Analisis Deteksi *Malware* pada Aplikasi Android *Fintech* berdasarkan Permissions dengan menggunakan Naive Bayes dan Random Forest

Bangkit Prasetyo[1], Vera Suryani[2], Dhika Rizki Anbiya[3]

[1,2,3] Universitas Telkom, Bandung
[1]bprasetyo@students.telkomuniversity.ac.id, [2]verasuryani@telkomuniversity.ac.id, [3]danbiya@telkomuniversity.ac.id

Abstrak

Fintech merupakan jenis perusahaan di bidang jasa keuangan yang memberikan berbagai kemudahan seperti mengelola pajak secara otomatis, memperkecil risiko ketidakpatuhan pajak dan masih banyak lagi. Aplikasi fintech memiliki banyak kemudahan tetapi tidak semua aplikasi fintech dapat dipercaya aplikasi fintech akan meminta akses izin yang tidak relevan seperti akses kontak, mikrofon, lokasi dan akses lainnya yang digunakan sebagai syarat konfirmasi kepada calon peminjam. Permission merupakan salah satu fitur yang ada pada android yang mengelola akses izin aplikasi semakin banyak permission yang tidak berhubungan dengan kebutuhan aplikasi maka kemungkinan besar bahwa aplikasi tersebut terdapat malware. Pada penelitian ini akan mendeteksi malware pada aplikasi android fintech berdasarkan permission dengan menggunakan metode klasifikasi naive bayes dan random forest Penelitian ini menggunakan 160 dataset dari sampel aplikasi fintech. Hasil yang didapatkan random forest memiliki akurasi yang lebih tinggi yaitu sebesar 80,1% dibandingkan dengan naive bayes yang mendapatkan akurasi sebesar 78%.

Kata kunci : Aplikasi Fintech, Permission, Analisis malware, Naive bayes, Random Forest

Abstract

Fintech is a type of company in the field of financial services that provides various conveniences such as managing taxes automatically, minimizing the risk of tax non-compliance and still bayak again. Fintech applications have many conveniences but not all fintech applications can be trusted fintech applications will ask for access to irrelevant permissions such as contact access, microphone, location and other access used as a condition of confirmation to prospective borrowers. Permission is one of the features that exist on android that manages application permission access the more permission that is not related to the needs of the application then most likely that the application has malware. This study will detect malware in fintech android applications based on permission using naïve bayes classification method and random forest This study uses 160 datasets from a sample of fintech applications. The results obtained by random forest have a higher accuracy of 80.1% compared to naïve bayes who get accuracy of 78%.

Keywords: Fintech Apps, Permission, Malware analysis, Naïve bayers, Random Forest

1. Pendahuluan

Latar Belakang

Fintech adalah singkatan dari *financial technology* merupakan jenis perusahaan di bidang jasa keuangan yang digabungkan dengan teknologi. *Fintech* diubah menjadi sebuah aplikasi android untuk mempermudah dalam menghemat waktu yang dapat mengelola pajak secara otomatis, memperkecil risiko ketidakpatuhan pajak, menerima pembayaran lebih cepat dan banyak kemudahaan lainnya [15]. Aplikasi *fintech* memiliki banyak kemudahan tetapi tidak semua aplikasi *fintech* dapat dipercaya terbukti pada september 2020 ditemukan aplikasi *fintech* yang sengaja memanfaatkan kesulitan keuangan sebagian masyarakat di masa pandemi ini. Pinjaman dari *fintech* tersebut mengenakan suku bunga yang tinggi dan persyaratan pinjaman jangka waktu pendek serta meminta semua akses data di telepon genggam, yang digunakan untuk mengintimidasi saat penagihan [16]. Aplikasi *fintech* akan meminta akses izin yang tidak relevan seperti akses kontak, mikrofon, lokasi dan masih banyak lagi di ponsel pengguna sebagai cara konfirmasi data calon peminjam. Bahkan terdapat penggunaan aplikasi *fintech* yang memberikan akses terhadap data di ponsel dapat membuat riwayat transaksi bocor sehingga dapat digunakan pihak yang tidak bertanggung jawab [17].

Malware berasal dari kata *malicious* dan *software* dapat diartikan perangkat lunak yang digunakan untuk melakukan perusakan sistem, pencurian atau pengumpulan informasi, hingga mendapatkan akses terhadap suatu sistem [18]. Terdapat beberapa cara penyebaran *malware* seperti email *phising*, serangan rekayasa sosial, dan *downloader*. Tujuan dari penyebaran *malware* adalah untuk pencurian data rahasia, pengumpulan informasi seperti password dan email, serta *spamming* [18]. Analisis *malware* adalah sebuah metode untuk mengidentifikasi *malware* atau dapat diartikan suatu aktivitas yang dilakukan untuk mendeteksi dan mengamati perilaku *software* ketika dieksekusi serta metode pencegahan paling efektif terhadap *malware*. Penelitian mengenai deteksi *malware*

telah banyak dilakukan oleh beberapa orang atau instansi di dunia umumnya dalam deteksi *malware* dapat menggunakan pendekatan tradisional seperti antivirus, namun menurut [1] dalam deteksi *malware* pada perangkat android dengan menggunakan pendekatan tradisional seperti antivirus tidaklah efektif dikarenakan *malware* di setiap tahunya selalu berbeda dan terus berkembang sehingga membutuhkan pembaruan database pada antivirus. Untuk mengatasi hal tersebut maka dibutuhkan *machine learning* untuk deteksi *malware* pada aplikasi android.

