

Komparasi Kinerja Algoritma *Support Vector Machine- Recursive Feature Elimination* (SVM-RFE) dan *Recurrent Neural Network - Long Short Term Memory* (RNN-LSTM) untuk Analisis Sentimen *Review* Mahasiswa

M. Burhanis Sulthan^{1*}, Tizanatun Nafisah², Luluk Suhartini³

^{1,2,3} Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Annuqayah

¹burhan.sulthan33@gmail.com, ²tizana02@gmail.com, ³lulukkhafi@gmail.com

*Penulis Korespondensi

ABSTRAK

Analisis sentimen terhadap ulasan mahasiswa berperan penting dalam meningkatkan kualitas layanan biro akademik. Penelitian ini membandingkan kinerja algoritma *Support Vector Machine* (SVM) dengan *Recursive Feature Elimination* (RFE) dan *Recurrent Neural Network* dengan *Long Short-Term Memory* (RNN-LSTM) dalam menganalisis sentimen ulasan mahasiswa terkait pelayanan Biro Akademik Universitas Annuqayah. Data dikumpulkan melalui *Google Form* dan diproses menggunakan tahapan *cleaning*, *case folding*, *stopword removal*, *tokenization*, dan *stemming*. *Dataset* yang dikumpulkan memiliki jumlah 72 data berlabel positif dan 31 data berlabel negatif. Hasil evaluasi menunjukkan bahwa SVM mencapai akurasi tertinggi sebesar 93,75%, diikuti oleh SVM-RFE (90,63%), RNN-LSTM (87,50%), dan RNN (81,25%). Temuan ini mengindikasikan bahwa SVM lebih unggul dalam mengklasifikasikan sentimen pada *dataset* ini, sementara SVM-RFE menunjukkan performa yang kompetitif dengan seleksi fitur yang tepat. Meskipun RNN dan RNN-LSTM dikenal efektif dalam menangani data sekuensial, pada penelitian ini, SVM dan SVM-RFE terbukti lebih optimal untuk analisis sentimen ulasan mahasiswa. Hasil penelitian ini memberikan wawasan bagi pemilihan algoritma analisis sentimen di lingkungan pendidikan tinggi. Pemilihan model yang tepat bergantung pada karakteristik data serta tujuan analisis yang diinginkan.

Kata kunci: Analisis sentimen, *Support Vector Machine*, *Recursive Feature Elimination*, *Recurrent Neural Network*, *Long Short-Term Memory*

ABSTRACT

Sentiment analysis of student feedback is crucial in enhancing the quality of academic office services. This research compares the performance of Support Vector Machine (SVM) with Recursive Feature Elimination (SVM-RFE) and Recurrent Neural Network with Long Short-Term Memory (RNN-LSTM) algorithms in analysing the sentiment of student reviews related to the services of the Academic Bureau of Annuqayah University. The data was collected through Google Form and processed through cleaning, case folding, stop word removal, tokenisation and stemming stages. The collected dataset has a total of 72 positively labeled data and 31 negatively labeled data. The evaluation results showed that SVM achieved the highest accuracy of 93.75%, followed by SVM-RFE (90.63%), RNN-LSTM (87.50%) and RNN (81.25%). This result indicates that SVM is superior in classifying sentiment on this dataset, while SVM-RFE shows competitive performance with proper feature selection. Although RNN and RNN-LSTM are known to be effective in handling sequential data, in this study SVM and SVM-RFE proved to be more optimal for sentiment analysis of student reviews. The results of this study provide insights for the selection of sentiment analysis algorithms in the higher education environment. The selection of an appropriate model depends on the characteristics of the data as well as the desired analytical objectives.

Keywords: *Sentiment analysis, Support Vector Machine, Recursive Feature Elimination, Recurrent Neural Network, Long Short-Term Memory*

1. PENDAHULUAN

Untuk meningkatkan kesejahteraan masyarakat dan kemajuan teknologi dan ilmu pengetahuan, Pendidikan tinggi memiliki peranan penting. Di Indonesia perguruan tinggi diklasifikasikan menjadi dua jenis, yakni Perguruan Tinggi Negeri (PTN) dan Perguruan Tinggi Swasta (PTS). Universitas Annuqayah adalah salah satu perguruan tinggi swasta di Jawa Timur yang berkomitmen untuk mencetak lulusan yang berakhlak mulia dan kompetitif. Layanan akademik berperan penting dalam menumbuhkan kepercayaan siswa dan menciptakan citra positif institusi. Oleh karena itu, sangat penting untuk menilai layanan yang ditawarkan oleh biro akademik untuk memastikan bahwa kebutuhan mahasiswa dipenuhi dengan baik. Analisis sentimen memiliki peran yang sangat penting dalam konteks pendidikan karena memungkinkan institusi untuk memahami pengalaman, kebutuhan, dan kepuasan mahasiswa secara mendalam. Meningkatkan kualitas pelayanan akademik dengan memahami sentimen

mahasiswa terhadap berbagai aspek layanan akademik, seperti administrasi, bimbingan, atau fasilitas institusi. sehingga dapat mengidentifikasi kelemahan dan memperbaikinya. Hal ini akan berdampak langsung pada tingkat kepuasan mahasiswa.

Salah satu metode untuk mengevaluasi kualitas layanan akademik adalah melalui analisis sentiment Terhadap ulasan mahasiswa. Analisis sentiment merupakan teknik pemrosesan bahasa alami (*Natural Language Processing*) yang bertujuan untuk mengkategorikan opini menjadi positif, negatif atau netral[1]. Berbagai penelitian telah membandingkan metode dalam analisis sentimen, seperti *Support Vector Machine* (SVM) [2] dan *Recurrent Neural Network* (RNN) [3]. Studi sebelumnya menunjukkan bahwa metode *deep learning* seperti *Recurrent Neural Network*, terutama dengan arsitektur *Long-Short Term Memory* (LSTM), sering memberikan akurasi yang lebih baik dibandingkan metode berbasis pembelajaran mesin tradisional seperti SVM [4]. Namun, metode SVM tetap relevan, terutama jika dikombinasikan dengan teknik seleksi fitur seperti *Recursive Feature Elimination* (RFE) yang dapat meningkatkan efisiensi klasifikasi[5]

