Wydział Informatyki i Sztucznej Inteligencji

LAB5-a: Opracowanie sieci neuronowej do odtwarzania warunków brzegowych w problemie odwrotnym przepływu ciepła w materiale 1D i 2D

dr inż. Konrad M. Gruszka,*

5 września 2024

Streszczenie

Ten dokument stanowi instrukcję do wykonania ćwiczenia z zagadnienia transferu ciepła w jednowymiarowym, stacjonarnym przypadku do przedmiotu Rozwiązywanie Zadań Odwrotnych. Celem ćwiczenia jest odtworzenie warunków brzegowych w 1D i 2D materiale jednorodnym stacjonarnym za pomocą sieci neuronowej.

1 Informacje wstępne

Wprowadzenie
Niniejsza instrukcja bazuje na wcześniej napisanym oprogramowaniu implementującym metodę MRS do rozwiązywania jednowymiarowego problemu transferu ciepła dla przypadku jednorodnego. W ramach ćwiczenia, wprowadzamy nową funkcjonalność polegającą na wykorzystaniu sztucznej sieci neuronowej (SSN) do rozwiązania problemu odwrotnego, w którym sieć będzie próbowała odtworzyć warunki brzegowe na podstawie rozkładu temperatury w materiale.

Cel ćwiczenia
Celem tego ćwiczenia jest zrozumienie zasad stosowania sieci neuronowych do problemów odwrotnych w fizyce. Zadaniem jest stworzenie sieci neuronowej, która na podstawie rozkładu temperatury w 1D i 2D materiale jednorodnym (stacjonarnie) będzie odtwarzać warunki brzegowe.

Środowisko pracy

- Python w wersji > 3
- Visual Studio Code z dodatkiem Jupyter Notebook

Biblioteki

- numpy
- matplotlib
- time
- keras
- sklearn
- os - do wczytywania/zapisywania plików

*Katedra Informatyki, Wydział Inżynierii Mechanicznej i Informatyki (kgruszka@icis.pcz.pl)

2 Modyfikacja kodu MRS 1D i 2D w przypadku jednorodnym stacjonarnym

Zakres zadania: Wykorzystaj wygenerowane wcześniej pliki rozkładu temperatur z MRS dla przypadków 1D oraz dla 2D oraz funkcje wczytujące te dane do celów uczenia sieci.

Wydajność obliczeniowa
W celu ustalenia metryk wydajności obliczeniowej oraz ustalenia, w jaki sposób dobór parametrów empirycznych symulacji wpływa na całkowity czas wykonania należy zaimplementować:

- Pomiar czasu generowania jednego pliku
- Pomiar całkowitego czasu wygenerowania wszystkich tworzonych plików,

3 Opracowanie sieci neuronowej do odtwarzania warunków brzegowych

W tym segmencie zadania należy zbudować model sieci neuronowej, który będzie w stanie odtworzyć warunki brzegowe (lewa i prawa temperatura) na podstawie rozkładu temperatury w materiale 1D lub 2D.

Wczytywanie danych: Dane z wcześniej wyeksportowanych plików należy wczytać do macierzy *ndarray*, zapewniając odpowiedni wymiar tej macierzy.

1. **Ogólna budowa modelu sieci neuronowej:**
   - Zdefiniuj architekturę sieci neuronowej, która będzie uczyć się na podstawie wczytanych danych.
   - Zastosuj bibliotekę Keras do definicji i trenowania modelu.
   - Zaimplementuj podział danych na zestawy uczące i testowe.
   - Pokaż dane pozwalające ocenić skuteczność modelu na danych testowych.
2. **Wizualizacja wyników:**
   - Użyj biblioteki matplotlib do porównania przewidywanych warunków brzegowych z rzeczywistymi danymi.

Sieć neuronowa:

1. **Inicjalizacja modelu:** Sieć ma realizować model sekwencyjny.
2. **Dodawanie warstw:**
   (a) Warstwa wejściowa dla 1D: Przyjmie 4 cechy (temperatura na obu końcach pręta, tolerancja, liczba iteracji zbieżności). Użyj lyr.Input(shape=(4,)). Dla przypadku 2D postępuj analogicznie.
   (b) Warstwy ukryte: Dodaj kilka warstw gęstych (Dense) z funkcjami aktywacji ReLU. Początkowo zaleca się użyć co najmniej dwóch warstw gęstych, np. 256 i 512 neuronów.
   (c) Warstwa wyjściowa: Powinna zwracać dwa wyniki (temperatura na lewej i prawej stronie pręta). Użyj warstwy Dense z funkcją aktywacji liniową.
3. **Kompilacja modelu:** Skonfiguruj proces uczenia poprzez wybór optymalizatora (np. Adam) i funkcji straty (np. średni błąd bezwzględny - MAE).

Przykładowy kod definicji modelu:

```
model = keras.Sequential([
    lyr.Input(shape=(4,)),
    lyr.Dense(256, activation='relu'),
    lyr.Dense(512, activation='relu'),
    lyr.Dense(2)  # Zwracamy 2 wartości: temperatura na lewym i prawym końcu
])
model.compile(optimizer='adam', loss='mae')
```

Trenowanie i testowanie modelu:

1. Podziel dane na zestawy treningowe i testowe.
2. Przeprowadź proces trenowania z użyciem metody `model.fit()`.
3. Oceń skuteczność modelu za pomocą `model.evaluate()` na danych testowych.
4. Przedstaw wykresy porównujące przewidywane i rzeczywiste warunki brzegowe.

4 Forma i ocena wykonania ćwiczenia

Forma oddania zadania
Zadanie należy oddać w formie plików z rozszerzeniem .py zawierających implementację algorytmu SSN. Nie należy przesyłać plików zawierających dane uczące do SSN, ale zapewnić możliwość ich łatwego generowania i wczytywania z aktualnego katalogu (podkatalogi "data"i "data2D").

Następnie, należy przeprowadzić testy poprawności działania dla alg. 1D i 2D.
Wyniki należy umieścić w raporcie (.ipynb lub .pdf), w którym znaleźć się mają następujące informacje:

- Udokumentowanie przeprowadzonych testów, wyniki oraz odpowiednie komentarze.
- Parametry sieci dla jakich uzyskano prezentowane wyniki.
- Wykresy przedstawiające porównanie przewidywanych warunków brzegowych z rzeczywistymi.
- Uzyskane czasy całkowite generowania danych z MRS.

Ocena ćwiczenia będzie bazować na:

1. Poprawności implementacji algorytmu.
2. Kompletności przeprowadzonych testów.
3. Jakości dokumentacji i wyjaśnień teoretycznych.
4. Implementacji usprawnień działania algorytmu.