E-ISSN : 3025-1311
https://ejournal.warunayama.org/kohesi

Kohesi: Jurnal Multidisiplin Saintek
Volume 2, No. 9 2024, pp.51-64

ANALISIS PERBANDINGAN DATASET KDD'99, UNSW-Nb15, dan CICIDS2017 UNTUK SISTEM DETEKSI INTRUSI

Julian Dewanto
Informatika, Fakultas Teknologi Informasi dan Sains Data, Universitas Sebelas Maret
Email: juliandewanto@student.uns.ac.id

Abstract

Perkembangan teknologi informasi telah meningkatkan kebutuhan akan sistem deteksi intrusi yang efektif dalam menghadapi ancaman keamanan yang terus berkembang. Penelitian ini melakukan analisis perbandingan terhadap tiga dataset yang umum digunakan dalam pengujian sistem deteksi intrusi, yaitu KDD'99, UNSW-NB15, dan CICIDS2017. Tujuan utama penelitian ini adalah untuk mengevaluasi kecocokan dan kinerja masing-masing dataset dalam mendukung pengembangan sistem deteksi intrusi yang handal. Metode analisis yang digunakan meliputi pembandingan fitur, distribusi kelas, serta kompleksitas dan variasi serangan yang terdapat dalam setiap dataset. Hasil analisis dapat memberikan pandangan yang lebih jelas bagi peneliti dan praktisi keamanan dalam memilih dataset yang paling sesuai dengan kebutuhan spesifik mereka untuk pengembangan sistem deteksi intrusi yang efektif.

Kata Kunci: Analisis Perbandingan, Sistem Deteksi Intrusi, Fiture Dataset

PENDAHULUAN

Dalam era digital yang semakin berkembang, keamanan jaringan menjadi aspek kritis yang perlu diperhatikan secara serius. Serangan terhadap sistem dan jaringan komputer semakin kompleks dan dapat menimbulkan dampak serius terhadap keberlanjutan operasional organisasi. Sistem deteksi intrusi (IDS) memainkan peran penting dalam melindungi jaringan komputer dari aktivitas jahat. Untuk meningkatkan efektivitas IDS, peneliti sering kali mengandalkan kumpulan data acuan untuk mengevaluasi dan membandingkan berbagai metode deteksi. Tiga kumpulan data yang banyak digunakan untuk tujuan ini adalah KDD'99, UNSW-Nb15, dan CICIDS2017.

Dataset KDD'99 adalah dataset yang digunakan untuk pengembangan sistem deteksi intrusi jaringan pada tahun 1999. Dataset ini terdiri dari sekitar 5 juta koneksi jaringan yang direkam selama seminggu. Dataset ini digunakan untuk mengembangkan model deteksi intrusi jaringan dengan menggunakan teknik pembelajaran mesin seperti Decision Tree, Neural Network, dan Support Vector Machine (Tavallaee et al., 2009). Dataset UNSW-Nb15 adalah dataset yang digunakan untuk pengembangan sistem deteksi intrusi jaringan pada tahun 2015. Dataset ini terdiri dari sekitar 2 juta koneksi jaringan yang direkam selama satu bulan. Dataset ini digunakan untuk mengembangkan model deteksi intrusi jaringan dengan menggunakan teknik pembelajaran mesin seperti Random Forest, Naive Bayes, dan K-Nearest Neighbor (Moustafa & Slay, 2015). Dataset CICIDS2017 adalah dataset yang digunakan untuk pengembangan sistem deteksi intrusi jaringan pada tahun 2017. Dataset ini terdiri dari sekitar 80 juta koneksi jaringan yang

direkam selama satu bulan. Dataset ini digunakan untuk mengembangkan model deteksi intrusi jaringan dengan menggunakan teknik pembelajaran mesin seperti Decision Tree, Random Forest, dan Gradient Boosting (Panigrahi & Borah, 2018).

Penggunaan dataset tersebut untuk mengembangkan model deteksi intrusi jaringan yang dapat mengenali pola-pola yang mencurigakan pada jaringan. Dengan menggunakan model deteksi intrusi jaringan yang baik, maka dapat meningkatkan keamanan jaringan dan melindungi sistem dan data dari serangan yang dapat merusak atau mencuri informasi. Menurut (Farnaaz & Akhil, 2016) bahwa semakin besar ukuran sampel dataset, semakin baik performa model deteksi intrusi jaringan. Kemudian, (Liu et al., 2020) menemukan bahwa pada data yang tidak seimbang, hampir semua algoritma klasifikasi akan menghasilkan akurasi yang jauh lebih tinggi untuk kelas mayoritas daripada kelas minoritas, sehingga teknik sampling ulang sampel kelas minoritas dapat digunakan untuk menyeimbangkan dataset.