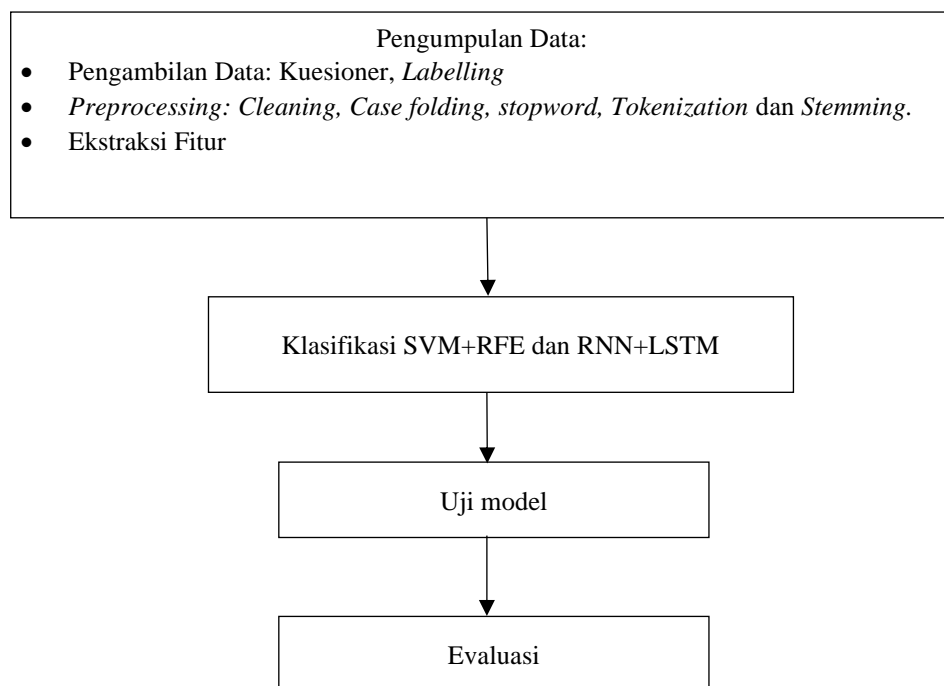
Penelitian ini bertujuan untuk membandingkan Kinerja metode *Support Vector Machine* (SVM) dengan fitur *Recursive Feature Elimination* (RFE) dan *Recurrent Neural Network* (RNN) dengan fitur *Long Short-Term Memory* (LSTM) dalam menganalisis sentimen ulasan mahasiswa terhadap layanan biro akademik Universitas Annuqayah. Hasil dari penelitian ini diharapkan dapat memberikan rekomendasi metode terbaik untuk mengevaluasi persepsi mahasiswa Terhadap layanan akademik, sehingga dapat digunakan sebagai dasar perbaikan kualitas layanan di institusi Pendidikan tinggi.

2. METODOLOGI PENELITIAN

2.1 Persiapan Data

Dalam penelitian ini menggunakan data yang didapat berdasarkan kuesione yang diberikan kepada mahasiswa Univesitas Annuqayah menggunakan *Google Form* dengan kurun waktu 33 hari dimulai dari tanggal 15 Juni sampai 18 Juli 2024. Pada *dataset* tersebut memiliki 19 atribut, namun hanya beberapa atribut yang akan digunakan untuk proses klasifikasi.

Data yang digunakan yakni data teks terkait ulasan mahasiswa terhadap pelayanan biro akademik Universitas Annuqayah berjumlah 103 *datarow*. Data teks tersebut kemudian diberikan label secara manual dengan menentukan *keyword* tertentu untuk menentukan *tag* utama yaitu, label positif dan negatif. Data tersebut kemudian dikelompokkan berdasarkan labelnya dan diperoleh 72 data berlabel positif dan 31 data berlabel *negative*. Dengan membagi data ke dalam kategori ini, analisis sentimen dapat dilakukan untuk memahami persepsi mahasiswa terhadap layanan biro akademik. Hasil analisis ini membantu dalam mengidentifikasi area yang membutuhkan perbaikan serta aspek-aspek layanan yang sudah diterima dengan baik[6].



Gambar 1. Proses Model

2.2 Pre-Processing data

Beberapa tahapan dalam *preprocessing data* adalah *cleaning*, *casefolding*, *stopword*, *tokenization* dan *stemming*[7]. *Cleaning* adalah Proses pembersihan data yang dilakukan untuk mengidentifikasi serta memperbaiki atau menghilangkan data yang tidak valid maupun tidak relevan dalam suatu *dataset*[8]. *Case folding* atau *case*

normalization merupakan salah satu langkah dalam *text preprocessing* yang bertujuan untuk menyamakan format huruf dalam teks dengan mengubahnya seluruhnya menjadi huruf kecil atau huruf besar. Proses ini membantu dalam mempermudah perbandingan dan analisis teks[9]. Kata-kata yang tidak memiliki makna signifikan atau kurang efektif akan dihapus dengan menerapkan *stopwords*[7]. Tokenisasi adalah tahap yang digunakan untuk memecah teks menjadi unit-unit kata. Proses analisis kata dalam teks menjadi krusial karena melalui tokenisasi, arti dari teks dapat lebih mudah diidentifikasi[7]. *Stemming* merupakan proses mengubah kata atau token menjadi bentuk dasar kata. Tujuan utamanya adalah untuk menyederhanakan kata-kata yang memiliki variasi menjadi satu bentuk dasar yang sama, sehingga memudahkan dalam pengelompokan atau analisis kata. Proses ini banyak diterapkan dalam *Natural Language Processing* (NLP) dan pencarian informasi (IR)[10].

Pada tahap *pre-processing data* ini data mentah yang berupa data *text* dilakukan proses *cleaning*, *casefolding*, *stopword*, *tokenization* dan *stemming*. Hasil dari proses ini akan menghasilkan *Output* dari tahap tersebut akan menghasilkan fitur-fitur yang selanjutnya digunakan sebagai input dalam proses pembelajaran mesin oleh SVM-RFE dan RNN-LSTM.

