In [ ]:
```
import os
import numpy as np
import pandas as pd
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

train_data = pd.read_csv("testy/toxic_eng/train.csv")
test_data = pd.read_csv("testy/toxic_eng/test.csv")

if "toxic" not in train_data.columns or "toxic" not in test_data.columns:
    raise ValueError("Missing 'toxic' column in the dataset!")

max_words = 20000
max_length = 128
tokenizer = Tokenizer(num_words=max_words, oov_token="<OOV>")
tokenizer.fit_on_texts(train_data['comment_text'])

train_sequences = tokenizer.texts_to_sequences(train_data['comment_text'])
test_sequences = tokenizer.texts_to_sequences(test_data['comment_text'])

train_padded = pad_sequences(train_sequences, maxlen=max_length, padding="post",
test_padded = pad_sequences(test_sequences, maxlen=max_length, padding="post", t

train_labels = np.array(train_data['toxic'])
test_labels = np.array(test_data['toxic'])



# v tomto pripade sme sigmoid dali preto lebo ide o binarnu klasifikaciu
model = Sequential([
    Embedding(max_words, 128, input_length=max_length),
    SimpleRNN(128, return_sequences=False),
    Dropout(0.3),
    Dense(64, activation="relu"),
    Dropout(0.3),
    Dense(1, activation="sigmoid")
])

model.compile(
    loss="binary_crossentropy",
    optimizer=Adam(learning_rate=0.001),
    metrics=["accuracy"]
)

model.fit(train_padded, train_labels, validation_data=(test_padded, test_labels)

test_loss, test_acc = model.evaluate(test_padded, test_labels)
print(f"Test Accuracy: {test_acc:.4f}")

model.save("final_rnn_model_tf.keras")
print("ulozime model do> final_rnn_model_tf.keras")
```

In [ ]:
```
from sklearn.metrics import precision_score, recall_score, f1_score
```

```
y_pred = model.predict(test_padded)
y_pred_binary = (y_pred > 0.5).astype(int)



precision = precision_score(test_labels, y_pred_binary)
recall = recall_score(test_labels, y_pred_binary)
f1 = f1_score(test_labels, y_pred_binary)

print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```