Analisis Sentimen Hate Speech Tweet di Indonesia dengan Metode Convolutional Neural Network (CNN) & Long Short Term Memory (LSTM)

Dessen

Elpi Sandra Yunvi

(Binar Academy)

Pendahuluan

Pendahuluan

Latar Belakang

Twitter merupakan salah satu media sosial yang sering digunakan oleh masyarakat Indonesia untuk menyampaikan opini. Opini tersebut dirangkum dalam 180 karakter yang disebut sebagai cuitan. Berdasarkan data jumlah pengguna Twitter di Indonesia pada tahun 2022 mencapai 18,45 juta pengguna.[1]

Jumlah tersebut tentu menimbulkan berbagai jenis sentimen dari cuitan yang akan berdampak pada cara bersosial media. Hal tersebut dapat digambar dari sentimen yang diberikan dari setiap tweet penggunanya.

Sehingga diharapkan dengan adanya penelitian ini dapat memudahkan pengguna dalam mengklasifikasikan sentimen dan mengukur apakah Twitter merupakan media sosial yang sehat atau perlu ditingkatkan kembali untuk kenyamanan penggunanya.

[1] Monavia, A. R. (2022).Diakses dari https://dataindonesia.id/digital/detail/pengguna-twitter-di-indonesia-capai-1845-juta-pada-2022

Pendahuluan

Tujuan

1. Melakukan klasifikasi pada sentimen tweet dengan menggunakan model mesin learning CNN dan LSTM yang dapat memproses text dan file pada API.
2. Mengevaluasi model CNN dan LSTM dari hasil akurasi yang diperoleh.
3. Melihat tingkat bermedia sosial di twitter yang masih tergolong sehat atau tidak.
4. Topik yang sering dibahas pada sentimen negatif oleh pengguna twitter.

Analytics Process

Analytics Process

BINAR

Identify problem

Data Exploratory

- Cek jumlah data
- Cek jumlah label
- Cek data Null
- Cek data duplikat

Data Preparation

- Cleansing data dengan regex
- Normalisasi
- Stopword

Make training models

CNN dan LSTM

Evaluation models

Dilakukan testing pada model untuk melihat apakah model sudah baik atau belum dengan menggunakan data dummy.

Metode Penelitian

Metode Penelitian

BINAR

Sumber Data

Sumber data pada penelitian ini merupakan data sekunder dimana diambil peneliti dari situs open source Kaggle yang dapat dilihat melalui link berikut. Data yang dianalisis adalah data tweet dalam bahasa indonesia dari platform social media Twitter.

Jenis Analisis

Jenis analisis yang dipakai peneliti adalah Descriptive dan Predictive Analytics. Descriptive Analytics bertujuan mengetahui dan mempelajari kondisi dan pola dari suatu data menggunakan berbagai operasi matematika dan statistika. Predictive bertujuan untuk memprediksi suatu data di masa depan menggunakan data di masa lampau

Jenis Exploratory Data Analysis (EDA) yang digunakan adalah 1 variabel (Univariate Analysis). Proses analisis yang digunakan adalah metode statistik deskriptif dan visualisasi data. Deskriptif statistik digunakan untuk mengetahui persebaran data. Visualisasi data dipakai untuk memudahkan pembaca memahami tren data tweet hate speech di Indonesia.

Data Preparation

Cek Data Duplicates & Missing Value

```
1 df_train.duplicated().sum()
```

```
67
```

```
1 df_train = df_train.drop_duplicates()
2 df_train.duplicated().sum()
```

```
0
```