2.3 Pembobotan Term

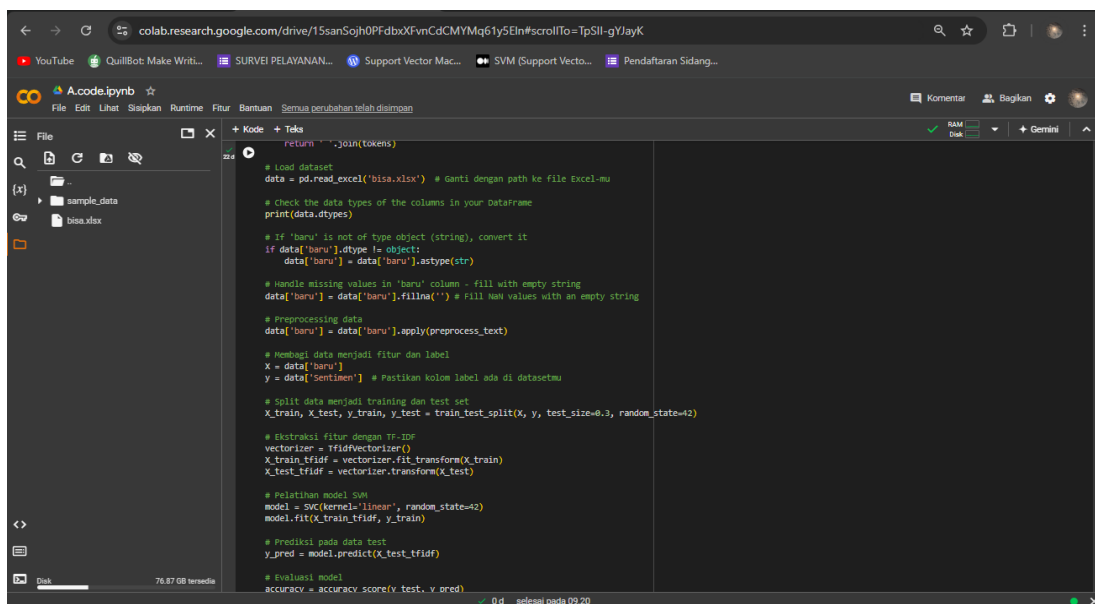
Proses pembobotan *term* dilakukan setelah data mentah diubah menjadi matriks dokumen melalui tahap sebelumnya, yaitu *pre-processing*. Pembobotan *term* ini menghasilkan matriks dokumen *term* yang merepresentasikan kumpulan dokumen yang digunakan untuk klasifikasi teks. Dalam penelitian ini, metode yang digunakan untuk pembobotan adalah TF-IDF, yang memberikan bobot pada setiap *term* berdasarkan frekuensi kemunculannya dalam kumpulan dokumen yang tersedia.

TF-IDF (*Term Frequency-Inverse Document Frequency*) adalah teknik yang digunakan untuk mengukur seberapa penting sebuah kata dalam sebuah dokumen dibandingkan dengan koleksi dokumen lainnya. Fitur TF-IDF ini digunakan untuk meningkatkan kinerja dalam analisis sentimen. Dengan mengkombinasikan ekstraksi fitur TF-IDF dan algoritma *stochastic gradient descent*, analisis sentimen dapat dengan akurat mengklasifikasikan teks berbahasa Indonesia berdasarkan sentimen positif atau negatif[11].

2.4 Klasifikasi Menggunakan Support Vector Machine

Sebelum diklasifikasi menggunakan algoritma SVM-RFE dan RNN-LSTM, data dibagi menjadi data latih dan data uji. Ini memungkinkan proses klasifikasi menjadi lebih mudah. Dataset pelatihan digunakan untuk mengembangkan model klasifikasi, yang berisi informasi esensial untuk memprediksi kelas sentimen yang baru. Dengan semakin banyaknya data yang digunakan dalam pelatihan, algoritma akan semakin mampu memahami data dan menginterpretasikan data tersebut secara akurat. Selain itu, data latihan dan uji dibagi dengan perbandingan data 80:20, 70:30, dan 60:40 dalam analisis ini untuk mengetahui persentase algoritma klasifikasi yang berhasil mengklasifikasikan dengan benar. Proporsi 80:20, 70:30, dan 60:40 memberikan peneliti cukup sampel untuk melatih model, sekaligus menyediakan data uji yang cukup untuk evaluasi. Hal ini krusial karena model klasifikasi membutuhkan data yang cukup untuk belajar dan memastikan generalisasi yang baik.

Algoritma *Support Vector Machine* digunakan untuk mengklasifikasikan data terlebih dahulu. Proses pengklasifikasian data menggunakan algoritma *Support Vector Machine* digambarkan pada gambar berikut ini.



```
return ''.join(tokens)

# Load dataset
data = pd.read_excel('bisa.xlsx') # Ganti dengan path ke file Excel-mu

# Check the data types of the columns in your DataFrame
print(data.dtypes)

# If 'baru' is not of type object (string), convert it
if data['baru'].dtype != object:
    data['baru'] = data['baru'].astype(str)

# Handle missing values in 'baru' column - fill with empty string
data['baru'] = data['baru'].fillna("") # Fill NaN values with an empty string

# Preprocessing data
data['baru'] = data['baru'].apply(preprocess_text)

# Bagi data menjadi fitur dan label
X = data['baru']
y = data['Sentimen'] # Pastikan kolom label ada di datasetmu

# Split data menjadi training dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Ekstraksi fitur dengan TF-IDF
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Relasi model SVM
model = SVC(kernel='linear', random_state=42)
model.fit(X_train_tfidf, y_train)

# Prediksi pada data test
y_pred = model.predict(X_test_tfidf)

# Evaluasi model
accuracy = accuracy_score(y_test, y_pred)
```

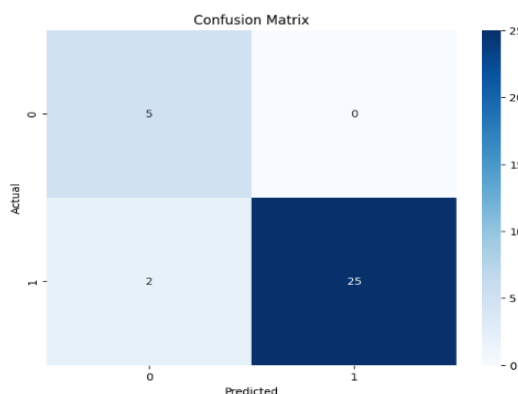
Gambar 2. Proses klasifikasi dengan Support Vector Machine

Berdasarkan hasil dari penghitungan yang ditunjukkan oleh gambar 2, *script* yang telah dijalankan sebelumnya dapat dilihat pada table 1. Dari tabel 1 tersebut menunjukkan hasil akurasi dengan model SVM yang tertinggi didapatkan dengan nilai sebesar 93,75% dengan perbandingan 70:30 data latih dan data uji.

Tabel 1. Hasil Akurasi SVM berdasarkan Data Latih dan Data Uji

Perbandingan Data Latih & Data Uji	Hasil Akurasi SVM
80:20	85,71%
70:30	93,75%
60:40	86%

Dari tabel 1 dapat diketahui bahwa metode klasifikasi *Support Vector Machine* memiliki akurasi paling tinggi dalam mengklasifikasikan data ulasan mahasiswa terkait pelayanan biro akademik Universitas Annuqayah berdasarkan data latih dan data uji 70:30 dengan hasil tertinggi yaitu sebesar 93,75%.



