KLASIFIKASI CITRA PORNOGRAFI DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* PADA PERANGKAT *SMARTPHONE* BERBASIS ANDROID

Tugas Akhir
Untuk memenuhi sebagian persyaratan
mencapai derajat Sarjana S-1 Program Studi Teknik Informatika

Oleh :

Heru Mulyana

F1D 015 031

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MATARAM

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Saya yang bertanda tangan di bawah ini bahwa dalam Skripsi ini tidak terdapat karya yang pernah di ajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Mataram, 5 Juni 2020

Heru Mulyana

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabaraktuh

Segala puji bagi Allah SWT yang telah memberikan kami kemudahan sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan tepat waktu. Tanpa pertolongan-Nya penulis tidak akan sanggup menyelesaikan Tugas Akhir ini dengan baik. Shalawat serta salam semoga terlimpah curahkan kepada baginda tercinta Nabi Muhammad SAW yang telah membawa dari alam gelap menuju alam terang benderang. Penulis mengucapkan syukur kepada Allah SWT atas limpahan nikmat sehat-Nya, baik jasmani maupun rohani, sehingga penulis mampu menyelesaikan pembuatan Tugas Akhir dengan judul "Klasifikasi Citra Pornografi Dengan Metode *Convolutional Neural Network* Pada Perangkat Smartphone Berbasis Android".

Penulis tentu menyadari Tugas Akhir ini masih jauh dari kata sempurna dan masih terdapat kesalahan dan kekurangan didalamnya. Untuk itu, diharapkan kritik serta saran dari pembaca untuk Tugas Akhir ini, agar Tugas Akhir ini dapat menjadi lebih baik lagi. Penulis juga mengucapkan banyak-banyak terimakasih kepada semua pihak yang telah membantu penulis sehingga Tugas Akhir ini dapat selesai.

Demikian, semoga Tugas Akhir ini bermanfaat. Terimakasih

Mataram, 5 Juni 2020

Penulis

UCAPAN TERIMAKASIH

Tugas Akhir ini dapat diselesaikan berkat bimbingan dan dukungan ilmiah maupun materil dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis menyampaikan ucapan terimakasih yang setulus-tulusnya kepada :

1. Drs. H. Moh. Mugni dan Dwi Eriyanti, selaku orangtua, pemberi dukungan utama yang selalu memberikan do'a dan dukungan baik moril maupun materil yang tiada henti kepada penulis sehingga penulis dapat menyelesaikan pembuatan Tugas Akhir dengan baik.
2. Bapak Prof. I GP Suta Wijaya, S.T.,M.T.,D.Eng. selaku dosen pembimbing utama yang telah memberikan bimbingan dan arahan kepada penulis selama penyusunan Tugas Akhir sehingga dapat selesai dengan baik.
3. Bapak Ramaditia Dwiyansaputra S.T., M.Eng. selaku dosen pembimbing pendamping yang telah memberikan bimbingan dan arahan kepada penulis selama penyusunan Tugas Akhir sehingga dapat selesai dengan baik.
4. Bapak I Wayan Agus Arimbawa S.T., M.Eng. selaku dosen pembimbing pendamping yang telah memberikan bimbingan dan arahan kepada penulis selama penyusunan Tugas Akhir sehingga dapat selesai dengan baik.
5. Ibu Dr. Eng. Budi Irmawati, S.Kom., MT., selaku dosen pendamping yang telah memberikan bimbingan, arahan dan dukungan moril kepada penulis selama penyusunan Tugas Akhir sehingga dapat selesai dengan baik.
6. Semua pihak yang tidak dapat penulis sebutkan satu per satu, yang telah memberikan do'a dan dukungan baik moril maupun materil sehingga penulis dapat menyelesaikan pembuatan Tugas Akhir dengan baik.

Semoga Allah SWT selalu memberikan rahmat dan hidayah-Nya dan memberikan imbalan yang setimpal atas bantuan yang diberikan kepada penulis.

TUGAS AKHIR

KLASIFIKASI CITRA PORNOGRAFI DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* PADA PERANGKAT *SMARTPHONE* BERBASIS ANDROID

Oleh:

HERU MULYANA

F1D 015 031

Telah dipertahankan di depan Dewan Penguji
Pada tanggal 22 Mei 2020
dan dinyatakan telah memenuhi syarat mencapai derajat Sarjana S-1
Program Studi Teknik Informatika

Susunan Tim Penguji:

1. Penguji II

   Fitri Bimantoro, S.T., M.Kom. Tanggal: 4 Juni 2020
   NIP. 19860622 201504 1 002

2. Penguji III

   Royana Afwani, S.T., M.T. Tanggal: 4 Juni 2020
   NIP. 19850707 201404 2 001

3. Penguji III

   Arik Aranta S.Kom., M.Kom. Tanggal: 4 Juni 2020
   NIP. 199402202019031004

Mataram, Juni 2020
Dekan Fakultas Teknik
Universitas Mataram

Akmaluddin, S.T., M.Sc.(Eng.), Ph.D.
NIP. 19681231 199412 1 001

TUGAS AKHIR

KLASIFIKASI CITRA PORNOGRAFI DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK* PADA PERANGKAT *SMARTPHONE* BERBASIS ANDROID

Oleh:

HERU MULYANA

F1D 015 031

Telah diperiksa dan disetujui oleh:

1. Pembimbing Utama

Prof. I GP Suta Wijaya, S.T.,M.T.,D.Eng. Tanggal: 12 Mei 2020
NIP. 19731130 200003 1 001

2. Pembimbing Pendamping

Ramaditia Dwiyansaputra S.T., M.Eng Tanggal: 12 Mei 2020
NIP. -

Mengetahui
Ketua Program Studi Teknik Informatika
Fakultas Teknik
Universitas Mataram

Prof. I GP Suta Wijaya, S.T.,M.T.,D.Eng.
NIP. 19731130 200003 1 001

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Klasifikasi pornografi sulit di lakukan karna citra pornografi mempunyai kompleksitas yang sangat tinggi, variasi yang dapat terdiri dari warna kulit, jenis kelamin, bentuk tubuh, pose dan latar objek pornografi tersebut, oleh karena itu diperlukan pendekatan yang dapat mengklasifikasikan citra dengan tingkat variasi yang tinggi dengan mendapatkan informasi penting yang terdapat di dalamnya dengan cara di pecah menjadi beberapa lapisan-lapisan. Salah satu pendekatan tersebut adalah jaringan syaraf tiruan berjenis *Convolutional Neural Network* (CNN). Keuntungan dari metode CNN adalah kemampuannya untuk mempelajari hubungan yang tidak diketahui sebelumnya (*hidden pattern*) antara data *input* dan *output* dari setiap sistem. Hasil pembelajaran dari metode CNN adalah suatu model yang dapat digunakan untuk klasifikasi citra pornografi, model tersebut terdiri dari arsitektur yang dibentuk dan *weight* yang merupakan pengetahuan yang didapat dari hasil pembelajaran. Model tersebut akan di sematkan dalam *smartphone* berbasis Android sebagai *engine* untuk proses klasifikasi. Dengan demikian aplikasi untuk mendeteksi, menghapus citra pornografi dalam *smartphone* berbasis android yang di beri nama aplikasi *Porn Away*. Menurut hasil akhir penelitian, model menghasilkan akurasi sebesar 91%, tingkat presisi 91%, tingkat *recall* 92%, dan dengan waktu komputasi 6 *ms.*

Kata kunci: Andorid, *Convolutional Neural Network*, Pornografi.

ABSTRACT

The classification task of pornography is difficult to do because pornographic imagery has very high complexity and variations that can consist of skin color, gender, body shape, pose and background of pornographic objects, therefore an approach that can classify images with a high level of variation to get important information contained in it by breaking it into layers, one such approach is the Convolutional Neural Network (CNN). The advantage of the CNN method is its ability to study previously unknown relationships (hidden patterns) between *input* and *output* data from each system. Learning outcomes from the CNN method is a model that can be used for the classification task of pornographic images, the model consists of a formed architecture and weight which is a knowledge obtained from learning outcomes. The model will be installed on an Android-based smartphone as an engine for the classification process. Thus the application to detect, delete pornographic images in an android-based smartphone that is named the Porn Away. According to the final results of the research, the model produces an accuracy of 91%, a precision level of 91%, a recall rate of 92%, and with a computing time of 6 ms.

Keyword: Android, *Convolutional Neural Network*, *Pornography*.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring perkembangan zaman, teknologi dan komunikasi mengalami kemajuan yang sangat pesat, perlu kita ketahui juga bahwa dengan teknologi segala aktivitas bisa berjalan dengan mudah dan praktis sehingga membantu banyak untuk kegiatan kita ke depannya. Hal tersebut ditunjang dengan teknologi yang sangat mudah di jangkau oleh masyarakat sekarang ini, seperti *smarthphone*. Namun di satu sisi ada juga dampak negatifnya yaitu, saat ini banyak remaja yang salah menggunakan teknologi yang mendapatkan informasi hanya untuk kepentingan pribadi, dan dimanfaatkan untuk hal-hal yang tidak bertanggung jawab atas apa yang mereka kerjakan, seperti konten pornografi berupa gambar, sketsa, ilustrasi, foto, tulisan, suara, bunyi, gambar bergerak, animasi, kartun, percakapan, gerak tubuh, atau bentuk pesan lainnya melalui berbagai bentuk media komunikasi dan/atau pertunjukan di muka umum, yang memuat kecabulan atau eksploitasi seksual yang melanggar norma kesusilaan dalam masyarakat [1]. Hal tersebut dapat mempengaruhi sikap dan perilaku masyarakat khususnya anak di bawah umur untuk melakukan hal-hal yang tidak senonoh. Menurut hasil survei, penetrasi pengguna internet di Indonesia dalam rentang umur 5-9 tahun berkisar 25% sedangkan umur 10-14 tahun sebanyak 66% [2].

Menurut hasil survei pengguna internet di Indonesia, sebanyak 55,9% pengguna aktif internet pada segala rentang umur, mengalami secara tiba-tiba kemunculan konten porno pada konten yang sedang dikunjungi [2]. Hal tersebut menunjukkan lebih tingginya kemungkinan seorang pengguna melihat konten pornografi secara disengaja maupun tidak. Karena hal itu, dibutuhkan tindakan preventif untuk mengurangi konten pornografi.

Citra pornografi mempunyai kompleksitas yang sangat tinggi, variasi yang dapat terdiri dari warna kulit, jenis kelamin, bentuk tubuh, pose dan latar objek pornografi tersebut. Untuk itu perlu pendekatan yang dapat mengklasifikasikan citra dengan tingkat variasi yang tinggi dengan mendapatkan informasi penting yang terdapat di dalamnya dengan cara di pecah menjadi beberapa lapisan-lapisan, salah satu pendekatan tersebut adalah Jaringan Syaraf Tiruan (*Artificial Neural Network*). Metode ANN mempunyai kemampuan belajar dari contoh-contoh data yang ada atau yang di sebut pembelajaran diawasi (*supervised learning)*. Hal ini memungkinkan ANN dapat belajar sesuai dengan

banyaknya data yang kita sediakan, yang berarti memiliki potensi yang sangat besar untuk mempelajari data yang kompleks.

Pengembangan dari ANN dengan konvolusi sebagai ekstraksi fitur disebut *Convolutional Neural Network* (CNN). CNN merupakan salah satu metode *Machine Learning* (ML) yang dapat digunakan untuk mengklasifikasikan suatu citra digital. Metode CNN menggunakan lapisan yang lebih banyak dalam implementasinya, hal ini membuat metode CNN tergolong metode *Deep Learning* yang merupakan sub-bagian dari *Machine Learning* dimana kata "*Deep"* merujuk ke kedalaman atau tingkat lapisan Jaringan syaraf tiruan itu sendiri.

Pemilihan *smartphone* berbasis Android sebagai *platform* model CNN untuk mengenali citra pornografi ini berdasarkan kepopuleran Android dari sisi pengguna aktifnya yang mencapai lebih dari 1 miliar pengguna dari seluruh dunia [3], dan 124 juta pengguna aktif internet *smartphone* di Indonesia [4]. Karena tingginya pengguna *smartphone* Android, diharapkan sistem ini dapat menjangkau lebih banyak pengguna nantinya.

Hasil pembelajaran dari metode CNN adalah suatu model yang dapat digunakan untuk klasifikasi citra pornografi, model tersebut terdiri dari arsitektur yang dibentuk dan *weight* yang merupakan pengetahuan yang didapat dari hasil pembelajaran. model tersebut akan disematkan dalam *smartphone* berbasis Android sebagai *engine* untuk proses klasifikasi. dengan demikian aplikasi untuk mengenali dan menangani pornografi dalam *smartphone* berbasis android yang diberi nama *Porn Away* dapat diwujudkan, terlebih lagi, penelitian dalam membangun model klasifikasi citra pornografi dengan implementasi aplikasinya belum pernah dilakukan sebelumnya.

1.2 Rumusan Masalah

Berdasarkan pada latar belakang yang telah diuraikan, adapun permasalahan yang akan dikaji dalam penelitian ini adalah :

a. Bagaimana membangun sebuah arsitektur CNN untuk mengenali citra pornografi?
b. Bagaimana mengukur unjuk kerja metode CNN seperti akurasi, presisi, *recall* dan waktu komputasi dalam pengenalan citra pornografi?
c. Bagaimana membuat aplikasi untuk mengenali dan menangani pornografi dalam *smartphone* berbasis android?

1.3 Batasan Masalah

Berdasarkan pembahasan pada skripsi ini maka, diuraikan batasan-batasan masalah dalam pembangunan aplikasi ini yaitu:

1. Klasifikasi dibagi menjadi dua kelas yaitu *porn*, dan *non porn.*
2. Data yang digunakan adalah *KIAdataset* dari grup riset AI Program Studi Teknik Informatika Universitas Mataram.
3. Penyematan model dalam *smartphone* Android dilakukan dengan memanfaatkan *library* Tensorflow dan Keras.
4. Aplikasi menggunakan *smartphone* dengan sistem operasi Android minimal versi 8.0 (Oreo).
5. Aplikasi dirancang hanya untuk mencari, mengenali, mengkarantina, dan menghapus citra pornografi.

1.4 Tujuan

Adapun tujuan yang akan dicapai dari tugas akhir ini yaitu:

1. Membangun arsitektur CNN untuk mengenali citra pornografi.
2. Mengetahui akurasi, presisi, *recall* dan waktu komputasi metode CNN dalam pengenalan citra pornografi.
3. Mengembangkan aplikasi untuk mengenali dan menangani citra pornografi dalam *smartphone* berbasis Android.

1.5 Manfaat

Manfaat dari penelitian ini secara umum dapat diperoleh oleh dua subjek antara lain.

1. Bagi penulis
   a. Dapat menerapkan pengetahuan selama di perkuliahan terutama pengetahuan tentang pengenalan pola dan jaringan saraf tiruan.
   b. Dapat menambah pengetahuan di bidang *machine learning, computer vision* dan Android.
2. Bagi pembaca
   a. Memberikan pengetahuan baru tentang metode yang dapat digunakan dalam pengenalan citra pornografi.
   b. Dapat dijadikan sebagai rujukan untuk mengembangkan model *deep learning* agar mendapat performa yang lebih baik.

c. Menghasilkan aplikasi untuk mengenali dan menangani citra pornografi dalam *smartphone* berbasis Android untuk digunakan langsung oleh masyarakat.

1.6 Sistematika Penulisan

Sistematika penulisan dalam penyusunan tugas akhir ini adalah sebagai berikut:

a. Bab I Pendahuluan

   Bab ini menjelaskan dasar-dasar dari penulisan laporan tugas akhir, yang terdiri dari latar belakang permasalahan yang akan diangkat, perumusan masalah, batasan masalah yang dibahas dalam penelitian, tujuan penelitian, serta sistematika penulisan laporan tugas akhir.

b. Bab II Tinjauan Pustaka dan Dasar Teori

   Bab ini membahas tentang penelitian-penelitian terkait dengan topik penelitian yang dibahas kemudian teori-teori yang berhubungan dengan topik penelitian, pornografi, dan *Convolutional Neural Network* (CNN).

c. Bab III Metode Penelitian

   Bab ini membahas tentang metodologi yang digunakan dalam penelitian dan pengembangan perangkat lunak mulai dari tahap perancangan, *preprocessing*, *training* hingga pengujian serta jadwal yang dilakukan.

d. Bab IV Penjelasan

   Bab ini berisi hasil penelitian serta pembahasan dari penelitian yang telah dilaksanakan. Dalam bab ini disajikan gambar, tabel, serta grafik hasil penelitian.

e. Bab V Kesimpulan

   Bab ini merupakan kesimpulan dari keseluruhan pembahasan pada Bab sebelumnya terutama hasil penelitian yang telah dijelaskan pada Bab IV. Bab ini juga berisi saran untuk penelitian selanjutnya yang akan mengangkat tema yang sama dengan penelitian ini.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian yang terkait dengan pengenalan konten pornografi sudah banyak dilakukan karena dampaknya yang luas dan meresahkan masyarakat, terdapat penelitian-penelitian sebelumnya yang akan dijadikan sebagai rujukan dalam penelitian ini.

Dari referensi yang diperoleh, sebagian besar penelitian mencatatkan tingkat akurasi yang tinggi. Detail dari setiap penelitian dapat dilihat pada Tabel 2.1.

Tabel 2.1 Akurasi model *deep learning* menggunakan *convolution neural network.*

No	Penulis	Judul	Akurasi
1	Karen Simonyan (VGG)	Large Scale Visual Recognition Challenge 2014 (ILSVRC2014)	0.07405 (*Error Rate*)
2	M. Zufar dan B. Setiyono	Convolutional Neural Networks untuk Pengenalan Wajah Secara Real - Time	87.48%
3	H. Abhirawa, Jondri, dan A. Arifianto	Pengenalan Wajah Menggunakan Convolutional Neural Networks (CNN)	86.71%

Tabel 2.2 Akurasi kasus klasifikasi pornografi.

No	Penulis	Judul	Akurasi
1	I. G. P. Sutawijaya dan I. B. K. Widiartha	Pengenalan Citra Porno Berbasis Kandungan Informasi Citra (*Image Content*)	67.02%
2	J. A. M. Basilio, G. A. Torres, G. S. Pérez, L. K. T. Medina, dan H. M. P. Meana	Explicit Image Detection using YCbCr Space Color Model as Skin Detection	64.3%
3	Applying deep learning to classify pornographic images and videos	Mohamed N. Moustafa	92%-94%

Penelitian "Pengenalan Citra Porno Berbasis Kandungan Informasi Citra (*Image Content*)" menggunakan informasi warna dan ciri-ciri citra (*image signature*) yang diperoleh dengan cara teknik histogram warna dan mentransformasi-*wavelet*-kan citra

[5]. Hasil pengujian penelitian ini menunjukkan bahwa tingkat kesuksesan pengenalan citra porno menggunakan metode ini sebesar 67.02% (dapat mengenali sebanyak 84 citra sebagai citra porno dari 125 citra porno yang diuji) dan mendeteksi citra non porno menjadi porno sebanyak 36 citra dari 375 citra non porno (9.06 %).

Penelitian kedua mengusulkan suatu algoritma untuk mendeteksi gambar pornografi dalam gambar berwarna dengan menggunakan ruang warna YCbCr sebagai deteksi kulit dengan asumsi citra yang memiliki presentasi penampakan kulit yang banyak merupakan citra porno [6]. Deteksi tersebut berfungsi efektif untuk mengenali kulit akan tetapi menemukan beberapa kesalahan karena gambar kondisi pencahayaan ketika diambil, faktor lain itu bisa dengan interpretasi yang buruk terhadap sistem. Penelitian ini menghasilkan akurasi paling tinggi yaitu 64.3% pada *threshold* 50.

Kedua penelitian tersebut menunjukkan hasil yang kurang maksimal dalam mengenali citra porno. Pengenalan citra porno menjadi sulit karena citra porno memiliki tingkat heterogen yang tinggi seperti memiliki pose, warna latar belakang, diambil dari sudut kamera, warna kulit, dan etnis yang berbeda. Oleh karena itu, salah satu metode pengenalan yang saat ini banyak digunakan karena bisa beradaptasi dengan tingkat heterogenitas data yang tinggi yaitu metode *Convolutional Neural Network*.

Penelitian ketiga dengan metode CNN pada pengenalan wajah secara *realtime*. Penelitian ini menghasilkan akurasi 87.48% pada kondisi lingkungan yang dinamis dan membutuhkan waktu pengenalan yang sangat singkat karena dapat mengenali wajah secara *realtime* [7]. Selain itu terdapat penelitian pengenalan wajah dengan CNN dengan mengkombinasikan *dropout* dan ukuran filter sebagai parameter uji dan di dapatkan akurasi terbesar sebanyak 86.71% pada sistem yang menggunakan *dropout* dan ukuran filter 5x5 [8].

Penelitian keempat dengan kasus pornografi dilakukan oleh Mohammed Moustofa dari Department of Computer Science and Engineering, The American University Cairo dengan metode *deep learning* untuk klasifikasi nya [9], dengan menggunakan berbagai arsitektur yang sudah ada seperti AlexNet, GoogleNet, serta mengkombinasikannya, Hasil pengujian penelitian ini menunjukkan bahwa tingkat kesuksesan pengenalan citra porno menggunakan metode ini sebesar 92%-94% bervariasi antar arsitektur yang digunakan.

Penelitian kelima pada tahun 2014, Karen Simonyan dengan timnya VGG, model CNN miliknya berhasil menjuarai kompetisi ImageNet Large Scale Visual Recognition

Challenge 2014 (ILSVRC2012) *classification and localization task* dari 1000 kelas yg berbeda dengan 1,2 juta *datasets*. Prestasi tersebut menjadi momen pembuktian bahwa metode *Neural Network*. Detail tim dan abstraksi metode dari setiap peserta lomba dapat dilihat pada Lampiran 1.

Penelitian keenam adalah pengenalan wajah dengan metode CNN dengan dataset yang digunakan adalah The Extended Yale Face Database B, yang berupa dataset foto wajah. Dengan menggunakan proses dropout diperoleh hasil terbaik dengan tingkat akurasi pengenalan setinggi 89.73%. Sedangkan apabila dilakukan pengujian terhadap data testing akan diperoleh hasil akurasi pengenalan setinggi 75.79% [8].

Berdasarkan penelitian-penelitian yang sudah dilakukan sebelumnya, dapat dilihat bahwa *convolution neural network* dapat bekerja dengan baik untuk tugas klasifikasi citra khususnya dalam mengenali banyak kelas dengan heterogenitas yang tinggi. Pada kompetisi internasional, metode CNN berhasil mengungguli metode *Machine Learning* lainnya pada kasus klasifikasi citra yang memiliki heterogenitas yang tinggi [10]. Oleh karena itu, penulis bermaksud untuk menggunakan metode CNN untuk mengklasifikasikan citra pornografi, dengan memanfaatkan *KIA dataset* milik grup riset AI Program Studi Teknik Informatika Universitas Mataram.

2.2 Dasar Teori

2.2.1 Citra dan Pornografi

Citra digital merupakan representasi dari citra analog kontinu yang diubah ke dalam bentuk diskrit [11]. Suatu citra didefinisikan sebagai fungsi dua dimensi, f(x, y), dimana x dan y adalah koordinat spasial, dan amplitudo dari f pada tiap sembarang pasangan koordinat (x, y) disebut sebagai intensitas atau *gray* level di level tertentu. Ketika (x, y) dan nilai intensitas dari semua f terbatas dan nilainya diskrit maka disebut sebagai citra digital [12].

