DOI: https://dx.doi.org/10.21111/fij.v3i1.1875
Diterima: 10 April 2018 Revisi: - Terbit: 4 Mei 2018

Implementasi Teknik Seleksi Fitur Pada Klasifikasi *Malware Android* Menggunakan *Support Vector Machine*

Hendra Saputra[1) *], Setio Basuki[2)], Mahar Faiqurahman[3)]

Teknik Informatika, Universitas Muhammadiyah Malang[1),2),3)]
*hendra@webmail.umm.ac.id [1) *], setiobasuki.umm@gmail.com [2)], mahar@umm.ac.id [3)]*

Abstrak

Pertumbuhan Malware Android telah meningkat secara signifikan seiring dengan majunya jaman dan meninggkatnya keragaman teknik dalam pengembangan Android. Teknik Machine Learning adalah metode yang saat ini bisa kita gunakan dalam memodelkan pola fitur statis dan dinamis dari Malware Android. Dalam tingkat keakurasian dari klasifikasi jenis Malware peneliti menghubungkan antara fitur aplikasi dengan fitur yang dibutuhkan dari setiap jenis kategori Malware. Kategori jenis Malware yang digunakan merupakan jenis Malware yang banyak beredar saat ini. Untuk mengklasifikasi jenis Malware pada penelitian ini digunakan Support Vector Machine (SVM). Jenis SVM yang akan digunakan adalah class SVM one against one menggunakan Kernel RBF. Fitur yang akan dipakai dalam klasifikasi ini adalah Permission dan Broadcast Receiver. Untuk meningkatkan akurasi dari hasil klasifikasi pada penelitian ini digunakan metode Seleksi Fitur. Seleksi Fitur yang digunakan ialah Correlation-based Feature Selection (CSF), Gain Ratio (GR) dan Chi-Square (CHI). Hasil dari Seleksi Fitur akan di evaluasi bersama dengan hasil yang tidak menggunakan Seleksi Fitur. Akurasi klasifikasi Seleksi Fitur CFS menghasilkan akurasi sebesar 90.83%, GR dan CHI sebesar 91.25% dan data yang tidak menggunakan Seleksi Fitur sebesar 91.67%. Hasil dari pengujian menunjukan bahwa Permission dan Broadcast Receiver bisa digunakan dalam mengklasifikasi jenis Malware, akan tetapi metode Seleksi Fitur yang digunakan mempunyai akurasi yang berada sedikit dibawah data yang tidak menggunakan Seleksi Fitur.

Kata kunci: *klasifikasi malware android, seleksi fitur, SVM dan multi class SVM one agains one*

Abstract

[Implementation of Technic Feature Selection on Android Malware Classification using Support Vector Machine] *Android Malware hasgrowth significantly along with the advance of the times and the increasing variety of technique in the development of Android. Machine Learning technique is a method that now we can use in the modeling the pattern of a static and dynamic feature of Android Malware. In the level of accuracy of the Malware type classification, the researcher connect between the application feature with the feature required by each types of Malware category. The category of malware used is a type of Malware that many circulating today, to classify the type of Malware in this study used Support Vector Machine (SVM). The SVM type wiil be used is class SVM one against one using the RBF Kernel. The feature will be used in this classification are the Permission and Broadcast Receiver. To improve the accuracy of the classification result in this study used Feature Selection method. Selection of feature used are Correlation-based Feature Selection (CFS), Gain Ratio (GR) and Chi-Square (CHI). Result from Feature Selection will be evaluated together with result that not use Feature Selection. Accuracy Classification Feature Selection CFS result accuracy of 90.83%, GR and CHI of 91.25% and data that not use Feature Selection of 91.67%. The result of testing indicate that permission and broadcast receiver can be used in classyfing type of Malware, but the Feature Selection method that used have accuracy is a little below the data that are not using Feature Selection.*

Keywords*: Classification Android Malware, Feature Selection, SVM and Multi Class SVM one against one*

1. PENDAHULUAN

Android merupakan salah satu sistem operasi yang marak digunakan saat ini. Android merupakan sebuah nama dari sistem operasi pada suatu gadget seperti computer atau tablet, smartphone, dan telpon seluler. Sistem operasi yang digunakan ialah berbasis linux. Sistem Android ini dikembangkan oleh Google Inc. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang akan di gunakan oleh bermacam perangkat seluler. Google mengatakan, 1,3 juta perangkat Android yang diaktifkan setiap hari[1]. Aplikasi yang tersedia di pasar Google Play Android sendiri berdasarkan situs appbrain.com sudah mencapai sekitar 2.673.850 apps pada awal 2017[2]. Android sudah sangat jauh meninggalkan para pesaingnya, pada 2013 saja Android sudah menguasai 78% dari total pasar smartphone [3]. Menurut data International korporasi IDC, Android OS mendominasi dengan 82,8% dari total pangsa pasar pada 2Q 2015 dari hal ini bisa dilihat bahwa android menjadi salah satu sistem operasi yang paling banyak digunakan selama bertahun-tahun[4].

