

SPRAWOZDANIE

Zajęcia: Matematyka Konkretna

Prowadzący: prof. dr hab. Vasyl Martsenyuk

| | |
|--|---|
| Laboratorium Nr 8 Data 30.06.2025 Temat: Algorytm LSTM dla tekstu Wariant 6 | Imię Nazwisko Hubert Mentel Informatyka II stopień, niestacjonarne, 2 semestr, gr.1a |
|--|---|

1. Zadanie:

Opracować sieć LSTM w celu nauczenia się tekstu z dokładnością 0.1

Wariant 6: "AI research has tried and discarded many different approaches, including simulating the brain, modeling human problem solving, formal logic, large databases of knowledge, and imitating animal behavior"

Pliki dostępne są pod linkiem:

<https://github.com/HubiPX/NOD/tree/master/MK/Zadanie%208>

2. Opis programu opracowanego (kody Źródłowe, zrzuty ekranu)

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Embedding
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping

# Tekst bazowy
text = ("AI research has tried and discarded many different approaches, including "
        "simulating the brain, modeling human problem solving, formal logic, "
        "large databases of knowledge, and imitating animal behavior")

# Tworzenie słownika znaków
chars = sorted(list(set(text)))
char_to_idx = {c: i for i, c in enumerate(chars)}
idx_to_char = {i: c for i, c in enumerate(chars)}

# Parametry
seq_length = 40
step = 1

# Tworzenie danych treningowych
X = []
y = []
for i in range(0, len(text) - seq_length, step):
    X.append([char_to_idx[c] for c in text[i: i + seq_length]])
    y.append(char_to_idx[text[i + seq_length]])

X = np.array(X)
y = to_categorical(y, num_classes=len(chars))





















# Model
model = Sequential()
model.add(Embedding(input_dim=len(chars), output_dim=32))
model.add(LSTM(128, return_sequences=False))
model.add(Dense(len(chars), activation='softmax'))

# Kompilacja
model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=0.01))

# Trenowanie z EarlyStopping, żeby zatrzymać się przy stracie < 0.1
early_stop = EarlyStopping(monitor='loss', patience=10, restore_best_weights=True)

history = model.fit(X, y, batch_size=32, epochs=20, callbacks=[early_stop])

# Sprawdzenie straty
final_loss = history.history['loss'][-1]
print(f"Final loss: {final_loss:.4f}")
```

```
Epoch 1/20
5/5  2s 13ms/step - loss: 3.2486
Epoch 2/20
5/5  0s 12ms/step - loss: 3.1319
Epoch 3/20
5/5  0s 12ms/step - loss: 2.9091
Epoch 4/20
5/5  0s 12ms/step - loss: 2.8991
Epoch 5/20
5/5  0s 12ms/step - loss: 2.8112
Epoch 6/20
5/5  0s 12ms/step - loss: 2.5892
Epoch 7/20
5/5  0s 12ms/step - loss: 2.4861
Epoch 8/20
5/5  0s 12ms/step - loss: 2.2341
Epoch 9/20
5/5  0s 12ms/step - loss: 1.9394
Epoch 10/20
5/5  0s 12ms/step - loss: 1.8786
Epoch 11/20
5/5  0s 12ms/step - loss: 1.5739
Epoch 12/20
5/5  0s 12ms/step - loss: 1.2074
Epoch 13/20
5/5  0s 12ms/step - loss: 0.9176
Epoch 14/20
5/5  0s 12ms/step - loss: 0.6876
Epoch 15/20
5/5  0s 12ms/step - loss: 0.4583
Epoch 16/20
5/5  0s 12ms/step - loss: 0.2975
Epoch 17/20
5/5  0s 12ms/step - loss: 0.1694
Epoch 18/20
5/5  0s 12ms/step - loss: 0.1197
Epoch 19/20
5/5  0s 12ms/step - loss: 0.0761
Epoch 20/20
5/5  0s 12ms/step - loss: 0.0514
Final loss: 0.0508
```

3. Wnioski

W przeprowadzonym eksperymencie zbudowano sieć neuronową typu LSTM, której celem było nauczenie się ciągów znaków z krótkiego tekstu dotyczącego historii badań nad sztuczną inteligencją. Model został poprawnie przygotowany i wytrenowany z użyciem warstwy Embedding, jednej warstwy LSTM oraz warstwy wyjściowej typu Dense z aktywacją softmax. W trakcie treningu zastosowano mechanizm wczesnego zatrzymywania (EarlyStopping), co pozwoliło na optymalizację procesu uczenia. Już od 19. epoki model osiągnął stratę (loss) poniżej 0.1, spełniając założone kryterium dokładności. Ostatecznie uzyskano skuteczny model, zdolny do przewidywania kolejnych znaków z wysoką trafnością.