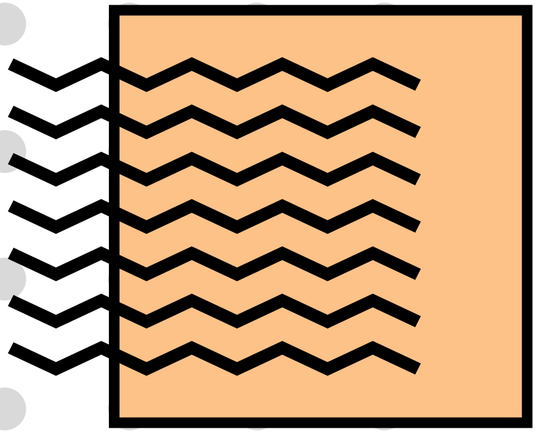
BINAR ACADEMY - WAVE 9
KELOMPOK 4



SENTIMENT ANALYSIS

BY :
AHMAD FADLAN AMIN
SUSILAWATY

TABLE OF CONTENT

1. LATAR BELAKANG

- Analisis sentimen, atau disebut juga opinion mining, adalah teknik Natural Language Processing (NLP) untuk menganalisis emosi dari suatu teks.
- Analisis sentimen dapat dilakukan dengan mengidentifikasi kata dan frasa positif, negatif, atau netral, serta dengan mencari pola penggunaan kata-kata tersebut.
- Analisis sentimen kini telah menjadi lebih populer pada bidang opinion mining (penambangan opini) pengguna terhadap produk, ulasan politik, ulasan film dll. Produser, produsen, pembuat film, dan politisi dapat mengetahui pandangan dan pemikiran konsumen, penonton dengan menganalisis ulasan mereka melalui banyak situs online seperti Facebook, Twitter, IMDb dll
- Pada umumnya terdapat dua jenis pendekatan dalam analisis sentimen yaitu supervised, dan unsupervised learning. Supervised learning dalam analisis sentimen melatih data training untuk mengklasifikasikan teks, sementara Unsupervised leaning tanpa data (x) training alias belajar dengan mempelajari pola x.
- Dalam kasus ini, kita menggunkan pendekatan Supervised learning untuk meneliti data set yang akan kita olah. Sebelum teks di klasifikasikan berdasarkan sentimen, data teks terlebih dahulu dilakukan pembobotan dalam bentuk numerik agar bisa di proses lebih lanjut menggunakan machine learning. Proses ini disebut dengan feature extraction yang mana ada dua Teknik fearure extaction yaitu BOW or TF-IDF. Feature selection adalah tahapan dalam pemrosesan data yang dapat mempengaruhi tingkat akurasi untuk meningkatkan bidang klasifikasi analisis sentiment. Langkah selanjutnya di perlukan memisahkan data menjadi data latih dan data uji mengunakan metode neural network dengan library Sklearn dan LSTM dengan library Tensorflow.
- Pada penelitian ini untuk mengekstrak data kalimat sentimen akan lebih mudah untuk di klasifikasi agar lebih akurat.

2. TUJUAN PENELITIAN & RUMUSAN MASALAH

Tujuan Penelitian

- Tujuan dari penelitian ini adalah untuk mengevaluasi, percobaan hasil model training, testing dan melakukan visualisasi sebuah data sentimen yang dapat menghasilkan output berupa sentimen yang dapat di kategorikan sebagai sentimen positive, negative dan netral.
- Mengetahui tingkat accuracy, Precision, Recall, F1-score and Support dari dua motode berbeda.
- Mengaplikasikan hasil prediksi model yang telah di uji dan di latih ke sebuah model deployment API flask dan swagger UI.

Rumusan Masalah

- Bagaimana analisis dan tingkat akurasi yang di hasilkan dan performa analisis sentimen pada model neural network?
- Bagaimana analisis dan tingkat akurasi yang di hasilkan dan performa analisis sentimen pada model long short term memory?
- Bagaimana hasil pengaplikasian cross validation terhadap model yang sudah di buat?

3. METODE PENELITIAN

Neural Network

Neural Network adalah algoritma yang bekerja dengan cara mengidentifikasi hubungan mendasar dalam sekumpulan data melalui proses meniru cara kerja otak manusia. Dalam data science, Neural Network melakukan pengelompokkan dan mengklasifikasikan hubungan yang sudah diidentifikasi tersebut. Neural Network dapat digunakan untuk membuat kelompok dari data yang tidak berlabel sesuai dengan kesamaan yang dimiliki.
Neural network terdiri dari tiga bagian yaitu:
1.Input layer : input sinyal atau data
2.Middle layer/ hidden layer :memproses data dari inputan tadi
3.Output layer :output dari data yang sudah diproses
salah satu kelebihan nya dapat menyimpan data di seluruh jaringan. Alih-alih disimpan di database, data yang digunakan akan disimpan di seluruh jaringan. Dengan begitu, proses kerja jaringan tidak akan terhambat bila terjadi data hilang.

Long short term memory

Long short term memory network (LSTM) adalah sistem penyimpanan data yang dapat memproses, memprediksi, dan mengklasifikasikan informasi yang telah disimpan dalam jangka waktu lama sekali pun.
LSTM juga punya arsitektur yang terdiri dari tiga gerbang yaitu:
1.Gerbang input (input gate)
2.Gerbang lupa (forget gate)
3.Gerbang keluarga (output gate) yang mengontrol aliran informasi
Ketiga gate tersebut fungsinya sama seperti input, hidden dan output layer, bedanya adalah dia punya gate. Hidden layer output dari LSTM termasuk hidden state dan memory cell namun cuma hidden state yang dilewatkan ke output layer. Memori dalam arsitektur LSTM ini sepenuhnya internal dimana mengarah ke output itu cuman hidden state.

