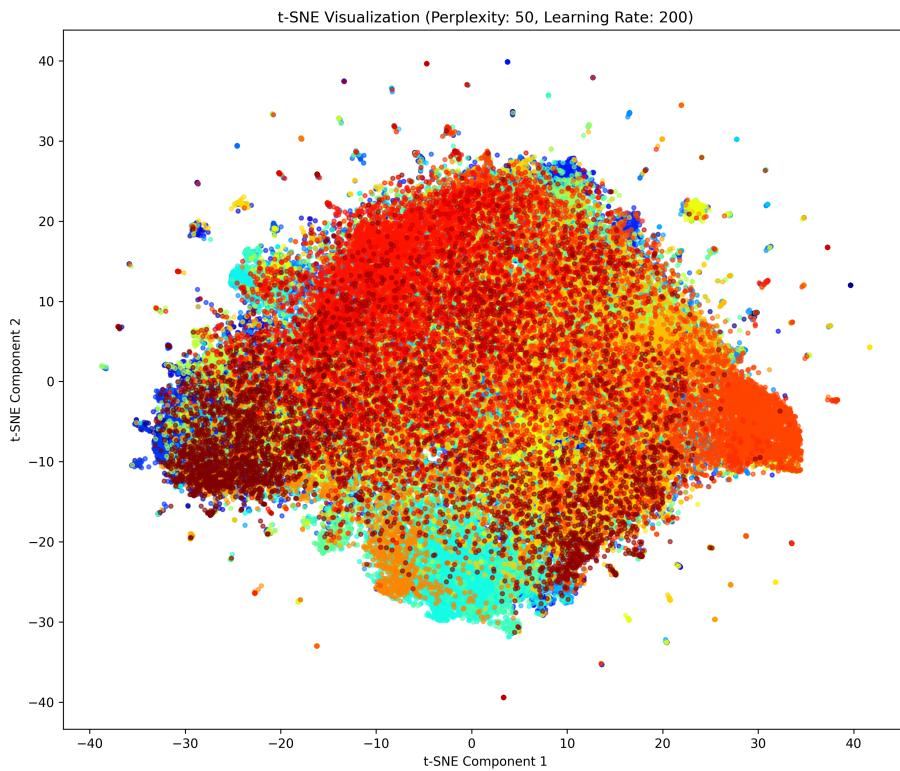
prototypy tworzą dobrze odseparowane punkty, co świadczy o ich reprezentatywności i potencjale w uproszczeniu zadania klasyfikacji w scenariuszach z ograniczoną ilością danych.

4.2 Wyniki ResNet50

W analogiczny sposób jak dla autokodera, sieć ResNet50 została oceniona pod kątem zdolności do generowania cech reprezentatywnych dla zadania klasyfikacji grzybów. Wyniki uzyskane w eksperymentach z wykorzystaniem metod klasyfikacji KNN oraz centroidów przedstawiono w Tabelach 5 oraz 6.

Wykresy procesu treningowego Rysunki 10 i 11 przedstawiają przebieg funkcji straty oraz dokładności podczas procesu treningowego sieci ResNet50.

Na Rysunku 10 widoczny jest stopniowy spadek wartości funkcji straty zarówno dla zbioru treningowego, jak i walidacyjnego. Na początku procesu wartości te były wyższe, jednak w miarę postępu treningu systematycznie malały, osiągając stabilny poziom pod koniec. Brak dużych różnic



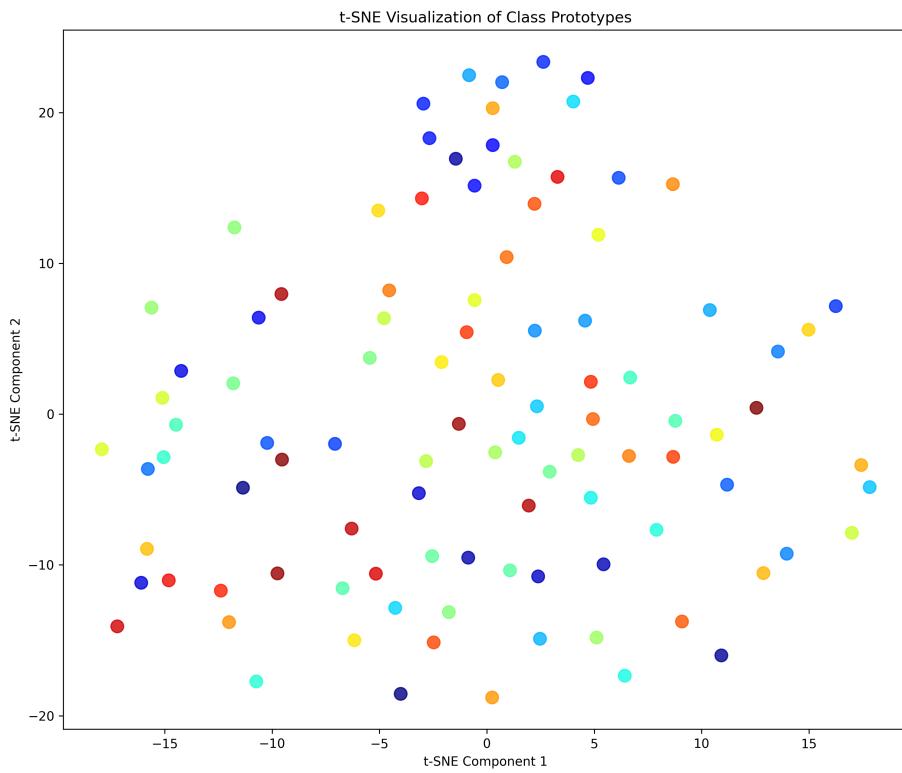
Rysunek 7. Wykres t-SNE dla autokodera - pełny zbiór danych

między stratą na zbiorze treningowym a walidacyjnym wskazuje na stabilność treningu oraz dobrą generalizację modelu.

Rysunek 11 przedstawia zmianę dokładności modelu w czasie. Obserwujemy stopniowy wzrost dokładności na zbiorze treningowym i walidacyjnym, co wskazuje na skuteczne przyswajanie przez model wzorców z danych. Ostateczne wartości dokładności sugerują, że ResNet50 skutecznie nauczył się rozróżniać klasy na podstawie danych wizualnych, jednak ostateczna różnica między zbiorami treningowym i walidacyjnym wskazuje na możliwość dalszych usprawnień w kontekście zdolności generalizacyjnych.