Pornografi adalah gambar, sketsa, ilustrasi, foto, tulisan, suara, bunyi, gambar bergerak, animasi, kartun, percakapan, gerak tubuh, atau bentuk pesan lainnya melalui berbagai bentuk media komunikasi dan/atau pertunjukan di muka umum, yang memuat kecabulan atau eksploitasi seksual yang melanggar norma kesusilaan dalam masyarakat [1].

Berdasarkan Undang-Undang No 44 tahun 2009 tentang pornografi melarang setiap orang untuk memproduksi, membuat, memperbanyak, menggandakan,

menyebarluaskan, menyiarkan, mengimpor, mengekspor, menawarkan, memperjualbelikan, menyewakan, atau menyediakan pornografi yang secara eksplisit memuat: a. persenggamaan, termasuk persenggamaan yang menyimpang; b. kekerasan seksual; c. masturbasi atau onani; d. ketelanjangan atau tampilan yang mengesankan ketelanjangan; e. alat kelamin; atau f. pornografi anak.

Video porno dapat mempengaruhi sikap dan perilaku masyarakat terutama kalangan remaja yang memiliki rasa penasaran tinggi di mana sikap dan perilaku tersebut dapat terjadi apabila terdapat dorongan dalam diri remaja untuk menyaksikan tayangan dan meniru hal-hal yang terdapat dalam video porno. Dari hasil penelitian yang telah dilakukan oleh Nurhayati didapatkan bahwa 96% siswa Sekolah Menengah pernah menonton video porno [13]. Hal ini juga hampir serupa dengan survei yang dilakukan oleh Komisi Nasional Perlindungan Anak pada tahun 2010 bahwa 97% remaja pernah menonton atau mengakses materi pornografi, 93% remaja pernah berciuman, 62,7% remaja pernah berhubungan badan dan 21% remaja Indonesia telah melakukan aborsi.

Seiring perkembangan zaman, teknologi dan komunikasi mengalami kemajuan yang sangat pesat, perlu kita ketahui juga bahwa konten pornografi saat ini lebih banyak di akses dalam bentuk citra digital.

2.2.2 *Convolutional Neural Network* (CNN)

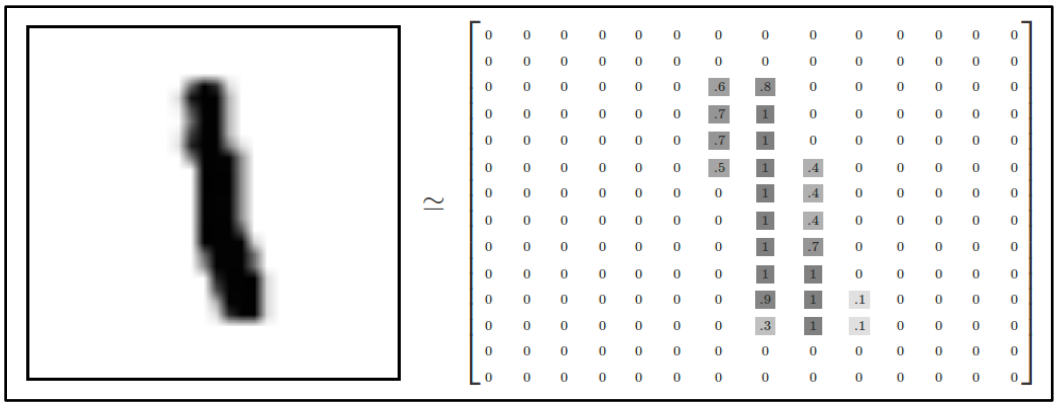
Convolution neural *network* merupakan salah satu jenis *neural network* yang bisa digunakan untuk klasifikasi citra. *Convolution* neural *network* bekerja dengan beberapa tahap dimana masukan dan keluaran dari tiap tahap terdiri dari beberapa *feature map*. Tiap tahapan terdiri dari tiga layer utama yaitu konvolusi, layer fungsi aktivasi, dan layer *pooling* [14].

Convolutional Klasifikasi *convolution neural network* dimulai dengan cara melakukan pelatihan terhadap masukan data latih. Masukan dari pembangunan model ini adalah citra dari data latih dengan ukuran 256 x 256. Citra masukan akan memasuki tahapan *feature learning* terlebih dahulu yang terdiri dari layer konvolusi dan layer *pooling*. Selanjutnya, tahapan yang dilakukan sebelum menentukan jenis objek adalah tahap klasifikasi yang terdiri dari *fully connected layer* dengan *dropout.*

1. *Input* citra

Citra yang masuk akan direpresentasikan ke dalam bentuk matriks citra 2 dimensi

seperti terlihat pada Gambar 2.1.



Gambar 2.1 Representasi citra menjadi matriks [15].

Kemudian matrix tersebut diperkecil nilai tiap *pixel*-nya dengan dibagi dengan 255 untuk normalisasi (*scaling*).

200	100	34	121
231	8	11	34
32	11	23	26
13	91	87	72

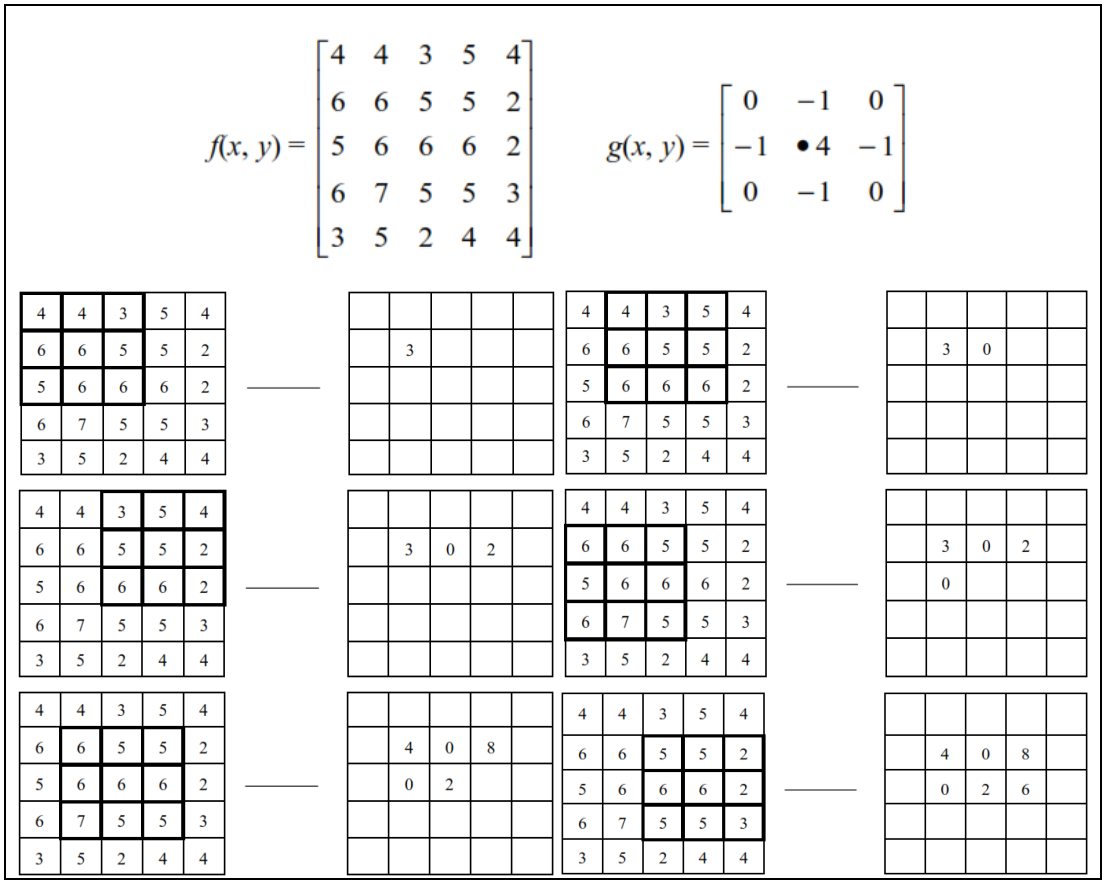
/255 =>

0.784	0.392	0.133	0.474
0.905	0.031	0.043	0.133
0.125	0.043	0.090	0.101
0.050	0.356	0.341	0.282

Gambar 2.2 *Scaling* matriks *input.*

2. *Convolution*

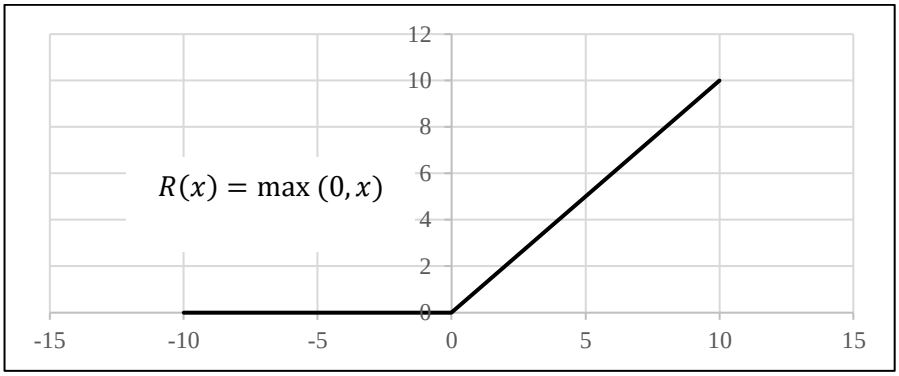
Pada layer konvolusi akan dilakukan proses konvolusi terhadap matriks citra masukan. Proses *convolution* bertujuan untuk melakukan filter terhadap matriks dari citra *input*. Kemudian untuk mempertahankan ukuran dari matriks, digunakan *zero padding*. Proses konvolusi ini menggunakan matriks *kernel* dengan ukuran 3x3. Keluaran dari layer konvolusi ini kemudian akan menjadi masukan pada layer *pooling.*



Gambar 2.3 Proses convolution.

3. Fungsi Aktivasi

Fungsi Aktivasi dalam jaringan saraf secara biologis, biasanya merupakan abstraksi yang mewakili laju aksi potensial pembakaran di dalam sel. Dalam bentuknya yang paling sederhana, fungsi ini adalah biner atau diantara ya dan tidak yaitu, apakah neuron itu ditembakkan atau tidak. Dalam jaringan saraf tiruan, fungsi aktivasi dari suatu simpul mendefinisikan keluaran dari simpul itu, atau "*neuron*" yang diberikan suatu *input* atau serangkaian *input*. *Output* ini kemudian digunakan sebagai *input* untuk simpul berikutnya dan seterusnya hingga ditemukan solusi yang diinginkan untuk masalah aslinya [16]. Pada hasil konvolusi, jika ditemukan adanya pola negatif dari perhitungan yang dihasilkan. diperlukan perhitungan tambahan untuk menghilangkan nilai negatif tersebut. Pada penelitian ini akan dilakukan beberapa pendekatan fungsi aktivasi.Fungsi aktivasi ReLU akan merubah index yang bernilai negatif menjadi 0. Sedangkan, untuk gambaran dari fungsi aktivasi ReLU dapat dilihat pada Gambar 2.4.



Gambar 2.4 Fungsi aktivasi ReLU.

Fungsi aktivasi ReLU digunakan karena fungsi aktivasi ini banyak digunakan pada penelitian menggunakan *convolution neural network* sebelumnya dan mendapat performa yang baik. Hasil dari fungsi aktivasi ReLU teradap hasil konvolusi adalah sebagai berikut.

	-0.816	2.165	0.816	0.310		0	2.165	0.816	0.310
	-0.498	2.412	0.498	0.310	ReLU	0	2.412	0.498	0.310
	-0.118	0.933	0.118	0.565	→	0	0.933	0.118	0.565
	-0.757	-0.545	0.090	0.773		0	0	0.090	0.773

Gambar 2.5 Penerapan fungsi aktivasi pada matriks hasil *convolution*.

4. *Max Pooling*

Layer *pooling* berguna untuk mengurangi ukuran citra sehingga proses *feature map* menjadi lebih cepat. *Pooling* pada penelitian ini menggunakan matriks 2x2 dengan *stride* sebesar 2. Artinya, *pooling* akan bergeser sebanyak 2 indeks dan mencari nilai terbesar dari *pooling* atau bisa disebut dengan istilah *max pooling.*

0	0.923	0.505	0.349
0.098	1	0.407	0.349
0.216	0.542	0.289	0.428
0.018	0.084	0.281	0.492

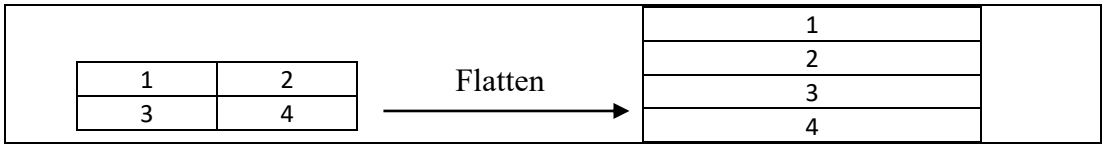
Max Pooling →

1	0.505
0.542	0.492

Gambar 2.6 Proses *max pooling.*

5. *Flatten*

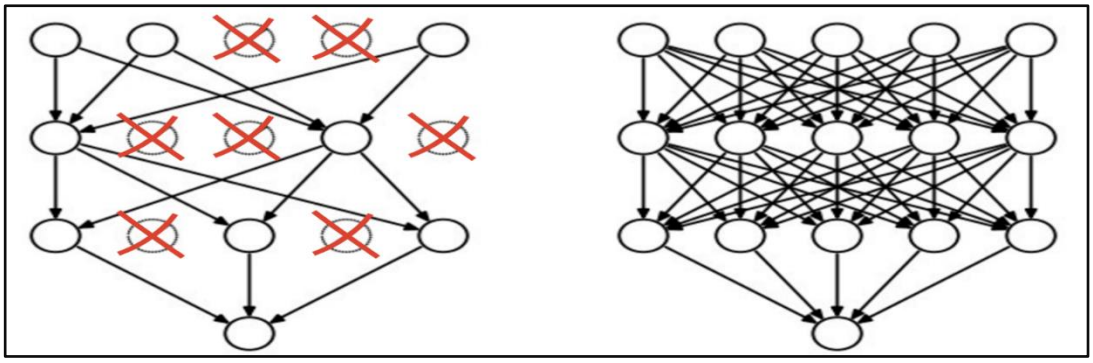
Matriks hasil *feature learning* (konvolusi dan *max pooling*) kemudian akan di *flatten* dimana tiap citra akan dirubah menjadi vektor. Keluaran dari *flatten* selanjutnya akan menjadi masukan pada klasifikasi dengan arsitektur *fully connected layer. Flatten* diterapkan untuk mengubah matriks menjadi vektor untuk menyesuaikan dengan format masukan pada *layer neural network*.



Gambar 2.7 Proses *flatten.*

6. *Fully Connected Layer* dengan *dropout*

Layer yang digunakan pada model yang dibangun berjumlah 2 layer dimana seluruh neuron akan terhubung secara penuh. Akan tetapi, model *fully connected layer* kemungkinan akan menimbulkan *overfitting.* Oleh karena itu, diterapkan *dropout* pada model ini untuk mengurangi hal tersebut.



Gambar 2.8 *Fully connected layer* dengan *dropout* [17].

Dropout akan memilih secara acak *neuron* mana yang akan di *nonaktif*-kan sehingga tidak terlalu banyak *weight* dan *neuron* yang terlibat di dalam perhitungan layer *neural network.* Probabilitas neuron yang dilibatkan dalam perhitungan akan di tetapkan misalnya antara 50% sampai 70%, *neuron* yang tidak aktif ditandai dengan tanda silang.

2.2.3 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis Linux yang mencakup sistem operasi, middleware, dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka [18]. Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya Open Handset Alliance, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel Android pertama mulai dijual pada bulan Oktober 2008.

Pada Google I/O developer conference 2017, CEO Google Sundar Pichai mengumumkan bahwa sistem operasi Android baru-baru ini melewati tonggak utama yaitu memiliki lebih dari 2 miliar perangkat aktif bulanan [3].

2.2.4 Arsitektur Android

Secara garis besar arsitektur android dapat dijelaskan dan digambarkan sebagai berikut [19]:

1. *Applications dan Widgets*

   Applications dan *Widgets* ini adalah layer dimana kita berhubungan dengan aplikasi saja, dimana biasanya kita download aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain- lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Applications Frameworks*

   Android adalah "*Open Development Platform*" yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi *resources*, menjalankan *service background*, mengatur alarm, dan menambahkan status *notifications*, dan sebagainya.

3. *Libraries*

   Libraries ini adalah layer dimana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya. Berjalan di atas *kernel*, layer ini meliputi berbagai library C/C++ inti seperti Libc dan SSL, serta libraries lainnya.

4. *Android Run Time*

   Layer yang membuat aplikasi Android dapat dijalankan dimana dalam prosesnya menggunakan Implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android.

5. *Linux Kernel*

   Linux *Kernel* adalah layer dimana inti dari operating system dari Android itu berada. Berisi file-file system yang mengatur *system processing*, *memory*, *resource*, *drivers,* dan system-sistem operasi android lainnya.

BAB III
METODE PENELITIAN

3.1 Bahan dan Alat Penelitian

Dataset yang digunakan dalam penelitian ini *adalah Kia Dataset* yang merupakan dataset dari grup riset AI program studi teknik informatika Universitas Mataram yang khusus digunakan untuk kebutuhan penelitian saja, data di dapatkan dengan teknik *wrangling* yakni pengumpulan gambar langsung berbagai sumber di internet, dataset ini terdiri atas 8950 citra untuk setiap kelasnya. Citra dari kelas citra porno dan tidak porno berformat bebas dengan resolusi terkecil 96x96 *px*. Distribusi *datasets* seperti pada Tabel 3.1.

Tabel 3.1 Distribusi *dataset*

Kelas	Jumlah Data	
	Latih	Uji
Porno	6,265	2,685
Tidak Porno	6,265	2,685

Alat yang digunakan dalam proses penelitian ini terbagi menjadi dua bagian yaitu:

1. Perangkat keras

   Perangkat keras yang digunakan dalam penelitian ini adalah komputer dengan spesifikasi sebagai berikut:

Tabel 3.2 Kebutuhan Perangkat Keras

No.	Nama Perangkat Keras	Spesifikasi
1	Prosesor	Intel® Xeon(R) E3-1225 @ 3.30GHz × 4
2	GPU	GeForce GTX 1050 (4GB VRAM)
3	*Smartphone*	Android Asus Zenfone Max Pro M2

2. Perangkat lunak

   Perangkat lunak yang digunakan dalam penelitian ini, yaitu:

Tabel 3.3 Kebutuhan perangkat lunak untuk membangun dan menguji sistem

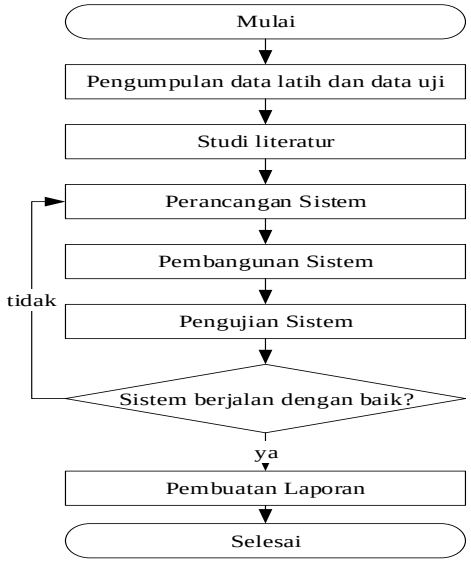
No.	Nama Perangkat Lunak	Spesifikasi
1	Sistem Operasi	Windows 10 64bit dan Linux Mint
2	Bahasa Pemrograman	Python 3.6
3	*Microsoft Off*ice	Office 2019
4	*Text Editor*	Visual Studio Code, JupyterLab
5	IDE Android	Android Studio 3.2

3.2 Studi Literatur

Guna mendukung berjalannya penelitian, studi literatur dilakukan dengan mempelajari buku-buku, jurnal penelitian serta sumber lain yang berkaitan dengan permasalahan yang diangkat. Adapun materi yang dipelajari dalam studi literatur berkaitan dengan jenis-jenis teknik sampling yang ada serta kelebihan dan kekurangan masing-masing metode serta penelitian-penelitian tentang *Convolutional Neural Network* serta akurasi pada masing-masing penerapan dan pengembangan aplikasi Android. Studi literatur ini dilakukan dengan mencari jurnal-jurnal dan *e-book* di internet serta membaca buku cetak.

3.3 Rancangan Penelitian

Diagram alir pembuatan sistem dari mulai pengumpulan data hingga pembuatan laporan tertuang dalam Gambar 3.1.



Gambar 3.1 Diagram alir pembuatan sistem

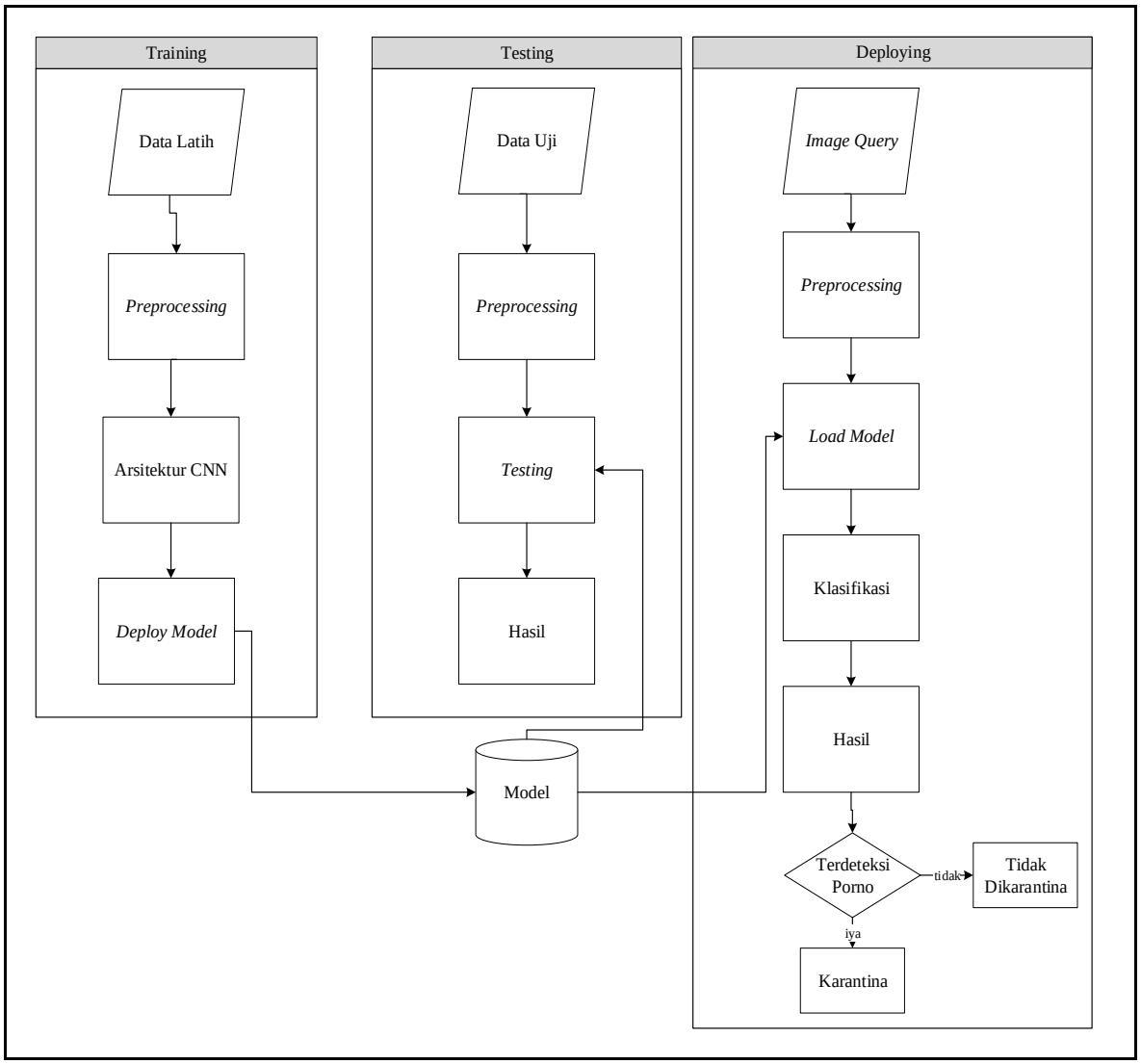
3.4 Pengumpulan Datasets

Langkah pertama dalam pembuatan sistem ini yaitu proses pengumpulan *dataset* berupa citra porno dan non-porno. Data tersebut merupakan *Kia Dataset* grup riset AI Program Studi Teknik Informatika Universitas Mataram. Citra pada *datasets* tersebut berjumlah 18354 untuk kedua kelas. Setiap citra dilakukan proses penyaringan dengan konfigurasi tertentu sehingga dihasilkan jumlah *datasets* baru sebanyak 8950 data yang akan di distribusikan menjadi data latih dan uji.

