



**POLITECHNIKA  
RZESZOWSKA**  
im. IGNACEGO ŁUKASIEWICZA

Politechnika Rzeszowska im. Ignacego Łukasiewicza  
Wydział Matematyki i Fizyki Stosowanej

## **PRACA MAGISTERSKA**

kierunek studiów: inżynieria i analiza danych

## **ROZPOZNAWANIE RYSUNKÓW GRAFÓW**

Gabriel Lichacz

Promotor  
dr dr Paweł Bednarz

Rzeszów 2024



# Spis treści

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Podstawowe definicje teorii grafów</b>                           | <b>11</b> |
| <b>2</b> | <b>Uczenie maszynowe</b>  | <b>13</b> |
| 2.1.     | Podstawowe pojęcia z zakresu uczenia maszynowego . . . . .          | 13        |
| 2.2.     | Warstwy w modelach sieci neuronowych . . . . .                      | 17        |
| 2.3.     | Rodzaje uczenia maszynowego . . . . .                               | 18        |
| 2.4.     | Proces uczenia maszynowego . . . . .                                | 19        |
| <b>3</b> | <b>Wykorzystywane technologie</b>                                   | <b>21</b> |
| 3.1.     | Język R . . . . .   | 21        |
| 3.2.     | Język Python . . . . .  | 21        |
| 3.3.     | Stanowisko pracy . . . . .  | 22        |
| <b>4</b> | <b>Opis modelu</b>  | <b>23</b> |
| 4.1.     | Generacja danych . . . . .  | 24        |
| 4.2.     | Dane zewnętrzne . . . . .   | 26        |
| 4.3.     | Opis ogólny skryptu . . . . .                                       | 27        |
| <b>5</b> | <b>Testy</b>  | <b>31</b> |
| 5.1.     | Model podstawowy . . . . .  | 31        |
| 5.2.     | Model z walidacją krzyżową . . . . .                                | 35        |
| 5.3.     | Model ze zmienną liczbą wierzchołków . . . . .                      | 53        |
| 5.4.     | Model ze zmienną liczbą wierzchołków i walidacją krzyżową . . . . . | 60        |
| 5.5.     | Modyfikacje modelu o najlepszej dokładności realnej . . . . .       | 68        |
|          | Literatura . . . . .  | 76        |
|          | Załączniki . . . . .  | 78        |



# Wykaz symboli

$G$  - graf

$V(G)$  - zbiór wierzchołków grafu  $G$

$E(G)$  - zbiór krawędzi grafu  $G$

$C_n$  - cykl  $n$ -wierzchołkowy

$D$  - digraf

$K_n$  - graf pełny

$N_n$  - graf bezkrawędziowy

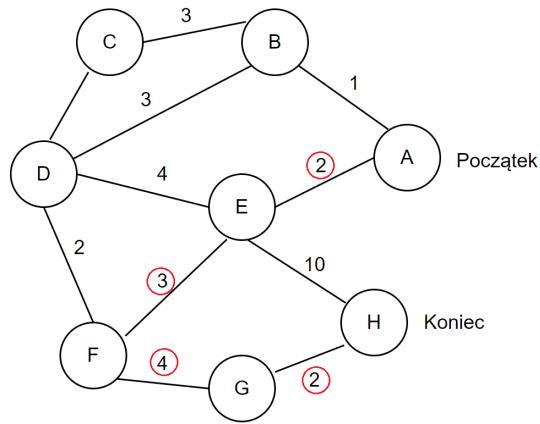
$P_n$  - ścieżka  $n$ -wierzchołkowa

# Wstęp

Graf definiuje się jako pewną parę uporządkowaną  $G = (V, E)$ , gdzie  $V$  to zbiór wierzchołków, a  $E$  to zbiór krawędzi. Takie obiekty można przedstawić graficznie jako reprezentację danych, w której wartości są przedstawione w pewien uporządkowany sposób, zwykle w relacji do siebie nawzajem. „*Stanowią wygodny aparat do modelowania różnych obiektów, (...) i odpowiednio interpretowane - mogą zawierać pewne informacje*”[15].

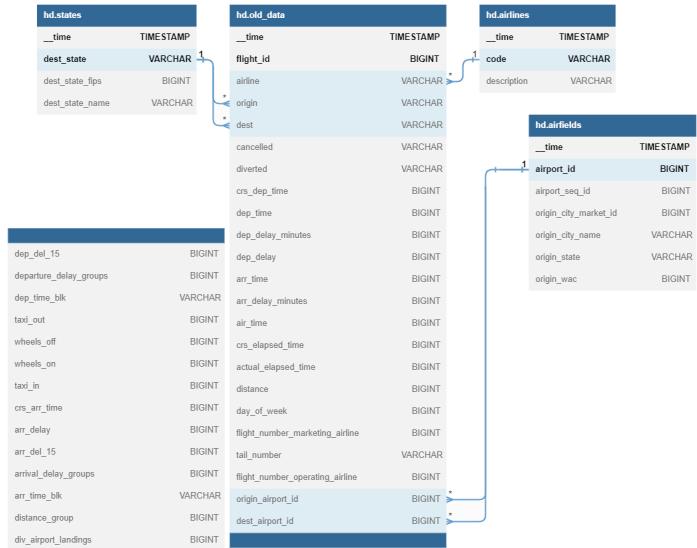
Teoria grafów to dziedzina matematyki zajmująca się badaniem właściwości grafów, będąca bardzo ważnym narzędziem w wielu „*dziedzinach od rachunku operacyjnego, chemii, po genetykę, lingwistykę oraz od elektroniki i geografii po socjologię i architekturę*”[14]. Grafy dają możliwość zobrazowania pewnych modeli, co jest szczególnie korzystne w analizie wzorców. W kontekście grafów warto podkreślić ich zastosowanie poza teoretycznymi analizami.

W dziedzinach informatycznych, grafy stanowią fundament wielu algorytmów, takich jak algorytmy przeszukiwania, algorytmy najkrótszej drogi, drzew rozpinających, czy modeli sieci. Przykładem może być tutaj wyszukiwanie najkrótszej trasy, chociażby w nawigacji GPS, gdzie wierzchołki odpowiadają skrzyżowaniom, a krawędzie drogom. W przypadku znajdowania najbardziej optymalnych tras, warto wymienić takie algorytmy jak  $A^*$ , Bellmana-Forda czy Dijkstry.



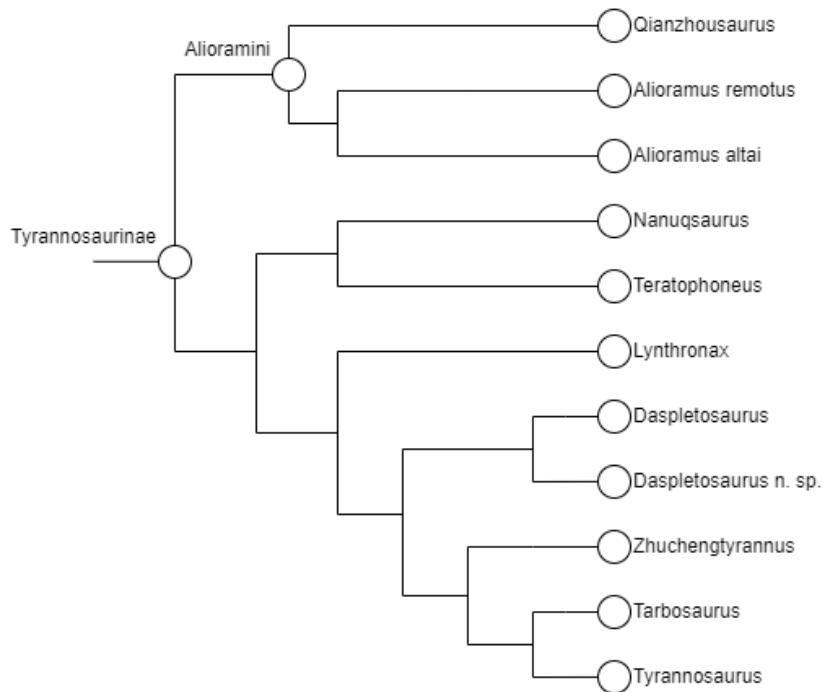
Rysunek 0.1: Przykład grafu z wyznaczoną najkrótszą drogą od wierzchołka  $A$  do  $H$ . Zaznaczona została czeronymi okręgami. Liczby przy krawędziach grafu oznaczają odległość między wierzchołkami łączonymi daną krawędzią.

Grafy istotną rolę odgrywają w reprezentacji i modelowaniu struktur danych, takich jak bazy danych. Najczęściej stosowane bazy danych, tj. relacyjne, zbudowane są w sposób, który grafy mogą doskonale zobrazować - wierzchołki odpowiadają kolumnom w tabelach a połączenia między nimi to krawędzie, reprezentujące relacje.



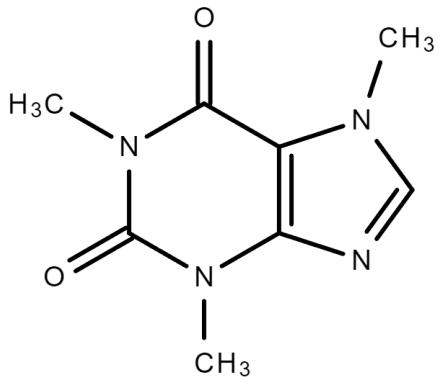
Rysunek 0.2: Przykładowy schemat relacyjnej bazy danych. Źródło: opracowanie własne na podstawie danych: <https://www.kaggle.com/datasets/robikscube/flight-delay-dataset-20182022>

W biologii grafy pełnią ważną rolę w modelowaniu układu nerwowego, sieci białek, szlaków metabolicznych oraz interakcji między genami. W genetyce wykorzystuje się je między innymi do analizy drzew filogenetycznych, co pozwala chociażby na śledzenie relacji ewolucyjnych między organizmami, z wierzchołkami stopnia 1 reprezentującymi żywe organizmy, a wierzchołkami pośrednimi jako ich wspólnymi przodkami [5].



Rysunek 0.3: Filogenetyczne relacje podrodzin dinozaurów rodzyiny Tyrannosaurinae.  
 Źródło: opracowanie własne na podstawie: Brusatte, S., Carr, T. The phylogeny and evolutionary history of tyrannosauroid dinosaurs. Sci Rep 6, 20252 (2016)

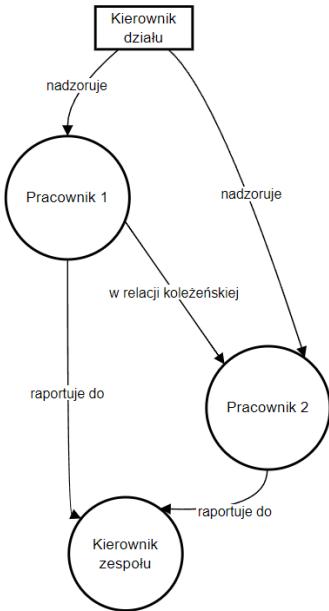
Natomiast w chemii grafy służą do reprezentacji struktury molekularnej związków chemicznych, umożliwiając naukowcom analizę ich właściwości i reaktywności. Znaczenie teorii grafów dla chemii wynika głównie z istnienia zjawiska izomeryzmu, które jest uzasadnione przez teorię struktury chemicznej. Wszystkie wzory strukturalne związków o wiązaniach kowalencyjnych są grafami, które nazywane są grafami molekularnymi [2].



Rysunek 0.4: Struktura molekularna kofeiny Źródło: opracowanie własne na podstawie:  
Rodak K., Kokot I., Kratz E.W.: Caffeine as a Factor Influencing the Functioning of  
the Human Body-Friend or Foe? Nutrients. 2021 Sep 2;13(9):3088

W lingwistyce, przy pomocy grafów możliwe jest modelowanie struktury języka, analiza morfologiczna czy syntaktyczna. Stosowane są również przez wiele innych dziedzin, takich jak gramatyka generatywna, będąca kandydatem na teoretyczną podstawę biolingwistyki. Przez ponad pół wieku, wykorzystywała ona notację drzewa jako pomocnicze narzędzie do wyrażania struktur językowych, lecz takie podejście zostało ostatecznie podważone przez jednego z autorytetów w dziedzinie lingwistyki - Noama Chomsky'ego [1]. Dzięki grafom, możliwe jest również lepsze zrozumienie i przetwarzanie języka naturalnego przez komputery, co stanowi podstawę technologii takich jak tłumaczenie automatyczne czy rozpoznawanie mowy.

Teoria grafów znajduje także zastosowanie w analizie sieci społecznych, gdzie pomagają w badaniu relacji między ludźmi. Wielu psychologów i socjologów zajmuje się tematem struktur wynikających z relacji między różnymi podmiotami. Przykładami takich zależności mogą być sieci komunikacyjne między ludźmi, relacje dominacji i uległości w grupie, wpływ lub władza jednych podmiotów nad innymi, czy relacje między różnymi aspektami pola psychologicznego danej osoby lub jej osobowości [9]. Bardzo dużym polem jest również analiza mediów społecznościowych, które to wpływają coraz bardziej na przeciętnego człowieka. Poprzez gromadzenie i analizę danych dotyczących połączeń między użytkownikami, wzorców interakcji i zachowań komunikacyjnych, analiza mediów społecznościowych pozwala zauważać pewne struktury społeczne i zidentyfikować wzorce leżące u podstaw interakcji w ich obrębie. Wszystko to możliwe jest do modelowania za pomocą struktur znanych z teorii grafów [12].



Rysunek 0.5: Przykład sieci relacji pomiędzy pracownikami w dziale danej firmy

Rozpoznawanie wzorców, nam ludziom, pozwala na szybszą naukę przez rozpoznawanie czegoś, co już wcześniej widzieliśmy. W bardzo dużym uproszczeniu, algorytmy uczenia maszynowego działają w podobny sposób. Gdy model zostanie prawidłowo nauczony na pewnych danych, jest w stanie rozpoznawać podobne wzorce w innych, nigdy wcześniej nie widzianych.

Podsumowując, grafy są niezwykle wszechstronnym narzędziem, które znajduje zastosowanie w bardzo wielu dziedzinach nauki i technologii. Ich zdolność do reprezentowania skomplikowanych struktur i relacji w sposób zrozumiały, przystępny i czytelny jest nieoceniona. Dzięki nim możliwe jest również analizowanie i przetwarzanie informacji w efektywniejszy sposób, niż informacji nieustrukturyzowanych.

Celem pracy jest zobrazowanie owej zależności, na przykładzie nauczenia sieci neuronowej, w taki sposób, by po wytrenowaniu na kilku typach grafów stworzonych sztucznie, model był w stanie rozpoznać dane wzorce i je nazwać, w przestrzeni rzeczywistej.

# Rozdział 1

## Podstawowe definicje teorii grafów

W tym rozdziale zostaną przedstawione podstawowe definicje z dziedziny teorii grafów. Definicje zostały zaczerpnięte z literatury, z pozycji [15], [14] oraz [16].

**Definicja 1.** Grafem nieskierowanym, skończonym  $G$  nazywamy parę  $(V, E)$ , gdzie  $V = V(G)$  jest zbiorem skończonym, niepustym, natomiast  $E = E(G)$  jest rodziną mogących się powtarzać dwuelementowych podzbiorów niekoniecznie różnych elementów ze zbioru  $V$ . Zbiór  $V(G)$  nazywamy zbiorem wierzchołków, a elementy tego zbioru nazywamy wierzchołkami i oznaczamy symbolami:  $x, y, x_i, y_i, 1, 2, \dots$ . Zbiór  $E(G)$  nazywamy zbiorem krawędzi grafu  $G$ . Mówimy, że krawędź  $\{v, w\}$  łączy wierzchołki  $v$  i  $w$ , i na ogólnie oznaczamy ją krócej symbolem  $vw$ . W wielu zagadnieniach nazwy wierzchołków są nieistotne, więc je pomijamy i mówimy wtedy, że graf jest nieoznakowany.

**Definicja 2.** Jeżeli w grafie  $G$  istnieją co najmniej dwie krawędzie  $xy$ , to krawędź tę nazywamy krawędzią wielokrotną.

**Definicja 3.** Krawędź  $xx$  w grafie  $G$  nazywamy pętlą.

**Definicja 4.** Graf mający krawędzie wielokrotne nazywamy multigrafem.

**Definicja 5.** Graf, który nie ma krawędzi wielokrotnych i pętli, nazywamy grafem prostym.

**Definicja 6.** Graf zawierający pętle nazywamy pseudografem.

**Definicja 7.** Drogę  $P$  z wierzchołka  $x_1$  do wierzchołka  $x_m$  w grafie  $G$  nazywamy skończony ciąg wierzchołków  $x_1, x_2, \dots, x_m, m \geq 2$  i krawędzi  $x_i, x_{i+1}, i = 1, \dots, m$ .

**Definicja 8.** Grafem spójnym nazywamy graf  $G$ , w którym każde dwa wierzchołki są połączone drogą dowolnej długości. Graf, który nie jest spójny, nazywamy grafem niespójnym.

**Definicja 9.** Dwa wierzchołki  $x, y$  w grafie  $G$  są sąsiednie, jeżeli  $xy \in E(G)$ . Mówimy wtedy, że wierzchołki  $x$  i  $y$  są incydentne z tą krawędzią.

**Definicja 10.** Stopień wierzchołka  $v$  oznaczany symbolem  $\deg(v)$  jest liczbą krawędzi incydentnych z  $v$ .

**Definicja 11.** Wierzchołek stopnia 0 nazywamy wierzchołkiem izolowanym, a wierzchołek stopnia 1, liściem.

**Definicja 12.** Graf  $G$  taki, że  $E(G) = \emptyset$ , nazywamy grafem bezkrawędziowym. Jeżeli  $|V(G)| = n$ , to graf bezkrawędziowy oznaczony symbolem  $N_n$ . Każdy wierzchołek grafu bezkrawędziowego jest wierzchołkiem izolowanym.

**Definicja 13.** Graf prosty  $G$  taki, że każde dwa wierzchołki są sąsiednie, nazywamy grafem pełnym. Jeżeli  $|V(G)| = n$ , to graf pełny oznaczamy  $K_n$ .

**Definicja 14.** Graf skierowany - graf niezawierający krawędzi nieorientowanych.

**Definicja 15.**

**Definicja 16.**

**Definicja 17.**

**Definicja 18.** Drzewem nazywany spójny graf bez cykli. Korzeń w drzewie jest jedynym wierzchołkiem, który nie ma przodka, wszystkie wierzchołki sąsiednie z korzeniem są jego potomkami.

