TUGAS AKHIR – EC184801

CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI CITRA ASAP PADA GAMBAR SATELIT

Kentani Langgalih Prioko
NRP 0721 15 4000 0045

Dosen Pembimbing
Reza Fuad Rachmadi, ST., MT., P.hD.
Dr. Supeno Mardi Susiki Nugroho, ST., MT.

DEPARTEMEN TEKNIK KOMPUTER
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

TUGAS AKHIR – EC184801

CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI CITRA ASAP PADA GAMBAR SATELIT

Kentani Langgalih Prioko
NRP 0721 15 4000 0045

Dosen Pembimbing
Reza Fuad Rachmadi, ST., MT., P.hD.
Dr. Supeno Mardi Susiki Nugroho, ST., MT.

DEPARTEMEN TEKNIK KOMPUTER
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

Institut
Teknologi
Sepuluh Nopember

FINAL PROJECT – EC184801

CONVOLUTIONAL NEURAL NETWORK FOR SMOKE IMAGES CLASSIFICATION ON SATELLITE IMAGERY

Kentani Langgalih Prioko
NRP 0721 15 4000 0045

Advisor
Reza Fuad Rachmadi, ST., MT., P.hD.
Dr. Supeno Mardi Susiki Nugroho, ST., MT.

Departement of Computer Engineering
Faculty of Intelligent Electrical and Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2020

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul **"*Convolutional Neural Network* Untuk Klasifikasi Citra Asap Pada Gambar Satelit"** adalah benar-benar hasil karya intelektual sendiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya orang lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun durujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku.

Surabaya, Juni 2020

Kentani Langgalih Prioko
NRP. 07211540000045

LEMBAR PENGESAHAN

***Convolutional Neural Network* Untuk Klasifikasi Citra Asap pada Gambar Satelit**

Tugas Akhir ini disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana Teknik di Institut Teknologi Sepuluh Nopember Surabaya

Oleh: Kentani Langgalih Prioko (NRP: 07211540000045)

Tanggal Ujian : 8 Juli 2020 Periode Wisuda : September 2020

Disetujui oleh:

Reza Fuad Rachmadi, ST., MT., Ph.D. (Pembimbing I)
NIP: 198504032012121001

Dr. Supeno Mardi Susiki Nugroho, ST., MT. (Pembimbing II)
NIP: 197003131995121001

Dr. Eko Mulyanto Yuniarno, ST., MT. (Penguji I)
NIP: 196806011995121009

Susi Juniastuti, ST., M.Eng. (Penguji II)
NIP. 196506181999032001

Eko Pramunanto, ST., MT. (Penguji III)
NIP. 196612031994121001

Mengetahui
Kepala Departemen Teknik Komputer

Dr. Supeno Mardi Susiki Nugroho, ST., MT.
NIP. 197003131995121001

ABSTRAK

Nama Mahasiswa	:	Kentani Langgalih Prioko
Judul Tugas Akhir	:	*Convolutional Neural Network* Untuk Klasifikasi Citra Asap Pada Gambar Satelit
Pembimbing	:	1.Reza Fuad Rachmadi, ST., MT., P.hD. 2. Dr. Supeno Mardi Susiki Nugroho, S.T., M.T.

Pendeteksian kebakaran dapat dilakukan sedini mungkin dengan bantuan teknologi agar titik api dapat diketahui dengan cepat dan dapat segera dilakukan tindakan lebih lanjut. Pada penelitian ini akan dikembangkan sebuah sistem yang diharapkan dapat mengklasifikasi citra asap pada gambar satelit menggunakan *Convolutional Neural Network* dengan menggunakan dataset *USTC SmokeRS* yang berjumlah sebanyak 6225 gambar dan terpisah menjadi 6 kelas. Proses klasifikasi citra akan menggunakan empat macam *pre-trained* model *Convolutional Neural Network* yaitu *MobileNet V2*, *Inception V3*, *InceptionResNet V2*, dan *NASNet*. Hasil pengujian menunjukkan bahwa *pre-trained* model *MobileNet V2*, *Inception V3*, *InceptionResNet V2*, dan *NASNet* masing - masing memiliki nilai akurasi sebesar 83,30%, 83,30%, 88,33%, dan 86,67%.

Kata Kunci: Kebakaran, Asap, Klasifikasi Citra, *Convolutional Neural Network*

Halaman ini sengaja dikosongkan

ABSTRACT

Name	:	Kentani Langgalih Prioko
Title	:	*Convolutional Neural Network for Smoke Images Classification on Satellite Imagery*
Advisors	:	1.Reza Fuad Rachmadi, ST., MT., P.hD. 2. Dr. Supeno Mardi Susiki Nugroho, S.T., M.T.

Fire detection can be done as early as possible with the help of technology so that hotspots can be known quickly and further action can be taken immediately. In this research, a system which is expected to be able to classify smoke images based on satellite images will be developed using *Convolutional Neural Network* using the *USTC SmokeRS* dataset, totaling 6225 images and separated into 6 classes. The image classification process will use four types of *pre-trained* model *Convolutional Neural Network* which are *MobileNet V2*, *Inception V3*, *InceptionResNet V2*, and *NASNet* . The results of the testing show that *pre-trained* model *MobileNet V2*, *Inception V3*, *InceptionResNet V2*, and *NASNet* respectively - has an accuracy value of 83.30%, 83.30%, 88.33%, and 86.67%.

Keywords: Fire, Smoke, Image Classification, Convolutional Neural Network

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT atas segala limpahan berkah, rahmat, serta hidayah-Nya, penulis dapat menyelesaikan penelitian ini dengan judul ***Convolutional Neural Network* Untuk Klasifikasi Citra Asap Pada Gambar Satelit.**

Penelitian ini disusun dalam rangka pemenuhan bidang riset di Departemen Teknik Komputer, serta digunakan sebagai persyaratan menyelesaikan pendidikan S1. Penelitian ini dapat terselesaikan tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Kedua orang tua yang telah memberikan dorongan spiritual dan material dalam penyelesaian buku penelitian ini.
2. Seluruh keluarga besar yang berada di Bekasi yang telah memberikan doa.
3. Bapak Kepala Departemen Teknik Komputer ITS Dr. Supeno Mardi Susiki Nugroho, ST., MT.
4. Bapak Reza Fuad Rachmadi, ST., MT., P.hD. dan Bapak Dr. Supeno Mardi Susiki Nugroho, ST., MT. selaku dosen pembimbing yang senantiasa memberikan motivasi, bimbingan, arahan serta dukungan selama mengerjakan penelitian ini.
5. Bapak-ibu dosen pengajar Departemen Teknik Komputer ITS, atas pengajaran, bimbingan, serta perhatian yang diberikan kepada penulis selama masa perkuliahan yang telah ditempuh.
6. Seluruh Staf Departemen Teknik Komputer ITS yang telah membantu penulis dalam hal administrasi dan perlengkapan penelitian.
7. Teman seperjuangan B507 yang ikut serta memberi semangat dalam proses pengerjaan penelitian ini.
8. Teman-teman dari ITSSMANSASI15 yang telah menemani selama masa studi saya di Surabaya.
9. Teman-teman e-55 dan teman-teman Teknik Komputer angkatan 2015 dan 2016 yang telah memberikan dukangan serta bantuan.

Kesempurnaan hanya milik Allah SWT, untuk itu penulis memohon segenap kritik dan saran yang membangun. Semoga penelitian ini dapat memberikan manfaat bagi kita semua. Aamiin.

Surabaya, Juni 2020

Kentani Langgalih Prioko

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR NOMENKLATUR

1. W : Panjang / tinggi input.
2. N : Panjang / tinggi filter.
3. P : Zero padding.
4. S : Stride.
5. x : Nilai aktivasi *layer*.
6. $\hat{x}$: Nilai normalisasi *hidden layer*.
7. BN : *Batch Normalization*.
8. β : *Mini batch*.
9. ϵ : *Variance hidden layer*.
10. $f(x_t)$: Turunan fungsi pada titik x_t.
11. x^i : Nilai observasi.
12. y^i : Nilai prediksi.

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

Penelitian ini di latar belakangi oleh berbagai kondisi yang menjadi acuan. Selain itu juga terdapat beberapa permasalahan yang akan dijawab sebagai luaran dari penelitian.

1.1 Latar belakang

Kebakaran hutan merupakan salah satu bencana alam yang menimbulkan ancaman serius bagi lingkungan hidup. Asap yang muncul akibat dari kebakaran hutan memperburuk kualitas udara karena mengandung zat-zat yang berbahaya seperti Karbon Monoksida (CO), Sulfur Dioksida (SO2), Nitrogen Dioksida (NO2), dan Ozon Permukaan (O3) [1]. Salah satu penyebab terjadinya kebakaran hutan adalah meningkatnya suhu bumi akibat perubahan iklim.

Berdasarkan data yang dirilis oleh NASA Goddard Institute for Space Studies, suhu bumi pada tahun 2014 lebih panas 0,75°C dari rata-rata suhu bumi. 0,9 °C (2015), 1,02 °C (2016), 0,93 °C (2017), dan 0,85 °C (2018) [2] [3]. Meningkatnya suhu bumi akan membuat udara lebih kering dan panas, sehingga akan menciptakan ekosistem yang rentan terbakar.

Berdasarkan data yang dirilis Kementerian Lingkungan Hidup dan Kehutanan pada tahun 2019, luas kebakaran hutan dan lahan (Karhutla) yang terjadi di Indonesia mencapai 44.411, 36 Ha. 2.611.411,44 Ha (2015), 438.363,19 Ha (2016), 165.473,92 Ha (2017), 510.562,21 (2018), dan 328.722 Ha (2019). Hal tersebut menunjukkan bahwa kebakaran hutan di Indonesia masih terus terjadi setiap tahunnya [4]. Oleh karena itu, pendeteksian asap diperlukan agar lokasi titik api dapat ditemukan dengan lebih cepat. Sehingga dapat membantu mengurangi luas kebakaran hutan dan lahan di Indonesia.

Salah satu metode yang saat ini sedang dikembangkan untuk mendeteksi dan mengenali objek pada sebuah citra adalah *Convolutional Neural Network* (CNN). CNN adalah pengembangan dari *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis *Deep Neural Network* ka-

rena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra [5]. Oleh karena itu, dibuatlah sebuah sistem yang dapat mengklasifikasi citra asap dalam sebuah gambar yang diambil dari satelit dengan menggunakan *Convolutional Neural Network*.