Langkah kedua yakni studi literatur untuk mempelajari cara membangun sistem sesuai dengan metode yang digunakan. Selanjutnya adalah tahap pembangunan sistem sesuai dengan rancangan yang telah dibuat. Tahap pengujian dilakukan untuk menguji apakah sistem berfungsi sesuai dengan tujuan, apabila belum sesuai maka langkah selanjutnya kembali ke studi literatur. Setelah sistem berhasil dibangun dan berjalan sesuai dengan fungsinya maka tahap terakhir yakni pembuatan laporan.

3.5 Rancangan Algoritma

Perancangan sistem ini dibuat dengan secara garis besar terdapat tiga proses utama yaitu *training* (pelatihan), *testing* (pengujian) dan *deploying* (penyematan). dapat dilihat pada Gambar 3.3.



Gambar 3.2 Proses training (pelatihan), testing (pengujian) dan deploying (penyematan)

Seperti terlihat pada Gambar 3.3, terdapat tiga proses utama dalam penelitian ini, yaitu proses pelatihan, pengujian dan penyematan. Proses tersebut dijelaskan sebagai berikut:

a. Proses Pelatihan

Proses pelatihan sistem meliputi tahap-tahap berikut ini:

1) Data citra pada Kiadatasets akan dilakukan proses pemilahan data.
2) Datasets akan diubah (*resize*) menjadi ukuran 150x150 px lalu disimpan untuk dijadikan data training.
3) Data training dimasukkan ke dalam arsitektur neural network dan dilakukan proses training/fitting.
4) Setelah proses training selesai maka model yang terbentuk akan di simpan.

b. Proses Pengujian

Proses Pengujian meliputi tahap-tahap berikut ini:

1) Distribusi datasets yang berperan sebagai data uji yang tidak digunakan dalam proses pelatihan akan digunakan sebagai data pengujian.
2) Model yang telah disimpan akan di load sebagai classifier yang akan di gunakan untuk mengklasifikasikan data uji tersebut.
3) Hasil prediksi keseluruhan digunakan untuk nilai dari metode tertentu.

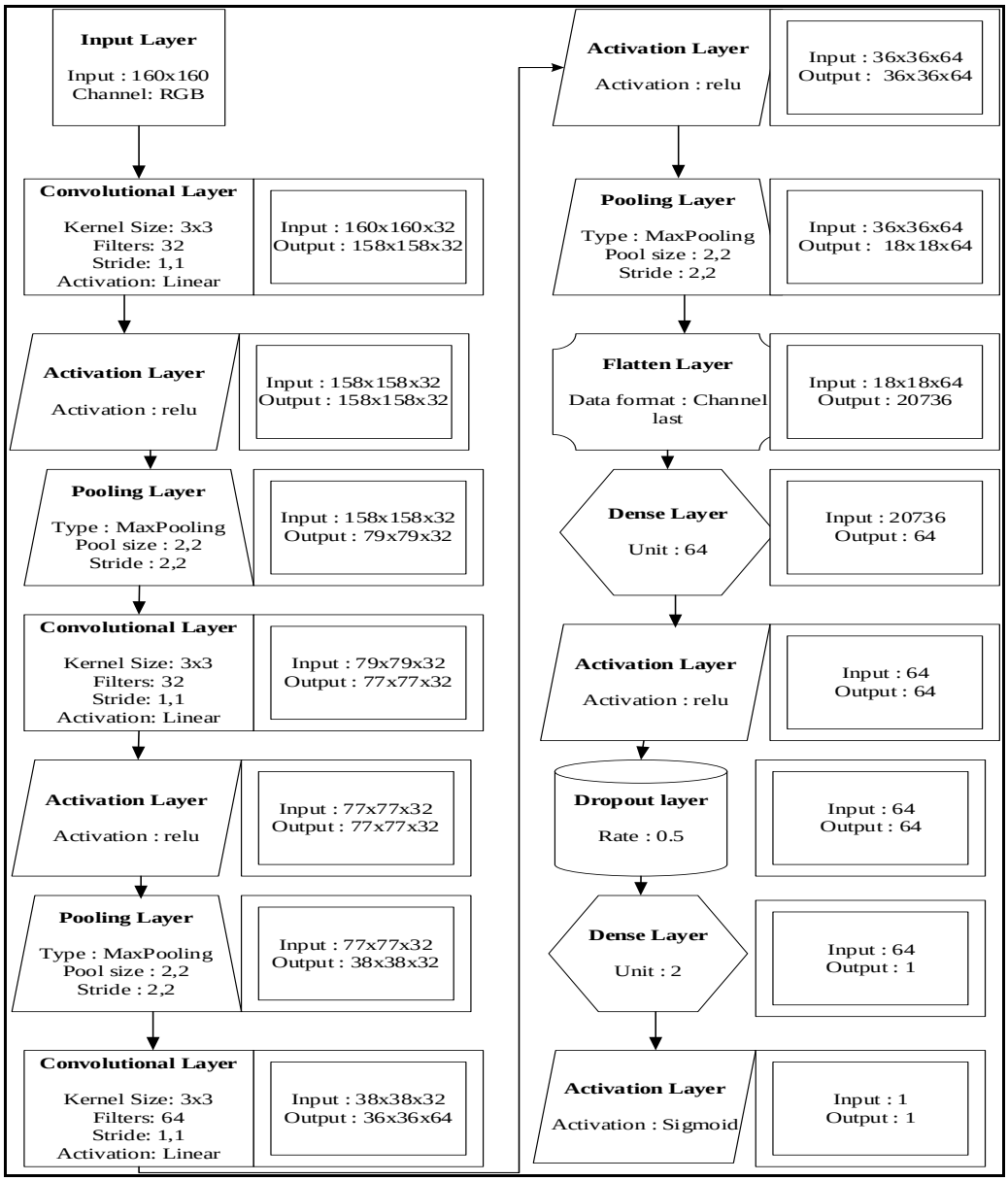
c. Proses *Deploying*

Proses *Deploying* meliputi tahap-tahap berikut ini:

1) Model dari proses *training* akan disematkan pada aplikasi Android
2) Model yang telah disimpan akan di load sebagai engine classifier yang akan di gunakan untuk mengklasifikasikan citra pornografi.
3) Jika hasil klasifikasi menunjukan kelas citra pornografi maka citra tersebut akan di karantina yakni akan di pindah ke tempat khusus dan di enkripsi agar pengguna yang tidak mempunyai hak akses aplikasi, contohnya anak-anak, tidak dapat mengakses citra pornografi tersebut.

3.5.1 *Training*

Tahap *Training* merupakan tahap pembentukan model klasifikasi dengan melatih model tersebut dengan data-data yang ada. Proses *training* tersebut dapat mengenali suatu kelas dengan mempelajari ciri atau karakteristik dari data latih yang digunakan. Proses *training* di lakukan dengan memanfaatkan fungsi *build in* dari *Framework* Tensorflow dan Keras. Berikut adalah contoh arsitektur CNN yang akan digunakan pada sistem:



Gambar 3.3 Arsitektur neural network pada sistem.

Arsitektur yang akan digunakan dalam jaringan ini terdiri dari 15 *Layer* menggunakan 3 *Convolution, 3 Pooling,* 5 *Activation* ,1 *Dropout*,2 *Dense*,1 *Flatten*. Data citra masukan berbentuk 160x160x3 dan akan menghasilkan 2 Kelas yaitu *Porn* dan *NonPorn*.

Selanjutnya, dilakukan konfigurasi terhadap model CNN yang dibangun untuk tahap *training*. Konfigurasi ini dilakukan dengan berbagai parameter sebagai berikut:

a. Ukuran *kernel* konvolusi (*kernel* 3x3 dan *kernel* 5x5)
b. Jumlah Layer Konvolusi *neural network* (2 Layer Konvolusi, dan 3 Layer Konvolusi)
c. Ukuran *neuron hidden layer* (32 *neuron*, 64 *neuron*, dan 96 *neuron*)

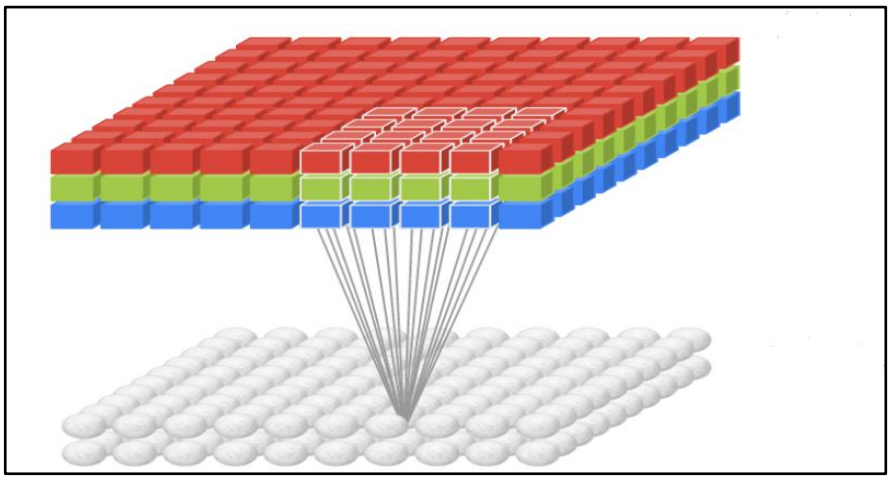
d. Ukuran citra masukan (96x96, 160x160, dan 224x224)

e. Dimensi citra (RGB dan *Grayscale*)

1. Ekstraksi Fitur

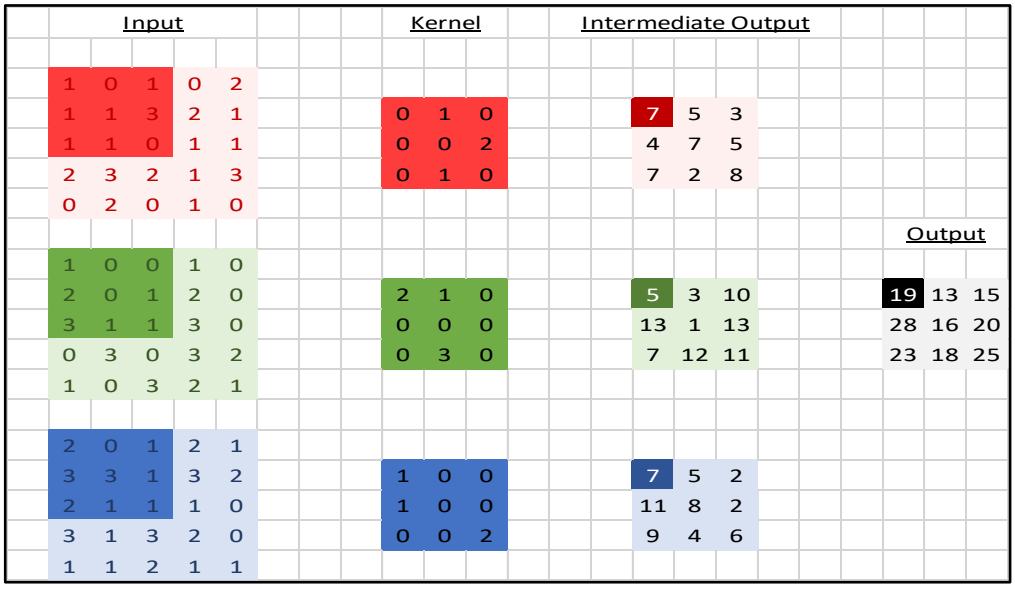
   Ekstraksi fitur merupakan proses untuk melakukan *encoding* atau merubah sebuah citra menjadi *features* yang berupa angka-angka yang merepresentasikan citra tersebut (*Feature Extraction*). *Ekstraksi fitur* terdiri dari dua bagian. *Convolutional Layer* dan *Pooling Layer*.

   1) *Convolutional Layer*



Gambar 3.4 Pemisahan dimensi warna pada citra [20].

Gambar 3.4 adalah RGB (*Red, Green, Blue*) *image* berukuran 32x32 *pixels* yang sebenarnya adalah *multidimensional array* dengan ukuran 32x32x3 (3 adalah jumlah *channel*). *Convolutional layer* terdiri dari *neuron* yang tersusun sedemikian rupa sehingga membentuk sebuah *filter* dengan panjang dan tinggi (*pixels*). Sebagai contoh, *layer* pertama pada *feature extraction layer* biasanya adalah *conv. Layer*. Ketiga *filter* ini akan digeser keseluruh bagian dari gambar. Setiap pergeseran akan dilakukan operasi “dot” antara *input* dan nilai dari *filter* tersebut sehingga menghasilkan sebuah *output* atau biasa disebut sebagai *activation map* atau *feature map*.



Gambar 3.5 Proses konvolusi 3 *channel.*

a. *Stride*

Stride adalah parameter yang menentukan berapa jumlah pergeseran *filter*. Jika nilai *stride* adalah 1, maka *conv. filter* akan bergeser sebanyak 1 *pixels* secara horizontal lalu *vertical*. Pada ilustrasi diatas, *stride* yang digunakan adalah 2.

Semakin kecil *stride* maka akan semakin detail informasi yang kita dapatkan dari sebuah *input*, namun membutuhkan komputasi yang lebih jika dibandingkan dengan *stride* yang besar.

Namun perlu diperhatikan bahwa dengan menggunakan *stride* yang kecil kita tidak selalu akan mendapatkan performa yang bagus.

b. *Padding*

Sedangkan *Padding* atau *Zero Padding* adalah parameter yang menentukan jumlah *pixels* (berisi nilai 0) yang akan ditambahkan di setiap sisi dari *input*. Hal ini digunakan dengan tujuan untuk memanipulasi dimensi *output* dari *conv. layer* (*Feature Map*). Tujuan dari penggunaan *padding* adalah:

Dimensi *output* dari *conv. layer* selalu lebih kecil dari *input*-nya (kecuali penggunaan 1x1 filter dengan *stride* 1). *Output* ini akan digunakan kembali sebagai *input* dari *conv. layer* selanjutnya, sehingga makin banyak informasi yang terbuang.

Dengan menggunakan *padding*, kita dapat mengatur dimensi *output* agar tetap sama seperti dimensi *input* atau setidaknya tidak berkurang secara drastis.

Sehingga kita bisa menggunakan *conv. layer* yang lebih dalam/*deep* sehingga lebih banyak *features* yang berhasil di-*extract*.

Meningkatkan performa dari model karena *conv. filter* akan fokus pada informasi yang sebenarnya yaitu yang berada diantara *zero padding* tersebut. Pada ilustrasi diatas, dimensi dari *input* sebenarnya adalah 5x5, jika dilakukan *convolution* dengan *filter* 3x3 dan *stride* sebesar 2, maka akan didapatkan *feature map* dengan ukuran 2x2. Namun jika kita tambahkan *zero padding* sebanyak 1, maka *feature map* yang dihasilkan berukuran 3x3 (lebih banyak informasi yang dihasilkan). Untuk menghitung dimensi dari feature map kita bisa gunakan rumus seperti dibawah ini:

$$output = \frac{W-N+2P}{S} + 1 \qquad (3\text{–}1)$$

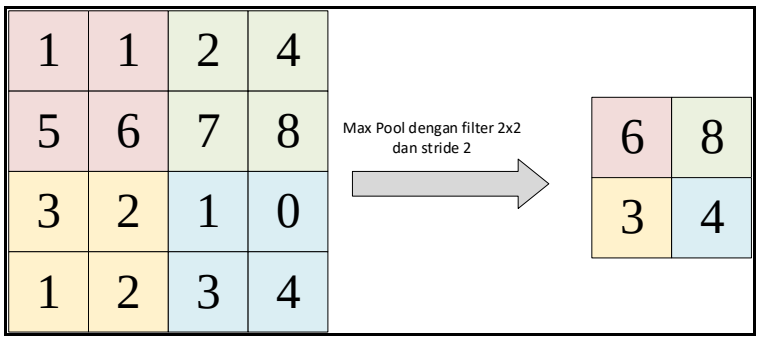
Keterangan:

- W = Panjang/Tinggi *Input*
- N = Panjang/Tinggi *Filter*
- P = *Zero Padding*
- S = *Stride*

2) *Pooling Layer*

Pooling layer biasanya berada setelah *conv. layer*. Pada prinsipnya *pooling layer* terdiri dari sebuah *filter* dengan ukuran dan *stride* tertentu yang akan bergeser pada seluruh area *feature map*.

Pooling yang biasa digunakan adalah *Max Pooling* dan *Average Pooling*. Sebagai contoh jika kita menggunakan *Max Pooling* 2x2 dengan *stride* 2, maka pada setiap pergeseran *filter*, nilai maximum pada area 2x2 *pixel* tersebut yang akan dipilih, sedangkan *Average Pooling* akan memilih nilai rata-ratanya.



Gambar 3.6 Proses *maxpool* dengan 2x3 *filter* dan 2 *stride*

Dimensi *output* dari *Pooling layer* juga menggunakan rumus yang sama seperti pada *conv. Layer.* Tujuan dari penggunaan *pooling layer* adalah mengurangi dimensi dari *feature* map (*downsampling*), sehingga mempercepat komputasi karena parameter yang harus di-*update* semakin sedikit dan mengatasi *overfitting*.

3.5.2 *Testing*

Proses *testing* yang dilakukan dalam penelitian ini bertujuan untuk menguji model yang akan di sematkan pada aplikasi. Pengujian dilakukan dengan memanfaatkan data uji pada *KIAdatasets*.

Tahap *testing* ini bertujuan untuk mengetahui akurasi yang terdiri dari akurasi latih dan uji, waktu komputasi yang terdiri dari waktu proses latih dan waktu proses klasifikasi serta kesenjangan antara akurasi latih dan uji. Adapun metode yang digunakan untuk melakukan pengujian tersebut adalah *confusion matrix*, *precision, recall* dan *time computing.*

1. Pengujian Aplikasi
   a. Kuisioner MOS (Mean Opinion Score)

      Kuesioner adalah suatu teknik pengumpulan informasi yang memungkinkan untuk mempelajari karakteristik dari aplikasi yang telah dibuat. Adapun standar penilaian MOS dapat dilihat pada Tabel 3.4. Kuesioner dilakukan untuk mengetahui *usability* dari aplikasi *Porn away* yang akan menghasilkan nilai tolak ukur aplikasi dari segi kinerja, keandalan, fitur, dan estetika, kuesioner akan menjadi acuan untuk perbaikan sistem.

Tabel 3.4 Tabel tingkatan dalam MOS

Nilai MOS	Tingkat Kepuasan
5	Sangat baik
4	Baik
3	Cukup Baik
2	Tidak Baik
1	Buruk

Pada saat pengujian, kuesioner dibagikan kepada 30 responden sebagai pengguna aplikasi untuk mengetahui bagaimana sistem bekerja saat digunakan oleh pengguna, apakah target atau tujuan sistem dapat dicapai dengan baik atau tidak.

2. Pengujian Model

a. *Confusion matrix*

Conffusion matrix merupakan metode yang digunakan untuk melakukan evaluasi terhadap model yang dihasilkan pada tahap *training*. Model *confusion matrix* yang digunakan pada penelitian ini adalah *confusion matrix* 2x2 yang terdiri dari *actual class* dan *predicted class*

Tabel 3.5 *Confussion Matrix*[21].

		Predict class	
		Non-Porn	**Porn**
Actual class	**Non-Porn**	True Negative	False Positive
	Porn	False Negative	True Positive

Perhitungan akurasi model dapat dilakukan dengan menggunakan persamaan [21].

$$akurasi = \frac{TP+TN}{TP+TN+FP+FN} \quad (3–2)$$

b. *Precision*

Precision adalah tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem, *precison* berguna untuk di gunakan saat harga atau dampak dari prediksi *false positive* sangat tinggi, dimana dalam sistem ini jika citra tidak porno di deteksi sebagai citra porno akan merugikan pengguna, dengan kata lain nilai *precision*-nya rendah. Perhitungan *precision* pada model dapat dilakukan dengan menggunakan persamaan

$$Precision = \frac{True\ Positive}{True\ Positive+False\ Positive} \quad (3–3)$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

c. *Recall*

Recall adalah tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi. *Recall* berguna untuk menghitung seberapa banyak *actual positive* yang model hasilkan dengan melabelinya sebagai *true positive*, *recall* digunakan saat harga atau dampak dari prediksi *false negative* bernilai tinggi, dimana citra porno di prediksi sebagai citra bukan porno. Perhitungan *recall* pada model dapat dilakukan dengan menggunakan persamaan.

$$Recall = \frac{True\ Positive}{True\ Positive+False\ Negative} \quad (3–4)$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

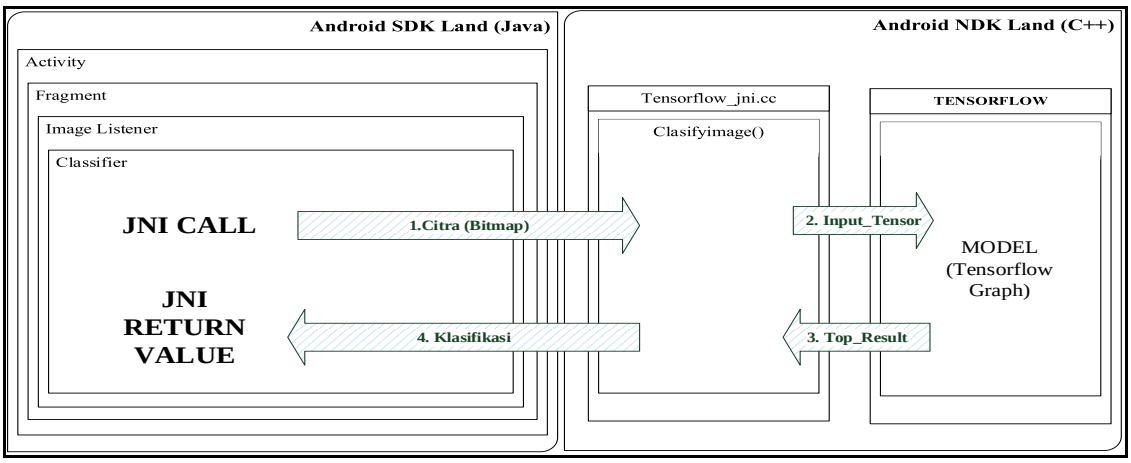
d. *Computation Time*

Metode ini bertujuan untuk mengukur waktu yang dibutuhkan oleh model untuk melakukan klasifikasi suatu citra, terhitung dari proses pemasukan citra sampai hasil klasifikasi didapat, dengan demikian didapatkan waktu komputasi (*computation time*) dari model yang di bangun.

