SPRAWOZDANIE

Zajęcia: Analiza Procesów Uczenia
Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium 9
5.12.2023
Temat: Nieliniowe sieci RNN w oparciu o tenzory
Wariant 1

Jarosław Waliczek
Informatyka II stopień,
stacjonarne,
2 semestr,
Gr.2

1. **Polecenie:** Celem jest nabycie podstawowej znajomości użycia propagacji wstecznej w czasie dla nieliniowych sieci RNN - podstawowe pojęcia oraz zagadnienia.

2. Wprowadzane dane:

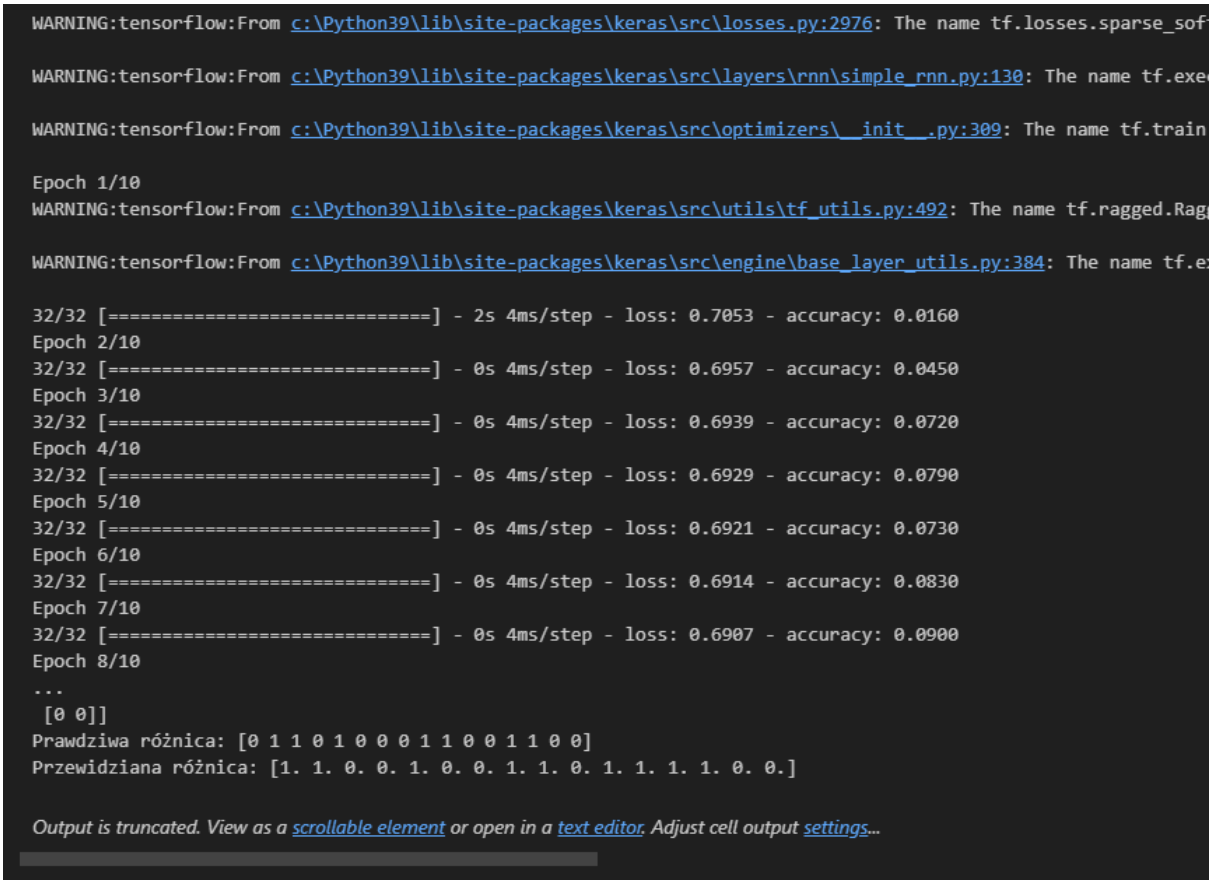
Dane wejściowe (X) reprezentują dwie liczby binarne o długości 16 bitów każda. Dane wyjściowe (Y) reprezentują różnicę między odpowiadającymi sobie bitami dwóch liczb binarnych.

3. Wykorzystane komendy:

```
import tensorflow as tf
import numpy as np

# Tworzymy dane treningowe
def generate_data(num_samples=1000):
    X = np.random.randint(0, 2, size=(num_samples, 16, 2))  # Generujemy dwie liczby binarne o długości 16 bitów
    Y = np.abs(X[:, :, 0] - X[:, :, 1])  # Obliczamy różnicę dwóch liczb binarnych
    return X, Y

# Tworzymy model RNN
model = tf.keras.Sequential([
    tf.keras.layers.SimpleRNN(8, input_shape=(16, 2), activation='relu', return_sequences=True),
    tf.keras.layers.SimpleRNN(8, activation='relu'),
    tf.keras.layers.Dense(16, activation='sigmoid')
])

# Kompilujemy model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Generujemy dane treningowe
X_train, Y_train = generate_data()

# Trenujemy model
model.fit(X_train, Y_train, epochs=10, batch_size=32)

# Testujemy model na nowych danych
X_test, Y_test = generate_data(10)
predictions = model.predict(X_test)

# Wyświetlamy wyniki
for i in range(10):
    input_data = X_test[i]
    true_output = Y_test[i]
    predicted_output = predictions[i].round()

    print(f"Wejście: {input_data}")
    print(f"Prawdziwa różnica: {true_output}")
    print(f"Przewidziana różnica: {predicted_output}")
    print()
```

4. Wynik działania:

rzuty ekranu:



5. **Wnioski:** Użyliśmy prostego modelu RNN, który może przechowywać informacje o poprzednich krokach sekwencji oraz funkcji aktywacji ReLU i funkcji straty binary crossentropy, co jest typowe dla problemów binarnych.

Repo: https://github.com/Jaro233/MK