1.2 Permasalahan

Citra asap pada gambar satelit memiliki bentuk dan warna yang mirip dengan fenomena lain seperti awan dan kabut. Oleh karena itu, diperlukan sebuah sistem yang dapat membedakan antara citra asap dan citra lainnya pada gambar yang diambil dari satelit.

1.3 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Membuat suatu sistem yang mampu mengklasifikasi citra asap pada gambar satelit menggunakan metode *Convolutional Neural Network* (CNN)
2. Melakukan analisis performa terhadap sistem yang dibuat untuk mendapatkan akurasi dari metode tersebut

1.4 Batasan Masalah

Untuk memfokuskan permasalahan yang diangkat maka dilakukan pembatasan masalah. Batasan-batasan masalah tersebut diantaranya adalah:

1. Dataset yang digunakan pada tugas akhir ini adalah USTC SmokeRS, terdiri dari 6.225 citra dengan ukuran (256 x 256) pixel.
2. Klasifikasi pada sistem ini terbatas pada enam kelas cita satelit, yaitu *Cloud, Dust, Haze, Land, Seaside, Smoke.*
3. *Training* model CNN pada sistem klasifikasi ini menggunakan *backend native-keras* serta *pretrained* model MobileNet v2, Inception V3, InceptionResNetV2, dan NASNet.

1.5 Sistematika Penulisan

Laporan penelitian tugas akhir ini tersusun dalam sistematika dan terstruktur sehingga mudah dipahami dan dipelajari oleh pembaca maupun seseorang yang ingin melanjutkan penelitian ini. Alur sistematika penulisan laporan penelitian ini yaitu :

1. BAB I Pendahuluan
   Bab ini berisi uraian tentang latar belakang permasalahan, penjelasan alasan pemilihan judul, sistematika laporan, tujuan dan batasan masalah pada peneltian ini.
2. BAB II Dasar Teori
   Bab ini berisi tentang uraian secara sistematis teori-teori yang berhubungan dengan permasalahan yang dibahas pada penelitian ini. Teori-teori ini digunakan sebagai dasar dalam penelitian, yaitu informasi terkait Penginderaan Jauh, *Machine Learning, Convolutional Neural Network* (CNN), dan teori-teori penunjang lainya..
3. BAB III Perancangan Sistem dan Impementasi
   Bab ini berisi tentang penjelasan-penjelasan terkait penelitian yang akan dilakukan, langkah-langkah pengolahan dataset, dan proses *training* model. Guna mendukung pengerjaan penelitian ini digunakanlah blok diagram atau *work flow* agar penjelasan tentang sistem yang akan dibuat dapat dipahami dengan mudah.
4. BAB IV Pengujian dan Analisis
   Bab ini menjelaskan tentang pengujian model CNN hasil *training*, pengujian performa model CNN dalam melakukan klasifikasi terhadap data tes, serta analisis hasil pengujian.
5. BAB V Penutup
   Bab ini merupakan penutup yang berisi kesimpulan yang diambil dari penelitian dan pengujian yang telah dilakukan. Saran dan kritik yang membangun untuk pengembangkan lebih lanjut juga dituliskan pada bab ini.

Halaman ini sengaja dikosongkan

BAB 2
TINJAUAN PUSTAKA

Demi mendukung penelitian ini, dibutuhkan beberapa teori penunjang sebagai bahan acuan dan refrensi. Dengan demikian penelitian ini menjadi lebih terarah.

2.1 Penginderaan Jauh

Penginderaan jauh adalah ilmu dan seni untuk mendapatkan informasi dari suatu objek, daerah, atau fenomena (geofisik) melalui analisis data, di mana dalam mendapatkan data ini tidak secara kontak langsung dengan objek, daerah, atau fenomena yang dikaji.

Objek penginderaan jauh adalah semua benda yang ada di permukaan bumi, seperti tanah, guning, air, vegetasi, dan hasil budidaya manusia, kota, lahan pertanian, hutan atau benda-benda yang ada di angkasa seperti awan [7]. Dengan semakin berkembangnya teknologi penginderaan jauh, tekstur detail dari suatu gambar bisa didapatkan [6].

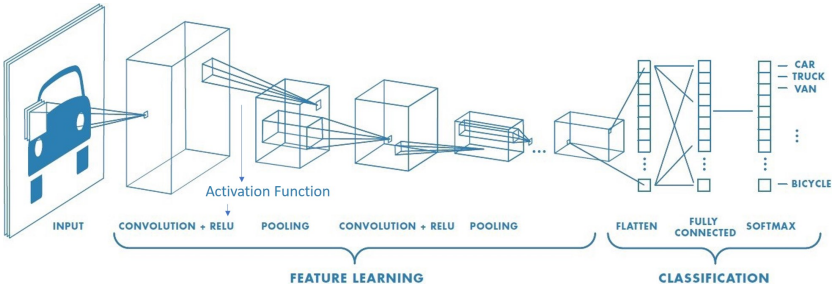
2.2 *Machine Learning*

Machine learning merupakan pendekatan dalam AI (*Artificial Intelligence*) yang banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah atau melakukan otomatisasi. Setidaknya ada dua aplikasi utama dalam *Machine learning* yaitu, klasifikasi dan prediksi. *Machine learning* membutuhkan data untuk dipelajari yang disebut sebagai data *training*. Klasifikasi adalah metode yang digunakan oleh mesin untuk memilah atau mengklasifikasikan obyek berdasarkan ciri tertentu sebagaimana manusia mencoba membedakan benda satu dengan yang lain. Sedangkan prediksi digunakan oleh mesin untuk menerka keluaran dari suatu data masukan berdasarkan data yang sudah dipelajari dalam *training* [8].

2.3 *Convolutional Neural Network* (CNN)

Convolutional Neural Network atau CNN adalah variasi dari Multilayer Perceptron yang terinspirasi dari jaringan syaraf manusia. CNN merupakan suatu layer yang memiliki susunan neuron

3D (lebar, tinggi, kedalaman). Lebar dan tinggi merupakan ukuran layer sedangkan kedalaman mengacu pada jumlah layer [9]. CNN dibagi menjadi dua tahapan besar yaitu *Feature Learning* dan *Classification* yang ditunjukan pada Gambar 2.1.

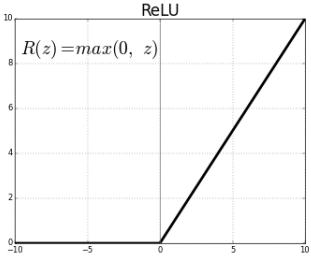


Gambar 2.1: *Convolutional Neural Network* Arsitektur[10]

2.3.1 *Activation Function*

Activation Function atau fungsi aktivasi merupakan sebuah fungsi yang menggambarkan hubungan antara tingkat aktivitas internal (*summation function*) yang mungkin berbentuk linear ataupun non-linear. Fungsi ini bertujuan untuk menentukan apakah *neuron* diaktifkan atau tidak. Berikut fungsi aktivasi yang digunakan pada model *Convolutional Neural Network* dalam penelitian ini:

1. *Rectified Linear Unit* (ReLU)
   Pada dasarnya fungsi ReLU melakukan *treshold* dari 0 hingga tidak terhingga (*infinity*). Fungsi ini menjadi salah satu fungsi aktivasi yang populer saat ini. Pada fungsi ini *input* dari neuron-neuron berupa bilangan negatif akan diterjemahkan kedalam nilai 0, dan jika masukan bernilai positif maka *output* dari neuron adalah nilai aktivasi itu sendiri. Fungsi aktivasi ini memiliki kelebihan yaitu dapat mempercepat proses konfigurasi yang dilakukan dengan *Stochastic Gradient Descent* (SGD). Kelemahan ReLU yaitu fungsi aktivasi ini bisa menjadi rapuh pada proses *training* dan bisa membuat unit tersbut mati. Grafik fungsi aktivasi ReLU ditunjukan pada Gambar 2.2 [10].



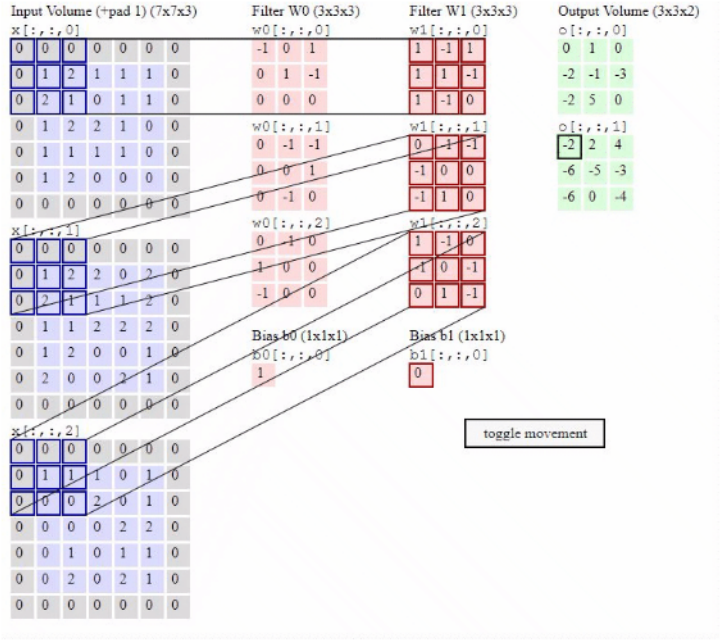
Gambar 2.2: Fungsi Aktivasi ReLU[10]

2. *Softmax*
   Dalam matematika, fungsi aktivasi *softmax* juga dikenal sebagai *softargmax* atau fungsi yang mendapatkan nilai *index* dari nilai maksimum (max) dalam sebuah tensor. *Softmax* adalah fungsi aktivasi yang mengambil input vektor dari bilangan *real* K , dan menormalkannya menjadi distribusi probabilitas yang terdiri dari probabilitas K. *Softmax* akan merubah beberapa komponen vektor negatif atau lebih besar dari satu dan mungkin tidak berjumlah 1 menjadi interval (0,1) dan komponen akan bertambah hingga 1. Sehingga hal ini dapat diartikan sebagai probabilitas. Komponen input yang lebih besar akan sesuai dengan probabilitas yang lebih besar. *Softmax* sering digunakan dalam jaringan saraf untuk memetakan output non-linear dari jaringan ke distribusi probabilitas lebih dari prediksi kelas *output*.