3.5.3 *Deploying*

Pada tahap ini model atau *engine* yang sudah dibuat akan disematkan pada aplikasi berbasis Android yang di kembangkan dengan Bahasa Java, digunakan untuk proses klasifikasi konten negatif.

Proses klasifikasi pada aplikasi ini dirancang secara mandiri (*standalone)* dimana proses klasifikasi berjalan pada perangkat pengguna saja (*client side*) tanpa membutuhkan koneksi internet atau ketergantungan pada layanan tertentu, aplikasi dapat melakukan klasifikasi pada citra dengan memanfaatkan *engine* atau model yang telah dibuat dari proses pelatihan. Aplikasi ini memanfaatkan *library* dari Tensorflow untuk proses klasifikasi dengan menggunakan model CNN yang telah dibuat. Cara kerja klasifikasi model pada Android dapat digambarkan seperti Gambar 3.8.



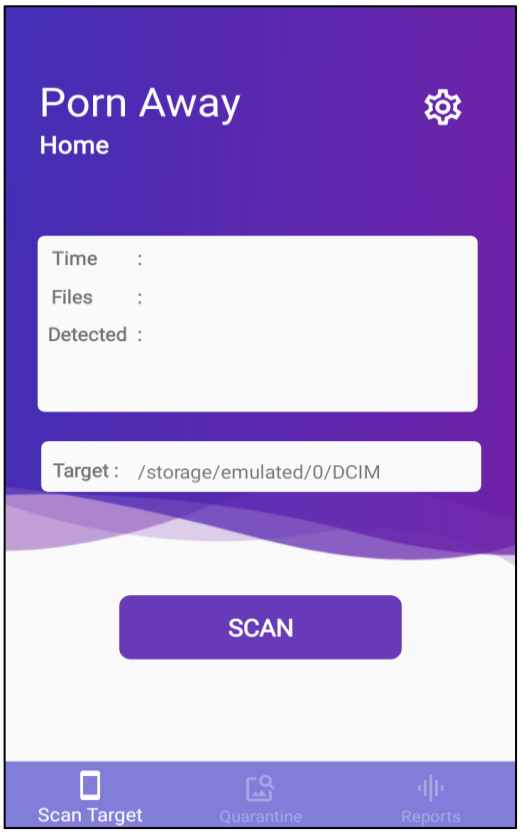
Gambar 3.7 Proses kerja klasifikasi model pada Android

Proses klasifikasi dimulai dengan mendapatkan citra (bitmap) dengan antarmuka aplikasi yang berada pada ranah Android SDK (java) dan melakukan proses normalisasi pada citra, selanjutnya citra akan di masukan pada fungsi *Classifyimage* dari *library* Tensorflow yang terdapat pada ranah Android NDK (C++), fungsi klasifikasi akan memuat model CNN yang telah di buat sebelumnya pada tahap training serta

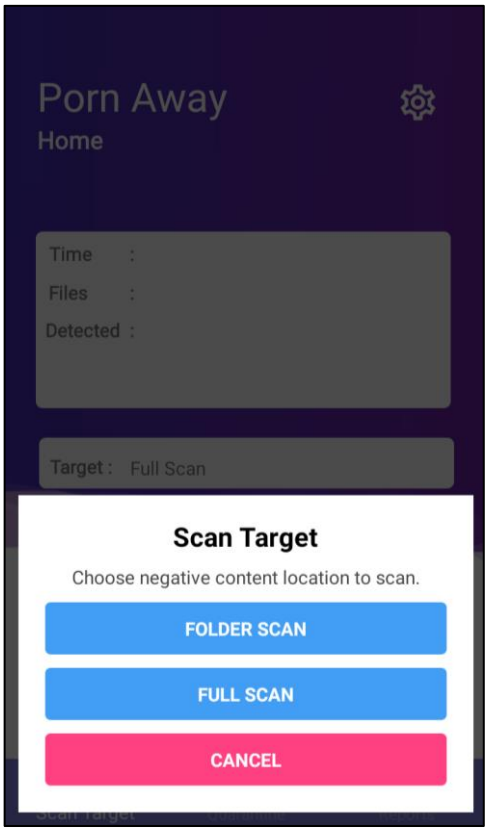
mengirimkan *input_tensor* yang merupakan *node* awal arsitektur CNN yang dibutuhkan untuk memulai proses klasifikasi, hasil proses klasifikasi adalah kelas porno atau tidak porno yang akan di tampilkan pada antarmuka aplikasi.

3.6 Perancangan Desain Sistem

Perancangan desain sistem pada sistem klasifikasi citra pornografi dengan metode CNN pada perangkat *smartphone* berbasis android ini dilakukan untuk mendefinisikan kebutuhan-kebutuhan sistem yang akan dibuat. Sistem akan dibuat menggunakan perangkat lunak Android Studio. Perancangan desain sistem yang dibuat dapat dilihat pada Gambar 3.8, Gambar 3.9, dan Lampiran.



Gambar 3.8 Tampilan Menu Utama Sistem



Gambar 3.9 Tampilan Pilihan Fitur *Scan* Sistem

BAB IV
HASIL DAN PEMBAHASAN

4.1. *Sample dataset*

Sebagaimana diuraiakan di Bab III bahwa pembuatan model CNN menggunakan *KiaDataset* yang berjumlah 18354 citra yang terdiri dari 2 kelas yaitu kelas citra porno sebanyak 9295 dan tidak porno sebanyak 9059. *dataset* tersebut berasal dari grup riset AI Program Studi Teknik Informatika Universitas Mataram. Contoh sampel dari kedua kelas dataset *KiaDataset* disajikan pada Gambar 4.1 dan 4.2.

(a) (b) (c)

Gambar 4.1 KiaDataset kelas Porno.

(a) (b) (c)

Gambar 4.2 KiaDataset Tidak Porno.

4.2. Mekanisme penelitian

Beberapa konfigurasi diterapkan pada model CNN untuk menentukan hasil terbaik. Tahap pertama adalah penyaringan dataset yakni hanya dengan memilih citra dengan resolusi di 96x96 *pixel* ke atas untuk menjaga kualitas *training*. Hal ini dilakukan karena masukan resolusi citra terkecil untuk *training* pada penelitian ini adalah 96x96 *px,* sehingga didapatkan 17900 citra yang terdiri dari 8950 citra porno dan 8950 citra tidak porno.

Selanjutnya, dilakukan pengujian terhadap model *CNN* dengan berbagai parameter pengujian sebagai berikut:

a. Ukuran *kernel* konvolusi (*kernel* 3x3 dan *kernel* 5x5)

b. Jumlah Layer Konvolusi *neural network* (2 Layer Konvolusi, dan 3 Layer Konvolusi)

c. Ukuran *neuron hidden layer* (32 *neuron*, 64 *neuron*, dan 96 *neuron*)

d. Ukuran citra masukan (96x96, 160x160, dan 224x224)

e. Dimensi citra (RGB dan *Grayscale*)

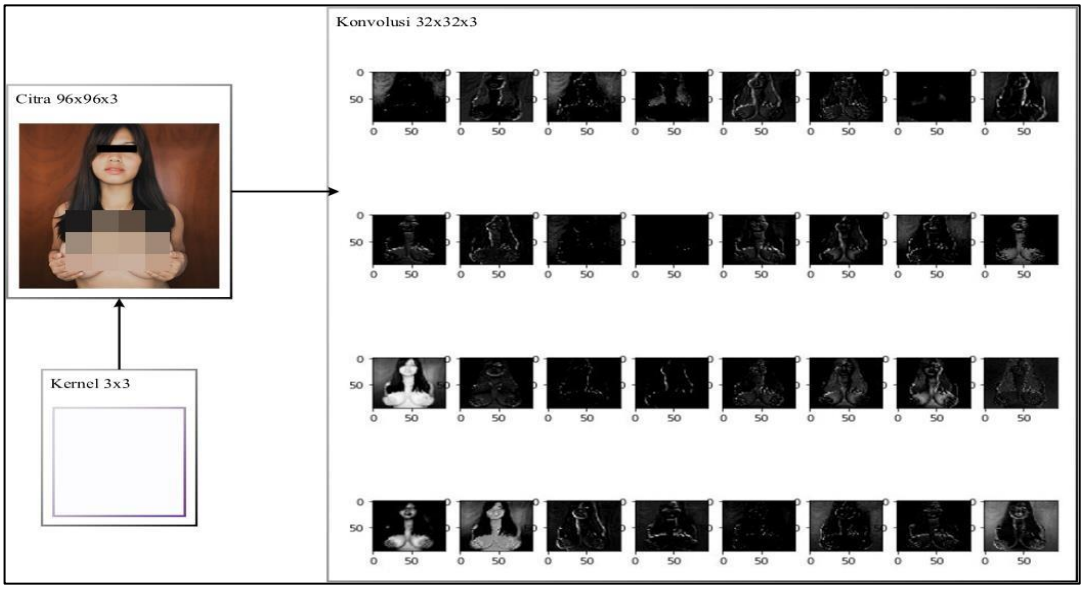
Adapun parameter-parameter awal yang digunakan untuk pengujian tersebut adalah 2 layer konvolusi, *kernel* 3x3, dengan *neuron* sebanyak 32, ukuran citra 96x96 dan citra masukan RGB. Penentuan parameter tersebut ditentukan berdasarkan nilai terkecil dari parameter pengujian.

Citra yang sudah di *pre-processing* di bagi menjadi 2 bagian yaitu citra *training* dan *citra testing* dengan distribusi 70 banding 30 hal ini ditentukan karna jumlah *dataset* yang ada cukup banyak untuk metode *deep learning*

Model CNN dengan hasil pengujian terbaik akan digunakan sebagai *engine* klasifikasi pada aplikasi *porn away* di *platform* android, performa aplikasi dalam mendeteksi citra porno akan di uji menggunakan metode *Mean Opinion Score* (MOS) yang melibatkan 30 responden dalam pengujiannya.

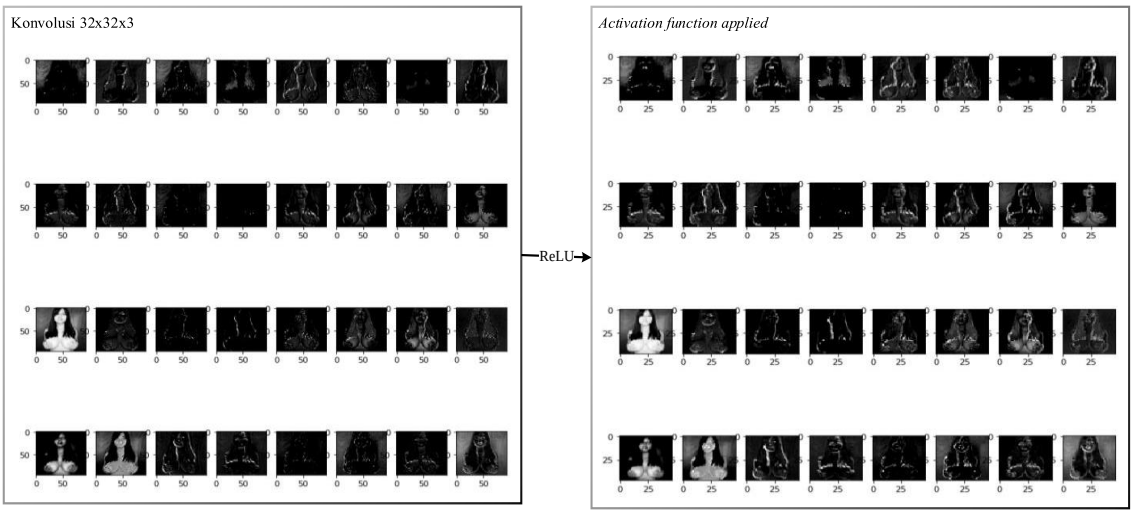
4.3. Arsitektur CNN

Lapis (*Layer*) pertama pada CNN adalah konvolusi. Semua nilai *pixel* citra inputan berbentuk *vector* akan langsung dimasukan ke dalam layer konvolusi. Pada penelitian ini digunakan 2 variasi jumlah layer konvolusi yaitu 2 dan 3 *layer* dengan variasi *neuron* 32,64, dan 96. Sedangkan untuk ukuran *kernel* digunakan 2 variasi ukuran yaitu *kernel* 3x3 dan 5x5. Pada saat proses layer konvolusi diterapkan *zero padding* untuk mengganti *pixel* citra yang hilang menjadi nilai 0.



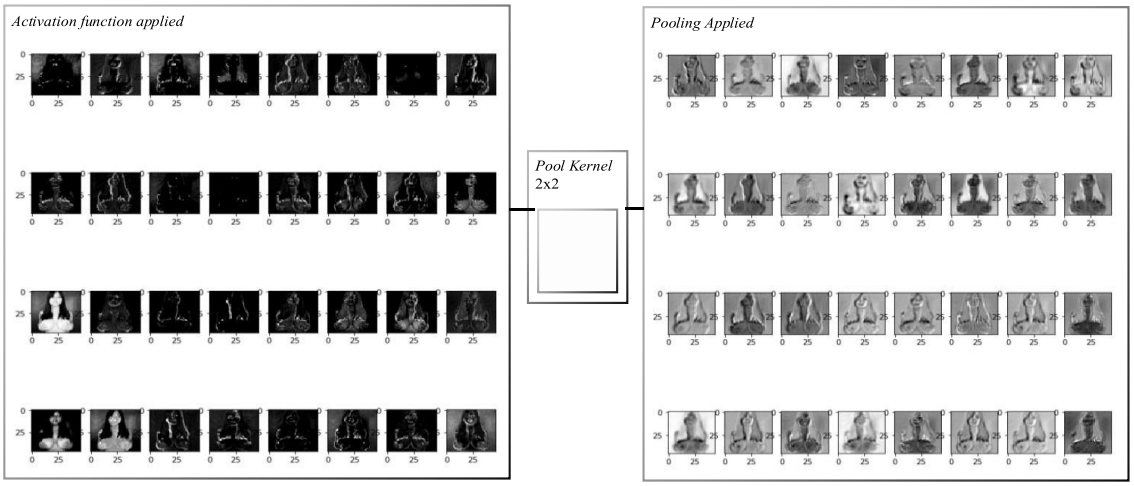
Gambar 4.3 Proses konvolusi

Selanjutnya, *Activation function* berjenis ReLU akan di terapkan pada hasil layer konvolusi bertujuan untuk mengenalkan non-linearitas pada *neural network*. *Activation function* ini bekerja dengan mengubah nilai yang bernilai kurang dari 0 menjadi 0 sehingga tidak ada nilai indeks citra yang bernilai minus (-) dan nilai yang lebih dari 0 menjadi angka itu sendiri.



Gambar 4.4 Proses *Activation Function* ReLU

Tahap selanjutnya adalah *pooling*, jenis *pooling* yang digunakan adalah *max pooling* yaitu mengambil nilai terbesar (*max*) dengan filter *pooling* yang berukuran 2x2. Penerapan teknik ini menyebabkan ukuran citra berkurang tanpa mengurangi informasi secara signifikan, tahap ini berguna untuk mempercepat proses komputasi pada *training*.



Gambar 4.5 Proses *Max pooling*

Setelah *pooling pixel* citra akan ditransformasi menjadi bentuk vektor (*flatten*) yang nantinya akan menjadi masukan pada *neuron fully-connected layer* yang merupakan *layer* yang bertanggung jawab untuk tugas klasifikasi pada CNN, pada penelitian ini variasi

jumlah *neuron fully-connected layer* sebesar 32 *neuron*, 64 *neuron*, dan 96 *neuron.*

Pada *fully-connected layer* di penelitian ini, telah diterapkan teknik *Dropout* yang berarti tidak semua *neuron* yang terlibat akan digunakan pada proses *training*, dimana penerapan *dropout* sebesar 20% berarti hanya 80% jumlah *neuron* yang digunakan digunakan pada tahap *training*. Hal ini berfungsi untuk mengurangi *overfiting* pada model CNN (akurasi *testing* lebih rendah dari *training*) dengan pertimbangan kompleksitas citra kasus klasifikasi porno yang sangat tinggi.

4.4. *Training*

Parameter *training* pada penelitian ini terdiri dari *Epoch* atau iterasi pelatihan sebanyak 10 kali, dan *bacth* yang merupakan jumlah citra yang diambil untuk setiap iterasi pada proses *training*. Penggunaan *batch* membuat beban memori yang dimuat secara bersamaan berkurang. Pada tahap *training*, dilakukan perhitungan *cost* dari *weight* yang digunakan pada model. Nilai dari *cost* inilah yang menjadi bahan evaluasi model untuk melakukan optimisasi dengan cara memperbarui *weight* pada model. Sehingga pada tiap *epoch*-nya, model cenderung mengalami peningkatan akurasi.

4.5. *Testing*

Pada tahap ini dilakukan pengujian terhadap model yang dihasilkan pada tahap *training* dan pengujian terhadap aplikasi *porn away*. Model yang didapat dalam proses *training* akan diuji dan diukur performanya dalam mengenali citra porno atau tidak porno dengan menggunakan *dataset testing* yang mana citra dalam *dataset testing* tersebut belum pernah digunakan dalam proses *training* sebelumnya. Pengujian aplikasi akan dilakukan dengan metode *Mean Opinion Score* yang melibatkan 30 responden.

4.6. Hasil pengujian

Pengamatan terhadap hasil pengujian dilakukan sesuai dengan yang telah dijabarkan pada subjudul mekanisme penelitian. Model terbaik di tentukan dengan memperoleh hasil kombinasi performa parameter-parameter terbaik. Pengujian ini menggunakan metode *precision*, *Recall* dan *Computation Time*.

4.7.1 Pengujian Terhadap Ukuran *Kernel*

Dalam layer konvolusi, filter konvolusi bergeser ke semua piksel gambar yang mengambil produk pada pixel citra. Ini dilakukan karna kombinasi linear dari piksel yang ditimbang oleh filter akan mengekstraksi beberapa jenis fitur sekaligus dari suatu citra. *Kernel* merupakan matrix yang menjadi pengali untuk mendapatkan fitur dari citra masukan. *Kernel* lazimnya berukuran ganjil karena semua piksel lapisan sebelumnya

akan simetris di sekitar piksel keluaran. Pada penelitian nilai kernel konvolusi yang digunakan akan ditentukan dengan algoritma Glorot Uniform dimana nilai kernel konvolusinya diperhitungkan berdasarkan input *neuron* dan *weight* nya. Hasil pengujian dari penggunaan *kernel* 3x3 dan 5x5 dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil *confusion matrix* pengujian *kernel*

	***Kernel* 3x3**		***Kernel* 5x5**	
	Non Porn	**Porn**	**Non Porn**	**Porn**
Non Porn	2615	70	2355	330
Porn	600	2085	211	2474

Tabel 4.2 Performa pengujian ukuran *kernel*

Ukuran *Kernel*	**Akurasi (%)**	**Presisi (%)**	***Recall***	***Computation Time* (ms)**
3x3	87.53	97	78	6
5x5	89.92	88	92	5

Berdasarkan Tabel 4., parameter *kernel* konvolusi berukuran 5x5 memiliki performa yang lebih baik daripada berukuran 3x3. Dengan demikian ukuran kernel 5x5 akan digunakan sebagai parameter pengujian selanjutnya.

4.7.2 Pengujian Terhadap Jumlah *Convolution Layer*

Beberapa kali penerapan operasi layer konvolusi dalam arsitektur *neural network* akan menghasilkan performa yang berbeda-beda tergantung kasus kelas citra yang ditangani. Oleh karena itu, dibutuhkan uji coba berapa banyak penerapan layer konvolusi dalam kasus klasifikasi citra porno ini yang menghasilkan performa yang terbaik, penelitian ini menggunakan 2 variasi jumlah layer konvolusi sebagai parameter uji dengan hasil seperti pada Tabel 4.3.

Tabel 4.3 Hasil *confusion matrix* pengujian terhadap *hidden layer*

	2CL		**3CL**	
	Non Porn	**Porn**	**Non Porn**	**Porn**
Non Porn	2355	330	2526	159
Porn	211	2474	347	2338

Tabel 4.4 Performa pengujian terhadap jumlah *hidden layer*

Jumlah CL	**Akurasi (%)**	**Presisi (%)**	***Recall***	***Computation Time* (ms)**
2	89.92	88	92	5
3	90.59	94	87	6

Berdasarkan hasil pengujian pada Tabel 4.4, didapatkan bahwa akurasi model dengan parameter layer konvolusi sebanyak 3 lebih baik jika dibandingkan dengan layer

konvolusi sebanyak 2, dengan demikian 3 layer konvolusi akan digunakan sebagai parameter pengujian selanjutnya.

4.7.3 Pengujian Terhadap Ukuran *Neuron Hidden layer*

Sama halnya dengan jumlah *Convolution layer*, beberapa variasi jumlah *neuron* menghasilkan performa yang berbeda-beda tergantung kasus kelas citra yang ditangani. Pada penelitian ini, digunakan 3 variasi jumlah *neuron* untuk tiap *hidden layernya* yaitu 32, 64, dan 96. Performa dari tiap variasi ukuran *neuron* disajikan pada Tabel 4.5.

Tabel 4.5 Hasil *confusion matrix* pengujian terhadap ukuran parameter *neuron*

	32		**64**		**96**	
	Non Porn	**Porn**	**Non Porn**	**Porn**	**Non Porn**	**Porn**
Non Porn	2526	159	2526	159	2298	387
Porn	347	2338	551	2134	181	2504

Tabel 4.6 Performa pengujian terhadap parameter jumlah *neuron*

Jumlah neuron	**Akurasi (%)**	**Presisi (%)**	***Recall***	***Computation Time* (ms)**
32	90.59	94	87	4
64	86.69	93	79	4
96	89.46	87	93	5

Berdasarkan hasil pengujian pada Tabel 4.6, didapatkan bahwa performa pengujian model dengan parameter jumlah *neuron* dengan ukuran berjumlah 32 lebih baik daripada *neuron* yang berjumlah 64 dan 96. Dengan demikian, pengujian untuk parameter selanjutnya, menggunakan arsitektur dengan *neuron hidden layer* berjumlah 32.

