WYDZIAŁ
MATEMATYKI
I FIZYKI STOSOWANEJ
POLITECHNIKI RZESZOWSKIEJ

Gabriel Lichacz

Rozpoznawanie rysunków grafów

Praca dyplomowa magisterska

Opiekun pracy:
dr Paweł Bednarz

Rzeszów, 2024

Spis treści

Wykaz symboli

C_n - graf cykliczny

D - digraf

$E(G)$ - zbiór krawędzi grafu G

$G(V_1, V_2)$ - graf dwudzielny

G - graf

$V(G)$ - zbiór wierchołków grafu G

K_n - graf pełny

m - liczba krawędzi

n - liczba wierzchołków

N_n - graf pusty

P_n - graf liniowy

u, v, w, z - wierchołki grafu G

W_n - koło

1. Wprowadzenie

Grafy w matematyce można zdefiniować jako graficzną reprezentację danych, która przedstawia wartości w uporządkowany sposób, najczęściej w relacji do samych siebie. Teoria grafów to dziedzina obejmująca badanie grafów. Jest to bardzo ważne narzędzie w „*dziedzinach od rachunku operacyjnego, chemii, po genetykę, lingiwistykę oraz od elektroniki i geografii po socjologię i architekturę*"[1]. Grafy dają możliwość zobrazowania pewnych modeli, co jest szczególnie korzystne w rozpoznawaniu wzorców.

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 miejscach.

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.

1.1. Przegląd literatury

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

2. Teoria grafów

2.1. Definicje

Definicje zostały zaczerpnięte z literatury, z pozycji [1], [2] oraz [3].

Definicja 1. Grafem nieskierowanym, skończonym G nazywamy parę (V, E), gdzie $V = V(G)$ jest zbiorem skończonym, niepustym, natomiast $E = E(V)$ jest rodziną mogących się powtarzać dwuelementowych podzbiorów niekoniecznie różnych elementów ze zbioru

V. Zbiór $V(G)$ nazywamy zbiorem wierzchołków (lub węzłami), a elementy tego zbioru nazywamy wierzchołkami i oznaczamy symbolami: x, y, x_i, y_i, 1, 2, ... Zbiór $E(G)$ nazywamy zbiorem krawędzi grafu G. Mówimy, że krawędź $\{v, w\}$ łączy wierzchołki v i w, i na ogół oznaczamy ją krócej symbolem vw.

Definicja 2. W wielu zagadnieniach nazwy wierzchołków są nieistotne, więc je pomijamy i mówimy wtedy, że graf jest nieoznakowany.

Definicja 3. Jeżeli w grafie G istnieją co najmniej dwie krawędzie $\{x, y\}$, to krawędź tę nazywamy krawędzią wielokrotną.

Definicja 4. Krawędź $\{x, x\}$ w grafie G nazywamy pętlą.

Definicja 5. Graf mający krawędzie wielokrotne nazywamy multigrafem.

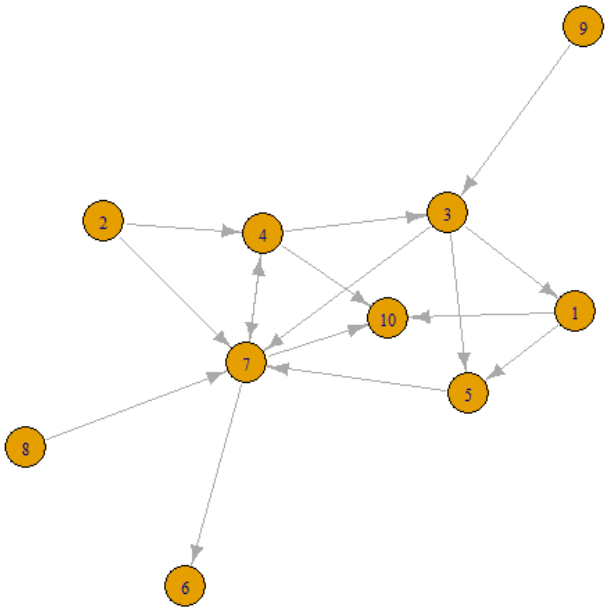
Definicja 6. Graf, który nie ma krawędzi wielokrotnych i pętli, nazywamy grafem prostym.