Kedua metode analisis yang di pakai dalam penelitian ini menggunakan descriptive analytics. Jenis analisis deskriptif ini di rasa cocok karena befokus untuk mencari tahu kondisi data dan mempelajari pola suatu data.

4. DATA EXPLORATION

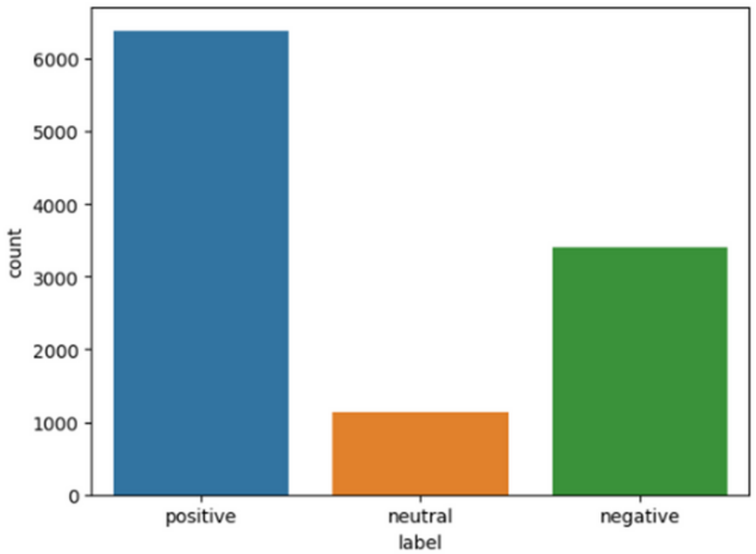
```
# Check Missing Value
df.isna().sum()
```
✓ 0.0s

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

```
label
positive    6416
negative    3436
neutral     1148
Name: count, dtype: int64
```

```
[5]: # Check data skewness accross label
sns.countplot(x='label', data=df)
```

```
[5]: <Axes: xlabel='label', ylabel='count'>
```



- Data terdiri dari 11000 text dan label
- Terdapat 3 sentiment : Positive, Negative dan Neutral.
- Tidak terdapat null value dalam data frame

Data Distribution :

- Positive : 6416 text (58%)
- Negative : 3436 (31%)
- Neutral : 1148 (10%)

5. DATA CLEANSING

text_clean
warung ini dimiliki oleh pengusaha pabrik tahu...
mohon ulama lurus dan k212 mmbri hujjah partai...
lokasi strategis di jalan sumatera bandung t...
betapa bahagia nya diri ini saat unboxing pake...
duh jadi mahasiswa jangan sombong dong kas...

Clean Text results.

Hasil clean text tersebut ditambahakn di kolom baru pada dataframe.

Membersihkan teks menggunakan Regex library.

Data Cleansing :

- Menghilangkan tanda baca
- Lower-case texts
- Menghilangkan whitespace sebelum & setelah kata
- Menghilangkan kata yg tidak perlu (www / user/ http etc)

```python
import re
def cleansing(text):
    text = text.lower()
    text = text.strip()
    text = re.sub(r'[^a-zA-Z]', ' ',text)
    text = re.sub(r'[^\w\s]', '',text)
    text = re.sub(r'\s+[a-zA-Z]\s+', ' ',text)
    text = re.sub(r'\s+', ' ',text)
    text = re.sub(r'rt @\w+:',' ',text)
    text = re.sub(r'((www\.[^\s]+)|(https?://[^\s]+)|(http?://[^\s]+))|([#@]\S+)|user|\n|\t', ' ', text)
    return text
```

6. FEATURE EXTRACTION AND TRAIN TEST SPLIT

Feature Extraction merubah data texts yang sudah di cleansing menjadi data vector dengan metode TF-IDF

Hasil feature extraction tersebut kemudian di simpan kedalam file dengan format pickle

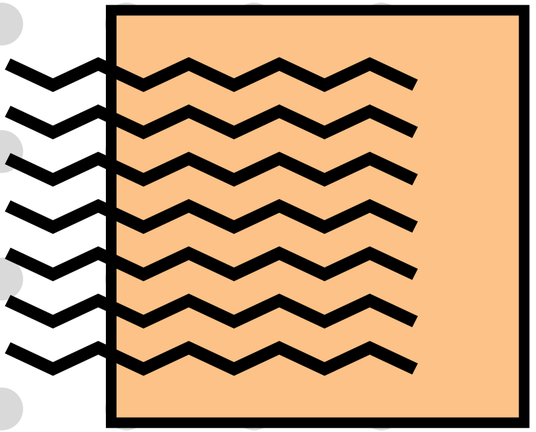
```
# feature extraction with metode TF-IDF (inverse Document Frequency)
from sklearn.feature_extraction.text import TfidfVectorizer

total_data = df['text_clean'].tolist()

# Proses Feature Extraction
tfidf_vect = TfidfVectorizer()
tfidf_vect.fit(total_data)

X = tfidf_vect.fit_transform(total_data)
print("Feature Extraction selesai")
```

```
# import and save to pickle
import pickle

with open('tfidf_vect.pkl', 'wb') as f:
    pickle.dump(tfidf_vect, f)
```

```
[14]:
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)

print(f"X_Train size: {X_train.shape[0]}")
print(f"y_train size: {y_train.shape[0]}")
print(f"X_test size: {X_test.shape[0]}")
print(f"y_test size: {y_test.shape[0]}")

X_Train size: 8800
y_train size: 8800
X_test size: 2200
y_test size: 2200

[15]: pd.Series(y_train).value_counts()

[15]: label
positive    5135
negative    2756
neutral      909
Name: count, dtype: int64

[16]: pd.Series(y_test).value_counts()

[16]: label
positive    1281
negative     680
neutral      239
Name: count, dtype: int64
```

Data yang sudah melalui feature extraction, kemudian dipisah untuk dilakukan training dan testing, dengan rasio 80% untuk training, dan 20% untuk testing.