Dalam penelitian ini, tujuan yang ingin dicapai adalah menentukan teknik preprocessing yang tepat untuk digunakan pada masalah ketidakseimbangan data pada dataset dengan atribut numerik dan kelas nominal. Metode yang digunakan adalah teknik preprocessing dengan menggunakan metode PCA (Principal Component Analysis).

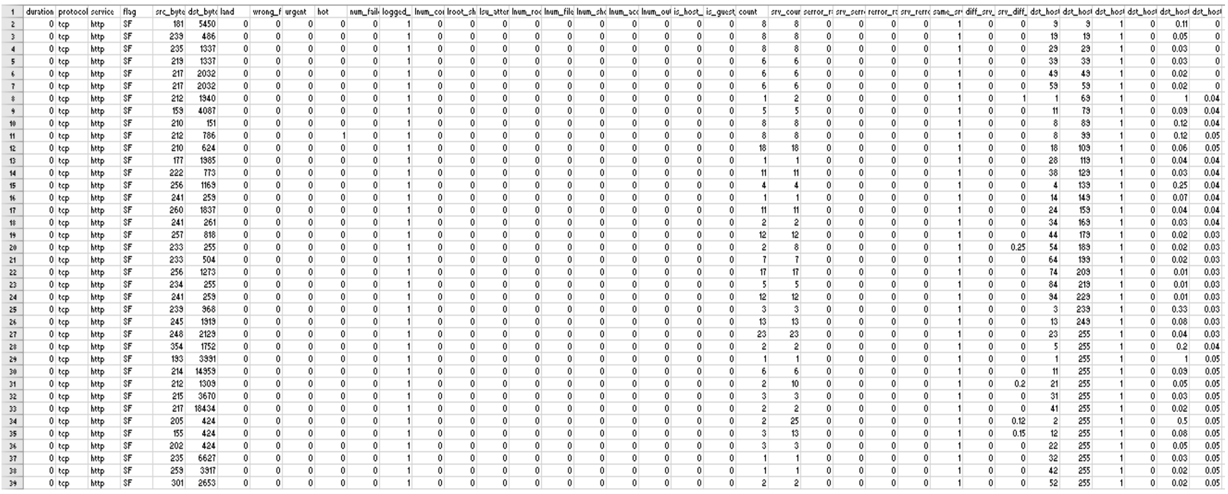
METODE

Dalam penelitian ini dilakukan serangkaian tahapan guna mencapai hasil analisis representatif. Analisis yang digunakan adalah dengan metode PCA (Principal Component Analysis) dengan melakukan pengambilan dataset yang diperoleh dari sumber yang ada seperti KDD'99 dari UC Berkeley, UNSW-Nb15 dari University of New South Wales, dan CICIDS2017 dari Nanyang Technological University. Kemudian, melakukan pemodelan konsep dengan menggunakan teknik preprocessing yang umum, seperti PCA (Principal Component Analysis). Dalam menganalisis dataset tersebut, dilakukan pengujian model dengan menggunakan metode Logistic Regression yang digunakan untuk memodelkan probabilitas keanggotaan suatu observasi ke dalam suatu kelas.