Gambar 3. Confusion Matrix metode SVM 70:30

Dari gambar 3 menunjukkan pada data ulasan mahasiswa terhadap pelayanan Biro Akademik Universitas Annuqayah menggunakan metode *Support Vector Machine* terdapat 25 data positif yang benar diprediksi dan termasuk ke dalam kelas sentiment positif (*True Positive*) dan 5 data negatif yang diprediksi termasuk ke dalam kelas sentiment negatif (*True Negative*). Kemudian terdapat 2 data negatif yang termasuk ke dalam kelas sentiment positif (*False Positive*) dan tidak ada data negatif yang termasuk ke kelas sentiment positif. Berikut pengitugan akurasi, presisi, *recall* dan *f1-score* secara manual engan *True Positive* (TP) 25 data, *True Negative* (TN) 5 data, *False Positive* (FP) 0 data dan *False Negative* (FN) 2 data.

$$\begin{aligned} \text{a. Akurasi} &= \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \\ &= \frac{25+5}{25+5+0+2} \times 100\% \\ &= 0,9375 \end{aligned} \quad (1)$$

$$\begin{aligned} \text{b. Presisi} &= \frac{TP}{TP+FP} \times 100\% \\ &= \frac{25}{25+0} \times 100\% \\ &= 1 \end{aligned} \quad (2)$$

$$\begin{aligned} \text{c. Recall} &= \frac{TP}{TP+FN} \times 100\% \\ &= \frac{25}{25+2} \times 100\% \\ &= 0,9259 \end{aligned} \quad (3)$$

$$\begin{aligned} \text{d. F1-Score} &= 2 \times \frac{\text{Presisi} \times \text{Recall}}{\text{Presisi} + \text{Recall}} \\ &= 2 \times \frac{1 \times 0,9259}{1 + 0,9259} \\ &= 0.9615 \end{aligned} \quad (4)$$

Dengan akurasi keseluruhan 93,75%, model SVM dengan perbandingan 70:30, sangat akurat dalam mendeteksi kelas "positif", yang memiliki presisi dan recall yang tinggi. presisi kelas "negatif" sangat tinggi (1,00), namun recallnya lebih rendah (0,67), menunjukkan bahwa model gagal mendeteksi beberapa sampel negatif dengan benar. Hasil ini menunjukkan kinerja yang baik meskipun datanya kecil. Berikut contoh penghitungan manual menggunakan algoritma *Support Vector Machine*:
Misalkan terdapat dua kelas data berikut:

- Kelas positif: (2,3) dan (3,3)
 - Kelas negatif: (1,1) dan (1,2)
- Label untuk setiap titik data:
- Kelas positif diberi label +1
 - Kelas negatif diberi label -1

Langkah selanjutnya Membuat Persamaan *Hyperplane*. Berikut penghitungannya adalah:

$$w_1 \cdot x_1 + w_2 \cdot x_2 + b = 0 \quad (5)$$

di mana w_1 dan w_2 adalah bobot (vektor normal terhadap *hyperplane*), dan b adalah bias. Tujuannya adalah untuk menemukan w_1 , w_2 , dan b sehingga persamaan ini memisahkan data positif dan negatif dengan margin terbesar [12].

Untuk data positif ($y = +1$):

$$w_1 \cdot x_1 + w_2 \cdot x_2 + b \geq 1 \quad (6)$$

Untuk data negatif ($y = -1$):

$$w_1 \cdot x_1 + w_2 \cdot x_2 + b \leq -1 \quad (7)$$

Misalkan kita coba untuk dua titik data:

- (2,3) dengan $y = +1$
- (1,1) dengan $y = -1$

data sementara $w_1 = 1$, $w_2 = -1$, dan $b = 0$. berikut uji persamaannya:

Untuk titik (2,3) dari kelas positif:

$$w_1 \cdot 2 + w_2 \cdot 3 + b = 1 \cdot 2 + (-1) \cdot 3 + 0 = 2 - 3 = -1$$

Ternyata, hasilnya tidak sesuai karena hasil harus lebih dari atau sama dengan 1 untuk kelas positif. Ini artinya nilai w_1 , w_2 , atau b perlu disesuaikan. Kemudian mencoba mengambil bobot baru:

Misalkan $w_1 = 1$, $w_2 = 1$, dan $b = -3$.

Kemudia lakukan pengujian kembali untuk titik positif (2,3):

$$w_1 \cdot 2 + w_2 \cdot 3 + b = 1 \cdot 2 + 1 \cdot 3 - 3 = 2 + 3 - 3 = 2$$

persamaan ini valid karena lebih besar dari 1.

Untuk titik negatif (1,1):

$$w_1 \cdot 1 + w_2 \cdot 1 + b = 1 \cdot 1 + 1 \cdot 1 - 3 = 1 + 1 - 3 = -1$$

persamaan ini juga valid karena kurang dari -1 untuk kelas negatif.

Selain itu, peneliti menambahkan fitur *Recursive Feature Elimination* pada algoritma *Support Vector Machine* dan bisa dilihat pada gambar berikut:

```

# Import libraries
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import RFE
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load data
data = data['sentimen']

# Split data menjadi training dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Vectorize the text data using TF-IDF
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Membuat model SVM
model = SVC(kernel='linear')

# Melatih model dengan data training
model.fit(X_train_tfidf, y_train)

# Menggunakan RFE untuk memilih fitur terbaik
rfe = RFE(model, n_features_to_select=10)
rfe.fit(X_train_tfidf, y_train)

# Menampilkan fitur yang dipilih
selected_features = np.array(vectorizer.get_feature_names_out())[rfe.support_]
print("Fitur yang dipilih:", selected_features)

# Melatih model dengan fitur yang dipilih
X_train_rfe = rfe.transform(X_train_tfidf)
X_test_rfe = rfe.transform(X_test_tfidf)
model.fit(X_train_rfe, y_train)

# Prediksi dan evaluasi
y_pred = model.predict(X_test_rfe)
print("Akurasi:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))

# Visualisasi dengan heatmap
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8,6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('')

```

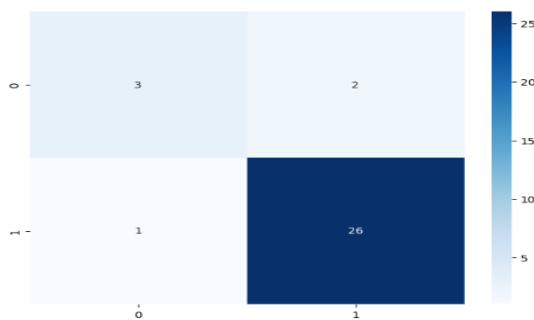
Gambar 4. Proses klasifikasi data menggunakan SVM dengan RFE

Tabel 2 menunjukkan hasil penghitungan berdasarkan gambar *script* yang telah dijalankan sebelumnya. Tabel tersebut menunjukkan nilai akurasi model SVM dengan *Recursive Feature Elimination* tertinggi sebesar 90,63% dengan perbandingan 70:30 antara data latih dan data uji.