2.3.2 *Convolutional Layer*

Convolution layer merupakan bagian dari tahap pada arsitektur CNN. Tahap ini melakukan operasi konvolusi pada output dari layer sebelumnya. Layer tersebut adalah proses utama yang mendasari jaringan arsitektur CNN. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *Feature Map* dari input citra [11]. *Convolutional layer* biasanya dijadikan layer pertama pada *feature learning* arsitektur CNN. Pada Gambar 2.3 ukuran *convolutional layer* adalah 7x7x3. Panjang

7 *pixels*, tinggi 7 *pixels* dan tebal 3 *channel*.



Gambar 2.3: *Convolutional Layer*[10]

Ketiga filter ini akan digeser keseluruh bagian dari citra. Setiap pergeseran akan dilakukan operasi *dot product* antara *input* dan nilai dari filter tersebut sehingga menghasilkan sebuah *output* atau biasa disebut sebagai *activation map* atau *feature map*. Parameter yang menentukan berapa jumlah pergeseran filter disebut *"Stride"*. Jika nilai *stride* adalah 1, maka convolutioal filter akan bergeser sebanyak 1 *pixels* secara horizontal lalu vertical. Semakin kecil *stride* maka akan semakin detail informasi yang akan didapatkan dari sebuah *input*, namun membutuhkan proses komputasi yang lebih jika dibandingkan dengan *stride* yang besar. Perlu diperhatikan bahwa dengan menggunakan *stride* bernilai kecil tidak selalu akan mendapatkan performa yang bagus. Persamaan 2.1 digunakan untuk

menghitung dimensi dari *feature map*.

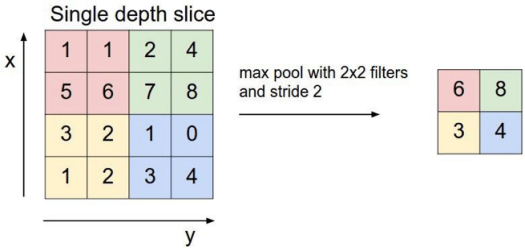
$$Output = \frac{W - N + 2P}{S} + 1 \tag{2.1}$$

Padding atau *Zero Padding* adalah parameter yang menentukan jumlah *pixels* (berisi nilai 0) yang akan ditambahkan di setiap sisi dari *input*. Dimensi *output* dari *convolutional layer* selalu lebih kecil dari *inputnya* (kecuali penggunaan 1x1 filter jumlah dengan *stride* 1). Output ini akan digunakan kembali sebagai input dari *convolutional layer* selanjutnya, sehingga informasi akan banyak yang terbuang. Dengan menggunakan *padding* dapat mengatur dimensi *output* agar tetap sama seperti dimensi *input* atau setidaknya tidak berkurang secara drastis, sehingga penggunaan *convolutional layer* yang lebih dalam dapat menghasilkan lebih banyak *features* yang di-*extract*[10]:

2.3.3 *Pooling Layer*

Pooling layer terdapat setelah *convolutional layer*. Terdiri dari sebuah filter dengan ukuran dan *stride* tertentu yang akan bergeser pada seluruh *activation map*. Fungsi dari pooling ini adalah untuk mereduksi input secara spasial (mengurangi jumlah parameter) dengan operasi *down-sampling*. Dalam *pooling layer* terdapat dua macam *pooling* yang biasa digunakan yaitu *average pooling* dan *maxpooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata, sedangkan pada *max-pooling* adalah nilai maksimal. Lapisan *pooling* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran volume *output* pada *feature map*, sehingga mengurangi jumlah parameter dan perhitungan di jaringan, untuk mengendalikan *overfitting*[11]. Berdasarkan Gambar 2.4 menunjukan proses dari *max-pooling*. *Output* dari proses *pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan dimensi citra awal. Lapisan *pooling* diatas akan beroperasi pada setiap irisan kedalaman volume input secara bergantian. Jika dilihat dari Gambar 2.4 operasi *max-pooling* menggunakan ukuran filter 2x2. Masukan pada proses tersebut berukuran 4x4, dari masing-masing 4 angka pada *input* operasi tersebut diambil nilai maksimalnya ke-

mudian dilanjutkan membuat ukuran output baru menjadi ukuran 2x2[11].



Gambar 2.4: *Pooling Layer*[10]

2.3.4 *Batch Normalization Layer*

Batch Normalization adalah sebuah solusi dari adanya masalah yang disebut internal covariate shift. Masalah tersebut terjadi karena pendistribusian input setiap layer akan mengalami perubahan selama proses training berlangsung yang disebabkan oleh perubahan parameter layer sebelumnya. Masalah tersebut akan memperlambat proses training karena harus merubah learning rates menjadi lebih rendah dan dibutuhkan inisialisasi parameter yang cermat. Dengan menggunakan batch normalization memungkinakan untuk meningkatkan learning rates menjadih lebih tiggi dan tidak membutuhkan insialisasi parameter yang cermat. Batch Normalization juga akan bertindak sebagai regulator dan dalam beberapa kasus akan menghilangkan kebutuhan penggunaan layer Dropout. Untuk meningkatkan stabilitas neural network, batch normalization akan menormalkan output dari lapisan aktivasi sebelumnya dengan mengurangi rata-rata batch dan membaginya dengan standar deviasi batch[12].

Setelah pergeseran skala output aktivasi dengan beberapa parameter yang diinisialisasi secara acak, weight pada lapisan berikutnya tidak lagi optimal. Optimzation Algorithm seperti SGD (Stochastic gradient descent) akan membatalkan normalisasi ini jika itu cara untuk meminimalkan fungsi kerugian (loss function). Akibatnya, Batch normalization akan menambahkan dua parameter yang dapat dilatihkan ke setiap layer, sehingga output yang dinor-

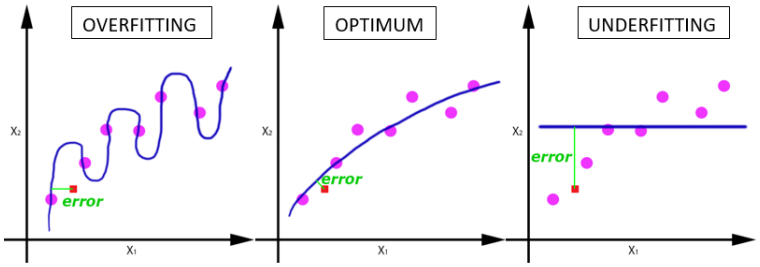
malisasi dikalikan dengan parameter standar deviasi (gamma) dan menambahkan parameter rata-rata (beta). Dengan kata lain, batch normalization memungkinkan optimization algorithm melakukan denormalisasi dengan mengubah hanya dua weight untuk setiap aktivasi, dan tidak kehilangan stabilitas jaringan dengan mengubah semua weight [12]. Persamaan 2.2 digunakan sebagai aktivasi pada *mini-batch*.

$$
\begin{aligned}
&\textbf{Input:}\text{Nilai x pada mini batch: } \beta = \{x_{1\ldots m}\};\\
&\text{(Parameter yang harus dipelajari: } \beta\gamma)\\
&\textbf{Output:}\{y_1 = BN_{\beta,\gamma}\,(x_1)\}\\
&\textit{mini-batch mean} = \mu\beta \leftarrow \tfrac{1}{m}\textstyle\sum_{i=1}^{m} x_i\\
&\textit{mini-batch variance} = \sigma_\beta^2 \leftarrow \tfrac{1}{m}\textstyle\sum_{i=1}^{m}(x_i - \gamma\beta)^2\\
&\textit{normalize} = \hat{x}_i \leftarrow \tfrac{x_i - \gamma\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}\\
&\textit{scale and shift} = y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}\,(x_i)
\end{aligned}
\tag{2.2}
$$

2.3.5 *Optimization Algorithm*

Terdapat dua masalah yang sering terjadi dalam proses *training*, yaitu *underfitting* dan *overfitting*. *Underfitting* terjadi karena tidak adanya perubahan nilai *loss validation* pada setiap epoch dalam proses *training*. *Overfitting* terjadi karena peningkatan nilai *loss validation* pada setiap epoch dalam proses *training*. *Optimization algorithm* bertujuan untuk meminimalkan terjadi *undrfitting* maupun *overfitting* dalam *Error function* $E(x)$ yang merupakan fungsi matematika yang bergantung pada parameter model internal learning yang digunakan dalam menghitung nilai target (Y) dari himpunan prediktor (X) yang digunakan dalam model. Terjadinya *underfitting* dan *overfitting* pada proses *training* model dapat dilihat pada Gambar 2.5.

Sebagai contoh adalah nilai *weight* (w) dan bias (b) dari *neural network* sebagai parameter internal *learning* yang digunakan



Gambar 2.5: Grafik terjadinya *Underfitting* dan *Overfitting*

dalam menghitung nilai *output* dan diperbarui sampai mendekati nilai optimal, yaitu dengan meminimalkan *Loss* ketika proses learning model *neural network*. Parameter internal model memainkan peran yang sangat penting dalam melakukan proses *learning* yang efisien dan efektif untuk mendapatkan hasil yang akurat. Penggunaan berbagai *optimization algorithm* untuk mengevaluasi nilai-nilai yang sesuai (optimal) dari model akan memengaruhi proses *learning* model dan output dari model tersebut. Dalam *optimization algorithm* terdapat dua kategori utama yakni :

1. *First Order Optimization Algorithms*
   Algoritma ini meminimalkan atau memaksimalkan *Loss function* $E(x)$ menggunakan nilai Gradien yang berhubungan dengan parameter. *First order algorithm* yang paling banyak digunakan adalah Gradient Descent. Algoritma ini memberikan informasi apakah nilai dari fungsi turunan tersebut menurun atau meningkat pada titik tertentu. Gradient adalah sebuah vektor yang merupakan generalisasi multi-variabel dari turunan (dy/dx) dari tingkat perubahan y dengan x. Nilai gradien dihitung menggunakan turunan parsial. Perbedaan utama antara Gradien dan turunannya adalah Gradien fungsi menghasilkan bidang vektor.
2. *Second Order Optimization Algorithms*
   Second order algorithm menggunanakan fungsi turunan orde kedua. Algoritma ini juga berfungsi untuk meminimalkan *loss function*. Second order algorithm berisikan turunan matriks

parsial orde kedua. Karena algoritma ini membutuhkan proses komputasi yang berat, *second order algorithm* jarang digunakan. Algoritma ini memberikan informasi apakah turunan pertama meningkat atau menurun yang mengisyaratkan kelengkungan dari fungsi. *Second order algorithm* memberikan sebuah gambaran permukaan kuadratik yang menyentuh permukaan kelengkungan *Error Surface*.

