Wizualizacja procesu uczenia SSN

Michał Kacprzak
Bartek Mikołajczyk
Mikołaj Marchewa

18 czerwca 2021

Spis treści

1 Opis projektu

Projekt zakłada stworzenie prostej sieci neuronowej, która zrealizuje realizuje dowolne zadanie z zbioru benchmarkowego UCI MLR `http://archive.ics.uci.edu/ml/datasets.php`. Drugą, najważniejszą częścią projektu jest przygotowanie wizualizacji procesu uczenia się sieci neuronowej, czyli pokazanie jak zmieniają się poszczególne wagi i biasy. Z założenia, projekt działa dla dowolnej sieci typu wielowarstwowego i dowolnej metody uczącej.

2 Kod źródłowy

Kod źródłowy projektu został udostępniony jako repozytorium na Githubie i jest dostępny pod poniższym linkiem:
`https://github.com/HappyButter/ML-learning-visualisation`

3 Podział pracy

Podział pracy między członków zespołów przedstawia się następująco:

- Michał Kacprzak - stworzenie dokumentacji projektu, stworzenie prostego modelu sieci neuronowej, który realizuje problem klasyfikacji dla datasetu Iris oraz Wine
- Bartek Mikołajczyk - stworzenie wyświeltania modelu przedstawiającego proces uczenia sieci neuronowej, stworzenie funkcji processingujących dane z sieci neuronowej
- Mikołaj Marchewa - stworzenie wyświeltania modelu przedstawiającego proces uczenia sieci neuronowej, stworzenie funkcji processingujących dane z sieci neuronowej

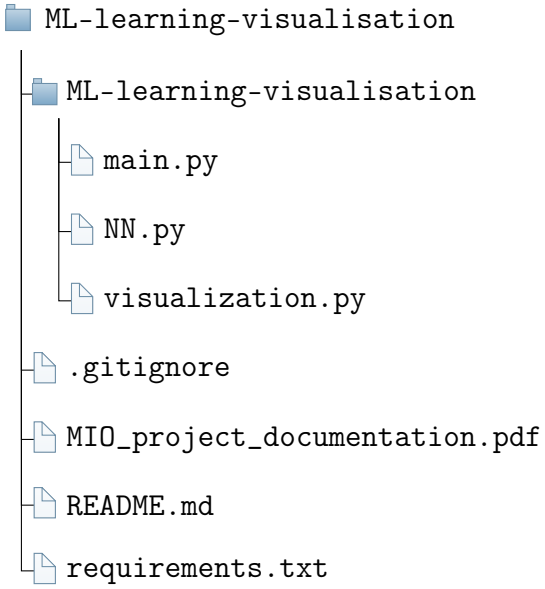
4 Wykorzystane technologie

Projekt został napisany w całości w Pythonie 3 a dokładniej w wersji 3.7, więc oczywiście taka wersja pythona jest wymagana do uruchomienia programu. Dodatkowo zostały wykorzystane następujące biblioteki:

- Pytorch - biblioteka została użyta w celu stworzenia modelu sieci neuronowej, który potem został użyty jako źródło danych do testowania wizualizacji;
- scikit-learn - biblioteka dostarczyła datasetu Iris, na którym był trenowany model sieci neuronowej oraz umożliwiła prostą analizą otrzymanych rezultatów w celu sprawdzenia poprawności treningu sieci;
- plotly - główna biblioteka odpowiadająca za wizualizacje procesu uczenia sieci neuronowej;
- dash - biblioteka pomocnicza, która umożliwiła wyświetlanie wizualizacji w przeglądarce;

5 Analiza projektu

5.1 Struktura projektu



5.2 Sieć neuronowa

Zadaniem wybranym ze zbioru benchmarkowego jest problem klasyfikacji danych dla zbioru Iris oraz zbioru określającego wina. W pliku `NN.py` znajduje się klasa `NN`. Obiekt tej klasy reprezentuje sieć neuronową zdolną, po wytrenowaniu do poprawnej klasyfikacji danych z zadanego zbioru. Obiekt klasy przechowuje także informacje o strukturze modelu oraz wagach i biasach w poszczególnych epochach, co jest możliwe dzięki dwóm funkcjom, znajdującym się również w pliku `NN.py`:

1. `get_layers`
2. `extract_weight_bias_from_model`

Pierwsza z nich jest wywoływana tylko raz, w momencie utworzenia obiektu klasy `NN`. Natomiast druga jest wywoływana dla każdej iteracji podczas treningu. Odpowiednia dokumentacja dla każdej z funkcji została sporządzona w formie docstringów.

5.3 Wizualizacja

Realizacja głównej części projektu znajduje się w pliku `visualization.py`. Główna funkcja odpowiadającą za wizualizację to `visualise_ML`. Na podstawie podanych parametrów dochodzi w niej do otworzenia struktury sieci, podziału danych na odpowiednią ilość klatek oraz utworzenia odpowiednich obiektów(neuronów, krawędzi) wraz z odpowiednimi kolorami i pozycjami. Efektem jest wyświetlenie się wizualizacji w przeglądarce. Odpowiednia dokumentacja dla każdej z funkcji została sporządzona w formie docstringów.