6. FEATURE EXTRACTION AND TRAIN TEST SPLIT



07 & 08

MODEL TRAINING EVALUATION

NEURAL NETWORK

```
# Initialize model

model = MLPClassifier(hidden_layer_sizes=(8,8,8), activation='relu', solver='adam', max_iter=300, random_state=1)

# Training

model.fit(X_train, y_train)
print("Training selesai")
```

```
Testing selesai
              precision    recall  f1-score   support

    negative       0.79      0.79      0.79       680
     neutral       0.88      0.68      0.76       239
    positive       0.87      0.91      0.89      1281

    accuracy                           0.85      2200
   macro avg       0.85      0.79      0.81      2200
weighted avg       0.85      0.85      0.85      2200
```

•Hasil model evaluasi Neural Network dari modul yang di pakai Sklearn menunjukan F-1 Score punya nilai cukup bagus di bandingkan 3 metode lainnya. Ini karena nilai F-1 pada masing-masing klasifikasi mulai dari 0.79 negative, 0.76 netral dan 0.89 positif.

CROSS VALIDATION : NEURAL NETWORK

```
Training ke- 1
              precision    recall  f1-score   support

    negative       0.77      0.78      0.77       680
     neutral       0.76      0.64      0.69       239
    positive       0.87      0.90      0.89      1281

    accuracy                           0.83      2200
   macro avg       0.80      0.77      0.78      2200
weighted avg       0.83      0.83      0.83      2200

====================================================
Training ke- 2
              precision    recall  f1-score   support

    negative       0.79      0.76      0.78       706
     neutral       0.73      0.70      0.71       220
    positive       0.88      0.90      0.89      1274

    accuracy                           0.83      2200
   macro avg       0.80      0.79      0.79      2200
weighted avg       0.83      0.83      0.83      2200

====================================================
Training ke- 3
              precision    recall  f1-score   support

    negative       0.80      0.80      0.80       682
     neutral       0.85      0.71      0.77       215
    positive       0.89      0.91      0.90      1303

    accuracy                           0.86      2200
   macro avg       0.85      0.81      0.82      2200
weighted avg       0.86      0.86      0.86      2200

====================================================
```

- Hasil cross validation dengan MLPClassifier Neural Network menunjukan rata-rata hasil training accuracy 0.84

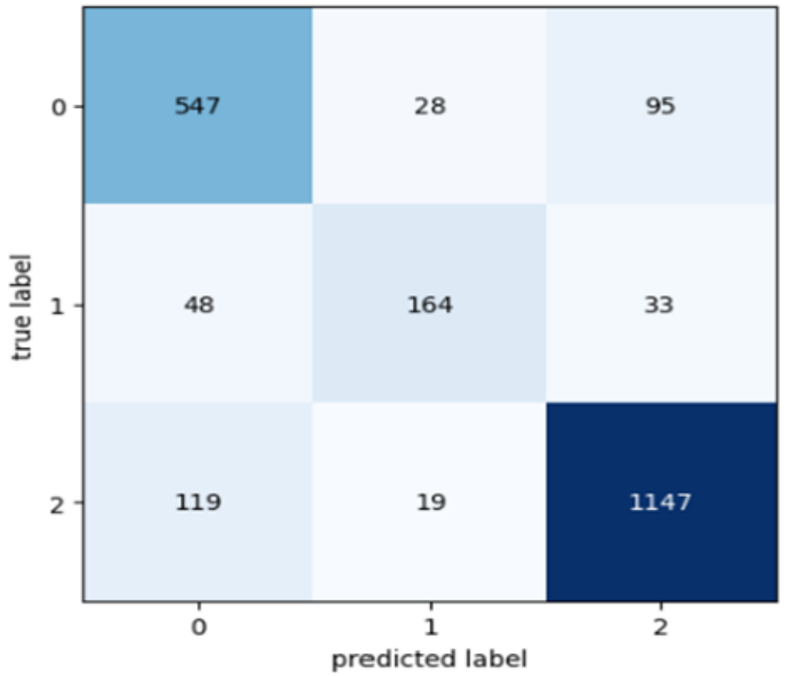
```
====================================================
Training ke- 4
              precision    recall  f1-score   support

    negative       0.78      0.80      0.79       698
     neutral       0.80      0.63      0.71       229
    positive       0.88      0.90      0.89      1273

    accuracy                           0.84      2200
   macro avg       0.82      0.78      0.79      2200
weighted avg       0.84      0.84      0.84      2200

====================================================
Training ke- 5
              precision    recall  f1-score   support

    negative       0.76      0.81      0.79       670
     neutral       0.79      0.65      0.71       245
    positive       0.89      0.89      0.89      1285

    accuracy                           0.84      2200
   macro avg       0.81      0.79      0.80      2200
weighted avg       0.84      0.84      0.84      2200

====================================================


Rata-rata Accuracy:  0.8408181818181817
```