4.7.4 Pengujian Terhadap Ukuran Citra Masukan

Ukuran citra masukan dalam arsitektur *neural network* adalah hal yang signifikan dalam mempengaruhi performa model, dikarenakan semakin kecil ukuran citra maka informasi dari citra itu sendiri semakin sedikit, sebaliknya jika semakin besar akan membutuhkan harga komputasi yang mahal dan terlalu banyak informasi citra cenderung akan menyebabkan *overfitting* pada model. Penelitian ini menggunakan 3 variasi ukuran citra yaitu 96x96, 160x160 dan 224x224. Hasil dari pengujian dapat dilihat pada Tabel 4.7

Tabel 4.7 Hasil *confusion matrix* pengujian terhadap ukuran citra masukan dengan citra *testing*

	96x96		**160x160**		**224x224**	
	Non Porn	**Porn**	**Non Porn**	**Porn**	**Non Porn**	**Porn**
Non Porn	2526	159	2455	230	2526	159
Porn	347	2338	213	2472	551	2134

Tabel 4.8 Performa pengujian terhadap ukuran citra masukan

Ukuran Citra	**Akurasi (%)**	**Presisi (%)**	***Recall***	***Computation Time* (ms)**
96x96	90.59	94	87	4
160x160	91.67	91	92	5
224x224	91.23	93	79	7

Berdasarkan hasil pengujian pada Tabel 4.8, performa ukuran masukan citra 160x160 dan 224x224 mempunyai perbedaan sangat sedikit dengan tingkat akurasi 91.67% dan 91.23%, namun tingkat presisi ukuran citra masukan 160x160 jauh lebih tinggi daripada masukan citra berukuran 224x224. Sehingga dapat disimpulkan parameter ukuran citra masukan terbaik adalah 160x160 sehingga akan digunakan untuk pengujian parameter selanjutnya.

4.7.5 Pengujian Terhadap Dimensi Citra

Pengaruh dimensi citra pada performa model tergantung pada kompleksitas kelas yang akan di klasifikasi, jika warna tidak memiliki arti dalam citra kelas untuk dijadikan pertimbangan klasifikasi maka lebih baik menggunakan grayscale untuk menghindari klasifikasi dan kompleksitas yang salah. Dimensi citra yang diuji pada penelitian ini adalah *grayscale* dan RGB. Hasil dari pengujian terhadap dimensi citra dapat dilihat pada Tabel 4.9.

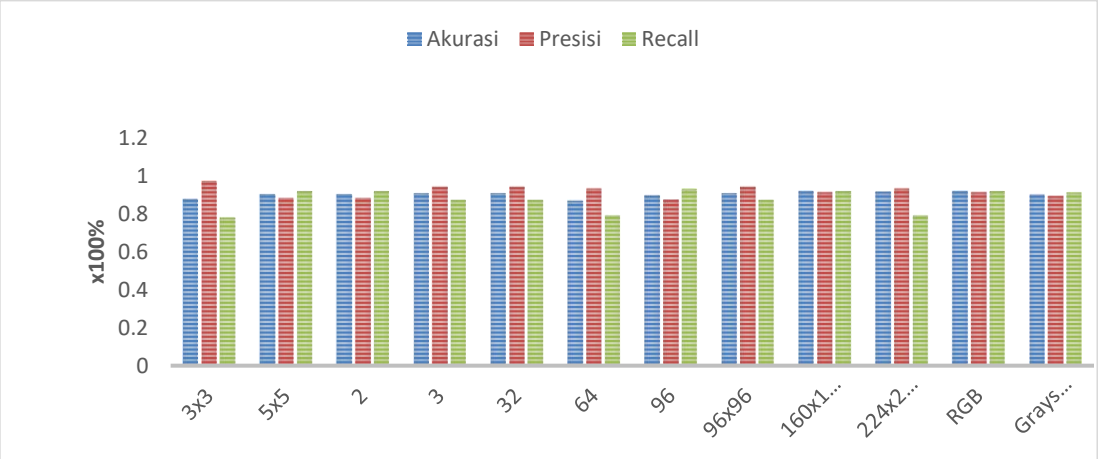
Tabel 4.9 Hasil *confusion matrix* pengujian terhadap dimensi citra dengan citra *testing*

	RGB		**Grayscale**	
	Non Porn	**Porn**	**Non Porn**	**Porn**
Non Porn	2455	230	2371	314
Porn	213	2472	243	2442

Tabel 4.10 Hasil pengujian terhadap dimensi citra

Dimensi Citra	**Akurasi *Testing* (%)**	**Akurasi *Testing* (%)**	**Recall**	***Computation Time* (ms)**
RGB	91.6	91	92	6
Grayscale	89.7	89	91	5

Berdasarkan Tabel 4., didapatkan bahwa model dengan jenis masukan berupa citra RGB memilki performa yang lebih bagus dibandingkan dengan citra *grayscale*, sehingga dimensi citra RGB akan di gunakan sebagai parameter pengujian model terbaik.



Gambar 4.6 Diagram hasil uji performa parameter

Jadi, dapat disimpulkan berdasarkan hasil pengujian performa parameter-parameter yang dapat dilihat pada Gambar 4.7 didapatkan 5 parameter terbaik yaitu dengan ukuran *kernel* berukuran 5x5, jumlah *layer* konvolusi sebanyak 3, jumlah neuron sebanyak 32, ukuran citra masukan berukuran 160x160 *pixel*, dan dimensi warna berjenis RGB, dengan demikian model terbaik akan menggunakan nilai parameter-parameter tersebut yang akan diuji kembali performanya dengan KIAdataset.

Pada hasil pengujian, telah didapatkan model terbaik berdasarkan pengujian terhadap 5 parameter uji seperti yang disebutkan pada mekanisme penelitian, dari pengujian tersebut telah di dapatkan performa terbaik yang disajikan pada Tabel 4.11.

Tabel 4.11 Hasil *confusion Matrix* pengujian model terbaik

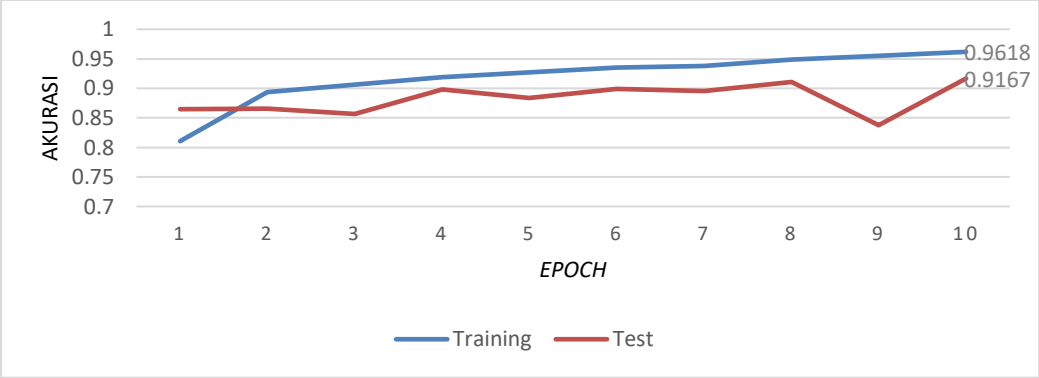
		Predict class	
		Non-Porn	**Porn**
Actual class	**Non-Porn**	2455	230
	Porn	213	2472

Tabel 4.12 Performa pengujian model terbaik

Akurasi Test (%)	**Presisi (%)**	***Recall* (%)**	***Computation Time* (ms)**
91.67	91	92	6

Tabel 4.13 menunjukkan bahwa model terbaik berdasarkan pengujian parameter-parameter sebelumnya dapat melakukan tugas klasifikasi citra porno dengan sangat baik dengan tingkat akurasi 91.67%, tingkat presisi 91%, tingkat *recall* 92%, dan dengan *Computation Time* 6 ms.

Nilai *recall* yang tinggi menandakan model bekerja dengan baik dalam mengenali *true positive* dan mengurangi jumlah *false negative,* dalam kasus ini performa *recall* lebih penting daripada presisi dikarenakan risiko jika citra porno di klasifikasikan sebagai tidak porno (*false negative)* lebih besar.



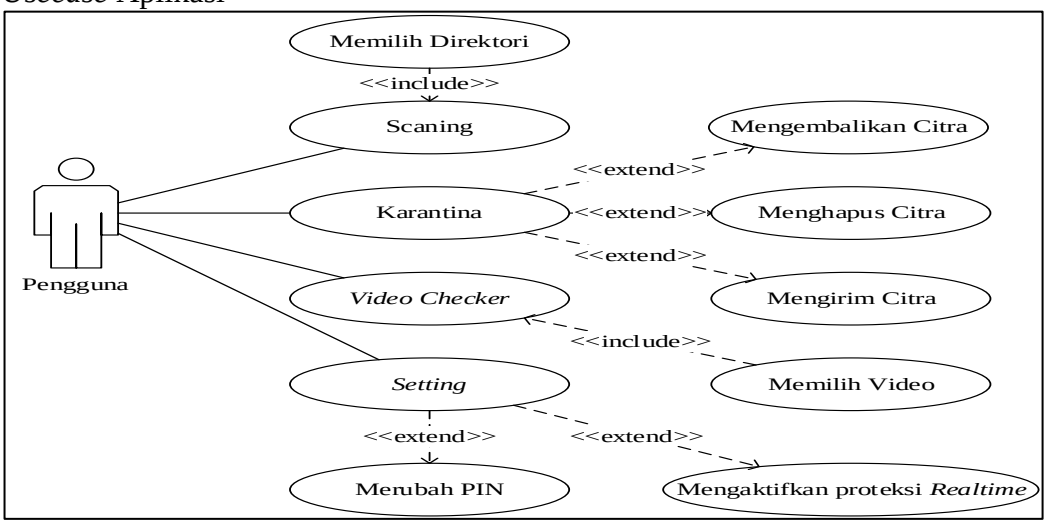
Gambar 4.7 Diagram hasil akurasi *training* dan *test* pada model.

Diagram di atas merupakan performa akurasi model saat di uji dengan dataset *training* dan dataset *testing*, diagram tersebut menunjukan performa model tidak mengalami *overfitting* atau *underfitting* karna selisih akurasi model dengan data *training* dan data *testing* sekitar 0.04 saja. Selisih ini terjadi karna model lebih banyak mempelajari *noise* dan detail yang tidak perlu yang ada pada data *training.*

4.7. *Deploying*

Pada tahap ini dilakukan pengembangan aplikasi berbasis Android, dengan menggunakan model dengan hasil performa terbaik dalam proses *training* sebelumnya sebagai *engine* klasifikasi citra porno pada aplikasi *Porn away*.

a. *Usecase* Aplikasi



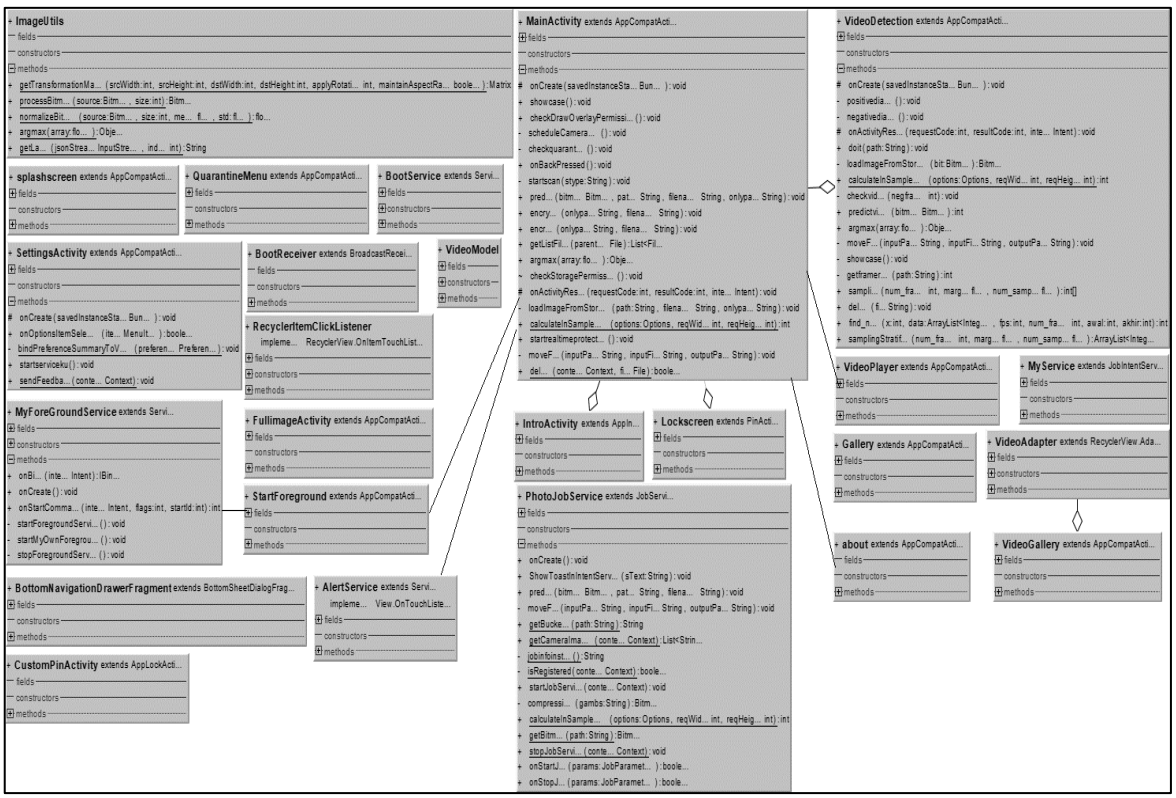
Gambar 4.8 Usecase aplikasi *Porn Away*.

Pada Gambar 4.11 terdapat pengguna aplikasi yang dapat :

1. *Scanning* citra, untuk itu pengguna harus memilih direktori terlebih dahulu.
2. Karantina citra, pengguna dapat menghapus, mengembalikan dan mengirim citra dalam karantina.
3. *Video checker,* yakni mendeteksi video pornografi, untuk ini pengguna harus memilih direktori terlebih dahulu.
4. *Setting,* yakni pengguna dapat merubah pengaturan aplikasi seperti merubah PIN dan mengaktifkan proteksi *realtime.*

b. *Class Diagram*

Aplikasi *Porn away* dikembangkan berbasis bahasa pemrograman java. *Class-class* ini merupakan *class* yang berisi bahasa pemrograman java yang dibuat untuk mengimplementasikan sistem sesuai dengan perancangan yang dilakukan berupa *sourcecode* (*coding*). Hasil dari proses *coding* tersebut akan menghasilkan *interface* yang akan berinteraksi langsung dengan pengguna. Berikut merupakan implementasi *class* yang dilakukan dalam pengembangan aplikasi *porn away*.



Gambar 4.9 Class Diagram Aplikasi *Porn away*

Class pada aplikasi *porn away* berjumlah 25 *class* yang berfungsi untuk menjalankan fitur-fitur aplikasi, dimana *class MainActivity* berfungsi sebagai *class* utama yang membentuk menu utama aplikasi sekaligus tempat berjalannya *engine* klasifikasi,

class VideoDetection, VideoAdapter, VideoModel, VideoPlayer, VideoGallery berperan sebagai layanan klasifikasi video, *class ImageUtils* berberan menjalankan *pre-processing* citra, *class CustomPinActivity* berperan sebagai proteksi pin aplikasi, *class AlertService, BootService, MyForegroundService* berperan sebagai *background service* untuk mendeteksi citra porno baru yang di buat pada *smartphone.* Adapun fungsi yang merepresentasikan fungsi utama dari aplikasi ini adalah :

1. *StartScan*()

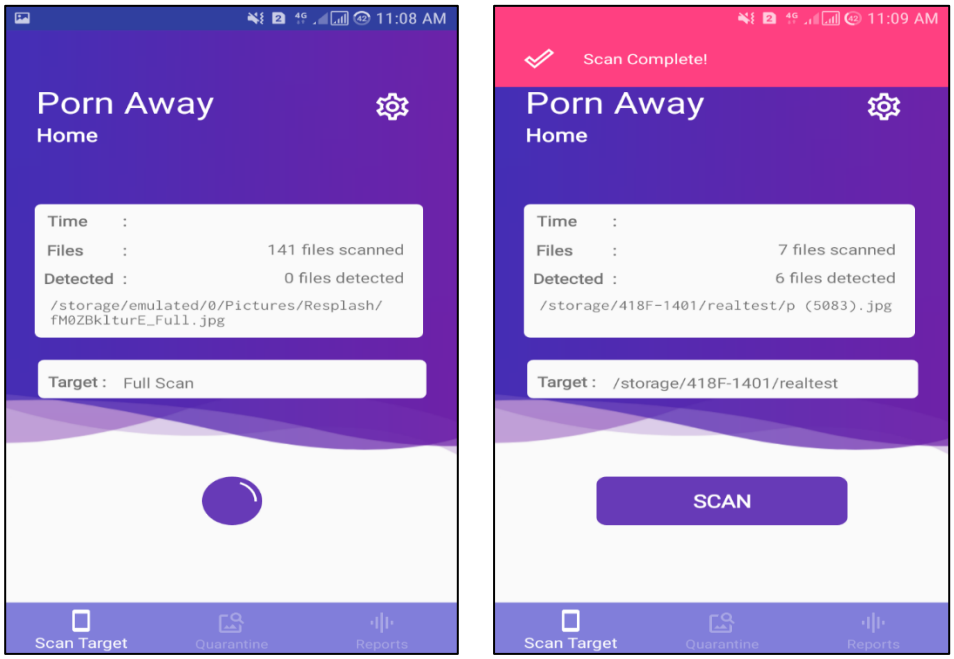
   Fungsi ini berguna untuk klasifikasi banyak citra sekaligus untuk menemukan citra pornografi pada direktori folder yang di pilih pengguna, jika citra terdeteksi fungsi ini akan membuat list citra yang terdeteksi pornografi yang akan di karantina.
2. *Encrypt()*

   Fungsi ini berperan untuk melindungi citra pornografi yang di karantina aplikasi agar tidak di akses oleh pengguna yang tidak mempunyai hak, dengan mengenkripsi semua citra pornografi yang terdeksi.
3. *Startforegroundservice()*

   Fungsi ini berguna untuk memulai *background service* yang akan mendeteksi citra pornografi baru yang dibuat pada kamera *smarthphone* bahkan ketika aplikasi ditutup, sehingga pendeteksian akan berlangsung *realtime.*

c. Implementasi *interface* aplikasi

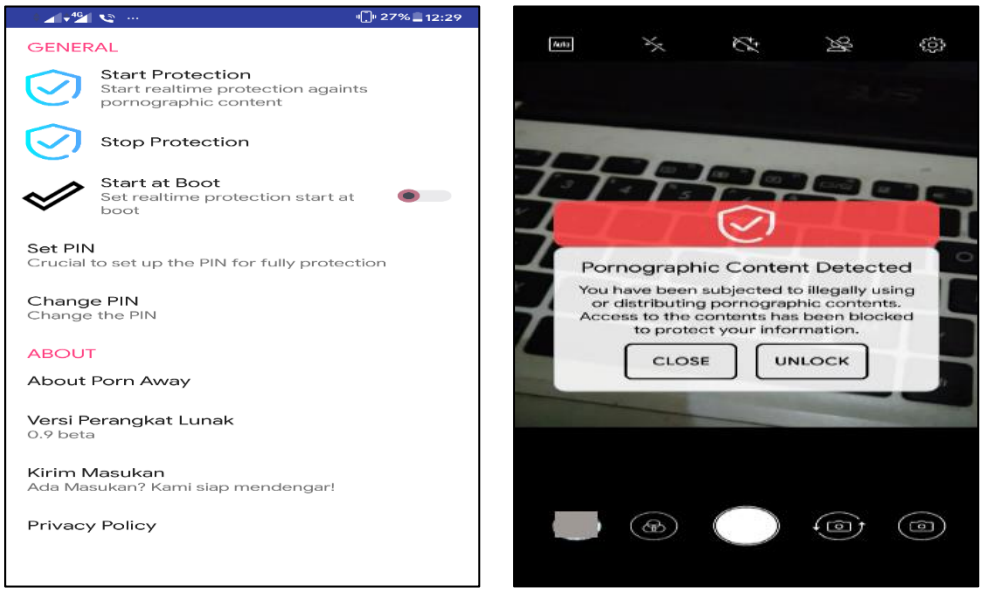
Interface merupakan antarmuka aplikasi yang akan digunakan oleh pengguna untuk berinteraksi langsung dengan aplikasi. Dalam implementasi *interface*, telah dikembangkan berdasarkan perancangan yang telah dibuat sebelumnya. Pada sistem ini hanya terdapat 1 macam sisi *interface,* yaitu pada sisi pengguna saja. Perancangan desain sistem yang dibuat dapat dilihat pada Gambar 4.9, Gambar 4.10, dan lampiran



(a) (b).

Gambar 4.10 Tampilan menu utama dan proses *scanning* citra.

Pada tampilan menu utama, pengguna dapat menentukan tipe *scan* dan memilih tombol *scan* maka proses *scan* akan dimulai, animasi proses *scaning* dan informasi terkini proses akan ditampilkan di tampilan utama aplikasi , setelah proses selesai maka animasi proses *scan* akan berhenti dan pengguna di perbolehkan untuk memulai proses *scan*.



(a) (b)

Gambar 4.11 Tampilan menu pengaturan dan tampilan skenario pendeteksian citra baru porno.

Pada tampilan menu pengaturan berfungsi untuk mengaktifkan real-time protection untuk memindai gambar porno baru pada smartphone dalam background, start at boot untuk

memulai aplikasi sesaat perangkat di nyalakan, mengganti PIN, melihat tentang aplikasi dan mengirim masukan terhadap aplikasi. Tampilan skenario pendeteksian citra baru merupakan tampilan disaat aplikasi mendeteksi pengguna memotret gambar porno dengan aplikasi kamera bawaan, akan muncul peringatan bahwa gambar telah di karantina oleh aplikasi dan membutuhkan PIN untuk mengaksesnya kembali.

4.8. Pengujian *Mean Opinion Score* (MOS)

Pengujian dilakukan dengan mencari responden untuk mencoba menjalankan aplikasi dan pengisian kuisioner. Tujuan dari pengujian ini adalah mengukur bagaimana kualitas aplikasi saat digunakan oleh pengguna. Metode yang digunakan dalam pengujian kuesioner ini adalah metode kuantitatif, yaitu hasil pengujian didefinisikan dalam suatu nilai angka. Pengujian ini dilakukan oleh 30 responden yang yang berasal dari kalangan masyarakat umum, dan mahasiswa.

Hasil dari jawaban responden nantinya akan di kalkulasi dan ditarik kesimpulan mengenai hasil pengujian sistem. Kuesioner pengujian sistem yang diberikan untuk masyarakat umum dan mahasiswa terdiri dari 13 pernyataan, yaitu:

1. *Smartphone* menjadi lambat ketika aplikasi di *install*?
2. *Smartphone* menjadi lambat ketika aplikasi dijalankan?
3. Aplikasi cepat dalam memuat tiap halamannya?
4. Fitur aplikasi sudah lengkap untuk melakukan *scanning*?
5. Fungsi tiap fitur sudah sangat jelas?
6. Aplikasi mampu mengenali gambar porno ketika *scanning*?
7. Aplikasi mampu mengenali gambar porno yang diambil melalui kamera?
8. Aplikasi mampu mengenali video porno?
9. Aplikasi mampu melakukan *scan* ke seluruh *file* foto dan yang tersimpan di *smartphone*?
10. Aplikasi mampu melakukan karantina, penghapusan, dan *restore* gambar porno yang terindikasi porno?
11. Desain antarmuka aplikasi bagus untuk dipandang?
12. Desain aplikasi *User-Friendly?*
13. Desain tata letak tiap fungsi memudahkan penggunaan?