Te wyniki podkreślają, że proces treningowy ResNet50 przebiegał stabilnie i efektywnie, co czyni model wartościowym narzędziem do ekstrakcji cech i ich dalszej analizy w kontekście klasyfikacji obrazów grzybów.

Wizualizacje t-SNE dla ResNet50 W celu analizy przestrzeni utajonej generowanej przez sieć ResNet50, zastosowano algorytm t-SNE, umożliwiający redukcję wymiarowości i wizualizację cech w przestrzeni dwuwymiarowej. Przeprowadzono cztery eksperymenty wizualizacyjne, pozwalające na ocenę zdolności sieci do separacji klas oraz reprezentatywności prototypów.

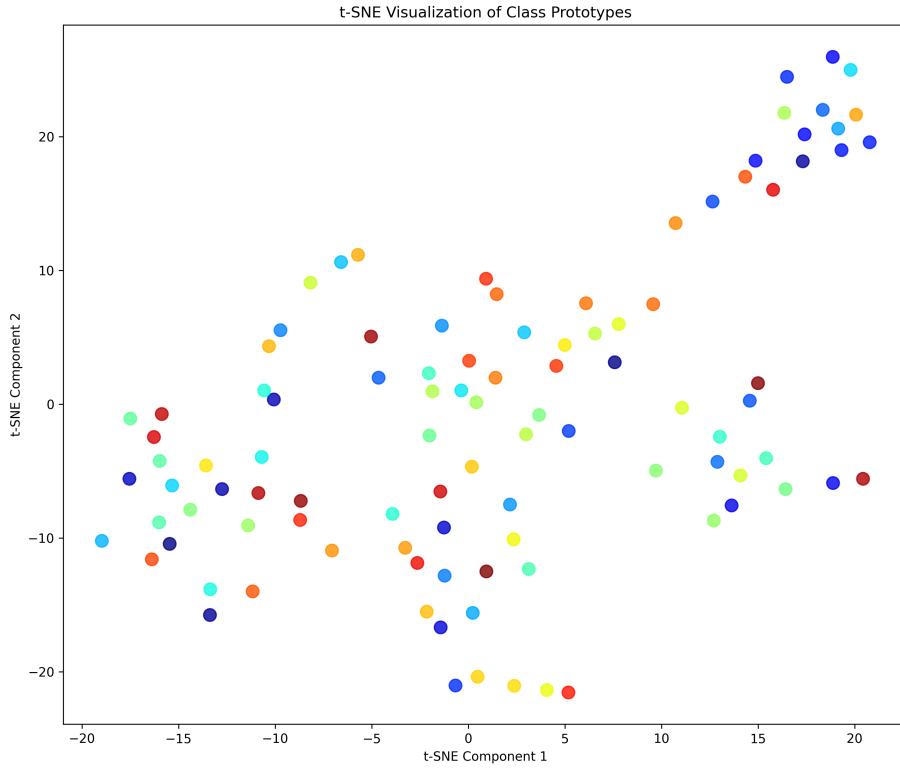


Rysunek 8. Wykres t-SNE dla prototypów klas dla autokodera – zbiór treningowy

Na Rysunku 12 przedstawiono rozmieszczenie próbek treningowych w przestrzeni utajonej. Widoczne są wyraźne skupiska reprezentacji cech, co świadczy o zdolności ResNet50 do odróżniania klas. Na Rysunku 13 zaprezentowano pełny zbiór danych, w którym zauważalna jest większa różnorodność i częstocie nachodzenie się klas.

W kolejnych eksperymentach przedstawionych na Rysunkach 14 i 15 zastosowano prototypy klas wyznaczone jako średnie reprezentacje cech dla każdej klasy. Wyniki pokazują, że prototypy są wyraźnie odseparowane, co potwierdza ich przydatność w uproszczeniu zadania klasyfikacji.

Wnioski z wizualizacji Wizualizacje wskazują, że ResNet50 generuje reprezentacje cech o wysokim stopniu separacji między klasami, szczególnie dla zbioru treningowego (Rysunek 12). Dla pełnego zbioru danych widoczne jest większe nachodzenie się klas, co wynika z większej różnorodności próbek (Rysunek 13). Prototypy klas (Rysunki 14 i 15) wykazują znaczną odseparowanie w przestrzeni utajonej, co potwierdza ich reprezentatywność i przydatność w klasyfikacji. Wyniki te podkreślają skuteczność ResNet50 w ekstrakcji cech i separacji klas w przestrzeni utajonej.



Rysunek 9. Wykres t-SNE dla prototypów klas dla autokodera - pełny zbiór danych

4.3 Problemy Napotkane Podczas Treningu

Podczas realizacji badań napotkano szereg problemów technicznych i praktycznych, które miały istotny wpływ na przebieg eksperymentów oraz ostateczne wyniki. W niniejszej podsekcji szczegółowo opisano najważniejsze z nich oraz podjęte kroki w celu ich rozwiązania.

- **Dobór współczynnika uczenia**

Jednym z kluczowych problemów było dobranie odpowiedniej wartości parametru współczynnika uczenia, który odgrywa istotną rolę w procesie uczenia sieci neuronowej. Za duży współczynnik uczenia powodował niestabilność procesu uczenia. Wagi modelu były aktualizowane zbyt gwałtownie, co prowadziło do przeskakiwania pomiędzy lokalnymi minimami funkcji straty i uniemożliwiało osiągnięcie globalnego minimum. W niektórych przypadkach prowadziło to również do eksplozji gradientów, skutkującej przepełnieniem wag i brakiem dalszej możliwości poprawy wyników.