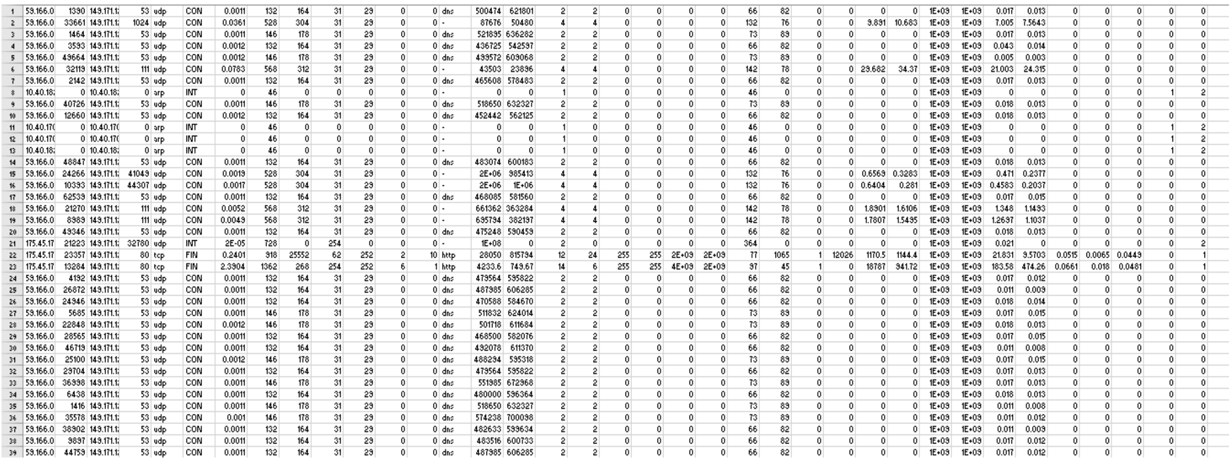
PEMBAHASAN

1. Memuat dan Membersihkan Dataset

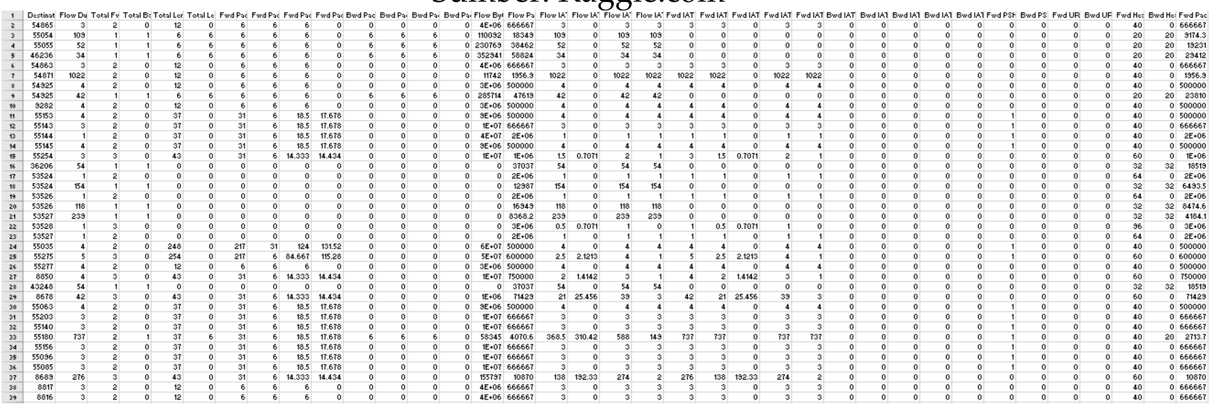
   Pada tahap ini, dilakukan muat dan pembersihan dataset untuk persiapan analisis. Tiga dataset utama digunakan, yaitu KDD'99, UNSW-NB15, dan CICIDS2017. Setiap dataset dimuat ke dalam DataFrame, dan beberapa langkah pembersihan dilakukan, seperti menghapus kolom yang tidak diperlukan dan mengonversi fitur kategorikal menjadi bentuk numerik menggunakan label encoding. Berikut penjelasan dari ketiga dataset yang digunakan, yaitu:



Gambar 1. Data KDD'99
Sumber: Kaggle.com



Gambar 2. Data UNSW-NB15
Sumber: Kaggle.com



Gambar 3. Data CICIDS2017
Sumber: Kaggle.com

Pada dataset diatas dilakukan pengambilan beberapa kolom yang diperlukan untuk dilakukan analisis terhadap dataset KDD'99, UNSW-NB15, dan CICIDS2017. Kemudian memuat dataset dengan langkah sebagai berikut.

a. KDD'99

```python
# Fungsi untuk memuat dataset KDD'99
def load_kdd_dataset():
    # Sesuaikan path dengan nama file dan path di Google Colab
    file_path = 'kddcup.data_10_percent'

    # Daftar nama kolom sesuai dengan dokumentasi KDD'99
    columns = [
        'duration', 'protocol_type', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land',
        'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in', 'num_compromised',
        'root_shell', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells',
        'num_access_files', 'num_outbound_cmds', 'is_host_login', 'is_guest_login', 'count',
        'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate',
        'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count',
        'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate',
        'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate', 'dst_host_serror_rate',
        'dst_host_srv_serror_rate', 'dst_host_rerror_rate', 'dst_host_srv_rerror_rate',
        'outcome'
    ]

    # Muat dataset KDD'99
    df_kdd = pd.read_csv(file_path, names=columns, low_memory=False)
    return df_kdd
```

