# Dokumentacja projektu

## Tytuł projektu

Detekcja sarkazmu w nagłówkach tweet’ów porównanie modeli BiGRU i BiLSTM + Attention

## Cel projektu

Celem projektu było stworzenie skutecznego modelu klasyfikującego nagłówki tweet’ów pod kątem obecności sarkazmu. Projekt obejmował budowę modeli sekwencyjnych z wykorzystaniem mechanizmów BiGRU oraz BiLSTM + Attention, a następnie porównanie ich wydajności.

## Zbiór danych

Sarcasm Headlines Dataset  
- Liczba przykładów: ~25 000 nagłówków  
- Kolumny:  
 - headline - nagłówek wiadomości (tekst)  
 - is\_sarcastic - etykieta binarna: 0 - nie-sarkazm, 1 - sarkazm

- article\_link – link do oryginalnego artykułu

## Proces realizacji

### Eksploracja danych

Przeanalizowano rozkład klas - dane były lekko niezrównoważone.  
Histogram długości nagłówków wykazał, że większość zawiera do 15 słów - ustalono SEQUENCE\_LENGTH = 15.

### Przetwarzanie tekstu

- Usunięto linki, interpunkcję, liczby.  
- Zamieniono tekst na małe litery.  
- Zastosowano warstwę TextVectorization:  
 - max\_tokens = 8000 - ograniczenie słownika.  
 - output\_sequence\_length = 15.

### Podział danych

Podział na zbiór treningowy i testowy w proporcji 80:20. Zachowano proporcje klas (stratify = y).

### Wektoryzacja

Każdy nagłówek zamieniono na sekwencję liczb (tokenów) o stałej długości 15.

## Budowa modeli

### Model BiGRU

Embedding -> BiGRU (64 units) -> Dropout (0.3) -> Dense(32, ReLU) -> Dropout (0.2) -> Dense(1, sigmoid).

### Model BiLSTM + Attention

Embedding -> BiLSTM (64 units, return\_sequences=True) -> Attention -> BatchNormalization -> Dense(32, ReLU, L2 regularization) -> Dropout (0.2) -> Dense(1, sigmoid).  
Parametry:  
- Optimizer: RMSprop (learning\_rate=1e-3).  
- Regularization: L2(0.01).  
- EarlyStopping (monitoring val\_loss, patience=2).

## Trening

- Zastosowano class\_weight='balanced' - model był uczony tak, aby lepiej rozpoznawać rzadziej występujący sarkazm.  
- Trening przeprowadzono z podziałem na zbiór walidacyjny.  
- Dla modelu BiLSTM + Attention zastosowano dodatkowe mechanizmy kontroli nad przeuczeniem (BatchNormalization, EarlyStopping, RMSprop).

## Wyniki

### Accuracy i loss

- BiGRU: accuracy = 86%, f1-score (sarkazm) = 0.85.  
- BiLSTM + Attention: accuracy = 86%, f1-score (sarkazm) = 0.85.

### Raport klasyfikacji BiLSTM + Attention

- Precision (sarkazm): 0.81  
- Recall (sarkazm): 0.88  
- F1-score (sarkazm): 0.85  
- Precision (nie-sarkazm): 0.87  
- Recall (nie-sarkazm): 0.87  
- F1-score (nie-sarkazm): 0.87

### Confusion matrix BiLSTM + Attention

- 2526 poprawnych predykcji nie-sarkazmu.  
- 2067 poprawnych predykcji sarkazmu.  
- 471 false positives (nie-sarkazm -> sarkazm).  
- 278 false negatives (sarkazm -> nie-sarkazm).

## Wnioski końcowe

- Model BiLSTM + Attention osiągnął bardzo dobry wynik (F1-score = 0.85), porównywalny z BiGRU.  
- Wprowadzenie Attention pozwoliło modelowi lepiej wychwytywać istotne fragmenty tekstu.  
- Mechanizmy kontroli nad przeuczeniem (BatchNormalization, EarlyStopping, L2 regularization, Dropout) zapewniły wysoką stabilność uczenia.  
- Model dobrze rozróżnia obie klasy, bez widocznego uprzywilejowania jednej z nich.  
- W porównaniu do BiGRU, model BiLSTM + Attention wykazał bardziej zbalansowane błędy (false positives vs false negatives).

## Możliwe ulepszenia

- Zastosowanie modeli opartych na Transformerach (np. BERT).  
- Wydłużenie sekwencji (np. do 20-25 słów).  
- Użycie wstępnie wytrenowanych embeddingów (np. GloVe).  
- Augmentacja danych tekstowych.

## Technologie

- Python 3.11  
- TensorFlow 2.x + Keras  
- Scikit-learn  
- Pandas, NumPy  
- Matplotlib, Seaborn