Spis treści

[WSTĘP 2](#_Toc495960268)

[ROZDZIAŁ PIERWSZY – CEL I ZAKRES PROJEKTU 3](#_Toc495960269)

[1.1 Cel projektu 3](#_Toc495960270)

[1.2 Zakres projektu 3](#_Toc495960271)

[ROZDZIAŁ DRUGI – OPIS DZIAŁANIA SIECI NEURONOWEJ 4](#_Toc495960272)

[ROZDZIAŁ TRZECI – IMPLEMENTACJA SYSTEMU WEBOWEGO 5](#_Toc495960273)

[3.1 Funkcjonalności systemu 5](#_Toc495960274)

[3.2 Baza Danych 5](#_Toc495960275)

[3.2 Środowisko Testowe 5](#_Toc495960276)

[ROZDZIAŁ CZWARTY – IMPLEMENTACJA SIECI NEURONOWEJ 6](#_Toc495960277)

[4.1 Biblioteka Keras 6](#_Toc495960278)

[4.2 Wczytywanie i obróbka obrazów wejściowych 6](#_Toc495960279)

[4.3 Zbieranie danych treningowych 6](#_Toc495960280)

[ROZDZIAŁ PIĄTY – TESTY I ANALIZA 7](#_Toc495960281)

[Bibliografia 8](#_Toc495960282)

# WSTĘP

Rozwój techniki dwudziestego pierwszego wieku, wraz ze stale rosnącymi możliwościami obliczeniowymi komputerów przyniosły przełom w dziedzinie uczenia maszynowego. Zadania, niegdyś możliwe do wykonania tylko na superkomputerach instytucji naukowych i potężnych firm dziś bez problemu wykonywane są za pomocą siły komputerów stacjonarnych, laptopów, a niedługo także i urządzenia mobilne zaczną sobie radzić z coraz to trudniejszymi zadaniami. Jednym z popularniejszych kierunków badań stały się sieci neuronowe.

„Sieć neuronowa jest bardzo uproszczonym modelem mózgu. Składa się ona z dużej liczby (od kilkuset do kilkudziesięciu tysięcy) elementów przetwarzających informacje”. Sieci neuronowe zdobyły swoją popularność dzięki dokładności połączonej z szybkością działania i zostaną szczegółowo opisane w rozdziale X.

Mimo swojej popularności trudno jest znaleźć system webowy, wykorzystujący algorytmy maszynowego uczenia, zwłaszcza jeśli chodzi o rozpoznawanie znaków w czasie rzeczywistym (z opóźnieniem, które nie spowoduje zakłócenia imersji w czasie gry). Istniejące projekty, takie jak na przykład ten na stronie www. deepart.io potrzebują wiele czasu na kalkulacje i przetworzenie obrazów wejściowych. Zaimplementowany w tej pracy system postara się to zmienić dzięki zastosowaniu danych wejściowych o małym rozmiarze, w dodatku mocno uproszczonych. Dokładność stawiana będzie na drugim miejscu względem szybkości działania.

Tak więc praca ta ma za zadanie zintegrowanie szybkiej i względnie dobrze nauczonej sieci neuronowej, z systemem webowym w taki sposób, by użytkownik nie odczuł dyskomfortu płynącego z opóźnienia spowodowanego predykcją. Składać się ona(praca) będzie z Y Rozdziałów, wstępu, oraz podsumowania.

# ROZDZIAŁ PIERWSZY – CEL I ZAKRES PROJEKTU

## 1.1 Cel projektu

Celem niniejszego projektu jest opracowanie systemu webowego, którego głównym elementem będzie środowisko testujące responsywność zaimplementowanej sieci neuronowej, w formie minigry.

## 1.2 Zakres projektu

Praca składać będzie się z trzech głównych części, oraz dwóch modułów pomocniczych.

Pierwszą, a zarazem największą częścią będzie system webowy. Oferować on będzie możliwość tworzenia oraz logowania klientów na własne konto, przeglądania swoich danych oraz statystyk, a także uruchomienia minigry/środowiska testowego, które omówione zostanie w akapicie poniżej.

Drugą częścią będzie środowisko testowe, dzięki któremu możliwe będzie zbadanie responsywności oraz jakości modelu sieci neuronowej. Zaimplementowane będzie w formie minigry, w której zadaniem gracza będzie narysowanie za pomocą ruchu myszy określonego znaku, który następnie będzie przekazywany do algorytmu neuralnego.

Trzecia część, to wspomniana wcześniej siec neuronowa. Zaimplementowana w języku Python, z pomocą biblioteki Keras będzie miała za zadanie najpierw nauczyć się rozpoznawania znaków wejściowych, a następnie rozpoznawania narysowanych przez gracza w czasie rzeczywistym.

Moduły pomocnicze, o których wspomniano w pierwszym akapicie tego rozdziału, to:

- Zaimplementowany w języku c++ z użyciem biblioteki sfml program pozwalający na rysowanie, przechwytywanie oraz zapisywanie obrazów na twardy dysk. Będzie on niezwykle pomocny, przy zbieraniu danych testowych potrzebnych do uczenia sieci neuronowej.

- Baza danych użytkowników systemu. Do implementacji posłuży język MySql.

# ROZDZIAŁ DRUGI – OPIS DZIAŁANIA SIECI NEURONOWEJ

# ROZDZIAŁ TRZECI – IMPLEMENTACJA SYSTEMU WEBOWEGO

## 3.1 Funkcjonalności systemu

## 3.2 Baza Danych

## 3.2 Środowisko Testowe

# ROZDZIAŁ CZWARTY – IMPLEMENTACJA SIECI NEURONOWEJ

## 4.1 Biblioteka Keras

## 4.2 Wczytywanie i obróbka obrazów wejściowych

## 4.3 Zbieranie danych treningowych

Na rysunkach od 4.3.1 do 4.3.9 przedstawiono pierwowzory obrazów, których rozpoznawania uczyć się będzie sieć neuronowa. Dla osiągnięcia zadowalających rezultatów nauczania sieci neuronowej zaimplementowano program pomocniczy, pozwalający na rysowanie na ekranie obrazu, który następnie jest przechwytywany przez bibliotekę SFML i zapisywany do określonego folderu (numerowane od 1 do 9). Funkcjonalność tego programu jest bardzo podobna do środowiska testowego umieszczonego w systemie będącym przedmiotem niniejszej pracy, jednak pozbawiony zbędnych obliczeń związanych z obecnością sieci neuronowej lepiej nadaje się do zapisu wielu plików jednocześnie. Na rysunkach 4.3.10 oraz 4.3.11 przedstawiono wygląd programu.

|  |  |  |
| --- | --- | --- |
| C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image1.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image2.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image3.png |
| C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image4.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image5.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image6.png |
| C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image7.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image8.png | C:\Users\Master\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image9.png |

Rysunki 4.3.1 -4.3.9

# 

Rysunek 0.1

# ROZDZIAŁ PIĄTY – TESTY I ANALIZA

# Bibliografia

**Brak źródeł w bieżącym dokumencie.**