Notebook NameError

```
# Sentiment Analysis of tweets global from dataset on twitter
```

Sentiment Analysis of tweets global from dataset on twitter

Import library

```
#library for pre-processing(cleaning data, transfromation data,  and processing tokenizer)
import pandas as pd
import re,string,unicodedata
import nltk

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from string import punctuation
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.model_selection import train_test_split


#library for model building
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import matplotlib.pyplot as plt
import datetime, os

```

```
pip install -U notebook-as-pdf
```

```
ady satisfied: notebook-as-pdf in /usr/local/lib/python3.10/dist-packages (0.5.0)
ady satisfied: nbconvert in /usr/local/lib/python3.10/dist-packages (from notebook-as-pdf) (6.5.4)
ady satisfied: pyppeteer in /usr/local/lib/python3.10/dist-packages (from notebook-as-pdf) (2.0.0)
ady satisfied: PyPDF2 in /usr/local/lib/python3.10/dist-packages (from notebook-as-pdf) (3.0.1)
ady satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (4.9.4)
ady satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (4.12.3)
ady satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (6.1.0)
ady satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (0.7.1)
ady satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (0.4)
ady satisfied: jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (3.1.4)
ady satisfied: jupyter-core>=4.7 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (5.7.2)
ady satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (0.3.0)
ady satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (2.1.5)
ady satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (0.8.4)
ady satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (0.10.0)
ady satisfied: nbformat>=5.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (5.10.4)
ady satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (24.1)
ady satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (1.5.1)
ady satisfied: pygments>=2.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (2.16.1)
ady satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (1.3.0)
ady satisfied: traitlets>=5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->notebook-as-pdf) (5.7.1)
ady satisfied: appdirs<2.0.0,>=1.4.3 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (1.4.4)
ady satisfied: certifi>=2023 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (2024.7.4)
ady satisfied: importlib-metadata>=1.4 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (8.0.0)
ady satisfied: pyee<12.0.0,>=11.0.0 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (11.1.0)
ady satisfied: tqdm<5.0.0,>=4.42.1 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (4.66.4)
ady satisfied: urllib3<2.0.0,>=1.25.8 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (1.26.19)
ady satisfied: websockets<11.0,>=10.0 in /usr/local/lib/python3.10/dist-packages (from pyppeteer->notebook-as-pdf) (10.4)
ady satisfied: zipp>=0.5 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata>=1.4->pyppeteer->notebook-as-pdf) (3.19
ady satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.7->nbconvert->notebook-as-pdf) (4.
ady satisfied: jupyter-client>=6.1.12 in /usr/local/lib/python3.10/dist-packages (from nbclient>=0.5.0->nbconvert->notebook-as-pdf)
ady satisfied: fastjsonschema>=2.15 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert->notebook-as-pdf) (2.2
ady satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert->notebook-as-pdf) (4.19.2)
ady satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from pyee<12.0.0,>=11.0.0->pyppeteer->notebook-as-pdf)
ady satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->nbconvert->notebook-as-pdf) (2.5)
ady satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert->notebook-as-pdf) (1.16.0)
ady satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert->notebook-as-pdf) (0.5.1)
ady satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert->notebook-as
ady satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>=5.1-
ady satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert->noteb
ady satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert->notebook-a
ady satisfied: pyzmq>=13 in /usr/local/lib/python3.10/dist-packages (from jupyter-client>=6.1.12->nbclient>=0.5.0->nbconvert->notebo
ady satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-client>=6.1.12->nbclient>=0.5.0->nbconv
ady satisfied: tornado>=4.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-client>=6.1.12->nbclient>=0.5.0->nbconvert->not
!pip install pyppeteer
```

```
Requirement already satisfied: pyppeteer in /usr/local/lib/python3.10/dist-packages (2.0.0)
Requirement already satisfied: appdirs<2.0.0,>=1.4.3 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (1.4.4)
Requirement already satisfied: certifi>=2023 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (2024.7.4)
Requirement already satisfied: importlib-metadata>=1.4 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (8.0.0)
Requirement already satisfied: pyee<12.0.0,>=11.0.0 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (11.1.0)
Requirement already satisfied: tqdm<5.0.0,>=4.42.1 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (4.66.4)
Requirement already satisfied: urllib3<2.0.0,>=1.25.8 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (1.26.19)
Requirement already satisfied: websockets<11.0,>=10.0 in /usr/local/lib/python3.10/dist-packages (from pyppeteer) (10.4)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata>=1.4->pyppeteer) (3.19
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from pyee<12.0.0,>=11.0.0->pyppeteer)
```