Definicja 7. Graf zawierający pętle nazywamy pseudografem.

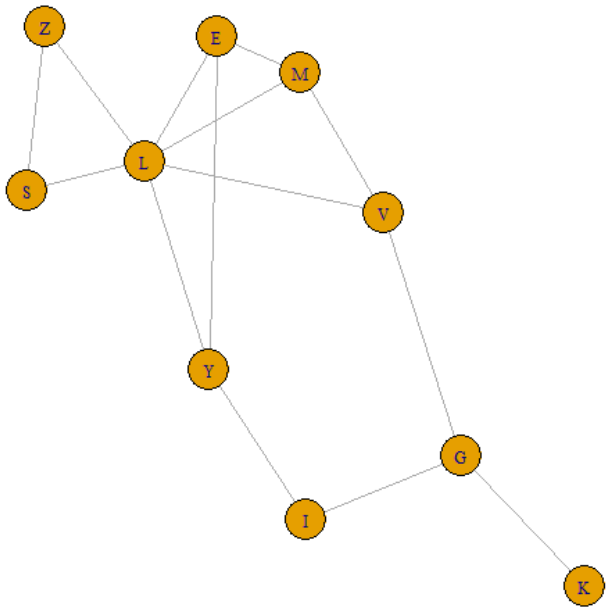
Definicja 8. Graf G taki, że $E(G) = \emptyset$, nazywamy grafem pustym. Jeżeli $|V(G)| = n$, to graf pusty oznaczony symbolem N_n. Każdy wierchołek grafu pustego jest wierchołkiem izolowanym.

Definicja 9. Graf prosty G taki, że każde dwa wierzchołki są sąsiednie, nazywamy grafem pełnym. Jeżeli $|V(G)| = n$, to graf pełny oznaczamy K_n.

Definicja 10. Graf G, którego zbiór wierchołków można podzielić na dwa rozłączne, niepuste podzbiory V_1 i V_2 tak, że jeżeli $\{x, y\} \in E(G)$, to $x \in V_1 \vee y \in V_2$ nazywamy grafem dwudzielnym.



Rysunek 2.1: Przykład grafu skierowanego



Rysunek 2.2: Przykład grafu nieskierowanego

3. Uczenie maszynowe

Uczenie maszynowe, znane również jako machine learning, to specjalistyczna gałąź sztucznej inteligencji, która koncentruje się na konstruowaniu modeli i algorytmów umożliwiajcych 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.

3.1. Rodzaje uczenia maszynowego

Według [5], 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.

- 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.

- Uczenie przez wzmacnianie (ang. reinforcement learning) to przypadek, gdzie model uczy się poprzez interakcję ze swoim otoczeniem, podejmując decyzje, które maksymalizują pewną nagrodę. Przykłady zastosowań to robotyka i gry komputerowe.

3.2. Proces uczenia maszynowego

Proces uczenia maszynowego można podzielić na kilka etapów, które są niezbędne do stworzenia skutecznego modelu zdolnego do samodzielnej nauki na podsta-

wie 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. Najczęściej, podział na zbiory dokonuje się w proporcjach 70-80Jest 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 hyperparametrów 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, zautomatyzoawanie 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 no-

wych danych, aby utrzymać jego dokładność i skuteczność, ciągłe monitorowanie, aby zapewnić, że model działa zgodnie z oczekiwaniami i nie występują niepożądane zachowania. Proces uczenia maszynowego jest iteracyjny i wymaga ciągłej interakcji między danymi, modelem i wynikami, aby osiągnąć optymalne rezultaty.

4. Wykorzystywane technologie

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

4.1. Język R

Rysunek 4.3: Logo R

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.

4.2. Język Python

Rysunek 4.4: Logo Python

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.

4.3. 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.

5. Testy

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.