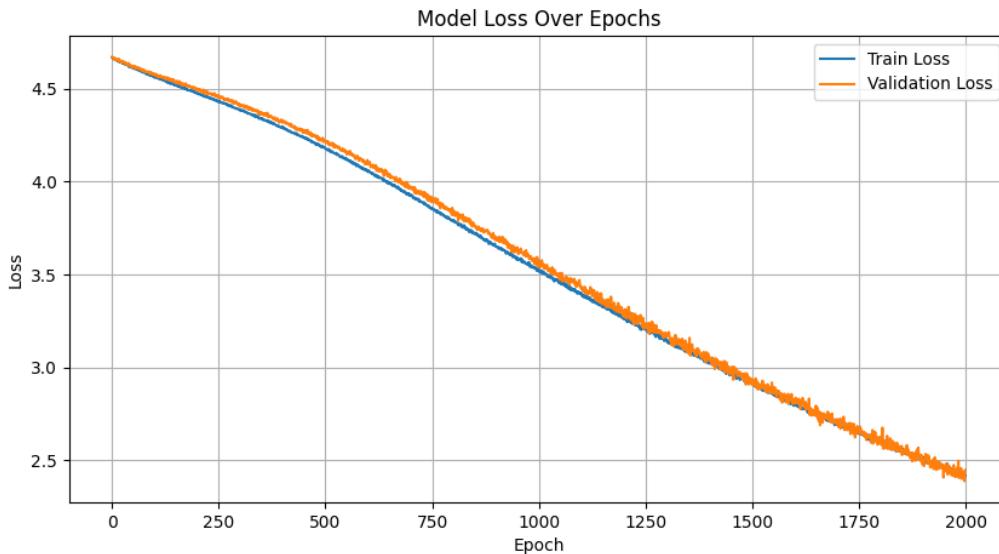
W celu uniknięcia tych problemów przeprowadzono serię eksperymentów z różnymi wartościami współczynnika uczenia. Okazało się, że wartość 10^{-6} była w stanie zapewnić najbardziej stabilny proces uczenia oraz największą efektywność zbieżności dla funkcji straty.

Tabela 5. Średnie wyniki dokładności KNN dla ResNet50

Model	KNN k=5	KNN k=3	KNN z prototypami
1	46,63%	44,83%	48,44%
2	46,63%	45,05%	48,48%
3	46,67%	44,93%	48,32%
4	46,59%	44,89%	48,40%
5	46,61%	44,98%	48,46%
Średnia ± Niegwość	46,63% ± 0,03%	44,94% ± 0,08%	48,42% ± 0,06%

Tabela 6. Średnie wyniki dokładności metody centroidów dla ResNet50

Model	Centroid	Centroid z prototypami
1	48,44%	48,45%
2	48,41%	48,72%
3	48,37%	48,68%
4	48,36%	48,82%
5	48,37%	48,56%
Średnia ± Niegwość	48,39% ± 0,03%	48,65% ± 0,14%

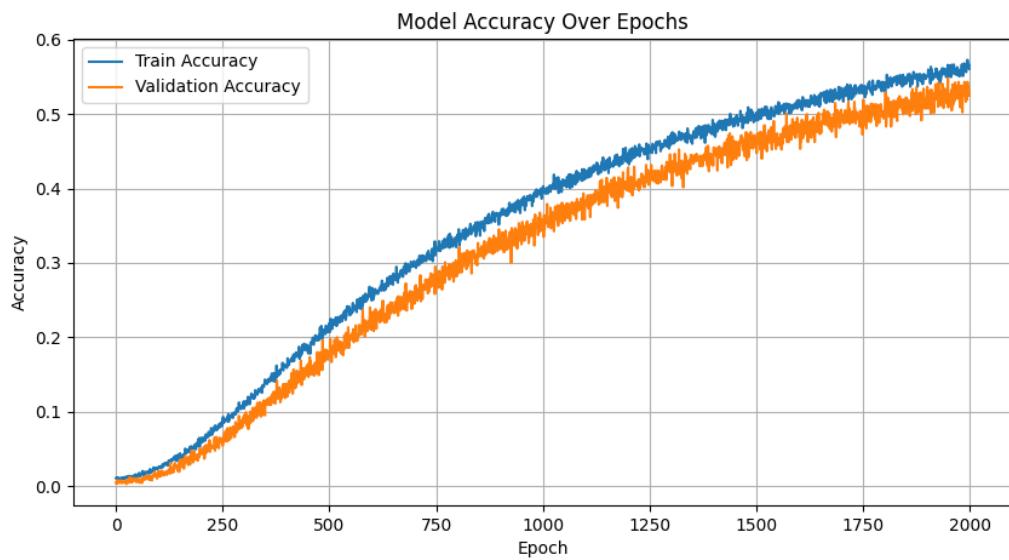


Rysunek 10. Funkcja straty podczas treningu i walidacji ResNet50

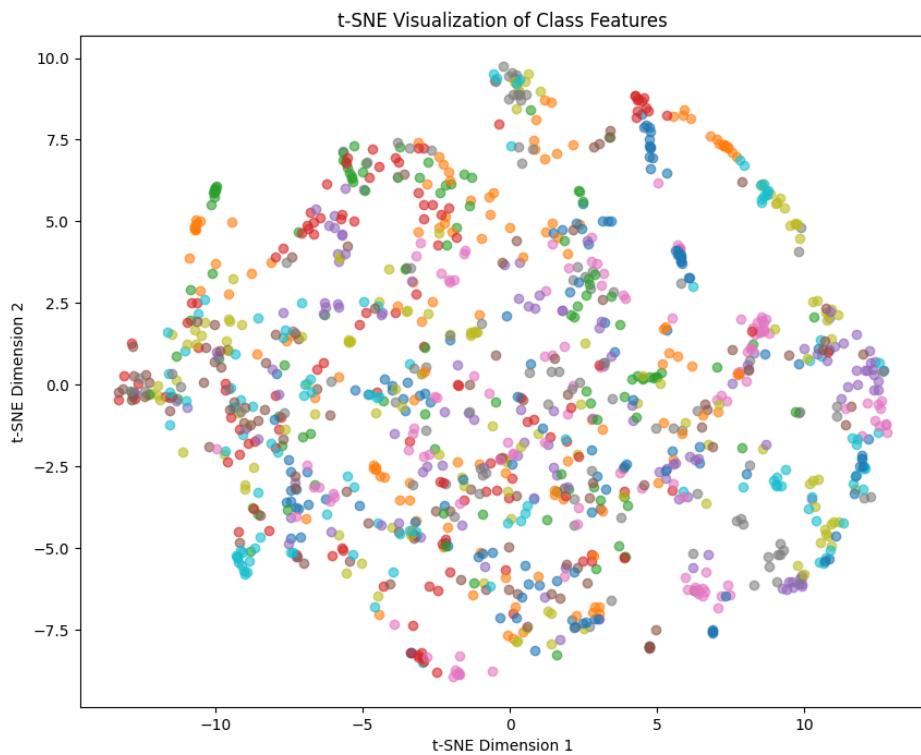
- **Problemy z funkcją straty**

Początkowo w procesie uczenia autoencodera zastosowano funkcję straty opartą na średnim błędzie kwadratowym (*Mean Squared Error, MSE*). Chociaż funkcja ta dobrze radzi się w przypadku zadań rekonstrukcji, w niniejszym projekcie prowadziła do niezadowalających wyników. Model uczył się przede wszystkim kompresji obrazów, co skutkowało niską jakością zdjęć wyjściowych, zamiast ekstrakcji istotnych cech potrzebnych do dalszej analizy.