Dari pertanyaan tersebut, responden diminta untuk menjawab dengan mencentang sesuai dengan Tabel 4.13.

Tabel 4.13 *Mean Opinion Score*

MOS	Keterangan	Bobot Nilai
SS	Sangat Setuju	5
S	Setuju	4
KS	Cukup Setuju	3
TS	Kurang Setuju	2
STS	Tidak Setuju	1

Pada Tabel 4.13 terdapat bobot nilai dari setiap jawaban pertanyaan kuesioner dan keterangan yang nantinya akan menentukan kualitas aplikasi yang di uji termasuk ke dalam daftar kelompok yang sesuai dengan nilai yang didapatkan. *Mean Opinion Score* dengan persamaan (4-1).

$$mean\ \mathrm{p_i} = \frac{\sum_{i=1}^{n} p_i}{n} \tag{4-1}$$

dengan :

$mean\ p_i$: Rata-rata skor dari setiap pertanyaan

p_i : Nilai skor

n : Jumlah responden .

Persamaan (4-1) digunakan untuk dapat menghitung jumlah skor rata-rata yang diberikan oleh responden pada setiap pertanyaannya, sedangkan untuk persamaan (4-2) digunakan untuk mencari *Mean Opinion Score* atau mencari total skor yang diberikan responden pada seluruh atribut pertanyaan.

$$MOS = \frac{\sum_{i=1}^{k} Mean\ p_i}{k} \tag{4-2}$$

dengan :

MOS : Total skor rata-rata seluruh atribut pertanyaan.

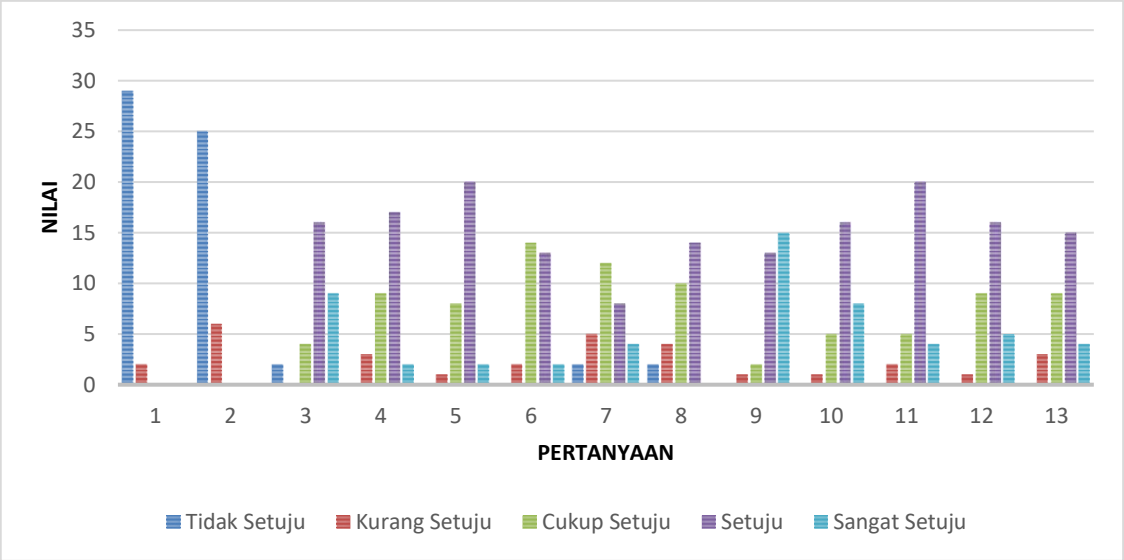
k : Jumlah atribut pertanyaan.

Pengujian dengan metode *Mean Opinion Score* pada Aplikasi *Porn away* didapatkan perhitungan yang sudah dilakukan pada jawaban kuisioner dari 30 orang responden yang terdiri dari mahasiswa dan masyarakat umum ditunjukan pada Tabel 4.14. Khusus untuk pertanyaan 1 dan 2 pada kuisioner bobot jawaban responden bernilai kebalikan dimana TS bernilai 5, KS bernilai 4, CS bernilai 3, S bernilai 2, dan SS bernilai 1.

Tabel 4.14 Hasil perhitungan MOS aplikasi *Porn away*

No	Pertannyaan	TS (1)	KS (2)	CS (3)	S (4)	SS (5)	Indeks (%)
1	Smartphone menjadi lambat ketika aplikasi di install	29	2	0	0	0	98.70
2	Smartphone menjadi lambat ketika aplikasi dijalankan	25	6	0	0	0	96.12
3	Aplikasi cepat me-load tiap halamannya	2	0	4	16	9	79.35
4	Fitur aplikasi sudah lengkap untuk melakukan scanning	0	3	9	17	2	71.61
5	Fungsi tiap fitur sudah sangat jelas	0	1	8	20	2	74.83
6	Aplikasi mampu mengenali gambar porno ketika scanning	0	2	14	13	2	69.67
7	Aplikasi mampu mengenali gambar porno yang diambil melalui kamera	2	5	12	8	4	64.51
8	Aplikasi mampu mengenali video porno	2	4	10	14	0	61.93
9	Aplikasi mampu melakukan scan ke seluruh file foto dan yang tersimpan di *smartphone*	0	1	2	13	15	87.09
10	Aplikasi mampu melakukan karantina, penghapusan, dan restore gambar porno yang terindikasi porno	0	1	5	16	8	78.06
11	Desain antarmuka aplikasi bagus untuk dipandang	0	2	5	20	4	76.77
12	Desain aplikasi U*ser-Friendly*	0	1	9	16	5	76.12
13	Desain tata letak tiap fungsi memudahkan penggunaan	0	3	9	15	4	72.90
MOS (Mean Opinion Score)							77.51

Berdasarkan hasil pengujian MOS yang dilakukan oleh 30 orang responden, Pengujian yang dilakukan pada mahasiswa dan masyarakat umum menghasilkan nilai MOS sebesar 77.51% menunjukkan bahwa aplikasi *Porn away* terlah berjalan dengan baik.



Gambar 4.12 Grafik hasil jawaban responden.

Adapun grafik hasil perhitungan *mean opinion score* di gambarkan dengan grafik pada Gambar 4.11.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan penelitian yang sudah dilakukan, terdapat beberapa hal yang bisa disumpulkan antara lain sebagai berikut.

1. Parameter-parameter model terbaik yang di dapatkan pada penelitian ini adalah *kernel* berukuran 5x5, jumlah *layer* konvolusi sebanyak 3, jumlah *neuron* sebanyak 32, ukuran citra masukan sebesar 160x160 *pixel,* dan dimensi citra masukan berjenis RGB.
2. Pada pengujian model terbaik dengan KIAdataset menghasilkan performa akurasi sebesar 91.67%, tingkat presisi 91%, tingkat *recall* 92%, dan dengan waktu komputasi 6 *ms*.
3. Nilai *recall* sebesar 92% menandakan model bekerja dengan baik dalam mengenali *true positive* dan mengurangi jumlah *false negative,* dalam kasus ini performa *recall* lebih penting daripada presisi dikarenakan risiko jika citra porno di klasifikasikan sebagai tidak porno (*false negative)* lebih besar.
4. Performa model pada pengujian *training* dan *testing* tidak mengalami *overfitting* atau *underfitting* karna hanya mempunyai selisih 4% saja.
5. Aplikasi untuk mengenali dan menangani citra pornografi berbasis *smartphone* Android (*Porn away*) terlah berjalan dengan baik berdasarkan hasil pengujian yang dilakukan pada mahasiswa dan masyarakat umum yang menghasilkan nilai *mean opinion score* sebesar 77.51%.

5.2. Saran

Saran-saran yang penulis berikan apabila penelitian ini akan dikembangkan kembali antara lain sebagai berikut.

1. Meningkatkan ukuran *dataset* untuk mendapatkan performa model yang lebih tinggi dengan pertimbangan metode CNN adalah metode *supervised learning* yang memiliki potensi lebih tinggi dengan ukuran *dataset* yang lebih banyak.
2. Mempertegas standarisasi suatu citra porno dalam *dataset* di dikarenakan masih ambigunya perbedaan antara citra porno dan citra seksi.
3. Menambah fitur-fitur yang ada dalam aplikasi *Porn away* yang melibatkan integrasi pengawasan orang tua.

DAFTAR PUSTAKA

[1] Republik Indonesia, *Undang-Undang Republik Indonesia Nomor 44 Tahun 2008 tentang Pornografi*. 2008.

[2] Asosiasi Penyelengara Jasa Internet (APJII), "Hasil Survei Penetrasi dan Perilaku Pengguna Internet Indonesia 2018," 2018.

[3] T. Verge, "Google announces over 2 billion monthly active devices on Android," 2017. [Online]. Available: https://www.theverge.com/2017/5/17/15654454/android-reaches-2-billion-monthly-active-users. [Accessed: 27-Mar-2019].

[4] datareportal.com, "Digital Indonesia : 2018," 2018. [Online]. Available: https://datareportal.com/reports/digital-2018-indonesia?rq=Indonesia. [Accessed: 07-Jun-2019].

[5] I. G. P. Sutawijaya and I. B. K. Widiartha, "Pengenalan Citra Porno Berbasis Kandungan Informasi Citra (Image Content)," vol. 4, no. 2, pp. 80–86, 2004.

[6] J. A. M. Basilio, G. A. Torres, G. S. Pérez, L. K. T. Medina, and H. M. P. Meana, "Explicit Image Detection using YCbCr Space Color Model as Skin Detection," *Appl. Math. Comput. Eng.*, no. December, pp. 123–128, 2011.

[7] M. Zufar and B. Setiyono, "Convolutional Neural Networks untuk Pengenalan Wajah Secara Real - Time," *J. Sains Dan Seni Its*, vol. 5, no. 2, pp. 72–77, 2016.

[8] H. Abhirawa, Jondri, and A. Arifianto, "Pengenalan Wajah Menggunakan Convolutional Neural Networks (CNN)," vol. 4, no. 3, pp. 4907–4916, 2016.

[9] M. Moustafa, "Applying deep learning to classify pornographic images and videos," *CoRR*, vol. abs/1511.08899, 2015.

[10] Imagenet, "Large Scale Visual Recognition Challenge (ILSVRC)," *ImageNet Large Scale Vis. Recognit. Chall.*, 2015.

[11] S. Y. Iriyanto and T. M. Zaini, *Pengolahan citra digital*. Lampung: Anuggrah Utama Raharja (AURA), 2014.

[12] R. Gonzalez and R. Woods, *Digital image processing*, 3rd ed. New Jersey: Pearson Education, 2002.

[13] A. Nurhayati, L. Wangi, and B. Poerwanto, "Analisis pengaruh frekuensi menonton blue film terhadap hasil belajar mahasiswa," *Univ. cokroaminoto palopo*, vol. 02, pp. 218–225, 2008.

[14] Jimmy Pujoseno, *Impelemntasi Deep Learning Menggunakan Convolution Neural*

Network untuk Klasifikasi Alat Tulis. Yogyakarta: Universitas Islam Indonesia, 2018.

[15] Google, "MNIST For ML Beginners," 2019. [Online]. Available: https://tensorflow.rstudio.com/tensorflow/articles/tutorial_mnist_beginners.html. [Accessed: 03-Mar-2019].

[16] A. L. HODGKIN and A. F. HUXLEY, "A quantitative description of membrane current and its application to conduction and excitation in nerve.," *J. Physiol.*, vol. 117, no. 4, pp. 500–544, Aug. 1952.

[17] T. Nurhikmat, *Implementasi Deep Learning untuk Image Classification Menggunakan Algoritma Convolutional Neural Network (CNN) Pada Citra Wayang Golek*. Yogyakarta: Universitas Islam Indonesia, 2018.

[18] Murtiwiyati and G. Lauren, "KOMPUTASI Komputer & Sistem Informasi," *J. Ilm. Komputasi*, 2013.

[19] G. Developer, "Android Developer," 2018. [Online]. Available: https://developer.android.com/.

[20] Suhyun Kim, "A Beginner's Guide to Convolutional Neural Networks (CNNs)." [Online]. Available: https://towardsdatascience.com/a-beginners-guide-to-convolutional-neural-networks-cnns-14649dbddce8. [Accessed: 20-Mar-2019].

[21] D. Iskandar and Y. K. Suprapto, "Perbandingan Akurasi Klasifikasi Tingkat," *J. Ilm. NERO*, vol. 2, no. 1, pp. 37–43, 2015.

LAMPIRAN

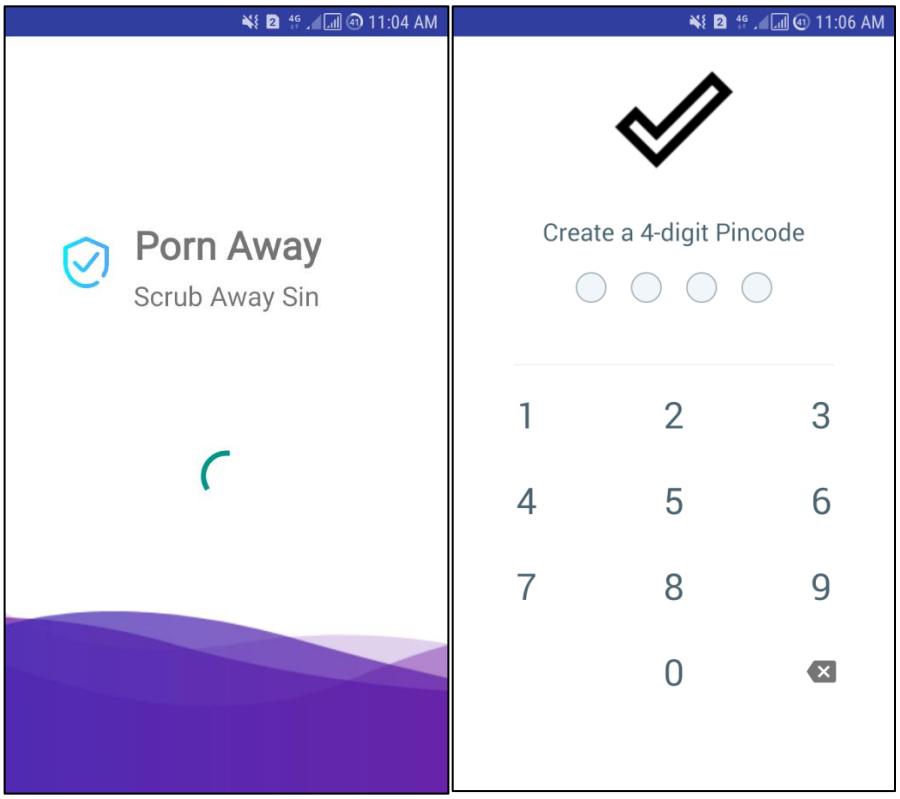
Lampiran 1 Informasi Tim dan Abstrak ILSVRC2014

Tim	Anggota	Abstrak
1-HKUST	Cewu Lu (Hong Kong University of Science and Technology)	For the detection task, we first generate some candidate bounding boxes, and then our system recognizes objects on these candidate proposals. We try to improve both localization and recognition. On the localization side, initial candidate proposals are generated from selective search [1], and a novel bounding boxes regression method is used for better object localization. On the recognition side, to represent a candidate proposal, we adopt many features such as RCNN features [2], IFV features [3], DPM features [4], to name a few. Given these features, category-specific combination functions are learnt to improve object recognition. Background priors and object interaction priors are also learnt and applied into our system. In addition, our framework involves some other novel techniques. The pertinent technical details for the submission are in preparation. In the ILSVRC2014 competition, we do not use any outside training data.
Adobe-UIUC	Hailin Jin (Adobe) Zhaowen Wang (UIUC) Jianchao Yang (Adobe) Zhe Lin (Adobe)	Our algorithm is based on an integrated convolutional neural network framework for both classification and localization. We train several 6-layer convnets using 3000 ImageNet classes for classification and then adapt one model for bounding box regression. At test time, we use k-means to find bounding box clusters and rank the clusters according to the classification scores.

Andrew Howard	Andrew Howard - Howard Vision Technologies	The final submission is made up of 2 sets of 3 networks plus 1 KNN prediction. The second set of networks are a smaller earlier version and only add a little value.
BDC-I2R,UPMC	Olivier Morère (1,2), Hanlin Goh (1), Antoine Veillard (2), Vijay Chandrasekhar (1)	Multiple deep convolutional neural networks (CNN) [Krizhevsky et al. 2012], each trained with a different set of parameters. The deep representations are extracted across multiple scales and positions within an image. Model fusion is adaptively performed within each CNN model, and subsequently across the different models. Class distribution priors are used to rectify the *output*s of the model. The CNN features are extracted across a GPU cluster, while a CPU cluster is used to optimize parameters in a MapReduce framework.
Trimps-Soushen	Jie Shao, Xiaoteng Zhang, JianYing Zhou, Jian Wang, Jian Chen, Yanfeng Shang, Wenfei Wang, Lin Mei, Chuanping Hu. The Third Research Institute of the Ministry of Public Security, P.R. China.	Task 2: Classification and localization Our model is based on large deep convolutional neural network. We use several methods to improve the performance. 1. Data Augmentation. Some of our models are trained on original data plus about 396000 external images from ILSVRC2010 and ILSVRC2011 training data. All training data belong to original 1000 object categories. Other data augmentation methods include random crops from Nx256 resized images, contrast and color jittering, and Gaussian noise. We use opencv to resize images with cubic interpolation, which we found very useful. 2. Model Details. The biggest model we trained has about 120M parameters.
UI	Fatemeh Shafizadegan, Msc	Our model is based on Spatial Pyramid Matching (SPM), similar to [1]. This is an extension of

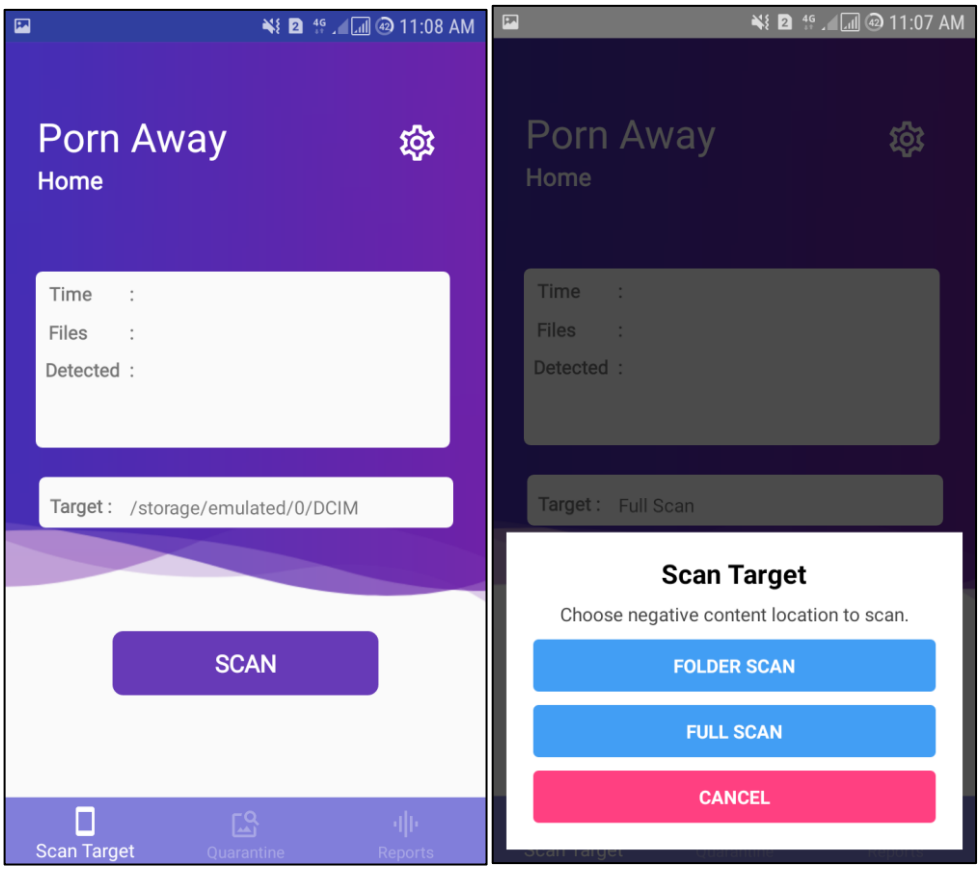
	student of Artificial Intelligence, University of Isfahan. Elham Shabaninia, PhD candidate of Artificial Intelligence,University of Isfahan.	SPM using sparse codes of SIFT features that propose a linear *kernel*. SIFT features are robust in rotation, scale, affine and different intensities. This approach reduce the complexity of SVM in training phase to O(n) and the complexity in testing phase doesn't change. This approach uses max spatial pooling that is robust to local spatial translations. The image representation turns out to work well with linear SVM classifiers.
VGG	Karen Simonyan, University of Oxford Andrew Zisserman, University of Oxford	In this submission we explore the effect of the convolutional network (ConvNet) depth on its accuracy. We have used three ConvNet architectures with the following weight layer configurations: 1) ten 3x3 convolutional layers, three 1x1 convolutional layers, and three fully-connected layers - 16 weight layers in total; 2) thirteen 3x3 convolutional layers and three fully-connected layers - 16 weight layers in total; 3) sixteen 3x3 convolutional layers and three fully-connected layers - 19 weight layers in total. All convolutional layers have stride 1 and are followed by ReLU non-linearity. The fully-connected layers are regularised with dropout. The networks were trained on fixed-size image crops, but at test time they were applied densely over the whole uncropped images.

Lampiran 2 Perancangan Desain Sistem



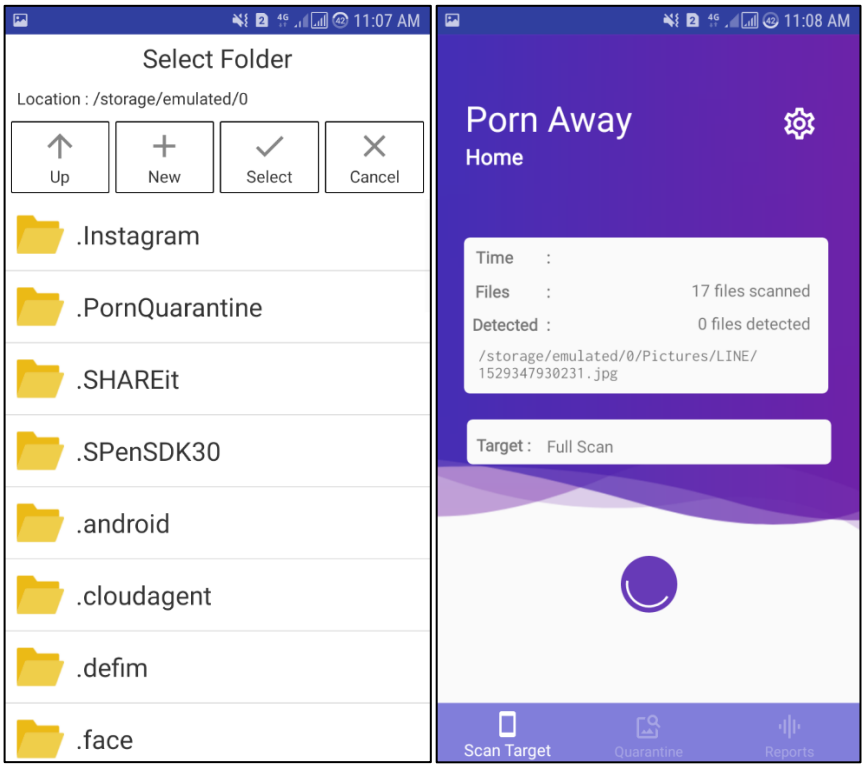
Gambar 1. a Tampilan *Splash Screen*; b. Tampilan Buat PIN

Sebelum menggunakan aplikasi, pengguna harus membuat PIN untuk penguncian aplikasi yang akan digunakan untuk mengakses fitur dalam aplikasi ini, Gambar 1 merupakan *splash screen* yang muncul kurang dari 2 detik dan selanjutnya diarahkan ke pembuatan PIN seperti pada Gambar 1 b.