Stworzone zostały 3 modele: - wytrenowany na danych ze stałą liczbą wierchołków - wytrenowany na danych ze stałą liczbą wierchołków oraz walidacją krzyżową - wytrenowany na danych ze zmienną liczbą wierzchołków

5.1. Generacja danych

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

W testach wykorzystane zostały wszystkie typy grafów. Każdy z nich, czyli dana liczba wierchołków i typ grafu, wygenerowany został w liczbie 500 sztuk oraz ze stałą krzywizną krawędzi, wynoszącą 0,3.

5.2. Dane zewnętrzne

Obrazy testowe, które nazwane są tutaj danymi zewnętrznymi, są rysunkami grafów pochodzącymi 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.

5.3. Opis skryptu

5.3.1. Przygotowanie

Skrypt rozpoczyna się od przygotowania środowiska do trenowania modelu. Najpierw ustawia ścieżki do katalogów z wygenerowanymi grafami oraz do katalogów na dane treningowe i walidacyjne. Następnie sprawdza, czy te katalogi istnieją, a jeśli nie, tworzy je. Dalej, definiuje parametry dotyczące wielkości obrazów oraz wielkości partii danych, które będą używane podczas treningu. Dla każdej wartości liczby wierzchołków ustawia ścieżkę do katalogu z wygenerowanymi grafami, pobiera listę podkatalogów oraz obrazów w każdym z nich, a następnie dzieli obrazy na zestawy treningowe i wali-

dacyjne w stosunku 80:20. Tworzy odpowiednie katalogi dla tych zestawów, jeśli jeszcze nie istnieją, i kopiuje obrazy do właściwych lokalizacji, segregując je na treningowe i walidacyjne.

5.3.2. Model

Na początku dane zostały przygotowane ze zbioru obrazów, znajdującego się w katalogu lokalnym. Dokonany został podział na zbiory treningowe i walidacyjne. Dla każdego przejścia walidacji krzyżowej, dane zostały podzielone inaczej. Po wczytaniu danych, zostały przeskalowane i przeskztałcone do odcieni szarości.

Model sieci neuronowej został zdefiniowany jako sekwencyjny stos warstw. Pierwsza warstwa to warstwa Rescaling, która normalizuje wartości pikseli do zakresu [0, 1]. Następne trzy warstwy to Conv2D z wybraną liczbą filtrów, z których każda jest poprzedzona warstwą MaxPooling2D. Warstwy konwolucyjne używają różnych funkcji akytwacji, np. ReLU. Po warstwach konwolucyjnych znajduje się warstwa Flatten, która przekształca mapy cech 2D w wektor 1D. Następnie dodana jest w pełni połączona (Dense) warstwa z wybraną liczbą jednostek i wybraną funkcją aktywacji, wraz z warstwą dropout. Zastosowana jest tam również regularyzacja L2 (zmniejszanie wag) z ustaloną siłą regularyzacji wynoszącą. Warstwa wyjściowa zawiera tyle jednostek, ile występuje klas. Zależnie od danego testu, może być to różna liczba.

Dla standaryzacji danego testu ustalono K-Fold z liczbą podziałów równą 5. Dla wszystkich warstw wybrana została funkcja aktywacji ReLU. Liczba epok wyniosła 75. W przypadku warstw konwolucyjnych, wybrano 32 filtry, a dla warstwy w pełni połączonej zastosowano 128 jednostek. Regularyzacja została zastosowana z siłą 0,01, a współczynnik dropout - 0,2.

5.3.3. Wyniki

Po wytrenowaniu modelu, skrypt wizualizuje dokładność i stratę 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.