Duplicate Data

Terdapat 67 data duplikat dan sudah dihapus pada dataframe

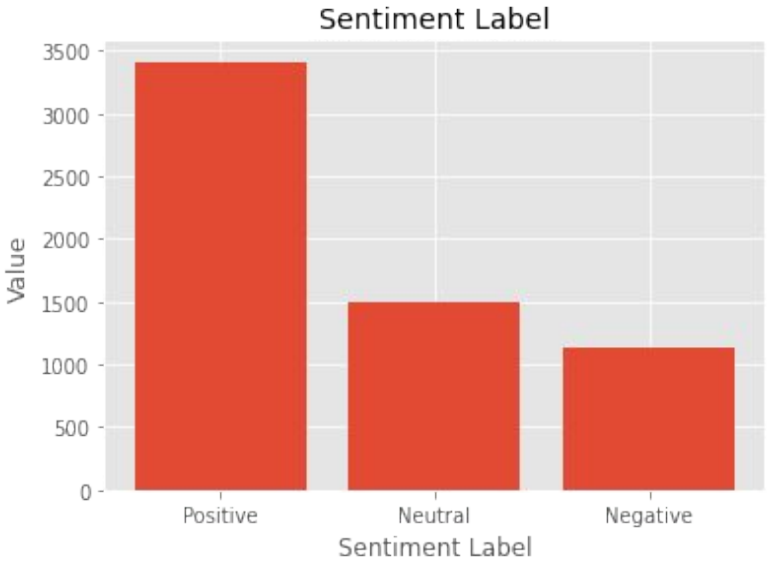
```
1 df_train.isnull().sum()
```

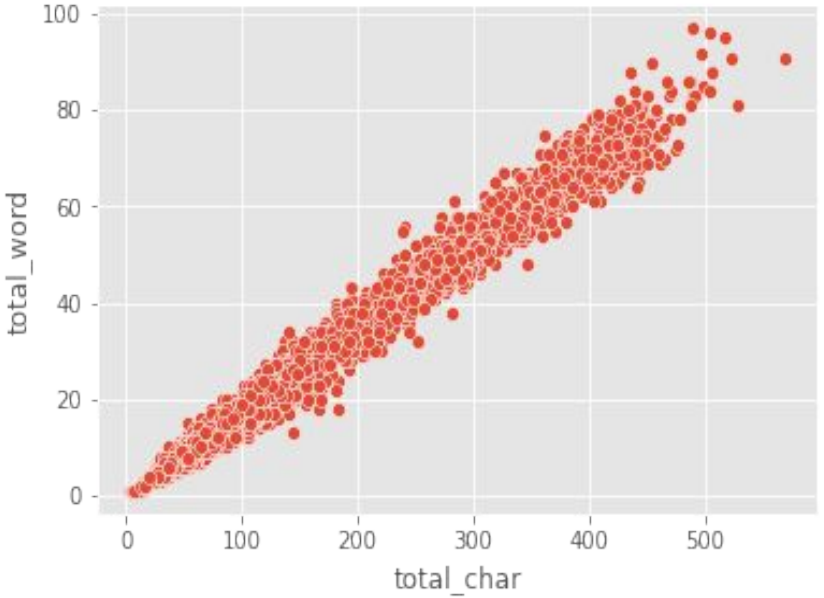
```
text     0
label    0
dtype: int64
```

Missing Value

Tidak terdapat missing value pada dataframe

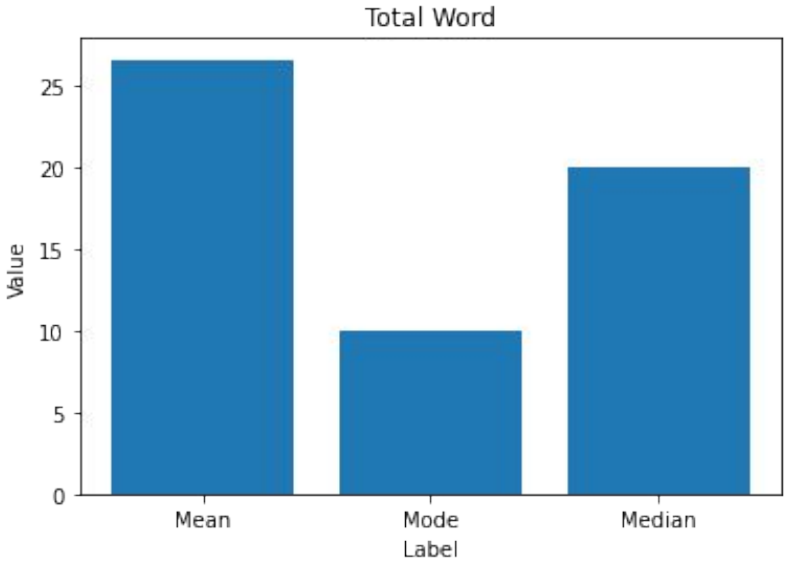
Data Preparation





Pada data train yang digunakan sentimen yang mendominasi yaitu positif, kemudian neutral, dan negatif. Untuk korelasi antara total kata dan total huruf dapat dilihat korelasinya linear.