Preprocessing

```
df = pd.read_csv('new.csv')
df.head(200)
```

	Sl no	Tweets	Search key	Feeling
0	1	#1: @fe ed "RT @MirayaDizon1: Time is ticking...	happy moments	happy
1	2	#2: @蓮花 &はすか ed "RT @ninjaryugo: ＃コナモンの日 だそうで...	happy moments	happy
2	3	#3: @Ris ♡ ed "Happy birthday to one smokin h...	happy moments	happy
3	4	#4: @월월 [씩찐사랑로봇] jwinnie is the best, cheer u...	happy moments	happy
4	5	#5: @Madhurima wth u vc♥ ed "Good morning dea...	happy moments	happy
...	...	...	...	...
195	196	#44: @Issac Guerrero ed "I'm never satisfied,...	satisfied	happy
196	197	#45: @Bryan ed "RT @LunkersTV: I somewhat fee...	satisfied	happy
197	198	#46: @Rachel Joy Larris ed "RT @Blackamazon: ...	satisfied	happy

Langkah berikutnya: Buat kode dengan df Lihat plot yang direkomendasikan

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10017 entries, 0 to 10016
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Sl no       10017 non-null  int64 
 1   Tweets      10017 non-null  object
 2   Search key  10017 non-null  object
 3   Feeling     10017 non-null  object
dtypes: int64(1), object(3)
memory usage: 313.2+ KB
```

```

nltk.download('stopwords')
nltk.download('punkt')

def clean_text(Text):
    if isinstance(Text, str):
        Text = re.sub(r'http\S+', '', Text)
        Text = re.sub(r'[^a-zA-Z\s]', '', Text)
        Text = Text.lower()
        Text = Text.strip()
        tokens = word_tokenize(Text)
        stop_words = set(stopwords.words('english'))
        tokens = [word for word in tokens if word not in stop_words]
        Text = ' '.join(tokens)
        return Text
    else:
        return ''

df['cleaned_content'] = df['Tweets'].apply(clean_text)
df.head()

```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

	Sl no	Tweets	Search key	Feeling	cleaned_content
0	1	#1: @fe ed "RT @MirayaDizon1: Time is ticking...	happy moments	happy	fe ed rt mirayadizon time ticking fast relive ...
1	2	#2: @蓮花 &はすか ed "RT @ninjaryugo: #コナモンの日 だそうで...	happy moments	happy	ed rt ninjaryugo
		#3: @Ris ♡ ed "Happy	happy		ris ed happy birthday one

Langkah berikutnya: Buat kode dengan `df` Lihat plot yang direkomendasikan