Dengan meningkatnya pengguna pada perangkat smartphone berbabasis android dikalangan pemerintah, masyarakat dan perusahaan saat ini menjadi celah peluang dan target akan tindakan *indtruder* dalam melakukan aktifitas berbahaya, seperti *MaliciousSoftware* atau disebut *Malware*. *Malware* merupakan perangkat lunak yang secara eksplisit didesain untuk melakukan aktifitas berbahaya atau perusak perangkat lainya, seperti *Trojan, Virus, Spyware* dan *Exploit*[5]. Aktifitas berbahaya yang diakibatkan dari *Malware*akan berdampak sangat merugikan bagi para korbanya ketika informasi pribadi dicuri, sistem yang dirusak, aktifitas disadap dan diintai. Hal ini merupakan sebuah bukti tindakan kejahatan digital yang dilakukan oleh seseorang *Intruder* dengan memanfaatkan *Malware* sebagai medianya. Google sendiri sudah membuat scanner keamanan berbasis cloud yang disebut Bouncer yang bertujuan untuk mendeteksi aplikasi berbahaya di Play Store. Akan tetapi masih ada saja aplikasi yang berisi *Malware* berhasil masuk kedalam Play Store dan masih banyaknya pengguna Android yang mendownload aplikasi pada pihak ketiga, sehingga smartphone mereka masih saja dapat terkena *Malware*. Oleh karena itu banyak dilakukan penelitan mengenai klasifikasi *Malaware* pada aplikasi Android yang di lakukan secara statik maupun dinamik menggunakan *mechinelearning* pada penelitian[1],[6],[7],[8],[9]. Penelitian yang dilakukan Saba Arshad dkk melakukan analisa *Malware* menggunakan metode statik dan dinamik, metode statik mengambil data berdasarkan *signature, permission* dan *dalvikBytecode* sedangkan metode dinamiknya berdasarkan *Anomaly, Traint, Emulasi* sebagai parameter fiturnya[1]. Pada penelitian Sharvari dan Narendra dalam deteksi *Malware* berfokus kepada *improvedmutualinformation*(IMIFS) sebagai seleksi fitur yang digunakanya[6]. Pada penelitian Naser dan Xingquan menggunakan permission dan APICalls sebagai fitur untuk deteksi Malware pada aplikasi Android [7]. Pada penelitian Suleiman dkk, digunakan fitur permission dan base-code properti sebagai fitur dan Baiyesian sebagai algortima klasifikasi, sebelum data di uji dilakukan terlebih dahulu seleksi fitur menggunakan *MutualInformation(MI)*[8]. Pada peneliatian Yeriam dkk, digunakan *Permssion, Command* dan *APICalls* sebagai fitur dan *MutualInformation*(MI) sebagai Seleksi Fitur[9]. Namun dari beberapa penelitian tersebut belum dilakukan klasifikasi jenis *Malware* dan analisa perbandingan mengenai Seleksi Fitur terhadap data yang ada. Seleksi Fitur sendiri bertujuan untuk menghilangkan fitur noise yang mungkin menyebabkan hasil klasifikasi menjadi kurang akurat[10].

Dengan semakin banyaknya *Malware* yang dibuat setiap tahunnya dan masih banyaknya user yang mendownload aplikasi pada pihak ketiga yang tidak diketahui asalnya, maka pada penelitian ini dibuatlah sistem untuk klasifikasi aplikasi yang mengandung Malware menggunakan mechinelearning dengan algoritma *Support Vector Machine* dan untuk mendapatkan hasil akurasi terbaik maka dilakukanlah analisis terhadap beberapa Seleksi Fitur untuk menentukan metode Seleksi Fitur mana yang lebih efisien terhadap data properti Android yang akan digunakan pada algoritma *Support Vector Machine* sehingga menghasilkan analisi klasifikasi yang lebih akurat.