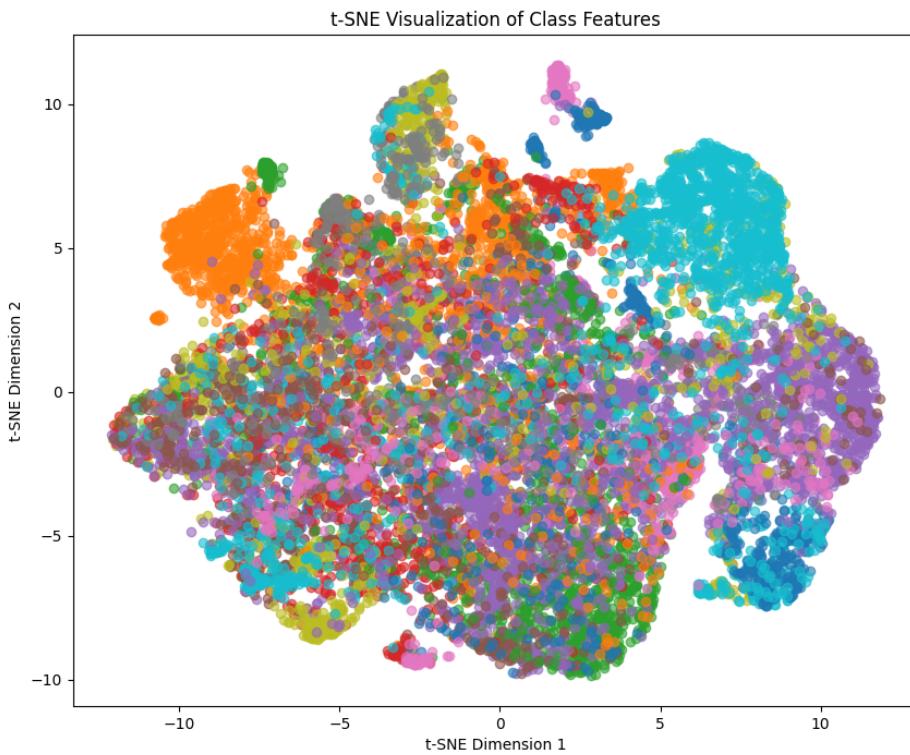
Rozwiązańiem tego problemu było zastosowanie funkcji straty opartej na kombinacji *Binarnej Entropii Krzyżowej (BCE)* oraz *Dywergencji Kullbacka-Leiblera (KLD)*, co jest standardowym



Rysunek 11. Funkcja dokładności podczas treningu i walidacji ResNet50



Rysunek 12. Wykres t-SNE dla ResNet50 - zbiór treningowy



Rysunek 13. Wykres t-SNE dla ResNet50 - pełny zbiór danych

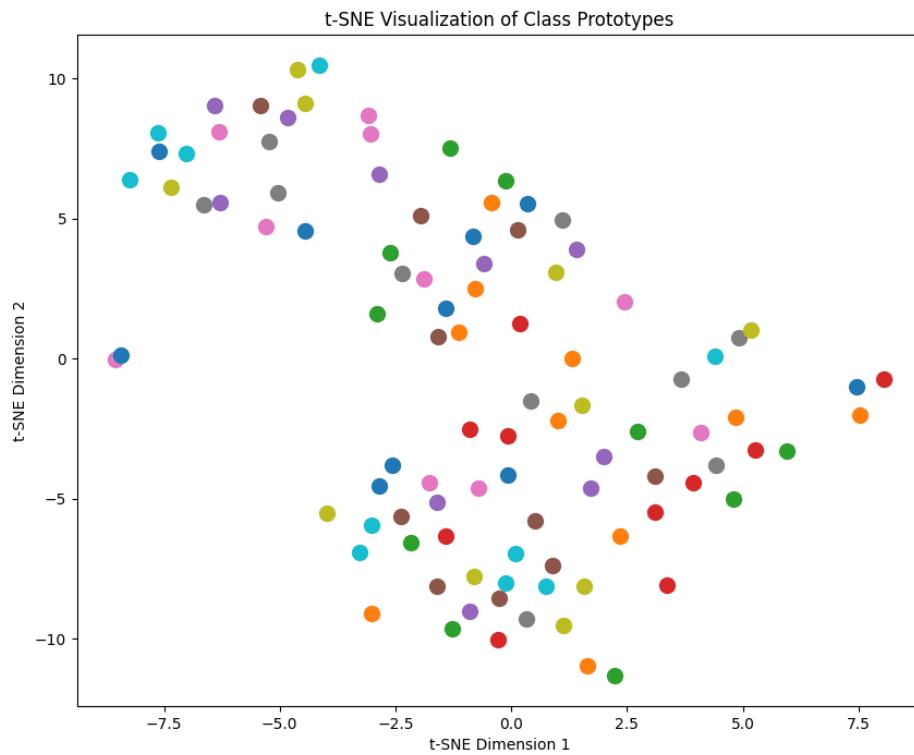
podejściem w przypadku autokoderów wariacyjnych (*Variational Autoencoders, VAE*). Zamiana MSE na BCE+KLD pozwoliła modelowi na bardziej efektywną naukę istotnych cech obrazów, jednocześnie zachowując wysoką jakość rekonstrukcji.

Dla sieci ResNet50 zastosowano funkcję straty entropii krzyżowej (Cross Entropy Loss). Ta funkcja straty, powszechnie używana w zadaniach klasyfikacyjnych, okazała się skuteczna w uczeniu modelu prawidłowego rozpoznawania klas na podstawie wyekstrahowanych cech. Jej zastosowanie pozwoliło na uzyskanie wysokiej skuteczności klasyfikacji oraz stabilnego procesu uczenia.

- **Wolne trenowanie sieci**

Kolejnym problemem, który znaczco wpływał na przebieg eksperymentów, była duża czasochłonność trenowania modelu. Przy początkowej implementacji proces uczenia był ograniczony do jednego GPU, co w przypadku dużego zbioru danych znaczco wydłużało czas potrzebny na każdą epokę.

Problem ten został rozwiązany dzięki zastosowaniu równoległego przetwarzania danych (*Data Parallel*) w bibliotece PyTorch. Wykorzystanie tej techniki umożliwiło rozłożenie obliczeń na wszystkie dostępne GPU serwera (4 jednostki), co pozwoliło na czterokrotne przyspieszenie



Rysunek 14. Wykres t-SNE dla prototypów klas dla ResNet50 - zbiór treningowy

procesu uczenia. Dzięki temu możliwe było przeprowadzenie większej liczby eksperymentów w krótszym czasie, co pozytywnie wpłynęło na jakość uzyskanych wyników.