Dalam penelitian ini sistem yang dibangun bertujuan untuk deteksi *malware* aplikasi *fintech* berdasarkan permissionnya untuk dataset diambil dari ekstraksi fitur permission dari 160 sampel aplikasi *fintech* dengan 80 sampel aplikasi *fintech* yang terdaftar di OJK [19] dan 80 sampel aplikasi *fintech* yang tidak terdaftar di OJK [20]. Untuk metode klasifikasinya menggunakan algoritma naïve bayes dan random forest serta K-*fold Cross Validation* yang tersedia di *tool* WEKA [21].

Tujuan penelitian ini adalah menganalisa deteksi *malware* pada aplikasi android *fintech* berdasarkan permission dengan menggunakan klasifikasi naïve bayes dan random forest. Bagian penelitian selanjutnya yaitu bagian 2 studi terkait yang membahas penelitian sebelumnya dan gambaran tentang *malware* pada aplikasi android berdasarkan permission. Bagian 3 membahas perancangan sistem yang dibangun berdasarkan teori yang terkait. Pada bagian 4 membahas hasil dan analisanya. Selanjutnya bagian 5 membahas kesimpulan dan saran untuk penelitian selanjutnya.

2. Studi Terkait

2.1 Penelitian Terkait

Terdapat beberapa penelitian terkait yang melakukan deteksi *malware* pada aplikasi android berdasarkan permission dengan menggunakan *machine learning*. Mahindru & Singh [2] mengusulkan deteksi *malware* pada android berdasarkan permission dengan menggunakan beberapa algoritma *machine learning* yang tersedia di *tool* WEKA. Hasilnya algoritma random forest, naive bayes, j48 dan simple logistic memiliki akurasi yang sebanding. Di paper tersebut juga menjelaskan dalam menggunakan 10 *fold Cross Validation* untuk pengklasifikasiannya.Verma & Muttoo[3] dalam penelitianya menganalisa permission untuk deteksi *malware* dengan menggunakan metode *hybrid* dengan menggunakan *machine learning* yang tersedia di *tool* WEKA. Hasilnya algoritma *machine learning* dalam deteksi *malware* memiliki persentase kesalahan 6 persen. Di paper tersebut juga dijelaskan ekstraksi fitur pada permission android dengan menggunakan salah satu bagian dari *reverse engineering*. Khariwal & Arora [4] mengusulkan deteksi *malware* pada android berdasarkan permission dengan menggunakan beberapa algoritma *machine learning* seperti SVM, naive bayes dan random forest. Hasil yang didapat random forest memiliki akurasi paling tinggi dalam deteksi *malware* android berdasarkan permission. Li., & Yao [5] mengusulkan metode deteksi malware android berdasarkan permission di file AndroidManifest dengan menggunakan klasifikasi naive bayes. paper tersebut juga membandingkan deteksi dengan pendekatan tradisional. Hasilnya metode yang diusulkan memiliki tingkat deteksi yang lebih tinggi dan tingkat kesalahan yang lebih rendah.

2.2 Arsitektur Aplikasi Android

APK adalah format yang digunakan untuk mengemas aplikasi android yang bisa didapat dari Google Play Store atau pasar pihak ketiga. File APK pada dasarnya adalah file ZIP yang dapat diganti format namanya dan dapat diekstraksi kembali isinya [6]. Tabel 1 menunjukkan arsitektur dasar aplikasi android

Tabel 2.1 Arsitektur Aplikasi Android

Entry	Deskripsi
AndroidManifest.xml	File manifes dalam format biner XML untuk mengatur sumber daya izin.
classes.dex	Kode aplikasi yang dikompilasi dalam format dexnya.
resources.arsc	File ini berisi precompiled sumber daya aplikasi, dalam biner XML.
META-INF/	Folder ini menyimpan metadata tentang isi JARnya. signature APK juga disimpan di folder ini
lib/	Folder ini berisi kompilasi kode, pustaka kode asli.
res/	Folder ini berisi sumber daya dikompilasi ke dalam resources.arsc
assets/	Folder ini berisi aplikasi aset, yang dapat diambil dengan Manajer aset.

2.3 Malware Android

Malware android adalah sebuah perangkat lunak berbahaya yang dibuat khusus untuk menyerang sistem *smartphone*, jenis *malware* yang menyerang biasanya akan mengeksploitasi sistem operasi dimana target utamanya adalah *personal digital assistant* (PDA) sehingga menyebabkan pengguna android mengalami kebocoran data informasi rahasia [7]. Berikut merupakan jenis *malware* yang paling banyak menyerang perangkat Android [8]. *Adware* adalah sebuah perangkat lunak yang memasang sendiri di sistem tanpa sepengetahuan pengguna android dan menampilkan iklan ketika pengguna menjelajahi Internet [8]. *Spyware* adalah sebuah program untuk memantau pola perilaku pengguna android dan mencuri data informasi pribadi, seperti melihat *log* aktivitas dan pola penelusuran oleh pengguna. setelah mendapatkan data Informasi pengguna kemudian dikirim kembali ke pembuat *spyware* dan digunakan sebagai dasar iklan bertarget seperti iklan *pop-up* [6]. *Scareware* adalah aplikasi bertindak sebagai perangkat lunak antivirus jahat dengan menyesatkan pengguna untuk membayar atau mengunduh konten dari situs web [8].