```
labels = pd.get_dummies(df['Feeling'])
df_baru = pd.concat([df, labels], axis=1)
df_baru = df_baru.drop(columns=['Feeling'])
df_baru.head(20)
```

	Sl no	Tweets	Search key	cleaned_content	angry	disgust	fear	happy	
0	1	#1: @fe ed "RT @MirayaDizon1: Time is ticking...	happy moments	fe ed rt mirayadizon time ticking fast relive ...	False	False	False	True	F
1	2	#2: @蓮花 &はすか ed "RT @ninjaryugo: #コナモンの日 だそうで...	happy moments	ed rt ninjaryugo	False	False	False	True	F
2	3	#3: @Ris ♡ ed "Happy birthday to one smokin h...	happy moments	ris ed happy birthday one smokin hot mama love...	False	False	False	True	F
3	4	#4: @월월 [씩찐사랑로봇] jwinnie is the best, cheer u...	happy moments	jwinnie best cheer jwinnie ed omg left min msg...	False	False	False	True	F
4	5	#5: @Madhurima wth u vc♥ ed "Good morning dea...	happy moments	madhurima wth u vc ed good morning dear vikram...	False	False	False	True	F
5	6	#6: @Jeinalís Ramos ed "Happy moments 🙏 http...	happy moments	jeinals ramos ed happy moments	False	False	False	True	F
6	7	#7: @Eric Rogers ed "@CaitlinUnruh The movie ...	happy moments	eric rogers ed caitlinunruh movie made happy s...	False	False	False	True	F
7	8	#8: @Yanny Sandal ed "I don't give two shits ...	happy moments	yanny sandal ed dont give two shits met gala w...	False	False	False	True	F
8	9	#9: @daynada ed "my beautiful barbie bride an...	happy moments	daynada ed beautiful barbie bride fixed dress ...	False	False	False	True	F
9	10	#10: @ß🌪 ed "Someone Great has been one of th...	happy moments	ed someone great one best chick flicks romance...	False	False	False	True	F
10	11	#11: @[17's Maga] 💎 #JONASBROTHERS ed "RT @pl...	happy moments	maga jonasbrothers ed rt pledisjp seventeen ja...	False	False	False	True	F
11	12	#12: @Mj ed "RT @AshTanDefenders: AshTan Mome...	happy moments	mj ed rt ashtandefenders ashtan moments best m...	False	False	False	True	F
12	13	#13: @Frances Kaye Alvarez ed "RT	happy moments	frances kaye alvarez ed rt ashtandefenders	False	False	False	True	F

Langkah berikutnya: Buat kode dengan `df_baru` Lihat plot yang direkomendasikan

```python
tweets = df_baru['cleaned_content'].values
emotion =  df_baru[['angry', 'disgust', 'surprise','sad', 'happy', 'fear']].values
tweets_train, tweets_test, emotion_train, emotion_test = train_test_split(tweets, emotion, test_size=0.2)

```

```python
# tweets = df_baru['cleaned_content'].values
# emotion =  df_baru[uniq].values
# tweets_train, tweets_test, emotion_train, emotion_test = train_test_split(tweets, emotion, test_size=0.2)

```

```python
# tokenizer = Tokenizer(num_words=5000, oov_token='x')
# tokenizer.fit_on_texts(tweets_train)
# tokenizer.fit_on_texts(tweets_test)


# sekuens_train = tokenizer.texts_to_sequences(tweets_train)
# sekuens_test = tokenizer.texts_to_sequences(tweets_test)

# padded_train = pad_sequences(sekuens_train,
#                              truncating = 'post',
#                              padding = 'post',
#                              maxlen=maxlen)
# padded_test = pad_sequences(sekuens_test,
#                             truncating = 'post',
#                              padding = 'post',
#                              maxlen=maxlen)
```

```python
tokenizer = Tokenizer(num_words=5000, oov_token='x')
tokenizer.fit_on_texts(tweets_train)
tokenizer.fit_on_texts(tweets_test)


sekuens_train = tokenizer.texts_to_sequences(tweets_train)
sekuens_test = tokenizer.texts_to_sequences(tweets_test)

padded_train = pad_sequences(sekuens_train,
                             truncating = 'post',
                             padding = 'post',
                            )
padded_test = pad_sequences(sekuens_test,
                            truncating = 'post',
                             padding = 'post',
                             )
```

Versi pakai transformer