Data Preparation




```
1 df['total_word'].mean()
```

```
26.59010425285454
```

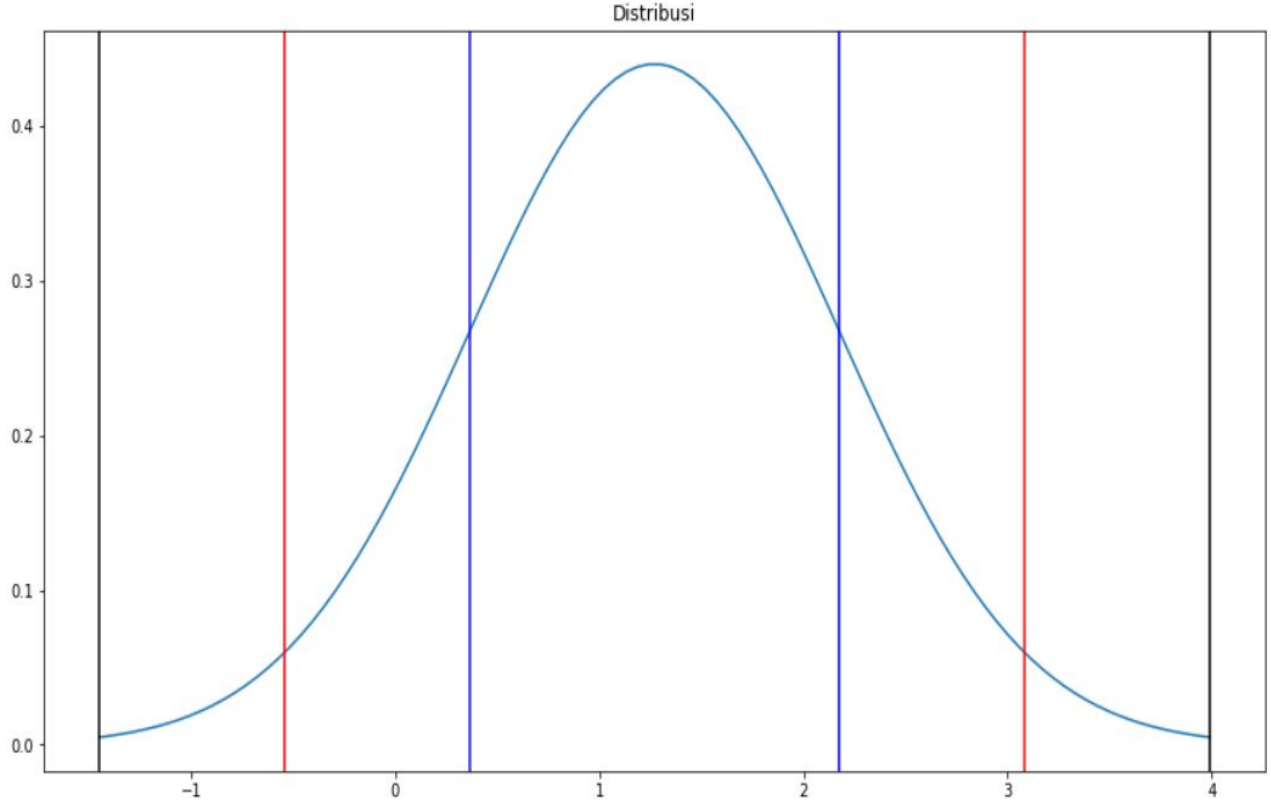
```
1 df['total_word'].mode()
```

```
0    10
dtype: int64
```

```
1 df['total_word'].median()
```

```
20.0
```

Data Preparation




```
1 df.mean()
```

```
<ipython-input-5-c61f0c8
  df.mean()
label    1.270909
dtype: float64
```

```
1 df.std()
```

```
<ipython-input-8-ce
  df.std()
label    0.906818
dtype: float64
```

Data Cleansing

BINAR

Regex

```python
def regex(text):
    text = re.sub('USER', '', text) #Remove USER
    text = re.sub('RT', '', text) #Remove RT
    text = re.sub('URL', '', text) #Remove URL
    text = re.sub(r'\\n+', '', text) #Remove \n
    text = re.sub(r'https\S+','', text) #Remove https
    text = re.sub(r'\\x[A-Za-z0-9./]+', '', text) #Remove \x96 etc
    text = re.sub('#[A-Za-z0-9./]+', '', text) #Remove hastag
    text = re.sub('  +', '', text) #Remove extra space
    text = text.lower() #Lowercase text
    text = re.sub('[^a-zA-Z]+', ' ', text) #Remove non alpha numeric
    return text
df_train['text'] = df_train['text'].apply(regex)
```