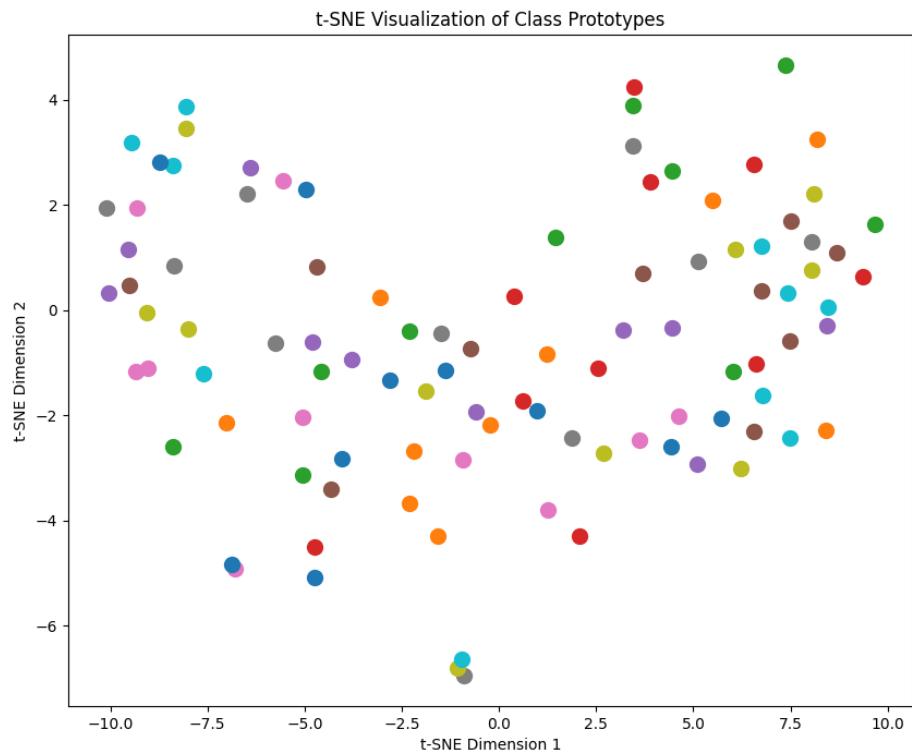
Dzięki rozwiązaniu powyższych problemów możliwe było przeprowadzenie badań zgodnie z założeniami, co pozwoliło na uzyskanie wiarygodnych i satysfakcjonujących wyników.

4.4 Wnioski

Eksperymenty przeprowadzone w ramach niniejszych badań dostarczyły istotnych informacji na temat skuteczności różnych metod ekstrakcji cech i klasyfikacji obrazów grzybów. Poniżej przedstawiono kluczowe wnioski:

4.4.1 Autokoder jako metoda ekstrakcji cech

Autokoder okazał się skutecznym narzędziem do reprezentacji cech obrazów w przestrzeni utajonej, choć jego wydajność była zauważalnie niższa w porównaniu do ResNet50. Wprowadzenie prototypów (średnich wartości cech dla każdej klasy) zauważalnie poprawiło dokładność klasyfikacji zarówno dla metody kNN, jak i centroidów. Wizualizacje t-SNE pokazały, że autokoder potrafi generować



Rysunek 15. Wykres t-SNE dla prototypów klas dla ResNet50 - pełny zbiór danych

skupiska reprezentacji cech dla poszczególnych klas, choć z widocznymi przypadkami nachodzenia się klas. Jest to szczególnie zauważalne w scenariuszach z większą różnorodnością próbek.

4.4.2 ResNet50 jako ekstraktor cech

ResNet50 osiągnął wyższą dokładność w porównaniu do autokodera, co potwierdza jego zdolność do efektywnego modelowania cech wizualnych w zadaniach klasyfikacyjnych. Podobnie jak w przypadku autokodera, wykorzystanie prototypów zauważalnie poprawiło wyniki klasyfikacji. Wprowadzenie metody centroidów na podstawie prototypów pozwoliło na osiągnięcie najwyższych wyników w eksperymentach. Wizualizacje t-SNE potwierdziły, że cechy generowane przez ResNet50 charakteryzują się większą separacją między klasami w przestrzeni utajonej, co przekłada się na jego wysoką skuteczność w zadaniach klasyfikacyjnych.

4.4.3 Porównanie metod klasyfikacji (kNN i centroidy)

Zarówno metoda kNN, jak i centroidów, osiągały lepsze wyniki po zastosowaniu prototypów, co wskazuje na przydatność tej techniki w uproszczeniu zadań klasyfikacyjnych. Metoda centroidów okazała się nieco skuteczniejsza niż kNN, co sugeruje, że reprezentowanie klas pojedynczym

centroidem może być bardziej stabilne i mniej podatne na wpływ szumu danych. Dla obu modeli i metod, dokładność klasyfikacji wzrosła wraz z wprowadzeniem prototypów, co podkreśla ich znaczenie w analizie danych o wysokiej różnorodności.

4.4.4 Wizualizacje t-SNE

Wizualizacje potwierdziły, że zarówno autokoder, jak i ResNet50 są zdolne do generowania przestrzeni cech o strukturze wspierającej klasyfikację. Prototypy okazały się szczególnie przydatne w analizie t-SNE, pokazując, że reprezentacje klas w przestrzeni utajonej stają się bardziej odseparowane, co sprzyja skuteczniejszej klasyfikacji. Autokoder wykazywał większe nachodzenie się klas w przestrzeni utajonej w porównaniu do ResNet50, co częściowo tłumaczy jego niższą skuteczność.

4.4.5 Praktyczne zastosowania wyników

Uzyskane wyniki i wyciągnięte wnioski mogą znaleźć szerokie zastosowanie praktyczne.

- **Ulepszanie systemów klasyfikacji**

Wyniki eksperymentów mogą zostać wykorzystane do projektowania bardziej efektywnych systemów klasyfikacji obrazów grzybów, szczególnie w scenariuszach, gdzie dostępne są ograniczone zasoby obliczeniowe. Metody oparte na prototypach (zarówno w kNN, jak i centroidach) mogą być przydatne do uproszczenia i przyspieszenia klasyfikacji bez znaczącego obniżenia dokładności.