```python
class TransformerBlock(layers.Layer):
    def __init__(self, embed_dim, num_heads, ff_dim, rate=0.1):
        super().__init__()
        self.att = layers.MultiHeadAttention(num_heads=num_heads, key_dim=embed_dim)
        self.ffn = keras.Sequential(
            [layers.Dense(ff_dim, activation="relu"), layers.Dense(embed_dim),]
        )
        self.layernorm1 = layers.LayerNormalization(epsilon=1e-6)
        self.layernorm2 = layers.LayerNormalization(epsilon=1e-6)
        self.dropout1 = layers.Dropout(rate)
        self.dropout2 = layers.Dropout(rate)

    def call(self, inputs, training):
        attn_output = self.att(inputs, inputs)
        attn_output = self.dropout1(attn_output, training=training)
        out1 = self.layernorm1(inputs + attn_output)
        ffn_output = self.ffn(out1)
        ffn_output = self.dropout2(ffn_output, training=training)
        return self.layernorm2(out1 + ffn_output)
```

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

# inputs = layers.Input(shape=(None,))
# encoder = layers.Embedding(vocab_size, 16)(inputs)
# transformer_block = TransformerBlock(embed_dim=16, num_heads=2, ff_dim=32)
# encoder = transformer_block(encoder)

# # Tambahkan LSTM di sini
# encoder = layers.LSTM(128)(encoder)

# outputs = layers.Reshape((-1, 128))(encoder)
# encoder = layers.Dropout(0.1)(encoder)
# outputs = layers.GlobalMaxPooling1D()(outputs)
# outputs = layers.Dropout(0.1)(outputs)

# outputs = layers.Dense(128, activation='relu')(outputs)
# outputs = layers.Dropout(0.1)(outputs)
# outputs = layers.Dense(64, activation='relu')(outputs)
# outputs = layers.Dropout(0.1)(outputs)
# outputs = layers.Dense(6, activation='softmax')(outputs)

# model = keras.Model(inputs=inputs, outputs=outputs)
```

```
vocab_size = len(tokenizer.word_index) + 1

inputs = layers.Input(shape=(None,))
encoder = layers.Embedding(vocab_size, 16)(inputs)
transformer_block = TransformerBlock(embed_dim=16, num_heads=2, ff_dim=32)
encoder = transformer_block(encoder)

# Tambahkan LSTM di sini
encoder = layers.LSTM(64)(encoder)
encoder = layers.Dropout(0.2)(encoder)

outputs = layers.Reshape((-1, 64))(encoder)
outputs = layers.GlobalMaxPooling1D()(outputs)
outputs = layers.Dropout(0.1)(outputs)

outputs = layers.Dense(64, activation='relu')(outputs)
outputs = layers.Dropout(0.15)(outputs)
outputs = layers.Dense(32, activation='relu')(outputs)
outputs = layers.Dropout(0.15)(outputs)
outputs = layers.Dense(6, activation='softmax')(outputs)

model = keras.Model(inputs=inputs, outputs=outputs)
```

Model building

```
# model = tf.keras.Sequential([
#     tf.keras.layers.Embedding(input_dim = 5000, output_dim=32),
#     tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True, kernel_regularizer=tf.keras.regularizers.l2(0.0
#     tf.keras.layers.GlobalMaxPooling1D(),
#     tf.keras.layers.Dense(512, activation='relu'),
#     tf.keras.layers.Dropout(0.1),
#     tf.keras.layers.Dense(256, activation='relu'),
#     tf.keras.layers.Dropout(0.1),
#     tf.keras.layers.Dense(128, activation='relu'),
#     tf.keras.layers.Dropout(0,1),
#     tf.keras.layers.Dense(6, activation='softmax'),
#     ])
```

```
# model = tf.keras.Sequential([
#     tf.keras.layers.Embedding(input_dim=5000, output_dim=16),
#     tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True, kernel_regularizer=tf.keras.regularizers.l2(0.0
#     tf.keras.layers.GlobalMaxPooling1D(),
#     tf.keras.layers.Dense(256, activation='relu'),
#     tf.keras.layers.Dropout(0.5),
#     tf.keras.layers.Dense(128, activation='relu'),
#     tf.keras.layers.Dropout(0.5),
#     tf.keras.layers.Dense(64, activation='relu'),
#     # tf.keras.layers.Dropout(0.5),
#     tf.keras.layers.Dense(6, activation='softmax')
# ])