2.4 Android Permission

Sistem operasi android menggunakan sistem yang berbasis izin yang fungsinya tidak hanya untuk membatasi perilaku aplikasi tetapi juga digunakan untuk memberi tahu pengguna tentang potensi perilaku aplikasi. Izin aplikasi diperlukan untuk digunakan akses izin yang diperlukan yang terdapat di dalam file AndroidManifest.xml. pengguna dapat menentukan daftar permintaan izin aplikasi yang akan dipasang dan pengguna aplikasi dapat membuat pilihan apakah akan menginstal aplikasi atau tidak berdasarkan daftar izin yang digunakan aplikasi. Setelah aplikasi diinstal, maka izin yang diminta akan mengakses pada *smartphone* pengguna [6]. Tabel 2 dibawah ini merupakan level permission

Tabel 2.2 Level Permission

Level Permission	Deskripsi
Normal	Tidak memberikan bahaya yang nyata bagi pengguna(misalnya mengganti wallpaper)
Berbahaya	Dapat memberikan bahaya yang nyata (misalya nomor, membuka koneksi internet, dll)
Signature	Secara otomatis diberikan kepada aplikasi jika aplikasi tersebut ditandatangani oleh kunci yang sama
Signature/ System	Sama seperti Signature, sistemnya menampilkan dengan mendapatkan izin secara otomatis dan dirancang hanya untuk digunakan oleh produsen perangkat

2.5 Analisis Malware

Dalam analisa *malware* android terdapat dua metode yaitu analisis statis dan analisis dinamis. Analisa statis merupakan proses menganalisis kode program tanpa benar-benar menjalankannya. Pendekatan ini memiliki kelebihan bagi peneliti karena dapat menutupi seluruh kode dan memungkinkan dapat mengamati perilaku program secara keseluruhan, sebelum berjalannya program selama *run-time* [9]. Sebuah teknik yang diperkenalkan dengan menggunakan model pemeriksaan untuk mengidentifikasi bagian dari sebuah program sebelum di *instal* atau dijalankan pada sebuah sistem [10], Kelebihan analisis statis ini hanya membutuhkan waktu dan sumber daya lebih sedikit. Sedangkan dalam analisa dinamis merupakan proses analisis yang meliputi perhitungan serta menguji aplikasi dengan mengeksekusi secara langsung tujuan dari analisa dinamis untuk menemukan kesalahan atau keanehan program selama dijalankan fokus utamanya yaitu menemukan *malware*. Kelebihan analisa dinamis yaitu dapat menganalisa perilaku aplikasi selama *run-time* dan menganalisa data penting atau data targetnya untuk menganalisa dinamis membutuhkan sumber daya yang besar dan waktu yang cukup lama [9].

2.6 Naïve bayes

Naïve bayes merupakan sebuah algoritma yang mengasumsikan bahwa atribut objek adalah *independen* dengan perkiraan probabilitas akhir dihitung sebagai jumlah frekuensi di mana fase pelatihan perlu mempertimbangkan setiap atribut di setiap kelas secara terpisah. Metode ini hanya membutuhkan sedikit data latih untuk menentukan estimasi parameter yang diperlukan dalam proses pengklasifikasiannya karena yang diasumsikan sebagai variabel *independen*, Sehingga hanya *varians* dari satu variabel dalam kelas yang diperlukan untuk menentukan klasifikasi, bukan seluruh matriks *kovarians* [2].

2.7 Random Forest

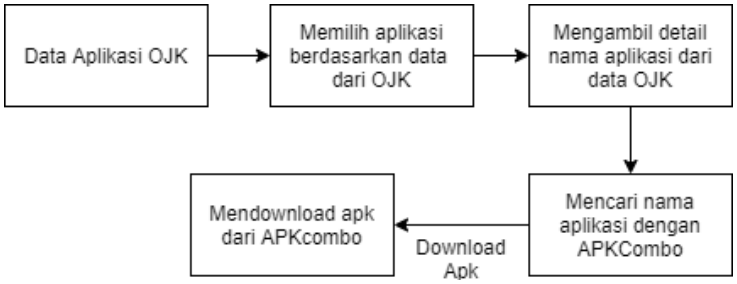
Random Forest adalah suatu algoritma yang digunakan untuk mengklasifikasikan data yang diperkenalkan oleh Leo Breiman pada tahun 2001 yang merupakan salah satu metode kombinasi dari *decision tree* yang digunakan untuk mengklasifikasikan suatu sampel data yang belum diketahui kelasnya. Penggunaan *decision tree* agar dapat menghindari *overfitting* pada sebuah set data saat mencapai akurasi yang maksimum. Kelebihan random forest terletak pada seleksi fitur yang secara acak untuk memilih setiap *node*, yang mampu menghasilkan kesalahan rendah. Random Forest memiliki kedalaman maksimal yang bergantung pada sebuah nilai *vector random* yang distribusinya sama di semua masing-masing *decision tree*. Random forest adalah sebuah unit yang akan memilih class yang paling popular berdasarkan input x dengan kumpulan klasifikasi yang terstruktur berbentuk pohon {h(x, θ k), k = 1, 2, 3 . .} dimana {θk} adalah *random vektor* yang didistribusikan *independen* [2] [11].

2.8 K-Fold Cross Validation

K-*Fold Cross Validation* merupakan salah satu metode *Cross Validation* untuk memperoleh hasil akurasi yang maksimal serta melihat nilai rata-rata tingkat keberhasilan dari sebuah model *machine learning*. Dengan cara melakukan beberapa perulangan dengan nilai sebanyak K kali untuk satu model dengan parameter yang sama Metode ini akan memecah data menjadi K bagian, dimana masing-masing memiliki jumlah data yang seimbang dan hasilnya berisi data *train* ,data *testing*. Akurasi yang didapat dari masing-masing iterasi sejumlah K yang dijalankan dirata-rata untuk mendapatkan nilai akurasi dari model yang dibangun [12].