Tabel 2. Hasil Akurasi SVM-RFE berdasarkan data latih dan data uji

Perbandingan Data Latih & Data Uji	Hasil Akurasi SVM+RFE
80:20	85,71%
70:30	90,63%
60:40	80,95%

Berdasarkan distribusi data pelatihan dan data pengujian dengan rasio 70:30, metode klasifikasi *Support Vector Machine-Recursive Feature Elimination* memiliki tingkat akurasi tertinggi dalam mengklasifikasikan data ulasan mahasiswa terkait pelayanan biro akademik Universitas Annuqayah, dengan hasil tertinggi sebesar 90,63%, seperti yang ditunjukkan dalam tabel 1.



Gambar 5. Confusion Matrix dari SVM-RFE

Gambar 5 menunjukkan pada data ulasan mahasiswa terhadap pelayanan Biro Akademik menggunakan metode *Support Vector Machine-Recursive Feature Elimination* terdapat 26 data positif yang benar diprediksi dan termasuk ke dalam kelas sentiment positif (*True Positive*) dan 3 data negatif yang diprediksi termasuk ke dalam kelas sentiment negative (*True Negative*). Kemudian terdapat 1 data negative yang termasuk ke dalam kelas sentiment positif (*False Positive*) dan 2 data negatif yang termasuk ke kelas sentiment positif (*False negative*).

Model SVM-RFE memiliki kinerja yang baik, dengan akurasi keseluruhan 90,63%. Kelas "positif" ditangani dengan sangat baik dengan skor F1 yang tinggi, presisi, dan *recall*, sedangkan kelas "negatif" menunjukkan kinerja yang solid dengan presisi dan *recall* yang setara pada 75%. Ini menunjukkan keseimbangan yang baik dalam performa model antara kedua kelas, meskipun kelas "positif" mendapatkan perhatian yang lebih besar pada *recall* dan presisi.

2.5 Klasifikasi *Recurrent Neural Network* dengan *Long Short-Term Memory*

Peneliti menggunakan algoritma *Deep Learning* seperti RNN-*Long Short-Term Memory* dan *Support Vector Machine-recursive Feature Elimination* untuk melakukan klasifikasi, selain juga menggunakan *Support Vector Machine* dan *Support Vector Machine-recursive Feature Elimination*. Gambar kode skrip dari prosedur klasifikasi *Recurrent Neural Network* dapat ditemukan di bawah ini.

```

colab.research.google.com/drive/15sanSoh0PfdbxKfVnCdCMYMQ61y5Eln#scrollTo=exvVry1Wfuh1
A.code.ipynb
File Edit Lihat Simpan Runtime Fitur Bantuan Semua perubahan telah disimpan
File
  sample_data
  data.xlsx
Kode
  1 # 1. Prapemrosesan Data (tokenisasi dan Padding)
  tokenizer = Tokenizer(num_words=5000)
  tokenizer.fit_on_texts(X)
  X_seq = tokenizer.texts_to_sequences(X)

  # 2. Padding
  max_len = 100 # sesuaikan panjang ini sesuai kebutuhan
  X_pad = pad_sequences(X_seq, maxlen=max_len)

  # 3. Membagi data menjadi set pelatihan dan pengujian
  X_train, X_test, y_train, y_test = train_test_split(X_pad, y, test_size=0.3, random_state=42)

  # 4. Membangun Model RNN
  model = Sequential([
    Embedding(input_dim=5000, output_dim=128, input_length=max_len),
    SimpleRNN(128, return_sequences=False),
    Dense(1, activation='sigmoid') # output layer for binary classification
  ])

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

  # 5. Melatih Model
  model.fit(X_train, y_train, epochs=5, batch_size=32, validation_data=(X_test, y_test))

  # 6. Evaluasi Model
  y_pred_prob = model.predict(X_test)
  y_pred = (y_pred_prob > 0.5).astype("int32") # Convert probabilities to binary predictions

  # 7. Hitung metrik
  accuracy = accuracy

  # 8. Hitung metrik
  accuracy = accuracy_score(y_test, y_pred)
  f1 = f1_score(y_test, y_pred)
  precision = precision_score(y_test, y_pred)
  recall = recall_score(y_test, y_pred)

  # cetak hasilnya
  
```

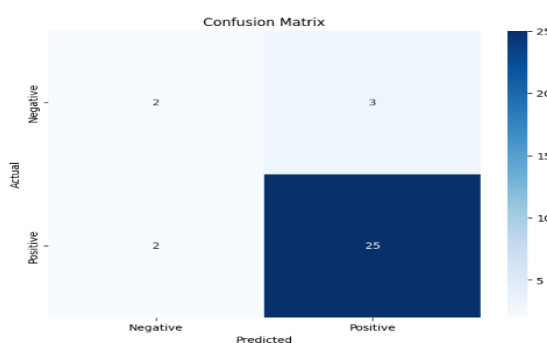
Gambar 6. Proses klasifikasi dengan metode *Recurrent Neural Network*

Berdasarkan gambar *script* yang telah dijalankan sebelumnya, hasil penghitungan disajikan dalam tabel 3. Tabel tersebut menunjukkan nilai akurasi tertinggi model *Recurrent Neural Network* sebesar 81,25%, dengan perbandingan 70:30 antara data uji dan data latih.

Tabel 3. Hasil Akurasi RNN Berdasarkan Data Latih & Data Uji

Perbandingan Data Latih & Data Uji	Hasil Akurasi RNN
80:20	76,19%
70:30	81,25%
60:40	73,81%

Dari tabel 3, dapat dilihat bahwa metode klasifikasi *Recurrent Neural Network* memiliki akurasi tertinggi dalam mengklasifikasikan data ulasan mahasiswa terkait pelayanan biro akademik, berdasarkan perbandingan data latih dan data uji 70:30. Metode ini juga memiliki hasil tertinggi, yaitu 81,25%.