```

```
%load_ext tensorboard
logdir = os.path.join("logs", datetime.datetime.now().strftime("%Y%M%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(logdir, histogram_freq = 1)
```

```
# custom_optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)
# model.compile(loss='categorical_crossentropy', optimizer=custom_optimizer, metrics=['accuracy'])
# model.summary()
```

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, None)]            0

 embedding (Embedding)       (None, None, 16)          571520

 transformer_block (Transfo  (None, None, 16)          3296
 rmerBlock)

 lstm (LSTM)                 (None, 64)                20736

 dropout_2 (Dropout)         (None, 64)                0

 reshape (Reshape)           (None, 1, 64)             0

 global_max_pooling1d (Glob  (None, 64)                0
 alMaxPooling1D)

 dropout_3 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 64)                4160

 dropout_4 (Dropout)         (None, 64)                0

 dense_3 (Dense)             (None, 32)                2080

 dropout_5 (Dropout)         (None, 32)                0

 dense_4 (Dense)             (None, 6)                 198

=================================================================
Total params: 601990 (2.30 MB)
Trainable params: 601990 (2.30 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
class AccuracyHistory(tf.keras.callbacks.Callback):
    def on_epoch_end(self, epoch, logs={}):
        if self.has_reached_accuracy(logs):
            print(' Stop training model, val_accuracy > 90%')
            self.model.stop_training = True

    def has_reached_accuracy(self, logs):
        return (logs.get('accuracy') > 0.90 and  logs.get('val_accuracy') > 0.90)

callbacks = AccuracyHistory()