Gambar 3. Pengambilan Kolom KDD'99
Sumber: Penulis

b. UNSW-NB15

```python
# Fungsi untuk memuat dataset UNSW-Nb15
def load_unsw_dataset():
    # Sesuaikan path dengan nama file dan path di Google Colab
    file_path = 'UNSW-NB15_1.csv'
    columns = [
        'srcip', 'sport', 'dstip', 'dsport', 'proto', 'state', 'dur', 'sbytes',
        'dbytes', 'sttl', 'dttl', 'sloss', 'dloss', 'service', 'Sload', 'Dload',
        'Spkts', 'Dpkts', 'swin', 'dwin', 'stcpb', 'dtcpb', 'smeansz', 'dmeansz',
        'trans_depth', 'res_bdy_len', 'Sjit', 'Djit', 'Stime', 'Ltime', 'Sintpkt',
        'Dintpkt', 'tcprtt', 'synack', 'ackdat', 'is_sm_ips_ports', 'ct_state_ttl',
        'ct_flw_http_mthd', 'is_ftp_login', 'ct_ftp_cmd', 'ct_srv_src', 'ct_srv_dst',
        'ct_dst_ltm', 'ct_src_ ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm', 'ct_dst_src_ltm',
        'attack_cat', 'Label'
    ]

    # Muat dataset UNSW-Nb15
    df_unsw = pd.read_csv(file_path, names=columns, low_memory=False)
    return df_unsw
```

Gambar 4. Pengambilan Kolom UNSW-NB15
Sumber: Penulis

c. CICIDS2017

```python
# Fungsi untuk memuat dataset CICIDS2017 dari file yang sudah ada di Google Colab
def load_cicids_dataset():
    # Sesuaikan path dengan nama file dan path di Google Colab
    file_path = 'Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv'
    columns = [
        'Flow ID', 'Src IP', 'Src Port', 'Dst IP', 'Dst Port', 'Protocol', 'Timestamp',
        'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets', 'Total Length of Fwd Packets',
        'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min',
        'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max',
        'Bwd Packet Length Min', 'Bwd Packet Length Mean', 'Bwd Packet Length Std',
        'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max',
        'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max',
        'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max',
        'Bwd IAT Min', 'Fwd PSH Flags', 'Bwd PSH Flags', 'Fwd URG Flags', 'Bwd URG Flags',
        'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s',
        'Packet Length Min', 'Packet Length Max', 'Packet Length Mean', 'Packet Length Std',
        'Packet Length Variance', 'FIN Flag Count', 'SYN Flag Count', 'RST Flag Count',
        'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'CWE Flag Count',
        'ECE Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size',
        'Avg Bwd Segment Size', 'Fwd Header Length.1', 'Fwd Avg Bytes/Bulk',
        'Fwd Avg Packets/Bulk', 'Fwd Avg Bulk Rate', 'Bwd Avg Bytes/Bulk',
        'Bwd Avg Packets/Bulk', 'Bwd Avg Bulk Rate', 'Subflow Fwd Packets',
        'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init_Win_bytes_forward',
        'Init_Win_bytes_backward', 'act_data_pkt_fwd', 'min_seg_size_forward', 'Active Mean',
        'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min',
        'Label', 'Attack Type'
    ]
    df_cicids = pd.read_csv(file_path, names=columns, low_memory=False)
    return df_cicids
```

Gambar 5. Pengambilan Kolom CICIDS2017
Sumber: Penulis

Kemudian setelah memuat dataset dilakukan pembersihan dataset dan mengkonversinya dari fitur kategorikal ke numerik (label encoding) dimana untuk dataset KDD'99 (protocol_type, service, flag, dan outcome), UNSW-NB15 (proto, state, dan service), dan CICIDS2017 (flow_id, src_ip, dst_ip, dan timestamp).

```
# Bersihkan nama kolom
df_kdd_cleaned = clean_column_names(df_kdd)

# Konversi fitur kategorikal ke numerik (label encoding) untuk dataset KDD'99
label_encoder_kdd = LabelEncoder()
df_kdd_cleaned['protocol_type'] = label_encoder_kdd.fit_transform(df_kdd_cleaned['protocol_type'])
df_kdd_cleaned['service'] = label_encoder_kdd.fit_transform(df_kdd_cleaned['service'])
df_kdd_cleaned['flag'] = label_encoder_kdd.fit_transform(df_kdd_cleaned['flag'])
df_kdd_cleaned['outcome'] = label_encoder_kdd.fit_transform(df_kdd_cleaned['outcome'])
# Bersihkan nama kolom
df_unsw_cleaned = clean_column_names(df_unsw)

# Konversi fitur kategorikal ke numerik (label encoding) untuk dataset UNSW-NB15
label_encoder_unsw = LabelEncoder()
df_unsw_cleaned['proto'] = label_encoder_unsw.fit_transform(df_unsw_cleaned['proto'])
df_unsw_cleaned['state'] = label_encoder_unsw.fit_transform(df_unsw_cleaned['state'])
df_unsw_cleaned['service'] = label_encoder_unsw.fit_transform(df_unsw_cleaned['service'])
# Bersihkan nama kolom
df_cicids_cleaned = clean_column_names(df_cicids)

# Hapus kolom yang tidak diperlukan atau memiliki nilai unik terlalu banyak
columns_to_exclude = ['Flow_ID', 'Src_IP', 'Dst_IP', 'Timestamp']  # Sesuaikan dengan kolom yang perlu diabaikan
df_cicids_cleaned = df_cicids_cleaned.drop(columns=columns_to_exclude, axis=1)

# Konversi fitur kategorikal ke numerik (label encoding) untuk dataset CICIDS2017
label_encoder_cicids = LabelEncoder()
for column in df_cicids_cleaned.select_dtypes(include='object').columns:
    df_cicids_cleaned[column] = label_encoder_cicids.fit_transform(df_cicids_cleaned[column])
```