Untuk menghapus karakter tertentu yang tidak berguna untuk analisis

Normalization

```python
alay_dict_map = dict(zip(alay_dict['original'], alay_dict['replacement']))
def normalize_alay(text):
    return ' '.join([alay_dict_map[word] if word in alay_dict_map else word for word in text.split(' ')])
```

```python
def normalization(text):
    text = normalize_alay(text)
    return text
df_train['text'] = df_train['text'].apply(normalization)
```

Untuk membuat kata-kata alay menjadi kata-kata baku

Data Cleansing

Stopword

```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords as stopwords_scratch
list_stopwords=stopwords_scratch.words('indonesian')
list_stopwords.extend(['ya','yg','ga','yuk','dah'])
stopwords=list_stopwords
```

```
def remove_stopword(text):
    text = ' '.join(['' if word in stopwords else word for word in text.split(' ')])
    text = re.sub('  +', ' ', text)
    text = text.strip()
    return text
def stopword(text):
    text = remove_stopword(text)
    return text
df_train['text'] = df_train['text'].apply(stopword)
```

Untuk menghapus kata tertentu yang tidak memiliki makna

Model

Feature Extraction

```python
import pickle
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from collections import defaultdict

max_features=10000
tokenizer=Tokenizer(num_words=max_features,split=' ',lower=True)
tokenizer.fit_on_texts(total_data)
with open('tokenizer.pickle','wb') as handle:
    pickle.dump(tokenizer,handle,protocol=pickle.HIGHEST_PROTOCOL)
    print("Tokenizer.pickle has created!")

X=tokenizer.texts_to_sequences(total_data)
vocab_size=len(tokenizer.word_index)
maxlen=max(len(x) for x in X)

X=pad_sequences(X)
with open('x_pad_sequences.pickle','wb') as handle:
    pickle.dump(X,handle,protocol=pickle.HIGHEST_PROTOCOL)
    print("X_pad_sequnces.pickle has created!")
```

```
Tokenizer.pickle has created!
X_pad_sequnces.pickle has created!
```

Tokenizer

mengubah teks menjadi urutan bilangan bulat atau menjadi vektor di mana koefisien untuk tiap token bisa biner berdasarkan jumlah kata berdasarkan tf-idf.

Pad Sequences

mengubah list dari sequence jadi vektor/array bentuk 2D biar bisa diproses modelnya.

Model

Training (CNN)

```
embed_dim=64
model=Sequential()
model.add(Embedding(max_features,embed_dim,input_length=maxlen))
model.add(layers.Conv1D(128,3,activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(32,activation='relu'))
model.add(layers.Dense(3,activation='softmax'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
print(model.summary())
es=EarlyStopping(monitor='val_loss',mode='min',verbose=1,patience=1)
history=model.fit(X_train,y_train,epochs=10,batch_size=32,validation_data=(X_test,y_test),verbose=1,callbacks=[es])
```

Input Layer = 64
1D CNN Layer = 128 filter dengan ukuran 3 dan activation function relu
Dropout Layer = 0.5
Hidden Layer = 32 neuron dengan activation function relu
Output layer = 3 dengan activation function softmax
Loss function = Binary crossentropy
Optimizer = adam
Batch size = 32
Epoch = 10

Model

Training (LSTM)

```
embed_dim = 100
units  = 64

model = Sequential()
model.add(Embedding(max_features, embed_dim, input_length=X.shape[1]))
model.add(LSTM(units, dropout=0.5))
model.add(Dense(3,activation='softmax'))
model.compile(loss = 'binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())

adam = optimizers.Adam(lr = 0.001)
model.compile(loss = 'binary_crossentropy', optimizer = adam, metrics = ['accuracy'])

es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=1)
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test), verbose=1, callbacks=[es])
```

Input Layer = 100
Units= 64
Dropout Layer = 0.5
Output layer = 3 dengan activation function softmax
Loss function = Binary crossentropy
Optimizer = adam