Gradient descent merupakan salah satu metode optimasi numerik iteratif untuk mencari titik yang meminimumkan suatu fungsi yang dapat diturunkan. Metode ini bekerja dengan memulai dari sebuah tebakan awal (boleh acak) dan secara iteratif tebakan ini diperbaiki berdasarkan suatu aturan yang melibatkan gradien/turunan pertama dari fungsi yang ingin diminimumkan. Secara formal, diberikan sebuah fungsi $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ untuk mencari nilai $x^* = argmin_x f(x)$ dengan $x \in \mathbb{R}^2$. Gradient descent mulai dengan suatu tebakan awal x_0, lalu tebakan ini diperbaiki dengan aturan seperti persamaan 2.3.

$$x_{t+1} = x_t - \eta \nabla f(x_t) \tag{2.3}$$

Dengan $\eta > 0$ biasa disebut sebagai *step size* atau *learning rate* dalam literatur *machine learning* dan $\nabla f(x_t)$ adalah turunan f di titik x_t. Dalam konteks *machine learning* fungsi f yang ingin diminimumkan adalah fungsi kerugian (*loss*) dari metode *machine learning* yang digunakan. Fungsi *loss* ini biasanya dihitung dari seluruh sampel yang ada pada data *learning*. Misalnya pada regresi linear, fungsi *loss* yang biasa digunakan adalah *Mean Squared Error* (MSE) dengan persamaan seperti 2.4:

$$L(0, D) = \sum_{i=1}^{M} (h(x^i; 0) - y^i)^2 \tag{2.4}$$

Stochastic Gradient Descent (SGD), *ADADELTA*, dan *Adams* adalah contoh turunan dari metode Gradient Descent. Ketiga metode tersebut memiliki tujuan yang sama, yaitu menemukan nilai optimal, namun memiliki cara yang berbeda dalam menentukan *le-*

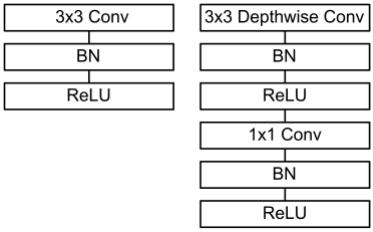
arning rate-nya. *Learning rate* adalah parameter yang menentukan kecepatan pembelajaran pada *neural network*. *Learning rate* yang besar berarti pembelajaran semakin cepat, namun beresiko terjadinya masalah *divergensi* (melenceng) yang melewati nilai minimum. *Learning rate* yang terlalu kecil memiliki kemungkinan besar untuk mencapai *konvergensi*, namun waktu yang dibutuhkan menjadi lebih lama[13].

2.4 *Transfer Learning*

Transfer learning adalah suatu teknik atau metode yang memanfaatkan model yang sudah dilatih terhadap suatu dataset untuk menyelesaikan permasalahan lain yang serupa dengan cara menggunakannya sebagai starting point, memodifikasi dan mengupdate parameternya sehingga sesuai dengan dataset yang baru[10]. Manfaat dari *Transfer Learning* adalah dapat mempercepat waktu yang dibutuhkan untuk mengembangkan dan melatih model dengan menggunakan kembali potongan-potongan modul dari model yang sudah dikembangkan. Model yang sudah dikembangkan sebelumnya dan dapat digunakan pada model lain disebut *Pre-Trained Models*. *Pre-Trained Models* telah dilatih menggunakan kumpulan dataset dengan jumlah yang begitu banyak. Pada tugas akhir ini menggunakan dua *Pre-Trained Models* berbeda, yaitu MobileNet v2, Inception V3, InceptionResNet V2, dan NASNet. Proses *training pre-trained* model tersebut menggunakan data citra satelit yang berasal dari *dataset* USTC SmokeRS.

2.4.1 *MobileNet*

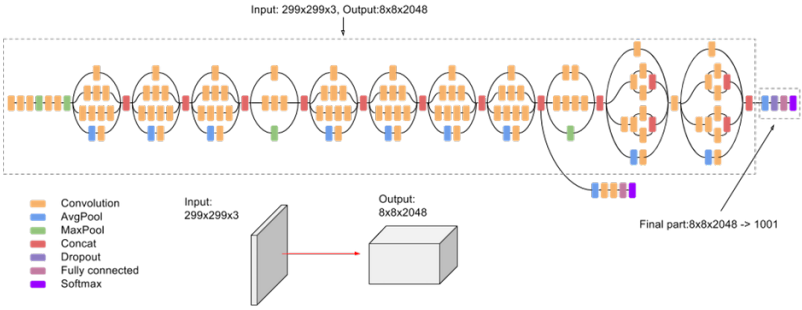
MobileNet merupakan salah satu arsitektur *Convolutional Neural Network* (CNN) yang dapat digunakan untuk mengatasi kebutuhan akan computing resource berlebih. MobileNet ditujukan untuk pengembangan *mobile application* dan *embedded vision applications*. Perbedaan mendasar antara arsitektur MobileNet dan arsitektur CNN pada umumnya adalah penggunaan *convolution layer* dengan ketebalan filter yang sesuai dengan ketebalan dari *input* citra. MobileNet membagi konvolusi menjadi dua, yaitu *depthwise convolution* dan *pointwise convolution* yang dicontohkan seperti Gambar 2.6[14].



Gambar 2.6: Arsitektur MobileNet[14]

2.4.2 *Inception V3*

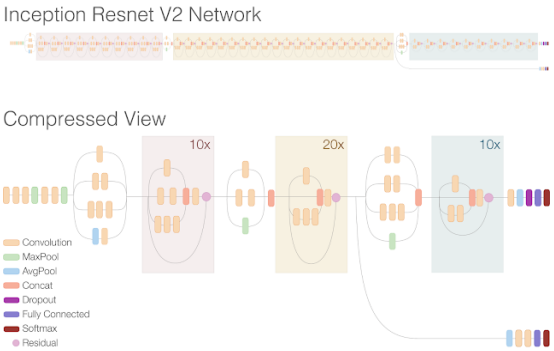
Inception V3 adalah model *image recognition* yang banyak digunakan dan telah terbukti mencapai akurasi lebih dari 78,1% pada dataset ImageNet. Model ini pertama kali diperkenalkan oleh GoogLeNet. Inception v3 terdiri dari blok simetris dan asimetris, yang berisikan *covolution layer*, *average pooling*, *max pooling*, *concat*, *dropout*, dan *fully connected layer*. *Batch normalization layer* digunakan secara luas di seluruh model dan diterapkan pada *activation inputs* dan *loss* dihitung menggunakan *Softmax*. Diagram model Inception v3 ditunjukkan pada Gambar 2.7[15].



Gambar 2.7: Arsitektur Model Inception V3[15]

2.4.3 *Inception ResNet V2*

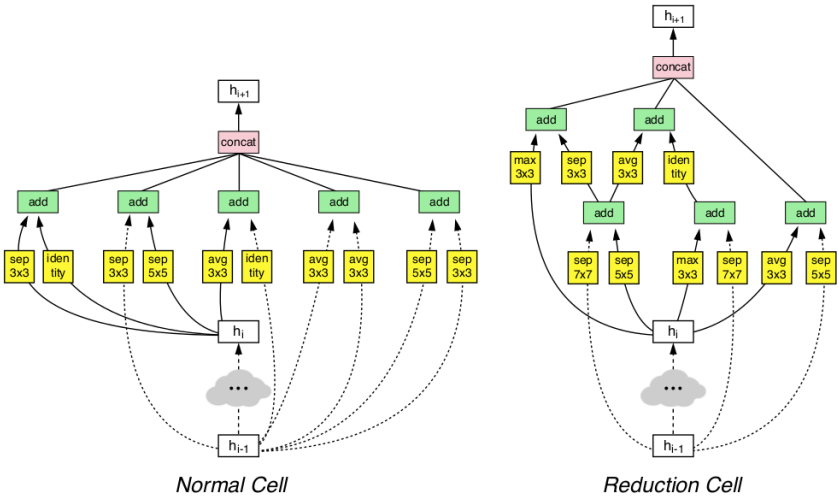
Inception ResNet V2 merupakan model *image recognition* yang telah terbukti mencapai akurasi 80,4 % pada dataset ImageNet. Pada model Inception ResNet V2, diperkenalkan *Reduction Blocks*. Versi-versi sebelumnya tidak memiliki *Reduction Blocks* secara eksplisit, tetapi fungsionalitas dari *Reduction Blocks* tersebut juga diterapkan. Diagram model Inception ResNet V2 ditunjukkan pada Gambar 2.8. [16].



Gambar 2.8: Arsitektur Model Inception ResNet V2[16]

2.4.4 *NASNet*

NASNet merupakan model *image recognition* yang telah terbukti mencapai akurasi 82,7 % pada dataset ImageNet. Arsitektur dari NASNet terdiri dari 2 type layer, yaitu *Normal Layer*, dan *Reduction Layer*. Kedua layer tersebut didesain oleh *AutoML*. Arsitektur dari NASNet ditunjukkan pada Gambar 2.9. [17].



Gambar 2.9: Arsitektur Model NASNet[17]

2.5 *Confusion Matrix*

Untuk mendapatkan nilai *precision* dan *recall* dapat dilakukan dengan memanfaatkan *Confusion Matrix* seperti pada Gambar 2.10. *Confusion Matrix* menyatakan jumlah data uji yang benar dan jumlah data uji yang salah saat dilakukan proses pendeteksian. Keberadaan confousion matrix ini dapat digunakan untuk membantu dalam menghitung nilai-nilai precision dan recall untuk menghitung besarnya nilai rata-rata harmonik nilai precision dan nilai recall tersebut (f-measure). Confusion matrix juga berguna untuk menghitung nilai accuracy yang pada umumnya digunakan untuk mengukur atau menguji preformasi sebuah teknik atau metode.