2. BAHAN DAN METODE

2.1 Pengumpulan data

Untuk bagian pertama dalam penelitian ini adalah pengumpulan file apk Malware dari beberapa Jenis Malware. Data yang digunakan dalam penelitian ini adalah data Android aplikasi Malware yang berformat apk yang didapat dari situs Virusshare dan Contagio. Kemudian data yang di dapat dibagi menjadi 4 target kelas, yang mana berupa jenis MalwareAdwo, TrojanSMS, Droidkungfu dan Plankton, setiap Malware berjumlah 200 file apk. Parameter yang akan digunakan ialah berupa Permission dan Broadcast Receiver serta jenis Malware sebagai label/kelas.

Setiap apk yang sudah dikumpulkan dari situs *VirusShare* tidak ada nama label jenis *Malwarenya.* Untuk mengetahui jenis *Malware* dari setiap aplikasi yang didapat dari *VirusShare*, disini peneliti menggunakan situs *VirusTotal*, hasil dari *VirusTotal* bisa dilihat pada gambar 1,2,3 dan 4. Dari semua data hasil dari *VirusTotal* menunjukan lebih banyaknya aplikasi-aplikasi yang mengandung *Malware Trojan SMS, Adwo, Droidkungfu dan Plankton*. Sehingga

peneliti akan menggunakan keempat *Malware* tersebut sebagai kelas dalam pemodelan klasifikasi. *Malware Trojan SMS*, *Adwo* dan *Droidkungfu* mencukupi jumlah *Malware* yang dibutuhkan, akan tetapi *Malware* jenis *Plankton* yang didapat masih kurang, jumlah *Plankton* yang didapat sekitar 160 *Malware* dan masih kurang 40 *Malware*. Sehingga untuk menutupi kekurangan *MalwarePlankton* tersebut peneliti mencoba memintalagi kesitus Contagio. Dari Contagio didapat Malware sebanyak 46 Malware sehingga jumlah Malware bisa tercukupi. Contoh hasil Upload situs VirusTotal bisa dilihat pada gambar dibawah ini.

Avira	ADWARE/ANDR.AdsWo.CG.Gen
BitDefender	Android.Adware.Adwo.A
Cyren	AndroidOS/AdWo.A
Emsisoft	Android.Adware.Adwo.A (B)
ESET-NOD32	a variant of Android/AdDisplay.AdsWo.A potentially unwanted
Fortinet	Adware/Adswo.KK
Ikarus	AdWare.AndroidOS.AdWo

Gambar 1. Hasil Uploadfile apk yang mempunyai label Adwo.

AVware	Trojan.AndroidOS.DroidKungFu.a
BitDefender	Android.Trojan.DroidKungFu.L
ClamAV	Andr.Trojan.KungFu-9
DrWeb	Android.Gongfu.6
eScan	Android.Trojan.DroidKungFu.L
F-Prot	AndroidOS/DroidKungFu.A
Fortinet	Android/DroidKungFu.B!tr

Gambar 2. Hasil Uploadfile apk yang mempunyai label DroidKungfu.

AhnLab-V3	Android-PUP/Plankton.1c00
Antiy-AVL	Trojan/Android.Plangton
Avira	ANDROID/Plankton.J.Gen
Baidu	Android.Trojan.Plankton.k
ClamAV	Andr.Trojan.Plangton-13
Cyren	AndroidOS/Plankton.A.gen!Eldorado
ESET-NOD32	a variant of Android/Plankton.H

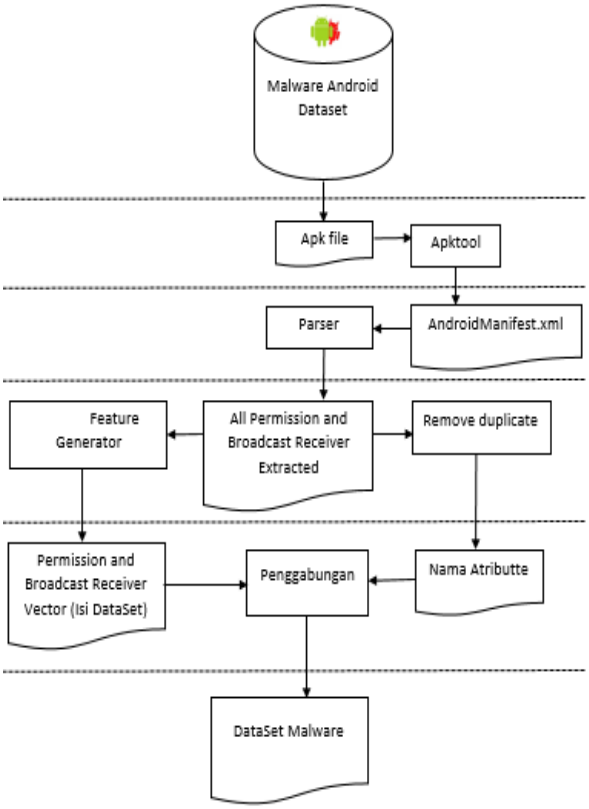
Gambar 3. Hasil Uploadfile apk yang mempunyai label Plankton.

