In [ ]:

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.datasets import imdb
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, Bidirectional, LSTM, Dense
from tensorflow.keras.preprocessing.sequence import pad_sequences
import logging

# 1. Konfiguracja TensorFlow
tf.get_logger().setLevel(logging.ERROR)
tf.autograph.set_verbosity(3)

# 2. Ustawienia i ładowanie danych
MAX_FEATURES = 10000
MAX_LEN = 100
EMBEDDING_DIM = 128
BATCH_SIZE = 32
EPOCHS = 5

print(f"Ładowanie zbioru danych IMDB (tylko {MAX_FEATURES} najczęściej występują
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=MAX_FEATURES)

print(f"Dane treningowe: {len(x_train)} recenzji | Dane testowe: {len(x_test)} r

# 3. Preprocessing: Standaryzacja długości sekwencji
print(f"Standaryzacja długości sekwencji do MAX_LEN = {MAX_LEN}...")
x_train = pad_sequences(x_train, maxlen=MAX_LEN)
x_test = pad_sequences(x_test, maxlen=MAX_LEN)

print(f"Kształt x_train po paddingu: {x_train.shape}")

# 4. Budowa Modelu BRNN (Bidirectional LSTM)
model = Sequential()

# 1. Warstwa osadzania (Embedding)
model.add(Embedding(MAX_FEATURES, EMBEDDING_DIM))

# 2. Warstwa Bi-directional RNN (Bidirectional LSTM)
model.add(Bidirectional(LSTM(64)))

# 3. Warstwa klasyfikacyjna (Dense)
model.add(Dense(1, activation='sigmoid'))

# 5. Kompilacja i Podsumowanie Modelu (Z Wymuszeniem Budowy)
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy']

# Wymuszenie budowy modelu, aby model.summary() wyświetlił poprawną liczbę param
model.build(input_shape=(None, MAX_LEN))

print("\nPodsumowanie modelu BRNN (Bidirectional LSTM):")
model.summary()

# 6. Trening Modelu
print("\nRozpoczęcie treningu...")
history = model.fit(x_train, y_train, epochs=EPOCHS, batch_size=BATCH_SIZE, vali

# 7. Ocena Modelu
print("\nOcena modelu na zbiorze testowym...")
```

```
loss, accuracy = model.evaluate(x_test, y_test, verbose=0)
print(f"\nWynik końcowy na zbiorze testowym:")
print(f"Loss (Strata): {loss:.4f}")
print(f"Accuracy (Dokładność): {accuracy*100:.2f}%")
```

```
Ładowanie zbioru danych IMDB (tylko 10000 najczęściej występujących słów)...
Dane treningowe: 25000 recenzji | Dane testowe: 25000 recenzji
Standaryzacja długości sekwencji do MAX_LEN = 100...
Kształt x_train po paddingu: (25000, 100)

Podsumowanie modelu BRNN (Bidirectional LSTM):
```

**Model: "sequential_1"**

| Layer (type) | Output Shape | Param # |
|---|---|---|
| embedding_1 (Embedding) | (None, 100, 128) | 1,280,000 |
| bidirectional_1 (Bidirectional) | (None, 128) | 98,816 |
| dense_1 (Dense) | (None, 1) | 129 |

**Total params:** 1,378,945 (5.26 MB)

**Trainable params:** 1,378,945 (5.26 MB)

**Non-trainable params:** 0 (0.00 B)

```
Rozpoczęcie treningu...
Epoch 1/5
625/625 ━━━━━━━━━━━━━━━━━━━━ 14s 21ms/step - accuracy: 0.8004 - loss: 0.4267 - va
l_accuracy: 0.8478 - val_loss: 0.3474
Epoch 2/5
625/625 ━━━━━━━━━━━━━━━━━━━━ 14s 22ms/step - accuracy: 0.8987 - loss: 0.2516 - va
l_accuracy: 0.8296 - val_loss: 0.3830
Epoch 3/5
625/625 ━━━━━━━━━━━━━━━━━━━━ 15s 24ms/step - accuracy: 0.9406 - loss: 0.1593 - va
l_accuracy: 0.8264 - val_loss: 0.4108
Epoch 4/5
625/625 ━━━━━━━━━━━━━━━━━━━━ 18s 28ms/step - accuracy: 0.9696 - loss: 0.0865 - va
l_accuracy: 0.8242 - val_loss: 0.5193
Epoch 5/5
625/625 ━━━━━━━━━━━━━━━━━━━━ 17s 27ms/step - accuracy: 0.9819 - loss: 0.0535 - va
l_accuracy: 0.8200 - val_loss: 0.6660

Ocena modelu na zbiorze testowym...

Wynik końcowy na zbiorze testowym:
Loss (Strata): 0.6578
Accuracy (Dokładność): 82.35%
```