**Definicja 19.** Drzewem binarnym nazywamy drzewo składające się z wyróżnionego wierzchołka nazywanego korzeniem oraz dwóch poddrzew binarnych - lewgo  $T_l$  oraz prawego  $T_p$ .

# Rozdział 2

## Uczenie maszynowe

Uczenie maszynowe, znane również jako machine learning, to specjalistyczna gałąź sztucznej inteligencji, która koncentruje się na konstruowaniu modeli i algorytmów umożliwiających komputerom samodzielne uczenie się z dostępnych danych. W przeciwieństwie do systemów, które są bezpośrednio programowane do wykonania określonych zadań, systemy uczenia maszynowego analizują dane, rozpoznają wzorce i podejmują decyzje oparte na zdobytej w ten sposób wiedzy.

### 2.1. Podstawowe pojęcia z zakresu uczenia maszynowego

Definicje zostały zaczerpnięte z literatury, z pozycji [6], [7], [11] , [8] oraz [13].

**Definicja 20.** Zbiór treningowy - zbiór danych, który jest używany do trenowania modelu uczenia maszynowego.

**Definicja 21.** Zbiór walidacyjny - zbiór danych, który jest używany do sprawdzenia wydajności modelu uczenia maszynowego.

**Definicja 22.** Zbiór testowy - zbiór danych używany do oceny wydajności modelu uczenia maszynowego po przeszkoleniu go na zbiorze treningowym i ocenie na zbiorze walidacyjnym.

**Definicja 23.** Klasyfikacja to proces polegający na przypisaniu obiektów do wcześniej zdefiniowanych klas na podstawie ich cech.

**Definicja 24.** Regresja liniowa - metoda, w której model liniowy przewiduje wyniki na podstawie ważonej sumy cech wejściowych oraz stałej, nazywanej punktem obciążenia lub punktem przecięcia.

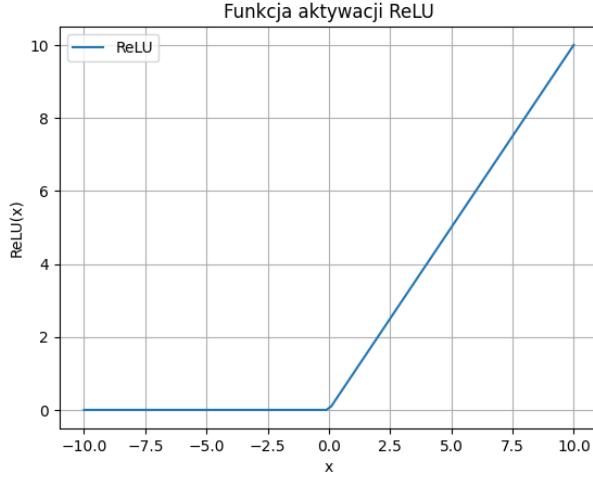
**Definicja 25.** Walidacja krzyżowa to proces, w którym dane dzielone są na kilka czę-

ści (przyjmujemy  $k$ ) zwanych „złożeniami” (lub „foldami” - stąd nazwa  $k$ -Fold Cross-Validation). Model jest trenowany na  $k - 1$  złożeniu, a testowany na pozostałym z nich. Proces ten jest powtarzany  $k$  razy, za każdym razem używając innego złożenia do testowania, a pozostałych do treningu. Jeżeli różnica w wydajności jest znacząca, uzasadniony jest sceptyzm odnośnie pojedynczych wyników pomiaru wydajności systemu. Z drugiej strony, jeżeli wszystkie wyniki są podobne, można mieć dużą dozę pewności, że niezależnie od konkretnego podziału na dane testowe i treningowe, wydajność systemu będzie podobna. Końcowa ocena modelu jest uzyskiwana poprzez uśrednienie wyników z każdej iteracji.

**Definicja 26.** Bias (błąd obciążenia) to błąd wynikający z niepoprawnych założeń w procesie uczenia maszynowego. Oznacza różnicę między przewidywaną wartością modelu a rzeczywistą wartością.

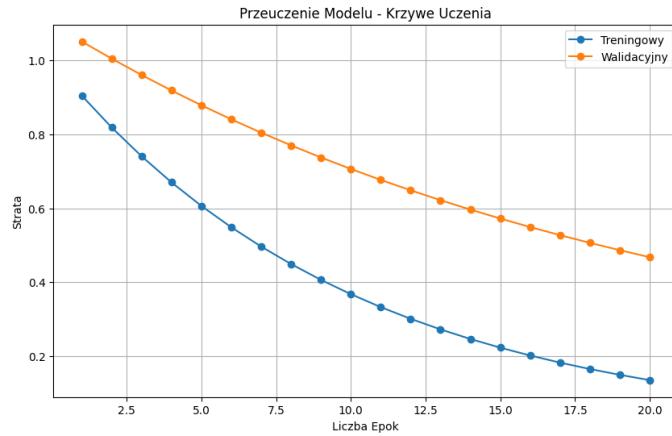
**Definicja 27.** Funkcja aktywacji w sieci neuronowej to nieliniowa funkcja matematyczna, która umożliwia sieci modelowanie złożonych zależności. Jej rolą jest przekształcanie sumy ważonej wejść i wartości biasu, co pozwala na tworzenie nielinowych funkcji wyjściowych. Równanie matematyczne opisujące działanie sieci neuronowej ma postać:  $Y' = g(W_o + X^T * W)$ , gdzie:  $Y'$  to przewidywana wartość wyjściowa,  $W_o$  to wartość bias,  $X^T$  to transpozycja macierzy wejściowej  $X$ ,  $W$  to przypisane wag, a  $g$  to funkcja aktywacji.

**Definicja 28.** Funkcja ReLU (ang. Rectified Linear Unit) - funkcja aktywacji. Jest ciągła, ale nieróżniczkowalna w punkcie  $z = 0$ , a jej pochodna dla  $z < 0$  wynosi 0. Spisuje się bardzo dobrze w modelowaniu złożonych funkcji, a dodatkowym aututem jest jej szybkość przetwarzania. Nie ma maksymalnej wartości wyjściowej. Funkcja ReLU definiowana jest wzorem:  $g(x) = \max(0, x)$



Rysunek 1.1: Funkcja aktywacji ReLU. Źródło: opracowanie własne na podstawie: Géron A.: Uczenie maszynowe z użyciem Scikit-Learn i TensorFlow. Helion SA, Gliwice 2020.

**Definicja 29.** Przeuczenie to sytuacja, w której algorytm dopasowuje się zbyt dokładnie do danych treningowych, co prowadzi do modelu, który nie potrafi dokładnie prognozować ani wnioskować na podstawie nowych danych spoza zbioru treningowego.



Rysunek 1.2: Zobrazowane przeuczenie modelu

**Definicja 30.** Regularizacja to technika stosowana w celu zapobiegania przeuczeniu modelu. Działa poprzez dodanie kary do funkcji kosztu, co penalizuje zbyt złożone modele.

**Definicja 31.** Regularizacja L2 służy do ograniczania wag sieci neuronowych, natomiast regularizacja L1 przydaje się do tworzenia modeli rzadkich (w których wiele wag ma wartość równą 0). Zazwyczaj powinno się stosować ten sam typ regularyzatora

we wszystkich wartościach sieci.

**Definicja 32.** Epoka to pełny cykl przez cały zbiór danych treningowych, w której model przetwarza wszystkie dostępne dane treningowe. Liczba epok określa ile razy model przejdzie przez cały zbiór danych treningowych.

**Definicja 33.** Dokładność modelu to stosunek oznaczonych prawidłowo wartości do przykładów sklasyfikowanych nieprawidłowo.

**Definicja 34.** Macierz pomyłek w sieci neuronowej to tabela służąca do oceny jakości modelu klasyfikacyjnego. Przedstawia liczbę przypadków, w których próbki z klasy A zostały zaklasyfikowane jako należące do klasy B. Aby utworzyć taką macierz, należy uzyskać zbiór prognoz, które porównywane są z rzeczywistymi wartościami docelowymi.

**Definicja 35.** Strata modelu to wartość, która wskazuje, jak bardzo prognozy modelu różnią się od rzeczywistych wartości dla pojedynczych przykładów. Idealnie przewidziane wartości mają stratę równą零, natomiast im większa różnica między prognozami a rzeczywistością, tym wyższa jest strata.

**Definicja 36.** Algorytm t-SNE (t-Distributed Stochastic Neighbor Embedding) to technika redukcji wymiarowości, która szczególnie dobrze nadaje się do wizualizacji wielowymiarowych zbiorów danych. Można ją zaimplementować przy użyciu aproksymacji Barnesa-Huta, co umożliwia stosowanie jej na dużych, rzeczywistych zbiorach danych.

**Definicja 37.** Entropia krzyżowa pomiędzy dwoma rozkładami prawdopodobieństwa  $p$  i  $q$  jest definiowana wzorem  $H(p, q) = -\sum p(x) \log q(x)$  (przynajmniej wtedy, gdy obydwa rozkłady są dyskretne).

**Definicja 38.** Konwergencja modelu to stopniowe poprawianie się wydajności podczas trwania procesu uczenia, aż do osiągnięcia stabilnego poziomu. Według założeń, po osiągnięciu takiego poziomu, kolejne treningi nie powinny przynosić dalszych korzyści lub owe korzyści są pomijalnie małe.

**Definicja 39.** Normalizacja wsadowa (ang. batch normalization) to operacja pozwalająca modelowi określić optymalną skalę i średnią danych wejściowych dla każdej warstwy. Z tego też powodu stosowana jest przed funkcją aktywacji w każdej warstwie. W jej wyniku, dane wejściowe zostają wyśrodkowane i znormalizowane,

**Definicja 40.** Caching danych przyspiesza naukę modelu. Proces ten umieszcza zawartość danych w pamięci podręcznej GPU. Dzięki temu każdy przykład ze zbioru

danych jest odczytywany i przetwarzany tylko raz, a nie jak domyslnie - raz na epokę.

**Definicja 41.** Pobieranie wstępne (ang. prefetching) to metoda, w której podczas przetwarzania przez algorytm uczenia jednej grupy, zestaw danych równolegle przygotowuje kolejną grupę.

## 2.2. Warstwy w modelach sieci neuronowych

Głębokie sieci neuronowe, opierają się na strukturze warstw, które przekształcają i analizują dane wejściowe, aby uzyskać z nich pożądane cechy. Każda z tych warstw pełni pewną, specyfczną funkcję, począwszy od przeskalowania danych, przez konwolucje (przekształcenie macierzowe danych), aż po bardziej zaawansowane operacje, umożliwiając modelowi sprawniejsze uczenie się bardziej złożonych wzorców.

- Warstwa Flatten (spłaszczona) ma za zadanie przekształcić każdy obraz wejściowy w tablicę jednowymiarową. Nie zawiera żadnych parametrów, a jej jedynym celem jest proste, wstępne przetworzenie danych [7].
- Warstwa Dense (w pełni połączona) zarządza samodzielnie swoją macierzą wag, zawierającą wszystkie wagi połączeń między neuronami, a wejściami do nich oraz wekorem obciążień. Zawiera najczęściej bardzo dużo parametrów, dzięki czemu model uzyskuje swobodę w dopasowaniu do danych treningowych. Jednocześnie, grozi mu również przez to ryzyko przetrenowania, zwłaszcza w przypadku korzystania z mniejszych zestawów danych [7].
- Warstwa Rescaling mnoży każde wejście przez ustalony współczynnik skalujący. Zastosowanie tej techniki jest przydatne, gdy różne cechy danych wejściowych mają różne zakresy wartości. Poprzez jednolite skalowanie, model może efektywniej uczyć się wzorców, a proces optymalizacji staje się stabilniejszy [6].
- Warstwa Conv2D tworzy jądro splotu, które jest nakładane na dane wejściowe w jednym wymiarze przestrzennym (lub czasowym), aby wygenerować tensor danych wyjściowych. Dodatkowo, jeśli stosowana jest funkcja aktywacji, jest ona stosowana również do danych wyjściowych [7].
- Warstwa MaxPooling2D dokonuje redukcji wymiarów danych wejściowych wzdłuż ich wymiarów przestrzennych (wysokości i szerokości), wybierając maksymalną wartość z każdego okna o rozmiarze określonym przez wybrany współczynnik *pool\_size*,

dla każdego kanału danych wejściowych. Okno to jest przesuwane o określoną liczbę kroków wzdłuż obu wymiarów [7].

- Warstwa Dropout losowo zeruje jednostki wejściowe z prawdopodobieństwem określonym przez wybrany współczynnik dropout na każdym etapie treningu, co pomaga unikać przeuczenia modelu. Jednostki, które nie zostały wyzerowane, są skalowane w górę przez mnożenie przez  $\frac{1}{1-współczynnikDropout}$ , aby suma wartości wejściowych pozostała niezmieniona [7].

## 2.3. Rodzaje uczenia maszynowego

Według [7], uczenie maszynowe można sklasyfikować na podstawie kilku kryteriów. Jest to nadzór człowieka w procesie trenowania, możliwość modelu do uczenia się w czasie rzeczywistym oraz sam sposób pracy (nauka z przykładów lub modelu). Kryteria te nie wykluczają się wzajemnie - można je dowolnie łączyć. Za przykład może posłużyć filtr antyspamowy, który ciągle się uczy, wykorzystując model sieci neuronowej i analizując wiadomości email. Taki system można określić przyrostowym, opartym na modelu i nadzorowanym.

Dodatkowe kryteria oceny rodzaju uczenia maszynowego:

- Uczenie nadzorowane (ang. supervised learning) to podejście, w którym model jest szkolony na danych, które są już odpowiednio oznaczone (np. rekordy mają przypisane odpowiednie klasy). Celem jest odkrycie funkcji, która przekształca dane wejściowe w oczekiwane wyjścia. Znajduje zastosowanie w klasyfikacji i regresji. Przykład: Klasyfikacja wiadomości e-mail jako spam lub nie-spam.
- Uczenie nienadzorowane (ang. unsupervised learning) - w tym przypadku model bada nieoznaczone dane, aby odkryć pewne wzorce lub struktury. Najczęściej stosowane w klasteryzacji, czy redukcji wymiarowości. Przykłady:
  - \* Klasteryzacja klientów w celu segmentacji rynku, gdzie klienci są grupowani na podstawie ich zachowań zakupowych.
  - \* Redukcja wymiarowości w celu wizualizacji danych wysokowymiarowych, np. za pomocą algorytmu t-SNE.

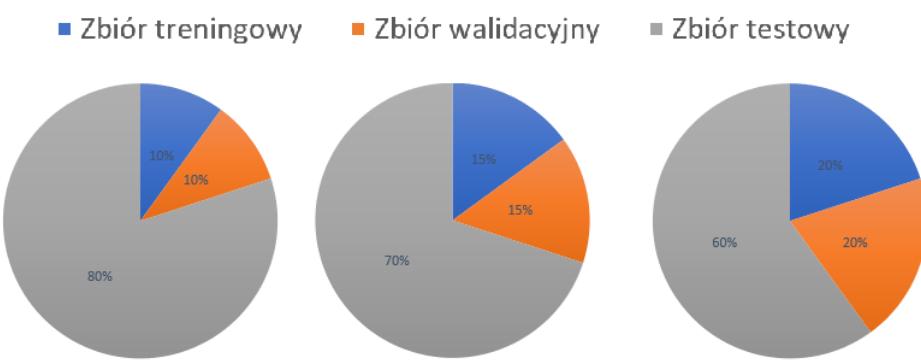
- Uczenie przez wzmacnianie (ang. reinforcement learning) to przypadek, gdzie model uczy się poprzez interakcję ze swoim otoczeniem, podejmując decyzje, które maksymalizują pewną nagrodę. Przykłady:
  - \* Algorytmy sterujące robotami, które uczą się poruszać w nieznanym terenie.
  - \* Programy grające w gry, takie jak AlphaGo (chińska gra Go), które uczą się strategii gry poprzez rozgrywanie wielu partii.

## 2.4. Proces uczenia maszynowego

Proces uczenia maszynowego można podzielić na kilka etapów, które są niezbędne do stworzenia skutecznego modelu zdolnego do samodzielnej nauki na podstawie zebranych danych.

Należy rozpocząć od zgromadzenia danych z odpowiednich źródeł. Mogą obejmować bazy danych, API, pliki CSV, czujniki, logi systemowe czy nawet wpisy z mediów społecznościowych. Dane mogą być ustrukturyzowane (np. tabele w bazach danych) lub nieustrukturyzowane (np. obrazy, tekst).

Dalej, konieczne jest przygotowanie danych do odpowiedniego formatu. Obejmuje to usunięcie brakujących, pustych oraz błędnych wartości, radzenie sobie z duplikatami i anomaliami, skalowanie cech, kodowanie zmiennych kategorycznych, normalizację danych oraz podzielenie danych na zbiory treningowe, walidacyjne i testowe.



Rysunek 4.3: Przykład podziału danych na zbiory treningowe, walidacyjne i testowe

Najczęściej, podział na zbiory dokonuje się w proporcjach 70-80% na trening, 10-15% na walidację i 10-15% na testy. Jest to zależne od specyfiki problemu, dlatego konieczne jest odpowiednie przygotowanie i zbadanie danych przed podjęciem decyzji.

Wybór modelu to proces, który zależy od rodzaju problemu (np. regresja, klasyfikacja, klasteryzacja) oraz charakterystyki danych, gdzie najpopularniejsze modele to drzewa decyzyjne, lasy losowe, maszyny wektorów nośnych (SVM), sieci neuronowe,  $k$ -najbliższych sąsiadów ( $k$ -NN) i regresja liniowa/logistyczna. Trenowanie modelu to kolejny etap, który polega na dostosowaniu parametrów modelu do danych treningowych, w tym dostosowaniu hiperparametrów modelu (parametrów, które nie są uczone, np. liczba warstw w sieci neuronowej) poprzez metodę walidacji krzyżowej lub inne techniki optymalizacji.

Ewaluacja obejmuje ocenę modelu za pomocą pewnych metryk, takich jak dokładność, precyzja, recall, F1-score, błąd średniokwadratowy (MSE), błąd absolutny (MAE), a także analizy wydajności modelu w przypadku klasyfikacji binarnej. Optymalizacja modelu to kolejny etap, który obejmuje dalsze dostosowanie hyperparametrów, wybór cech, które najbardziej wpływają na wynik modelu, próby różnych architektur modelu oraz zastosowanie technik takich jak L1, L2, dropout, które zapobiegają przeuczeniu modelu.

