# Dokumentacja do projektu na przedmiot Analiza Obrazów

Jan Wojdylak, Bartosz Balawender

2021/2022

## 1.Opis projektu

Naszym celem było napisanie aplikacji, która jest wstanie sklasyfikować ubranie znajdujące się na podanym obrazie na podstawie wytrenowanej sieci neuronowej. Naszą cechą są obrazy, które są tabelą w *Numpy* oraz mają rozmiar 28x28 pikseli obraz pokazywany jest w odcieniach szarości. Trenujemy naszą sieć neuronową na podstawie gotowej bazy ***„Fashion-MNIST”*** – jest to zbiór danych przygotowany przez zespół *Zalando* , znajdujący się w bibliotece *tensorflow* . Nasza aplikacja przewiduje jaka część garderoby znajduję się na przedstawionym w danym momencie w aplikacji obrazku. Naszą kategorią, którą będziemy przewidywać są liczby całkowite od 0 do 9, każda z liczb odpowiada jednej klasie ubrań:

| Label | Class |
|---|---|
| 0 | T-shirt/top |
| 1 | Trouser |
| 2 | Pullover |
| 3 | Dress |
| 4 | Coat |
| 5 | Sandal |
| 6 | Shirt |
| 7 | Sneaker |
| 8 | Bag |
| 9 | Ankle boot |

**Rys.1.Tabela klas ubrań**

## 2.Narzędzia potrzebne do uruchomienia projektu

Program został napisany w pythonie z wykorzystaniem następujących bilbiotek :

- **tensorflow**
- **tkinter**
- **matplotlib**

Aby uruchomić projekt należy zainstalować bibliotekę **tensorflow** korzystając polecenia :

**Dla pythona 3.9**

python -m pip install --upgrade https://storage.googleapis.com/tensorflow/windows/cpu/tensorflow_cpu-2.7.0-cp39-cp39-win_amd64.whl

**Dla pythona 3.8**

python -m pip install –upgrade https://storage.googleapis.com/tensorflow/windows/cpu/tensorflow_cpu-2.7.0-cp38-cp38-win_amd64.whl

**Dla pythona 3.7**

python -m pip install --upgrade https://storage.googleapis.com/tensorflow/windows/cpu/tensorflow_cpu-2.7.0-cp36-cp36m-win_amd64.whl

Oraz bibliotekę **matplotlib** *pip install matplotlib* w miejscu gdzie mamy zainstalowanego pythona.

# 3.Obsługa GUI projektu

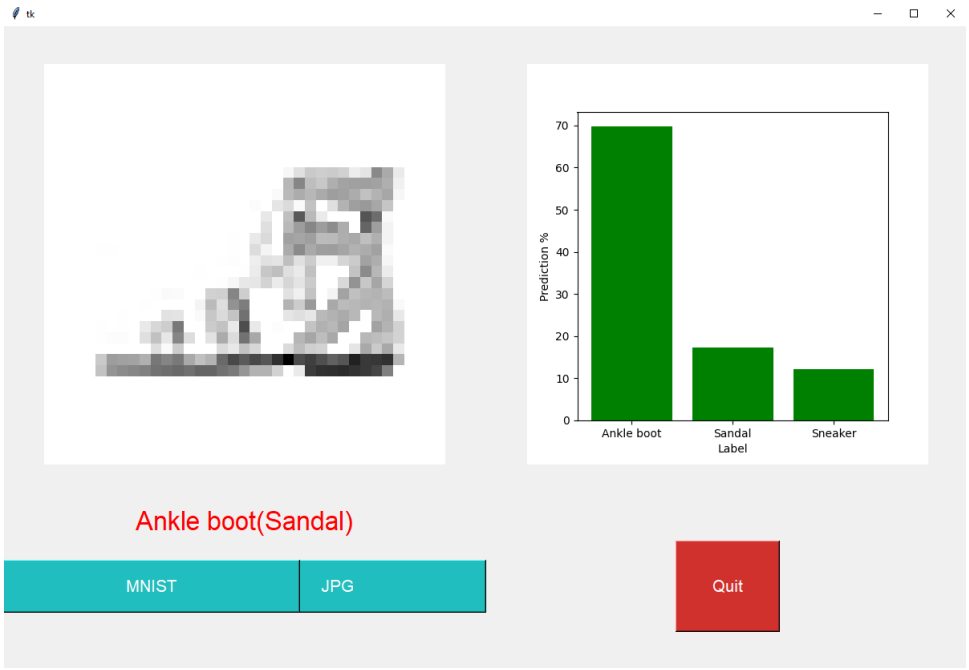
Poniżej przestawione jest GUI naszej aplikacji. W lewym dolny rogu znajdują się przyciski „MNIST”, „JPG” oraz „Quit”.