CONFUSION MATRIX : NEURAL NETWORK

•Hasil training model Confusion matrix pada model neural network diperoleh dari hasil test dengan menggunakan ytest dalam train test split dengan jumlah data sebanyak 2200 data

predict positif vs true label positif = 1147
predict neutral vs true label netral = 164
predict negative vs true label negative = 547

```
[25]: import matplotlib.pyplot as plt
      from mlxtend.plotting import plot_confusion_matrix
      from sklearn.metrics import confusion_matrix

      conf_mat = confusion_matrix(target_test,preds)
      plot_confusion_matrix(conf_mat)

      plt.show()
```



HASIL PREDIKSI : NEURAL NETWORK

•Hasil model prediksi dari model yang sudah kita latih

[20]:
```
# model prediksi
original_text = '''
hormati partai-partai yang telah berkoalisi
'''

text = count_vect.transform([cleansing(original_text)])

result = model.predict(text)[0]
print("Sentiment:")
print()
print(result)
```

```
Sentiment:

neutral
```

LSTM

```
# Split the data to 80% for training, 20% for testing
from sklearn.model_selection import train_test_split
file = open('x_pad_sequences.pickle' , 'rb')
X = pickle.load(file)
file.close()

file = open('y_labels.pickle', 'rb')
Y = pickle.load(file)
file.close()

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=1)
```

LSTM yaitu network yang punya memori dengan bentuk yang sama dengan hidden state untuk merekam informasi tambahan.

Data yang telah di cleansing, dilakukan feature extraction untuk merubah text menjadi vector. (tokenizer dan pad sequences)

```
embed_dim = 100
units = 64

model = Sequential()
model.add(Embedding(max_features, embed_dim, input_length=X.shape[1]))
model.add(LSTM(units, dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics = ['accuracy'])
print(model.summary())

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

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

```
predictions = model.predict(x_test)
y_pred = predictions
matrix_test = metrics.classification_report(y_test.argmax(axis=1), y_pred.argmax(axis=1))
print('Testing selesai')
print(matrix_test)
```

```
69/69 [==============================] - 2s 18ms/step
Testing selesai
              precision    recall  f1-score   support

           0       0.80      0.86      0.83       685
           1       0.88      0.73      0.80       233
           2       0.91      0.90      0.90      1282

    accuracy                           0.87      2200
   macro avg       0.86      0.83      0.84      2200
weighted avg       0.87      0.87      0.87      2200
```

LSTM RESULTS

Hasil testing dengan model LSTM menunjukkan f-1 Score untuk sentiment

Negative : 0.83

Neutral : 0.80

Positif : 0.90

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 91, 100)           10000000

 lstm (LSTM)                 (None, 64)                42240

 dense (Dense)               (None, 3)                 195

=================================================================
Total params: 10042435 (38.31 MB)
Trainable params: 10042435 (38.31 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
Epoch 1/10
880/880 [==============================] - 116s 131ms/step - loss: 0.4630 - accuracy: 0.8195 - val_loss: 0.3761 - val_accuracy: 0.8495
Epoch 2/10
880/880 [==============================] - 114s 130ms/step - loss: 0.1998 - accuracy: 0.9277 - val_loss: 0.3541 - val_accuracy: 0.8686
Epoch 3/10
880/880 [==============================] - 136s 155ms/step - loss: 0.1083 - accuracy: 0.9591 - val_loss: 0.4290 - val_accuracy: 0.8718
Epoch 3: early stopping
```

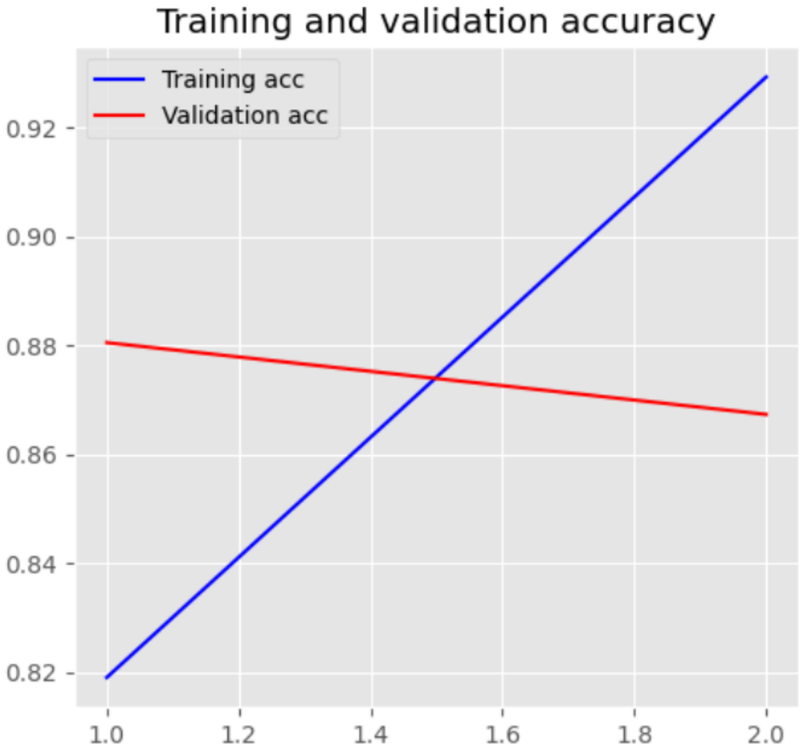
LSTM CROSS VALIDATION

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_5 (Embedding)     (None, 91, 100)           10000000