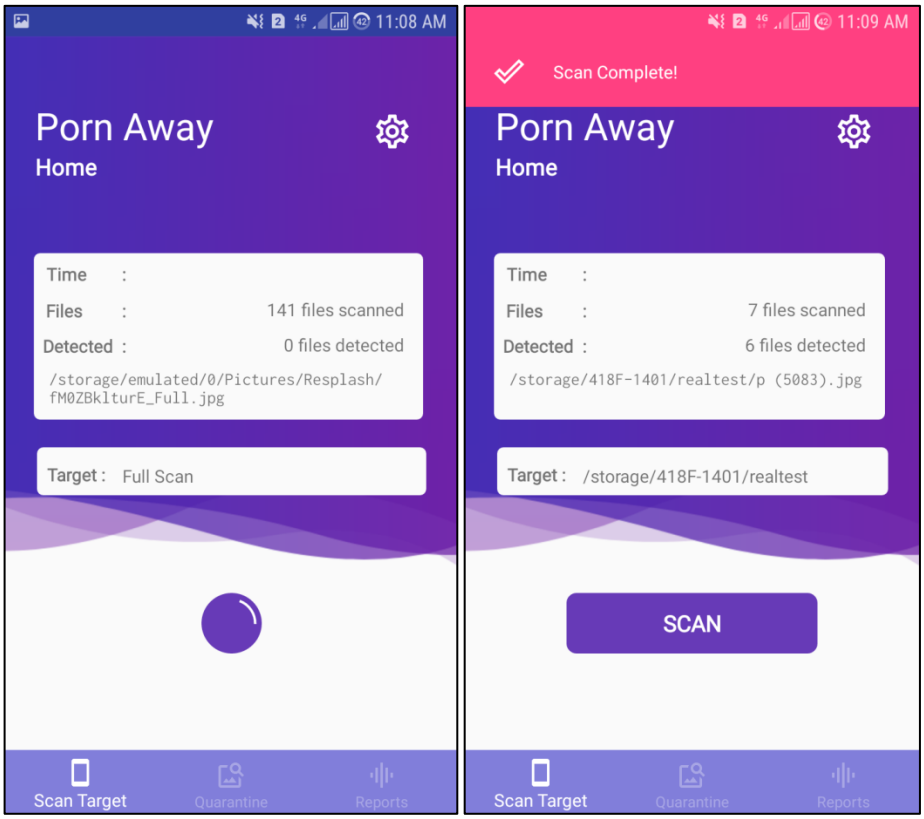
Gambar 2. a. Tampilan Awal Aplikasi; b. Tampilan memilih Scan Target

Gambar 2.a merupakan tampilan awal aplikasi yang akan menampilkan informasi dari proses *scan* terdiri dari *file* di *scan*,*file* terdeteksi konten negative, waktu, target *scan*, dan menu lainnya, jika pengguna memilih Scan target maka akan terlihat menu seperti gambar 3.b yang menampilkan pilihan tipe *scan*.



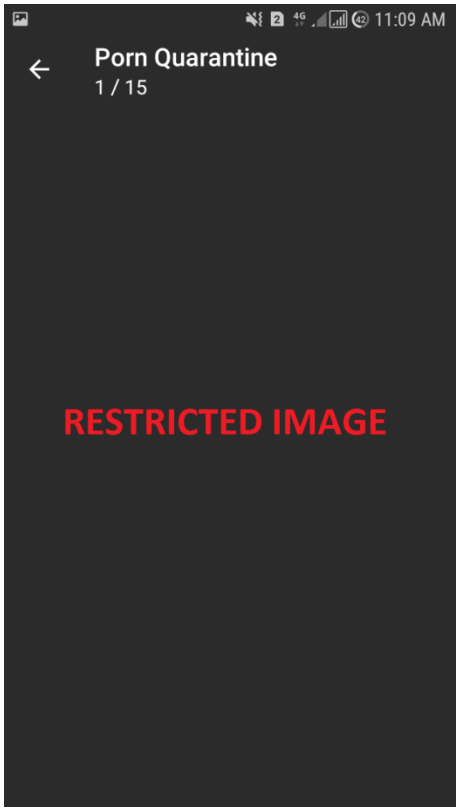
Gambar 3.a. *Folder Scan*; b. *Full Scan*.

Jika pengguna memilih *Folder Scan* pengguna harus memilih suatu direktori pada perangkatnya yang akan di *scan* seperti pada gambar 3.a, jika pengguna memilih *Full Scan* aplikasi akan melakukan *scan* secara menyeluruh, dan tipe *scan* akan ditampilkan di tampilan utama seperti pada gambar 3.b



Gambar 4.a. Tampilan proses *scan* ; b. Tampilan proses *scan* selesai

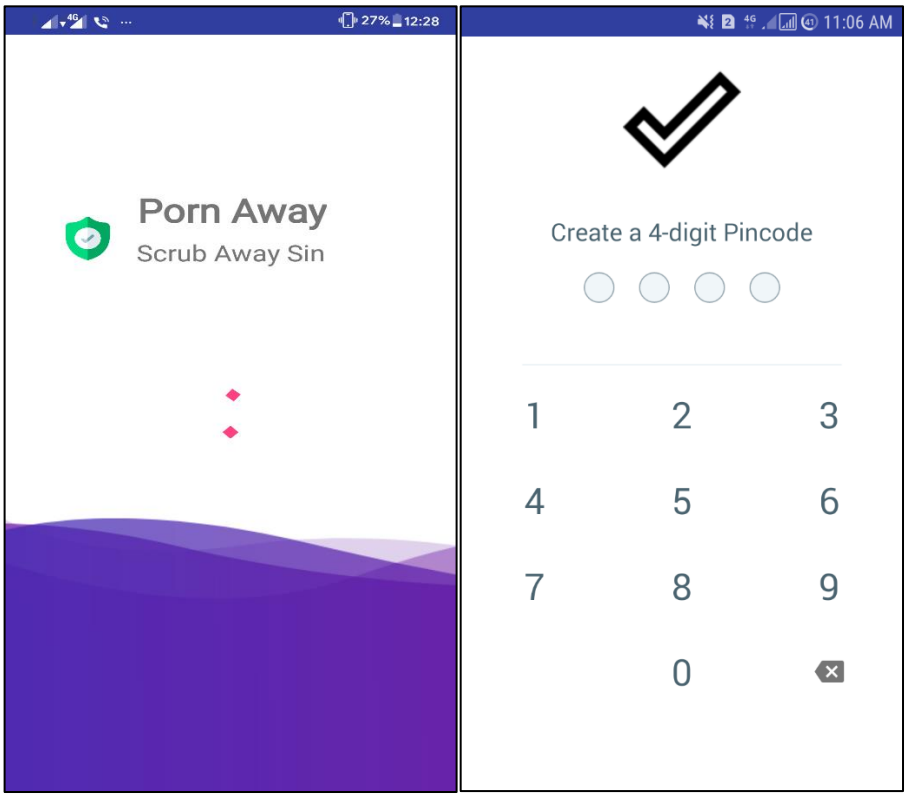
Saat pengguna telah menentukan tipe *scan* dan memilih tombol “SCAN” maka proses *scan* akan dimulai, animasi proses *scaning* dan informasi terkini proses akan ditampilkan di tampilan utama aplikasi , setelah proses selesai maka animasi proses *scan* akan berhenti pengguna di perbolehkan untuk memulai proses *scan* baru dan notifikasi *scan* selesai akan ditampilkan.



Gambar 5 Halaman *Quarantine*

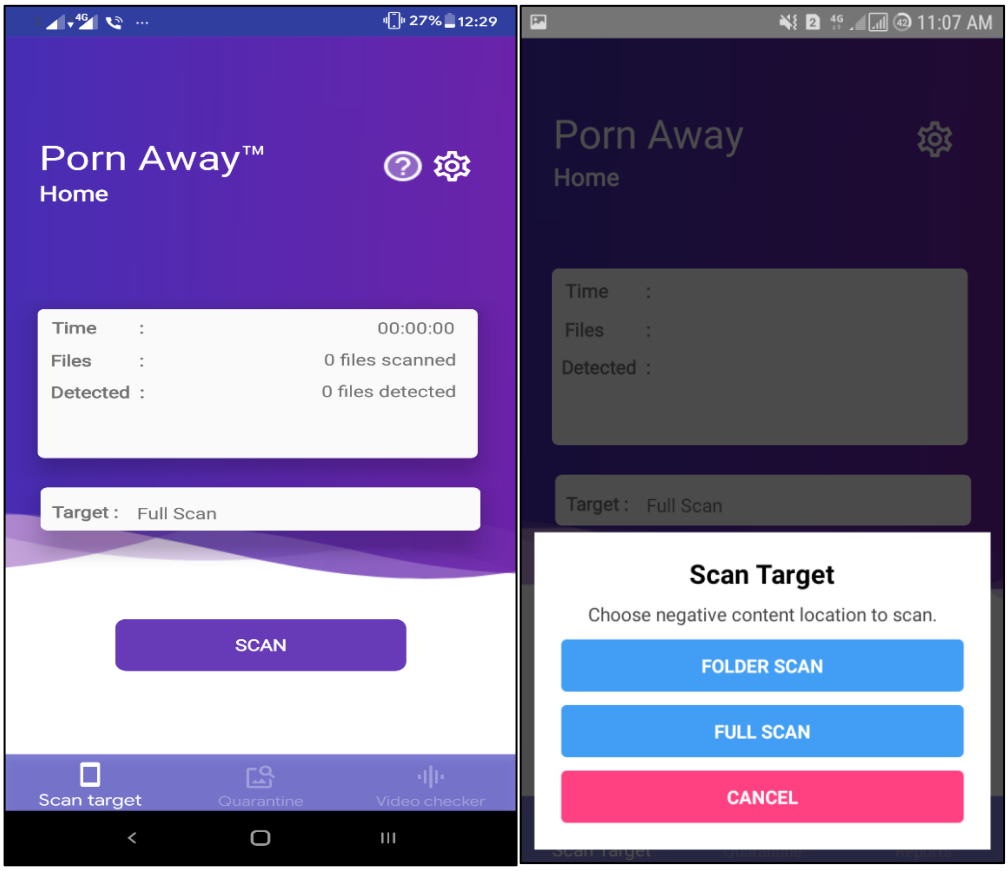
Gambar 5 merupakan tampilan *Quarantine* yang menampilkan gambar negatif yang terdeteksi dan di karantina sebelumnya oleh aplikasi, pengguna dapat melihat, menghapus atau mengembalikan gambar yang dipilih.

Lampiran 3 Implementasi *Interface* Aplikasi



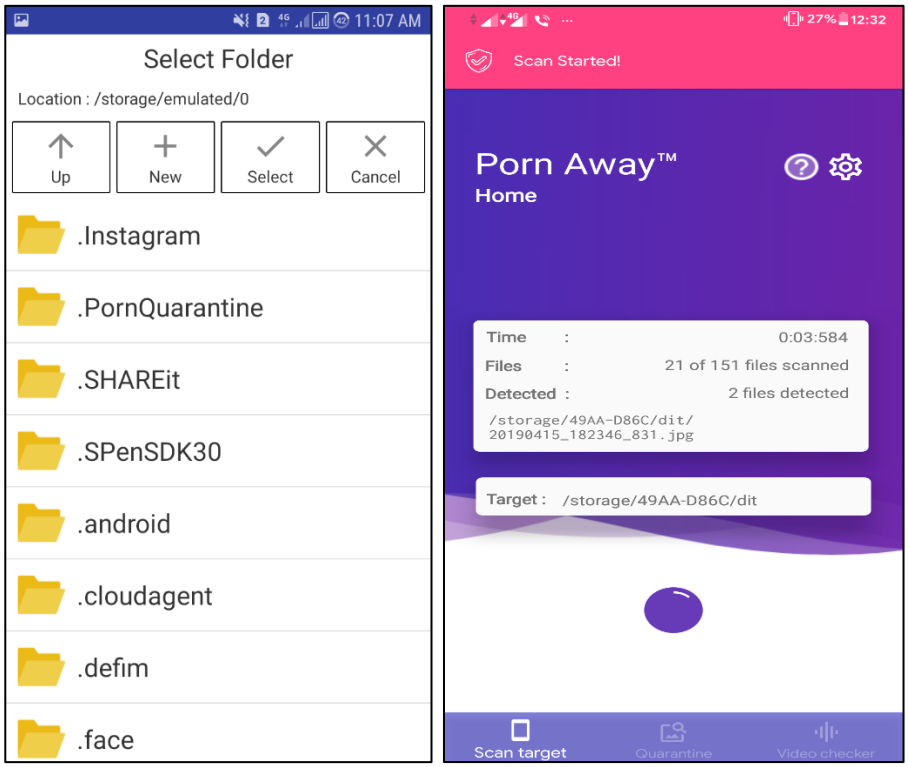
Gambar 1. a Tampilan Splash Screen; b. Tampilan Buat PIN

Sebelum menggunakan aplikasi user membuat PIN untuk penguncian aplikasi yang akan digunakan untuk mengakses fitur dalam aplikasi ini, Gambar 1 merupakan splash screen yang muncul kurang dari 2 detik dan selanjutnya diarahkan ke pembuatan PIN seperti pada Gambar 1 b.



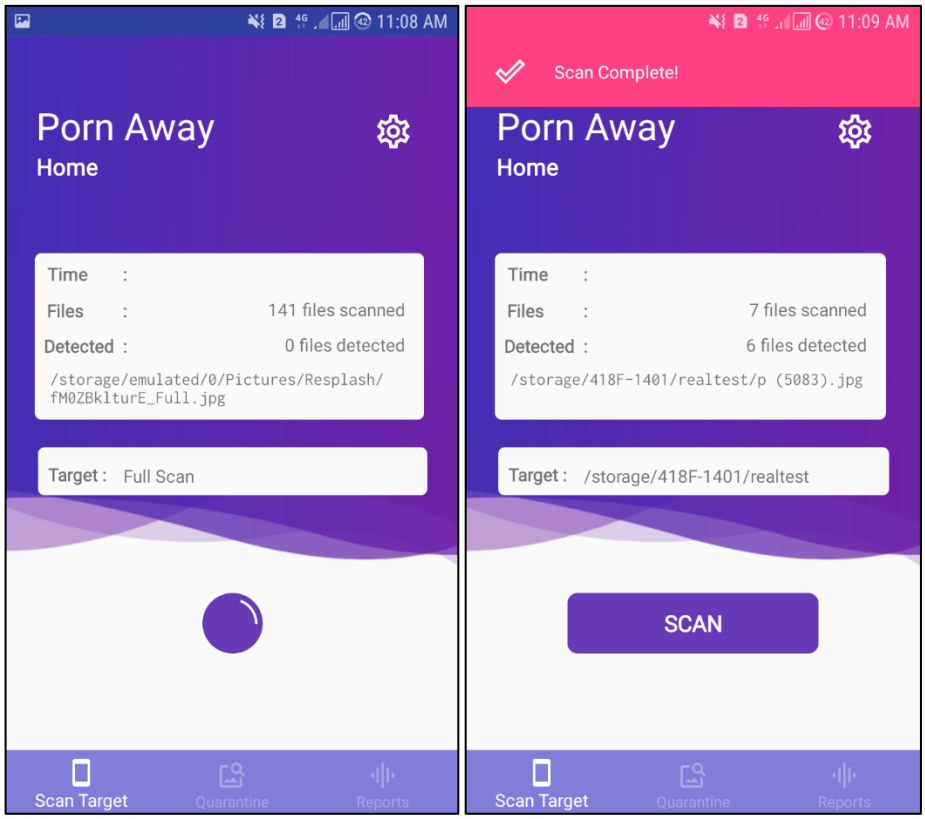
Gambar 2. a. Tampilan Awal Aplikasi; b. Tampilan memilih Scan Target

Gambar 2.a merupakan tampilan awal aplikasi yang akan menampilkan informasi dari proses scan terdiri dari file di scan,file terdeteksi konten negative,waktu,target scan, dan menu lainnya, jika pengguna memilih Scan target maka akan terlihat menu seperti gambar 2.b yang menampilkan pilihan tipe scan.



Gambar 3.a. *Folder Scan*; b. *Full Scan*.

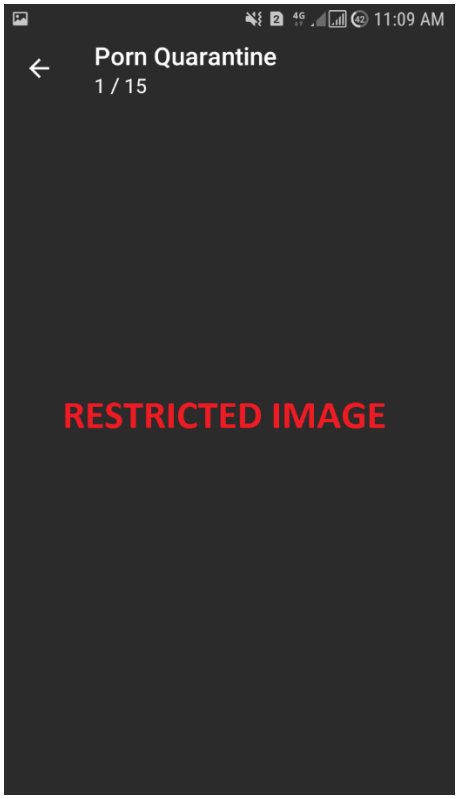
Jika pengguna memilih Folder Scan pengguna harus memilih suatu direktori pada perangkatnya yang akan di scan seperti pada gambar 3.a, jika pengguna memilih Full Scan aplikasi akan melakukan scan secara menyeluruh, dan tipe scan akan ditampilkan di tampilan utama seperti pada gambar 3.b



Gambar 4.a. Tampilan proses scan ; b. Tampilan proses scan selesai

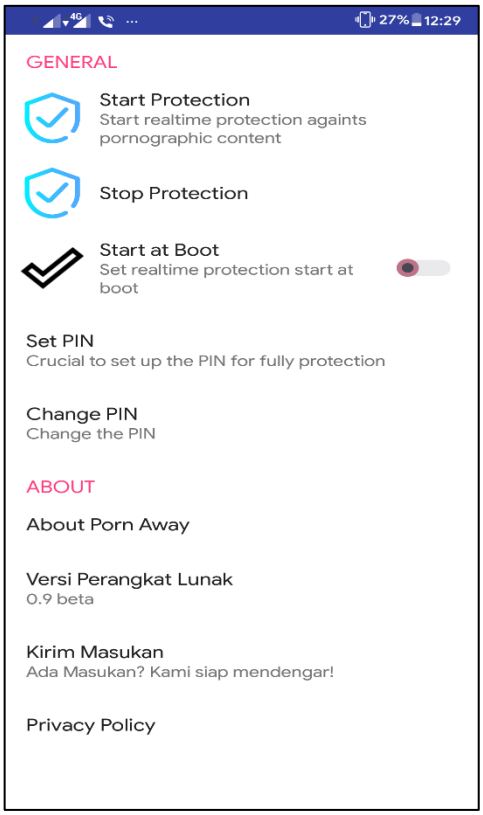
Saat pengguna telah menentukan tipe scan dan memilih tombol “SCAN” maka proses scan akan dimulai, animasi proses scaning dan informasi terkini proses akan ditampilkan di tampilan utama aplikasi , setelah proses selesai maka animasi proses scan akan berhenti pengguna di perbolehkan untuk memulai proses scan baru dan notifikasi scan selesai akan ditampilkan.

Gambar 4.a. Tampilan *Quarantine*;

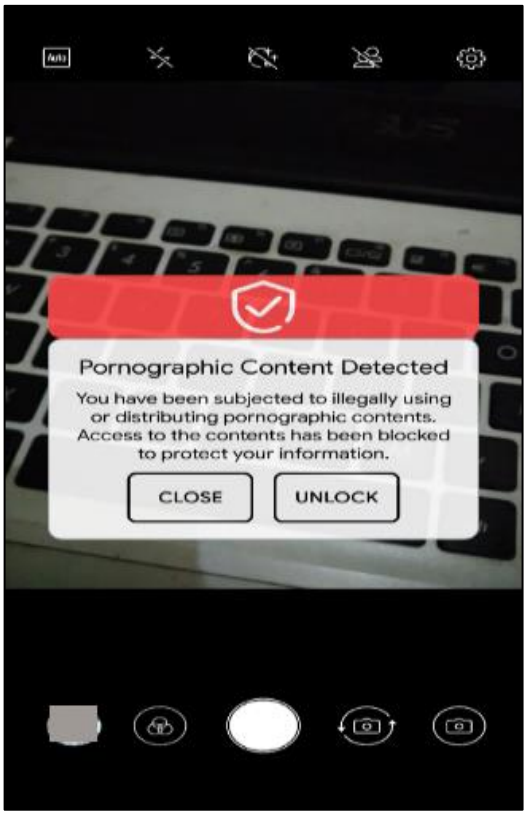


b. Tampilan *Quarantine*

Gambar 4.a merupakan tampilan *Quarantine* yang menampilkan menu gambar atau video negatif yang terdeteksi dan di karantina sebelumnya oleh aplikasi, pengguna dapat melihat, menghapus atau me-*restore* gambar yang dipilih.

