Raport – projekt zaliczeniowy

Autor: Przemysław Musz (**nr indeksu**: 294932)

Przedmiot: Python

Kierunek: Fizyka Techniczna

Stopień: II

Rok: I

Spis treści

1. Wstęp

W ramach niniejszego projektu zaliczeniowego opracowano skrypt w języku Python, służący do redukcji wymiarowości danych tensorowych oraz wizualizacji tak przetworzonych danych w formie graficznej.

Program przygotowano w środowisku Jupyter Notebook.

Dane maszyny:

- **Procesor**: Intel Core i7-8550 @ 1.8GHz x 8
- **RAM**: 8 GB
- **OS**: Ubuntu 18.04.5 LTS
- **Wersja środowiska Python**: 3.6.9
- **Wersja serwera Jupyter**: 6.0.3

2. Opis działania programu

Program korzysta z szeregu pakietów dla środowiska Python. Kluczowe znaczenie mają biblioteki: sklearn, mdshare, numpy, argparse, matplotlib. Dodatkowo zastosowano bibliotekę time w celu przebadania czasu wykonywania programu w zależności od aktualnych nastaw (rodzaju algorytmu oraz współczynnika decymacji danych wejściowych).

Po uruchomieniu programu zapamiętywany jest aktualny znacznik czasu, po czym następuje parsowanie argumentów wejściowych. Dane, wprowadzone przez użytkownika jako argumenty wywołania programu z poziomu terminala, są weryfikowane i – w razie błędu – podmieniane na dane domyślne, wraz z wyświetleniem stosownego komunikatu ostrzegawczego. Wyjątkiem jest nazwa pliku wejściowego – w przypadku podania nieprawidłowej nazwy, program zostaje zakończony i wyświetlana jest informacja o błędzie. Podobnie wygląda sytuacja po podaniu niewłaściwej nazwy algorytmu redukcji danych.

W przypadku poprawnego wyniku parsowania argumentów wywołania, program dokonuje redukcji wymiarowości zestawu danych z użyciem wbudowanej metody `fit_transform()` bibliteki `sklearn`, po czym skaluje dane do użytecznego zakresu z użyciem interpolacji liniowej z użyciem funkcji `interp()` z biblioteki np. Tuż po zakończeniu interpolacji program zapamiętuje kolejny *timestamp* w celu późniejszego określenia czasu, jaki zajęły operacje redukcji wymiarowości oraz interpolacji w stosunku do całkowitego czasu wykonywania programu.

Po wygenerowaniu diagramu rozrzutu (*scatter*) wykonywane są dodatkowe operacje, związane z obróbką okna graficznego wykresu. Na końcu pracy programu wyświetlane są dane dotyczące czasu wykonania.

W programie wykorzystano konstrukcję *`try - except`*, szereg instrukcji warunkowych oraz typ danych *tuple* w celu weryfikacji poprawności danych, wprowadzonych przez użytkownika. Warto dodać, że program może być wykonany bez podawania żadnych parametrów – przyjmuje wtedy wartości domyślne.

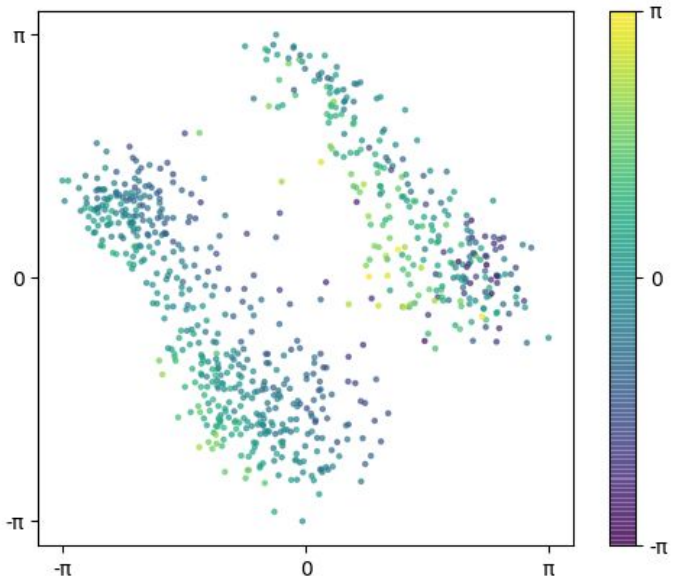
3. Wyniki działania programu dla poprawnych danych wejściowych