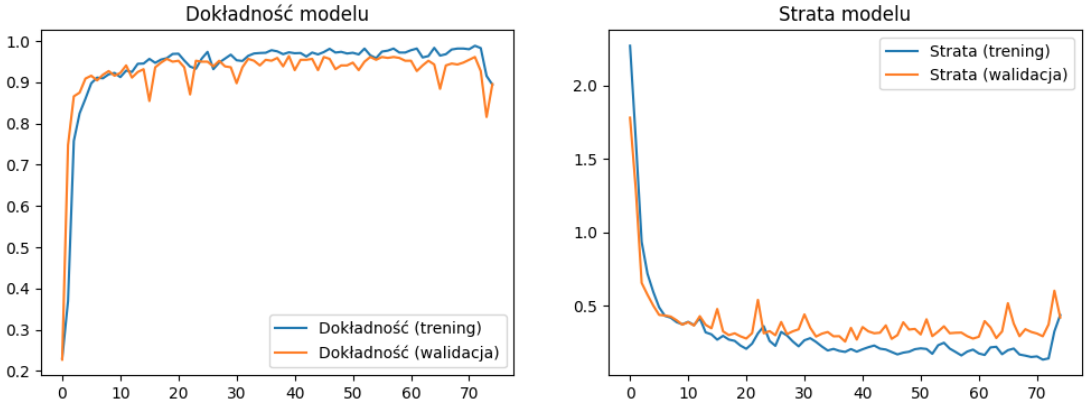
5.3.4. Testy na danych zewnętrznych

Po wyświetleniu dokładnosci 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, przeskalowuje do odpowiedniego rozmiaru i konwertuje do skali szarości Następnie model przewiduje klasę obrazu, a wynik jest wyświetlany w konsoli.

5.4. Testy modeli

5.4.1. Model podstawowy



Rysunek 5.5: Wyniki testów dla modelu podstawowego

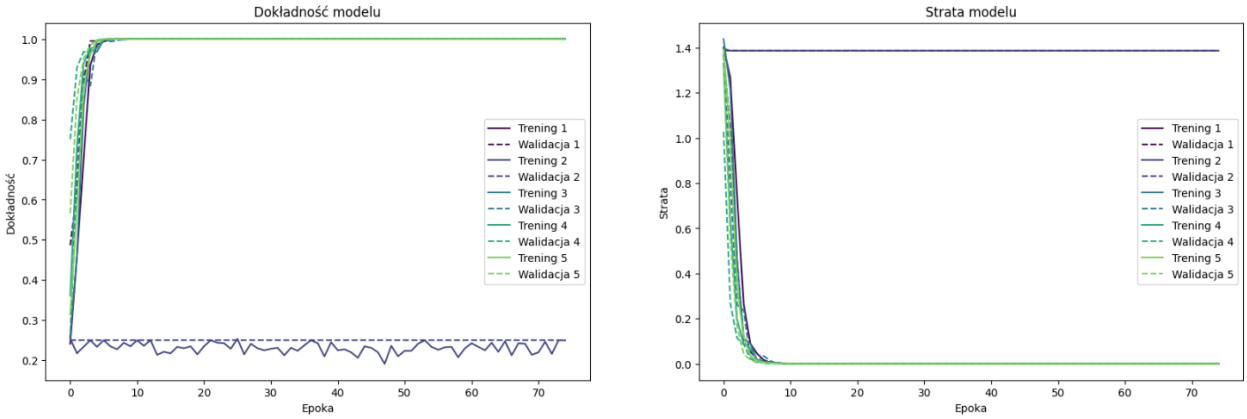
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\connected-drawn-1.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step
|- test_graphs\drawn\cycle-drawn-1.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\drawn\full-drawn-1.jpg -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\full-drawn-2.jpg -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\full-drawn-3.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
|- test_graphs\drawn\path-drawn-1.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\path-drawn-2.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\drawn\path-drawn-3.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 56.76 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- bipartite -| z prawdopodobieństwem 65.54 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 99.97 procent.
```

Rysunek 5.6: Klasyfikacja obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków

5.4.2. Model z walidacją krzyżową

W przypadku standardowego modelu z walidacją krzyżową model bardzo szybko uległ przeuczeniu. Już po szóstej iteracji dokładność na zbiorze walidaycjnym wyniosła