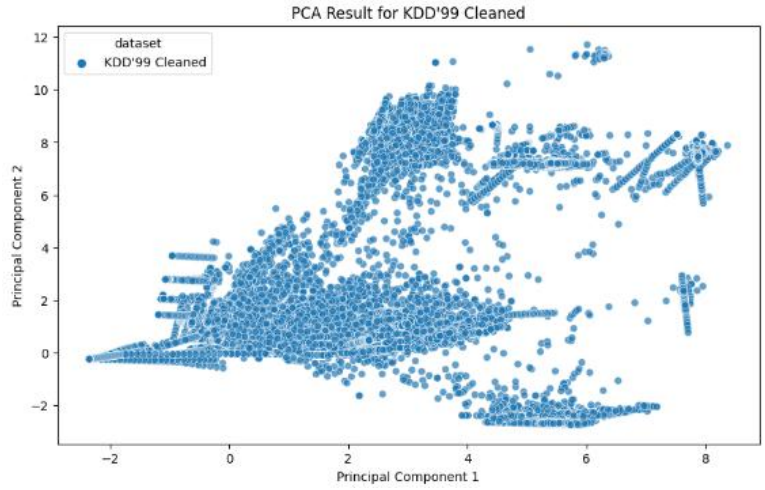
Gambar 6. Pembersihan dan Mengkonversi Dataset
Sumber: Penulis

2. Analisis Principal Component Analysis (PCA)
   Tahap ini fokus pada penerapan metode PCA untuk mengurangi dimensi dataset. PCA digunakan untuk mentransformasi data ke dalam bentuk komponen utama yang mewakili variasi maksimal dalam data. Hasil PCA kemudian divisualisasikan dalam scatter plot dua dimensi untuk masing-masing dataset, memungkinkan pemahaman yang lebih baik tentang struktur dan distribusi data dengan langkah sebagai berikut.

   a. KDD'99

```
# Pilih fitur yang akan digunakan untuk PCA (sesuaikan dengan dataset)
X_kdd_cleaned = df_kdd_cleaned.drop('outcome', axis=1).values

# Lakukan PCA dan visualisasi untuk KDD'99 setelah pembersihan
perform_pca_and_visualization(X_kdd_cleaned, 'KDD\'99 Cleaned')
```



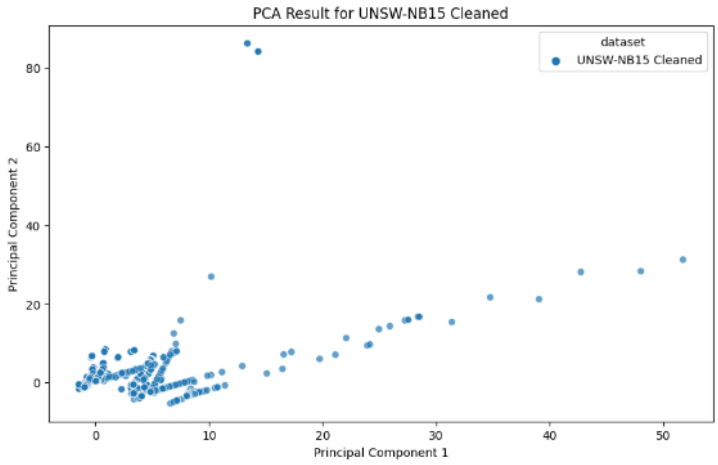
Gambar 7. PCA KDD’99
Sumber: Penulis

Pada dataset diatas dilakukan pemilihan fitur yang akan digunakan untuk PCA dimana untuk dataset KDD’99 memilih fitur outcome. Kemudian melakukan analisis PCA dan menampilkan hasil dalam bentuk visualisasi Scatter Plot.

b. UNSW-NB15

```
# Pilih fitur yang akan digunakan untuk PCA (sesuaikan dengan dataset)
X_unsw_cleaned = df_unsw_cleaned[['dur', 'proto', 'state', 'sbytes', 'dbytes', 'sttl', 'dttl']].values

# Lakukan PCA dan visualisasi untuk UNSW-NB15 setelah pembersihan
perform_pca_and_visualization(X_unsw_cleaned, 'UNSW-NB15 Cleaned')
```



Gambar 8. PCA UNSW-NB15
Sumber: Penulis

Pada dataset diatas dilakukan pemilihan fitur yang akan digunakan untuk PCA dimana untuk dataset UNSW-NB15 memilih fitur dur, proto, state, sbytes, dbytes, sttl, dan dttl. Kemudian melakukan analisis PCA dan menampilkan hasil dalam bentuk visualisasi Scatter Plot.