num_epochs = 30
history = model.fit(padded_train,
                    emotion_train,
                    epochs=num_epochs,
                    validation_data=(padded_test, emotion_test),
                    verbose=2,
                    callbacks=[callbacks, tensorboard_callback])
```

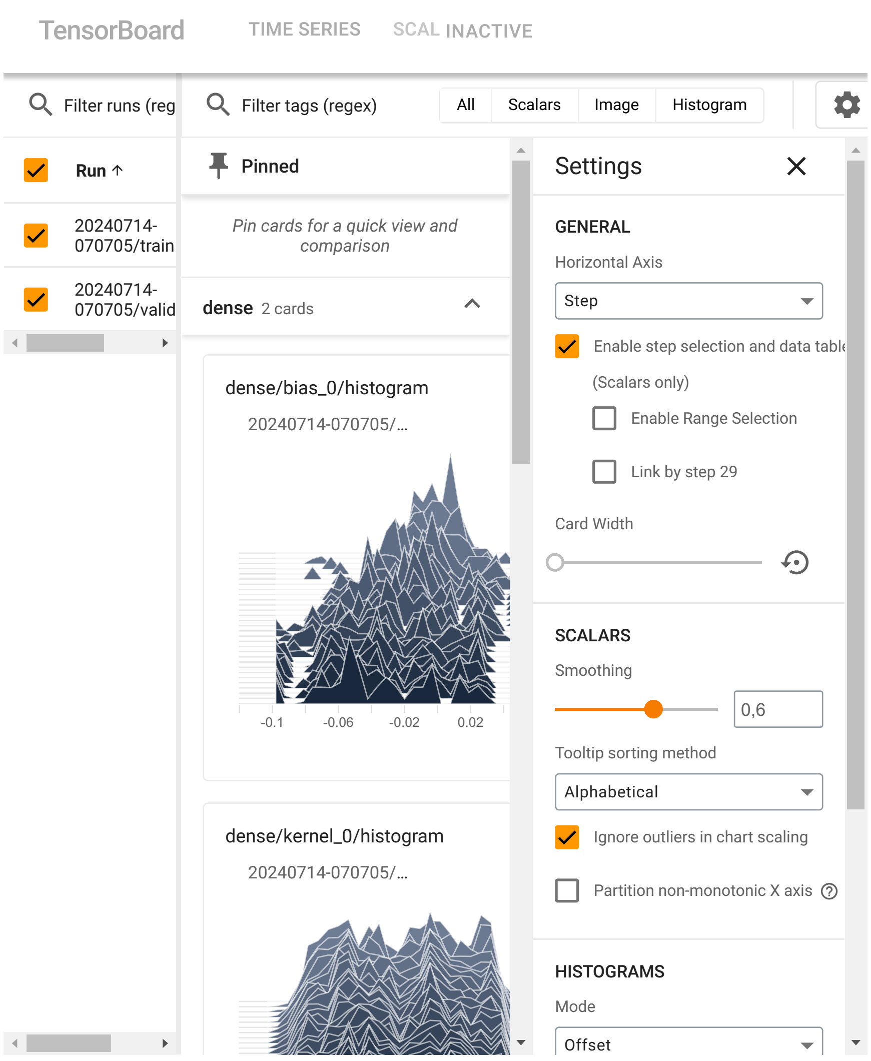
```
251/251 - 28s - loss: 1.5614 - accuracy: 0.3744 - val_loss: 1.5113 - val_accuracy: 0.3872 - 28s/epoch - 112ms/step
Epoch 2/30
251/251 - 19s - loss: 1.2244 - accuracy: 0.5148 - val_loss: 0.8727 - val_accuracy: 0.6861 - 19s/epoch - 78ms/step
Epoch 3/30
251/251 - 17s - loss: 0.6615 - accuracy: 0.7666 - val_loss: 0.4898 - val_accuracy: 0.8418 - 17s/epoch - 68ms/step
Epoch 4/30
251/251 - 17s - loss: 0.3840 - accuracy: 0.8735 - val_loss: 0.4627 - val_accuracy: 0.8538 - 17s/epoch - 69ms/step
Epoch 5/30
251/251 - 19s - loss: 0.2818 - accuracy: 0.9116 - val_loss: 0.4368 - val_accuracy: 0.8817 - 19s/epoch - 74ms/step
Epoch 6/30
251/251 - 18s - loss: 0.2389 - accuracy: 0.9245 - val_loss: 0.4291 - val_accuracy: 0.8867 - 18s/epoch - 70ms/step
Epoch 7/30
251/251 - 17s - loss: 0.2092 - accuracy: 0.9367 - val_loss: 0.4671 - val_accuracy: 0.8842 - 17s/epoch - 68ms/step
Epoch 8/30
251/251 - 17s - loss: 0.1781 - accuracy: 0.9447 - val_loss: 0.4721 - val_accuracy: 0.8812 - 17s/epoch - 68ms/step
Epoch 9/30
```