6 Sposób uruchomienia programu

Najprostszym sposobem na uruchomienie projektu jest sklonowanie repozytorium. Następnie należy zainstalować odpowiednie biblioteki oraz uruchomić plik `main.py`, w którym jest zapre-

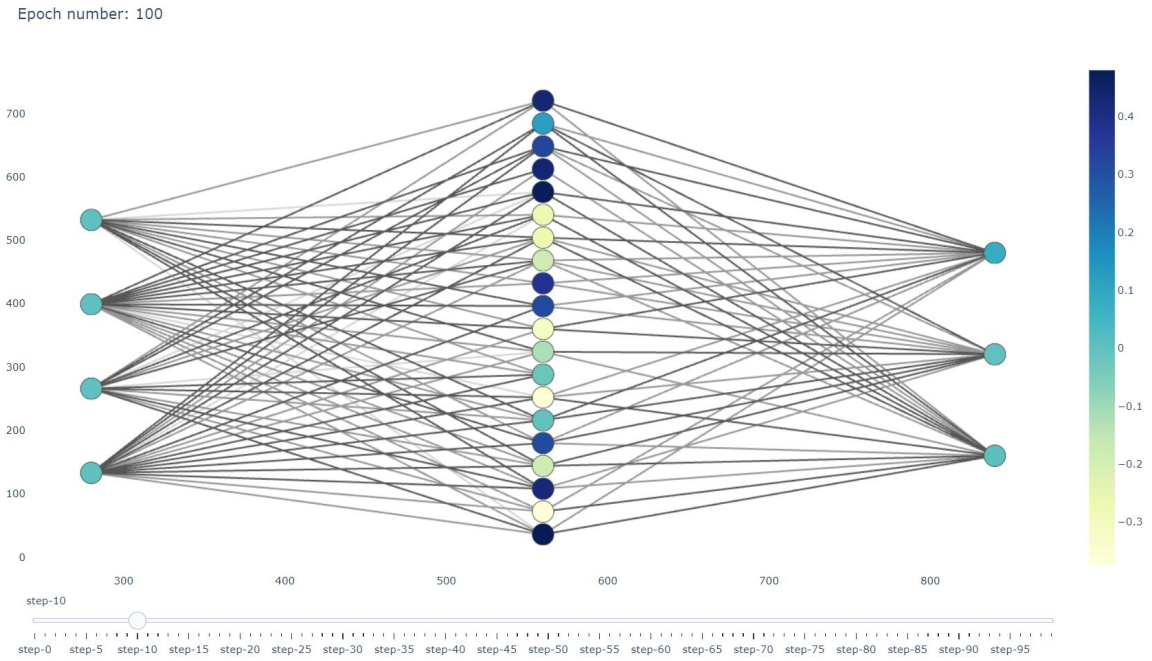
zentowana przykładowa wizualizacji procesu uczenia się sieci neuronowej. Poniższe komendy pozwalają na zrealizowanie wyżej opisanych kroków.

```
$ git clone https://github.com/HappyButter/ML-learning-visualisation.git
$ cd ML-learning-visualisation/
$ pip install -r requirements.txt
$ cd ML-learning-visualisation/
$ python main.py
```

Po wykonaniu powyższych komend wizualizacji powinna być dostępna pod następującym url: `http://127.0.0.1:8050/`

7 Wyniki

7.1 Wizualizacja



Rysunek 1: Wizualizacja wag i biasów

Powyższy rysunek przedstawia wagi i biasy dla 100 epochu, które zostały zarejestrowane podczas przykładowego procesu nauki sieci neuronowej który jest realizowany w pliku `main.py`. W projekcie skupiliśmy się na stworzeniu wizualizacji, która będzie jak najbardziej przyjazna użytkownikowi. W związku z tym użytkownik ma możliwość kontroli nad wizualizacją za pomocą slidera znajdującego się pod wykresem. W prawej części ekranu została także dodana legenda, która pozwala użytkownikowi dokładniej analizować biasy dla poszczególnych neuronów. Wartość wagi pomiędzy poszczególnymi neuronami jest skorelowana z kolorem krawędzi, która łączy dane dwa neurony. Im jest on ciemniejszy tym wartość wagi jest większa. W lewym górnym roku, dla wygody użytkownika została dodana informacja o epoce, której dotyczy dana klatka. Całość tworzy intuicyjną i łatwą w analizie wizualizacje procesu nauki sieci neuronowej.

7.2 Animacje

Na potrzeby prezentacji możliwości projektu zostały sporządzone dwa filmy:

1. Klasyfikacja danych z datasetu Iris - https://www.youtube.com/watch?v=bls9mZIWm_s
2. Klasyfikacja win https://www.youtube.com/watch?v=0VBh_IxQXN0

8 Podsumowanie

Efektem końcowym projektu jest czytelna i schludna wizualizacja procesu uczenia się sieci neuronowej. Niewątpliwą zaletą udostępnionej wizualizacji jest interaktywność oraz wysoka customizowalność. Niestety nie jest jednak pozbawiona wad. Największym ograniczeniem jest ilość klatek oraz neuronów, które użytkownik chce wyświetlić. Powyżej $\sim$100 klatek drastycznie spada płynność działania aplikacji. Wynika to wprost z ograniczeń biblioteki i technologii, ponieważ wszystkie klatki muszą zostać wyrenderowane zanim wizualizacja zostanie wyświetlona.

9 Literatura

- https://plotly.com/
- https://pytorch.org/
- https://dash.plotly.com/