3. Perancangan Sistem

3.1 Sampel aplikasi

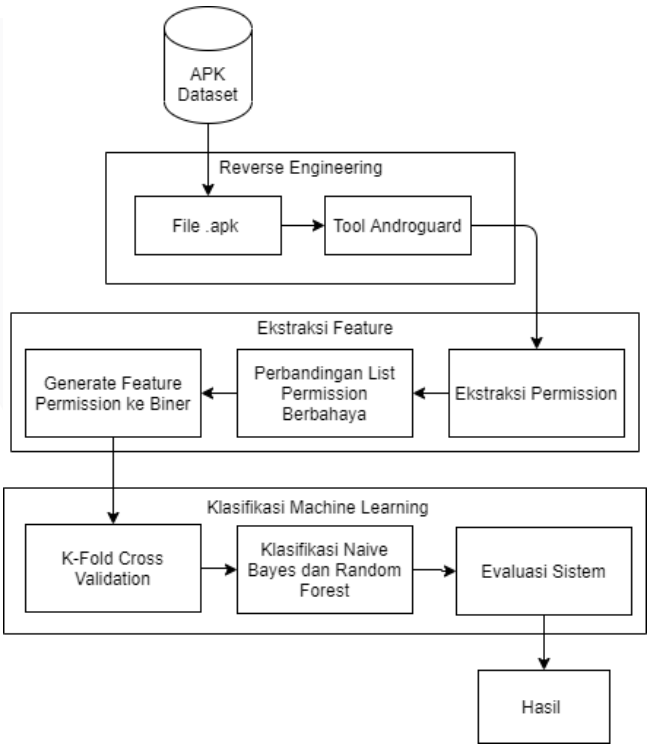


Gambar 3.1 Pengambilan sampel aplikasi dari data OJK

Dalam pengambilan sampel aplikasi diambil dari situs web playstore pihak ketiga yang bernama APKCombo situs tersebut menyediakan halaman web untuk aplikasi yang tersedia di playstore. Dalam pengambilannya berdasarkan data aplikasi menurut OJK yaitu 80 aplikasi yang terdaftar di OJK [19] dan 80 aplikasi yang tidak terdaftar di OJK [20]. Pengambilannya dengan memilih aplikasi berdasarkan data Ojk lalu melihat detail nama aplikasi *fintech* setelah itu dilakukan pencarian nama aplikasi di situs web APKCombo dan melakukan pengunduhan sampel aplikasi *fintech*.

3.2 Sistem yang dibangun

Pada penelitian ini bertujuan menganalisa deteksi *malware* pada aplikasi *fintech* berdasarkan permission. Sistem ini terbagi menjadi tiga bagian utama yaitu *Reverse Engineering*, Ekstraksi Feature dan klasifikasi *Machine learning*. Gambar 3.1 merupakan tahapan sistem yang akan dilakukan pada penelitian ini.



Gambar 3.2 Sistem yang dibangun

Tahap pertama dataset APK masuk ke bagian *Reverse Engineering* di tahap ini Dataset APK dilakukan pengambilan permission dengan menggunakan bantuan *tool androguard* [22]. Tahap kedua yaitu melakukan ekstraksi feature disini pengambilanya dengan cara memanggil fungsi a.get.permission. dari *tool androguard* [22] setelah mendapatkan permission lalu dilakukan perbandingan dengan list permission berbahaya pada tabel 3.1 dan sudah dilakukan feature pengecualian pada tabel 3.2 lalu

melakukan *generate* feature permission kedalam bentuk biner untuk dijadikan dataset. Setelah itu masuk ke tahap tiga yaitu klasifikasi *machine learning* untuk pengklasifikasiannya dilakukan dengan bantuan *tool* WEKA [21] Pertama melakukan K-*Fold Cross Validation* dengan K = 10, K yang di maksud disini merupakan percoban setelah melakukan percobaan lalu memilih algoritma Naïve bayes dan random Forest setelah itu mendapatkan evaluasi sistem dan yang terakhir mencatat hasil klasifikasinya.

Tabel 3.1 dibawah ini merupakan daftar tabel permission yang dikategorikan berbahaya yang biasa terdapat *malware* [13].

Tabel 3.1 Daftar Permissions Berbahaya

ACCESS_FINE_LOCATION	RESTART_PACKAGES
ACCESS_COARSE_LOCATION	GET_TASKS
READ_EXTERNAL_STORAGE	INSTALL_PACKAGES
WRITE_EXTERNAL_STORAGE	RECEIVE_BOOT_COMPLETED
CAMERA	DISABLE_KEYGUARD
READ_CONTACTS	WRITE_SETTING
WRITE_CONTACTS	GET_PACKAGE_SIZE
GET_ACCOUNTS	SET_ALARM
READ_PHONE_STATE	SET_WALLPAPER
READ_CALL_LOG	SYSTEM_ALERT_WINDOW
WRITE_CALL_LOG	CHANGE_CONFIGURATION
PROCESS_OUTGOING_CALLS	ACCESS_FINE_LOCATION
CALL_PHONE, READ_LOG	ACCESS_COARSE_LOCATION
ACCESS_WIFI_STATE, CHANGE	READ_EXTERNAL_STORAGE
WIFI STATE, INTERNET	WRITE_EXTERNAL_STORAGE
CHANGE_NETWORK_STATE	AUDIO_RECORD
WRITE_APN_SETTINGS	READ_CONTACTS
RECEIVE_SMS, READ_SMS	WRITE_CONTACTS
RECEIVE_WAP_PUSH	GET_ACCOUNTS
SEND_SMS	READ_PHONE_STATE
READ_HISTORY_BOOKMARKS	READ_CALL_LOG
WRITE_HISTORY_BOOKMARKS	WRITE_CALL_LOG
EXPANd_STATUS_BAR	PROCESS_OUTGOING_CALLS
PERSISTENT_ACTIVITY	CALL_PHONE, READ_LOG