 lstm_5 (LSTM)               (None, 64)                42240

 dense_5 (Dense)             (None, 3)                 195

=================================================================
Total params: 10042435 (38.31 MB)
Trainable params: 10042435 (38.31 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
Epoch 1/10
880/880 [==============================] - 167s 185ms/step - loss: 0.4550 - accuracy: 0.8190 - val_loss: 0.3311 - val_accuracy: 0.8805
Epoch 2/10
880/880 [==============================] - 165s 187ms/step - loss: 0.1914 - accuracy: 0.9292 - val_loss: 0.3548 - val_accuracy: 0.8673
Epoch 2: early stopping
69/69 [==============================] - 3s 33ms/step
```

```
Training ke- 5
              precision    recall  f1-score   support

           0       0.85      0.78      0.82       685
           1       0.85      0.73      0.79       233
           2       0.88      0.94      0.91      1282

    accuracy                           0.87      2200
   macro avg       0.86      0.82      0.84      2200
weighted avg       0.87      0.87      0.87      2200

========================================


Rata-rata Accuracy:  0.8704545454545455
```

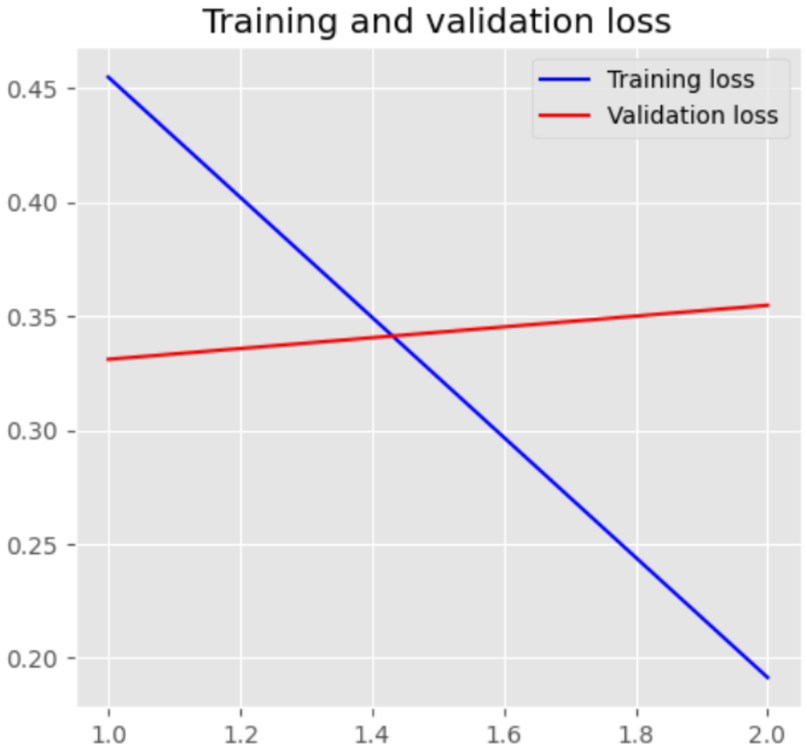
Setelah dilakukan cross validation, yaitu merotasi porsi training dari dataset agar bisa melihat model yang sudah kita buat untuk menguji apakah model tersebut “stabil” atau tidak ketika diberikan dataset yang berbeda. Terdapat hasil accuracy : 0.87