	PREDIKSI POSITIF	PREDIKSI NEGATIF
KONDISI BENAR	BENAR POSITIF *(TRUE POSITIVE)*	SALAH NEGATIF *(FALSE NEGATIVE)*
KONDISI SALAH	SALAH POSITIF (*FALSE POSITIVE*)	BENAR NEGATIF *(TRUE NEGATIVE)*

Gambar 2.10: *Confusion Matrix*

1. *Precision*
   Precision dapat dinyatakan sebagai data yang bernilai tepat antara prediksi dengan kondisi aktualnya dibagi dengan total data yang prediksinya bernilai tepat. Atau seperti yang dituliskan pada persamaan 2.5.

$$precision = \frac{TP}{TP + FP} \tag{2.5}$$

2. *Recall*
   Recall dapat dinyatakan sebagai pembagian antara data yang bernilai tepat antara prediksi dan kondisi aktualnya dengan total data yang kondisi aktualnya bernilai tepat. Dapat dinyatakan sebagai persamaan 2.6.

$$recall = \frac{TP}{TP + FN} \tag{2.6}$$

3. *F-measure*
   F-measure atau yang biasa juga disebut dengan F1 score memiliki pengertian sebagai nilai rata-rata harmonik dari nilai precision dan nilai recall. F-measure dapat dinyatakan sebagai persamaan 2.7.

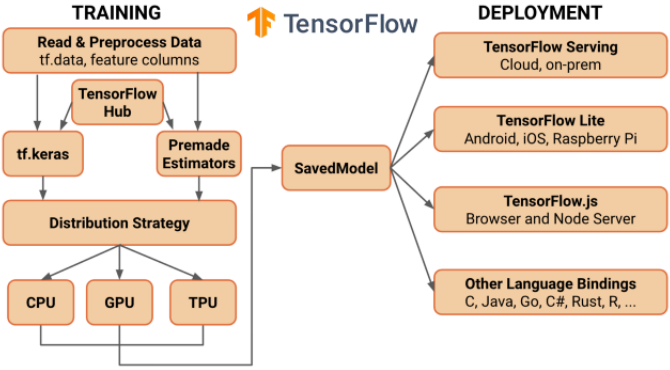
$$recall = \frac{2 \times precision \times recall}{precision + recall} \tag{2.7}$$

4. *Accuracy*
   Accuracy merupakan besarnya seluruh data hasil prediksi yang sesuai dengan kondisi sebenarnya terhadap seluruh data yang ada. Accuracy dapat dinyatakan sebagai persamaan 2.8.

$$accuracy = \frac{TP + TN}{total\ population} \tag{2.8}$$

2.6 *Tensorflow*

TensorFlow merupakan sebuah *Application Programming Interface* (API) untuk implementasi *machine learning algorithm* dan mengeksekusi *machine learning algorithm* tersebut. Komputasi yang dijalankan menggunakan *TensorFlow* dapat dilakukan melalui beragaam platform tanpa perlu merubah sistem heterogen, mulai dari perangkat seluler seperti ponsel, tablet hingga sistem terdistribusi berskala besar dari ratusan mesin dan ribuan perangkat komputasi seperti kartu GPU . Sistem ini fleksibel dan dapat digunakan untuk mengekspresikan berbagai macam algoritma, termasuk *training* dan algoritma inferensi untuk model *Deep Neural Network*. *Tensorflow* telah digunakan untuk melakukan penelitian dan pengembangan sistem *machine learning* ke ranah area ilmu komputer dan bidang lainnya, termasuk *speech recognition*, *computer vision*, robotika, pengambilan informasi, *natural language processing*, ekstraksi informasi geografis, dan *computational drug discovery*.Tensorflow API dikembangkan oleh Google dan dirilis sebagai paket *open-source* di bawah lisensi Apache 2.0 pada November 2015 dan tersedia di *www.tensorflow.org*. Arsitektur *TensorFlow* 2.0 menggunakan diagram konseptual yang disederhanakan seperti yang ditunjukkan pada Gambar 2.11[18].

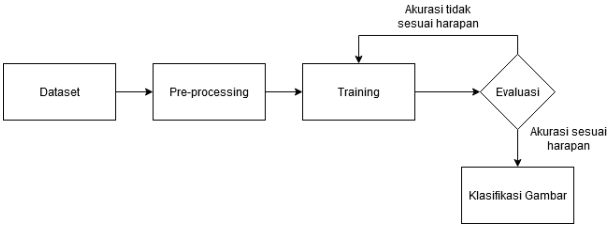


Gambar 2.11: *Arsitektur TensorFlow 2.0* [18]

2.6.1 Keras

Keras adalah *open source neural network library* yang ditulis menggunakan Python. Keras dapat berjalan pada backend TensorFlow, Microsoft Cognitive Toolkit (CNTK) dan Theano. Keras dirancang untuk memungkinkan eksperimen cepat menggunakan *deep neural network*. Keras dikembangkan sebagai bagian dari penelitian proyek *Open-ended Neuro-Electronic Intelligent Robot Operating System* (ONEIROS), pengelola utamanya adalah François Chollet, seorang insinyur Google. Keras berisi banyak implementasi dari bagian-bagian *neural network* yang biasa digunakan seperti *layer*, *objectives*, *activation function*, *optimizers*, dan sejumlah *tool* untuk memudahkan pengolahan data citra dan teks. Selain *neural network* standar, keras memiliki *tools* untuk pengembangan *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RCN) seperti *convolution layer*, *dropout*, *batch normalization*, dan *pooling*.

BAB 3
DESAIN DAN IMPLEMENTASI SISTEM



Gambar 3.1: Diagram alir sistem

Tujuan dari tugas akhir ini adalah untuk mengklasifikasi citra asap pada gambar yang diambil oleh satelit. Proses kerja dari sistem ini ditunjukkan pada gambar 3.1. Agar sistem mampu mencapai hal tersebut, maka dibutuhkan dataset. Setelah mendapatkan dataset, langkah selanjutnya adalah memisahkan dataset akan yang digunakan untuk *training* dan dataset yang akan digunakan untuk *testing* atau validasi. Metode yang digunakan pada sistem ini adalah CNN yang menggunakan 4 *pre-trained* model yang berbeda - beda. Hasil dari *training* tersebut adalah model yang bisa digunakan untuk mendeteksi *class* pada sebuah gambar. Proses selanjutnya adalah identifikasi, yaitu proses dimana sistem mengidentifikasi *class* yang terdeteksi di dalam sebuah gambar sesuai dengan *class* dari dataset yang digunakan.

3.1 Desain Sistem

Secara garis besar, terdapat empat proses yang ada pada bagian ini, yaitu :

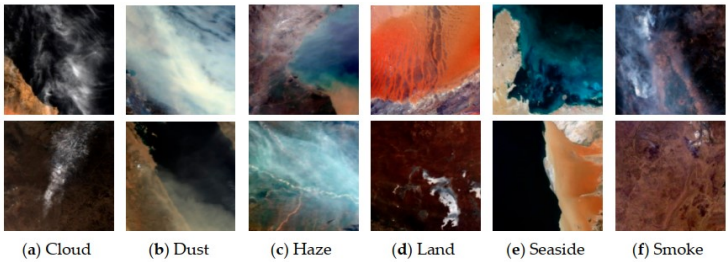
1. Dataset
   Dataset yang digunakan pada penelitian ini adalah dataset USTC SmokeRS yang berisi 6225 gambar.
2. *Pre-processing* Data
   Pre-processing data dilakukan sebelum proses *training* model, yaitu membagi dataset menjadi dua jenis data (*Split*), yaitu

data *training*, dan data validasi atau data *testing*.

3. *Training*
   Training merupakan sebuah proses pelatihan model CNN menggunakan dataset USTC SmokeRS. *Pre-trained* model yang digunakan dalam proses training ini adalah *MobileNet V2*, *Inception V3*, *InceptionResNet V2*, dan *NASNet*.
4. Evaluasi Model
   Model yang telah melalui proses *training* kemudian akan dievaluasi untuk menilai performanya. Evaluasi model akan melakukan evaluasi performa terhadap model hasil *training*.

3.2 Dataset

Dataset yang digunakan pada penelitian ini adalah Dataset USTC SmokeRS. Dataset USTC SmokeRS merupakan kumpulan gambar satelit dari enam *class* yang berbeda. Dataset tersebut didapatkan dari penelitian sebelumnya [19]. Gambar 3.2 menunjukan sample dataset pada enam *class* gambar satelit.



Gambar 3.2: Sample Dataset USTC SmokeRS[19]

Dataset USTC SmokeRS berisi 6225 gambar satelit yang terbagi menjadi enam *class*. Tabel 3.1 menunjukkan jumlah masing-masing *class* yang ada pada dataset USTC SmokeRS.

Tabel 3.1: Dataset USTC SmokeRS

Nama *Class*	**Jumlah Gambar**
Cloud	1164
Dust	1009
Haze	1002
Land	1027
Seaside	1007
Smoke	1016
Total	**6225**

3.3 *Pre-processing* Data

Semua data yang ada di dataset USTC SmokeRS akan dilakukan tahap *prepocessing* data terlebih dahulu. Proses ini bertujuan untuk menyetarakan seluruh format data yang berbeda-beda menjadi satu format yang sama agar dapat dilakukan pengolahan data secara digital[20].

3.3.1 Data *Split*

Sebelum melakukan proses *training* model, dataset terlebih dahulu dibagi menjadi dua bagian, yaitu data *train* dan data validasi yang sekaligus akan menjadi data tes. Proses pembagian ini disebut Data *Split*. Tabel 3.2 menunjukkan pembagian masing-masing *class* gambar pada dataset. Proses *Data Split* pada penelitian ini dijalankan pada *environment Google Colab*.

Tabel 3.2: Pembagian *class* gambar pada dataset USTC SmokeRS

***Class* Data**	***Data Training***	***Data Validation***
Cloud	932	232
Dust	808	201
Haze	802	200
Land	821	206
Seaside	806	201
Smoke	812	204
Total	**4981**	**1244**

3.4 *Training* Model

Training model dilakukan untuk melatih ulang *(re-train) pretrained model* CNN. *Pretrained model* yang digunakan dalam penelitian ini adalah *MobileNet V2*, *Inception V3*, *InceptionResNetV2*, dan *NASNet*. Proses *training model* dilakukan pada *environment Google Colab*. Pada proses *training* model ini digunakan data *train* yang telah dilakukan proses *pre-processing*. Jumlah data awal akan digunakan untuk menentukan ukuran *train steps*. Pada proses *training model* perlu dilakukan penentuan ukuran *batch size, train steps* dan *epoch* yang akan dijelaskan sebagai berikut:

1. *Batch Size*
   Penentuan *batch size* berdasarkan jumlah sampel data yang ingin disebarkan ke seluruh *neural network* dalam setiap iterasi *(train steps)*. Pada proses training kali ini ukuran batch size ditentukan sejumlah 32 untuk *MobileNet V2, Inception V3* dan , *InceptionResNet V2*. Sedangkan ukuran batch size untuk *NASNet* adalah 4, yang berarti setiap satu *step* akan disebarkan 32 atau 4 data training ke *neural network*.