Poniżej przedstawiono trzy przykładowe wywołania programu, różniące się parametrami wejściowymi; zaprezentowano informacje wyświetlone w terminalu oraz otrzymane wykresy.

Przykład 1

```
> python3.6 projektPM.py -d 1000 -i alanine-dipeptide-3x250ns-
heavy-atom-distances.npz -a PCA -c viridis
```

```
Total time elapsed:  0.478 seconds, 5.3% for data dimensionality
reduction.
```

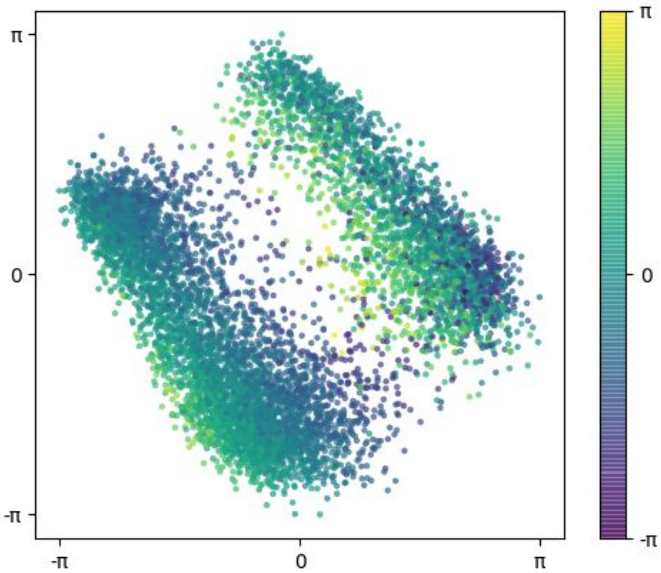


Rysunek 1: Wynik działania programu dla wywołania z przykładu 1

Przykład 2

```
> python3.6 projektPM.py -d 100 -i alanine-dipeptide-3x250ns-
heavy-atom-distances.npz -a PCA -c viridis


Total time elapsed:  0.493 seconds, 8.7% for data dimensionality
reduction.
```

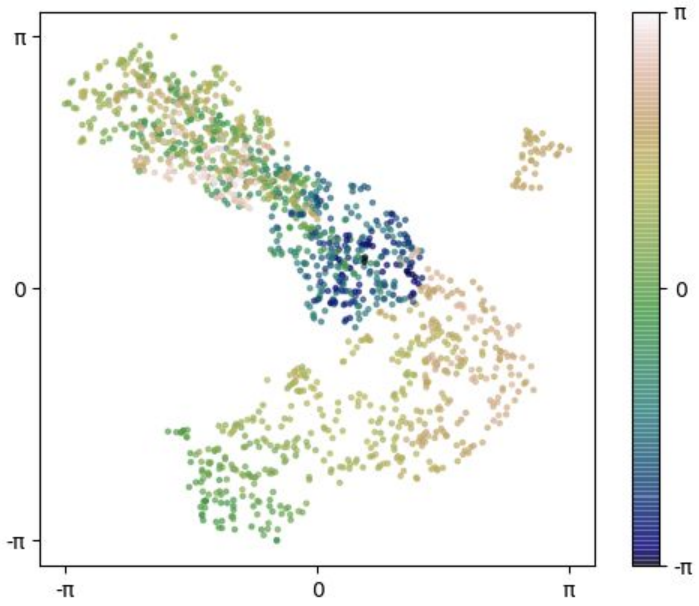


Rysunek 2: Wynik działania programu dla wywołania z przykładu 2

Przykład 3

```
> python3.6 projektPM.py -d 500 -i alanine-dipeptide-3x250ns-
heavy-atom-distances.npz -a TSNE -c gist_earth


Total time elapsed: 13.449 seconds, 96.9% for data dimensionality
reduction.
```



