SPRAWOZDANIE

Zajęcia: Matematyka Konkretna

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 6 Data 10.05.2025 Temat: „Liniowe RNN” Wariant 11	Anna Więzik Informatyka II stopień, niestacjonarne, 2 semestr, gr.1a TTO

1. Polecenie:

Link do repozytorium: https://github.com/AnaShiro/MK_2025

Rekurencyjne sieci neuronowe (RNN) stanowią klasę modeli głębokiego uczenia, które są szczególnie skuteczne w przetwarzaniu danych sekwencyjnych, takich jak tekst, sygnały czasowe czy dane szeregów czasowych. W odróżnieniu od klasycznych sieci neuronowych, RNN posiadają pamięć stanu, która umożliwia przechowywanie informacji z poprzednich kroków czasowych i ich wykorzystanie przy przetwarzaniu kolejnych elementów sekwencji. W prezentowanym zadaniu skupiono się na uproszczonej wersji RNN – tzw. liniowej sieci rekurencyjnej – gdzie funkcja aktywacji została pominięta, a obliczenia ograniczają się do operacji liniowych.

Celem implementacji było zbudowanie minimalnej wersji RNN przy użyciu języka Python i biblioteki NumPy. Model został przeszkolony przy użyciu algorytmu propagacji wstecznej w czasie (Backpropagation Through Time, BPTT), a następnie przeprowadzono analizę gradientów w czasie, aby zilustrować zjawiska ich zanikania i eksplozji. Dane treningowe stanowiły losowe binarne sekwencje długości 10, a celem sieci było nauczenie się sumowania wartości w każdej z nich. W ramach zadania przeprowadzono także wizualizację powierzchni błędu oraz gradientów w zależności od wartości wag wejściowych i rekurencyjnych, co pozwoliło lepiej zrozumieć dynamikę procesu uczenia się.

2. Opis programu opracowanego

```python
import numpy as np

n_sequences = 30
sequence_length = 20
possible_values = np.array([0.0, 0.2, 0.4, 0.6, 0.8, 1.0])

np.random.seed(42)
X = np.random.choice(possible_values, size=(n_sequences, sequence_length))

t = X.mean(axis=1)

for i in range(5):
    print(f"X[{i}]: {X[i]}")
    print(f"t[{i}]: {t[i]}\n")
```

```
X[0]: [0.6 0.8 0.4 0.8 0.8 0.2 0.4 0.4 0.4 0.8 0.6 0.4 1.  0.8 0.2 0.6 1.  1.
 0.2 0.6]
t[0]: 0.6

X[1]: [0.8 0.  0.6 0.2 1.  0.8 0.6 0.  0.  0.4 0.4 0.2 0.6 0.6 1.  1.  1.  0.4
 0.6 0.6]
t[1]: 0.5399999999999999

X[2]: [0.  0.4 0.8 0.4 0.8 0.  0.2 0.6 0.  0.6 1.  0.2 0.2 0.  0.2 0.8 0.2 0.6
 0.6 0.6]
t[2]: 0.41

X[3]: [0.6 0.8 0.4 1.  0.  0.6 0.2 0.6 0.2 1.  1.  1.  0.2 0.6 1.  0.8 0.2 0.2
 0.6 0.2]
t[3]: 0.5599999999999998

X[4]: [0.2 1.  0.6 1.  1.  0.6 0.  1.  0.8 0.8 0.2 0.8 0.2 0.  0.6 0.6 0.6 0.8
 0.  0.8]
t[4]: 0.5800000000000001
```

3. Wnioski

Z przeprowadzonego eksperymentu wynika, że nawet najprostsza forma sieci rekurencyjnej może posłużyć do badania fundamentalnych zjawisk wpływających na proces uczenia. Obserwacja gradientów wykazała ich niestabilność – w zależności od wartości wag mogły one bardzo szybko zanikać lub eksplodować, co bezpośrednio przekładało się na trudności w efektywnej optymalizacji modelu. Wizualizacja powierzchni błędu pokazała, że przestrzeń parametrów nie jest jednorodna, a lokalne minima mogą być oddzielone stromymi przejściami, co dodatkowo utrudnia proces trenowania. Zastosowanie prostego algorytmu BPTT umożliwiło śledzenie zmian gradientów w czasie i dostarczyło intuicyjnego wglądu w ich zachowanie. Wnioski te podkreślają znaczenie dobrze dobranych hiperparametrów i potencjalną potrzebę zastosowania technik takich jak normalizacja wag, krótsze sekwencje treningowe czy zaawansowane architektury RNN (np. LSTM, GRU), które są bardziej odporne na wspomniane problemy.