Penelitian ini menggunakan sampel aplikasi *fintech* yang merupakan aplikasi jasa keuangan sehingga membutuhkan akses tertentu untuk dapat digunakan pada *smartphone*. Dengan begitu kami memberikan pengecualian terhadap beberapa kategori yang dianggap normal terdapat pada aplikasi *fintech* menurut OJK tiga kategori tersebut yaitu lokasi, kamera dan mikrofon [15]. Dan beberapa kategori permission normal yang terdapat pada aplikasi yaitu kategori jaringan [14]. Kategori pengecualian tersebut dapat dilihat pada tabel 3.2

Tabel 3.2 Permission Normal

Kategori permission	Permissions
Camera	CAMERA
Location	ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION
mikrofon	AUDIO_RECORD
Jaringan	ACCESS_WIFI_STATE CHANGE WIFI STATE CHANGE_NETWORK_STATE WRITE_APN_SETTINGS INTERNET

3.3 Dataset

Pada penelitian ini dataset yang digunakan merupakan hasil dari ekstraksi feature permission pada sampel aplikasi *fintech* lalu diubah menjadi biner. Dataset berisi 160 baris data dan 31 kolom feature dan 1 kelas label yang merupakan target dari prediksi ini. Dataset dapat dilihat pada tabel 3.3

Tabel 3.3 Dataset

No	Android.Permission.											Class
	READ_EXTERNAL_STORAGE	WRITE_EXTERNAL_STORAGE	READ_CONTACTS	WRITE_CONTACTS	GET_ACCOUNTS	READ_PHONE_STATE	READ_CALL_LOG	WRITE_CALL_LOG	PROCESS_OUTGOING_CALLS	…	CHANGE_CONFIGURATION	
1	1	0	0	0	0	0	1	0	0	…	0	yesOJK
2	1	0	0	0	0	0	1	0	0	…	0	yesOJK
…	…	…	…	…	…	…	…	…	…	…	…	…
159	1	1	1	1	1	0	1	0	0	…	0	noOJK
160	1	1	1	0	0	1	1	0	0	…	0	noOJK

3.4 K-Fold Cross Validation

Pada penelitian ini akan menggunakan teknik K-*Fold Cross Validation* dalam membagi dataset Proses pembagian ini dilakukan secara berulang ulang hingga bagian ke-K menjadi data *train* dan bagian lainnya menjadi data *test* dengan tujuan untuk menemukan kombinasi validasi model terbaik. Penelitian akan melakukan percobaan dengan menggunakan K=10 yang biasanya sering disebutkan dengan 10-*Fold Cross Validation* [3] dengan menggunakan *cross validation* yang tersedia di *tool kit* WEKA [21] lalu hasilnya di rata-rata. Setelah dibagi dengan metode K-*Fold* dengan nilai K=10 maka selanjutnya melakukan klasifikasi dengan metode naïve bayes dan random forest.

3.5 Evaluasi Sistem

Pada penelitian ini kami mendefinisikan secara singkat matrik yang digunakan untuk mendapatkan hasil klasifikasi

1. **True Positif (TP)**
   Didefinisikan sebagai sampel aplikasi yang dianggap positif dan dalam faktanya positif
2. **True Negatif (TN)**
   Didefinisikan sebagai sampel aplikasi yang dianggap positif dan dalam faktanya negatif
3. **False Positif (FP)**
   Didefinisikan sebagai sampel aplikasi yang dianggap negatif dan dalam faktanya positif
4. **False Negatif (FN)**
   Didefinisikan sebagai sampel aplikasi yang dianggap negatif dan dalam faktanya negatif
5. **Confusion Matrix**
   Sebuah matrik yang terdiri dari baris dan kolom untuk setiap kelas yang berisikan nilai prediksi dan aktual. Untuk baris menunjukan setiap elemen matrik dari jumlah *instance* kelas aktual sedangkan kolom menunjukan prediksi kelasnya.

		Prediction class	
		Sampel Positif	Sampel negatif
Actual class	Sampel Positif	TP	FN
	Sampel Negatif	FP	TN

6. **True Positif Rate**
   True positif rate adalah untuk mengukur rasio prediksi yang dianggap positif dan dalam faktanya diidentifikasikan benar. Nilai TPR dapat diperoleh dengan persamaan.

$$True\ Positif\ Rate\ (TPR) = \frac{TP}{TP+FN} \times 100\% \quad (3)$$

7. Akurasi

Akurasi didefinisikan sebagai jumlah rasio prediksi yang benar positif dan negatif lalu dibagi dengan keseluruhan data. Nilai akurasi dapat diperoleh dengan persamaan.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (2)$$

8. Precision

Precision didefinisikan sebagai rasio prediksi benar positif dibandingkan dengan jumlah data yang negatif yang dianggap positif. Nilai precision diperoleh dengan persamaan.