Gambar 7. Confusion Matrix RNN

Dari gambar 7 menunjukkan pada data ulasan mahasiswa terhadap pelayanan Biro Akademik Universitas Annuqayah menggunakan metode *Support Vector Machine-Recursive Feature Elimination* terdapat 25 data positif yang benar diprediksi dan termasuk ke dalam kelas sentiment positif (*True Positive*) dan 2 data negatif yang diprediksi termasuk ke dalam kelas sentiment negatif (*True Negative*). Kemudian terdapat 3 data negatif yang termasuk ke dalam kelas sentiment positif (*False Positive*) dan 2 data negatif yang termasuk ke kelas sentiment negatif (*False negative*). Berikut perhitungan manual dengan metode *Recurrent Neural Network*: Contoh kalimat “Pelayanan biro akademik sangat buruk” dengan bobot sebagai berikut:

- Pelayanan : [0.1, 0.3]
- Biro : [0.2, 0.1]
- Akademik : [0.4, 0.5]
- Sangat : [0.2, 0.2]
- Buruk : [0.3, 0.4]

Langkah selanjutnya adalah menyusun model RNN dengan parameter sebagai berikut:

- 2 unit *Hidden layer*
- Bobot *input* ke *hidden* (W_x)

$$W_x = \begin{bmatrix} 0.5 & 0.1 \\ 0.4 & 0.2 \end{bmatrix}$$

- Bobot *hidden* ke *hidden* (W_h)

$$W_h = \begin{bmatrix} 0.3 & 0.2 \\ 0.1 & 0.4 \end{bmatrix}$$

- Bias* (b)

$$b = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$$

Langkah selanjutnya yakni menginisialisai *hidden state*. Untuk *hidden state* (h_0) = [0,0]. Kemudian menghitung *hidden state* dari setiap token. Berikut penghitungannya:

- Pelayanan

$$h_1 = \tanh(W_h * h_0 + W_x * [0.1, 0.3] + b)$$

$$h_1 = \tanh\left(\begin{bmatrix} 0.3 & 0.2 \\ 0.1 & 0.4 \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.5 & 0.1 \\ 0.4 & 0.2 \end{bmatrix} * \begin{bmatrix} 0.1 \\ 0.3 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}\right)$$

$$h_1 = \tanh\left(\begin{bmatrix} 0.05 + 0.03 + 0.1 \\ 0.04 + 0.06 + 0.2 \end{bmatrix}\right) = \tanh\left(\begin{bmatrix} 0.18 \\ 0.203 \end{bmatrix}\right)$$

$$h_1 \approx [0.18, 0.29]$$

b. Biro

$$h_2 = \tanh(W_h * h_1 + W_z * [0.2, 0.1] + b)$$

$$h_2 = \tanh\left(\begin{bmatrix} 0.3 & 0.2 \\ 0.1 & 0.4 \end{bmatrix} * \begin{bmatrix} 0.18 \\ 0.29 \end{bmatrix} + \begin{bmatrix} 0.5 & 0.1 \\ 0.4 & 0.2 \end{bmatrix} * \begin{bmatrix} 0.2 \\ 0.1 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}\right)$$

$$h_2 = \tanh\left(\begin{bmatrix} 0.3 * 0.18 + 0.2 * 0.29 + 0.1 * 0.2 + 0.1 \\ 0.1 * 0.18 + 0.4 * 0.29 + 0.2 * 0.1 + 0.2 \end{bmatrix}\right)$$

$$= \tanh\left(\begin{bmatrix} 0.0582 + 0.058 + 0.02 + 0.1 \\ 0.018 + 0.116 + 0.02 + 0.2 \end{bmatrix}\right)$$

$$h_2 = \tanh\left(\begin{bmatrix} 0.2362 \\ 0.354 \end{bmatrix}\right)$$

$$h_2 \approx [0.233, 0.29]$$

Untuk penghitungan kata akademik, sangat dan buruk dilakukan dengan penghitungan seperti di atas. setelah melakukan penghitungan di *hidden state*, langkah selanjutnya adalah menghitung *output layer*. Langkah pertama yakni menggunakan *hidden state* terakhir (h_T) untuk menghasilkan *output*. Misalkan dengan bobot *output* W_o dan *bias output* b_o sebagai berikut:

$$W_o = \begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.5 \end{bmatrix}$$

$$b_o = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$$

$$output = \text{softmax}(W_o * h_2 + b_o)$$

$$output = \text{softmax}\left(\begin{bmatrix} 0.6 * 0.233 + 0.4 * 0.339 + 0.1 \\ 0.3 * 0.233 + 0.5 * 0.339 + 0.2 \end{bmatrix}\right)$$

$$output = \text{softmax}\left(\begin{bmatrix} 0.299 + 0.136 + 0.1 \\ 0.070 + 0.170 + 0.2 \end{bmatrix}\right)$$

$$output = \text{softmax}\left(\begin{bmatrix} 0.535 \\ 0.440 \end{bmatrix}\right)$$

$$output = \left(\frac{e^{0.535}}{e^{0.535} + e^{0.440}}\right) \approx \begin{bmatrix} 0.53 \\ 0.47 \end{bmatrix}$$

dari penghitungan didapat hasil yaitu $\begin{bmatrix} 0.53 \\ 0.47 \end{bmatrix}$, karna menggunakan *softmax*, *output* tersebut menunjukkan probabilitas untuk masing-masing kelas, yakni kelas 0 untuk negatif dan kelas 1 untuk kelas positif. Dengan kelas 0 sebesar 0.53 dan kelas 1 sebesar 0.47. Dari nilai tersebut dapat dilihat bahwa probabilitas kelas 0 (negatif) lebih tinggi (0.53) dibandingkan dengan probabilitas kelas 1 (0.47), dapat disimpulkan bahwa teks tersebut menunjukkan kelas negatif.