VISUALISASI LSTM



accuracy = train acc 0.93, val acc 0.86

Epoch = 2 early stopping

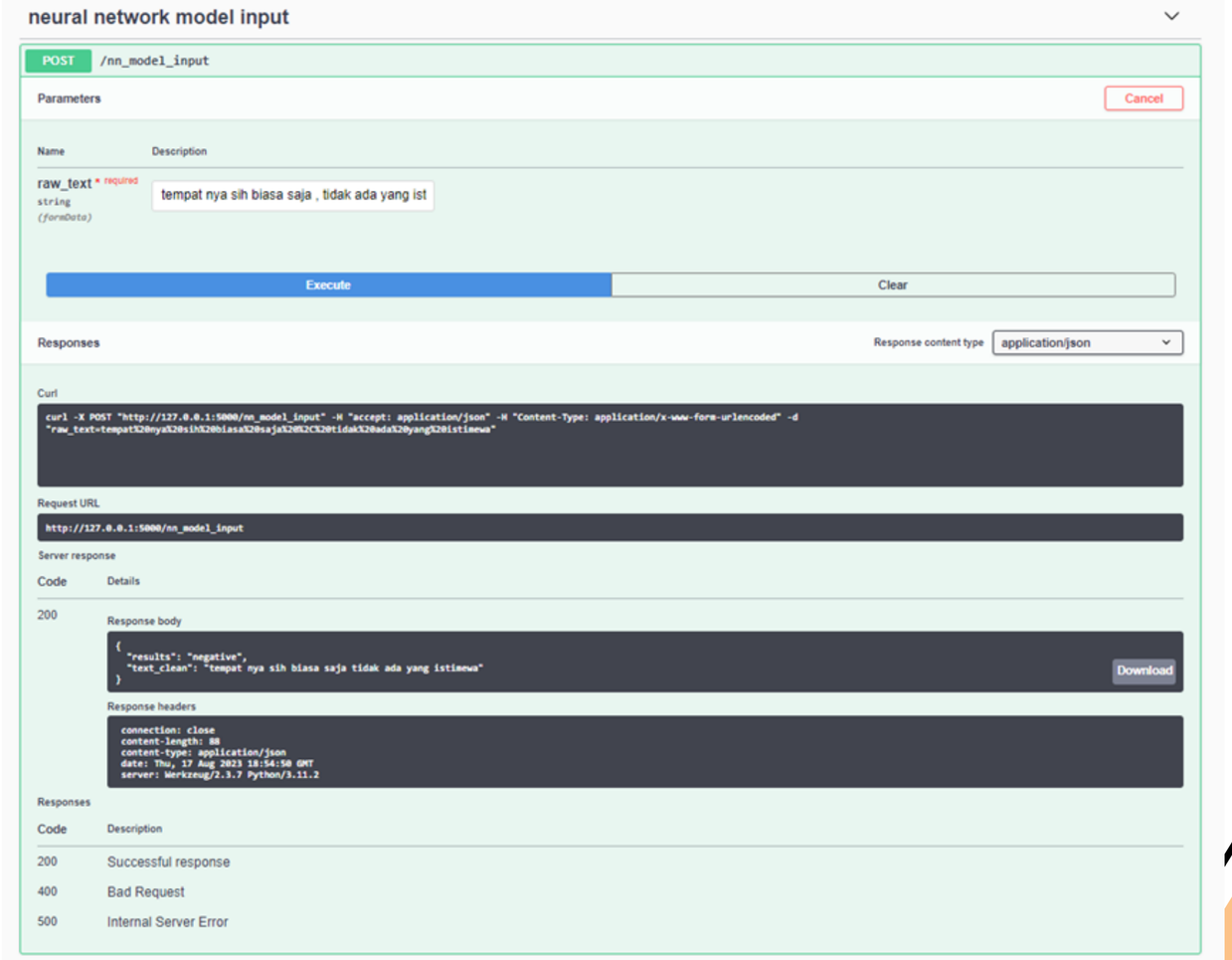


loss = Train loss 0.19, val loss 0.35

Epoch = 2 early stopping

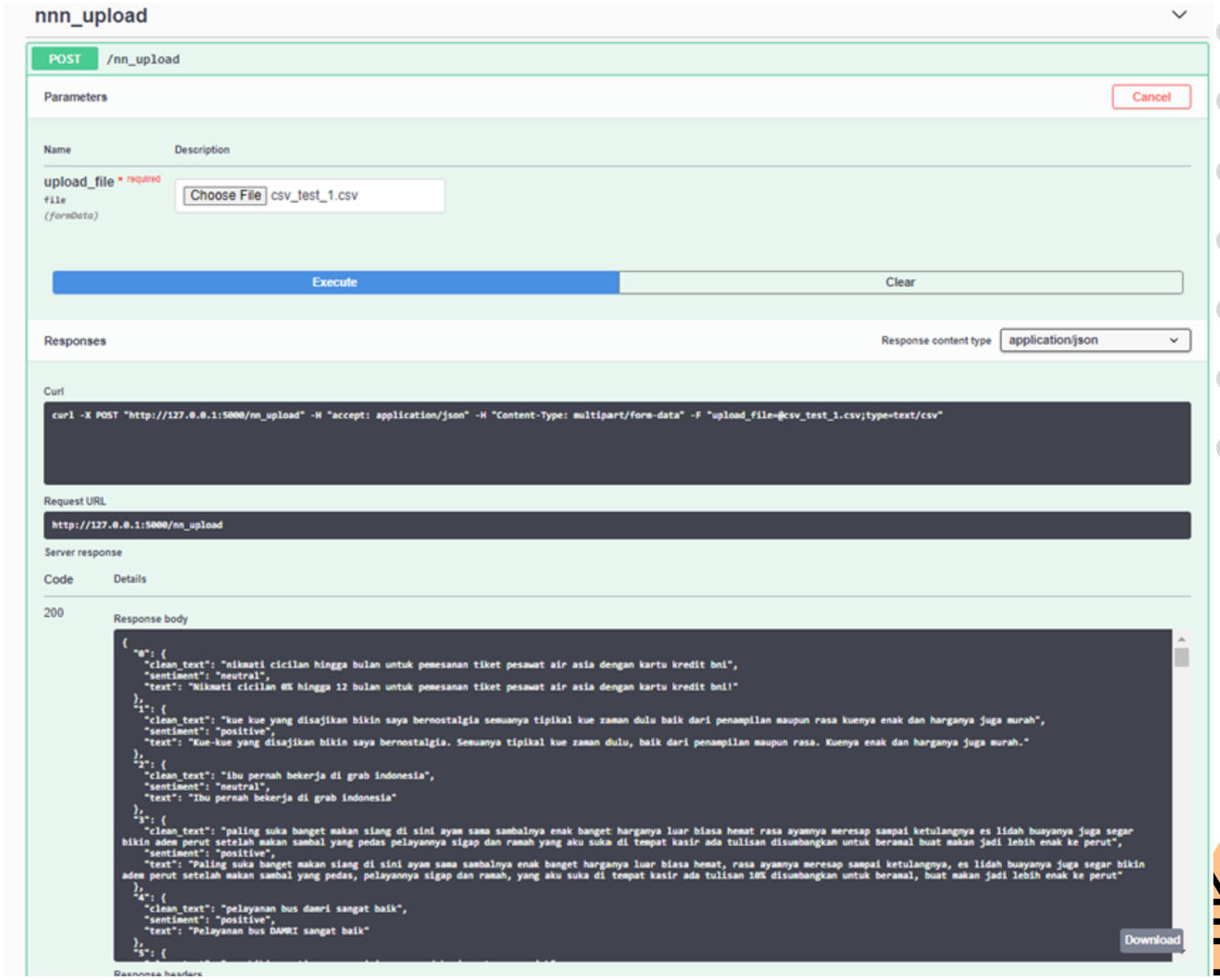
09. API RESULTS

NEURAL NETWORK - TEXT INPUT



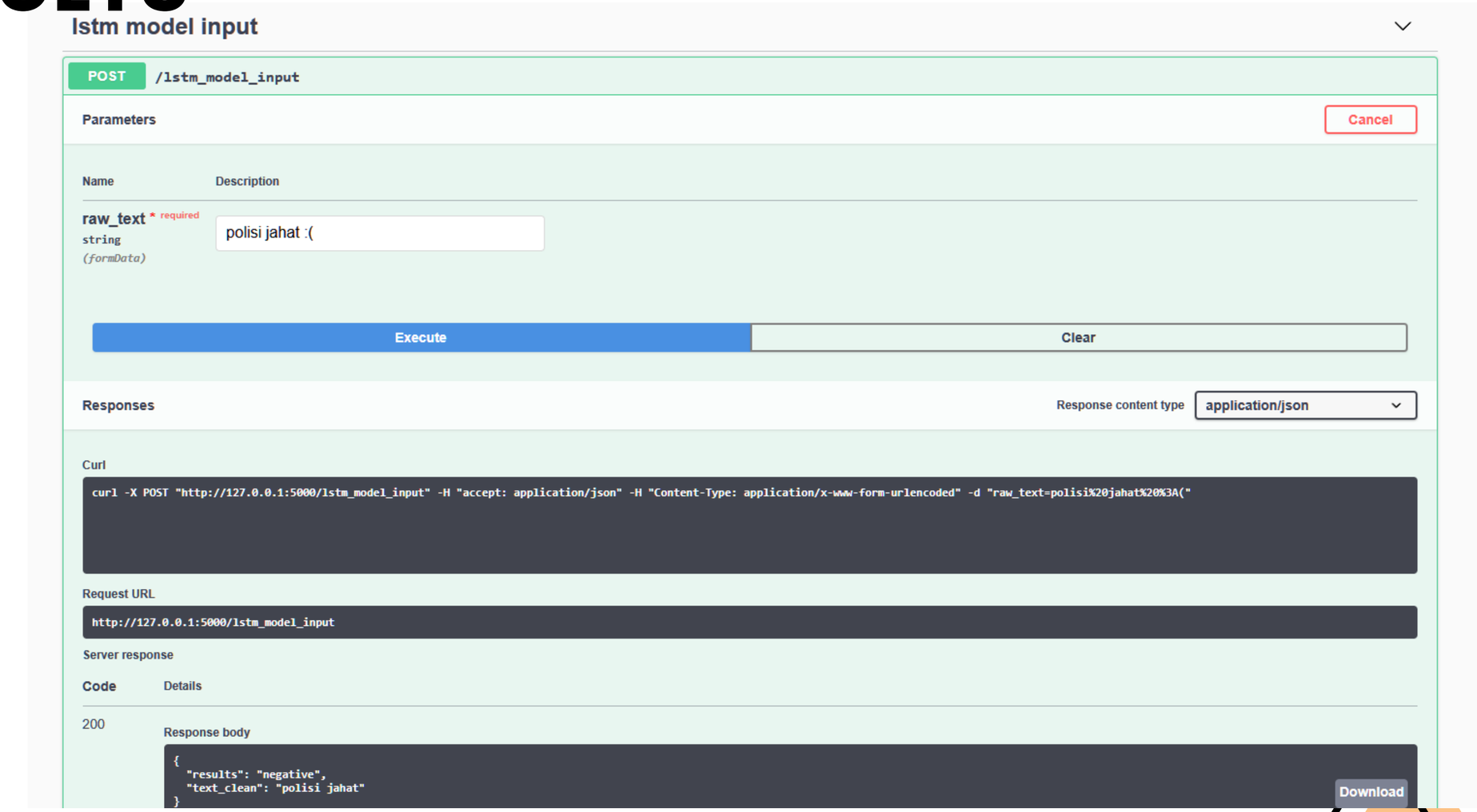
09. API RESULTS

NEURAL NETWORK – FILE UPLOAD



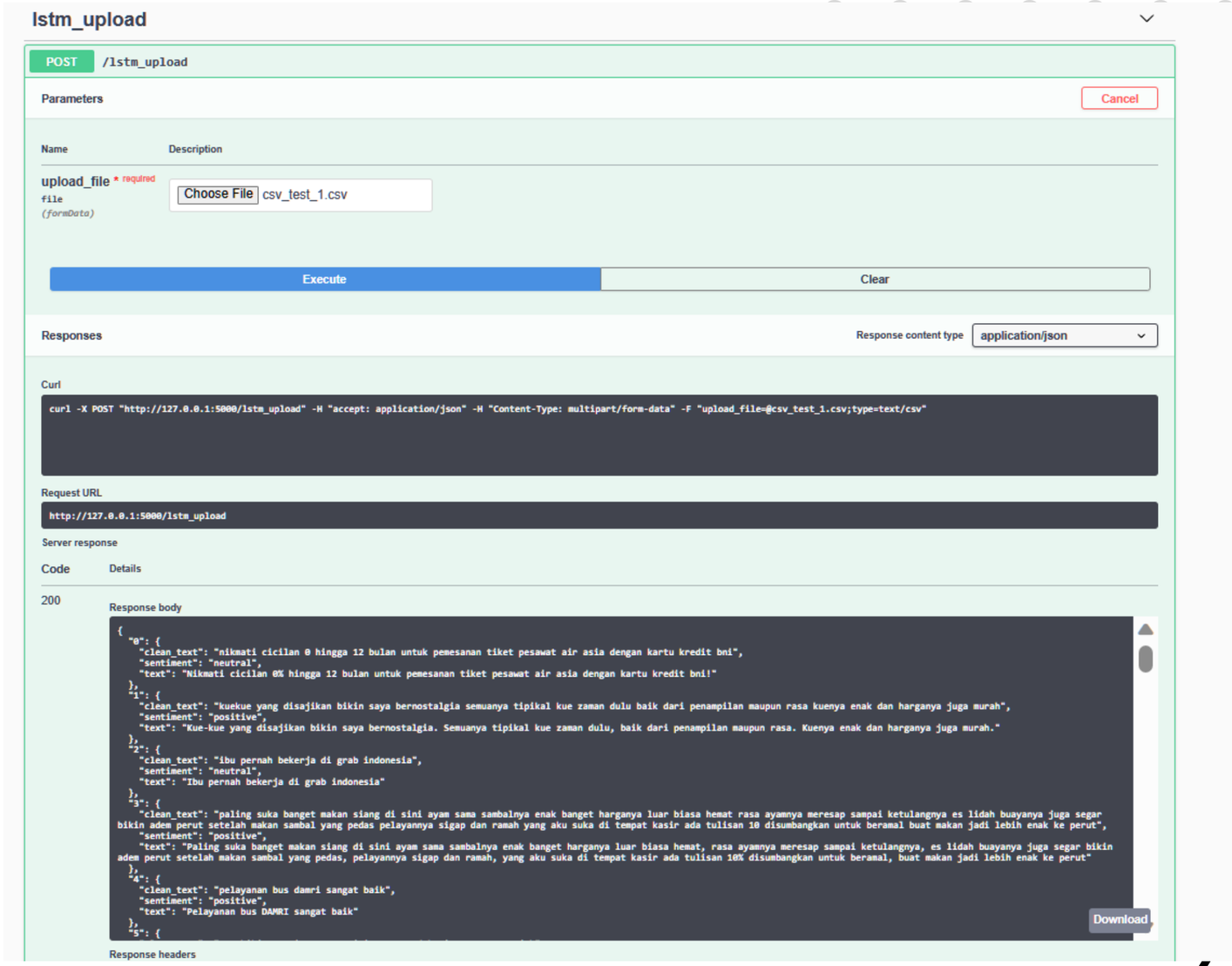
09. API RESULTS

LSTM – TEXT INPUT



09. API RESULTS

LSTM – FILE UPLOAD

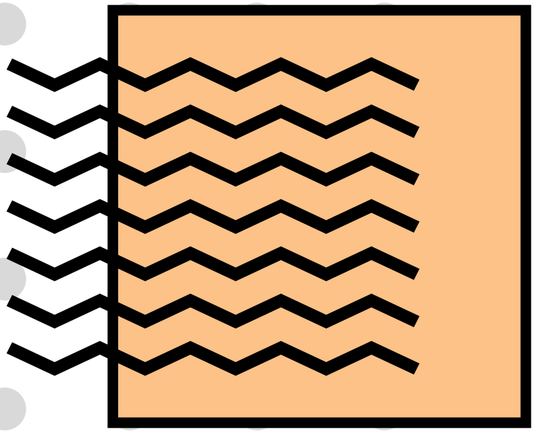


10. SUMMARY

HASIL ANALISIS SENTIMEN ATAS KEDUA MODEL TERGOLONG CUKUP BAIK NAMUM BELUM DAPAT DI KATEGORIKAN SEMPURNA DAN BELUM SIAP UNTUK