Implementacja modelu jest procesem, w którym wdrażany jest model w środowisku produkcyjnym. Zakłada ona przeprowadzenie integracji z aplikacjami zewnętrznymi, tworzenie API serwujących dane, zautomatyzowanie decyzji, czy też śledzenie wydajności modelu w czasie rzeczywistym, aby wykryć ewentualne pogorszenie jakości (drift danych) i regularne aktualizacje modelu. Aktualizacja i utrzymanie modelu to kolejny etap, który obejmuje regularne aktualizowanie modelu na podstawie nowych danych, aby utrzymać jego dokładność i skuteczność, ciągłe monitorowanie, aby zapewnić, że model działa zgodnie z oczekiwaniemi i nie występują niepożądane zachowania. Proces uczenia maszynowego jest iteracyjny i wymaga ciągłej interakcji między danymi, modelem i wynikami, aby osiągnąć optymalne rezultaty.

# Rozdział 3

## Wykorzystywane technologie

Praca opiera się na wykorzystaniu języka R oraz Python do generowania zbiorów danych, wszelkich manipulacji na nich oraz ich klasyfikacji.

### 3.1. Język R



Rysunek 1.1: Logo R [22]

Język R to szeroko stosowany w statystyce, analizie danych oraz naukach przyrodniczych język interpretowalny. Nie ma on skomplikowanej składni i jest przystosowany do bycia jak najbardziej przyjaznym dla nowego użytkownika. Oprócz dużych możliwości obliczeniowych, jest również świetnym narzędziem do wizualizacji danych, co spowodowało, że został wybrany do stworzenia zbioru danych. Grafy wygenerowane zostały przy pomocy biblioteki igraph w wersji 2.0.3. Jest to pakiet do tworzenia i analizy struktur sieci, a co za tym idzie oferuje bogaty wybór funkcji do generowania losowych i regularnych grafów oraz ich wizualizacji.

### 3.2. Język Python



Rysunek 2.2: Logo Python [21]

Język Python jest jednym z najpopularniejszych języków wysokopoziomowych ogólnego przeznaczenia. Zawdzięcza to swojej wszechstronności oraz prostocie składni. Znaczna liczba bibliotek pozwala na wykorzystywanie Pythona od prostych skryptów, przez analizę danych, aż po rozbudowane aplikacje, takie jak całe systemy największych gigantów technologicznych, np. Google. Język ten jest szeroko wykorzystywany w dziedzinie Data Science do wizualizacji, analizy i przetwarzania danych oraz w uczeniu maszynowym. Ostatnie z wymienionych zastosowań zadecydowało o wyborze języka Python jako narzędzia do stworzenia modelu klasyfikacji grafów. Wykorzystana została biblioteka Keras z pakietu Tensorflow.

### 3.3. Stanowisko pracy

Całość pracy, tj. generacja danych, modele oraz testy, została przygotowana na komputerze osobistym o parametrach:

- CPU: i5-10400F 2.9 GHz
- RAM: 32 GB 3200 MHz
- GPU: ADM Radeon RX 5600 XT 6GB
- Dysk: 2 x 1 TB HDD, 1 TB NVMe, 120 GB SSD
- System operacyjny: Windows 10

System nie posiada karty graficznej zoptymalizowanej pod zastosowania uczenia maszynowego. Biblioteki języka Python obsługują jednak karty graficzne AMD, co umożliwia pracę.

# Rozdział 4

## Opis modelu

Model uczenia maszynowego jaki został wykorzystany w testach to sieć neuronowa.

Do testów stworzone zostało kilka modeli sieci neuronowych, wytrenowanych na rysunkach grafów stworzonych za pomocą skryptów R. Implementacja została wykonana biblioteką TensorFlow oraz Keras w języku Python. Modele są w stanie rozpoznawać rysunki grafów i przypisywać im odpowiednie klasy. Celem było również przetestowanie modeli na rzeczywistych zdjęciach, zawierających wzorce przypominające grafy, bądź rysunkach grafów narysowanych ręcznie.

Klasy, których rozpoznawania uczony był model:

- Graf bezkrawędziowy
- Graf pełny
- Drzewo binarne
- Ścieżka
- Cykl

Stworzone zostały 4 typy modeli w dwóch wariantach - trenowany na grafach o stałej krzywiźnie krawędzi oraz o losowej krzywiźnie krawędzi:

- wytrenowany na danych ze stałą liczbą wierzchołków,
- wytrenowany na danych ze stałą liczbą wierzchołków oraz walidacją krzyżową,
- wytrenowany na danych ze zmienną liczbą wierzchołków,

- wytrenowany na danych ze zmienną liczbą wierzchołków oraz walidacją krzyżową.

## 4.1. Generacja danych

Dane wygenerowane zostały przy pomocy skryptu stworzonego w języku R oraz biblioteki igraph. Skrypt został zaprojektowany funkcyjnie, by osiągnąć możliwie największą automatyzację testów. Rysunki grafów tworzone były o wielkości 800x600 pikseli, na białym tle, z wierzchołkami w kolorze pomarańczowym, bez jakichkolwiek oznaczeń wierzchołków oraz zapisywane w odpowiednich katalogach, odpowiadających klasie grafu. Przygotowane zostały funkcje tworzące ścieżki, cykle, grafy pełne, grafy bezkrawędziowe oraz drzewa binarne. W każdej z funkcji możliwy jest wybór liczby generowanych grafów, liczba wierzchołków grafu oraz współczynnik odpowiadający za zakrzywienie krawędzi na rysunkach.

```

1  #' Rysuj graf
2  #
3  #' @param graph Graph - Graf do narysowania
4  #' @param pathName string - Sciezka
5  #' @param fileName string - Nazwa pliku
6  #' @param vertexNo int - liczba wierzcholkow
7  #' @param i int - Numer iteracji
8  #' @param plotCurve float
9  #' @return void
10 #
11 plotGraphHelper <- function(graph, pathName, fileName,
12   vertexNo, i, plotCurve)
13 {
14   path <- file.path(pathName, paste0(
15     fileName, "-", vertexNo, "-", i, ".png"
16   ))
17   png(path, width = 800, height = 600)
18   plot(graph, vertex.label = NA, edge.curved = plotCurve)
19   dev.off()
}

```

Listing 4.1: Listing skryptu rysującego grafy

```

1  #' Graf sciezka N wierzcholkow, nieskierowany
2  #
3  #' @param N int - liczba rysunkow
4  #' @param vertexNo int - liczba wierzcholkow
5  #' @return void
6  #
7  plotPaths <- function(N, vertexNo)
8  {
9    fileName <- 'path'
10   pathName <- createDir(vertexNo, fileName)
11   definition <- c()

```

```

12   for (index in 1:(vertexNo-1))
13   {
14     definition <- c(definition, index, index + 1)
15   }
16   definitionMatrix <- matrix(
17     definition, ncol = 2, byrow = TRUE
18   )
19
20   for (i in 1:N)
21   {
22     plotCurve <- generateGaussian(0.01, 0.99)
23     graph <- graph_from_edgelist(
24       definitionMatrix, directed = FALSE
25     )
26     E(graph)$weight <- runif(ecount(graph))
27     plotGraphHelper(
28       graph, pathName, fileName, vertexNo, i, plotCurve
29     )
30   }
31 }
```

Listing 4.2: Listing funkcji tworzącej ścieżkę

W testach wykorzystane zostały wszystkie wybrane typy grafów. Każdy z nich, czyli dana liczba wierzchołków i typ grafu, wygenerowany został w liczbie 500 sztuk. Warianty liczby wierzchołków generowanych grafów to 4, 5, 6 oraz 7 wierzchołków. W sumie daje to 10 tys. grafów na których uczone oraz testowane były modele. Testy zostały przeprowadzone na dwa sposoby - ze stałą krzywizną krawędzi, wynoszącą 0,3, oraz z losowym parametrem krzywizny krawędzi, mieszczącym się w przedziale od 0 do 1. Testy ze stałą krzywizną wierzchołków nie zostały przedstawione w dalszej części pracy, ze względu na ich niską wartość merytoryczną. Wyniki uzyskane z tych testów nie były zadowalające.



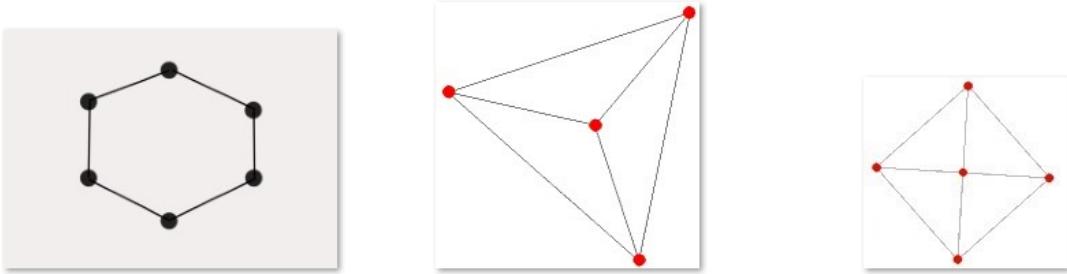
Rysunek 1.1: Przykładowe wygenerowane rysunki grafów z każdej klasy

## 4.2. Dane zewnętrzne

Obrazy testowe, które nazwane są tutaj danymi zewnętrznymi, są rysunkami grafów pochodzący spoza przygotowanego testu. Dzielą się na obrazy pobrane z internetu, obrazy wygenerowane przez skrypt w R, ale nie używane w treningu, oraz rysunki odręczne grafów. Typy danych zewnętrznych wybiegają poza klasy grafów wykorzystywanych przy uczeniu modelu.



Rysunek 2.2: Przykładowe zewnętrzne rysunki grafów narysowane odręcznie



Rysunek 2.3: Przykładowe zewnętrzne rysunki grafów pobrane z internetu

### 4.3. Opis ogólny skryptu

#### Przygotowanie

Wszystkie przygotowane skrypty testowe rozpoczynają się od przygotowania środowiska do trenowania modelu. Najpierw ustawiana jest ścieżka do katalogów z wygenerowanymi grafami oraz do katalogów na dane treningowe i walidacyjne. Następnie sprawdzane jest, czy te katalogi istnieją, a jeśli nie, są tworzone. Dalej, skrypty definiują parametry dotyczące wielkości obrazów oraz wielkości partii danych, które będą używane podczas treningu. Dla każdej wartości liczby wierzchołków ustawiana jest ścieżka do katalogu z wygenerowanymi grafami, pobierana lista podkatalogów oraz obrazów w każdym z nich. Następnie obrazy dzielone są na zestawy treningowe i walidacyjne w stosunku 80:20. Daje to 8 tys. grafów w zbiorze uczącym i 2 tys. grafów w zbiorze testowym. W przypadku modeli wykorzystujących wszystkie warianty liczby wierzchołków, dane przenoszone są do jednego katalogu i od razu dzielone na zbiory treningowe i walidacyjne.

#### Model

Każdy typ modelu tworzony jest w inny sposób. Opisana zostanie tu główna zasada i ich elementy wspólne. Na początku, skrypt wczytuje obrazy przygotowane na wcześniejszym etapie do odpowiednich zmiennych - treningowe i walidacyjne. W przypadku modeli z walidacją krzyżową, dla każdej iteracji walidacyjnej, dane zostały po-dzielone inaczej. Po wczytaniu danych, zostają one przeskalowane do wielkości 180x180 pikseli i przekształcone do odcieni szarości.

```

1 n_splits = 5
2 kfold = KFold(n_splits=n_splits, shuffle=True,
    random_state=42)

```

```

3     history = []
4     all_images = [os.path.join(dp, f) for dp, dn, filenames
5                   in os.walk(data_dir_model) for f in filenames if os.path
6                   .splitext(f)[1] == '.png']
7
8     for train_index, val_index in kfold.split(all_images):
9         train_images = [all_images[i] for i in train_index]
10        validation_images = [all_images[i] for i in val_index]
11
12        # Generowanie danych treningowych
13        train_ds = tf.keras.preprocessing.
14        image_dataset_from_directory(
15            train_dir,
16            image_size=(img_height, img_width),
17            batch_size=batch_size)
18
19        class_names = train_ds.class_names
20
21        train_ds = train_ds.map(lambda x, y: (rgb_to_grayscale(
22            x), y))
23
24        # Generowanie danych walidacyjnych
25        val_ds = tf.keras.preprocessing.
26        image_dataset_from_directory(
27            validation_dir,
28            image_size=(img_height, img_width),
29            batch_size=batch_size)
30
31        val_ds = val_ds.map(lambda x, y: (rgb_to_grayscale(x),
32            y))
33
34        # Tworzenie modelu
35        model = tf.keras.models.Sequential([
36            tf.keras.layers.Rescaling(1./255),
37            tf.keras.layers.Conv2D(32, 3, activation='relu'),
38            tf.keras.layers.MaxPooling2D(),
39            tf.keras.layers.Conv2D(32, 3, activation='relu'),
40            tf.keras.layers.MaxPooling2D(),
41            tf.keras.layers.Conv2D(32, 3, activation='relu'),
42            tf.keras.layers.MaxPooling2D(),
43            tf.keras.layers.Flatten(),
44            tf.keras.layers.Dense(128, activation='relu',
45            kernel_regularizer=tf.keras.regularizers.l2(0.01)),
46            tf.keras.layers.Dropout(0.2),
47            tf.keras.layers.Dense(len(class_names))])
48
49        # Kompilacja modelu
50        model.compile(
51            optimizer='adam',
52            loss=tf.losses.SparseCategoricalCrossentropy(
53                from_logits=True),
54            metrics=['accuracy'])
55    )

```

```

50 |     # Uczenie modelu
51 |     history.append(model.fit(
52 |         train_ds,
53 |         validation_data=val_ds,
54 |         epochs=75
55 |     ))

```

Listing 4.3: Listing skryptu tworzącego model z walidacją krzyżową oraz uczonym na wszystkich wariantach liczby wierzchołków grafów

Model sieci neuronowej został zdefiniowany jako sekwencyjny stos warstw. Dla standaryzacji danego testu, w przypadku modeli z walidacją krzyżową, ustalono  $k$ -Fold z liczbą podziałów równą 5. Pierwsza warstwa to warstwa Rescaling, która normalizuje wartości pikseli do zakresu [0, 1]. W przykładzie, parametr 1./255 oznacza, że każda wartość piksela mnożona jest przez  $\frac{1}{255}$ . Następne trzy warstwy to Conv2D, z których każda jest następowana warstwą MaxPooling2D. W przykładzie, warstwa kolwolucyjna stosuje 32 filtry o wymiarach 3x3 oraz funkcję aktywacji ReLU, która wprowadza nielinowość do modelu. MaxPooling2D redukuje rozmiar danych wejściowych, wybierając maksymalną wartość z każdego regionu (domyślnie oraz tutaj - 2x2). Po wyżej wymienionych warstwach, znajduje się warstwa Flatten, która przekształca mapy cech 2D w wektor 1D. Innymi słowy, przekształca wielowymiarową macierz wyjściową z poprzedniej warstwy do jednowymiarowego wektora. Następnie, dodana jest w pełni połączona (Dense) warstwa z 128 neuronami i funkcją aktywacji, podobnie jak w przypadku Conv2D, ReLU. Wprowadzona jest również regularyzacja L2, która dodaje karę za duże wartości wag, by zmniejszyć ryzyko przeuczenia. Została zastosowana z siłą 0,01. Kolejna warstwa to Dropout, która losowo wyłącza 20% neuronów podczas uczenia, co również jest metodą zapobiegającą przeuczeniu. Warstwa wyjściowa zawiera tyle jednostek, ile występuje klas w danych uczących. Zależnie od danego testu, może być to różna liczba. W przypadku warstw konwolucyjnych, wybrano 32 filtry, a dla warstwy w pełni połączonej zastosowano 128 jednostek. Liczba epok w podstawowej wersji modelu wyniosła 75.

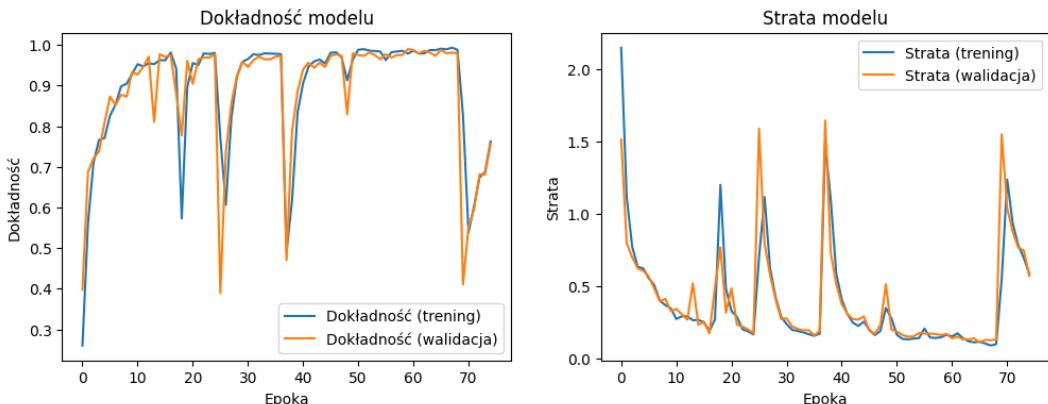
W kolejnych wariantach modeli, zmieniane były parametry poszczególnych warstw, funkcje aktywacji, czy również same warstwy, w celu znalezienia najbardziej optymalnej kombinacji.

## Wyniki

Po wytrenowaniu modelu, skrypt dokonuje wizualizacji dokładności i straty modelu. Najpierw wyświetla w konsoli wartości dokładności dla obu zbiorów z historii treningu. Dalej tworzy wykresy, gdzie na pierwszym z nich pokazuje dokładność na zbiorze treningowym i walidacyjnym, a na drugim wykresie prezentuje stratę modelu dla obu zbiorów. Jednostką straty jest entropia krzyżowa (cross-entropy), która jest wyrażana jako liczba bezwzględna. Entropia krzyżowa mierzy różnicę między rzeczywistymi etykietami a przewidywanymi prawdopodobieństwami klas. Im mniejsza wartość entropii krzyżowej, tym lepiej model przewiduje klasy. Dokładność jest wyrażana jako wartość procentowa lub ułamek, gdzie 1 oznacza 100% dokładności. Na przykład, jeśli model przewiduje poprawnie 90 na 100 przypadków, dokładność wynosi 0.9 lub 90%.

```
Dokładność na zbiorze treningowym: [0.23068182170391083, 0.3693181872367859, 0.7579545378684998, 0.824999988079071, 0.8602272868156433, 0.897727251
Dokładność na zbiorze walidacyjnym: [0.22727273404598236, 0.7477272748947144, 0.8659890995788574, 0.875, 0.909099361839294, 0.915909114997864, 0.]
```

Rysunek 3.4: Przykładowe wartości dokładności dla zbioru treningowego i walidacyjnego



Rysunek 3.5: Przykładowa wizualizacja dokładności i straty wytrenowanego modelu

## Testy na danych zewnętrznych

Po wyświetleniu dokładności modelu skrypt przeszukuje katalog z danymi i jego podkatalogi, by przygotować obrazy zewnętrzne. Następnie ustawia ścieżkę do katalogu z obrazami testowymi i pobiera ich listę. Dla każdego obrazu w tej liście wczytuje go, przeskaliwuje do odpowiedniego rozmiaru i konwertuje do skali szarości. Następnie model przewiduje klasę obrazu, a wynik jest wyświetlany w konsoli.