$$Precision = \frac{TP}{TP+FP} \times 100\% \quad (2)$$

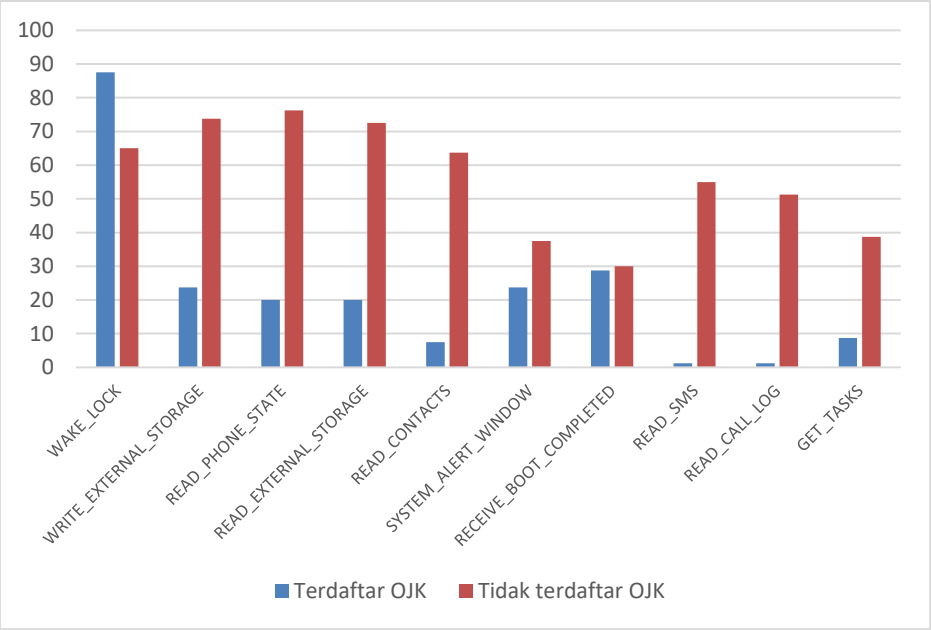
9. Recall

Recall didefinisikan sebagai rasio jumlah prediksi positif dibandingkan dengan jumlah data benar positif. Nilai Recall diperoleh dengan persamaan.

$$Recall = \frac{TP}{TP+FN} \times 100\% \quad (1)$$

4. Hasil Pengujian Dan Analisis

Pada penelitian ini dalam deteksi *malware* pada aplikasi android *fintech* berdasarkan permission menggunakan kumpulan data sampel aplikasi *fintech* yang berasal dari OJK yang terdiri dari 80 sampel aplikasi *fintech* yang terdaftar di OJK dan 80 sampel aplikasi *fintech* yang tidak terdaftar di OJK yang diubah menjadi sebuah dataset biner dengan melakukan ekstraksi feature permission. Untuk tujuan tersebut penelitian ini menggunakan *machine learning* naïve bayes dan random forest sebagai algoritma klasifikasi.



Gambar 4.1 Urutan 10 permision yang paling banyak terdapat di sampel aplikasi *fintech*

Gambar 4.1 merupakan 10 permission berbahaya yang memiliki frekuensi tertinggi dari ekstraksi feature dengan menggunakan bantuan *tool* androguard [22]. Permission tersebut didapatkan melalui ekstraksi feature dari 160 sampel aplikasi.*fintech*. permission.WAKE_LOCK merupakan persentase permission berbahaya paling tertinggi dengan 90% untuk sampel aplikasi *fintech* yang terdaftar di OJK dan 64% untuk sampel aplikasi *fintech* yang tidak terdaftar di OJK. Dalam *machine learning* permission-permission ini dapat membentuk rangkaian fitur penting untuk menganalisa permission baru atau permission yang tidak dikenali pada aplikasi *fintech*.

Tabel 4.1 Detail klasifikasi dengan algoritma Naïve Bayes

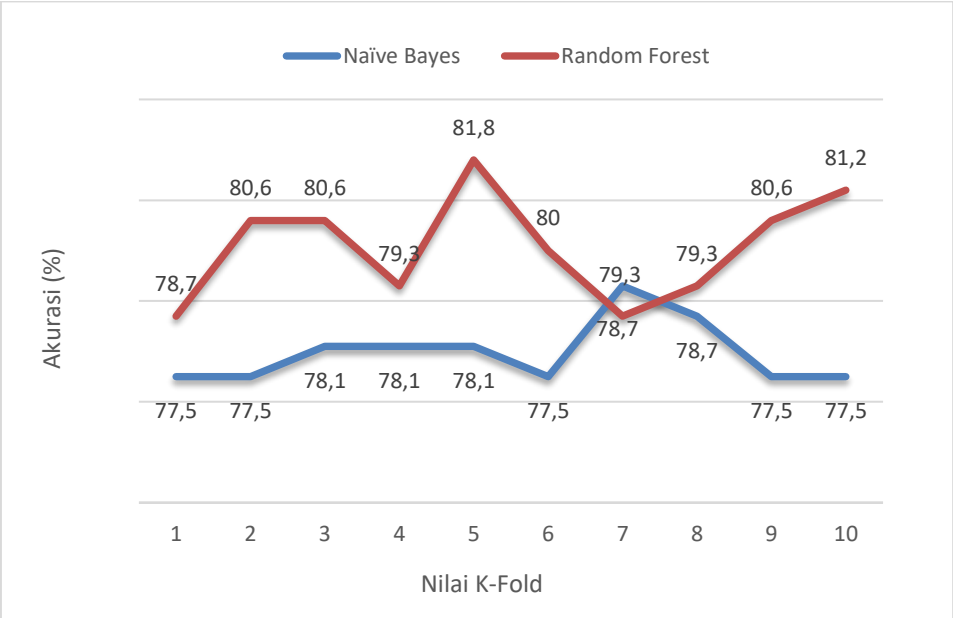
Algoritma	TP rate	Precision	Recall	Class
Naïve Bayes	0,893	0,7283	0,893	yesOJK
	0,6678	0,8614	0,6678	noOJK
Rata-rata	0,7804	0,79485	0,7804	