DILAKUKAN DEPLOYMENT KEPADA USER. ADA BEBERAPA TEXT YANG TERGOLONG SALAH DI PREDIKSI OLEH MODEL SEHINGGA PERLU DI LAKUKAN PENELITIAN LEBIH LANJUT UNTUK MENINGKATKAN PERFORMA DAN KEAKURATAN MODEL.

11. SARAN

- PENAMBAHAN PROSES CLEANSING DATA UNTUK MENINGKATKAN EFEKTIFITAS TRAINING MODEL
- PENAMBAHAN DATA TRAINING YANG LEBIH BERVARIASI UNTUK MENINGKATKAN PERFORMA LEARNING PADA MODEL
- PENGGUNAAN CALLBACKS YANG DIGUNAKAN SEBAGAI EARLYSTOPPING MENINGKATKAN EFEKTIFITAS TRAINING MODEL



THANK YOU

GITHUB CLONE REPO:
HTTPS://GITHUB.COM/FADLANAMIN/DSC2300944_AHMAD_SUSI_SENTIMEN_PLATINUM

SHARE GOOGLE DRIVE MODEL LSTM & NEURAL_NETWROK
HTTPS://DRIVE.GOOGLE.COM/DRIVE/FOLDERS/1Y7FQ8VZH_4LAWF3XTLUY69FB1JGHSEYO