```
251/251 - 17s - loss: 0.1482 - accuracy: 0.9533 - val_loss: 0.5360 - val_accuracy: 0.8787 - 17s/epoch - 67ms/step
Epoch 11/30
251/251 - 17s - loss: 0.1311 - accuracy: 0.9581 - val_loss: 0.5477 - val_accuracy: 0.8762 - 17s/epoch - 67ms/step
Epoch 12/30
251/251 - 18s - loss: 0.1171 - accuracy: 0.9628 - val_loss: 0.5659 - val_accuracy: 0.8847 - 18s/epoch - 70ms/step
Epoch 13/30
251/251 - 18s - loss: 0.1276 - accuracy: 0.9609 - val_loss: 0.5697 - val_accuracy: 0.8802 - 18s/epoch - 70ms/step
Epoch 14/30
251/251 - 19s - loss: 0.1223 - accuracy: 0.9637 - val_loss: 0.5890 - val_accuracy: 0.8757 - 19s/epoch - 75ms/step
Epoch 15/30
251/251 - 17s - loss: 0.0995 - accuracy: 0.9664 - val_loss: 0.6596 - val_accuracy: 0.8728 - 17s/epoch - 68ms/step
Epoch 16/30
251/251 - 17s - loss: 0.1239 - accuracy: 0.9624 - val_loss: 0.5997 - val_accuracy: 0.8748 - 17s/epoch - 68ms/step
Epoch 17/30
251/251 - 19s - loss: 0.1038 - accuracy: 0.9654 - val_loss: 0.6777 - val_accuracy: 0.8728 - 19s/epoch - 76ms/step
Epoch 18/30
251/251 - 17s - loss: 0.0943 - accuracy: 0.9673 - val_loss: 0.7018 - val_accuracy: 0.8748 - 17s/epoch - 66ms/step
Epoch 19/30
251/251 - 17s - loss: 0.0929 - accuracy: 0.9681 - val_loss: 0.6497 - val_accuracy: 0.8733 - 17s/epoch - 68ms/step
Epoch 20/30
251/251 - 17s - loss: 0.0866 - accuracy: 0.9704 - val_loss: 0.8135 - val_accuracy: 0.8733 - 17s/epoch - 67ms/step
Epoch 21/30
251/251 - 19s - loss: 0.1406 - accuracy: 0.9583 - val_loss: 0.6273 - val_accuracy: 0.8767 - 19s/epoch - 77ms/step
Epoch 22/30
251/251 - 17s - loss: 0.1069 - accuracy: 0.9631 - val_loss: 0.6203 - val_accuracy: 0.8733 - 17s/epoch - 67ms/step
Epoch 23/30
251/251 - 18s - loss: 0.0835 - accuracy: 0.9674 - val_loss: 0.7502 - val_accuracy: 0.8678 - 18s/epoch - 74ms/step
Epoch 24/30
251/251 - 20s - loss: 0.0816 - accuracy: 0.9707 - val_loss: 0.7242 - val_accuracy: 0.8777 - 20s/epoch - 78ms/step
Epoch 25/30
251/251 - 17s - loss: 0.0739 - accuracy: 0.9717 - val_loss: 0.7346 - val_accuracy: 0.8738 - 17s/epoch - 69ms/step
Epoch 26/30
251/251 - 17s - loss: 0.0804 - accuracy: 0.9688 - val_loss: 0.7478 - val_accuracy: 0.8738 - 17s/epoch - 68ms/step
Epoch 27/30
251/251 - 17s - loss: 0.0793 - accuracy: 0.9710 - val_loss: 0.7373 - val_accuracy: 0.8817 - 17s/epoch - 68ms/step
Epoch 28/30
251/251 - 19s - loss: 0.0838 - accuracy: 0.9700 - val_loss: 0.7252 - val_accuracy: 0.8748 - 19s/epoch - 77ms/step
Epoch 29/30
251/251 - 17s - loss: 0.0775 - accuracy: 0.9702 - val_loss: 0.7358 - val_accuracy: 0.8792 - 17s/epoch - 67ms/step
Epoch 30/30
251/251 - 17s - loss: 0.0784 - accuracy: 0.9702 - val_loss: 0.7643 - val_accuracy: 0.8728 - 17s/epoch - 66ms/step
```

Melihat plot akurasi training dan testing dengan tensorboard