Rysunek 3: Wynik działania programu dla wywołania z przykładu 3

4. Wybrane scenariusze nieprawidłowych wywołań

Poniżej zaprezentowano działanie programu w przypadku, gdy jeden z parametrów wywołania będzie nieprawidłowy (liczba poza zakresem, nieznana nazwa algorytmu, pliku lub mapy kolorystycznej).

Scenariusz 1: zbyt niska wartość współczynnika decymacji

```
> python3.6 projektPM.py -d 1 -i alanine-dipeptide-3x250ns-heavy-atom-distances.npz -a PCA -c viridis


WARNING: decimation ratio out of valid range; default value 500 will be used!
Total time elapsed:  0.567 seconds, 4.9% for data dimensionality reduction.
```

Scenariusz 2: Błędna nazwa pliku

```
> python3.6 projektPM.py -d 100 -i XYZalanine-dipeptide-3x250ns-heavy-atom-distances.npz -a PCA -c viridis


Exitning due to errors:
XYZalanine-dipeptide-3x250ns-heavy-atom-distances.npz [no match in repository]
```

Scenariusz 3: Błędna nazwa algorytmu

```
> python3.6 projektPM.py -d 100 -i alanine-dipeptide-3x250ns-heavy-atom-distances.npz -a PCAA -c viridis
ERROR: An error occured during selection of reduciton method. Exiting…
```

Scenariusz 4: Błędna nazwa mapy kolorystycznej

```
> python3.6 projektPM.py -d 100 -i alanine-dipeptide-3x250ns-heavy-atom-distances.npz -a PCA -c rainbow

WARNING: colormap name not recognized; default colormap 'ocean' will be used!

Total time elapsed:  0.459 seconds, 15.0% for data dimensionality reduction.

Exitning…
```

5. Podsumowanie i wnioski

W ramach realizacji niniejszego projektu zrealizowano podstawowe założenia otrzymanego zadania. Program poprawnie dokonuje przetworzenia danych tensorowych i wizualizacji na płaszczyźnie XY (z trzecim wymiarem zakodowanym w formie określonej mapy kolorów), właściwie działają też zabezpieczenia, przejmujące kontrolę nad programem w przypadku podania przez użytkownika błędnych danych. Dzięki temu program może w kontrolowany sposób zakończyć swoje działanie, wyświetlając stosowne komunikaty.

Zastosowana metoda oceny czasu wykonania pozwala zgrubnie porównać złożoność obliczeniową zastosowanych algorytmów (należy przy tym pamiętać, że użyta technik, polegająca na zapisywaniu *timestampów* systemowych, jest także podatna na zmiany obciążenia procesora przez inne aplikacje, działające w tym samym czasie; można zminimalizować ten wpływ przez ręczne wyłączenie niepotrzebnych aplikacji na czas wykonywania testów). Widoczne są bardzo duże różnice czasu realizacji procedury redukcji wymiarowości dla algorytmów PCA i TSNE ze znaczną przewagą wydajnościową metody PCA.

Otrzymane wyniki nie pokrywają się pod względem graficznym z przykładem, przedstawionym w opisie zadania. Przyczyn tego faktu prawdopodobnie jest wiele, jedną z nich jest zastosowana technika rysowania wykresu (*scatterplot* zamiast wykresu konturowego z widocznymi izoliniami). Przyczyn różnic w kształcie obszarów, zarysowanych na wykresie, należy doszukiwać się w zastosowanej metodzie redukcji wymiarowości oraz przetwarzania danych.