2. *Train Steps*
   Jumlah sampel data dalam persamaan tersebut adalah jumlah data yang digunakan untuk proses *training*. Dari operasi persamaan tersebut dihasilkan ukuran *train steps* pada proses *training* model *MobileNet V2, Inception V3* dan , *InceptionResNet V2* adalah 156 kemudian untuk model *NASNet*

adalah 1245. Ukuran *train steps* tersebut ditentukan dari persamaan 3.1 dan 3.2.

$$\begin{aligned} Train\_Step &= \frac{Jumlah\_Sampel\_Data}{Batch\_Size} \\ Train\_Step &= \frac{4981}{32} = 156 \end{aligned} \tag{3.1}$$

$$\begin{aligned} Train\_Step &= \frac{Jumlah\_Sampel\_Data}{Batch\_Size} \\ Train\_Step &= \frac{4981}{4} = 1245 \end{aligned} \tag{3.2}$$

3. *Epoch*
   Epoch merupakan satu set putaran *training steps*. Beberapa *epoch* diperlukan untuk pelatihan sebuah *neural network* sehingga didapatkan kesalahan (*loss*) mendekati nol. *Epoch* pada proses *training model* ini didefinisikan sejumlah 30 kali untuk model *MobileNet V2, dan Inception V3*. Sedangkan untuk model *InceptionResNet V2, NASNet* didefinisikan sejumlah 20 kali.

3.4.1 *Load Pretrained Model*

Setelah dilakukan penentuan ukuran *batch size, train steps* dan *epoch* tahap selanjutnya dalam proses *training* adalah melakukan proses *load pretrained model*. *Pretrained model* yang digunakan pada penelitian ini adalah *MobileNet V2*, *Inception V3*, *InceptionResNetV2*, dan *NASNet*. Proses *load pretrained model* ini dapat dilakukan dengan cara memanggil *class* yang ada pada Keras API. *Class* tersebut akan memuat semua *layer* yang terdapat pada arsitektur *MobileNet V2*, *Inception V3*, *InceptionResNetV2*, dan *NASNet*. Berikut code untuk memuat semua layer *MobileNet V2*, *Inception V3*, *InceptionResNetV2*, dan *NASNet*.

```
MobileNetV2 = keras.applications.mobilenet_v2.MobileNetV2()

InceptionV3 = keras.applications.inception_v3.InceptionV3()

InceptionResNetV2 =  keras.applications.inception_resnet_v2.
    InceptionResNetV2()

NASNet = keras.applications.nasnet.NASNetLarge()
```

3.4.2 *Optimization and Callbacks*

Untuk mencegah terjadi *overfitting* dan *underfitting* diperlukan sebuah algoritma yang bertugas untuk melakukan optimalisasi saat proses *training* berlangsung. Adam merupakan algoritma pengoptimalan yang dapat dipilih sebagai metode turunan dari *stochastic gradient descent* klasik untuk memperbarui *weight* jaringan yang berdasarkan data pelatihan secara berulang. Menurut penelitian Diederik dan Jimmy [21], adam merupakan sebuah algoritma pengoptimalan yang memiliki kinerja lebih baik dalam menurunkan *loss* daripada algoritma pengoptimalan lain.

Berikut penerapan algoritma pengoptimalan Adam pada proses *training* model *MobileNet V2*, *Inception V3*, *InceptionResNetV2*, dan *NASNet*. *Source code* dibawah merupakan penerapan algoritma Adam pada MobileNet V2, *Inception V3*, *InceptionResNetV2*, dan *NASNet*.

```
--Penerapan Algoritma Adam--
optimizer = Adam (lr=0.0005, beta_1=0.9, beta_2=0.999,
    epsilon=1e-08, decay=5e-7, amsgrad=False)
  model.compile(optimizer = optimizer , loss = "
    categorical_crossentropy", metrics=["accuracy"])
```

Callbacks pada Keras API merupakan sebuah fungsi yang dapat dapat membantu memperbaiki kesalahan pendefinisian *learning rate* jika terjadi *overfitting* dengan menerapkan penghentian dini atau menyesuaikan tingkat *learning rate* pada setiap iterasi. Fungsi ini juga dapat melakukan visualisai saat proses *training* model berjalan. Fungsi *callback* yang digunakan pada proses *training* model ini adalah *ReduceLROnPlateau*. Penerapan fungsi *callback Reduce-*

LROnPlateau dapat dilihat pada *source code* berikut.

```
--Penerapan ReduceLRonPlateau--
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.1,
     patience=10, verbose=0, mode='auto', min_delta=0.0001,
    cooldown=0, min_lr=0)
```

Pada model *MobileNet V2, Inception V3, Inception ResNet V2, dan NASNet ReduceLROnPlateau* bertugas untuk memonitor *metric val_loss* dengan tingkat kesabaran 10 epoch dan *learning rate* minimal 0. Konfigurasi pada Inception V3 tersebut memilki arti jika nilai akurasi pada *val_loss* terus mengalami peningkatan selama 10 epoch beruntun maka learning rate akan diturunkan.

3.5 Evaluasi Model

Setelah proses *training* model selesai, proses selanjutnya adalah melakukan evaluasi terhadap model hasil *training* tersebut. Evaluasi ini bertujuan untuk melakukan pengecekan terhadap proses *training* model yang telah dilakukan. Untuk meengevaluasi model dan memvalidasinya digunakan metode *cross validation*. Metode ini merupakan metode statistik untuk mengevaluasi dan membandingkan algoritma *learning*. Dalam *cross validation*, data *train* dan data validasi disilangkan dalam putaran berturut-turut sehingga setiap *node* data memiliki peluang untuk divalidasi. Dalam setiap satu iterasi akan menggunakan lipatan pertama (k-1) data untuk mempelajari satu model, dan selanjutnya model yang dipelajari diminta untuk membuat prediksi tentang data tersebut. Performa setiap algoritma *learning* pada setiap lipatan akan dapat diketahui akurasinya menggunakan *confusion matrix*. Jika model tersebut memiliki tingkat akurasi rendah maka akan dilakukan konfigurasi ulang paramater yang digunakan dan kemudian akan dilakukan *training* ulang.

3.5.1 Evaluasi Model *MobileNet V2*

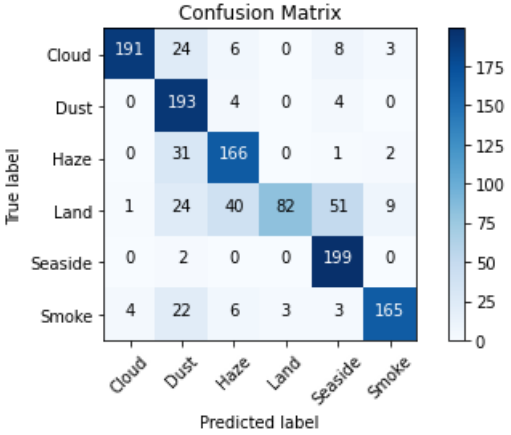
Evaluasi model *MobileNet V2* hasil dari proses training dilakukan untuk mendapatkan performa model dalam melakukan klasifikasi gambar satelit. Tabel 3.3 menunjukkan hasil 5 kali evaluasi setelah dilakukan 5 kali proses *training* model *MobileNet V2*.

Tabel 3.3: Hasil evaluasi model *MobileNet V2*

No	Akurasi
1	80,0643%
2	76,1254%
3	83,3601%
4	79,1800%
5	73,5530%
Rata-rata	78,4566%

3.5.1.1 Evaluasi Model *MobileNet V2* Pertama

Evaluasi pertama hasil proses *training* pada model *MobileNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 191 dari 232 gambar. 193 dari 201 gambar dengan *class Dust*. 166 dari 200 gambar dengan *class Haze*. 82 dari 207 gambar dengan *class Land*. 199 dari 201 gambar dengan *class Seaside*. Dan 165 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi pertama hasil proses *training* pada model *MobileNet V2* memiliki akurasi sebesar 80,0643%. Gambar 3.3 menunjukkan *confusion matrix* hasil dari *training* pertama pada model *MobileNet V2*.



Gambar 3.3: Confusion matrix training pertama MobileNet V2

Tabel 3.4 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi pertama pada model *MobileNet V2*.

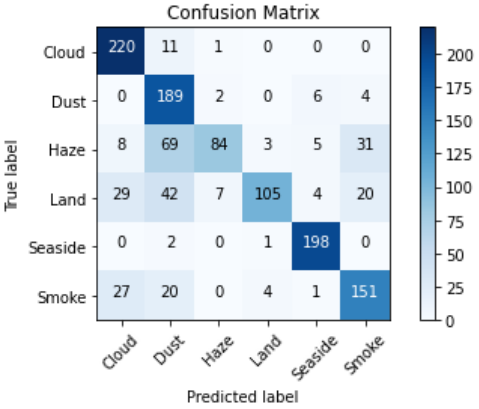
Tabel 3.4: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi pertama MobileNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.97	0.82	0.89	232
Dust	0.65	0.96	0.78	201
Haze	0.75	0.83	0.79	200
Land	0.96	0.40	0.56	207
Seaside	0.75	0.99	0.85	201
Smoke	0.92	0.81	0.86	203
Macro Avg	0.83	0.80	0.79	1244
Weighted Avg	0.84	0.80	0.79	1244

3.5.1.2 Evaluasi Model *MobileNet V2* Kedua

Evaluasi kedua hasil proses *training* pada model *MobileNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 220 dari

232 gambar. 189 dari 201 gambar dengan *class Dust*. 84 dari 200 gambar dengan *class Haze*. 105 dari 207 gambar dengan *class Land*. 198 dari 201 gambar dengan *class Seaside*. Dan 151 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kedua hasil proses *training* pada model *MobileNet V2* memiliki akurasi sebesar 76,1254%. Gambar 3.4 menunjukkan *confusion matrix* hasil dari *training* kedua pada model *MobileNet V2*.