CAT-QuickHeal	Android.FakeInst.EJ
Comodo	UnclassifiedMalware
DrWeb	Android.SmsSend.232
ESET-NOD32	Android/TrojanSMS.Boxer.AQ.Gen
F-Secure	Trojan:Android/Boxer.C
GData	Android.Trojan.FakeInst.B

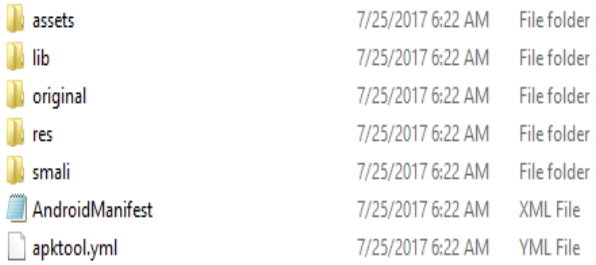
Gambar 4. Hasil Uploadfile apk yang mempunyai label TrojanSMS.

2.2 Extraksi Aplikasi

File Aplikasi apk yang akan digunakan harus di extrak terlebih dahulu agar bisa diambil isi data Properti pada aplikasi Android. Untuk alur extraksi aplikasi Android bisa dilihat pada gambar 5. Pada gambar 5 menjelaskan alur extraksi dari sebuah aplikasi Android sampai menjadi data atribut dan data dalam bentuk vector. Untuk extraksi aplikasi android disini peneliti menggunakan aplikasi Apktool dengan cara mengetikan pada CMD *"apktool d (nama_file)"*.



Gambar 5. Desain Alur Extraksi Aplikasi Android**.**



Gambar 6. Hasil ekstraksi Apk Android dengan menggunakan Apktool**.**

Dari hasil ekstraksi pada gambar 6 menghasilkan beberapa file, akan tetapi file yang akan digunakan dalam penelitian ini adalah file AndroidManifest.xml yang didalam file tersebut terdapat permission dan broadcast receiver.

2.3 Parsing Data

Setelah semua aplikasi yang dibutuhkan selesai diekstrak, maka akan dilakukan parsing data*permission*yang bisa dilihat pada gambar 7 dan *broadcastreceiver* yangbisa dilihat pada gambar 8 dari *AndroidManifest.xml* kedalam file yang berbentuk format txt yang mana nantinya akan digunakan dalam penentuan jenis atribut beserta isi atribut yang dibutuhkan dalam proses klasifikasi.

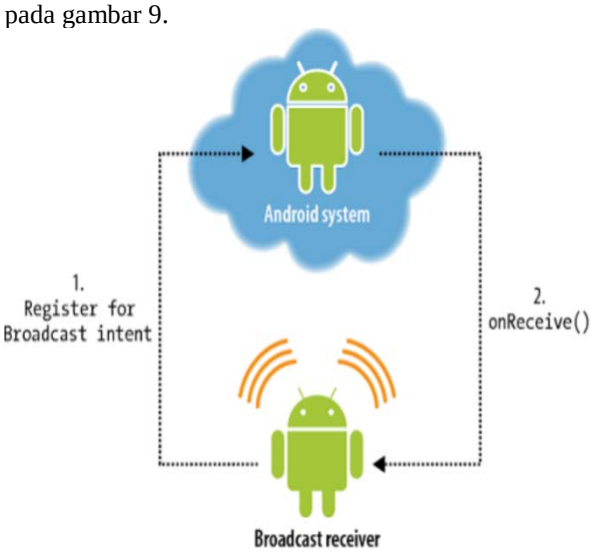
```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.WRITE"/>
<uses-permission android:name="android.permission.CALL_PHONE"/>
```