Batch size = 32
Epoch = 10

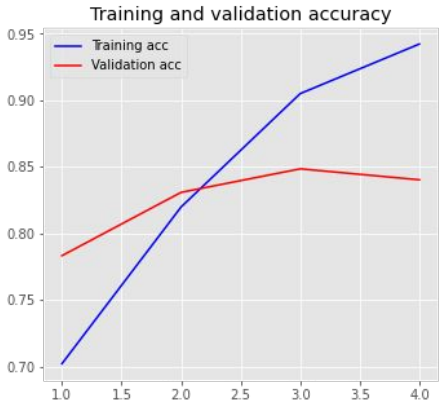
Model

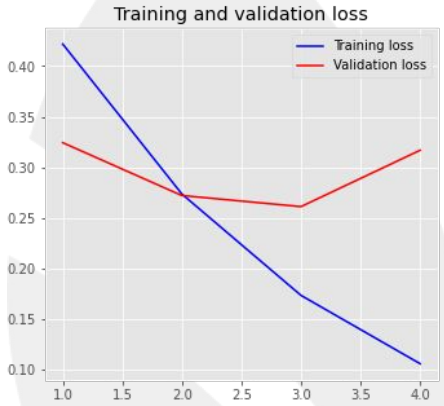
Evaluasi (CNN)

```
103/103 [==============================] - 0s 3ms/step
Testing selesai
              precision    recall  f1-score   support

           0       0.81      0.73      0.77      1017
           1       0.75      0.62      0.68       334
           2       0.86      0.94      0.90      1929

    accuracy                           0.84      3280
   macro avg       0.81      0.76      0.78      3280
weighted avg       0.84      0.84      0.84      3280
```





Akurasi

Rata-rata akurasi model CNN adalah 84% dimana untuk data sentimen positif mengalami overfitting pada angka 90% karena 60% data training memiliki sentimen positif sedangkan data sentiment netral underfitting pada angka 68% karena 10% data training memiliki sentimen netral

Kemudian dilihat dari grafik sebelah kanan dapat dilihat model masih belum good fit karena grafik loss belum stagnan

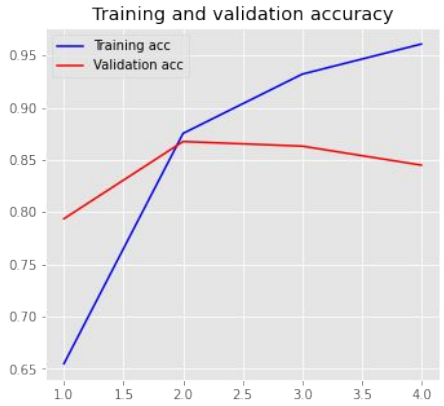
Model

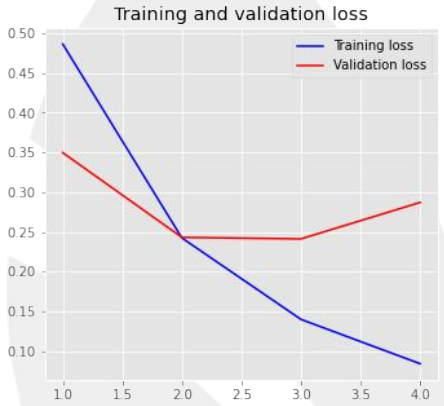
Evaluasi (LSTM)

```
57/57 [==============================] - 1s 16ms/step
Testing selesai
              precision    recall  f1-score   support

           0       0.90      0.90      0.90      1027
           1       0.83      0.84      0.83       345
           2       0.85      0.83      0.84       441

    accuracy                           0.87      1813
   macro avg       0.86      0.86      0.86      1813
weighted avg       0.87      0.87      0.87      1813
```





Akurasi

Rata-rata akurasi model LSTM adalah 87% dimana untuk data sentimen positif mengalami overfitting pada angka 90%. Kemudian dilihat dari grafik sebelah kanan dapat dilihat validation loss berada pada angka 24% dimana menunjukkan model masih cenderung overfitting.