Peneliti juga menambahkan *Long Short-Term Memory* pada algoritma *Recurrent Neural Network* untuk melakukan klasifikasi data lanjutan, seperti yang ditunjukkan oleh *script* pada gambar berikut:

```

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
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, LSTM, Dense, Dropout
import matplotlib.pyplot as plt

# Mem baca data
data = pd.read_excel('final.xlsx') # ubah nama file sesuai dengan dataset Anda

# Mem baca fitur dan label
x = data['text']
y = data['Sentimen']

# Encode label menjadi angka
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Tokenisasi teks
tokenizer = Tokenizer(num_words=10000) # Sesuaikan num_words sesuai kebutuhan
tokenizer.fit_on_texts(x)
X_sequences = tokenizer.texts_to_sequences(x)

# Padding sequences
X_padded = pad_sequences(X_sequences, maxlen=100) # Sesuaikan maxlen sesuai kebutuhan

# Membagi data menjadi set pelatihan dan pengujian
X_train, X_test, y_train, y_test = train_test_split(X_padded, y_encoded, test_size=0.2, random_state=42)

# Membangun model RNN
model = Sequential()
model.add(Embedding(input_dim=10000, output_dim=128, input_length=100)) # Sesuaikan input_dim dan output_dim

```

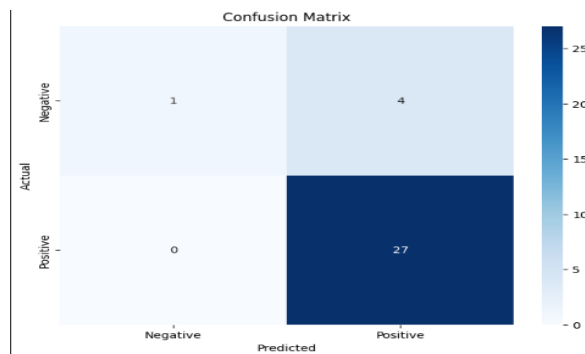
Gambar 8. Proses Klasifikasi dengan RNN-LSTM

Berdasarkan gambar *script* yang telah dijalankan sebelumnya, hasil penghitungan ditunjukkan dalam Tabel 4. Tabel tersebut menunjukkan bahwa model RNN mencapai nilai akurasi tertinggi sebesar 82,38% dengan perbandingan 70:30 antara data latihan dan data uji.

Tabel 4. Hasil Akurasi RNN-LSTM Berdasarkan Data Latih Dan Data Uji

Perbandingan Data Lath & Data Uji	Hasil Akurasi RNN-LSTM
80:20	80,95%
70:30	87,50%
60:40	76,19%

Berdasarkan perbandingan data latih dan data uji 70:30, terbukti memiliki akurasi tertinggi dalam mengklasifikasikan data ulasan mahasiswa terkait pelayanan biro akademik Universitas Annuqayah dengan metode klasifikasi *Recurrent Neural Network* dengan *Long Short-Term Memory*, seperti yang ditunjukkan dalam tabel 4. Metode ini juga menghasilkan hasil tertinggi, 87,50%.



Gambar 9. Confusion Matrix RNN-LSTM

Dari gambar 9 menunjukkan pada data ulasan mahasiswa terhadap pelayanan Biro Akademik Universitas Annuqayah menggunakan metode *Support Vector Machine-Recursive Feature Elimination* terdapat 27 data positif yang dengan tepat diklasifikasikan sebagai sentiment positif (*True Positive*) dan 1 data negatif yang dengan tepat diklasifikasikan sebagai sentiment negatif (*True Negative*). Kemudian didapati 4 Data yang sebenarnya positif tetapi salah diidentifikasi sebagai sentiment negatif (*False Positive*) dan tidak ada Data yang sebenarnya negatif tetapi salah diidentifikasi sebagai sentiment positif (*False Negative*). Berikut contoh penghitungan manual dengan metode RNN-LSTM:

Bobot *Output* dan *Bias*

Semisal Bobot *Output* (w_{out}):

$$\begin{bmatrix} 0.3 & 0.4 \\ 0.5 & 0.6 \end{bmatrix}$$

Dan *bias Output* (b_{out}):

$$\begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$$

Menghitung *Output Layer*

$$Output = w_{out} * h_t + b_{out}$$

$$\begin{aligned} w_{out} * h_t &= \begin{bmatrix} 0.3 & 0.4 \\ 0.5 & 0.6 \end{bmatrix} * \begin{bmatrix} 0.421 \\ 0.404 \end{bmatrix} = \begin{bmatrix} 0.3 * 0.421 + 0.4 * 0.404 \\ 0.5 * 0.421 + 0.6 * 0.404 \end{bmatrix} \\ &= \begin{bmatrix} 0.1263 + 0.1616 \\ 0.2105 + 0.2424 \end{bmatrix} = \begin{bmatrix} 0.2879 \\ 0.4529 \end{bmatrix} \\ Output &= \begin{bmatrix} 0.2879 \\ 0.4529 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix} = \begin{bmatrix} 0.3879 \\ 0.6529 \end{bmatrix} \end{aligned}$$

Menggunakan *Softmax Activation* untuk mendapatkan probabilitas:

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

$$exp(x) = \begin{bmatrix} e^{0.3879} \\ e^{0.6529} \end{bmatrix} \approx \begin{bmatrix} 1.474 \\ 1.921 \end{bmatrix}$$

$$Sum(exp(x)) = 1.474 + 1.921 = 3.395$$

$$Softmax = \begin{bmatrix} \frac{1.474}{3.395} \\ \frac{1.921}{3.395} \end{bmatrix} \approx \begin{bmatrix} 0.434 \\ 0.566 \end{bmatrix}$$

Dari penghitungan ini didapat probabilitas untuk kelas positif yaitu 0.434 dan probabilitas untuk kelas negatif yaitu 0.566. dengan probabilitas ini dapat disimpulkan bahwa dokumen tersebut diklasifikasikan sebagai kelas negatif, karena nilai probabilitas lebih tinggi dari kelas positif.

3. HASIL DAN PEMBAHASAN

Penelitian ini melakukan perbandingan antara metode yang diusulkan dengan beberapa metode lainnya, seperti *Support Vector Machine* (SVM) dan *Recurrent Neural Network* (RNN) dan metode yang diusulkan yaitu *Support Vector Machine* (SVM) dengan *Recursive Feature Elimination* (RFE) dan *Recurrent Neural Network* (RNN) dengan *Long Short-Term Memory* (LSTM).

