```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import os
# Set direktori kerja ke folder yang berisi file Anda
os.chdir('/content/drive/My Drive/Colab Notebooks/Natural Language Processing/Task 2/')
```

```
!pip install numpy pandas tensorflow nltk rouge-score
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.26.4)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.2.2)
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.17.0)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: rouge-score in /usr/local/lib/python3.10/dist-packages (0.1.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=3.10.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.12.1)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (24.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.64.1)
Requirement already satisfied: tensorboard<2.18,>=2.17 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.17.0)
Requirement already satisfied: keras>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.37.1
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2024.9.11)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.6)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow) (0.44.
Requirement already satisfied: rich in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->tensorflow) (13.9.3)
Requirement already satisfied: namex in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->tensorflow) (0.13.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflow
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflow) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflow) (2.2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflow) (202
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.18,>=2.17->tensorflow) (3.
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.18,>
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.18,>=2.17->tensorflow) (3.
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->tensorboard<2.18,>=2.
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich->keras>=3.2.0->tensorflow) (3
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich->keras>=3.2.0->tensorflow)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0->rich->keras>=3.2.0->te
```

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense, TimeDistributed, Masking
from sklearn.model_selection import train_test_split


# Memuat dataset
df = pd.read_csv('Dataset/liputan6.csv')
df = df[['clean_article', 'clean_summary']]

# Lihat contoh data
df.head()
```

	clean_article	clean_summary
0	Liputan6. com, Surabaya : Radiogram Direktorat...	Gubernur Jatim Imam Utomo tak mau melantik Bup...
1	Liputan6. com, Jakarta : Berbeda dengan aliran...	Pelukis RM Koestarto memamerkan hasil karyanya...
2	Liputan6. com, Jambi : Ratusan orang dari Kesa...	Dua kelompok pengunjuk rasa di Jambi, menuntut...
3	Liputan6. com, Jakarta : Badan Penyehatan Perb...	BPPN masih mengkaji bank rekap yang dianggap p...

```python
# Menampilkan jumlah total data dalam dataset
print("Total data dalam dataset:", len(df))

# Menampilkan jumlah data yang ada di setiap kolom 'clean_article' dan 'clean_summary'
print("Total data di kolom 'clean_article':", df['clean_article'].notnull().sum())
print("Total data di kolom 'clean_summary':", df['clean_summary'].notnull().sum())

# Menjumlahkan total data non-null dari kedua kolom
total_non_null = df['clean_article'].notnull().sum() + df['clean_summary'].notnull().sum()
print("Total dari kedua kolom (clean_article dan clean_summary):", total_non_null)

# Menampilkan informasi jumlah kolom dan jenis data
print("\nInfo Dataset:")
df.info()
```

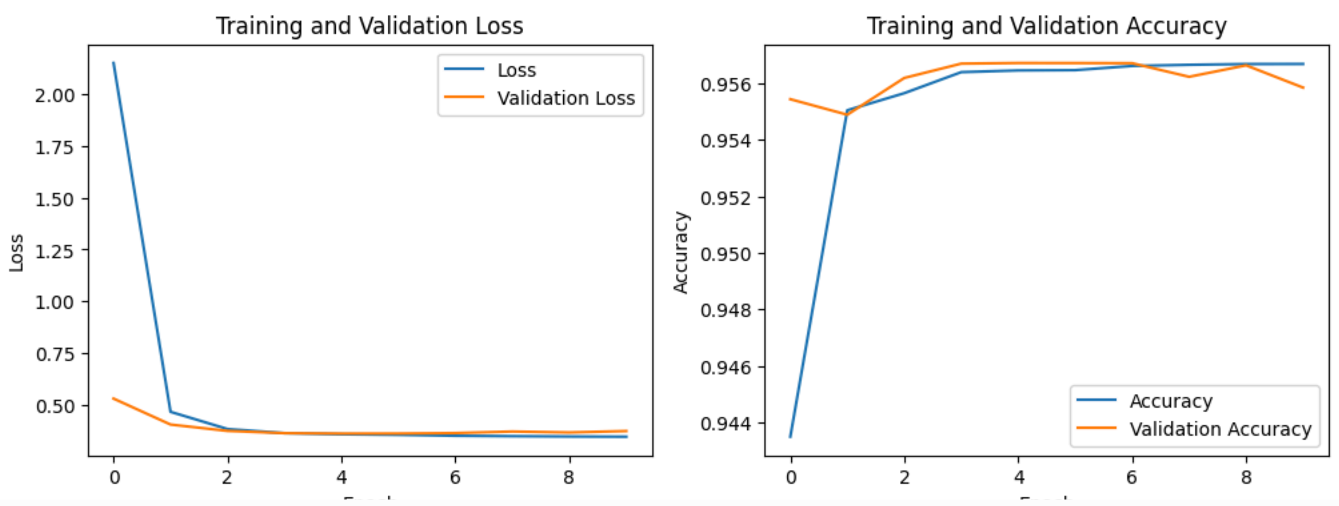
```
Total data dalam dataset: 10972
Total data di kolom 'clean_article': 10972
Total data di kolom 'clean_summary': 10972
Total dari kedua kolom (clean_article dan clean_summary): 21944