100%, co nie jest realistycznie możliwe. Została podjęta próba ograniczenia overfittingu poprzez zwiększenie zbioru danych, zmiany liczby epok w modelu oraz manipulacji współczynnikami dropout i regularyzacji. W każdym przypadku model zwracał niezadowalające wyniki wynoszące 100% po jednej z początkowych iteracji.



Rysunek 5.7: Wyniki testów dla modelu z walidacją krzyżową

Z powodu przeuczenia model nie radził sobie z zewnętrznymi obrazkami testowymi. Większość grafów określił jako grafy pełne, co nie jest zgodne ze stanem rzeczywistym.

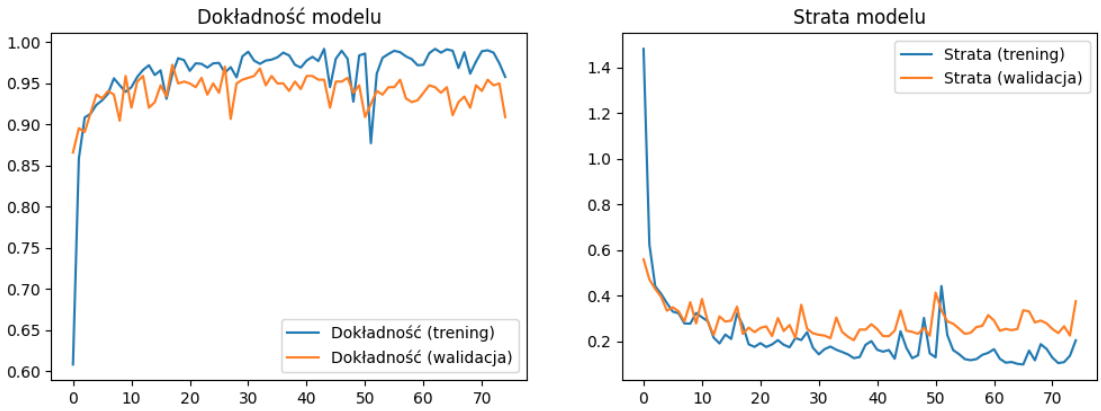
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 104ms/step
|- test_graphs\drawn\connected-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
|- test_graphs\drawn\cycle-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
|- test_graphs\drawn\full-drawn-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\drawn\full-drawn-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\drawn\full-drawn-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\drawn\path-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
|- test_graphs\drawn\path-drawn-2.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\drawn\path-drawn-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
|- test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 100.00 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 |- empty -| z prawdopodobieństwem 99.93 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
|- test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- empty -| z prawdopodobieństwem 100.00 procent.
```