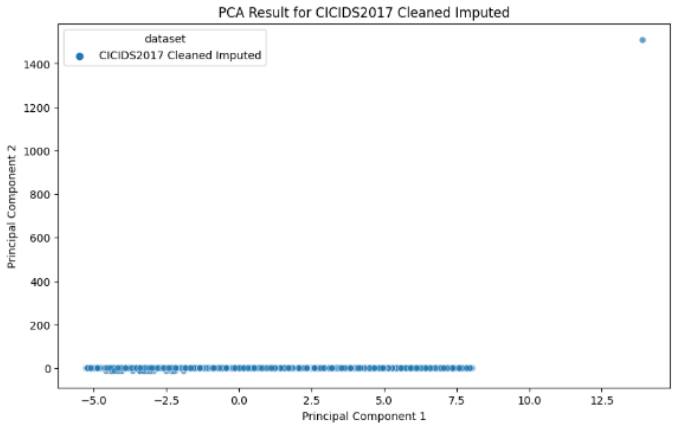
c. CICIDS2017

```
# Konfigurasi imputer untuk mempertahankan seluruh kolom
imputer = SimpleImputer(strategy='mean', add_indicator=True)

# Terapkan imputasi pada DataFrame
df_cicids_cleaned_imputed = pd.DataFrame(imputer.fit_transform(df_cicids_cleaned), columns=df_cicids_cleaned.columns)

# Pilih fitur yang akan digunakan untuk PCA (sesuaikan dengan dataset)
X_cicids_cleaned = df_cicids_cleaned_imputed.drop('Label', axis=1).values

# Lakukan PCA dan visualisasi untuk CICIDS2017 setelah pembersihan dan imputasi
perform_pca_and_visualization(X_cicids_cleaned, 'CICIDS2017 Cleaned Imputed')
```

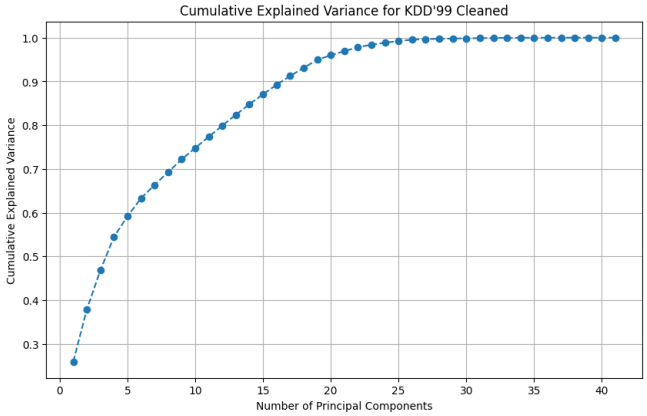


Gambar 9. PCA CICIDS2017
Sumber: Penulis

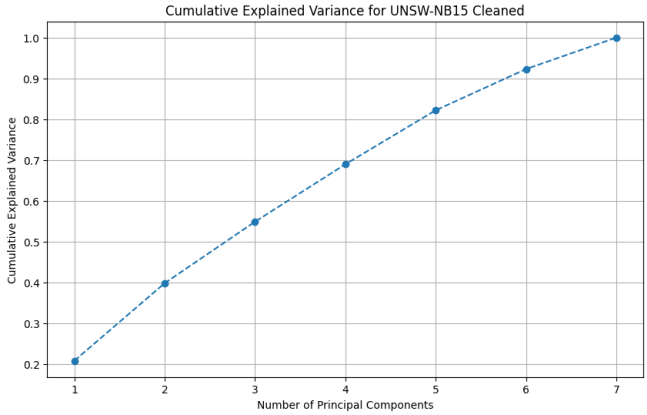
Pada dataset diatas dilakukan pemilihan fitur yang akan digunakan untuk PCA dimana untuk dataset CICIDS2017 memilih fitur Label. Kemudian melakukan analisis PCA dan menampilkan hasil dalam bentuk visualisasi Scatter Plot. Untuk dataset ini diharuskan melakukan Imputasi Dataframe