Gambar 7. Contoh Request Permissions Aplikasi Android**.**

```
<receiver android:name="net.robotmedia.billing.BillingReceiver">
  <intent-filter>
    <action android:name="com.android.vending.billing.IN_APP_NOTIFY"/>
    <action android:name="com.android.vending.billing.RESPONSE_CODE"/>
    <action android:name="com.android.vending.billing.PURCHASE_STATE_CHANGED"/>
  </intent-filter>
</receiver>
```

Gambar 8. Contoh Request Broadcast Receiver Aplikasi Android

Android memungkinkan aplikasi untuk bisa berinteraksi dengan system dan aplikasi lainnya dengan mengirim dan mendengarkan untuk menyiarkan yang dikirim memalui system dan aplikasi yang terpasang pada Android. Oleh karena itu dalam peneliti juga menggunakan *broadcast receiver*yang berada pada file *AndroidManifest.xml*sebagai atribut untuk klasifikasi, karena sebuah aplikasi juga bisa memantau atau mengirimkan pemberitahuan pada aplikasi lainnya. Yang bisa mengakibatkan terjadinya pemantauan sebuah aktifitas pada sebuah Smartphone. Untuk mekanisme *Broadcast Receiver* bisa dilihat pada gambar 9.



Gambar 9. Mekanisme Broadcast pada Android [5]

2.4 Penentuan Attribut dan Binari Vector

- Penentuan Attribut

Kemudian setalah didapat semua properti yg di butuhkan dari semua apk, maka akan ada dua pengerjaan dari properti dalam format txt yang sudah diparsing dari *AndroidManifest.xml*. Yang pertama menentukan atribut yang akan di pakai nantinya dalam pemodelan Klasifikasi, untuk menentukan atribut properti disini yaitu *Permission* dan *Broadcast Receiver* yang sudah kita ambil dari AndroidManifest.xml semuanya akan disusun kedalam file excel. Setelah semua properti dari 800 *Malware* tersebut sudah tersusun semua, maka akan dilakukan penghapus data yang duplikat sehingga data tidak ada yang sama yang nantinya akan digunakan sebagai atribut Klasifikasi. Setelah dilakukan penghapusan properti yang duplikat maka pada penelitian ini didapat total 402 atribut yang nantinya akan dipakai dalam proses Klasisfikasi

- Binari Vector

Yang kedua ialah dari setiap properti apk yang ada dalam file txt akan diubah kedalam bentuk vector. Untuk mengubah properti yang sudah di hasilkan kedalam bentuk vector yang berformat csv, maka dibuat program sederhana yang mana nantinya jika atribut ada pada fitur dalam suatu aplikasi, maka atribut tersebut akan bernilai 1 dan bila tidak ada maka akan bernilai 0. Agar lebih jelas bisa dilihat pada gambar 10.

$$R_i = \begin{cases} 1, \textit{jika property ada pada atribute yang sudah di tentukan} \\ \hline 0, \textit{jika tidak ada property pada atribute yang sudah ditentukan} \end{cases} \quad (1)$$

B	OC	OD	OE	OF	OG	OH	OI	OJ	OK	OL	OM	ON
	com.wwefightgame.freeplay.permission.C2D_MESS/	com.your.domain.PAYMENT_BROADCAST_PERMISSI(	installer.com.dama.papercamera.permission.C2D_M	installer.net.zhuowei.zhang.mcpelauncher.pro.permissi	org.adw.launcher.permission.READ_SETTINGS	org.adw.launcher.permission.WRITE_SETTINGS	org.adwfreak.launcher.permission.READ_SETTINGS	org.adwfreak.launcher.permission.WRITE_SETTINGS	ru.android.apps.permission.C2D_MESSAGE	android.intent.category.HOME		
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	
0	0	0	0	0	0	0	0	0	0	1	droidkungfu	
0	0	0	0	0	0	0	0	0	0	1	droidkungfu	
0	0	0	0	0	1	1	1	1	0	0	droidkungfu	
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	
0	0	0	0	0	1	0	0	0	0	0	droidkungfu	
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	
0	0	0	0	0	0	0	0	0	0	1	droidkungfu	
0	0	0	0	0	1	0	0	0	0	1	droidkungfu	

Gambar 10. Contoh data dalam bentuk Vector

Pada gambar 10 memperlihatkan contoh dataset yang mana setiap atribut akan bernilai 1 apabila atribut tersebut ada pada aplikasi yang mengandung *malware*dan akan bernilai 0 apabila tidak terdapat pada aplikasi yang mengandung *malware*.

