

Intrusion Detection

Erwin Erikson
Faculty of Information Technology
Institut Teknologi Batam
Batam, Indonesia
1822003@student.iteba.ac.id

Muhammad Al Imron
Faculty of Information Technology
Institut Teknologi Batam
Batam, Indonesia
1822007@student.iteba.ac.id

Farhan Ghulam Hadi Saputra
Faculty of Information Technology
Institut Teknologi Batam
Batam, Indonesia
1822014@student.iteba.ac.id

Abstract—Meningkatnya perkembangan internet tidak terlepas dari serangan seperti malware infection. Intrusion Detection System (IDS) adalah masalah nonlinier, rumit dan berhubungan dengan data lalu lintas jaringan. IDS mencoba untuk mengidentifikasi dan memberi tahu aktivitas pengguna sebagai anomali normal. Untuk mendeteksi berbagai serangan jaringan dapat dilatih dengan melakukan percobaan menggunakan dataset NSL-KDD dengan beberapa algoritma yaitu Random Forest, K-Neighbors, SVM dan Ensemble Learning.

Keywords—intrusion detection, Random Forest, K-Neighbors, SVM, Ensemble Learning, NSL-KDD dataset.

I. PENDAHULUAN

Pengguna internet yang terus meningkat dari tahun ke tahun tidak terlepas dari serangan yang timbul dari teknologi jaringan seperti serangan *malware infection*. Oleh karena itu, diperlukan keamanan dalam sistem komputer untuk mencegah dari serangan.

IDS (*Intrusion Detection System*) merupakan sebuah aplikasi yang mampu mencatat kegiatan dalam suatu jaringan dan menganalisa paket-paket yang dikirim melalui lalu lintas jaringan secara *realtime*. Tujuan dari sistem ini yaitu mengawasi jika terjadi penetrasi ke dalam sistem, mengawasi *traffic* yang terjadi pada jaringan, mendeteksi *anomaly* terjadinya penyimpangan dari sistem yang normal atau tingkah laku user.

Dalam melakukan deteksi serangan, dapat digunakan beberapa algoritma yaitu Random Forest, K-Neighbors, SVM dan Ensemble Learning. Dan tujuan dari penelitian ini adalah untuk membandingkan performa dari masing-masing algoritma.

II. PENJELASAN TEORI

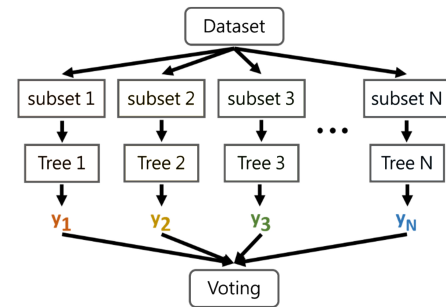
A. Random Forest

Hutan acak (*Random Forest*) adalah kumpulan pohon keputusan yang digunakan untuk meningkatkan akurasi, biasanya dilatih dengan metode "*bagging*". Ide umum dari metode *bagging* adalah bahwa kombinasi model pembelajaran meningkatkan hasil secara keseluruhan.

Keuntungan *Random Forest* adalah sebagai berikut [1]:

- 1) Hutan yang dihasilkan dapat disimpan untuk referensi di masa mendatang.
- 2) Hutan acak mengatasi masalah penyesuaian.
- 3) Dalam akurasi RF dan kepentingan variabel secara otomatis dihasilkan.

Flowchart dari proses pemodelan algoritma *Random Forest* dapat dilihat pada "Gambar. 1".



Gambar 1: Flowchart Algoritma Random Forest

B. K-Nearest Neighbors

K-nearest neighbors (knn) adalah algoritma yang berfungsi untuk melakukan klasifikasi suatu data berdasarkan data pembelajaran (*train data sets*), yang diambil dari k tetangga terdekatnya (*nearest neighbors*), dengan k merupakan banyaknya tetangga terdekat [2]. Beberapa formula yang digunakan adalah:

• Euclidean Distance

Untuk mendefinisikan jarak antara dua titik yaitu titik pada data training (x) dan titik pada data testing (y), maka digunakan rumus *Euclidean* [3], yaitu:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

dimana:

d = jarak antara 2 titik

x = data uji

y = data latih

i = merepresentasikan nilai atribut

n = merupakan dimensi atribut.