Rysunek 5.8: Klasyfikacja obrazów zewnętrznych dla modelu z walidacją krzyżową

5.4.3. Model ze zmienną liczbą wierzchołków

Najlepsze wyniki pod względem rozpoznawania zewnętrznych obrazków testowych oraz realistycznej dokładności na zbiorze walidacyjnym, zostały uzyskane przy

użyciu modelu sieci neuronowej uczonej na rysunkach grafów z różną liczbą wierzchołków. Było to odpowiednio 4, 5, 6 oraz 7 wierzchołków.



Rysunek 5.9: Wyniki testów dla modelu ze zmienną liczbą wierzchołków

Model nie poradził sobie zbyt dobrze z obrazami zewnętrznymi, lecz znacznie lepiej niż model z walidacją krzyżową. Poprawnie wskazanych klas grafów było 5 z 14 wszystkich rysunków. Mimo, że model jest w stanie poprawnie określić niektóre typy grafów poprawnie, wciąż jest to dokładność niższa niż 50%.

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 120ms/step
|- test_graphs\drawn\connected-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 61.37 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\drawn\cycle-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 64.43 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\drawn\full-drawn-1.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\full-drawn-2.jpg -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 99.80 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\full-drawn-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
|- test_graphs\drawn\path-drawn-1.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 62.29 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\drawn\path-drawn-2.png -| najprawdopodobniej należy do klasy |- connected -| z prawdopodobieństwem 99.46 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\drawn\path-drawn-3.png -| najprawdopodobniej należy do klasy |- full -| z prawdopodobieństwem 69.90 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\generated\cycle-45.png -| najprawdopodobniej należy do klasy |- bipartite -| z prawdopodobieństwem 55.28 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
|- test_graphs\generated\full-113.png -| najprawdopodobniej należy do klasy |- bipartite -| z prawdopodobieństwem 56.59 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\generated\path-78.png -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
|- test_graphs\internet\internet-cycle-1.png -| najprawdopodobniej należy do klasy |- cycle -| z prawdopodobieństwem 100.00 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\internet\internet-full-1.jpg -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 99.71 procent.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
|- test_graphs\internet\internet-full-2.jpg -| najprawdopodobniej należy do klasy |- path -| z prawdopodobieństwem 39.16 procent.
```

Rysunek 5.10: Klasyfikacja obrazów zewnętrznych dla modelu z walidacją krzyżową

5.4.4. Model ze zmienną liczbą wierzchołków i walidacją krzyżową

Rysunek 5.11: Wyniki testów dla modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

Rysunek 5.12: Klasyfikacja obrazów zewnętrznych dla modelu ze zmienną liczbą wierzchołków i walidacją krzyżową

5.5. Wnioski

W przypadku uczenia modeli z wykorzystaniem grafów pełnych, najczęściej dominowały one cały zbiór danych, przez co modele w kolejnych testach klasyfikowały większość testowych grafów rysowanych odręcznie jako właśnie grafy pełne.

Testy z wykorzystaniem stałej liczby wierzchołków grafów okazały się mniej owocne niż testy z rysunkami grafów o zmiennej liczbie wierchołków.

Wystąpiła tendencja do niepoprawnego określania innych grafów, grafami dwudzielnymi, jeśli takie znajdowały się w zbiorze danych uczących.

6. Podsumowanie i wnioski końcowe

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Załączniki

- Skrypt generujący obrazy grafów
- Skrypt testowy z modelem podstawowym
- Skrypt testowy z modelem, z walidacją krzyżową
- Skrypt testowy z modelem dostosowanym do nauki grafów o różnej liczbie wierzchołków
- Skrypt testowy z modelem, z walidacją krzyżową, dostosowanym do nauki grafów o różnej liczbie wierzchołków

Literatura

[1] Wilson R.J.: Wprowadzenie do teorii grafów. PWN, Warszawa 2012.

[2] Włoch A., Włoch I.: Matematyka dyskretna. Oficyna Wydawnicza Politechniki Rzeszowskiej, Rzeszów 2008.

[3] Wojciechowski J., Pieńkosz K.: Grafy i sieci. PWN, Warszawa 2013.

[4] Fenner M.E.: Uczenie maszynowe w Pythonie dla każdego. Helion SA, Gliwice 2020.

[5] Géron A.: Uczenie maszynowe z użyciem Scikit-Learn i TensorFlow. Helion SA, Gliwice 2020.

[6] Seenappa M.G.: Graph Classification using Machine Learning Algorithms. Master's Projects. 725, DOI: https://doi.org/10.31979/etd.b9pm-wpng, San Jose State University 2019.

[7] http://student.krk.pl/026-Ciosek-Grybow/rodzaje.html. Dostęp 26.03.2024.

[8] http://wms.mat.agh.edu.pl/~md/ang-pol.pdf. Dostęp 29.03.2024.

[9] https://cran.r-project.org/web/packages/igraph/index.html. Dostęp 10.03.2024.

POLITECHNIKA RZESZOWSKA im. I. Łukasiewicza Rzeszów, 2024
Wydział Matematyki i Fizyki Stosowanej

STRESZCZENIE PRACY DYPLOMOWEJ MAGISTERSKIEJ

ROZPOZNAWANIE RYSUNKÓW GRAFÓW

Autor: Gabriel Lichacz, nr albumu: 164174
Opiekun: dr Paweł Bednarz
Słowa kluczowe: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

Treść streszczenia po polsku

RZESZOW UNIVERSITY OF TECHNOLOGY Rzeszow, 2024
The Faculty of Mathematics and Applied Physics

MSC THESIS ABSTRACT

RECOGNITION OF GRAPHS

Author: Gabriel Lichacz, nr albumu: 164174
Supervisor: Paweł Bednarz PhD
Key words: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

Treść streszczenia po angielsku