2.5 Seleksi Fitur

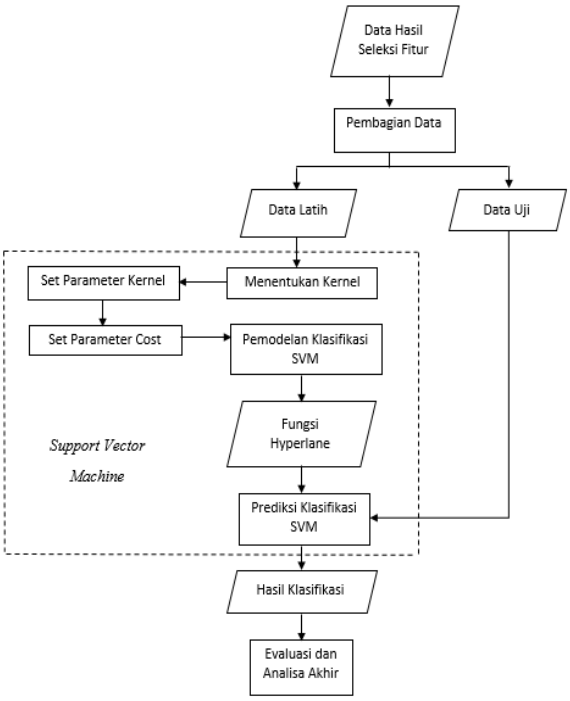
Dari beberapa atribut hasil extraksi data, beberapa di antaranya mubazir atau tidak relevan yang bisa mempengaruhi hasil kinerja dari algoritma *Mechine Learning*, maka dari itu digunakan *Seleksi Fitur* untuk mengurangi atribut yang tidak penting atau tidak relevan dalam data. *Seleksi Fitur* yang akan digunakan bisa di lihat pada tabel 1. Pada tabel 1 di ambil 3 terbaik dari penelitian yang sudah ada[11], teknik *Seleksi Fitur* yang akan di bandingkan memakai metode scoring untuk nominal ataupun pembobotan attribut kontinus yang didiskretkan menggunakan maksimasi entropy[11].

Tabel 1. Metode Seleksi Fitur Yang Di Gunakan

No	Nama Seleksi Fitur
1	Correlation-based Feature Selection (CFS)
2	Gain Ration (GR)
3	Chi-Square (CHI)

2.6 Perancangan Aplikasi

Hasil yang di dapat dari beberapa proses *Seleksi Fitur*akan di olah menggunakan *Algortima Klasifikasi SVM*. Dari penelitian yang sudah pernah di lakukan, SVM menunjukkan ketahanan dan kemampuan generalisasi yang lebih tinggi dibandingkan dengan algoritma yang lain [12]. SVM juga mempunyai akurasi klasifikasi yang lebih stabil dibandingkan dengan algoritma yang lain [13]. Algortima SVM menghasilkan nilai >90% pada deteksi *Malware* menggunakan *permission based dan API Calls* pada aplikasi *Android* [7]. Dari pernyataan sebelumnya maka algoritma SVM bisa digunakan untuk melihat hasil akurasi dari *SeleksiFitur*. Pertama data yang sudah dibagi menjadi empat kategori yaitu data hasil CFS, GR, CHI dan data original (DataSet tanpa seleksi fitur) masing-masing akan dibagi menjadi 70% Data Latih dan 30% Data Uji. Setelah setiap data kategori terbagi maka akan dilakukan proses Klasifikasi pada setiap kategori data. Proses kerja Klasifikasi bisa dilihat pada gambar 11.



Gambar 11. Proses Klasifikasi

Setelah hasil dari ketiga *seleksifitur* didapat maka akan dilakukan pembagian data yaitu sebanyak 70% untuk Data Latih dan 30% Untuk Data Uji. Sebelum melakukan proses klasifikasi pertama kita harus menentukan Parameter Kernel yang akan dipakai, disini peneliti akan menggunakan kernel Radial basis function (RBF) dikarenakan kernel ini cukup populer dan memberikan hasil yang stabil terhadap beberapa data yang sudah di uji pada penelitian lainnya. Setelah menentukan kernel maka selanjutnya akan menentukan Paramete Cost, default nilai Cost adalah 1.0 yang merupakan Low Tolerant dari fungsi Kernel dan pada penelitian lain nilai Cost tidak terlalu berpengaruh terhadap hasil prediksi dari klasifikasi, sehingga pada penelitian ini akan digunakanlah nilai default tersebut.