Model

Predict (CNN)

```python
from keras.models import load_model
import re
sentiment=['negative','neutral','positive']
input_text='''bangsat'''
def regex(sent):
  text = sent.lower() #Lowercase text
  text = re.sub('USER', '', text) #Remove USER
  text = re.sub('RT', '', text) #Remove RT
  text = re.sub('URL', '', text) #Remove URL
  text = re.sub(r'\\n+', '', text) #Remove \n
  text = re.sub(r'https\S+','', text) #Remove https
  text = re.sub(r'\\x[A-Za-z0-9./]+', '', text) #Remove \x96 etc
  text = re.sub('#[A-Za-z0-9./]+', '', text) #Remove hastag
  text = re.sub('  +', '', text) #Remove extra space
  text = re.sub('[^a-zA-Z]+', ' ', text) #Remove non alpha numeric
  return text
text=[regex(input_text)]
predicted=tokenizer.texts_to_sequences([text])
guess=pad_sequences(predicted,maxlen=X.shape[1])
model=load_model('model.h5')
prediction=model.predict(guess)
polarity=np.argmax(prediction)
print("Text: ",text[0])
print("Sentiment: ",sentiment[polarity])
```

```
1/1 [==============================] - 0s 281ms/step
Text:  bangsat
Sentiment:  negative
```

```python
from keras.models import load_model
import re
sentiment=['negative','neutral','positive']
input_text='''syukur'''
def regex(sent):
  text = sent.lower() #Lowercase text
  text = re.sub('USER', '', text) #Remove USER
  text = re.sub('RT', '', text) #Remove RT
  text = re.sub('URL', '', text) #Remove URL
  text = re.sub(r'\\n+', '', text) #Remove \n
  text = re.sub(r'https\S+','', text) #Remove https
  text = re.sub(r'\\x[A-Za-z0-9./]+', '', text) #Remove \x96 etc
  text = re.sub('#[A-Za-z0-9./]+', '', text) #Remove hastag
  text = re.sub('  +', '', text) #Remove extra space
  text = re.sub('[^a-zA-Z]+', ' ', text) #Remove non alpha numeric
  return text
text=[regex(input_text)]
predicted=tokenizer.texts_to_sequences([text])
guess=pad_sequences(predicted,maxlen=X.shape[1])
model=load_model('model.h5')
prediction=model.predict(guess)
polarity=np.argmax(prediction)
print("Text: ",text[0])
print("Sentiment: ",sentiment[polarity])
```

```
1/1 [==============================] - 0s 164ms/step
Text:  syukur
Sentiment:  neutral
```

```python
from keras.models import load_model
import re
sentiment=['negative','neutral','positive']
input_text='''enak'''
def regex(sent):
  text = sent.lower() #Lowercase text
  text = re.sub('USER', '', text) #Remove USER
  text = re.sub('RT', '', text) #Remove RT
  text = re.sub('URL', '', text) #Remove URL
  text = re.sub(r'\\n+', '', text) #Remove \n
  text = re.sub(r'https\S+','', text) #Remove https
  text = re.sub(r'\\x[A-Za-z0-9./]+', '', text) #Remove \x96 etc
  text = re.sub('#[A-Za-z0-9./]+', '', text) #Remove hastag
  text = re.sub('  +', '', text) #Remove extra space
  text = re.sub('[^a-zA-Z]+', ' ', text) #Remove non alpha numeric
  return text
text=[regex(input_text)]
predicted=tokenizer.texts_to_sequences([text])
guess=pad_sequences(predicted,maxlen=X.shape[1])
model=load_model('model.h5')
prediction=model.predict(guess)
polarity=np.argmax(prediction)
print("Text: ",text[0])
print("Sentiment: ",sentiment[polarity])
```

```
1/1 [==============================] - 0s 204ms/step
Text:  enak
Sentiment:  positive
```