Tabel 4.2 Detail klasifikasi dengan algoritma Random Forest

Algoritma	TP rate	Precision	Recall	Class
Random Forest	0,864	0,7677	0,864	yesOJK
	0,7389	0,8449	0,7389	noOJK
Rata-rata	0,80145	0,8063	0,80145	

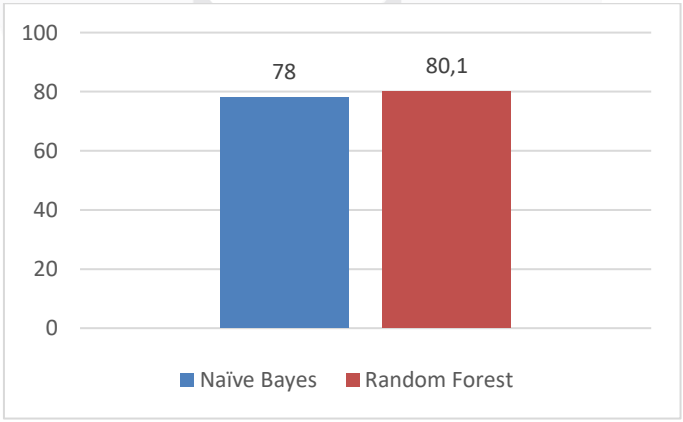
Tabel 4.1 dan tabel 4.2 merupakan hasil klasifikasi K *Fold Cross Validation* dengan nilai K = 10 pada *tool* WEKA [21]. Hasilnya dari 160 kumpulan dataset yang terdiri dari 80 data set terdaftar di OJK dan 80 data set yang tidak terdaftar di OJK pada algoritma naïve bayes mendapatkan hasil akurasi rata-rata sebesar 78% dan disini kami juga mendapatkan hasil secara detail dalam mengklasifikasikan sampel aplikasi *fintech* yang tidak terdaftar di OJK dengan true positif ratenya sebesar 66,7% dan untuk sampel aplikasi yang terdaftar

di OJK true positif ratenya sebesar 89% detailnya dapat dilihat pada tabel 4.1 untuk hasil klasifikasinya. Sedangkan dalam algoritma random forest mendapatkan hasil akurasi rata-rata sebesar 80,1% dari 160 sampel aplikasi *fintech*. Selanjutnya dalam random forest kami mencapai akurasi true positif ratenya sebesar 73,8% untuk sampel aplikasi yang tidak terdaftar di OJK dan mencapai akurasi true positif ratenya 86% untuk sampel aplikasi yang terdaftar di OJK. Hasil klasifikasinya random forest untuk dapat dilihat pada tabel 4.2 detail klasifikasi random forest



Gambar 4.2 Akurasi dari K *fold Cross Validation* pada nilai ke- K

Gambar 4.2 merupakan akurasi hasil percobaan dengan mengganti nilai K dengan tujuan untuk menemukan kombinasi validasi model terbaik. Pada naïve bayes validasi model terbaik ketika melakukan percobaan K=7 dengan akurasi sebesar 79,3% sedangkan kombinasi validasi model terbaik untuk Random Forest ketika melakukan percobaan K=5 dengan nilai akurasi sebesar 81,8%. Hasil percobaan dengan K *fold Cross validation* selanjutnya dirata-rata untuk menentukan perbandingan antara akurasi Naïve Bayes dan Random Forest yang dapat dilihat pada gambar 4.3.



Gambar 4.3 Rata-rata akurasi

Hasil analisa akhirnya dalam deteksi *malware* pada aplikasi android *fintech* berdasarkan permission dengan menggunakan *machine learning* setelah dilakukan pengujian algoritma random forest memiliki akurasi yang lebih tinggi jika dibandingkan dengan naïve bayes dengan menggunakan K-*Fold Cross Validation* dengan K = 10 hal ini dapat dibuktikan pada perbandingan di gambar 4.3 akurasi dalam random forest mempunyai nilai akurasi lebih sebesar dengan rata-rata 80,1% sedangkan naïve bayes mempunya nilai akurasi rata-rata 78%.

5. Kesimpulan

Pada penelitian ini dengan tujuan analisa deteksi *malware* pada aplikasi *fintech* berdasarkan permission hasil yang didapat dengan menggunakan dataset yang diambil melalui ekstraksi fitur permission pada 160 sampel aplikasi *fintech* dengan 80 aplikasi yang terdaftar di OJK dan 80 aplikasi yang tidak terdaftar di OJK *machine learning* dapat membentuk rangkaian fitur baru yang belum dikenali pada permission untuk mendeteksi *malware* aplikasi *fintech*. Dalam *machine learning* algoritma klasifikasi naïve bayes dan random forest hasil performa akurasi random forest lebih tinggi dibandingkan dengan naïve bayes. Hal ini dapat dibuktikan setelah melakukan percobaan menggunakan K *fold Cross Validation* dengan K = 10 *Cross Validation* dalam algoritma random forest mencapai akurasi rata-rata sebesar 80,1% sedangkan dalam algoritma naïve bayes mencapai akurasi rata-rata sebesar 78%.