# Rozdział 5

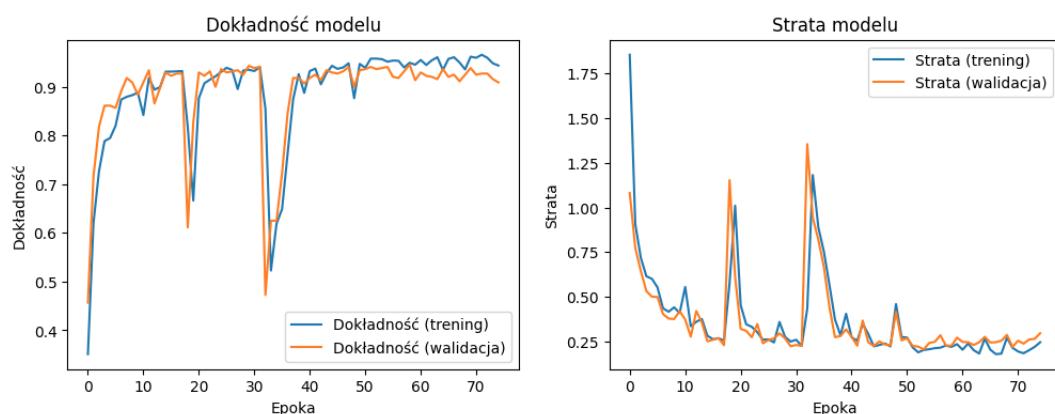
## Testy

### 5.1. Model podstawowy

#### Model podstawowy uczony na grafach czterowierzchołkowych

Dokładność modelu podstawowego, uczonego na grafach z czterema wierzchołkami stopniowo rośnie, zaczynając od około 40% i osiągając prawie 90% pod koniec procesu uczenia. Może to sugerować, że model dobrze uczy się na danych treningowych. Dokładność na danych walidacyjnych jest zbliżona do wcześniej przytoczonej. Wskazuje to, że model dobrze radzi sobie z generalizacją na nowych danych.

Strata na danych treningowych gwałtownie spada z około 1,75 do około 0,25 w ciągu pierwszych dziesięciu epok, po czym stabilizuje się. Wskazuje to na szybkie uczenie się na danych treningowych. Strata na danych walidacyjnych jest nieznacznie bardziej zmienna, z kilkoma wzrostami w późniejszych epokach. Możliwa jest więc trudność z generalizacją na nowych danych.



Rysunek 1.1: Wyniki testów dla modelu podstawowego, liczba wierzchołków  $n = 4$

Ogólnie rzecz biorąc, model wydaje się dobrze uczyć na danych treningowych i generalizować na danych walidacyjnych, chociaż zmienność straty walidacyjnej może wskazywać na pewne problemy z przeuczeniem.

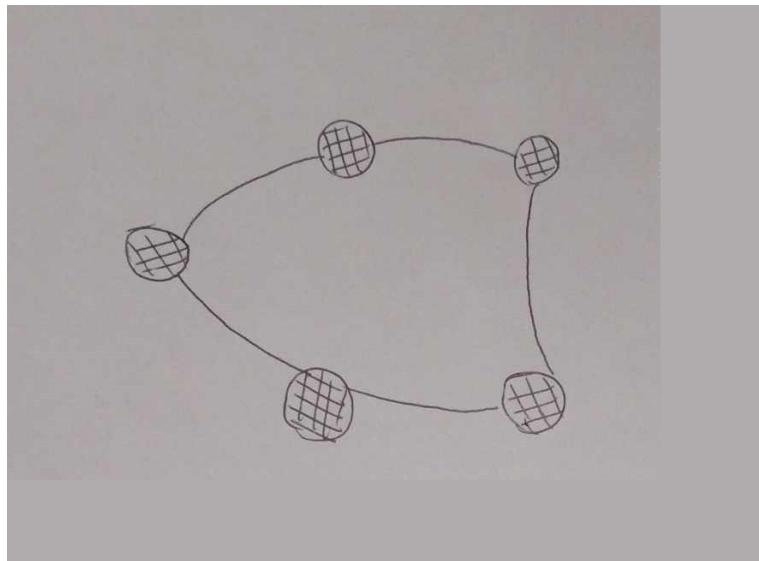
```
| - ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.99 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.92 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.37 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 19ms/step
| - ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.99 procent.
1/1   0s 17ms/step
| - ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 72.96 procent.
1/1   0s 17ms/step
| - ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 14ms/step
| - ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 92.26 procent.
1/1   0s 10ms/step
| - ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 69.63 procent.
1/1   0s 7ms/step
| - ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.77 procent.
1/1   0s 11ms/step
| - ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.90 procent.
```

Rysunek 1.2: Klasyfikacja obrazów zewnętrznych dla modelu podstawowego, liczba wierzchołków n = 4



Rysunek 1.3: Wizualizacja klasyfikacji obrazów zewnętrznych dla modelu podstawowego, liczba wierzchołków n = 4

Model przewidział poprawnie 25% grafów, co nie jest najgorszym wynikiem, zauważając że jest to najbardziej podstawowa wersja testowanego modelu.

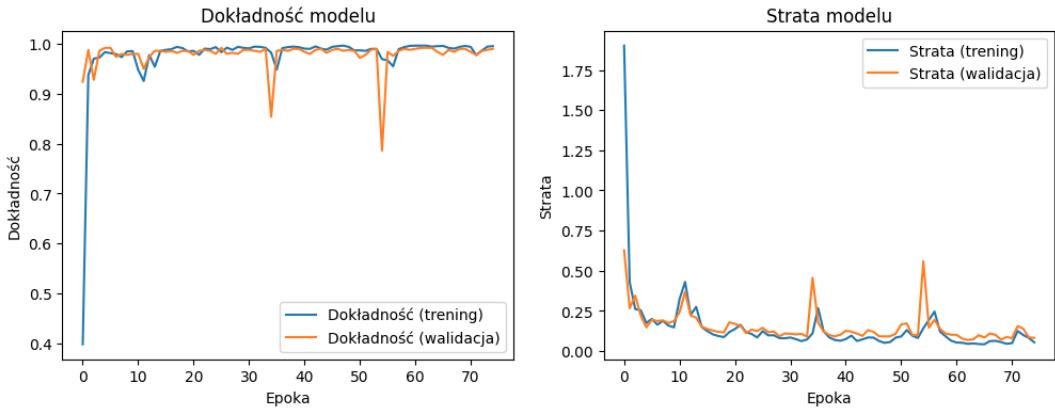


Rysunek 1.4: Klasyfikacja przykładowego grafu zewnętrznego przez model podstawowy uczony na grafach czterowierzchołkowych. Przypisana klasa to graf pełny z 99,97% pewnością.

### **Model podstawowy uczony na grafach sześciowierzchołkowych**

Kolejny warty uwagi wynik został uzyskany za pomocą podstawowego modelu, ale uczonego na grafach o sześciu wierzchołkach. Dokładność zarówno dla zbioru treningowego, jak i walidacyjnego, bardzo szybko wzrasta już w kilku pierwszych epokach i osiąga bardzo wysoki wynik, bo prawie 100%. Występuje kilka spadków obu krzywych, z czego jeden nawet do 80%. Ogólnie, po około 10 epokach, dokładność dla obu zestawów danych jest bardzo zbliżona i stabilizuje się na poziomie około 99%.

Strata, podobnie jak dokładność, spada w ciągu początkowych epok procesu nauczania do niskich wartości, bo około 0,25. W ciągu kolejnych epok, aż do końca, widać minimalny trend malejący, z trzema wzrostami straty do około wartości 0,5 na zbiorze walidacyjnym. Może to oznaczać, że w tych punktach model napotyka na trudniejsze przypadki testowe.



Rysunek 1.5: Wyniki testów dla modelu podstawowego, liczba wierzchołków  $n = 6$

Różnice między wartościami straty i dokładności dla zbioru walidacyjnego i testowego są niewielkie. Nie widać również znaczących fluktuacji. Krzywe dokładności i straty sugerują, że model może być poprawnie nauczony ogólnych wzorców i nie być nadmiernie dopasowanym do danych treningowych.

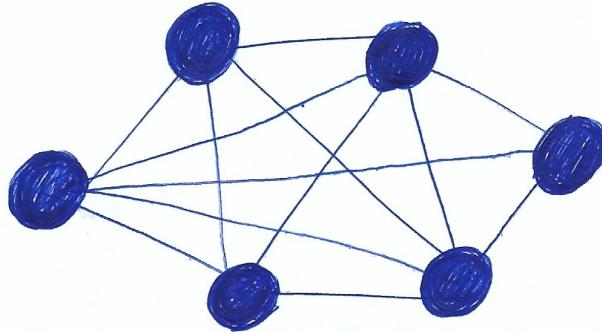
```
| - ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 14ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.74 procent.
1/1   Os 18ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 98.83 procent.
1/1   Os 18ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.10 procent.
1/1   Os 6ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 74.17 procent.
1/1   Os 16ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 90.70 procent.
1/1   Os 18ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 19ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 17ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 92.80 procent.
1/1   Os 18ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 97.19 procent.
1/1   Os 12ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 86.45 procent.
1/1   Os 10ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.90 procent.
1/1   Os 28ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   Os 29ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 98.80 procent.
1/1   Os 16ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 98.36 procent.
```

Rysunek 1.6: Klasyfikacja obrazów zewnętrznych dla modelu podstawowego, liczba wierzchołków  $n = 6$



Rysunek 1.7: Wizualizacja klasyfikacji obrazów zewnętrznych dla modelu podstawowego, liczba wierzchołków  $n = 6$

Model poprawnie klasyfikuje ponad 45% przypadków grafów zewnętrznych, co jest zaskakująco wysokim wynikiem, zważając na niską złożoność owego modelu. Czterdziestoprocentowa dokładność przy klasyfikacji pięciu klas, jest wynikiem zdecydowanie lepszym niż przypisywanie klas grafów losowo.



Rysunek 1.8: Klasyfikacja przykładowego grafu zewnętrznego przez model podstawowy uczony na grafach sześciowierzchołkowych. Przypisana klasa to graf pełny z 100% pewnością.

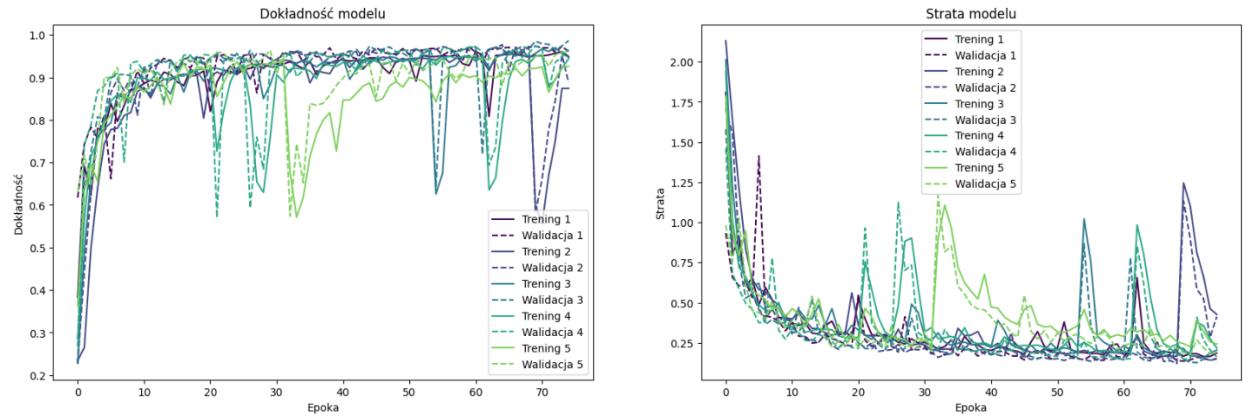
## 5.2. Model z walidacją krzyżową

### Wersja podstawowa modelu z walidacją krzyżową

W przypadku modelu z walidacją krzyżową, uczonego na grafach z 4 wierzchołkami, dokładność wzrasta gwałtownie na początku treningu, osiągając wartości powyżej 0.8 już po około 10 epokach. Dokładność stabilizuje się w okolicach 90%, ale mimo to widać pewne fluktuacje, zwłaszcza na danych walidacyjnych. Możliwe do zaobserwowania są regularne spadki dokładności w niektórych epokach, co może wynikać z niestabilnego treningu lub problemów modelu w generalizacji dla niektórych danych

walidacyjnych.

Dla straty modelu można zaobserować spadek w pierwszych 10 epokach, co mogłoby wskazywać na szybkie uczenie się modelu. Zaraz po nim, następuje stabilizacja na niskim poziomie, z pojedynczymi skokami, głównie na zbiorze walidacyjnym. Nie-regularne wzrosty straty, podobnie jak w przypadku dokładności, mogą wskazywać na problemy z przeuczeniem.



Rysunek 2.9: Wyniki testów dla modelu z walidacją krzyżową

Podsumowując, ten wariant modelu generalnie uczy się poprawnie, dzięki czemu osiąga wysoką dokładność i niską stratę. Fluktuacje jakie występują w wynikach, szczególnie na danych walidacyjnych, sugerują jednak potencjalne problemy z generalizacją, co może być wynikiem niestabilności modelu, przeuczenia modelu, lub trudności w rozpoznawaniu bardziej złożonych przykładów w danych walidacyjnych.

W przypadku tego modelu, zwiększenie liczby epok, nie przyniosłoby zamierzonych skutków. Model zbyt szybko się przeucza, a więc większa liczba iteracji nie wpłynęłaby w żaden znaczący sposób na wynik.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 19ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 97.59 procent.
1/1   0s 17ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.96 procent.
1/1   0s 17ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 97.16 procent.
1/1   0s 18ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.99 procent.
1/1   0s 17ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 51.15 procent.
1/1   0s 17ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent
1/1   0s 17ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 92.25 procent.
1/1   0s 18ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 94.90 procent.

```

Rysunek 2.10: Klasyfikacja obrazów zewnętrznych dla modelu z walidacją krzyżową



Rysunek 2.11: Wizualizacja klasyfikacji obrazów zewnętrznych dla modelu z walidacją krzyżową

Z powodu przeuczenia model nie radził sobie z zewnętrznymi obrazkami testowymi. Większość obrazów testowych określił jako grafy pełne, a tylko 3 jako jedna z innych klas, co oczywiście nie jest zgodne ze stanem rzeczywistym. Realna dokładność modelu wynosi więc około 24%.



Rysunek 2.12: Klasyfikacja przykładowego grafu zewnętrznego przez model z walidacją krzyżową. Przypisana klasa to graf pełny z 99,62% pewnością.

### Modyfikacje modelu z walidacją krzyżową

W celu poprawy dokładności i zapobiegnięciu przeuczenia wprowadzone zostały następujące modyfikacje do modelu z walidacją krzyżową. Każde z nich zostało przetestowane w osobnym modelu. Stworzony został również jeden model ze wszystkimi połączonymi modyfikacjami.

- Zmieniono liczbę filtrów w warstwach Conv2D z 32 w każdej warstwie, do kolejno 32, 64 oraz 128. Jednocześnie zwiększyliśmy parametr Dropout z 0,2 do 0,5.
- Zastosowano normalizację wsadową pomiędzy warstwami modelu - konkretnie po każdej warstwie Conv2D.
- Wprowadzenie augmentacji danych przed budową modelu, która wprowadza więcej wariacji do zbioru treningowego, w celu poprawy zdolności generalizacyjnych.

Wykorzystano również GPU w procesie pobierania wstępnego danych i cachingu zbiorów danych, by przypsieszyć przetwarzanie danych.

- Skorzystanie z wywołania zwrotnego, które zmniejsza szybkość uczenia. W przypadku stagnacji dokładności w procesie przechodzenia przez kolejne epoki uczenia modelu może pomóc w lepszej konwergencji modelu.

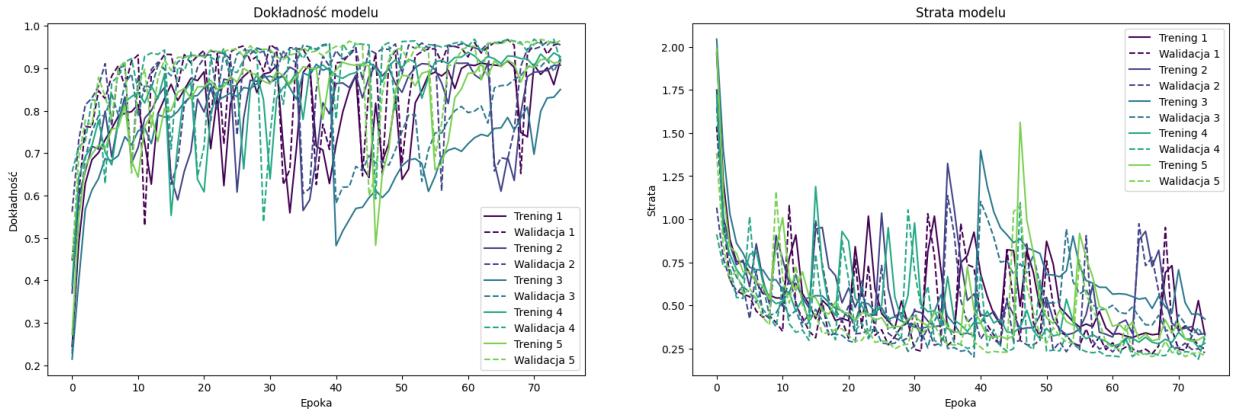
### Zmodyfikowany model z walidacją krzyżową - Conv2D i Dropout