- **Zastosowanie w nauce o środowisku i bioróżnorodności**

Modele opracowane w ramach pracy mogą znaleźć zastosowanie w automatycznych systemach identyfikacji grzybów, wspierając naukowców w analizie bioróżnorodności, monitorowaniu ekosystemów czy klasyfikacji gatunków zagrożonych.

- **Wykorzystanie prototypów w transfer learningu**

Zastosowanie prototypów może być rozwijane w kontekście transfer learningu, gdzie informacje z jednego zestawu danych mogą zostać użyte do klasyfikacji w innym, zbliżonym zestawie. Może to wspierać zadania związane z adaptacją modeli do nowych klas lub nowych zbiorów danych.

- **Rozwój systemów o ograniczonej liczbie danych treningowych**

Prototypy szczególnie dobrze sprawdzają się w scenariuszach, gdzie liczba próbek treningowych dla każdej klasy jest ograniczona. Dalsze badania w tym kierunku mogą przyczynić się do rozwoju systemów zdolnych do klasyfikacji przy minimalnej liczbie danych.

Rozdział 5

Podsumowanie

Celem niniejszej pracy było przeprowadzenie badań skuteczności metody prototypowania w zadaniu rozpoznawania obrazów grzybów oraz ocena jej efektywności w generowaniu reprezentacji cech wizualnych. Cel ten został spełniony, badania obejmowały implementację i analizę metod prototypowania przy użyciu autokodera oraz sieci ResNet50, ewaluację ich skuteczności w generowaniu reprezentacji cech wizualnych, a także wykorzystanie tych reprezentacji w zadaniach porównawczych między prototypowaniem oraz klasycznymi metodami klasyfikacji. Wyniki eksperymentów dostarczyły wartościowych wniosków na temat potencjału prototypowania w rozwiązywaniu problemów związanych z wizją komputerową, szczególnie w kontekście klasyfikacji obrazów w warunkach ograniczonych danych treningowych. Przeprowadzono szereg eksperymentów, których celem było zbadanie zdolności głębokich sieci neuronowych autokoder oraz ResNet50 do generowania reprezentatywnych cech umożliwiających skutecną klasyfikację przy użyciu algorytmu k-Nearest Neighbors (kNN) oraz metod centroidowych. Eksperymenty zostały zaprojektowane w sposób uwzględniający ograniczoną liczbę próbek treningowych, co miało na celu symulację warunków rzeczywistych, gdzie dostęp do dużych zbiorów danych może być ograniczony. W kontekście badań nad few-shot learning [33] oraz zero-shot learning [18], niniejsza praca stanowi wkład w rozwój metod rozpoznawania obrazów o dużej różnorodności wizualnej.

Eksperymenty obejmowały zastosowanie obu metod (autokodera i ResNet50) w czterech konfiguracjach:

- klasyfikacja z użyciem kNN dla liczby sąsiadów $k = 5$,
- klasyfikacja z użyciem kNN dla liczby sąsiadów $k = 3$,
- klasyfikacja z użyciem klasycznych centroidów,
- klasyfikacja z użyciem centroidów opartych na prototypach (średnich wartości cech dla każdej klasy).

Dla każdej konfiguracji przeprowadzono pięć powtórzeń eksperymentu, a uzyskane wyniki zostały uśrednione w celu zwiększenia ich wiarygodności.

Eksperymenty przeprowadzone w ramach niniejszej pracy potwierdziły skuteczność metody prototypowania w problemie rozpoznawania obrazów grzybów. Zastosowanie prototypów umożliwiło

uproszczenie procesu klasyfikacji oraz poprawę jego odporności na zakłócenia w danych, uzyskując najlepsze wyniki we wszystkich przypadkach testowych.

Dodatkowo, analiza przestrzeni cech za pomocą metody t-SNE pozwoliła na ocenę separacji klas, ukazując zalety zastosowania prototypów w zwiększeniu separacji pomiędzy poszczególnymi klasami. Wyniki te podkreślają potencjał metod prototypowania w scenariuszach z ograniczoną ilością danych treningowych oraz w zadaniach wymagających efektywności obliczeniowej.

W przyszłości możliwe jest dalsze rozwijanie zaproponowanych metod poprzez:

- wykorzystanie nowszych architektur, takich jak transformers, które zyskują na popularności w zadaniach wizji komputerowej [30],
- badanie innych algorytmów klasyfikacyjnych, takich jak SVM czy Random Forest, w celu porównania z wynikami kNN i centroidów,
- implementację mechanizmów aktywnego uczenia (active learning), co pozwoliłoby na lepsze radzenie sobie z ograniczonymi danymi [32],
- analizę odporności modeli na zakłócenia w danych, co jest szczególnie istotne w rzeczywistych scenariuszach aplikacyjnych,
- rozszerzenie zestawu metryk oceny, co pozwoli na bardziej wszechstronną ocenę modeli w scenariuszach wieloklasowych.

Kolejnym kierunkiem rozwoju może być adaptacja zaproponowanych modeli do scenariuszy zero-shot learning, co umożliwiłoby klasyfikację nowych gatunków bez konieczności przeprowadzania pełnego treningu. Metody hierarchicznego uczenia prototypowego [18, 34] mogą stanowić solidną podstawę do realizacji tego celu.

Podsumowując, praca dostarcza istotnych wniosków na temat skuteczności prototypowania w kontekście klasyfikacji obrazów grzybów. W trakcie pracy zastosowano różnorodne podejścia, w tym implementacje głębokich sieci neuronowych autokoder oraz ResNet50, a także klasyfikację z użyciem algorytmu kNN i metody centroidów. Ponadto przeprowadzono wizualizację przestrzeni cech przy użyciu algorytmu t-SNE, co pozwoliło na lepsze zrozumienie struktury danych i zachowania modeli.