Untuk penelitian selanjutnya, dapat mengembangkan fitur deteksi *malware* lainnya pada aplikasi *fintech* selain berdasarkan permission misalnya berdasarkan API call atau behaviour serta menggunakan metode klasifikasi *machine learning* lainnya untuk mengetahui metode mana yang dapat menghasilkan tingkat akurasi yang paling baik dan dapat menemukan fitur terbaru pada aplikasi *fintech*.

REFERENSI

[1] F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani, "Evaluation of machine learning classifiers for mobile malware detection," *Soft Comput.*, vol. 20, no. 1, pp. 343–357, 2016, doi: 10.1007/s00500-014-1511-6.

[2] A. Mahindru and P. Singh, "Dynamic permissions based android malware detection using machine learning techniques," *ACM Int. Conf. Proceeding Ser.*, pp. 202–210, 2017, doi: 10.1145/3021460.3021485.

[3] S. Verma and S. K. Muttoo, "An android malware detection framework-based on permissions and intents," *Def. Sci. J.*, vol. 66, no. 6, pp. 618–623, 2016, doi: 10.14429/dsj.66.10803.

[4] K. Khariwal, J. Singh, and A. Arora, "IPDroid: Android malware detection using intents and permissions," *Proc. World Conf. Smart Trends Syst. Secur. Sustain. WS4 2020*, pp. 197–202, 2020, doi: 10.1109/WorldS450073.2020.9210414.

[5] Y. Huan, Z. Yu-qing, and H. Yu-pu, "Android malware detection method based on permission sequential pattern mining algorithm," *J. Commun.*, 2013.

[6] S. M. Myat, "Analysis of Android Applications by Using Reverse Engineering Techniques," vol. 4, no. 3, pp. 551–558, 2019.

[7] V. Saxena, S. Shrivastava, and S. Mourya, "Behavior Analysis of Android malware detection for Smart phone," *Int. J. Eng. Res. Sci.*, vol. 2, no. 12, pp. 55–60, 2016.

[8] M. Shohel Rana, C. Gudla, and A. H. Sung, "Evaluating machine learning models for android malware detection - A comparison study," *ACM Int. Conf. Proceeding Ser.*, pp. 17–21, 2018, doi: 10.1145/3301326.3301390.

[9] M. Choudhary and B. Kishore, "HAAMD: Hybrid Analysis for Android Malware Detection," *2018 Int. Conf. Comput. Commun. Informatics, ICCCI 2018*, pp. 1–4, 2018, doi: 10.1109/ICCCI.2018.8441295.

[10] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, "Semantics-Aware Malware Detection," 2005.

[11] Z. Jin, J. Shang, Q. Zhu, C. Ling, W. Xie, and B. Qiang, "RFRSF: Employee Turnover Prediction Based on Random Forests and Survival Analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12343 LNCS, pp. 503–515, 2020, doi: 10.1007/978-3-030-62008-0_35.

[12] S. Yadav and S. Shukla, "Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification," *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, no. Cv, pp. 78–83, 2016, doi: 10.1109/IACC.2016.25.

[13] R. Kumar, X. Zhang, W. Wang, R. U. Khan, J. Kumar, and A. Sharif, "A Multimodal Malware Detection Technique for Android IoT Devices Using Various Features," *IEEE Access*, vol. 7, no. c, pp. 64411–64430, 2019, doi: 10.1109/ACCESS.2019.2916886.

[14] E. Tramontana and G. Verga, "Mitigating Privacy-Related Risks for Android Users," *Proc. - 2019 IEEE 28th Int. Conf. Enabling Technol. Infrastruct. Collab. Enterp. WETICE 2019*, pp. 243–248, 2019, doi: 10.1109/WETICE.2019.00059.

[15] Otoritas jasa keungan 2019 | Available:https://www.ojk.go.id/id/kanal/iknb/datadanstatistik/direktori/fintech/Documents/FAQ%20Fintech%20Lending.pdf

[16] Otoritas jasa keungan 2020 | Available: https://www.ojk.go.id/id/berita-dankegiatan/siaran-pers/Pages/Siaran-Pers-Satgas-Waspada-Investasi-Tutup-126-Fintech-Lending-Ilegal-dan-32-Investasi-Tanpa-Izin-.aspx

[17] Cahyo Prayogo (Ed.). Warta Ekonomi 2019 | Available: https://www.wartaekonomi.co.id/read 238509/waspadai -spyware-di-aplikasi-fintechilegal

[18] Zeltser, L. (2014). What is Malware. Diakses dari https://www.phas.ubc.ca/sites/default /files/shared/it-service-catalogue/ouch/ouch-201402_en.pdf

[19] Otoritas jasa keuangan 2021 | Available:https://www.ojk.go.id/id/kanal/iknb /financialtechnology/Documents/penyelenggara%20berizin%20terdaftar%20per%2022%20Januari%202021.pdf

[20] Otoritas jasa keuangan 2020 | Available: https://www.ojk.go.id/id/berita-dankegiatan/siaran-pers/Documents/Pages/Siaran-Pers-Satgas-Kembali-Temukan-182-Fintech-Peer-To-Peer-Lending-TanpaIzin/DAFTAR%20Fintech% 20LENDING %20TAK%20BERIZIN%20-%20SEPT%20(1).pdf

[21] Machine Learning Software in Java 2021 :Available : https://www.cs.waikato.ac.nz/ml/weka/

[22] androguard 2021 | Available : https://github.com/androguard/androguard