```
1  model = tf.keras.models.Sequential([
2      tf.keras.layers.Rescaling(1./255),
3      tf.keras.layers.Conv2D(32, 3, activation='relu'),
4      tf.keras.layers.MaxPooling2D(),
5      tf.keras.layers.Conv2D(64, 3, activation='relu'),
6      tf.keras.layers.MaxPooling2D(),
7      tf.keras.layers.Conv2D(128, 3, activation='relu'),
8      tf.keras.layers.MaxPooling2D(),
9      tf.keras.layers.Flatten(),
10     tf.keras.layers.Dense(128, activation='relu',
11     kernel_regularizer=tf.keras.regularizers.l2(0.01)),
12     tf.keras.layers.Dropout(0.5),
13     tf.keras.layers.Dense(len(class_names))])
```

Listing 5.1: Listing zmodyfikowanego skryptu tworzącego model z walidacją krzyżową  
- wersja 1

Wszystkie przebiegi walidacji krzyżowej osiągają wysoką dokładność po kilku pierwszych epokach. Model bardzo szybko uczy się rozpoznawać wzorce. Walidacja również osiąga zadowalające wyniki, tj. około 90%. Może to wskazywać na poprawną generalizację do nowych danych. Występuje jednak niewielka niesabilność, występująca pomiędzy epokami, co widać po gwałtownych spadkach i wzrostach dokładności walidacji.

Strata na zbiorze treningowym i walidacyjnym systematycznie maleje z kolejnymi epokami, co może wskazywać na dobre dopasowanie do danych treningowych. Zauważalnym problemem jest jednak spora fluktuacja obu wskaźników. Model może napotykać trudności z pewnymi próbками w zbiorze danych walidacyjnych.



Rysunek 2.13: Dokładność i walidacja dla zmodyfikowanego modelu z walidacją krzyżową - Conv2D i Dropout

Modifikacja modelu wydaje się osiągać zamierzone skutki, ponieważ model wykazuje lepszą zdolność uczenia z danych treningowych oraz osiąga wysoką dokładność na danych walidacyjnych. Model może być jednak wrażliwy na trudniejsze przypadki ze zbioru danych walidacyjnych, zważając na wahania wskaźników.

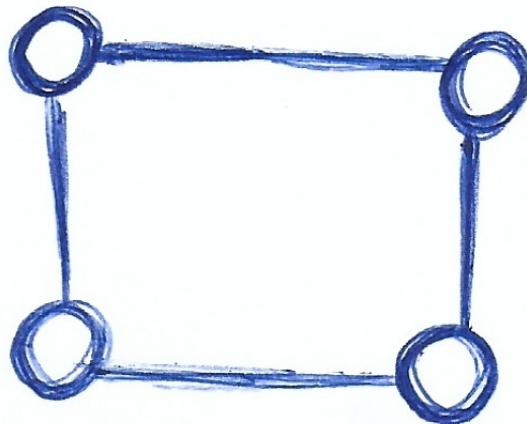
```
|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 82.23 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 51.17 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 45.67 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 98.89 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 48.27 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 69.93 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 85.87 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 98.27 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 99.79 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.89 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 93.79 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 53.22 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.83 procent.
```

Rysunek 2.14: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - Conv2D i Dropout



Rysunek 2.15: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - Conv2D i Dropout

Model poprawnie sklasyfikował aż 40% rysunków grafów, co jest znacznym połepszeniem w stosunku do początkowego modelu z zastosowaną walidacją krzyżową. Wynik, o ile daleko mu do pełnej pewności, jest znacznie lepszy niż losowe przypisywanie klas grafów. Model ten więc, choć nie idealny, wykazuje pewną realną użyteczność.



Rysunek 2.16: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model z walidacją krzyżową - Conv2D i Dropout. Przypisana klasa to cykl z 96,57% pewnością.

### Zmodyfikowany model z walidacją krzyżową - normalizacja wsadowa

Model został zmodyfikowany poprzez zastosowanie normalizacji wsadowej pomiędzy kolejnymi warstwami Conv2D modelu. Ma to na celu poprawienie stabilności treningu oraz przyspieszenie uczenia. Użyte zostało również zwiększenie liczby filtrów w warstwach Conv2D oraz zwiększenie parametru Dropout z poprzedniej modyfikacji

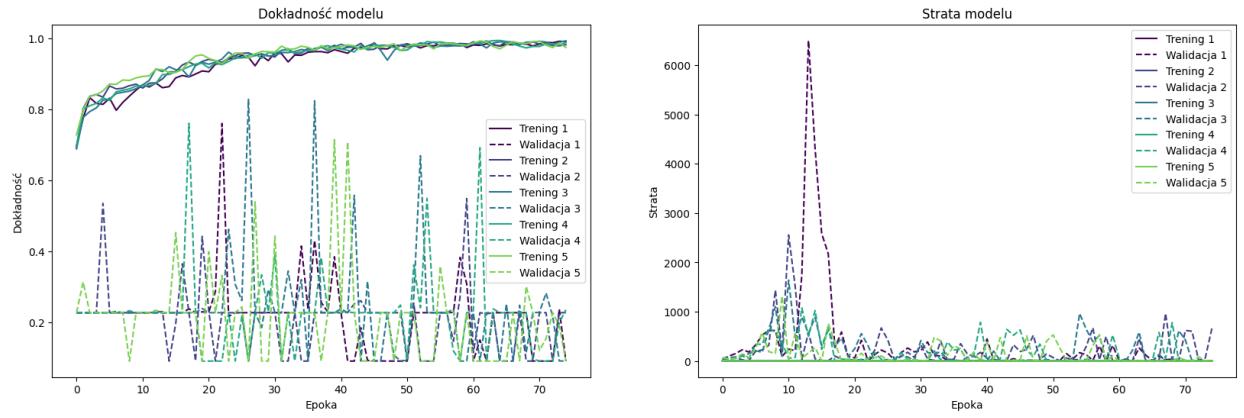
modelu, ponieważ osiągnęła ona zamierzone cele.

```
1 model = tf.keras.models.Sequential([
2     tf.keras.layers.Rescaling(1./255),
3     tf.keras.layers.Conv2D(32, 3, activation='relu'),
4     tf.keras.layers.BatchNormalization(),
5     tf.keras.layers.MaxPooling2D(),
6     tf.keras.layers.Conv2D(64, 3, activation='relu'),
7     tf.keras.layers.BatchNormalization(),
8     tf.keras.layers.MaxPooling2D(),
9     tf.keras.layers.Conv2D(128, 3, activation='relu'),
10    tf.keras.layers.BatchNormalization(),
11    tf.keras.layers.MaxPooling2D(),
12    tf.keras.layers.Flatten(),
13    tf.keras.layers.Dense(128, activation='relu',
14        kernel_regularizer=tf.keras.regularizers.l2(0.01)),
15        tf.keras.layers.Dropout(0.5),
16        tf.keras.layers.Dense(len(class_names))
])
```

Listing 5.2: Listing zmodyfikowanego skryptu tworzącego model z walidacją krzyżową  
- wersja 2

Na wykresach dokładności treningowych widać stały wzrost od około 70% do prawie 100%. Jest to oczywiście pozytywna cecha modelu, lecz po analizie krzywych dokładności walidacyjnych, należy stwierdzić, że jest to przeuczenie. Model poprawnie nauczył się danych treningowych, lecz nie zapamiętał ogólnych wzorców, zważając na wysokie wahania oraz niestałość walidacji.

Strata treningowa, co jest spodziewane po uznaniu model za przeuczony, osiąga bardzo niskie wartości treingowe - przez wszystkie epoki jest bliska 0. Strata walidacyjna zaś, mimo że pozornie wygląda na stabilną, taka nie jest - należy zwrócić uwagę na jednostki na skali. Wahania są bardzo duże oraz występuje pojedyncza wartość odstająca, która osiąga ponad 6000 jednostek.



Rysunek 2.17: Dokładność i walidacja dla zmodyfikowanego modelu z walidacją krzyżową - normalizacja wsadowa

Model jest nadmiernie dopasowany do danych treningowych i nie pot

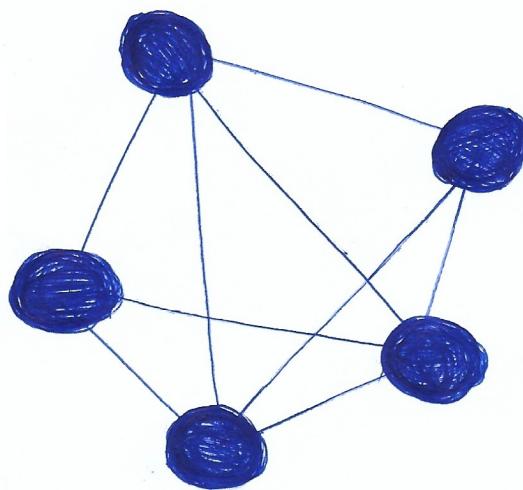
```
|= ./test_graphs\drawn\path-6.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 45.31 procent.
1/1   | 0s 20ms/step
|- ./test_graphs\drawn\path-7.png |- najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 39.12 procent.
1/1   | 0s 22ms/step
|- ./test_graphs\drawn\path-8.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 95.23 procent.
1/1   | 0s 22ms/step
|- ./test_graphs\drawn\path-9.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 99.94 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\drawn\tree-binary-1.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 99.52 procent.
1/1   | 0s 26ms/step
|- ./test_graphs\drawn\tree-binary-2.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 55.49 procent.
1/1   | 0s 14ms/step
|- ./test_graphs\drawn\tree-binary-3.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 97.57 procent.
1/1   | 0s 14ms/step
|- ./test_graphs\drawn\tree-binary-4.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 100.00 procent.
1/1   | 0s 16ms/step
|- ./test_graphs\drawn\tree-binary-5.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 99.91 procent.
1/1   | 0s 17ms/step
|- ./test_graphs\generated\cycle-45.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 98.65 procent.
1/1   | 0s 11ms/step
|- ./test_graphs\generated\full-113.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 81.94 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\generated\path-78.png |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 67.32 procent.
1/1   | 0s 24ms/step
|- ./test_graphs\internet\internet-cycle-1.png |- najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   | 0s 21ms/step
|- ./test_graphs\internet\internet-full-1.jpg |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 80.81 procent.
1/1   | 0s 19ms/step
|- ./test_graphs\internet\internet-full-2.jpg |- najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 89.91 procent.
```

Rysunek 2.18: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - normalizacja wsadowa



Rysunek 2.19: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - normalizacja wsadowa

Model sklasyfikował poprawnie prawie 31% grafów zewnętrznych. Jest to o 9 punktów procentowych mniej niż poprzednia modyfikacja modelu, co prowadzi do wniosków, że zastosowanie normalizacji wsadowej nie spełniło założonej funkcji. Nie jest to udany eksperyment, lecz model wciąż prezentuje pewną użyteczność.



Rysunek 2.20: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model z walidacją krzyżową - normalizacja wsadowa. Przypisana klasa to graf bezkrawędziowy z 100% pewności.

### Zmodyfikowany model z walidacją krzyżową - augmentacja danych

Przed rozpoczęciem nauki modelu, zastosowana została augmentacja danych treningowych. Najpierw, stworzony został sekwencyjny model, składający się z pewnych warstw. Pierwsza z nich - RandomFlip, odwraca losowe obrazy w poziomie, zwiększając tym samym różnorodność danych. Kolejna - RandomRotation, losowo obraca obrazy

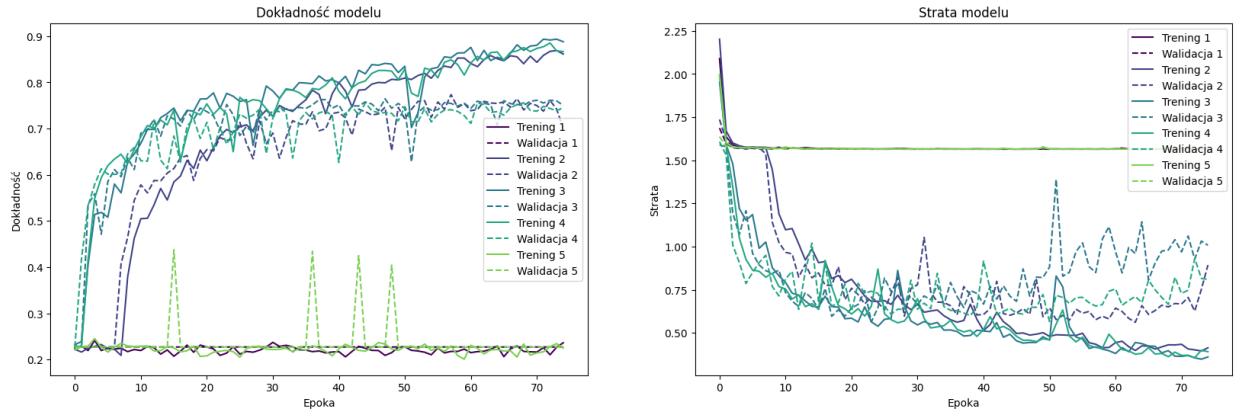
o kąt w zakresie od -0.1 do 0.1 radianów, tym samym pomagając modelowi stać się odpornym na rotacje grafów. RandomZoom z kolei przybliża lub oddala obrazy o wartości oscylujące w zakresie 10%. W teorii, powinno to uodpornić model na grafy różnej wielkości. W kolejnej linii skryptu, wyżej stworzony sekwencyjny model został zastosowany do zbioru danych treningowych. Dalej, na zbiorach uczących i walidacyjnych, użyte zostały funkcje cachujące dane - przyspiesza to proces uczenia.

```
1 data_augmentation = tf.keras.Sequential([
2     tf.keras.layers.RandomFlip("horizontal"),
3     tf.keras.layers.RandomRotation(0.1),
4     tf.keras.layers.RandomZoom(0.1),
5 ])
6 train_ds = train_ds.map(lambda x, y: (data_augmentation
7 (x), y))
train_ds = train_ds.cache().shuffle(1000).prefetch(
8 buffer_size=tf.data.AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=tf.data.
AUTOTUNE)
```

Listing 5.3: Listing zmodyfikowanego skryptu poprzedzającego tworzenie modelu z walidacją krzyżową - wersja 3

Dokładność treningowa i walidacyjna w ciągu kilku pierwszych epok wzrasta znacząco, od około 23% do 50%, po czym rośnie w stałym tempie do 90%. Wydają się to być dość realistyczne wartości dokładności dla nieprzeuczonego modelu. Głównym problemem w tym przypadku wydają się być niektóre przebiegi walidacji krzyżowej. W ich przypadku, dokładność na danych treningowych oraz walidacyjnych wynosi przez wszystkie epoki uczenia, około 23%. Przy każdej iteracji walidacji krzyżowej generowane są nowe zbiorы walidacyjne i treningowe, co w połączeniu z augmentacją danych, może powodować, że niektóre z tych zbiorów są wyjątkowo trudne do nauki dla modelu. Możliwe jest również, że niektóre zbiorы na których dokonano augmentacji, zważając na to, że jest ona losowa, są nieprzystosowane do nauki modeli uczenia maszynowego.

Strata zachowuje się podobnie jak dokładność. Dla pewnych przejść walidacji krzyżowej, osiąga ona realistyczne i spadające z biegiem epok wartości. Dla innych zaś, wartości straty są stałe przez wszystkie epoki uczenia.



Rysunek 2.21: Dokładność i walidacja dla zmodyfikowanego modelu z walidacją krzyżową - augmentacja danych

Model wydaje się radzić poprawnie z niektórymi wariantami zaumentowanych danych, a z innymi - całkowicie nie, co wyraża się w krzywych dokładności oraz straty modelu.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 19ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 17ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.
1/1 _____ 0s 18ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 22.88 procent.

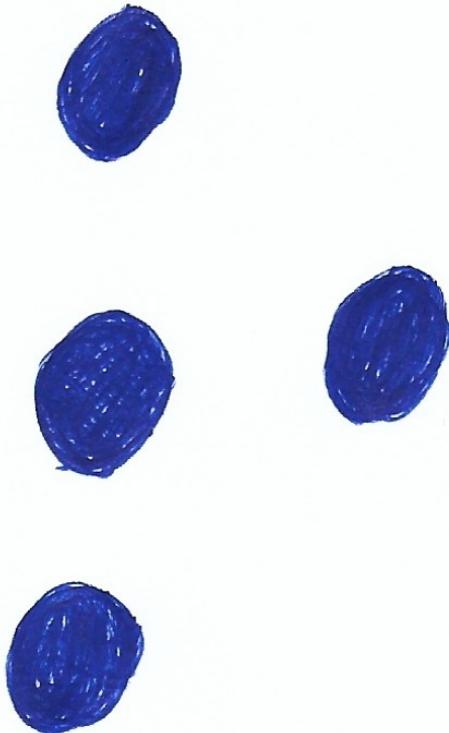
```

Rysunek 2.22: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - augmentacja danych



Rysunek 2.23: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - augmentacja danych

Model w takiej postaci nie sklasyfikował poprawnie nawet 10% grafów testowych. Nie jest to zaskoczenie, mając na uwadze osiągniętą dokładność modelu na pewnych przejściach walidacji krzyżowej, która wynosiła około 22-23%.



Rysunek 2.24: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model z walidacją krzyżową - augmentacja danych. Przypisana klasa to drzewo binarne z 22,88% pewnością.

#### Zmodyfikowany model z walidacją krzyżową - spowolnienie uczenia

W tym modelu zostało zastosowane wywołanie zwrotne, które spowalnia proces uczenia. Współczynnik procesu uczenia decyduje o tym, jak duże kroki wykonuje algorytm optymalizacyjny podczas aktualizacji wag sieci neuronowej. Gdy owy współczynnik jest zbyt duży, model może oscylować wokół minimum funkcji straty i nigdy go nie osiągnąć. W przypadku kiedy jest zbyt mały, może prowadzić do bardzo wolnego uczenia się, czy nawet utknienia w lokalnym minimum. Parametr *factor* w skrypcie decyduje o tym, o jaką wartość zredukować współczynnik uczenia, jeśli nie nastąpi poprawa uczenia przez okres wyrażony parametrem *patience*. *min\_lr* to minimalna wartość współczynnika uczenia, poniżej której wsółczynnik nie zostanie już zredukowany. Zapobiega to zbyt drastycznemu zmniejszeniu learning rate, które mogłoby zahamować

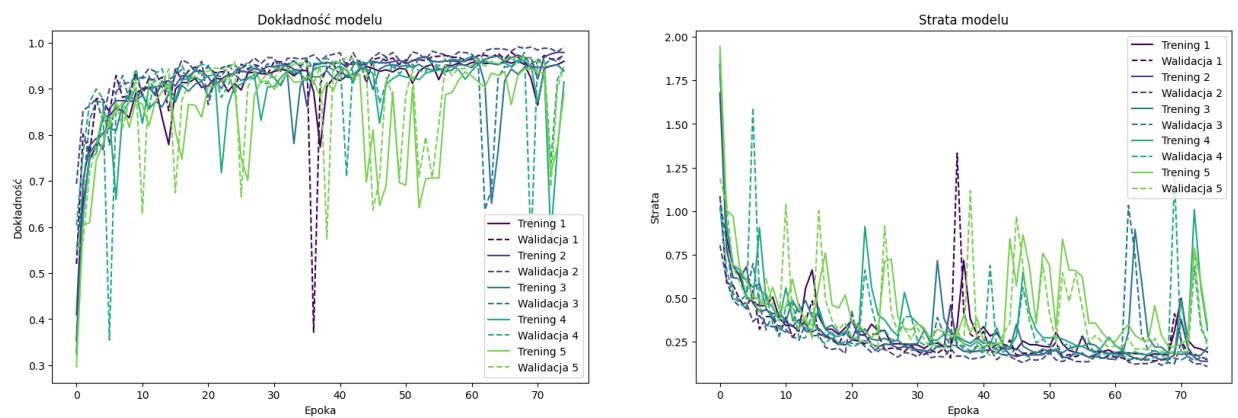
proces uczenia się modelu.

```
1  reduce_lr = tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5, min_lr=0.001)
```

Listing 5.4: Listing zmodyfikowanego skryptu znajdującego się bezpośrednio po tworzeniu modelu z walidacją krzyżową - wersja 4

Krzywa dokładności treningowej oraz walidacyjnej wzrasta gwałtownie w początkowych epokach nauki. Oznacza to, że model szybko uczy się nieskomplikowanych wzorców. W kolejnej części procesu nauki, dokładność na obu zbiorach stabiizuje się między 90%, a 100%. Z powodu widocznych sporych spadków na zbiorze walidacyjnym, można założyć pewne problemy z modelem - przeuczenie.