Bibliografia

- [1] „Centroids by Integration,” *University of Memphis*, 2023. adr.: <https://www.ce.memphis.edu/2131/PDFsF12/Centroids%20by%20Integration.pdf>.
- [2] Corleone, M., „Centroids and Centres of Gravity,” *Academia.edu*, 2023. adr.: https://www.academia.edu/31110335/Centroids_and_Centres_of_Gravity.
- [3] Creswell, A., Arulkumaran, K. i Bharath, A. A., „On Denoising Autoencoders Trained to Minimise Binary Cross-Entropy,” *arXiv preprint arXiv:1708.08487*, 2017. adr.: <https://arxiv.org/abs/1708.08487>.
- [4] Cunningham, P. i Delany, S. J., „k-Nearest Neighbour Classifiers: 2nd Edition (with Python examples),” *arXiv preprint arXiv:2004.04523*, 2020. adr.: <https://arxiv.org/abs/2004.04523>.
- [5] Dautov, A. i Damir, A., *Mushroom species dataset*, <https://www.kaggle.com/datasets/thehir0/mushroom-species>, Over 50,000 photos of 100 species of mushrooms taken in Russia, 2023.
- [6] Duch, W., „Historia rozwoju sztucznych sieci neuronowych,” *Prace Naukowe Politechniki Wrocławskiej*, t. 22, s. 45–60, 2022. adr.: https://kcir.pwr.edu.pl/~witold/ai/ml_nndeep_s.pdf.
- [7] Frąckowiak, M., „Analiza obrazów – przegląd metod i inspiracji teoretycznych,” *Folia Sociologica*, t. 32, s. 6–27, 2007. adr.: <https://dspace.uni.lodz.pl/bitstream/handle/11089/11073/Folia%20Sociologika%2032%202007%206-27.pdf?sequence=1>.
- [8] Fu, Z., Xiang, T., Kodirov, E. i Gong, S., „Zero-Shot Learning on Semantic Class Prototype Graph,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, t. 40, nr. 8, s. 2009–2021, 2018.
- [9] Gu, J. i in., „A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects,” *arXiv preprint arXiv:2004.02806*, 2018.
- [10] Halder, R. K., Uddin, M. N., Uddin, A., Aryal, S. i Khraisat, A., „Enhancing K-nearest neighbor algorithm: a comprehensive review and applications,” *Journal of Big Data*, 2024. adr.: <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-024-00973-y>.
- [11] He, K., Zhang, X., Ren, S. i Sun, J., „Deep Residual Learning for Image Recognition,” *arXiv preprint arXiv:1512.03385*, 2015.

Bibliografia

- [12] Islam, S., „An Introduction to Convolutional Neural Networks,” *arXiv preprint arXiv:1511.08458*, 2015.
- [13] Kim, J., Oh, T.-H., Lee, S., Pan, F. i Kweon, I. S., *Variational Prototyping-Encoder: One-Shot Learning with Prototypical Images*, <https://github.com/mibastro/VPE>, [Online; dostęp: 8 grudnia 2024], 2019.
- [14] Kingma, D. P. i Welling, M., „An Introduction to Variational Autoencoders,” *Foundations and Trends in Machine Learning*, 2019. adr.: <https://arxiv.org/abs/1906.02691>.
- [15] Krawiec, K., „Metody rozpoznawania obrazów,” *Politechnika Poznańska Prace Naukowe*, t. 15, s. 15–30, 2005. adr.: <https://www.cs.put.poznan.pl/kkrawiec/wiki/uploads/Zajecia/po0ld1.pdf>.
- [16] Linderman, G. C. i Steinerberger, S., „Clustering with t-SNE, provably,” *arXiv preprint arXiv:1706.02582*, 2017. adr.: <https://arxiv.org/pdf/1706.02582v1.pdf>.
- [17] Liu, X., Liu, C., Li, X. i Gong, S., „Fast Multi-View Clustering via Prototype Graph,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, t. 43, nr. 6, s. 1893–1904, 2021.
- [18] Luo, C., Li, Z., Huang, K., Feng, J. i Wang, M., „Zero-Shot Learning via Attribute Regression and Class Prototype Rectification,” *IEEE Transactions on Image Processing*, t. 27, nr. 2, s. 637–647, 2018.
- [19] Maaten, L. van der i Hinton, G., „Visualizing Data using t-SNE,” *Journal of Machine Learning Research*, t. 9, s. 2579–2605, 2008. adr.: <https://jmlr.csail.mit.edu/papers/v9/vandermaaten08a.html>.
- [20] Mukherjee, S., „The Annotated ResNet-50,” *Towards Data Science*, 2022, [Online; dostęp: 8 grudnia 2024]. adr.: <https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758>.
- [21] Rocca, J., „Understanding Variational Autoencoders (VAEs),” *Towards Data Science*, 2019. adr.: <https://towardsdatascience.com/understanding-variational-autoencoders-vae-f70510919f73>.
- [22] Rosch, E., „Natural categories,” *Cognitive Psychology*, t. 4, nr. 3, s. 328–350, 1973.
- [23] Shlens, J., „Notes on Kullback-Leibler Divergence and Likelihood,” *arXiv preprint arXiv:1404.2000*, 2014. adr.: <https://arxiv.org/abs/1404.2000>.
- [24] Snell, J., Swersky, K. i Zemel, R., „Prototypical Networks for Few-shot Learning,” *Advances in Neural Information Processing Systems*, 2017. adr.: <https://arxiv.org/abs/1703.05175>.
- [25] Sosnowski, Z., „Sieci neuronowe w przetwarzaniu obrazów: przegląd wybranych osiągnięć,” *Symulacje komputerowe w badaniach naukowych*, t. 1, s. 213–230, 2020. adr.: <https://pb.edu.pl/oficyna-wydawnicza/wp-content/uploads/sites/4/2020/11/pod-red-Z-Sosnowskiego-Symulacje-rozdz-13.pdf>.
- [26] Szegedy, C., Ioffe, S., Vanhoucke, V. i Alemi, A., „Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” *arXiv preprint arXiv:1602.07261*, 2016.

- [27] Tadeusiewicz, R., „Sieci neuronowe – przewodnik problemowy,” Wydawnictwo Akademii Górnictwa i Hutniczej, t. 3, s. 10–25, 1991. adr.: https://www.academia.edu/29855346/Sieci_neuronowe_przewodnik_problemowy.
- [28] Tan, M. i Le, Q. V., „EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” *arXiv preprint arXiv:1905.11946*, 2019.
- [29] Wang, K., Liew, J. H., Zou, Y., Zhou, D. i Feng, J., „PANet: Few-Shot Image Semantic Segmentation with Prototype Alignment,” *Proceedings of the IEEE International Conference on Computer Vision*, s. 9197–9206, 2019.
- [30] Wang, K., Zhou, D. i Feng, J., „Improved Prototypical Network for Active Few-Shot Learning,” *Pattern Recognition Letters*, t. 130, s. 15–23, 2020.
- [31] Wang, K., Zou, Y. i Liew, J., „Hyperbolic Prototypical Network for Few-Shot Remote Sensing Scene Classification,” *Remote Sensing*, t. 14, s. 35–50, 2022.
- [32] Yu, T., Lin, J. i Zhu, X., „Episode-Based Prototype Generating Network for Zero-Shot Learning,” w *CVPR*, 2020.
- [33] Zhang, B., Li, X., Ye, Y., Huang, Z. i Zhang, L., „Prototype Completion with Primitive Knowledge for Few-Shot Learning,” w *CVPR*, 2021.
- [34] Zhang, Z., Chen, R., Cao, W., Tai, Y. i Wang, C., „Learning Neural Proto-Face Field for Disentangled 3D Face Modeling in the Wild,” w *CVPR*, 2023.
- [35] Zhang, Z., Xiang, T., Feng, J. i Wang, M., „Transferable Prototypical Networks for Unsupervised Domain Adaptation,” *CVPR*, s. 123–135, 2020.