Setelah parameter Kernel dan Cost sudah ditentukan maka selanjutkan akan masuk kedalam proses Pemodelan Klasifikasi SVM. Dalam pemodelan klasifikasi data yang berbentuk non-*linier*akan dimasukan kedalam dimensi data yang baru yang berdimensi lebih tinggi sedemikian hingga data dapat terpisah secara *linier*. Setelah data-data dapat terpisah secara *linier* maka akan terbentuk *Hyperplane* yang mana berfungsi sebagai pemisah pola secara *linier* dengan memaksimalkan margin disekitar garis

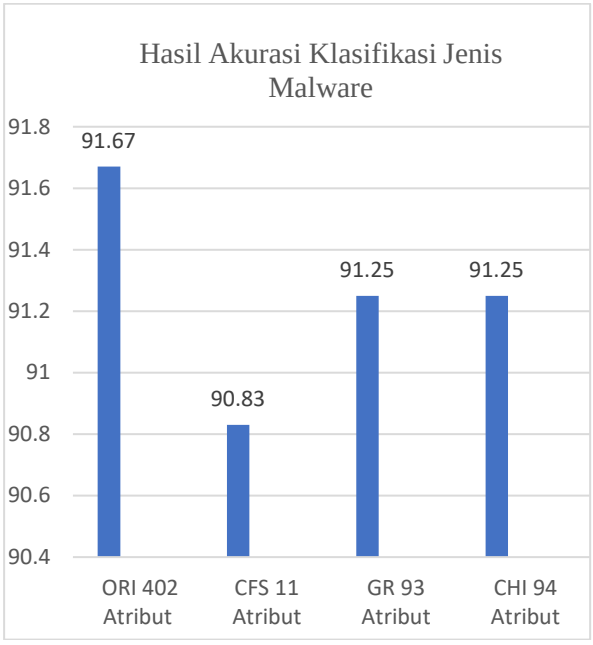
pemisah *Hyperplane*. Setelah *Hyperplane* sudah terbentuk maka bisa dilakukan proses klasifikasi dengan data yang baru untuk memprediksi kelas dari data tersebut. Dari hasil prediksi klasifikasi tersebut kita bisa membandingkannya dengan kelas yang sebenarnya, apakah data tersebut diprediksi dengan kelas yang benar ataupun salah yang mana setiap hasil prediksi tersebut akan kita analisa lebih jauh lagi sehingga akan menentukan apakah *permission* dan *brodacastreceiver* bisa digunakan dalam mengklasifikasikan jenis *Malware* serta mencari hasil prediksi *seleksifitur* mana yang lebih baik pada dataset *Malware* tersebut.

3. HASIL DAN PEMBAHASAN

Pengujian dilakukan dengan menghitung hasil yang diklasifikasikan secara benar yang mana nantinya akan dilakukan perhitungan akurasi dari klasifikasi. Perhitungan akurasi dibagi menjadi 4 bagian yaitu klasfikasi data Original(tanpa seleksi fitur), CFS, GR dan CHI.

Dalam proses klasifikasi disini digunakan *SVM Kernel RBF*sebagai Algortima Klasifikasnya. Untuk Mengukur evaluasi kinerja model klasifikasi pada peneliti ini hanya menggunakan akurasi dari hasil klasifikasi untuk menganalisa apakah *permission* dan *broadcastreceiver* bisa dipakai sebagai atribut dalam kelasifikasi jenis *Malware* Android dan seberapa besar pengaruh *seleksifitur* pada hasil klasifikasi.

$$Akurasi = \frac{jumlah\ yang\ di\ klasifikasi\ secara\ benar}{total\ testing\ sampel\ yang\ di\ uji}\ X\ 100\% \qquad (2)$$



Gambar 12. Perbandingan Hasil Klasifikasi Jenis Malware

Tabel 2. Hasil dari Klasifikasi algortima SVM

No	Jenis	Total Atribut	Benar	Akurasi
1	ORI	402	220	91.67 %
2	CFS	12	218	90.83 %
3	GR	93	219	91.25 %
4	CHI	94	219	91.25 %