Podobnie jak dokładność, strata gwałtownie obniża się na początku procesu uczenia. W kolejnych epokach dokonuje się pewna stabilizacja, lecz mimo tego widoczne są fluktuacje, szczególnie na zbiorze walidacyjnym. Możliwe, że model jest zbyt dopasowany do specyficznych cech danych treningowych i nie generalizuje odpowiednio na nowe dane.



Rysunek 2.25: Dokładność i walidacja dla zmodyfikowanego modelu z walidacją krzyżową - spowolnienie uczenia

Model wykazuje pewne nieporządane cechy, takie jak wahania dokładności i straty na zbiorach walidacyjnych, co wskazuje na przeuczenie modelu.

```

| ..\test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 51.36 procent.
1/1          @ 17ms/step
| ..\test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 19ms/step
| ..\test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 96.37 procent.
1/1          @ 18ms/step
| ..\test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 18ms/step
| ..\test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 99.96 procent.
1/1          @ 18ms/step
| ..\test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 17ms/step
| ..\test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 99.98 procent.
1/1          @ 17ms/step
| ..\test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 18ms/step
| ..\test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 17ms/step
| ..\test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 99.96 procent.
1/1          @ 17ms/step
| ..\test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 99.99 procent.
1/1          @ 16ms/step
| ..\test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 86.63 procent.
1/1          @ 17ms/step
| ..\test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- full |- z prawdopodobieństwem 100.00 procent.
1/1          @ 17ms/step
| ..\test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 82.64 procent.
1/1          @ 18ms/step
| ..\test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- path |- z prawdopodobieństwem 77.67 procent.

```

Rysunek 2.26: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - spowolnienie uczenia



Rysunek 2.27: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - spowolnienie uczenia

Model poprawnie sklasyfikował około 31% grafów zewnętrznych. Jest to w zasadzie identyczny wynik, jak dla modelu z zastosowaniem normalizacji wsadowej. Zważając na to, że ogólną większość grafów oznaczył jako pełne, może to również być przypadek.

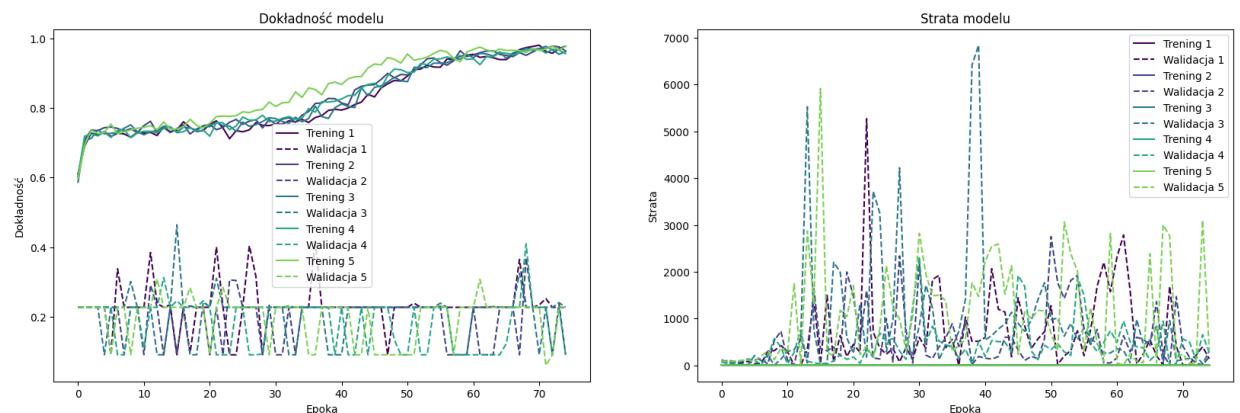


Rysunek 2.28: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model z walidacją krzyżową - spowolnienie uczenia. Przypisana klasa to ścieżka z 95,52% pewnością.

### Zmodyfikowany model z walidacją krzyżową - modyfikacje połączone

Dokładność dla wszystkich przebiegów stopniowo wzrasta wraz z liczbą epok i osiąga wysoki poziom, bo powyżej 80%, pod koniec treningu. W przypadku walidacji jednak, wyniki nie są zadowalające, a wreszcie bardzo niestabilne. Przez większość przebiegów uczenia utrzymuje się na niskim poziomie, co może sugerować problemy z generalizacją danych.

Strata treningowa maleje w większości przebiegów, co jest spodziewane podczas uczenia. Widać jednak pewne fluktuacje, świadczące o problemach z konwergencją. Dla straty walidacyjnej można zaobserwować wielką niestabilność - osiąga bardzo wysokie wartości, nawet rzędu kilku tysięcy, jak również bardzo niskie, bliskie zeru.



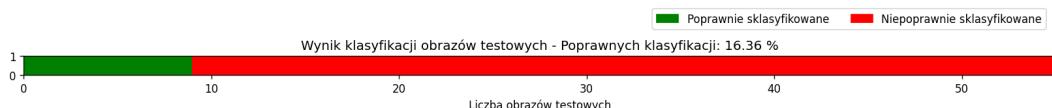
Rysunek 2.29: Dokładność i walidacja dla zmodyfikowanego modelu z walidacją krzyżową - modyfikacja połączone

Ogólne wnioski jakie można wyciągnąć z procesu uczenia tego modelu, są ta-

kie, że model kompletnie nie radzi sobie z danymi walidacyjnymi. Model ewidentnie przeucza się na danych treningowych, podczas gdy jego wydajność na zbiorze walidacyjnym jest bardzo słaba. Zastosowanie wszystkich zaproponowanych technik na raz, nie osiągnęło zamierzonego skutku.

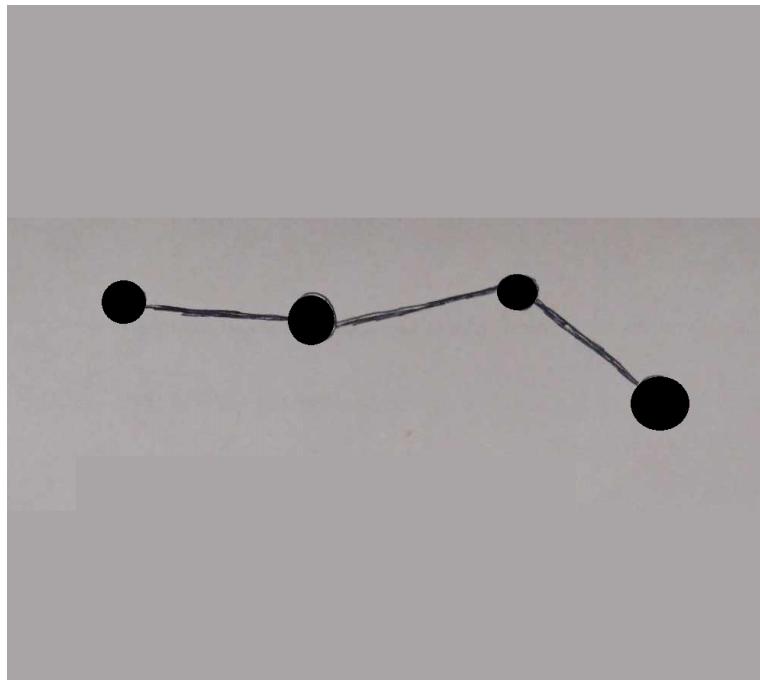
```
| - ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
| - ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
| - ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| - ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
```

Rysunek 2.30: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - modyfikacja połączone



Rysunek 2.31: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu z walidacją krzyżową - modyfikacja połączone

Model poprawnie wskazał klasy tylko 16% grafów testowych, co jest znacznie poniżej oczekiwanych rezultatów, zważając na liczbę zastosowanych modyfikacji modelu. Wnioskować można, że nie każda modyfikacja była odpowiednio dobrana do danego przypadku lub niektóre z nich nie łączą się zbyt dobrze.



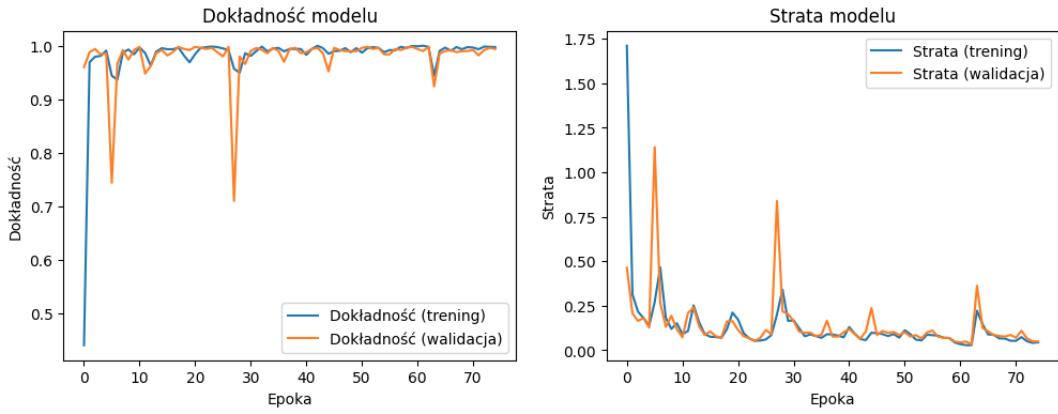
Rysunek 2.32: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model z walidacją krzyżową - modyfikacje połączone. Przypisana klasa to graf bezkrawędziowy z 70,87% pewnością.

### 5.3. Model ze zmienną liczbą wierzchołków

#### Wersja podstawowa modelu ze zmienną liczbą wierzchołków

Dokładność modelu uczonego na grafach treningowych z liczbą wierzchołków od czterech do siedmiu, prezentuje się dość stabilnie po początkowej fazie wzrostu. Występują tylko drobne fluktuacje. Po kilku początkowych epokach, dokładność oscyluje wokół 95%, dochodząc nawet do 100%. Linie walidacji i treningu są bardzo blisko siebie, co sugeruje dobrą generalizację modelu i nie wskazuje na przeuczenie.

W przypadku straty modelu, początkowy gwałtowny spadek sugeruje, że model dość szybko się uczy. Po 10 epokach następuje stabilizacja straty na niskim, bo wynoszącym około 0.1, poziomie. Podobnie jak w przypadku dokładności, strata dla zbioru walidacyjnego jest blisko straty treningowej. Można zaobserować pewne wzrosty, które mogą być spowodowane trudniejszymi przypadkami w zbiorze walidacyjnym.



Rysunek 3.33: Dokładność i walidacja dla modelu ze zmienną liczbą wierzchołków

Model ten wydaje się być dobrze dopasowywany i stabilny oraz poprawnie generalizujący. Z uwagi na bliskość wyników dla treningu i walidacji, można stwierdzić, że model nie jest przeuczony.

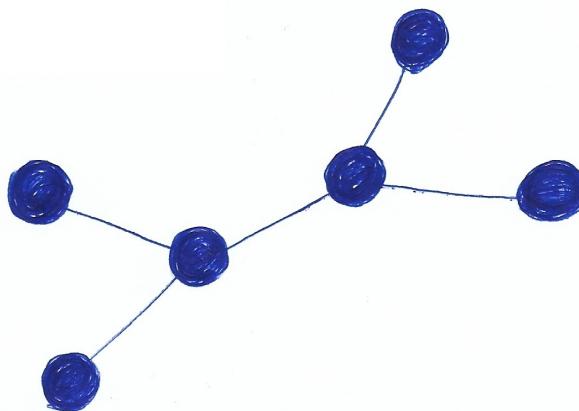
```
| - ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 22ms/step
| - ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 19ms/step
| - ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
| - ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.98 procent.
1/1   0s 21ms/step
| - ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 7ms/step
| - ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.99 procent.
1/1   0s 4ms/step
| - ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| - ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 24ms/step
| - ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 15ms/step
| - ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 98.57 procent.
1/1   0s 19ms/step
| - ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.79 procent.
1/1   0s 4ms/step
| - ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 100.00 procent.
1/1   0s 7ms/step
| - ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 11ms/step
| - ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.18 procent.
1/1   0s 18ms/step
| - ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 95.82 procent.
```

Rysunek 3.34: Klasyfikacja obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków



Rysunek 3.35: Wizualizacja klasyfikacji obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków

Model poprawnie sklasyfikował ponad 34% rysunków zewnętrznych. Nie jest to zły wynik, biorąc pod uwagę niską złożoność modelu.



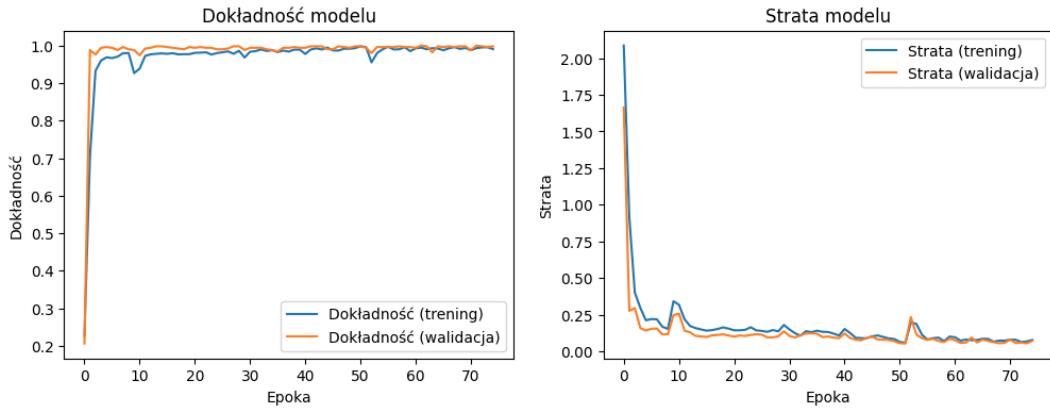
Rysunek 3.36: Klasyfikacja przykładowego grafu zewnętrznego przez model ze zmienną liczbą wierzchołków. Przypisana klasa to drzewo binarne z 100% pewnością.

### Zmodyfikowany model ze zmienną liczbą wierzchołków - Conv2D i Dropout oraz spowolnienie uczenia

W modelu wprowadzone zostało zwiększenie liczby filtrów dla warstw Conv2D, zwiększenie parametru Dropout oraz zastosowano wywołanie zwrotne, które spowalnia uczenie.

Dokładność treningowa i walidacyjna modelu bardzo szybko wzrasta do wartości bliskich 100% i stabilizuje się na takim poziomie do końca procesu nauki. Gdyby tylko dokładność treningowa osiągała taki poziom, można by założyć z dużą dozą pewności, przeuczenie modelu. W tym przypadku jednak, różnice między dokładnością walidacyjną a treningową są marginalne.

Straty modelu wykazują podobne cechy do dokładności - szybki spadek wartości oraz stabilizacja na niskim poziomie.