```
Mulai coding atau buat kode dengan AI.
```

```
%tensorboard --logdir logs
```



Settings

GENERAL

Horizontal Axis: Step

✓ Enable step selection and data table (Scalars only)

☐ Enable Range Selection

☐ Link by step 29

Card Width

SCALARS

Smoothing: 0,6

Tooltip sorting method: Alphabetical

✓ Ignore outliers in chart scaling

☐ Partition non-monotonic X axis

HISTOGRAMS

Mode: Offset

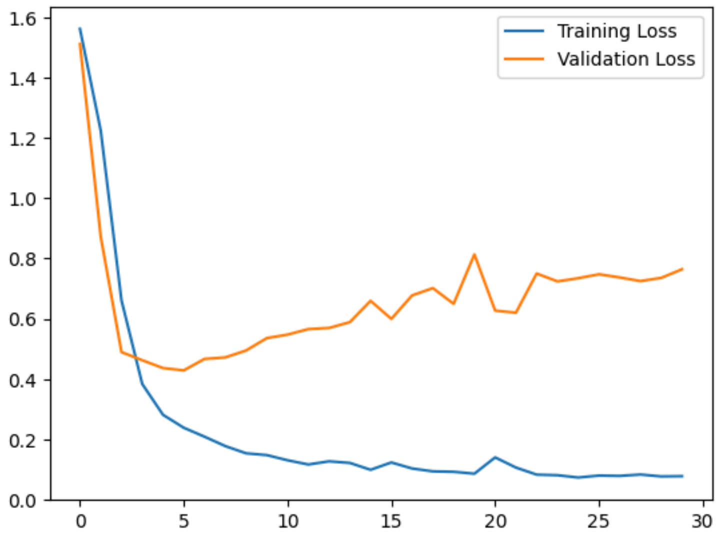
Melihat plot akurasi training dan testing dengan library matplotlib

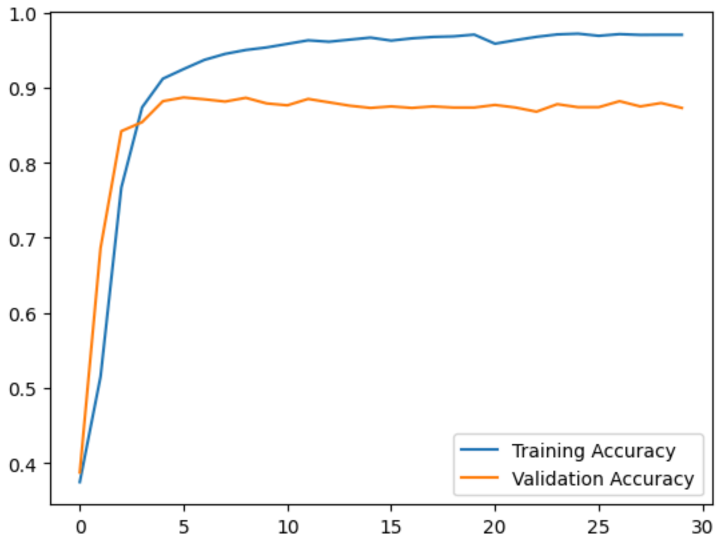
```
import matplotlib.pyplot as plt

# Menampilkan grafik loss
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.show()

# Menampilkan grafik akurasi
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.show()

```





```
import numpy as np
maxlen = max([len(seq) for seq in sekuens_train])

def predict_sentiment(text, maxlen=maxlen):  # Use calculated maxlen
  # Convert text to sequence
  sequence = tokenizer.texts_to_sequences([text])
  sequence = pad_sequences(sequence, maxlen=maxlen, truncating='post')

  prediction = model.predict(sequence)

  predicted_class = np.argmax(prediction)

  sentiment_labels = ['angry', 'disgust', 'surprise', 'sad', 'happy', 'fear']
  predicted_sentiment = sentiment_labels[predicted_class]

  print(f"Predicted sentiment: {predicted_sentiment}")

# Example usage
text = "hey, don't touch me!"
predict_sentiment(text)
```

```
1/1 [==============================] - 0s 28ms/step
Predicted sentiment: angry
```

```
Mulai coding atau buat kode dengan AI.
```

```
Collecting notebook-as-pdf
  Downloading notebook_as_pdf-0.5.0-py3-none-any.whl (6.5 kB)
Requirement already satisfied: nbconvert in /usr/local/lib/python3.10/dist-packages
Collecting pyppeteer (from notebook-as-pdf)
  Downloading pyppeteer-2.0.0-py3-none-any.whl (82 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.9/82.9 kB 2.7 MB/s eta 0:00:00
Collecting PyPDF2 (from notebook-as-pdf)
  Downloading pypdf2-3.0.1-py3-none-any.whl (232 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 232.6/232.6 kB 6.8 MB/s eta 0:00:00
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: jinja2>=3.0 in /usr/local/lib/python3.10/dist-package
```