Przycisk „MNIST” losuje nam obraz z bazy „***Fashion-MNIST”*** , które następnie jest wyświetlane w naszej aplikacji. Przycisk „JPG” pozwala nam wczytać obraz z pliku *.jpg* , który następnie jest konwertowany do odcieni szarości oraz przeskalowany do rozmiarów 28x28 pikseli i również wyświetlany na ekranie.

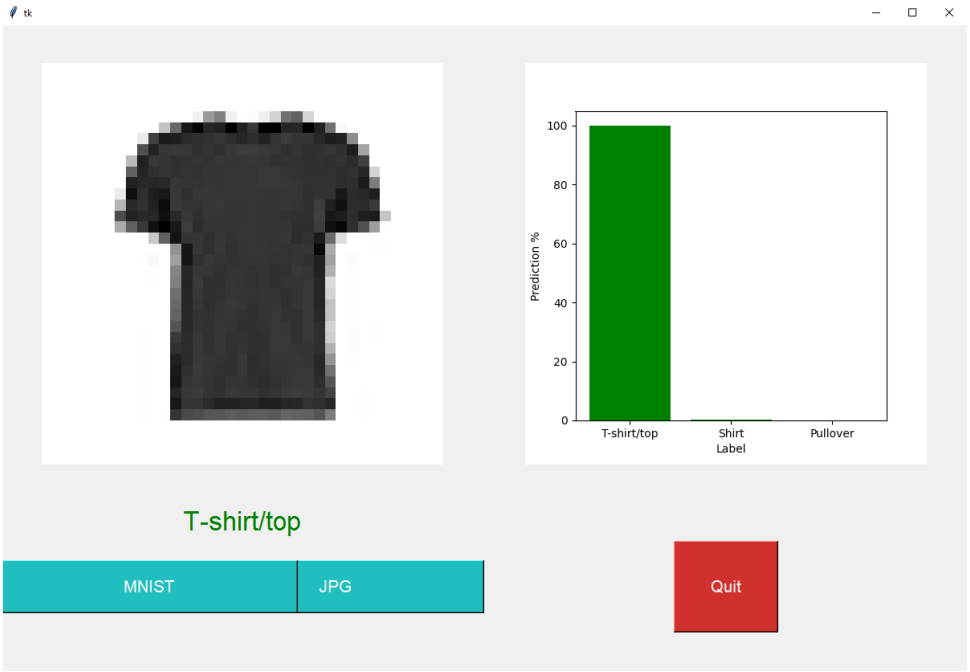
Po wczytaniu obrazu nasza sieć przewiduje do jakiej klasy ubrań zaliczyć wybrany obiekt. Jak widać na Rys.1 i Rys.2. pod obrazem pojawia się nazwa klasy obiektu – Zielona w przypadku poprawnej odpowiedzi (Rys .2), czerwona wraz z podaną w () poprawną klasą w przypadku błędnej spekulacji.(Rys 1.).

W prawej kolumnie naszej aplikacji znajduję się histogram pokazujący 3 najbardziej prawdopodobne klasy, do których należy nasz obraz według przewidywań naszego modelu.

Przyciskiem „Quit” zamykamy naszą aplikację i kończymy działanie pogramu.




**Rys.2. GUI – przykład, gdy nasza sieć neuronowa błędnie sklasyfikuje obraz.**




**Rys.3. GUI – przykład, gdy nasza sieć neuronowa poprawnie sklasyfikuje obraz.**

# 4.Co nie działa?

1) Wczytywany plik *.jpg* powinien zawierać tylko i wyłącznie białe tło. Inaczej program nie zadziała tak jak powinien.

2) Do wczytanego zdjęcia oraz wykresu „mogą" pojawić się dodatkowe osie utrudniające nieco odczyt „Prediction %" oraz 3 klas ubrań z największymy przewidywaniami. Zależy to od wersji Pythona posiadanej na komputerze przez użytkownika.

# 5.Podział pracy

- Praca wspólna:
  - założenia dotyczące projektu
  - analiza funkcjonalności

- Jan Wojdylak:
  - odczyt danych wejściowych
  - projekt sieci neuronowej
  - GUI

- Bartosz Balawender :
  - projekt sieci neuronowej (patrzyłem jak Janek robi)
  - dokumentacja