Rysunek 3.37: Dokładność i weryfikacja dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków - Conv2D i Dropout

Wydaje się, że model osiągnął pewną stabilność na niskim poziomie, co może pozytywnie świadczyć o jego zdolności generalizacji na nowe dane.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.09 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.73 procent.
1/1   0s 21ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.41 procent.
1/1   0s 19ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 98.44 procent.
1/1   0s 19ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 89.99 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 98.98 procent.
1/1   0s 21ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 99.99 procent.
1/1   0s 19ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 87.34 procent.
1/1   0s 19ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.91 procent.
1/1   0s 20ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 96.23 procent.
1/1   0s 19ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 59.33 procent.

```

Rysunek 3.38: Klasifikacja obrazów zewnętrznych dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków - Conv2D i Dropout



Rysunek 3.39: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków - Conv2D i Dropout

Osiągnięta stabilizacja modelu i brak znaczących różnic między wartościami dokładności, czy straty na zbiorach treningowych i wdrożeniowych, nie przełożyła się w sposób rewolucyjny na osiągi modelu. Model jednakże nie wypadł całkowicie źle - poprawnie sklasyfikował 36% zewnętrznych grafów testowych, co plasuje go w czołówce testowanych modeli pod kątem realnej dokładności.



Rysunek 3.40: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model ze zmienną liczbą wierzchołków - Conv2D i Dropout oraz spowolnienie uczenia. Przypisana klasa to cykl z 98,88% pewnością.

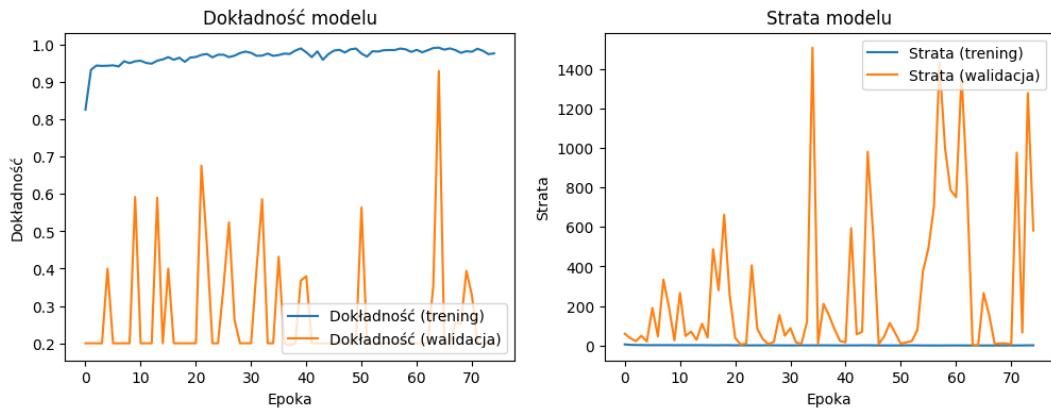
### Zmodyfikowany model ze zmienną liczbą wierzchołków - modyfikacje połączone

Do modelu wprowadzone zostały modyfikacje na wzór tych ze zmodyfikowanego modelu z walidacją krzyżową. Przetestowany został wariant z włączonymi wszystkimi modyfikacjami na raz.

Podobnie jak dla zmodyfikowanego modelu z walidacją krzyżową, dokładność treningowa jest bardzo wysoka (prawie 100%), ale dokładność walidacyjna nie jest ustabilizowana. Jej wartości oscylują w przedziale od 0,2, aż do 0,9. To sugeruje, że

model nie generalizuje dobrze na nowych danych i może być nadmiernie dopasowany.

Wykresy straty prezentują się w podobny sposób jak i wykresy dokładności. Strata treningowa jest bliska zeru, co mówi o świetnym zapamiętywaniu danych treningowych przez model, kosztem generalizacji na nowe dane. Strata walidacji jednak jest niestabilna z bardzo wysokimi wartościami w niektórych epokach.



Rysunek 3.41: Dokładność i walidacja dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków

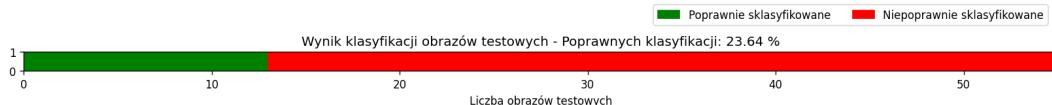
Model wykazuje cechy modelu przeuczonego. Są to bardzo wysoka dokładność i niemal zerowa strata na zbiorze treningowym oraz bardzo niska dokładność i wysoka, ale niestabilna strata na zbiorze walidacyjnym. Wniosek jaki można wyciągnąć jest taki, że model zapamiętał dane treningowe, ale nie nauczył się żadnych wzorców.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 99.40 procent.
1/1   0s 7ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 31ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.95 procent.
1/1   0s 31ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 31ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 35ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 31ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.

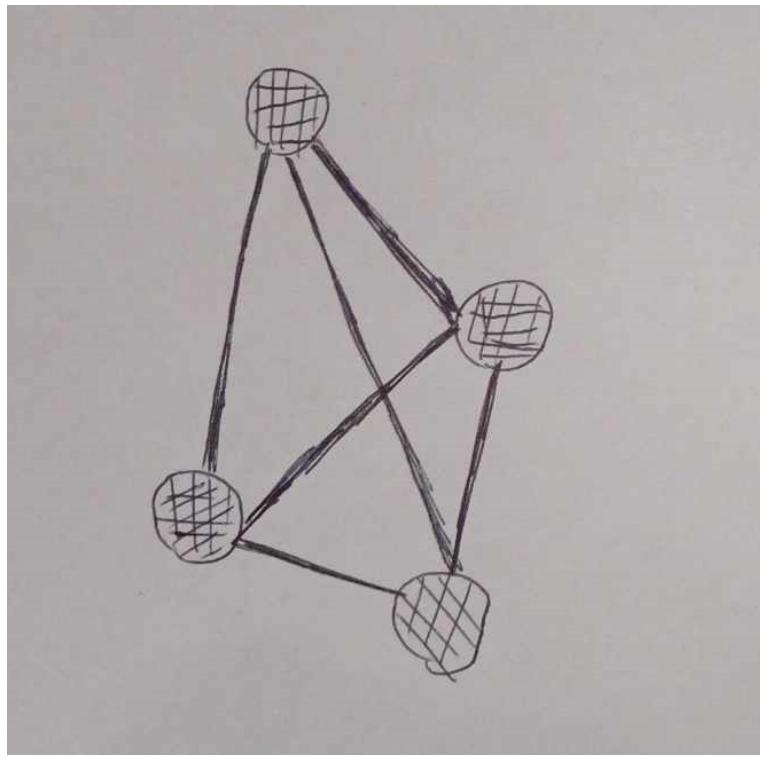
```

Rysunek 3.42: Klasyfikacja obrazów zewnętrznych dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków



Rysunek 3.43: Wizualizacja klasyfikacji obrazów zewnętrznych dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków

Model poprawnie wskazał tylko około 24% grafów, lecz analiza jego wskaźników mówi nam, że jest to dzieło raczej przypadku, aniżeli poprawnego nauczenia się ogólnych wzorców danych. Jego dokładność walidacyjna była bardzo niestabilna.



Rysunek 3.44: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model ze zmienną liczbą wierzchołków - modyfikacje połączone Przypisana klasa to ścieżka z 99,68% pewnością.

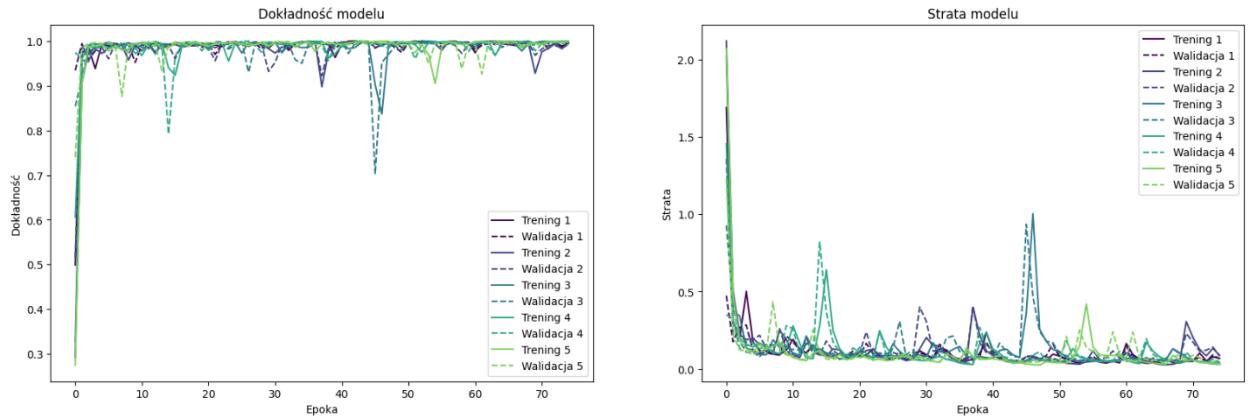
#### **5.4. Model ze zmienną liczbą wierzchołków i walidacją krzyżową**

##### **Wersja podstawowa modelu ze zmienną liczbą wierzchołków i walidacją krzyżową**

Model uczony na grafach ze zmienną liczbą wierzchołków oraz zastosowaną walidacją krzyżową, bardzo szybko osiąga poziom dokładności biski 100%, bo już w kilku pierwszych epokach. Po około 10 epoce, dokonuje się stabilizacja, oscylująca między 95%, a 100%. W przypadku tego modelu, fluktuacje dokładności są znikome, co wskazuje na dobrą stabilność modelu. Pojedyncze przypadki spadu dokładności, mogą być spowodowane bardziej skomplikowanymi przypadkami w zbiorze danych walidacyjnych.

Strata tego modelu gwałtownie spada na początku procesu uczenia, po czym stabilizuje się na zadowalająco niskim poziomie - poniżej 20%. Skoki wskaźnika są bardziej zauważalne na zbiorze walidacyjnym, ale nie wydają się być regularne i nie wpływają na ogólny wynik. Mogą być wynikiem, przeuczenia na pojedynczych epokach

lub naturalną zmiennością walidacyjnego zbioru danych.



Rysunek 4.45: Dokładność i walidacja dla modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

Model wydaje się dokładny na zbiorze treningowym i walidacyjnym, co może skutkować polepszoną skutecznością w klasyfikacji grafów. Minimalne różnice pomiędzy dokładnością treningową a walidacyjną wskazują na dobrą zdolność generalizacji. Z otrzymanych wyników, wydawałoby się, że model nie uległ przeuczeniu, choć jest to również możliwe, zważając na bardzo wysokie wyniki dokładności.

```
| .../test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| .../test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 19ms/step
| .../test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| .../test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 24ms/step
| .../test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 24ms/step
| .../test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.95 procent.
1/1   0s 19ms/step
| .../test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
| .../test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| .../test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
| .../test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 17ms/step
| .../test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 76.12 procent.
1/1   0s 15ms/step
| .../test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.99 procent.
1/1   0s 19ms/step
| .../test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
| .../test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 18ms/step
| .../test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 99.87 procent.
```

Rysunek 4.46: Klasyfikacja obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków i walidacją krzyżową



Rysunek 4.47: Wizualizacja klasyfikacji obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

Model poprawnie sklasyfikował aż 38% testowanych danych zewnętrznych. Jest to zadowalający wynik, zważyając na trudności innych modeli w poprawnym wskazywaniu klas sprawdzanych grafów. Podejrzenia dotyczące przeuczenia modelu były więc bezpodstawne. Model całkiem dobrze nauczył się żądanych wzorców.



Rysunek 4.48: Klasyfikacja przykładowego grafu zewnętrznego przez model ze zmienną liczbą wierzchołków i walidacją krzyżową. Przypisana klasa to graf pełny z 100% pewnością.

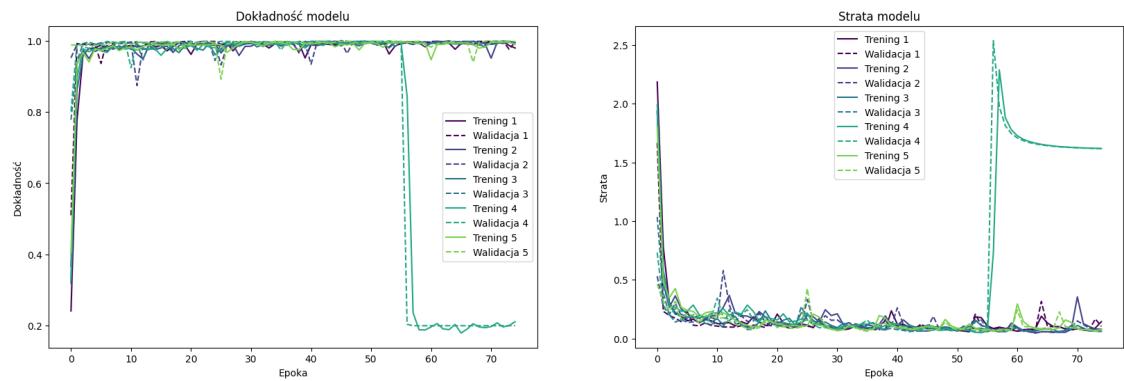
#### **Zmodyfikowany model ze zmienną liczbą wierzchołków i walidacją krzyżową - Conv2D i Dropout oraz spowolnienie uczenia**

Model stosuje zwiększone parametry filtrów dla warstw Conv2D, powiększony parametr Dropout z 0,2 do 0,5 oraz spowolnienie uczenia.

Dokładność osiąga poziom bliski 100% na zbiorze treningowym i walidacyjnym w bardzo szybkim tempie. Po osiągnięciu tego pułapu, utrzymuje się na nim do ostatniej epoki procesu nauczania modelu. Jedyną anomalię można zaobserować około 55 epoki dla jednego z przejść walidacji krzyżowej. Spada ona gwałtownie do poziomu 23% i pozostaje w takim stanie do końca nauki.

Strata prezentuje się podobnie jak dokładność. Wartości utrzymują się na dość

niskich poziomach przez cały proces uczenia po początkowym spadku. Około 55 epoki, dla jednego przejścia walidacji krzyżowej, da się zaobserować nagły wzrost straty do około 2,5 jednostek, który stopniowo spada do około 1,8.



Rysunek 4.49: Dokładność i walidacja dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową - Conv2D i Dropout

W przypadku pojedynczej anomalii jaką da się zaobserować na krzywych dokładności oraz straty, można wnioskować, że dla jednego przejścia walidacji krzyżowej, a konkretnie czwartego, wystąpił pewien problem z uczeniem modelu. Mogła to być zmiana wartości współczynnika uczenia się lub inna zmiana w procesie treningu. Różnice pomiędzy wartościami dla danych treningowych oraz walidacyjnych są znikome - nawet w przypadku odstającym wartościami od reszty. Ogólnie, model osiągnął bardzo dobre wyniki, co sugeruje, że proces nauczania przebiegł pomyślnie.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.14 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 95.58 procent.
1/1   0s 20ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 28ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 14ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 21ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 94.25 procent.
1/1   0s 16ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 16ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 22ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   0s 7ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 99.83 procent.
1/1   0s 28ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 92.52 procent.
1/1   0s 17ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.34 procent.
1/1   0s 10ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   0s 22ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   0s 19ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 84.79 procent.

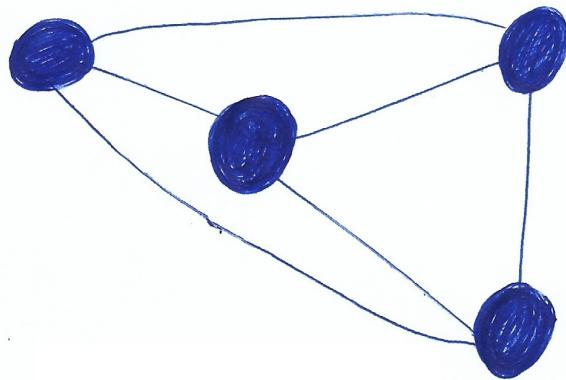
```

Rysunek 4.50: Klasyfikacja obrazów zewnętrznych dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową - Conv2D i Dropout



Rysunek 4.51: Wizualizacja klasyfikacji obrazów zewnętrznych dla zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową - Conv2D i Dropout

Poprawna klasyfikacja 29% grafów jest w pewnym sensie rozczarowującym wynikiem, zważajac na wysokie wartości dokładności, nawet na zbiorach walidacyjnych oraz całkiem wysoki wynik modelu bez modyfikacji. Wnioskować można, że spowolnienie uczenia i zwiększenie liczby filtrów w warstwach Conv2D oraz zwiększenie parametru dropout, nie przyniosło oczekiwanych skutków.



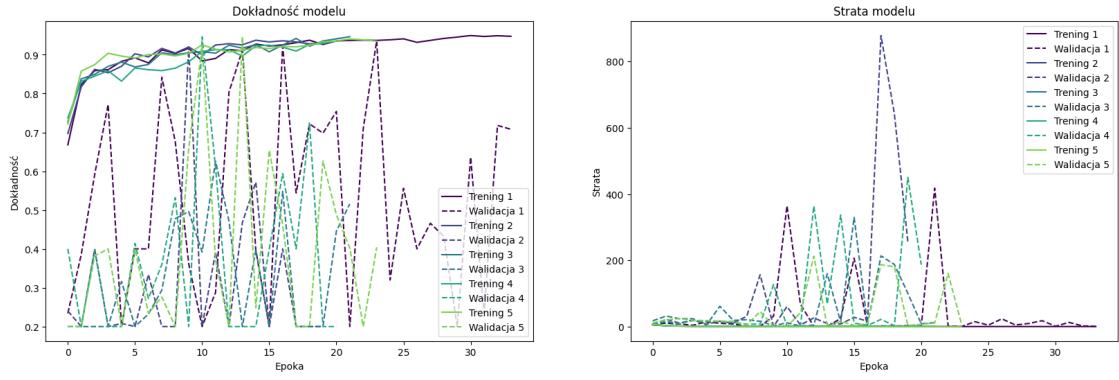
Rysunek 4.52: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model ze zmienną liczbą wierzchołków i walidacją krzyżową - Conv2D i Dropout oraz spowolnienie uczenia. Przypisana klasa to drzewo binarne z 99,95% pewnością.

### **Zmodyfikowany model ze zmienną liczbą wierzchołków i walidacją krzyżową - modyfikacje połączone**

Do modelu wprowadzone zostały wszystkie modyfikacje wymienione w rozdziale z modelem z walidacją krzyżową, czyli zwiększenie liczby filtrów w warstwach, normalizacja wsadowa, augmentacja danych oraz zmniejszenie szybkości uczenia.

Model dość szybko osiąga wysoką dokładność na zbiorze treningowym - stabilizuje się około 90-95%. Podobnie jak i w innych modelach z zastosowanymi wszystkimi modyfikacjami, dokładność na zbiorze walidacyjnym jest niestabilna. Można zaobserwować duże wahania pomiędzy kolejnymi epokami procesu uczenia. Sugeruje to trudności z uczeniem się wzorców.

Model szybko zmniejsza stratę treningową i pozostaje ona na bardzo niskim poziomie przez cały proces uczenia. Wskazuje to na sprawność w uczeniu się na danych treningowych. Wykresy strat przedstawiają się w nietypowej formie. Dla początkowych i końcowych epok, straty walidacji niewiele się wahają, a wręcz przeciwnie sytuacja wygląda dla epok od około dziesiątej do dwudziestej. Model ma więc trudności z radzeniem sobie z danymi, które są dla niego nowe.



Rysunek 4.53: Dokładność i walidacja dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

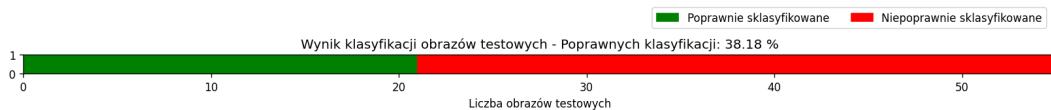
Model osiąga bardzo wysokie wyniki na zbiorze treningowym, ale ma problemy z generalizacją na zbiorze walidacyjnym. Wskazuje to na przeuczenie modelu i brak uczenia się wzorców ogólnych.

```

|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1   Os 16ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.85 procent.
1/1   Os 16ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 75.03 procent.
1/1   Os 21ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 95.39 procent.
1/1   Os 21ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 21ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 21ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.55 procent.
1/1   Os 20ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 20ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 21ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   Os 20ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 21ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 91.66 procent.
1/1   Os 21ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1   Os 20ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1   Os 20ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 99.99 procent.

