import tensorflow as tf

import numpy as np # Added import for NumPy

from tensorflow.keras.preprocessing.text import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad\_sequences

# Step 1: Prepare a dataset of labeled emails (spam and non-spam)

emails = [

"Buy cheap watches! Free shipping!",

"Meeting for lunch today?",

"Claim your prize! You've won $1,000,000!",

"Important meeting at 3 PM.",

]

labels = [1, 0, 1, 0]

# Step 2: Tokenize and pad the email text data

max\_words = 1000

max\_len = 50

tokenizer = Tokenizer(num\_words=max\_words, oov\_token="<OOV>")

tokenizer.fit\_on\_texts(emails)

sequences = tokenizer.texts\_to\_sequences(emails)

X\_padded = pad\_sequences(sequences, maxlen=max\_len, padding="post", truncating="post")

# Step 3: Define the neural network model

model = tf.keras.Sequential([

tf.keras.layers.Embedding(input\_dim=max\_words, output\_dim=16, input\_length=max\_len),

tf.keras.layers.Flatten(),

tf.keras.layers.Dense(16, activation='relu'),

tf.keras.layers.Dense(1, activation='sigmoid')

])

# Compile the model

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

# Step 4: Define training data and labels as NumPy arrays

# This is where you convert your training data and labels to NumPy arrays.

# For this example, we will use the same data as 'emails' and 'labels'.

training\_data = np.array(X\_padded)

training\_labels = np.array(labels)

# Step 5: Train the model

model.fit(training\_data, training\_labels, epochs=10) # You can adjust the number of epochs

# Step 6: Test if 'Spam.txt' is spam or not

file\_path = "Spam.txt"

# Read the content of the 'Sdam.txt' file

with open(file\_path, "r", encoding="utf-8") as file:

sample\_email\_text = file.read()

# Tokenize and pad the sample email text

sequences\_sample = tokenizer.texts\_to\_sequences([sample\_email\_text])

sample\_email\_padded = pad\_sequences(sequences\_sample, maxlen=max\_len, padding="post", truncating="post")

# Use the trained model to make predictions

prediction = model.predict(sample\_email\_padded)

# Set a classification threshold (e.g., 0.5)

threshold = 0.5

# Classify the sample email based on the threshold

if prediction > threshold:

print(f"Sample Email ('{file\_path}'): SPAM")

else:

print(f"Sample Email ('{file\_path}'): NOT SPAM")