• City Block Distance

City Block Distance umumnya dihitung antara 2 koordinat objek yang berpasangan. Ini adalah penjumlahan dari perbedaan absolut antara 2 koordinat. *City Block Distance* 2-titik a dan b dengan dimensi k dihitung secara matematis menggunakan rumus berikut ini:

$$d_{ij} = \sum_{i=1}^k |a_i - b_i|$$

- *Manhattan Distance*

Manhattan Distance merupakan salah satu pengukuran yang paling banyak digunakan meliputi pengurangan perbedaan kuadrat dengan menjumlahkan perbedaan *absolute* dari variabel-variabel. Fungsi ini hanya akan menjumlahkan selisih nilai x dan y dari dua buah titik.

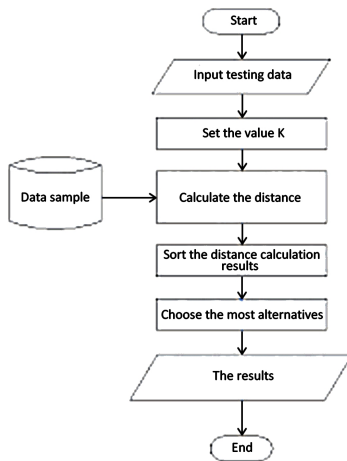
- *Minkowski Distance*

Minkowski Distance adalah metrik dalam ruang vektor bernorma yang dapat dianggap sebagai generalisasi dari kedua jarak *Euclidean* dan jarak *Manhattan*. Jarak *Minkowski* antara dua variabel X dan Y didefinisikan sebagai:

$$d = \left(\sum_{i=1}^n |X_i - Y_i|^p \right)^{1/p}$$

Kasus di mana $p = 1$ setara dengan jarak *Manhattan* dan kasus di mana $p = 2$ setara dengan jarak *Euclidean*.

Flowchart dari proses pemodelan algoritma *K-nearest neighbors* dapat dilihat pada “Gambar. 2” [4].



Gambar 2: Flowchart Algoritma K-nearest neighbors

C. SVM

Teori SVM berasal dari statistik dan prinsip dasar SVM adalah menemukan *hyperplane* linier yang optimal dalam ruang fitur yang secara maksimal memisahkan dua kelas target [5].

Dalam kaitannya dengan fungsi kernel, fungsi diskriminan mengambil bentuk berikut:

$$f(x) = \sum_i \alpha_i k(x, x_i) + b$$

Dalam pekerjaan ini, kernel *Gaussian* telah digunakan untuk membangun pengklasifikasi SVM.

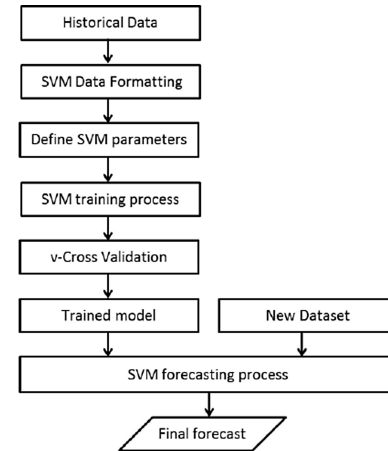
Gaussian kernel:

$$K(x_i, x_j) = \exp \left(-\frac{\|x_i - x_j\|^2}{2\sigma} \right)$$

dimana σ adalah lebar fungsi.

Fungsi kernel dan parameternya harus dipilih untuk membangun pengklasifikasi SVM. Melatih SVM menemukan *hyperplane* margin besar, yaitu menetapkan parameter α .

Flowchart dari proses pemodelan algoritma SVM dapat dilihat pada “Gambar. 3” [6].