Spis rysunków

1	Schemat budowy autokodera [21].	16
2	Schemat fazy treningowej autokodera wariacyjnego [13]	18
3	Schemat fazy testowej autokodera wariacyjnego [13]	19
4	Schemat działania sieci ResNet50 [20]	21
5	Funkcja straty podczas treningu autokodera	34
6	Wykres t-SNE dla autokodera - zbiór treningowy	38
7	Wykres t-SNE dla autokodera - pełny zbiór danych	39
8	Wykres t-SNE dla prototypów klas dla autokodera - zbiór treningowy	40
9	Wykres t-SNE dla prototypów klas dla autokodera - pełny zbiór danych	41
10	Funkcja straty podczas treningu i walidacji ResNet50	42
11	Funkcja dokładności podczas treningu i walidacji ResNet50	43
12	Wykres t-SNE dla ResNet50 - zbiór treningowy	43
13	Wykres t-SNE dla ResNet50 - pełny zbiór danych	44
14	Wykres t-SNE dla prototypów klas dla ResNet50 - zbiór treningowy	45
15	Wykres t-SNE dla prototypów klas dla ResNet50 - pełny zbiór danych	46

Spis tabel

1	Średnie wyniki dokładności kNN dla autokodera	33
2	Średnie wyniki dokładności metody centroidów dla autokodera	33
3	Rekonstrukcje obrazów widzianych podczas treningu	35
4	Rekonstrukcje obrazów niewidzianych podczas treningu	36
5	Średnie wyniki dokładności KNN dla ResNet50	42
6	Średnie wyniki dokładności metody centroidów dla ResNet50	42

Spis załączników

1	Mushroom species dataset	61
---	--------------------------------	----

Załącznik 1

Mushroom species dataset

Zbiór danych **Mushroom Species Dataset** [5] został pobrany z platformy **iNaturalist**¹. Zawiera zdjęcia przedstawiające 100 różnych gatunków grzybów i porostów, zebranych w Federacji Rosyjskiej. Dane zostały oznaczone tagiem *Fungi Including Lichens*.

Zbiór składa się z obrazów o różnorodnych rozdzielczościach, które przedstawiają gatunki w ich naturalnym środowisku. Poniżej znajduje się pełna lista gatunków:

1. *Amanita citrina*
2. *Amanita muscaria*
3. *Amanita pantherina*
4. *Amanita rubescens*
5. *Apioperdon pyriforme*
6. *Armillaria borealis*
7. *Artomyces pyxidatus*
8. *Bjerkandera adusta*
9. *Boletus edulis*
10. *Boletus reticulatus*
11. *Calocera viscosa*
12. *Calycina citrina*
13. *Cantharellus cibarius*
14. *Cerioporus squamosus*
15. *Cetraria islandica*
16. *Chlorociboria aeruginascens*
17. *Chondrostereum purpureum*
18. *Cladonia fimbriata*
19. *Cladonia rangiferina*

¹iNaturalist - <https://www.inaturalist.org/>

Spis tabel

20. *Cladonia stellaris*
21. *Clitocybe nebularis*
22. *Coltricia perennis*
23. *Coprinellus disseminatus*
24. *Coprinellus micaceus*
25. *Coprinopsis atramentaria*
26. *Coprinus comatus*
27. *Crucibulum laeve*
28. *Daedaleopsis confragosa*
29. *Daedaleopsis tricolor*
30. *Evernia mesomorpha*
31. *Evernia prunastri*
32. *Flammulina velutipes*
33. *Fomes fomentarius*
34. *Fomitopsis betulina*
35. *Fomitopsis pinicola*
36. *Ganoderma applanatum*
37. *Graphis scripta*
38. *Gyromitra esculenta*
39. *Gyromitra gigas*
40. *Gyromitra infula*
41. *Hericium coralloides*
42. *Hygrophoropsis aurantiaca*
43. *Hypholoma fasciculare*
44. *Hypholoma lateritium*
45. *Hypogymnia physodes*
46. *Imleria badia*

47. *Inonotus obliquus*
48. *Kuehneromyces mutabilis*
49. *Lactarius deliciosus*
50. *Lactarius torminosus*
51. *Lactarius turpis*
52. *Laetiporus sulphureus*
53. *Leccinum albostipitatum*
54. *Leccinum aurantiacum*
55. *Leccinum scabrum*
56. *Leccinum versipelle*
57. *Lepista nuda*
58. *Lobaria pulmonaria*
59. *Lycoperdon perlatum*
60. *Macrolepiota procera*
61. *Merulius tremellosus*
62. *Mutinus ravenelii*
63. *Nectria cinnabarina*
64. *Panellus stipticus*
65. *Parmelia sulcata*
66. *Paxillus involutus*
67. *Peltigera aphthosa*
68. *Peltigera praetextata*
69. *Phaeophyscia orbicularis*
70. *Phallus impudicus*
71. *Phellinus igniarius*
72. *Phellinus tremulae*
73. *Phlebia radiata*

Spis tabel

74. *Pholiota aurivella*
75. *Pholiota squarrosa*
76. *Physcia adscendens*
77. *Platismatia glauca*
78. *Pleurotus ostreatus*
79. *Pleurotus pulmonarius*
80. *Pseudevernia furfuracea*
81. *Rhytisma acerinum*
82. *Sarcomyxa serotina*
83. *Sarcoscypha austriaca*
84. *Sarcosoma globosum*
85. *Schizophyllum commune*
86. *Stereum hirsutum*
87. *Stropharia aeruginosa*
88. *Suillus granulatus*
89. *Suillus grevillei*
90. *Suillus luteus*
91. *Trametes hirsuta*
92. *Trametes ochracea*
93. *Trametes versicolor*
94. *Tremella mesenterica*
95. *Trichaptum biforme*
96. *Tricholomopsis rutilans*
97. *Urnula craterium*
98. *Verpa bohemica*
99. *Vulpicida pinastri*
100. *Xanthoria parietina*