Model

Predict (LSTM)

```python
import re
from keras.models import load_model

input_text = """
Bangsat!.
"""

def cleansing(sent):
  string = sent.lower()

  string = re.sub(r'[^a-zA-Z0-9]',' ', string)
  return string

sentiment = ['negative', 'neutral', 'positive']

text = [cleansing(input_text)]
predicted = tokenizer.texts_to_sequences(text)
guess = pad_sequences(predicted, maxlen=X.shape[1])

model = load_model('model.h5')
prediction = model.predict(guess)
polarity = np.argmax(predictions[0])

print("Text: ", text[0])
print("Sentiment: ", sentiment[polarity])
```

```
1/1 [==============================] - 0s 464ms/step
Text:   bangsat
Sentiment:  positive
```

```python
import re
from keras.models import load_model

input_text = """
Baik..
"""

def cleansing(sent):
  string = sent.lower()

  string = re.sub(r'[^a-zA-Z0-9]',' ', string)
  return string

sentiment = ['negative', 'neutral', 'positive']

text = [cleansing(input_text)]
predicted = tokenizer.texts_to_sequences(text)
guess = pad_sequences(predicted, maxlen=X.shape[1])

model = load_model('model.h5')
prediction = model.predict(guess)
polarity = np.argmax(predictions[0])

print("Text: ", text[0])
print("Sentiment: ", sentiment[polarity])
```

```
1/1 [==============================] - 0s 424ms/step
Text:   baik
Sentiment:  positive
```

```python
import re
from keras.models import load_model

input_text = """
Cantik
"""

def cleansing(sent):
  string = sent.lower()

  string = re.sub(r'[^a-zA-Z0-9]',' ', string)
  return string

sentiment = ['negative', 'neutral', 'positive']

text = [cleansing(input_text)]
predicted = tokenizer.texts_to_sequences(text)
guess = pad_sequences(predicted, maxlen=X.shape[1])

model = load_model('model.h5')
prediction = model.predict(guess)
polarity = np.argmax(predictions[0])

print("Text: ", text[0])
print("Sentiment: ", sentiment[polarity])
```

```
1/1 [==============================] - 0s 457ms/step
Text:   cantik
Sentiment:  positive
```

Model Evaluation

BINAR

CNN

```
Rata-rata Accuracy: 0.8462195121951218
```

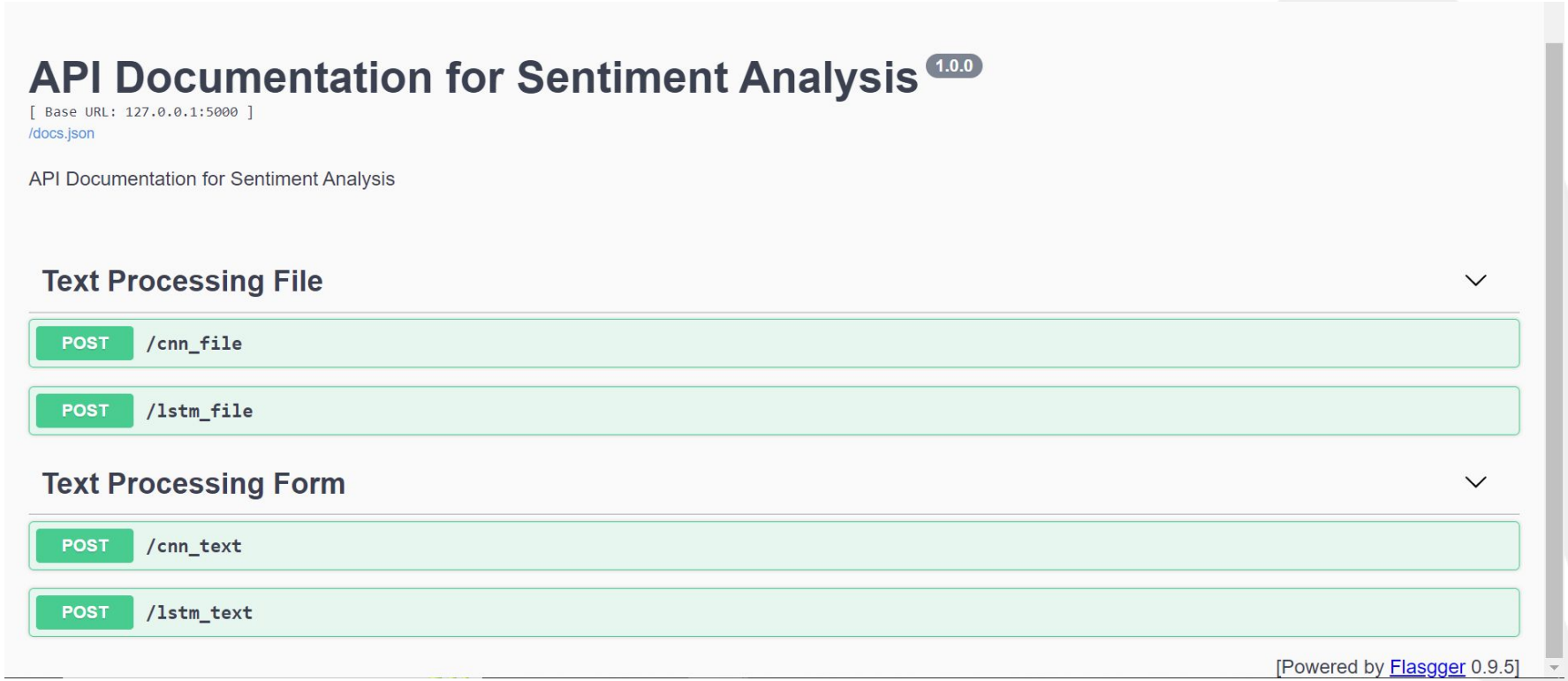
LSTM

```
Rata-rata Accuracy:  0.8595697738554883
```