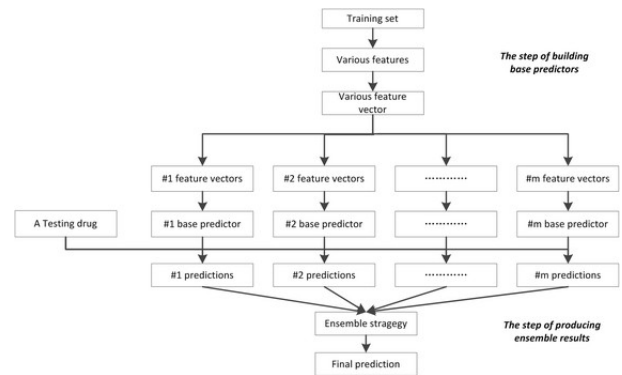


Gambar 3: Flowchart Algoritma SVM

D. Ensemble Learning

Metode ini adalah menggabungkan beberapa fitur dengan pembelajaran *Ensemble*.

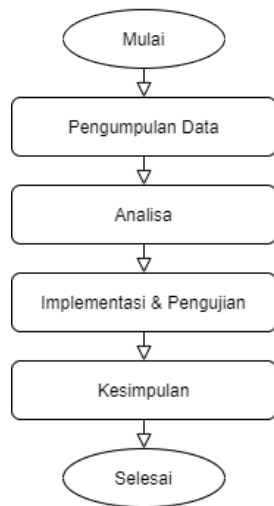
Flowchart dari proses pemodelan algoritma *Ensemble Learning* dapat dilihat pada “Gambar. 4” [7].



Gambar 4: Flowchart Algoritma Ensemble Learning

III. METODOLOGI

Metodologi adalah tahapan yang akan dilakukan dalam melakukan penelitian agar dapat memenuhi tujuan sesuai dengan yang diharapkan. Tahapan penelitian yang akan dilakukan dapat dilihat pada “Gambar. 5”.



Gambar 5: Tahapan Penelitian

Tahapan pada penelitian ini dapat dijelaskan sebagai berikut:

Pengumpulan Data

Pengumpulan data yang dilakukan dengan membaca dan mempelajari penelitian sebelumnya yang berhubungan dengan IDS.

Analisa

Pada tahap ini adalah menganalisa data yaitu data latih yang digunakan untuk standarisasi melakukan pengujian, dan data uji yang digunakan untuk mengetes penilaian yang dihasilkan dari data latih.

Tahap ini juga menganalisa metode yang digunakan dalam penelitian yang berkaitan dengan sistem yang digunakan.

Implementasi dan Pengujian

Proses implementasi adalah merealisasikan aplikasi IDS sesuai dengan dengan bahasa pemrograman yang digunakan yaitu Phytion menggunakan JupyterLab.

Tahap pengujian adalah tahap yang dilakukan untuk menguji masing-masing metode yang digunakan dalam penelitian dengan tujuan untuk mengetahui perbandingan performa dari setiap algoritma.

Kesimpulan

Merupakan tahap penentuan kesimpulan terhadap hasil pengujian yang telah dilakukan.

IV. HASIL DAN PEMBAHASAN

Untuk melakukan pelatihan/pengujian digunakan dataset NSL-KDD dimana NSL_KDD_Train sebagai data latih dan NSL_KDD_Test sebagai data uji seperti pada “Gambar. 6” [8].

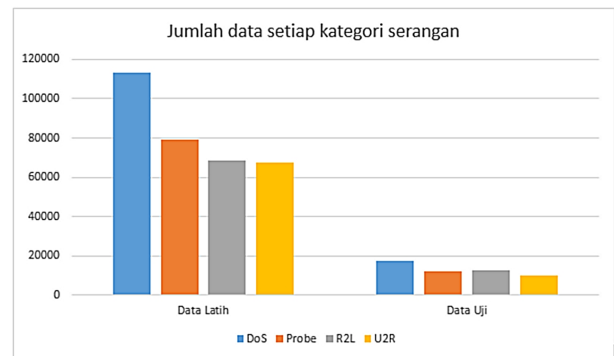
```

[1]: import pandas as pd
import numpy as np
import sys
import sklearn
import random

[2]: train_url = "https://raw.githubusercontent.com/neretorgiu/NSL-KDD-Network-Intrusion-Detection/master/NSL_KDD_Train.csv"
test_url = "https://raw.githubusercontent.com/neretorgiu/NSL-KDD-Network-Intrusion-Detection/master/NSL_KDD_Test.csv"
  
```

Gambar 6: Data Latih dan Data Uji

Dari dataset tersebut dibagi untuk setiap kategori serangan yaitu 0 = Normal, 1 = DoS, 2 = Probe, 3 = R2L, 4 = U2R. Hasil pembagian setiap kategori serangan dari data latih dan data uji dapat dilihat pada “Gambar. 7”.