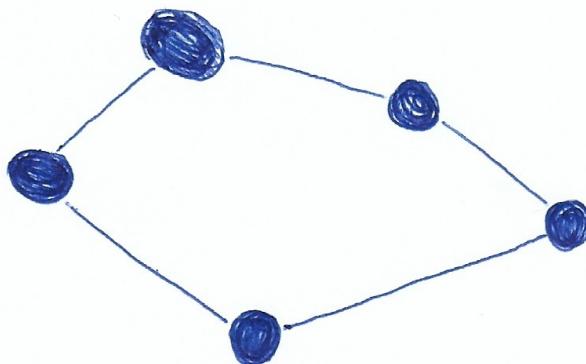
```

Rysunek 4.54: Klasyfikacja obrazów zewnętrznych dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową



Rysunek 4.55: Wizualizacja klasyfikacji obrazów zewnętrznych dla w pełni zmodyfikowanego modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

Model ocenił poprawnie około 38% klas grafów zewnętrznych, co nie jest złym wynikiem. W przypadku modeli z walidacją krzyżową, zastosowanie wszystkich przygotowanych modyfikacji modelu, nie do końca spełniło swoje złożenia, ale również nie pogorszyło wyniku.



Rysunek 4.56: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany model ze zmienną liczbą wierzchołków i walidacją krzyżową - modyfikacje połączone. Przypisana klasa to cykl z 100% pewności.

## 5.5. Modyfikacje modelu o najlepszej dokładności realnej

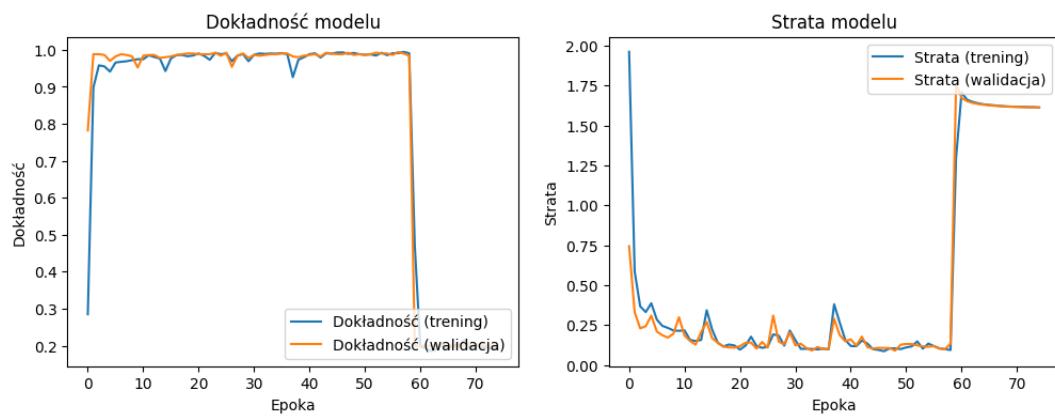
W wyniku testów wyłoniony został model o najlepszej dokładności realnej, tj. dokładności klasyfikacji zewnętrznych grafów testowych. Model ten okazał się jednym z najbardziej podstawowych z przygotowanych, bo jest to podstawowy model uczony na grafach sześciowierzchołkowych. W ramach eksperymentu, zastosowano dla niego modyfikacje, które pomogły zwiększyć realną dokładność modelu. Modyfikacje te zostały wprowadzone i przetestowane w podrozdziale z testami modeli z walidacją krzyżową. Jedyną zmianą z zaproponowanych, która przynosi realne korzyści, jest zwiększenie liczby filtrów w warstwach sieci neuronowej, kolejno 32, 64 oraz 128 dla Conv2D, oraz

zastosowanie zwiększonego parametru dropout - 0,5.

### Zmodyfikowany model podstawowy uczony na grafach sześciowierzchołkowych

Model osiąga wysokie i stabilne wyniki, lecz po około 55 epoce procesu nauczania, dokładność gwałtownie spada, aż do wartości około 23%.

Ta sama sytuacja dotyczy straty, która gwałtownie wzrasta około 60 epoki, z około 0,25, do aż 1,75.



Rysunek 5.57: Wyniki testów dla zmodyfikowanego modelu podstawowego, liczba wierzchołków  $n = 6$

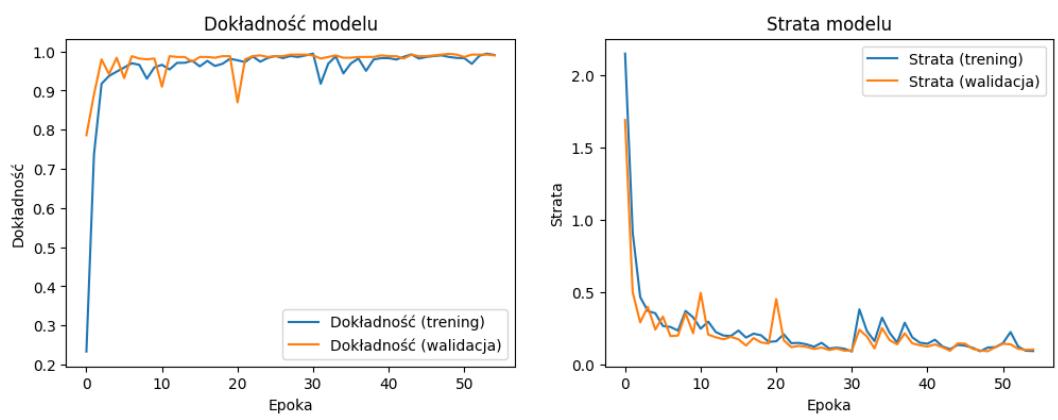
Powyższa sytuacja wskazuje na poważny problem z wydajnością modelu, np. przetrenowanie lub problem techniczny w procesie treningu - być może był to zbyt niski współczynnik procesu uczenia, który spowodował, że model „przyspieszył” w ostatnich epokach.

### Zmodyfikowany poprawiony model podstawowy uczony na grafach sześciowierzchołkowych

Wyniki uzyskane z procesu nauki poprzedniego modelu sugerują zmniejszenie liczby epok do około 55. Taka modyfikacja została zastosowana, a test przeprowadzono ponownie.

Po skróceniu procesu uczenia, dokładność modelu nie spada gwałtownie w końcowych epokach. Dokładność szybko rośnie na początku treningu i osiąga wysokie wartości już po kilku epokach. Dokonuje się również stabilizacja w okolicach 98%-100%, co wskazuje na dobre dopasowanie modelu do danych. Fluktuacje walidacji są niewielkie i nie wskazujące na problemy z modelem.

Strata, po modyfikacji, już nie wzrasta gwaltownie pod koniec procesu uczenia. Oba typy strat maleją po kilku pierwszych epokach i pozostają na niskim poziomie (0,1 - 0,2). Model wygląda więc na dobrze dopasowany do danych treningowych, jak i walidacyjnych.



Rysunek 5.58: Wyniki testów dla poprawionego zmodyfikowanego modelu podstawowego, liczba wierzchołków  $n = 6$

Wygląda na to, że model efektywnie klasyfikuje grafy ze zbioru walidacyjnego. Zwracając uwagę na powyższe wskaźniki, można spodziewać się dobrej jakości przewidywań. Model działa bardzo dobrze, z minimalnymi fluktuacjami, które mogą być naturalne przy tego typu zadaniach.

```
1/1 0s 21ms/step
|- ./test_graphs\drawn\path-6.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 0s 21ms/step
|- ./test_graphs\drawn\path-7.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 95.28 procent.
1/1 0s 21ms/step
|- ./test_graphs\drawn\path-8.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 86.42 procent.
1/1 0s 22ms/step
|- ./test_graphs\drawn\path-9.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1 0s 21ms/step
|- ./test_graphs\drawn\tree-binary-1.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 94.86 procent.
1/1 0s 20ms/step
|- ./test_graphs\drawn\tree-binary-2.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 100.00 procent.
1/1 0s 20ms/step
|- ./test_graphs\drawn\tree-binary-3.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.99 procent.
1/1 0s 20ms/step
|- ./test_graphs\drawn\tree-binary-4.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.99 procent.
1/1 0s 19ms/step
|- ./test_graphs\drawn\tree-binary-5.png -| najprawdopodobniej należy do klasy |- tree-binary -| z prawdopodobieństwem 99.90 procent.
1/1 0s 20ms/step
|- ./test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1 0s 21ms/step
|- ./test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.98 procent.
1/1 0s 20ms/step
|- ./test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.99 procent.
1/1 0s 21ms/step
|- ./test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1 0s 20ms/step
|- ./test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 87.29 procent.
1/1 0s 19ms/step
|- ./test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 97.58 procent.
```

Rysunek 5.59: Klasyfikacja obrazów zewnętrznych dla poprawionego zmodyfikowanego modelu podstawowego, liczba wierzchołków  $n = 6$



Rysunek 5.60: Wizualizacja klasyfikacji obrazów zewnętrznych dla poprawionego zmodyfikowanego modelu podstawowego, liczba wierzchołków  $n = 6$

Zmodyfikowany model poprawnie sklasyfikował prawie 42% grafów zewnętrznych. Jest to wynik gorszy o 3 punkty procentowe od wariantu tego samego modelu z mniejszą liczbą filtrów w warstwach Conv2D oraz niższym wskaźnikiem dropout. Wynik jednak jest wciąż zadowalający, bo czterdziestodwuprocentowa pewność, jest wynikiem lepszym niż przypisywanie klas losowo.



Rysunek 5.61: Klasyfikacja przykładowego grafu zewnętrznego przez zmodyfikowany poprawiony model podstawowy uczony na grafach sześciowierzchołkowych. Przypisana klasa to drzewo binarne z 100% pewnością.

Ponownie, to prostszy model okazał się tym, który dokonuje lepszej generalizacji na nowe dane i lepiej zapamiętuje ogólne wzorce. Sugeruje to, że zbyt duże skomplikowanie modelu do zadania o niewielkiej złożoności, przynosi korzyści odwrotne od zamierzonych.

# Podsumowanie i wnioski końcowe

W przeprowadzonym badaniu przedstawiono proces budowy modeli sztucznych sieci neuronowych (ANN) do rozpoznawania grafów, a także analizę ich skuteczności zarówno na danych syntetycznych, jak i rzeczywistych. Sieci neuronowe zostały wyszkolone na wygenerowanych rysunkach grafów, co umożliwiło stworzenie efektywnych modeli do klasyfikacji. Następnie modele te przetestowano na rzeczywistych rysunkach grafów, aby zweryfikować ich praktyczną użyteczność oraz ocenić, w jakim stopniu ich teoretyczna dokładność przekłada się na realne zastosowania.

Wyniki eksperymentów pokazały, że modele dobrze radzą sobie z zadaniem klasyfikacji grafów, osiągając wysokie wartości dokładności zarówno w trakcie treningu, jak i walidacji. Dokładność teoretyczna modeli była bardzo zbliżona do wyników na rzeczywistych danych testowych, co potwierdza skuteczność zastosowanych architektur ANN. W referacie szczegółowo opisano także metody generowania grafów, skrypty użyte do treningu modeli, oraz charakterystykę samych sieci neuronowych.

Pomimo ogólnie dobrych wyników, zaobserwowano drobne różnice pomiędzy wynikami na danych syntetycznych i rzeczywistych, co sugeruje, że dalsze badania mogą skupić się na udoskonaleniu procesu generowania grafów lub wprowadzeniu dodatkowych mechanizmów regularyzacyjnych w celu poprawy zdolności generalizacji modeli. Przedstawione w referacie badania wskazują na wysoki potencjał sztucznych sieci neuronowych w zadaniach związanych z analizą i klasyfikacją grafów, co otwiera nowe możliwości w kontekście przetwarzania danych graficznych w praktycznych zastosowaniach.

W testach zastosowano naukę na maksymalnie 75 epokach. Analizując krzywe dokładności i straty modeli, można dojść do wniosku, że dłuższe uczenie nie przyniosłoby pozytywnych skutków lub znikome pozytywne. Przeprowadzono również testy na mniejszych liczbach epok, lecz wartości takie jak 10, czy 20, są za małe, by po-

prawnie nauczyć większość modeli. Modeli osiągały dość wysokie dokładności już po kilku pierwszych epokach, ale zdecydowana większość z nich, z biegiem kolejnych epok, nabierała jeszcze lepszej dokładności.

W przypadku jednego modelu, konieczne było zmniejszenie liczby epok do 55, ze względu na powstające problemy z procesem nauczania, po owej epoce.

Uczenie modeli z wykorzystaniem stałej liczby wierzchołków grafów okazało się bardziej skuteczne niż nauka na rysunkach grafów o zmiennej liczbie wierzchołków.

Skomplikowanie modelu i zastosowane techniki optymalizacyjne rzadko przekładały się na zwiększoną realną dokładność modelu.

W przypadku uczenia najbardziej zmodyfikowanych modeli, to grafy pełne najczęściej dominowały cały zbiór danych, przez co modele te testach na danych zewnętrznych, klasyfikowały większość testowych grafów rysowanych odręcznie jako właśnie grafy pełne.

# Bibliografia

- [1] Arikawa K.: *Graph Theory Teaches Us Something About Grammaticality*. The Prague Bulletin of Mathematical Linguistics No. 112, 2019, pp. 55-82.
- [2] Balaban A.T.: *Applications of Graph Theory in Chemistry*. Department of Organic Chemistry, Polytehnic Institute. 76206 Bucharest, Roumania, 1985.
- [3] Brusatte, S., Carr, T.: *The phylogeny and evolutionary history of tyrannosauroid dinosaurs*. Sci Rep 6, 20252 (2016). <https://doi.org/10.1038/srep20252>.
- [4] Chung M.K.: *Graph Theory in Brain Networks*, University of Wisconsin-Madison, 2021.
- [5] Erciyes K.: *Graph-Theoretical Analysis of Biological Networks: A Survey*. Computation 2023, 11, 188, DOI: <https://doi.org/10.3390/computation11100188>.
- [6] Fenner M.E.: *Uczenie maszynowe w Pythonie dla każdego*. Helion SA, Gliwice 2020.
- [7] Géron A.: *Uczenie maszynowe z użyciem Scikit-Learn i TensorFlow*. Helion SA, Gliwice 2020.
- [8] Goodfellow I., Bengio Y., Courville A.: *Deep Learning*, MIT Press 2016.
- [9] Harary F., Norman R.Z.: *Graph Theory as a Mathematical Model in Social Science*, Research Center for Group Dynamics, University of Michigan, 1953.
- [10] Rodak K., Kokot I., Kratz E.W.: *Caffeine as a Factor Influencing the Functioning of the Human Body-Friend or Foe?* Nutrients. 2021 Sep 2;13(9):3088.
- [11] Seenappa M.G.: *Graph Classification using Machine Learning Algorithms*. Master's Projects. 725, San Jose State University 2019, DOI: <https://doi.org/10.31979/etd.b9pm-wpng>.

- [12] Umami M.H., Prihandini R.M., Agatha A.B.: *Application of Graph Theory to Social Network Analysis*, Department of Mathematics Educations, University of Jember, Jember, Indonesia, 2024.
- [13] L.J.P. van der Maaten, Hinton G.E.: *Visualizing High-Dimensional Data Using t-SNE*. Journal of Machine Learning Research 9(Nov):2579-2605, 2008.
- [14] Wilson R.J.: *Wprowadzenie do teorii grafów*. PWN, Warszawa 2012.
- [15] Włoch A., Włoch I.: *Matematyka dyskretna. Podstawowe metody i algorytmy teorii grafów*. Oficyna Wydawnicza Politechniki Rzeszowskiej, Rzeszów 2008.
- [16] Wojciechowski J., Pieńkosz K.: *Grafy i sieci*. PWN, Warszawa 2013.
- [17] <https://www.kaggle.com/datasets/robikscube/flight-delay-dataset-20182022>. Dostęp 06.01.2024.
- [18] <https://cran.r-project.org/web/packages/igraph/index.html>. Dostęp 10.03.2024.
- [19] <https://developers.google.com/machine-learning>. Dostęp 20.07.2024.
- [20] <https://www.ibm.com/topics>. Dostęp 20.07.2024.
- [21] <https://www.python.org/>. Dostęp 07.08.2024.
- [22] <https://www.r-project.org/>. Dostęp 07.08.2024.
- [23] <http://student.krk.pl/026-Ciosek-Grybow/rodzaje.html>. Dostęp 26.03.2024.
- [24] [https://www.tensorflow.org/api\\_docs](https://www.tensorflow.org/api_docs). Dostęp 21.07.2024.
- [25] <http://wms.mat.agh.edu.pl/~md/ang-pol.pdf>. Dostęp 29.03.2024.

# Załączniki

- Skrypt generujący obrazy grafów
- Skrypty testowe z modelem podstawowym
- Skrypty testowe z modelem, z walidacją krzyżową
- Skrypty testowe z modelem dostosowanym do nauki grafów o różnej liczbie wierzchołków
- Skrypty testowe z modelem, z walidacją krzyżową, dostosowanym do nauki grafów o różnej liczbie wierzchołków
- Zbiór zewnętrznych grafów testowych

Wersja online dokumentu, dane oraz kod źródłowy w języku Python i R dostępne na stronie: [https://github.com/gabriellichacz/graph\\_classification](https://github.com/gabriellichacz/graph_classification)

**POLITECHNIKA RZESZOWSKA**  
**im. Ignacego Łukasiewicza**  
**WYDZIAŁ MATEMATYKI I FIZYKI STOSOWANEJ**

Rzeszów, 2024

**STRESZCZENIE PRACY DYPLOMOWEJ**

**Tytuł:** Rozpoznawanie rysunków grafów

**Autor:** Gabriel Lichacz

**Promotor:** dr Paweł Bednarz

**Słowa klucze:** Uczenie maszynowe, Grafy, Sieci Neuronowe, Python

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis viverra ut sem sed lobortis. Morbi id faucibus diam, dictum scelerisque lectus. Nulla tempor eros non augue sagittis mollis. Nam ut sapien in orci aliquam lacinia quis eu nibh. Suspendisse vestibulum volutpat dui, ultricies consectetur ligula tempor ac. Nullam blandit eget quam vitae imperdiet. Phasellus luctus augue aliquet quam aliquam vulputate.

---

**RZESZOW UNIVERSITY OF TECHNOLOGY**  
**FACULTY OF MATHEMATICS AND APPLIED PHYSICS**

Rzeszów, 2024

**DIPLOMA THESIS ABSTRACT**

**Title:** Recognition of graphs

**Author:** Gabriel Lichacz

**Supervisor:** Paweł Bednarz PhD

**Key words:** Machine learning, Graphs, Neural Networks, Python

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis viverra ut sem sed lobortis. Morbi id faucibus diam, dictum scelerisque lectus. Nulla tempor eros non augue sagittis mollis. Nam ut sapien in orci aliquam lacinia quis eu nibh. Suspendisse vestibulum volutpat dui, ultricies consectetur ligula tempor ac. Nullam blandit eget quam vitae imperdiet. Phasellus luctus augue aliquet quam aliquam vulputate.