Tabel 5. Hasil Komparasi

<i>Method</i>	<i>Precision</i>	<i>Recall</i>	<i>Accuracy</i>	<i>F1-Score</i>
SVM	86,21%	92%	93,75%	89%
RNN	86,21%	92,59%	81,25%	89,29%
SVM+RFE	90%	90,62%	90,63%	90%
RNN+LSTM.	87,10%	100%	87,50%	93,10%

Tabel 5 menampilkan perbandingan hasil beberapa metode. Metode *Support Vector Machine* (SVM) memperoleh akurasi sebesar 93,75%, sementara *Recurrent Neural Network* (RNN) mencatatkan akurasi 81,25%. Di sisi lain, akurasi yang diperoleh oleh *Support Vector Machine* dengan *Recursive Feature Elimination* mencapai 90,63%, dan *Recurrent Neural Network* dengan *Long Short-term Memory* mencapai 87,50%. Hasil ini menunjukkan bahwa metode SVM lebih akurat dalam menganalisis sentimen ulasan mahasiswa terhadap pelayanan Biro Akademik Universitas Annuqayah. Meskipun SVM-RFE unggul dalam akurasi dan precision, metode RNN-LSTM lebih baik dalam memahami konteks sentimen, terbukti dengan recall dan F1-Score yang lebih tinggi. Hal ini mengindikasikan bahwa untuk ulasan mahasiswa yang lebih kompleks, RNN-LSTM mungkin lebih sesuai digunakan. Secara keseluruhan, penelitian ini menunjukkan bahwa algoritma RNN-LSTM memberikan performa yang lebih baik dalam memahami konteks sentimen mahasiswa secara lebih mendalam, meskipun SVM-RFE memiliki tingkat presisi yang lebih tinggi.

4. KESIMPULAN

Penelitian ini menganalisis sentiment mahasiswa terhadap pelayanan biro akademik Universitas Annuqayah menggunakan algoritma SVM-RFE dan RNN-LSTM. Hasil penelitian menunjukkan bahwa metode SVM memiliki akurasi tertinggi yakni sebesar 93,75%, sementara SVM-RFE unggul dalam presisi sebesar 90%. Meskipun demikian, algoritma RNN-LSTM lebih baik dalam recall dan F1-score. Hal ini menunjukkan bahwa algoritma RNN-LSTM mampu menangkap dengan baik konteks sentiment secara lebih menyeluruh. Namun, penelitian ini memiliki keterbatasan pada jumlah data yang terbatas, yakni hanya 103 data, sehingga studi lanjutan sangat disarankan untuk menggunakan dataset yang lebih besar dan mengeksplorasi algoritma lain seperti transformer atau BERT. Implementasi sistem otomatis berbasis *machine learning* di biro akademik juga direkomendasikan untuk meningkatkan analisis feedback mahasiswa secara real-time dan meningkatkan kualitas layanan. Pemilihan metode terbaik bergantung pada tujuan analisis, di mana SVM/SVM-RFE cocok untuk presisi dan akurasi, sedangkan RNN-LSTM lebih sesuai untuk data sekuensial dengan ketergantungan jangka Panjang.

DAFTAR PUSTAKA

- [1] S. Farisi and S. Hadi, "Analisis Sentimen menggunakan Recurrent Neural Network Terkait Isu Anies Baswedan Sebagai Calon Presiden 2024," *eProceedings ...*, vol. 10, no. 2, p. 1682, 2023, [Online].
- [2] T. A. Zuraiyah, M. M. Mulyati, and G. H. F. Harahap, "Perbandingan Metode Naïve Bayes, Support Vector Machine Dan Recurrent Neural Network Pada Analisis Sentimen Ulasan Produk E-Commerce," *Multitek Indones.*, vol. 17, no. 1, pp. 28–44, 2023, doi: 10.24269/mtkind.v17i1.7092.
- [3] S. J. and D. K. U., "Comparison of Sentiment Analysis on Online Product Reviews Using Optimised RNN-LSTM with Support Vector Machine," *Webology*, vol. 19, no. 1, pp. 3883–3898, 2022, doi: 10.14704/web/v19i1/web19256.
- [4] K. Solecha and O. Irnawati, "Komparasi Algoritma Support Vector Machine Dan Naïve Bayes Berbasis Particle Swarm Optimization Pada Analisis Sentimen Ulasan Aplikasi Flip," *JIEET (Journal Inf. Eng. Educ. Technol.)*, vol. 07, no. 1, pp. 10–15, 2023.
- [5] I. Budianto, S. N. Anwar, J. T. Lomba, J. Nomor, and K. Semarang, "Analisis Sentiment Pengguna Twitter Mengenai Program Vaksinasi Covid-19 Menggunakan Algoritma Naïve Bayes," *J. Teknol. Inf.*, vol. 6, no. 1, pp. 37–43, 2022, [Online]. Available: <https://www.neliti.com/publications/498397/>
- [6] R. D. Fitriani, H. Yasin, and T. Tarno, "PENANGANAN KLASIFIKASI KELAS DATA TIDAK SEIMBANG DENGAN RANDOM OVERSAMPLING PADA NAIVE BAYES (Studi Kasus: Status

- Peserta KB IUD di Kabupaten Kendal),” *J. Gaussian*, vol. 10, no. 1, pp. 11–20, 2021, doi: 10.14710/j.gauss.v10i1.30243.
- [7] R. R. Salam, M. F. Jamil, Y. Ibrahim, R. Rahmaddeni, S. Soni, and H. Herianto, “Analisis Sentimen Terhadap Bantuan Langsung Tunai (BLT) Bahan Bakar Minyak (BBM) Menggunakan Support Vector Machine,” *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 3, no. 1, pp. 27–35, 2023, doi: 10.57152/malcom.v3i1.590.
- [8] K. Maharana, S. Mondal, and B. Nemade, “A review: Data pre-processing and data augmentation techniques,” *Glob. Transitions Proc.*, vol. 3, no. 1, pp. 91–99, 2022, doi: 10.1016/j.gltp.2022.04.020.
- [9] G. Yunanda, D. Nurjanah, and S. Meliana, “Recommendation System from Microsoft News Data using TF-IDF and Cosine Similarity Methods,” *Build. Informatics, Technol. Sci.*, vol. 4, no. 1, pp. 277–284, 2022, doi: 10.47065/bits.v4i1.1670.
- [10] M. Rahardi, A. Aminuddin, F. F. Abdulloh, and R. A. Nugroho, “Sentiment Analysis of Covid-19 Vaccination using Support Vector Machine in Indonesia,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 6, pp. 534–539, 2022, doi: 10.14569/IJACSA.2022.0130665.
- [11] V. Dwi Antonio, S. Efendi, and H. Mawengkang, “Sentiment analysis for covid-19 in Indonesia on Twitter with TF-IDF featured extraction and stochastic gradient descent,” *Int. J. Nonlinear Anal. Appl.*, vol. 13, no. 1, pp. 2008–6822, 2022, [Online]. Available: <http://dx.doi.org/10.22075/ijnaa.2021.5735>
- [12] M. B. Sulthan, I. Wahyudi dan L. Suhartini, “Analisis Sentimen Pada Tweet Bencana Alam Menggunakan Deep Neural Network Dan Information Gain,” *Jurnal Aplikasi Teknologi Informasi dan Manajemen*, pp. 65-71, 2021.