Gambar 7: Jumlah Data Setiap Kategori Serangan

Untuk menghitung precision dan recall dengan cepat dari setiap serangan dilakukan tahap *Prediction* dan *Evaluation (validation)* dengan *confusion matrix* seperti pada “Gambar. 8”.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Gambar 8: Confusion Matrix

Hasil *Prediction* dan *Evaluation (validation)* metode *Random Forest* semua fitur dapat dilihat pada “Gambar. 9”.

DoS			Probe			R2L			U2R		
Predicted attacks	0	1	Predicted attacks	0	2	Predicted attacks	0		Predicted attacks	0	
Actual attacks	0	9686 25	Actual attacks	0	9364 347	Actual attacks	0	9711	Actual attacks	0	9711
	1	7447 13		2	1076 1345		3	2885		4	67

Gambar 9: Confusion Matrix Random Forest All Fitur

Hasil *Prediction* dan *Evaluation (validation)* metode *Random Forest* 13 fitur dapat dilihat pada “Gambar. 10”.

DoS			Probe			R2L			U2R		
Predicted attacks	0	1	Predicted attacks	0	2	Predicted attacks	0	3	Predicted attacks	0	4
Actual attacks	0	9162 549	Actual attacks	0	9413 298	Actual attacks	0	9693 18	Actual attacks	0	9711 0
	1	2136 5324		2	975 1446		3	2880 5		4	60 7

Gambar 10: Confusion Matrix Random Forest 13 Fitur

Hasil *Prediction* dan *Evaluation (validation)* metode *K-Neighbors* dapat dilihat pada “Gambar. 11”.

DoS			Probe			R2L			U2R		
Predicted attacks	0	1	Predicted attacks	0	2	Predicted attacks	0	3	Predicted attacks	0	4
Actual attacks	0	9422 289	Actual attacks	0	9437 274	Actual attacks	0	9706 5	Actual attacks	0	9711 0
	1	1573 5887		2	1272 1149		3	2883 2		4	65 2

Gambar 11: Confusion Matrix K-Neighbors

Hasil *Prediction* dan *Evaluation (validation)* metode SVM dapat dilihat pada “Gambar. 12”.

DoS	Probe	R2L	U2R
Predicted attacks 0 1	Predicted attacks 0 2	Predicted attacks 0 3	Predicted attacks 0 4
Actual attacks	Actual attacks	Actual attacks	Actual attacks
0 9455 256	0 9576 135	0 9639 72	0 9710 1
1 1359 6101	2 1285 1136	3 2737 148	4 67 0

Gambar 12: Confusion Matrix SVM

Hasil *Prediction* dan *Evaluation (validation)* metode *Ensemble Learning* dapat dilihat pada “Gambar. 13”.

DoS	Probe	R2L	U2R
Predicted attacks 0 1	Predicted attacks 0 2	Predicted attacks 0 3	Predicted attacks 0
Actual attacks	Actual attacks	Actual attacks	Actual attacks
0 9607 104	0 9550 161	0 9711 0	0 9711
1 1919 5541	2 1294 1127	3 2884 1	4 67

Gambar 13: Confusion Matrix Ensemble Learning

Dengan *confusion matrix* dilakukan penghitungan *Accuracy*, *Precision*, *Recall*, dan *F-measure* dari nilai masing-masing dalam matriks dengan menerapkan persamaan berikut:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - measure = 2 * \frac{precision * recall}{precision + recall}$$

dimana:

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

Dari hasil evaluasi kinerja dari masing-masing model atau algoritma, dapat dilihat pada tabel-tabel berikut:

Pengujian menggunakan algoritma *Random Forest* untuk semua fitur dapat dilihat pada “Tabel. I”.

Tabel I: Algoritma Random Forest untuk semua Fitur

	Accuracy	Precision	Recall	F-measure
DoS	0.99796	0.99906	0.99651	0.99765
Probe	0.99670	0.99675	0.99262	0.99469
R2L	0.99775	0.94964	0.82014	0.87775
U2R	0.98047	0.97293	0.96844	0.97290

Pengujian menggunakan algoritma *Random Forest* untuk 13 fitur dapat dilihat pada “Tabel. II”.