4. KESIMPULAN

Berdasarkan hasil pengujian klasifikasi jenis *MalwareAndroid* yang menggunakan *Permission* dan *BroadcastReceiver* sebagai atribut dan dengan menggunakan algortima *Support Vector Machine* (SVM) dengan *Kernel RBF* menunjukan hasil akurasi yang cukup tinggi, yaitu dataset hasil seleksifitur CFS dengan jumlah 12 atribut menghasilkan nilai akurasi sebesar 90.83%, dataset GR dengan jumlah 93 atribut dan CHI dengan jumlah 94 atribut sama menghasilkan nilai akurasi sebesar 91.25% dan pada dataset Original(dataset tanpa seleksi fitur) yang mempunyai 402 atribut menghasilkan nilai akurasi sebesar 91.67%. Sehingga dapat disimpulkan dari akurasi tersebut bahwa *Permission* dan *BroadcastReceiver*,dapat digunakan dalam pengembangan klasifikasi jenis *MalwareAndroid* untuk kedepannya. Tetapi pada pengujian *Seleksi Fitur* tidak sesuai dengan yang diharapkan, karena hasil klasifikasi yang menggunakan *SeleksiFitur* berada sedikit dibawah hasil klasifikasi yang tidak menggunakan *SeleksiFitur* (data Original). Akan tetapi metode *SeleksiFitur* sendiri bisa mengurangi jumlah atribut yang akan dipakai dalam klasifikasi jenis *MalwareAndroid* dengan sangat signifikan sehingga dalam waktu proses bisa lebih cepat dan memberikan hasil yang hampir sama dengan data tanpa *Seleksifitur*.

5. DAFTAR PUSTAKA

[1] N. Yadav, A. Sharma, and A. Doegar, “A Survey on Android Malware Detection,” vol. 7, no. 12, pp. 47–53, 2016.

[2] www.appbrain.com, “Number of Android applications,” 2017. [Online]. Available: https://www.appbrain.com/stats/number-of-android-apps/. . [Accessed: 25-Jan-2017].

[3] J. Rivera and R. van der Meulen, “Gartner Says Annual Smartphone Sales Surpassed Sales of Feature Phones for the First Time in 2013,” Gartner Newsroom, 2014. [Online]. Available: http://www.gartner.com/newsroom/id/2665715.

[4] C. La and P. Myo, “Manifest Files Classification of,” vol. 2, no. 2, pp. 119–133, 2014.

[5] V. Wahanggara and Y. Prayudi, “Sistem Deteksi Malicious Software Berbasis System Call untuk Klasifikasi Barang Bukti Digital Menggunakan Metode Support Vector

Machine," SENTRA (Seminar Nas. Teknol. dan Rekayasa), no. July, pp. 1–8, 2015.

[6] S. P. Chorghe and N. Shekokar, "An Innovative Technique to Detect Malicious Applications in Android," Int. J. Sci. Res., vol. 4, no. 12, pp. 2013–2016, 2015.

[7] N. Peiravian and X. Zhu, "Machine learning for Android malware detection using permission and API calls," in Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI, 2013, pp. 300–305.

[8] S. Y. Yerima, S. Sezer, and G. McWilliams, "Analysis of Bayesian Classification-based Approaches for Android Malware Detection," Inf. Secur. IET, vol. 8, no. July 2013, pp. 25–36, 2014.

[9] S. Yerima and G. Mcwilliams, "Machine Learning based malware detection for Android using static analysis," no. January, 2012.

[10] E. Rasywir and A. Purwarianti, "Eksperimen pada Sistem Klasifikasi Berita Hoax Berbahasa Indonesia Berbasis Pembelajaran Mesin," J. Cybermatika, vol. 3, no. 2, pp. 1–8, 2015.

[11] T. Djatna and Y. Morimoto, "Pembandingan Stabilitas Algoritma Seleksi Fitur menggunakan Transformasi Ranking Normal," J. Ilmu Komput., vol. 6, no. 2, pp. 1–6, 2008.

[12] H. Yu, X. Huang, X. Hu, and H. Cai, "A Comparative Study on Data Mining Algorithms for Individual Credit Risk Evaluation," in 2010 International Conference on Management of e-Commerce and e-Government, 2010, pp. 35–38.

[13] D. Zhang, H. Huang, Q. Chen, and Y. Jiang, "A comparison study of credit scoring models," in Proceedings - Third International Conference on Natural Computation, ICNC 2007, 2007, vol. 1, pp. 15–18.