Info Dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10972 entries, 0 to 10971
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   clean_article  10972 non-null  object
 1   clean_summary  10972 non-null  object
dtypes: object(2)
memory usage: 171.6+ KB
```

```python
# Menggunakan Tokenizer untuk teks
tokenizer = Tokenizer()
tokenizer.fit_on_texts(df['clean_article'].tolist() + df['clean_summary'].tolist())

# Konversi teks menjadi sequences
input_sequences = tokenizer.texts_to_sequences(df['clean_article'].tolist())
target_sequences = tokenizer.texts_to_sequences(df['clean_summary'].tolist())

# Padding sequences untuk memastikan panjangnya seragam
max_input_len = 500  # Batasi panjang maksimum input sequence
max_target_len = 50  # Batasi panjang maksimum target sequence

# Padding sequences untuk memastikan panjangnya seragam
input_sequences = pad_sequences(input_sequences, maxlen=max_input_len, padding='post')
target_sequences = pad_sequences(target_sequences, maxlen=max_target_len, padding='post')

# Mendapatkan ukuran vocabulary
vocab_size = len(tokenizer.word_index) + 1
```

```
# Periksa panjang input dan target sequences yang sudah diproses
print("Panjang input sequences:", max_input_len)
print("Panjang target sequences:", max_target_len)

# Model dengan validasi
X_train, X_val, y_train, y_val = train_test_split(input_sequences, target_sequences, test_size=0.2, random_state=42)

# Menggunakan `max_input_len` untuk padding
y_train = pad_sequences(y_train, maxlen=max_input_len, padding='post')
y_val = pad_sequences(y_val, maxlen=max_input_len, padding='post')

# Model
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=64))
model.add(SimpleRNN(64, return_sequences=True))
model.add(TimeDistributed(Dense(vocab_size, activation='softmax')))
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Melatih model dengan validasi
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_val, y_val))
```

```
Panjang input sequences: 500
Panjang target sequences: 50
Epoch 1/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 803s 2s/step - accuracy: 0.8922 - loss: 4.8159 - val_accuracy: 0.9554 - val_loss: 0.5292
Epoch 2/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 236s 777ms/step - accuracy: 0.9548 - loss: 0.5095 - val_accuracy: 0.9549 - val_loss: 0.4037
Epoch 3/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 257s 759ms/step - accuracy: 0.9550 - loss: 0.3940 - val_accuracy: 0.9562 - val_loss: 0.3730
Epoch 4/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 266s 775ms/step - accuracy: 0.9563 - loss: 0.3630 - val_accuracy: 0.9567 - val_loss: 0.3620
Epoch 5/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 213s 775ms/step - accuracy: 0.9565 - loss: 0.3553 - val_accuracy: 0.9567 - val_loss: 0.3601
Epoch 6/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 257s 759ms/step - accuracy: 0.9564 - loss: 0.3535 - val_accuracy: 0.9567 - val_loss: 0.3605
Epoch 7/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 266s 775ms/step - accuracy: 0.9565 - loss: 0.3493 - val_accuracy: 0.9567 - val_loss: 0.3625
Epoch 8/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 262s 775ms/step - accuracy: 0.9568 - loss: 0.3458 - val_accuracy: 0.9562 - val_loss: 0.3699
Epoch 9/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 262s 775ms/step - accuracy: 0.9566 - loss: 0.3454 - val_accuracy: 0.9566 - val_loss: 0.3655
Epoch 10/10
275/275 ━━━━━━━━━━━━━━━━━━━━ 257s 759ms/step - accuracy: 0.9567 - loss: 0.3443 - val_accuracy: 0.9558 - val_loss: 0.3722
```

```
# Plotting Loss dan Accuracy
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



```
def summarize_text(input_text):
    # Proses teks input
    input_seq = tokenizer.texts_to_sequences([input_text])
    input_seq = pad_sequences(input_seq, maxlen=max_input_len, padding='post')

    # Prediksi output
    predicted_seq = model.predict(input_seq)
    predicted_seq = np.argmax(predicted_seq, axis=-1)[0]  # Ambil indeks dengan probabilitas tertinggi

    # Konversi output menjadi teks
    summary = ' '.join(tokenizer.index_word[idx] for idx in predicted_seq if idx > 0)
    return summary

# Mengambil input dari pengguna dan menampilkan ringkasan
input_text = input("Masukkan teks yang ingin diringkas: ")
ringkasan = summarize_text(input_text)
print("Ringkasan:", ringkasan)
```

```
Masukkan teks yang ingin diringkas: halo nama saya admin
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
Ringkasan: pemerintah dan
```