Tabel II: Algoritma Random Forest untuk 13 Fitur

	Accuracy	Precision	Recall	F-measure
DoS	0.99796	0.99839	0.99651	0.99718
Probe	0.99382	0.98973	0.98668	0.98758
R2L	0.97856	0.97280	0.96525	0.96838
U2R	0.99693	0.96256	0.83183	0.90644

Pengujian menggunakan algoritma *K-Neighbors* dapat dilihat pada “Tabel. III”.

Tabel III: Algoritma K-Neighbors

	Accuracy	Precision	Recall	F-measure
DoS	0.99715	0.99678	0.99665	0.99672
Probe	0.99077	0.98606	0.98508	0.98553
R2L	0.96705	0.95265	0.95439	0.95344
U2R	0.99703	0.93143	0.85073	0.87831

Pengujian menggunakan algoritma SVM dapat dilihat pada “Tabel. IV”.

Tabel IV: Algoritma SVM

	Accuracy	Precision	Recall	F-measure
DoS	0.99371	0.99107	0.99450	0.99278
Probe	0.98450	0.96907	0.98365	0.97613
R2L	0.96793	0.94854	0.96264	0.95529
U2R	0.99632	0.91056	0.82909	0.84869

Pengujian menggunakan algoritma *Ensemble Learning* dapat dilihat pada “Tabel. V”.

Tabel V: Algoritma Ensemble Learning

	Accuracy	Precision	Recall	F-measure
DoS	0.99808	0.99852	0.99718	0.99772
Probe	0.99275	0.98765	0.98953	0.98841
R2L	0.97158	0.95838	0.96409	0.96079
U2R	0.99744	0.94270	0.88758	0.91119

V. KESIMPULAN

Dalam penelitian ini, kami membandingkan beberapa model untuk sistem deteksi trusi menggunakan *Random Forest*, *K-Neighbors*, *Support Vector Machine*, dan *Ensemble Learning* dengan ketiga model diatas. Performa keempat pendekatan ini telah diamati berdasarkan *accuracy*, *precision*, *recall*, dan *f-measure (F₁-score)*.

Dari hasil pengujian dari masing-masing algoritma yang ada pada tabel, menunjukkan kemampuan klasifikasi algoritma *Ensemble Learning* lebih tinggi tingkat akurasi dan ketepatan.

Hasil penelitian ini sangat berguna untuk penelitian masa depan dengan cara memaksimalkan tingkat kinerja serta meminimalkan tingkat *false negative*.

REFERENCES

- [1] N. Farnaaz and M. Jabbar, "Random forest modeling for network intrusion detection system," *Procedia Computer Science*, vol. 89, pp. 213–217, 2016.
- [2] Y. Liao and R. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection," *Computers and Security*, vol. 21, pp. 439–448, 10 2002.
- [3] N. Nurhadi, *Aplikasi Intelligence Intrusion Detection System (IIDS) Dengan Menggunakan Metode K-Nearest Neighbor Untuk Mendeteksi Serangan Pada Jaringan*. PhD thesis, Universitas Islam Negeri Sultan Syarif Kasim Riau, 2017.
- [4] Z. Lubis, P. Sihombing, and H. Mawengkang, "Optimization of k value at the k-nn algorithm in clustering using the expectation maximization algorithm," in *IOP Conference Series: Materials Science and Engineering*, p. 012133, IOP Publishing, 2020.
- [5] M. A. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (ids)," *Journal of Intelligent Learning Systems and Applications*, vol. 06, pp. 45–52, 01 2014.
- [6] E. Xydas, C. Marmaras, L. Cipcigan, A. Sani Hassan, and N. Jenkins, "Forecasting electric vehicle charging demand using support vector machines," *Proceedings of the Universities Power Engineering Conference*, pp. 1–6, 09 2013.
- [7] W. Zhang, F. Liu, L. Longqiang, and J. Zhang, "Predicting drug side effects by multi-label learning and ensemble learning," *BMC Bioinformatics*, vol. 16, 11 2015.
- [8] Mamcose, "Nsl-kdd-network-intrusion-detection," 2019.