3. Analisis Hasil CPA

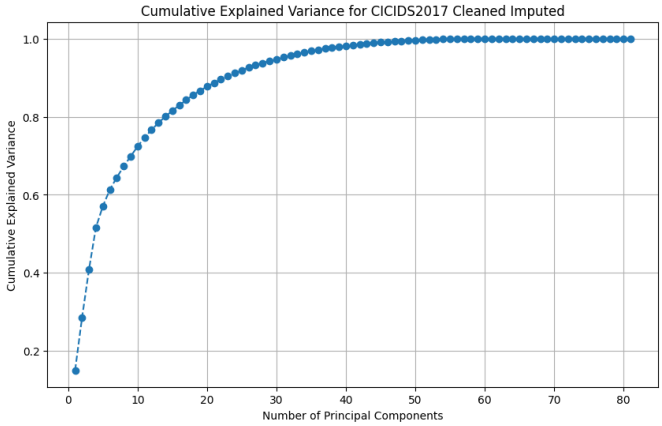
Pada tahap ini, dilakukan analisis lebih lanjut terkait hasil PCA. Varians yang dijelaskan oleh setiap komponen utama dievaluasi, dan kumulatif varians yang dijelaskan divisualisasikan. Informasi ini membantu dalam pemilihan jumlah komponen utama yang sesuai untuk menjaga sebagian besar variasi data dengan hasil sebagai berikut.



Gambar 10. Hasil Analisis CPA KDD'99
Sumber: Penulis



Gambar 11. Hasil Analisis CPA UNSW-NB15
Sumber: Penulis

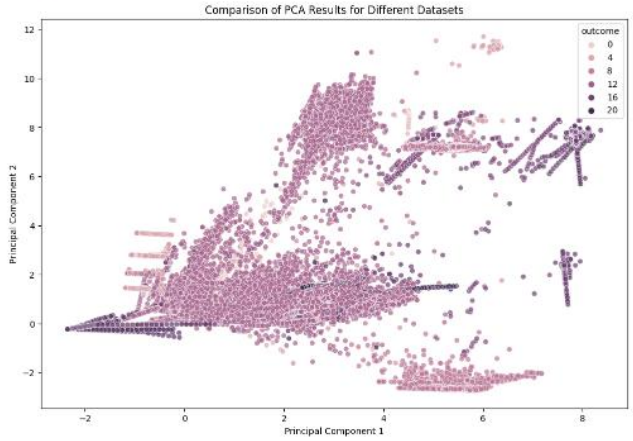


Gambar 12. Hasil Analisis CPA CICIDS2017
Sumber: Penulis

4. Gabungan Hasil CPA dari Ketiga Dataset
   Hasil PCA dari ketiga dataset digabungkan menjadi satu dataframe. Langkah ini memungkinkan perbandingan langsung antara distribusi data dari berbagai sumber. Scatter plot digunakan untuk memvisualisasikan hasil PCA dari gabungan dataset, menyoroti perbedaan dan pola yang mungkin muncul dengan langkah sebagai berikut.

```
# Gabungkan hasil PCA dari ketiga dataset
df_combined = pd.concat([
    pd.concat([df_kdd_cleaned[['outcome']].reset_index(drop=True), pd.DataFrame(X_kdd_pca[:, :2], columns=['PC1', 'PC2'])], axis=1),
    pd.concat([df_unsw_cleaned[['Label']].reset_index(drop=True), pd.DataFrame(X_unsw_pca[:, :2], columns=['PC1', 'PC2'])], axis=1),
    pd.concat([df_cicids_cleaned[['Label']].reset_index(drop=True), pd.DataFrame(X_cicids_pca[:, :2], columns=['PC1', 'PC2'])], axis=1)
], ignore_index=True)

# Visualisasi Scatter Plot untuk Membandingkan Hasil PCA
plt.figure(figsize=(12, 8))
sns.scatterplot(x='PC1', y='PC2', hue=df_combined['outcome'], data=df_combined, alpha=0.7)
plt.title('Comparison of PCA Results for Different Datasets')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```



Gambar 13. Menggabungkan Ketiga CPA Dataset
Sumber: Penulis

5. Evaluasi Model dengan Logistic Regression

   Dilakukan pembagian dataset menjadi data latih dan data uji untuk setiap dataset. Model Logistic Regression dilatih pada data latih dan dievaluasi pada data uji. Akurasi model menjadi metrik evaluasi utama untuk memahami seberapa baik model berkinerja dalam mengklasifikasikan data. Kemudian, hasil akurasi dari model Logistic Regression untuk masing-masing dataset dievaluasi. Akurasi memberikan gambaran tentang seberapa baik model dapat memprediksi kelas target. Hasil ini memberikan pemahaman tentang keefektifan model pada dataset yang berbeda.

   a. KDD'99

```
# Bagi dataset KDD'99
X_train_kdd, X_test_kdd, y_train_kdd, y_test_kdd = train_test_split(
    X_kdd_cleaned, df_kdd_cleaned['outcome'], test_size=0.2, random_state=42
)

