Dokumentacja

1. Instrukcja obsługi

Aby uruchomić projekt, należy wykonać w terminalu następujące komendy:

- pip install -r requirements.txt
- sudo apt-get install python3-tk python3-pil python3-pil.imagetk
- python3 GUI.py

2. Opis projektu

a. Interfejs graficzny

Do wykonania GUI skorzystaliśmy z biblioteki Tkinter. Jako tło użyliśmy labela, który zawiera zdjęcie naszego autorstwa. Również pole na wybrane zdjęcie jest labelem, któremu podczas wybierania pliku zmieniamy wartość image na ta wybrana przez użytkownika. Nasz interfejs posiada również listę do wyboru z konkretnych kategorii, która wybiera użytkownik i porównuje z tym, który obliczył nasz algorytm (zdecydowanie najlepiej by było gdyby użytkownik wybierał zdjęcie należące do jednej z tych kategorii).

Wynik wyświetlany jest na ekranie w miejsce labela który wcześniej jest pusty (przez co widać taki szary pasek na początku - wynika to z tego, że w tkinterze domyślnie tło labelu jest szare, a zrobienie go przezroczystym jest zależne od systemu operacyjnego, w związku z czym poświęciliśmy wygląd na rzecz kompatybilności). Posiadamy też przycisk wyboru pliku, który jak sama nazwa wskazuje wyświetla okienku umożliwiajace wybór zdjęcia do analizy (program posiada buga mianowicie jeśli wybierzemy zdjęcie a potem będziemy chcieli ponownie wybrać jakieś inne, ale zamiast tego w oknie wyboru pliku damy “cancel” to zdjecie zamiast pozostać takie same wraca do czarnego tła (które też jest

zdjęciem), gdybyśmy tak nie robili to poprzednio wybrane zdjęcie znika, a nie udało się nam inaczej obejść tego problemu).

Ostatecznie posiadamy też przycisk do wysłania żądania analizy, który wywołuje skrypty określające z jakiej kategorii jest to obiekt i zwraca odpowiedź w formie tekstowej.

b. Rozpoznawanie zdjęć

Do rozpoznawania zdjęć wykorzystaliśmy bibliotekę Tensorflow, a model trenowaliśmy oraz testowaliśmy przy użyciu zbioru danych Kaggle "Intel Image Classification". Tworząc program wzorowaliśmy się na tutorialach (linki w źródłach) dotyczących klasyfikacji danych.

Model składa się z trzech warstw konwolucyjnych (kolejno 16, 32 i 64 jednostki) oraz tzw. warstwy gęstej (128 jednostek), przez którą "przechodzi" wynik tuż przed warstwą określającą kategorię (5 jednostek - liczba kategorii). Wszystkie warstwy korzystają z funkcji aktywacji "ReLu", a model jest kompilowany przy użyciu optymalizatora "adam".

Testowaliśmy rożne wielkości zbiorów (batch size) oraz cykli (epoch), lecz najlepsze wyniki osiągnęliśmy przy ustawieniu: *batch_size=32, epochs=12.* Ten właśnie model został zapisany do ewaluacji zdjęć w naszej aplikacji.

Program został podzielony na funkcje, które umożliwiły stworzenie i wytrenowanie opisanego wyżej modelu, jego zapisanie do pliku, załadowanie z pliku oraz ewaluację przykładowych danych wejściowych przez wytrenowany już model.

3. Podział obowiązków

- Interfejs użytkownika - Jędrzej Szostak
- Rozpoznawanie zdjęć - Bartosz Bardian
- Przygotowanie merytoryczne do projektu, dokumentacja - Jędrzej Szostak, Bartosz Bardian

4. Źródła

- https://www.tensorflow.org/api_docs/python/tf/all_symbols
- https://stackoverflow.com/
  Zapisywanie modelu:
- https://www.tensorflow.org/tutorials/keras/save_and_load
  Tutorial do klasyfikacji:
- https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/images/classification.ipynb