Gambar 3.4: Confusion matrix training kedua MobileNet V2

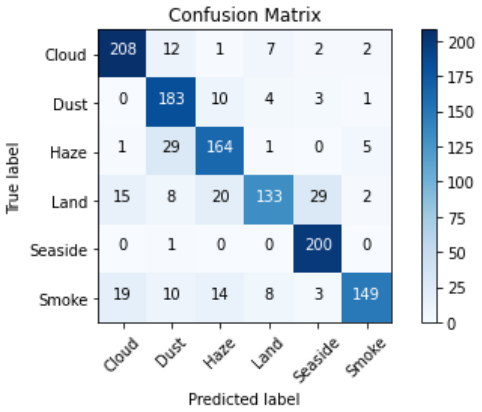
Tabel 3.5 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kedua pada model *MobileNet V2*.

Tabel 3.5: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kedua MobileNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.77	0.95	0.85	232
Dust	0.57	0.94	0.71	201
Haze	0.89	0.42	0.57	200
Land	0.93	0.51	0.66	207
Seaside	0.93	0.99	0.95	201
Smoke	0.73	0.74	0.74	203
Macro Avg	0.80	0.76	0.75	1244
Weighted Avg	0.80	0.76	0.75	1244

3.5.1.3 Evaluasi Model *MobileNet V2* Ketiga

Evaluasi ketiga hasil proses *training* pada model *MobileNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 208 dari 232 gambar. 183 dari 201 gambar dengan *class Dust*. 164 dari 200 gambar dengan *class Haze*. 133 dari 207 gambar dengan *class Land*. 200 dari 201 gambar dengan *class Seaside*. Dan 149 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi ketiga hasil proses *training* pada model *MobileNet V2* memiliki akurasi sebesar 83,3601%. Gambar 3.5 menunjukkan *confusion matrix* hasil dari *training* ketiga pada model *MobileNet V2*.



Gambar 3.5: Confusion matrix training ketiga MobileNet V2

Tabel 3.6 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi ketiga pada model *MobileNet V2*.

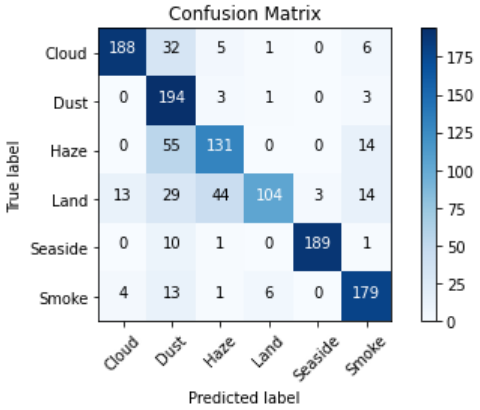
Tabel 3.6: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi ketiga MobileNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.86	0.90	0.88	232
Dust	0.75	0.91	0.82	201
Haze	0.78	0.82	0.80	200
Land	0.87	0.64	0.74	207
Seaside	0.84	1.00	0.91	201
Smoke	0.94	0.73	0.82	203
Macro Avg	0.84	0.83	0.83	1244
Weighted Avg	0.84	0.83	0.83	1244

3.5.1.4 Evaluasi Model *MobileNet V2* Keempat

Evaluasi keempat hasil proses *training* pada model *MobileNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 188 dari 232 gambar. 194 dari 201 gambar dengan *class Dust*. 131 dari 200 gambar dengan *class Haze*. 104 dari 207 gambar dengan *class*

Land. 189 dari 201 gambar dengan *class Seaside*. Dan 179 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi keempat hasil proses *training* pada model *MobileNet V2* memiliki akurasi sebesar 79,1800%. Gambar 3.6 menunjukkan *confusion matrix* hasil dari *training* keempat pada model *MobileNet V2*.



Gambar 3.6: Confusion matrix training keempat MobileNet V2

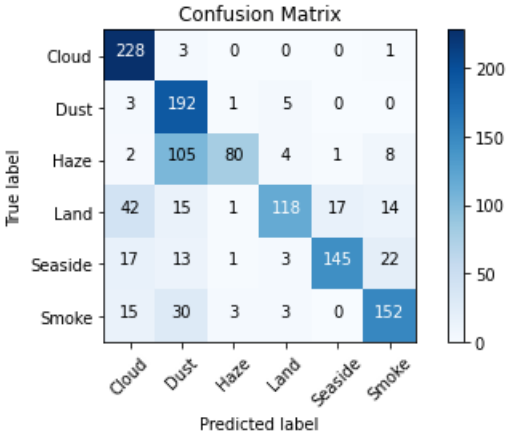
Tabel 3.7 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi keempat pada model *MobileNet V2*.

Tabel 3.7: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi keempat MobileNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.92	0.81	0.86	232
Dust	0.58	0.97	0.73	201
Haze	0.71	0.66	0.68	200
Land	0.93	0.50	0.65	207
Seaside	0.98	0.94	0.96	201
Smoke	0.82	0.88	0.85	203
Macro Avg	0.82	0.79	0.79	1244
Weighted Avg	0.83	0.79	0.79	1244

3.5.1.5 Evaluasi Model *MobileNet V2* Kelima

Evaluasi kelima hasil proses *training* pada model *MobileNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 228 dari 232 gambar. 192 dari 201 gambar dengan *class Dust*. 80 dari 200 gambar dengan *class Haze*. 118 dari 207 gambar dengan *class Land*. 145 dari 201 gambar dengan *class Seaside*. Dan 152 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kelima hasil proses *training* pada model *MobileNet V2* memiliki akurasi sebesar 73,5530%. Gambar 3.7 menunjukkan *confusion matrix* hasil dari *training* kelima pada model *MobileNet V2*.



Gambar 3.7: Confusion matrix training kelima MobileNet V2

Tabel 3.8 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kelima pada model *MobileNet V2*.

Tabel 3.8: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kelima MobileNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.74	0.98	0.85	232
Dust	0.54	0.96	0.69	201
Haze	0.93	0.40	0.56	200
Land	0.89	0.57	0.69	207
Seaside	0.89	0.72	0.80	201
Smoke	0.77	0.75	0.76	203
Macro Avg	0.79	0.73	0.72	1244
Weighted Avg	0.79	0.74	0.73	1244

3.5.2 Evaluasi Model *Inception V3*

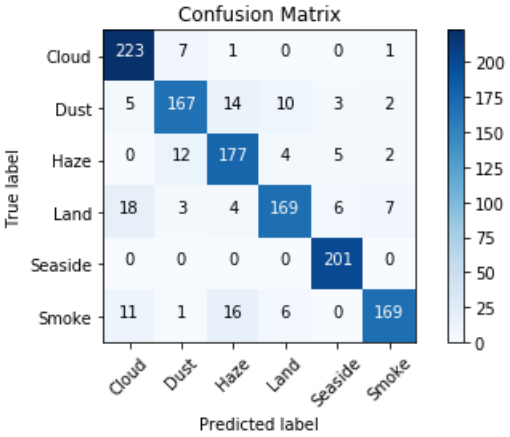
Evaluasi model *Inception V3* hasil dari proses training dilakukan untuk mendapatkan performa model dalam melakukan klasifikasi gambar satelit. Tabel 3.9 menunjukkan hasil 5 kali evaluasi setelah dilakukan 5 kali proses *training* model *Inception V3*.

Tabel 3.9: Hasil evaluasi model *Inception V3*

No	**Akurasi**
1	88.9067%
2	91.4790%
3	92,5241%
4	84,1639%
5	92,7652%
Rata-rata	89,9678%

3.5.2.1 Evaluasi Model *Inception V3* Pertama

Evaluasi pertama hasil proses *training* pada model *Inception V3* berhasil memprediksi gambar dengan *class Cloud* sebanyak 223 dari 232 gambar. 167 dari 201 gambar dengan *class Dust*. 177 dari 200 gambar dengan *class Haze*. 169 dari 207 gambar dengan *class Land*. 201 dari 201 gambar dengan *class Seaside*. Dan 169 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi pertama hasil proses *training* pada model *Inception V3* memiliki akurasi sebesar 88,9067%. Gambar 3.8 menunjukkan *confusion matrix* hasil dari *training* pertama pada model *Inception V3*.



Gambar 3.8: Confusion matrix training pertama Inception V3

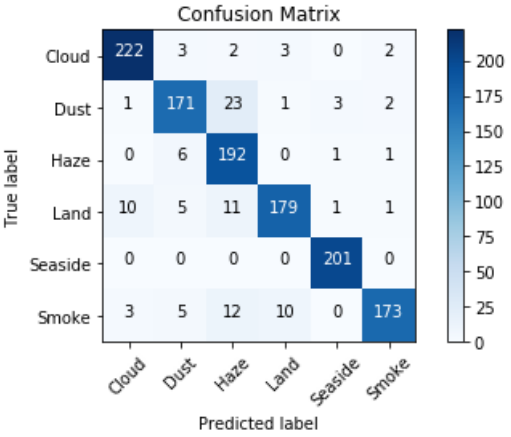
Tabel 3.10 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi pertama pada model *Inception V3*.

Tabel 3.10: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi pertama Inception V3

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.87	0.96	0.91	232
Dust	0.88	0.83	0.85	201
Haze	0.83	0.89	0.86	200
Land	0.89	0.82	0.85	207
Seaside	0.93	1.00	0.97	201
Smoke	0.93	0.83	0.88	203
Macro Avg	0.89	0.89	0.89	1244
Weighted Avg	0.89	0.89	0.89	1244

3.5.2.2 Evaluasi Model *Inception V3* Kedua

Evaluasi kedua hasil proses *training* pada model *Inception V3* berhasil memprediksi gambar dengan *class Cloud* sebanyak 222 dari 232 gambar. 171 dari 201 gambar dengan *class Dust*. 192 dari 200 gambar dengan *class Haze*. 179 dari 207 gambar dengan *class Land*. 201 dari 201 gambar dengan *class Seaside*. Dan 173 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kedua hasil proses *training* pada model *Inception V3* memiliki akurasi sebesar 91,4790%. Gambar 3.9 menunjukkan *confusion matrix* hasil dari *training* kedua pada model *Inception V3*.