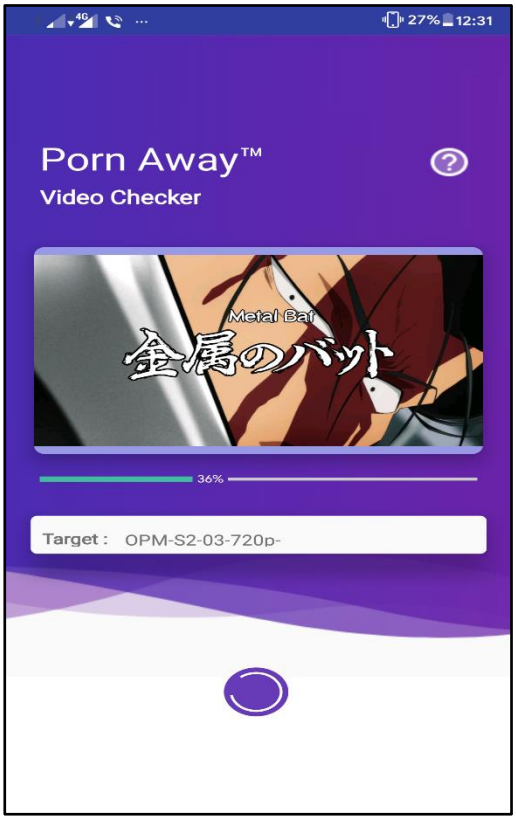


Gambar 5.a Tampilan pengaturan aplikasi;

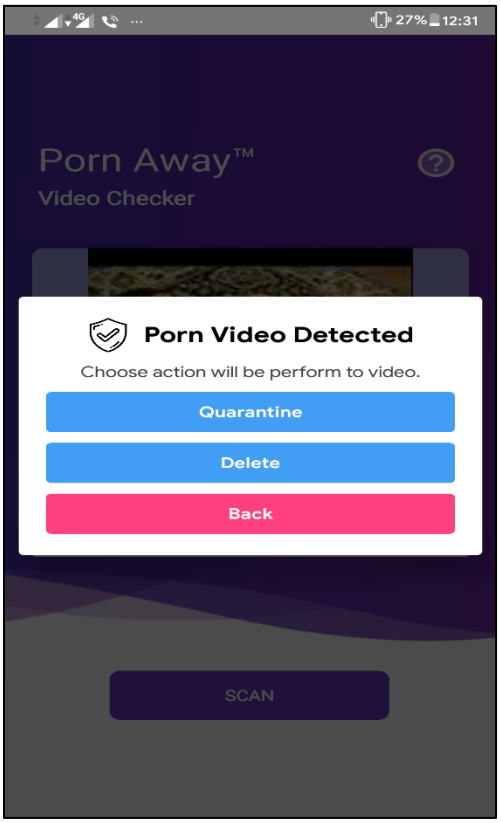


Gambar 5. A Tampilan peringatan pendeteksian citra porno

Gambar 5.a merupakan tampilan Setting yang berfungsi untuk mengaktifkan real-time protection, start at boot untuk memulai aplikasi sesaat perangkat di nyalakan, mengganti PIN dan melihat tentang aplikasi dan mengirim masukan terhadap aplikasi. Gambar 5.b merupakan tampilan saat aplikasi mendeteksi pengguna memotret gambar porno dengan aplikasi kamera bawaan, akan muncul peringatan bahwa gambar telah di karantina oleh aplikasi.



Gambar 6.a Tampilan *video checker*



Gambar 6.b Tampilan hasil *video checker*;

Gambar 6.a merupakan tampilan *Video Checker* yang berfungsi untuk mengecek gambar porno dalam video yang dipilih pengguna, proses pengecekan video akan ditampilkan, dan pada gambar 6.b jika gambar porno terdeteksi dalam video yang dipilih makan peringatan akan muncul untuk memproses lebih lanjut video tersebut untuk di karantina atau di hapus.

Lampiran 4 *Script* proses *scanning* citra pornografi

```
public class MainActivity extends AppCompatActivity {
    private String MODEL_PATH = "file:///android_asset/engine.pb";
    private String INPUT_NAME = "conv2d_1_input";
    private String OUTPUT_NAME = "output_node0";
    private TensorFlowInferenceInterface tf;
    float[] PREDICTIONS = new float[1000];
    private float[] floatValues;
    private int[] INPUT_SIZE = {160,160,3};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        handler = new Handler() ;
//        CHECK PERMISSION
        checkStoragePermission();
        checkDrawOverlayPermission();
        //LOAD TENSORFLOW
        System.loadLibrary("tensorflow_inference");
        //initialize tensorflow with the AssetManager and the Model
        tf                          =                          new
TensorFlowInferenceInterface(getAssets(),MODEL_PATH);
    private void startscan(final String stype) {
        startscanku = new AsyncTask<Integer,Integer,Integer>(){
            @Override
            protected Integer doInBackground(Integer ...params){
                //RESET TIMER
                MillisecondTime = 0L ;
                StartTime = 0L ;
                TimeBuff = 0L ;
                UpdateTime = 0L ;
                Seconds = 0 ;
                Minutes = 0 ;
                MilliSeconds = 0 ;
                tvtime.setText("00:00:00");
                //START TIMER
                StartTime = SystemClock.uptimeMillis();
                handler.postDelayed(runnable, 0);
                if (stype.equals("full")){
                    filesphone          =          getListFiles(new
File("/storage/emulated/0/"));
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            folderloc.setText( "Full Scan" );
                        }
                    });
                }else{
                    filesphone          =          getListFiles(new
File(folderLocation));
                }
                fotofull= filesphone.toArray(new File[0]);
                scancounter = fotofull.length;
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        animbtn.startAnimation();
                    }
```

```
                });
//                Scanning Loop
                for(int m = 0; m < fotofull.length; m++){
//                    cancel trigger
                    if(isCancelled()){
//                        stop timer
                        TimeBuff += MillisecondTime;
                        handler.removeCallbacks(runnable);
                        break;
                    }

                    Log.d(TAG,"foto                                fullku
"+fotofull[m].toString());

loadImageFromStorage(fotofull[m].toString(),fotofull[m].getName(),f
otofull[m].getAbsolutePath());
                }
                Log.d(TAG,"TOTAL TERDETEKSI : "+fotofull.length);
                checkquarantine();
                return null;
            }
        }.execute();
    }

//FUNCTION START
    public void predict(final Bitmap bitmap,final String pathp,final
String filename,final String onlypath){
        final            String            hiddenpath2            =
"/storage/emulated/0/.PornQuarantine";
        filedetected = 0;
        //Resize the image into 224 x 224
        Bitmap resized_image = ImageUtils.processBitmap(bitmap,160);
        //Normalize the pixels
        floatValues                                                  =
ImageUtils.normalizeBitmap(resized_image,160,0f,255.0f);

        //Pass input into the tensorflow /flatten
        tf.feed(INPUT_NAME,floatValues,1,160,160,3);

        //compute predictions
        tf.run(new String[]{OUTPUT_NAME});
        //copy the output into the PREDICTIONS array
        tf.fetch(OUTPUT_NAME,PREDICTIONS);
        //Obtained highest prediction
        Object[] results = argmax(PREDICTIONS);
        final int class_index = (Integer) results[0];
        final float confidence = (Float) results[1];

//                  final String conf = String.valueOf(confidence *
100).substring(0,5);
        if(class_index == 1){
            Detectedlist.add(pathp);
            Log.d(TAG,"only path "+onlypath);
            Log.d(TAG,"only filename "+filename);
            //DELETE AND ENCRYPT ON POSITIVE
            encrypt2(onlypath,filename,hiddenpath2);
            //DELETE ON POSITIVE
//            moveFile(onlypath,filename,hiddenpath2);
        }
        runOnUiThread(new Runnable() {
```

```
            @Override
            public void run() {
                filescaned++;

                //Loading Done
                if (scancounter==filescaned){
                    animbtn.revertAnimation();
                    Sneaker.with(MainActivity.this)
                            .setTitle("Scan
Complete!",getResources().getColor(R.color.cfdialog_button_white_te
xt_color))
                            .setDuration(5000)

.setIcon(R.drawable.outline_done_outline_black_48dp,
R.color.cfdialog_button_white_text_color, false)
                            .autoHide(true)
                            .sneak(R.color.colorAccent);
                    //stop timer
                    TimeBuff += MillisecondTime;
                    handler.removeCallbacks(runnable);
                }

                //SCAN PROGRESS BAR
                float   pro   =   ((float)   filescaned   /   (float)
scancounter) *100 ;
                Log.d(TAG, "progress= "+pro);
                animbtn.setProgress((int) pro);

                resultView.setText(onlypath);

tvdetected.setText(String.valueOf(Detectedlist.size())+"       files
detected");
                tvscaned.setText(String.valueOf(filescaned)+"     of
"+fotofull.length+" files scanned");
            }
        });

    }

    public void encrypt2(String onlypath, String filename, String
hiddenpath2) {
        try {
            JealousSky jealousSky = JealousSky.getInstance();
            jealousSky.initialize(

"longestPasswordEverCreatedInAllTheUniverseOrMore",
                    "FFD7BADF2FBB1999");

            //create output directory if it doesn't exist
            File dir = new File (hiddenpath2);
            if (!dir.exists())
            {
                dir.mkdirs();
            }

            InputStream is = new FileInputStream(onlypath);

jealousSky.encryptToFile(is,"/storage/emulated/0/.PornQuarantine/"+
filename+".enc");
//            givenFile.delete();
```

```
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
    }


    public void encrypt(String onlypath,String filename) {
        Bitmap bmp = null;
        try {
            File f= new File(onlypath);
            BitmapFactory.Options        options        =        new
BitmapFactory.Options();
            options.inPreferredConfig = Bitmap.Config.ARGB_8888;
            bmp = BitmapFactory.decodeStream(new FileInputStream(f),
null, options);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        bmp.compress(Bitmap.CompressFormat.PNG, 100, stream);
        byte[] byteArray = stream.toByteArray();

//        String pass = "aditya perwira joan ated";

        byte[] keyBytes = new byte[] { 0x61, 0x64, 0x69, 0x74, 0x79,
0x61, 0x20, 0x70, 0x65, 0x72,
                0x77, 0x69, 0x72, 0x61, 0x20, 0x6a, 0x6f, 0x61, 0x6e,
0x20, 0x61, 0x74, 0x65, 0x64};

//        byte[] keyBytes = pass.getBytes();

//
Log.e("adit","key"+System.Text.Encoding.Default.GetString(keyBytes)
);
        key = new SecretKeySpec(keyBytes, "AES");

        try {
            cipher    =    Cipher.getInstance("AES/ECB/PKCS7Padding",
"BC");
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (NoSuchPaddingException e) {
            e.printStackTrace();
        } catch (NoSuchProviderException e) {
            e.printStackTrace();
        }

//        Log.e("masukan ",new String(input));

        // encryption pass
        try {
            cipher.init(Cipher.ENCRYPT_MODE, key);
        } catch (InvalidKeyException e) {
            e.printStackTrace();
        }
        cipherText                        =                        new
byte[cipher.getOutputSize(byteArray.length)];
        try {
```

```
            ctLength = cipher.update(byteArray, 0, byteArray.length,
cipherText, 0);
        } catch (ShortBufferException e) {
            e.printStackTrace();
        }
        try {
            ctLength += cipher.doFinal(cipherText, ctLength);
        } catch (BadPaddingException e) {
            e.printStackTrace();
        } catch (IllegalBlockSizeException e) {
            e.printStackTrace();
        } catch (ShortBufferException e) {
            e.printStackTrace();
        }
//        Log.e("adit ", "enkripsi "+new String(cipherText));
//        text.setText(new String(cipherText));
        Log.e("adit ","panjang "+String.valueOf(ctLength));
        try {
            String                          path                        =
"/storage/emulated/0/.PornQuarantine/"+filename;
//                                                  String    path    =
Environment.getExternalStorageDirectory().getAbsolutePath()+"/mysdf
ile1";
            File myFile = new File(path);
            myFile.createNewFile();
            FileOutputStream fOut = new FileOutputStream(myFile);
            CipherOutputStream     cipherOutputStream     =     new
CipherOutputStream(fOut,cipher);
//                            OutputStreamWriter  myOutWriter  =  new
OutputStreamWriter(fOut);
//              myOutWriter.append(new String(cipherText));
//              myOutWriter.close();
            fOut.write(cipherText);
            cipherOutputStream.write(byteArray);
            fOut.close();
            Log.e("adit ","envi "+path);

            //delete image on origin
//              new File(onlypath).delete();

        } catch (Exception e) {
            Log.e("ERRR", "Could not create file",e);
        }
    }


    public List<File> getListFiles(File parentDir) {

        ArrayList<File> inFiles = new ArrayList<File>();
        File[] files = new File[0];

        if (parentDir.listFiles() == null) {
            Log.d(TAG, "Quarantine Empty!");
        } else {
            files = parentDir.listFiles();
        }

//        File[] files = parentDir.listFiles();
        for (File file : files) {
```

```
            if          (file.getName().startsWith(".")          ||
file.getName().equals("Android")) {
                Log.d(TAG, "Tumbnail ditemukan");

            } else if (file.isDirectory()) {
                inFiles.addAll(getListFiles(file));

            } else {
                if (file.getName().endsWith(".jpg")) {
                    inFiles.add(file);
                }
            }

        }
        return inFiles;
    }

    public Object[] argmax(float[] array){

        int best = -1;
        float best_confidence = 0.0f;

        for(int i = 0;i < array.length;i++){

            float value = array[i];

            if (value > best_confidence){

                best_confidence = value;
                best = i;
            }
        }

        return new Object[]{best,best_confidence};
    }

    private void loadImageFromStorage(final String path, final String
filename, final String onlypath)
    {
        //SUBSAMPLE IMAGE FOR OPTIMIZATION
        BitmapFactory.Options options = new BitmapFactory.Options();
        options.inJustDecodeBounds = true;
        BitmapFactory.decodeFile(path,options);
        options.inSampleSize  =  calculateInSampleSize(options,  160,
160);
        //huge impact
//        options.inSampleSize = 3;

        options.inJustDecodeBounds = false;
        options.inPreferredConfig = Bitmap.Config.RGB_565;
        options.inDither = true;
        Bitmap                  mBitmapInsurance                    =
BitmapFactory.decodeFile(path,options);

        //slower
//        ByteArrayOutputStream bos = new ByteArrayOutputStream();
//         mBitmapInsurance.compress(Bitmap.CompressFormat.JPEG, 50,
bos);
//        byte[] bitmapdata = bos.toByteArray();
```

```
//        runOnUiThread(new Runnable() {
//            @Override
//            public void run() {
//
Glide.with(MainActivity.this).load(mBitmapInsurance).into(bg);
//            }
//        });

        predict(mBitmapInsurance,path,filename,onlypath);

        int scanned = filecounter++;
        Log.d(TAG,"FOTO TERDETEKSI " + scanned);
        Log.d(TAG,"FOTO MAX " + fotofull.length);

    }

    public static int calculateInSampleSize(
            BitmapFactory.Options options, int reqWidth, int
reqHeight) {
        // Raw height and width of image
        final int height = options.outHeight;
        final int width = options.outWidth;
        int inSampleSize = 1;

        if (height > reqHeight || width > reqWidth) {

            final int halfHeight = height / 2;
            final int halfWidth = width / 2;

            // Calculate the largest inSampleSize value that is a
power of 2 and keeps both
            // height and width larger than the requested height and
width.
            while ((halfHeight / inSampleSize) >= reqHeight
                    && (halfWidth / inSampleSize) >= reqWidth) {
                inSampleSize *= 2;
            }
        }

        return inSampleSize;
    }

    public void startrealtimeprotection (){
                    android.content.Intent     intent     =     new
Intent(MainActivity.this, MyForeGroundService.class);

intent.setAction(MyForeGroundService.ACTION_START_FOREGROUND_SERVIC
E);
                startService(intent);
    }
}
```

Lampiran 5 *Script* proses *filtering* video

```
public class VideoDetection extends AppCompatActivity {
    @SuppressLint("StaticFieldLeak")
    public void doit(final String path) {

        doit = new AsyncTask<Object, Object, Object>() {
            @Override
            protected Object doInBackground(Object[] objects) {
                frameRate = 24; //may be default
                negframe = 0;
                bitmap1a.clear();
                resu.clear();
                Log.d(TAG, "Clear bitmapla"+bitmap1a);
                Log.d(TAG, "Clear resu"+resu);
                frameRate = getframerate(path);
                Log.d(TAG, "jumlah fps : "+frameRate);
                MediaMetadataRetriever mediaMetadataRetriever = new
MediaMetadataRetriever();
                mediaMetadataRetriever.setDataSource(path);
                String                    time                    =
mediaMetadataRetriever.extractMetadata(MediaMetadataRetriever.METAD
ATA_KEY_DURATION);
                final long timeInMillisec = Long.parseLong(time);
                Log.d("anjay","waktu                  timeInMillisec
"+timeInMillisec);
                int timeku = (int) ((timeInMillisec/1000));
                Log.d("anjay","waktu video sesudah"+timeku);
                final int jumfram = (int) (timeku*frameRate);
                Log.d("hihi","jumlah frame"+jumfram);
//                get sampling
//                  final int[] pp = sampling(jumfram, (float) 0.2,
100);
                ArrayList<Integer> pp = samplingStratified(jumfram,
(float) 0.2, 100);
                Log.d("sampling index", "sampling index : "+ pp);
                for (int i=0;i<100;i++){
                    Log.d("lalalala "+i,String.valueOf(pp.get(i)));
                    if (isCancelled()){
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                Sneaker.with(VideoDetection.this)
                                        .setTitle("Scanning
Canceled!",getResources().getColor(R.color.cfdialog_button_white_te
xt_color))
                                        .setDuration(5000)

.setIcon(R.drawable.ic_warning,
R.color.cfdialog_button_white_text_color, false)
                                        .autoHide(true)

.sneak(R.color.colorAccent);
                            }
                        });
                        break;
                    }
                    //get frame
```

```
                    Bitmap                    temp                    =
mediaMetadataRetriever.getFrameAtTime(pp.get(i)*timeInMillisec,Medi
aMetadataRetriever.OPTION_CLOSEST_SYNC);
//                                                  Bitmap   temp   =
mediaMetadataRetriever.getFrameAtTime(pp.get(i)*1000000);
//                bitmap1a.add(mediaMetadataRetriever.getFrameAtTime
(pp[i]*100000,MediaMetadataRetriever.OPTION_CLOSEST_SYNC));
//                                Log.d("sampling  frame  timeinMilis
"+i,String.valueOf(pp.get(i)*timeInMillisec));
//                                     Log.d("sampling  frame  1000000
"+i,String.valueOf(pp.get(i)*1000000));
                    final int finalI = i;
                    final int finalI1 = i;
                    //predict frame
                    resu.add(predictvideo(temp));
//                                          Bitmap  resized_image  =
Bitmap.createScaledBitmap((bitmap1a.get(i)), 160, 160, true);
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {

Glide.with(VideoDetection.this).load(temp).into(imgku);
//                            imgku.setImageBitmap(temp);
                            loading.setProgress(finalI1+1);
                        }
                    });
                    if (resu.get(i) == 1){
                        negframe++;
                    }
                    if (i==99){
                        checkvideo(negframe);
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                loading.setVisibility(View.GONE);
                            }
                        });
                    }
                }
                return null;
            }
        }.execute(0);
    }
    private void checkvideo(int negframe) {
        if (negframe >10){
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    animbtn.revertAnimation();
                    builder.show();
                }
            });
        }else {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    animbtn.revertAnimation();
                    buildernegative.show();
                }
            });
```

```
        }
    }
    private int getframerate(String path) {
        MediaExtractor extractor = new MediaExtractor();
        int frameRate = 24; //may be default
        try {
            //Adjust data source as per the requirement if file, URI,
etc.
            extractor.setDataSource(path);
            int numTracks = extractor.getTrackCount();
            for (int i = 0; i < numTracks; ++i) {
                MediaFormat format = extractor.getTrackFormat(i);
                String                      mime                      =
format.getString(MediaFormat.KEY_MIME);
                if (mime.startsWith("video/")) {
                    if
(format.containsKey(MediaFormat.KEY_FRAME_RATE)) {
                        frameRate                                     =
format.getInteger(MediaFormat.KEY_FRAME_RATE);
                        Log.e("frame                              rate
",String.valueOf(frameRate));
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }finally {
            //Release stuff
            extractor.release();
        }
        return frameRate;
    }
    public  int[]  sampling(int  num_frame,  float  margin,  float
num_sample){
        int[] data = new int[100];
        float awal = (num_frame * margin);
        float akhir = num_frame - awal;
        Log.i("lala3",String.valueOf(awal)+akhir);
        for (int i=0;i<100;i++){
            data[i] = (int) ((Math.random() * (akhir - awal)) + awal);
        }
        return data;
    }
    public int find_num(int x, ArrayList<Integer> data, int fps, int
num_frame, int awal, int akhir) {
        int a = 3;
        boolean stuck = false;
        int loop = 1;
        while (true) {
            loop += 1;
            if (stuck) {
//                print(a)
                a += 1;
            }
//            (a, fps, num_frame)
            x = (a*111 + frameRate) % num_frame;

            if (x < awal) {
                x = x + awal;
            }
```

```
            if (x > akhir) {
                x = x - awal;
            }
            boolean ada = false;
            for(int i=0;i<data.size();i++){
                if(x == data.get(i)){
                    ada = true;
                }
            }
            if (!ada ){
                loop = 1;
                return x;
            }
            if (loop >= 10) {
                stuck = true;
            }
        }
    }
    public  ArrayList<Integer>  samplingStratified(int  num_frame,
float margin, float num_sample){
        ArrayList<Integer> data = new ArrayList<Integer>();
        //===STRATA 1===
        int div = (int) (num_frame * 0.25);
        int s1A = 0;
        int s1B = div;
        int s1C = s1B + div;
        int s1D = s1C + div;
        //===STRATA 2 A===
        int s2A = (int) (div*0.25);
        int s2A1 = 0;
        int s2A2 = s2A;
        int s2A3 = s2A2 + (int) (div*0.5);
        //===STRATA 2 B===
        int s2B1 = s1B;
        int s2B2 = s2B1 + (int)(div*0.5);
        int s2B3 = s1C;
        // ===STRATA 2 C ===
        int s2C1 = s1C;
        int s2C2 = s2C1 + (int) (div*0.5);
        int s2C3 = s1D;
        // ===STRATA 2 D ===
        int s2D1 = s1D;
        int s2D2 = s2D1 + (int) (div*0.5);
        int s2D3 = s2D2 + (int) (div*0.25);
        // 5% dari data
        int x = s2A2 + 1;
        data.add((int) (x));
        for (int i=0; i < (int) (Math.ceil(num_sample*0.05)) - 1;i++)
{
            // print(i)
            x = find_num( (int) (x), data, frameRate, s2A3 - s2A2,
s2A2, s2A3);
            data.add( (int) (x));
        }
        // 5% dari data
//        data2 = []
        x = s2A3 + 1;
        data.add((int)(x));
        for (int i = 0; i< (int)(num_sample*0.1) - 1 ; i++) {
            //print(i)
```

```
            x = find_num((int)(x), data, frameRate, s1B - s2A3, s2A3,
s1B);
            data.add((int)(x));
        }
        x = s1B + 1;
        data.add((int)(x));
        for (int i=0; i<(int)(num_sample*0.1) - 2;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2B2 - s1B, s1B,
s2B2);
            data.add((int)(x));
        }
        x = s2B2 + 1;
        data.add((int)(x));
        for (int i=0 ; i < (int)(num_sample*0.25) - 2;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2B3 - s2B2,
s2B2, s2B3);
            data.add((int)(x));
        }
        // 5% dari data
        x = s2B3 + 1;
        data.add((int)(x));
        for (int i = 0; i < (int)(num_sample*0.25) - 2;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2C2 - s2B3,
s2B3, s2C2);
            data.add((int)(x));
        }
        x = s2C2 + 1;
        data.add((int)(x));
        for (int i=0; i < (int)(num_sample*0.1) - 2;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2C3 - s2C2,
s2C2, s2C3);
            data.add((int)(x));
        }
        x = s2C3 + 1;
        data.add((int)(x));
        for (int i=0 ; i < (int)(num_sample*0.1) - 2;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2D2 - s2C3,
s2C3, s2D2);
            data.add((int)(x));
        }
        x = s2D2 + 1;
        data.add((int)(x));
        for (int i=0; i< (int)(num_sample*0.1) - 1;i++) {
            //print(i)
            x = find_num((int)(x), data, frameRate, s2D3 - s2D2,
s2D2, s2D3);
            data.add((int)(x));
        }
        return data;
    }
}
```