Hasil & Kesimpulan

Hasil

API


API Documentation for Sentiment Analysis
1.0.0
[Base URL: 127.0.0.1:5000]
/docs.json
API Documentation for Sentiment Analysis
Text Processing File
POST /cnn_file
POST /lstm_file
Text Processing Form
POST /cnn_text
POST /lstm_text
[Powered by Flasgger 0.9.5]

Hasil

Hasil (LSTM)


```
[
  {
    "sentiment": "negative",
    "text_clean": "disaat cowok berusaha melacak perhatian loe lantas remehkan perhatian kasih khusus elo basic elo cowok bego"
  },
  {
    "sentiment": "negative",
    "text_clean": "telat ngasih tau elu edan sarap bergaul cigax jifla calis noh licew"
  },
  {
    "sentiment": "negative",
    "text_clean": "kadang berfikir percaya tuhan jatuh berkali kali kadang tuhan ninggalkan orangtuaku berencana berpisah kakakku memilih kristen anak ter"
  },
  {
    "sentiment": "negative",
    "text_clean": "akuku tau matamu sipit diliat"
  },
  {
    "sentiment": "negative",
    "text_clean": "kaum cebong kapir udah keliatan dongoknya dongok hahahah"
  },
  {
    "sentiment": "negative",
    "text_clean": "bani taplak dkk"
  },
  {
    "sentiment": "negative",
    "text_clean": "deklarasi pilkada aman anti hoax warga dukuh sari jabon"
  },
```

Hasil

Hasil (CNN)


```
{
  "sentiment": "neutral",
  "text_clean": "disaat cowok berusaha melacak perhatian loe lantas remehkan perhatian kasih khusus elo basic elo cowok bego"
},
{
  "sentiment": "neutral",
  "text_clean": "telat ngasih tau elu edan sarap bergaul cigax jifla calis noh licew"
},
{
  "sentiment": "neutral",
  "text_clean": "kadang berfikir percaya tuhan jatuh berkali kali kadang tuhan ninggalkan orangtuaku berencana berpisah kakakku memilih kristen anak ter"
},
{
  "sentiment": "neutral",
  "text_clean": "akuku tau matamu sipit diliat"
},
{
  "sentiment": "neutral",
  "text_clean": "kaum cebong kapir udah keliatan dongoknya dongok hahahah"
},
{
  "sentiment": "neutral",
  "text_clean": "bani taplak dkk"
```


API

Dari hasil sentiment analysis diatas semua tweet mendapat label neutral seharusnya mendapat label negative. Alasannya dapat dilihat pada slide selanjutnya

Word Cloud

Data Train

Positive

Neutral

Negative

Word Cloud Data Train

Dari word cloud diatas dapat disimpulkan bahwa text label positive dan negative memiliki topik makanan sedangkan untuk sentiment neutral memiliki topik politik. Data tweet yang dijadikan sebagai data testing memiliki topik politik sehingga karena data train sentiment neutral dan data test memiliki sentiment neutral maka pada saat predict sentiment analysis memiliki sentiment yang sama

Summary

Kesimpulan

- Data training yang digunakan kurang tepat karena untuk text sentiment positive dan negative memiliki topik berbeda dengan data test yaitu politik sehingga sentiment yang dihasilkan kurang tepat
- Model LSTM memiliki akurasi lebih baik daripada model CNN karena LSTM memiliki sel memori
- Dikarenakan data training yang kurang tepat maka sulit untuk menentukan tingkat kesehatan bermedia sosial Twitter dari hasil sentiment yang ada
- Untuk sentimen negatif pada data train sebagian besar memiliki topik makanan

Saran

- Untuk peneliti selanjutnya yang ingin melakukan penelitian serupa kami menyarankan untuk menambah data training atau mengubah data training yang memiliki tipikal yang sama dengan data testing atau data hate speech tweet