# Latih model pada dataset KDD'99
model_kdd = LogisticRegression()
model_kdd.fit(X_train_kdd, y_train_kdd)

# Evaluasi model pada data uji KDD'99
y_pred_kdd = model_kdd.predict(X_test_kdd)
accuracy_kdd = accuracy_score(y_test_kdd, y_pred_kdd)
print(f"Accuracy for KDD'99: {accuracy_kdd:.2%}")
```

```
Accuracy for KDD'99: 94.68%
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Gambar 14. Evaluasi Model KDD'99
Sumber: Penulis

   b. UNSW-NB15

```
# Bagi dataset UNSW-NB15
X_train_unsw, X_test_unsw, y_train_unsw, y_test_unsw = train_test_split(
    X_unsw_cleaned, df_unsw_cleaned['Label'], test_size=0.2, random_state=42
)

# Latih model pada dataset UNSW-NB15
model_unsw = LogisticRegression()
model_unsw.fit(X_train_unsw, y_train_unsw)

# Evaluasi model pada data uji UNSW-NB15
y_pred_unsw = model_unsw.predict(X_test_unsw)
accuracy_unsw = accuracy_score(y_test_unsw, y_pred_unsw)
print(f"Accuracy for UNSW-NB15: {accuracy_unsw:.2%}")
```

```
Accuracy for UNSW-NB15: 98.87%
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Gambar 14. Evaluasi Model UNSW-NB15
Sumber: Penulis

c. CICIDS2017

```
# Bagi dataset CICIDS2017
X_train_cicids, X_test_cicids, y_train_cicids, y_test_cicids = train_test_split(
    X_cicids_cleaned, df_cicids_cleaned['Label'], test_size=0.2, random_state=42
)

# Latih model pada dataset CICIDS2017
model_cicids = LogisticRegression()
model_cicids.fit(X_train_cicids, y_train_cicids)

# Evaluasi model pada data uji CICIDS2017
y_pred_cicids = model_cicids.predict(X_test_cicids)
accuracy_cicids = accuracy_score(y_test_cicids, y_pred_cicids)
print(f"Accuracy for CICIDS2017: {accuracy_cicids:.2%}")
```

```
Accuracy for CICIDS2017: 46,59%
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Gambar 14. Evaluasi Model CICIDS2017
Sumber: Penulis

Setelah dilakukan evaluasi model dengan Logistic Regression, maka hasil akurasi yang didapatkan pada ketiga dataset tersebut, yaitu:
KDD'99 hasil akurasi 94,68%
UNSW-NB15 hasil akurasi 98.87%
CICIDS2017 hasil akurasi 46,59%

KESIMPULAN

Berdasarkan pembahasan yang telah dilakukan, maka dapat disimpulkan penelitian ini menunjukkan bahwa penggunaan metode PCA dalam preprocessing dan pengurangan dimensi dapat memberikan hasil yang baik dalam konteks deteksi intrusi jaringan. Meskipun terdapat perbedaan dalam karakteristik dataset, evaluasi model dengan Logistic Regression memberikan gambaran tentang keefektifan deteksi intrusi pada masing-masing dataset. Dengan demikian, hasil ini dapat membantu peneliti dan praktisi dalam memilih pendekatan yang sesuai untuk meningkatkan keamanan jaringan berbasis deteksi intrusi. Hasil evaluasi model menunjukkan akurasi yang baik untuk KDD'99 (94,68%) dan UNSW-NB15 (98,87%), sementara CICIDS2017 memiliki akurasi yang lebih rendah sebesar 46,59%. Hal ini mengindikasikan perbedaan dalam karakteristik dataset dan kompleksitas deteksi intrusi.

DAFTAR PUSTAKA

Farnaaz, N., & Akhil, J. (2016). Random Forest Modeling for Network Intrusion Detection System. *Procedia Computer Science*, *89*, 213–217. https://doi.org/10.1016/j.procs.2016.06.047

Liu, L., Wang, P., Lin, J., & Liu, L. (2020). Intrusion Detection of Imbalanced Network Traffic Based on Machine Learning and Deep Learning. *IEEE Access*, *PP*, 1. https://doi.org/10.1109/ACCESS.2020.3048198

Moustafa, N., & Slay, J. (2015). *UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)*. https://doi.org/10.1109/MilCIS.2015.7348942

Panigrahi, R., & Borah, S. (2018). A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *International Journal of Engineering & Technology*, *7*, 479–482.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. (2009). A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium. Computational Intelligence for Security and Defense Applications, CISDA*, *2*. https://doi.org/10.1109/CISDA.2009.5356528