Gambar 3.9: Confusion matrix training kedua Inception V3

Tabel 3.11 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kedua pada model *Inception V3.*

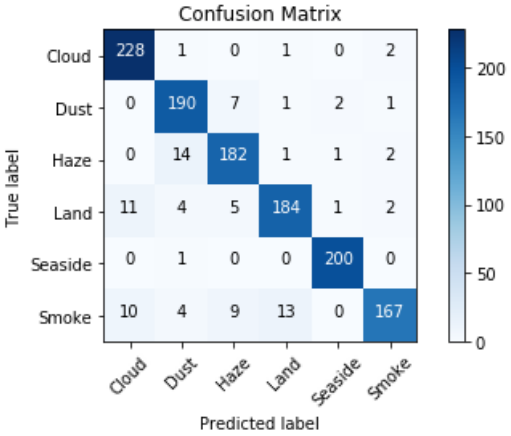
Tabel 3.11: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kedua Inception V3

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.94	0.96	0.95	232
Dust	0.90	0.85	0.87	201
Haze	0.80	0.96	0.87	200
Land	0.93	0.86	0.89	207
Seaside	0.98	1.00	0.99	201
Smoke	0.97	0.85	0.91	203
Macro Avg	0.92	0.91	0.91	1244
Weighted Avg	0.92	0.91	0.91	1244

3.5.2.3 Evaluasi Model *Inception V3* Ketiga

Evaluasi ketiga hasil proses *training* pada model *Inception V3* berhasil memprediksi gambar dengan *class Cloud* sebanyak 228 dari

232 gambar. 190 dari 201 gambar dengan *class Dust*. 182 dari 200 gambar dengan *class Haze*. 184 dari 207 gambar dengan *class Land*. 200 dari 201 gambar dengan *class Seaside*. Dan 167 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi ketiga hasil proses *training* pada model *Inception V3* memiliki akurasi sebesar 92,5241%. Gambar 3.10 menunjukkan *confusion matrix* hasil dari *training* ketiga pada model *Inception V3*.



Gambar 3.10: Confusion matrix training ketiga Inception V3

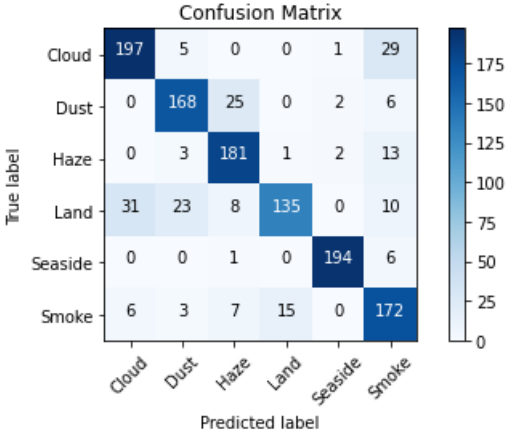
Tabel 3.12 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi ketiga pada model *Inception V3*.

Tabel 3.12: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi ketiga Inception V3

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.92	0.98	0.95	232
Dust	0.89	0.95	0.92	201
Haze	0.90	0.91	0.90	200
Land	0.92	0.89	0.90	207
Seaside	0.98	1.00	0.99	201
Smoke	0.96	0.82	0.89	203
Macro Avg	0.93	0.92	0.92	1244
Weighted Avg	0.93	0.93	0.92	1244

3.5.2.4 Evaluasi Model *Inception V3* Keempat

Evaluasi keempat hasil proses *training* pada model *Inception V3* berhasil memprediksi gambar dengan *class Cloud* sebanyak 197 dari 232 gambar. 168 dari 201 gambar dengan *class Dust*. 181 dari 200 gambar dengan *class Haze*. 135 dari 207 gambar dengan *class Land*. 194 dari 201 gambar dengan *class Seaside*. Dan 172 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi keempat hasil proses *training* pada model *Inception V3* memiliki akurasi sebesar 84,1639%. Gambar 3.11 menunjukkan *confusion matrix* hasil dari *training* keempat pada model *Inception V3*.



Gambar 3.11: Confusion matrix training keempat Inception V3

Tabel 3.13 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi keempat pada model *Inception V3*.

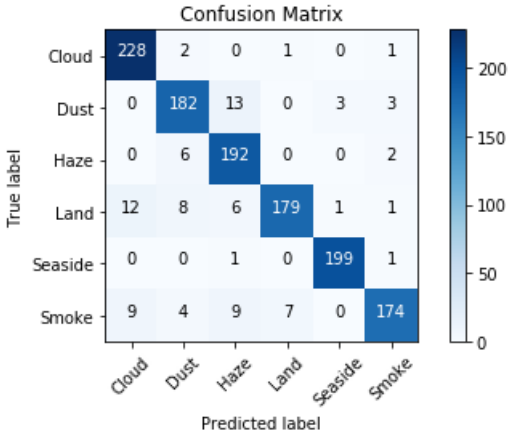
Tabel 3.13: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi keempat Inception V3

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.84	0.85	0.85	232
Dust	0.83	0.84	0.83	201
Haze	0.82	0.91	0.86	200
Land	0.89	0.65	0.75	207
Seaside	0.97	0.97	0.97	201
Smoke	0.73	0.85	0.78	203
Macro Avg	0.85	0.84	0.84	1244
Weighted Avg	0.85	0.84	0.84	1244

3.5.2.5 Evaluasi Model *Inception V3* Kelima

Evaluasi kelima hasil proses *training* pada model *Inception V3* berhasil memprediksi gambar dengan *class Cloud* sebanyak 228 dari

232 gambar. 182 dari 201 gambar dengan *class Dust*. 192 dari 200 gambar dengan *class Haze*. 179 dari 207 gambar dengan *class Land*. 199 dari 201 gambar dengan *class Seaside*. Dan 174 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi keempat hasil proses *training* pada model *Inception V3* memiliki akurasi sebesar 92,7652%. Gambar 3.12 menunjukkan *confusion matrix* hasil dari *training* kelima pada model *Inception V3*.



Gambar 3.12: Confusion matrix training kelima Inception V3

Tabel 3.14 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kelima pada model *Inception V3*.

Tabel 3.14: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kelima Inception V3

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.92	0.98	0.95	232
Dust	0.90	0.91	0.90	201
Haze	0.87	0.96	0.91	200
Land	0.96	0.86	0.91	207
Seaside	0.98	0.99	0.99	201
Smoke	0.96	0.86	0.90	203
Macro Avg	0.93	0.93	0.93	1244
Weighted Avg	0.93	0.93	0.93	1244

3.5.3 Evaluasi Model *Inception Resnet V2*

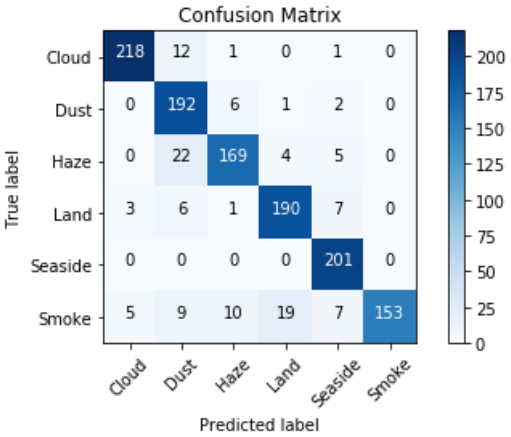
Evaluasi model *Inception ResNet V2* hasil dari proses training dilakukan untuk mendapatkan performa model dalam melakukan klasifikasi gambar satelit. Tabel 3.15 menunjukkan hasil 5 kali evaluasi setelah dilakukan 5 kali proses *training* model *Inception ResNet V2*.

Tabel 3.15: Hasil evaluasi model *Inception ResNet V2*

No	**Akurasi**
1	90,2733%
2	88,1028%
3	91,6398%
4	93,4083%
5	90,5144%
Rata-rata	90,7877%

3.5.3.1 Evaluasi Model *Inception ResNet V2* Pertama

Evaluasi pertama hasil proses *training* pada model *Inception ResNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 218 dari 232 gambar. 122 dari 201 gambar dengan *class Dust*. 169 dari 200 gambar dengan *class Haze*. 190 dari 207 gambar dengan *class Land*. 201 dari 201 gambar dengan *class Seaside*. Dan 153 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi pertama hasil proses *training* pada model *Inception ResNet V2* memiliki akurasi sebesar 90,2733%. Gambar 3.13 menunjukkan *confusion matrix* hasil dari *training* pertama pada model *Inception ResNet V2*.



Gambar 3.13: Confusion matrix training pertama Inception ResNet V2

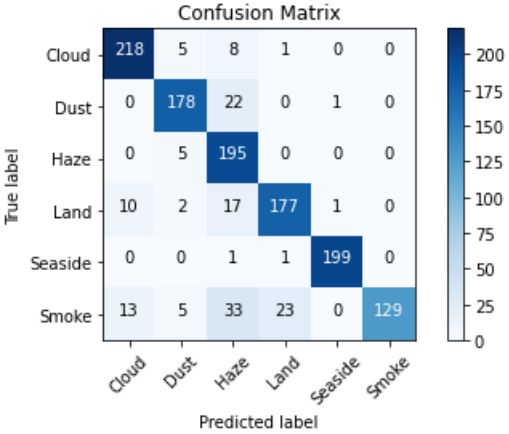
Tabel 3.16 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi pertama pada model *Inception ResNet V2.*

Tabel 3.16: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi pertama Inception ResNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.96	0.94	0.95	232
Dust	0.80	0.96	0.87	201
Haze	0.90	0.84	0.87	200
Land	0.89	0.92	0.90	207
Seaside	0.90	1.00	0.95	201
Smoke	1.00	0.75	0.86	203
Macro Avg	0.91	0.90	0.90	1244
Weighted Avg	0.91	0.90	0.90	1244

3.5.3.2 Evaluasi Model *Inception ResNet V2* Kedua

Evaluasi kedua hasil proses *training* pada model *Inception ResNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 218 dari 232 gambar. 178 dari 201 gambar dengan *class Dust.* 195 dari 200 gambar dengan *class Haze.* 177 dari 207 gambar dengan *class Land.* 199 dari 201 gambar dengan *class Seaside.* Dan 129 dari 203 gambar dengan *class Smoke.* Jika dirata-rata, hasil dari evaluasi kedua hasil proses *training* pada model *Inception ResNet V2* memiliki akurasi sebesar 88,1028%. Gambar 3.14 menunjukkan *confusion matrix* hasil dari *training* kedua pada model *Inception ResNet V2.*



Gambar 3.14: Confusion matrix training kedua Inception ResNet V2

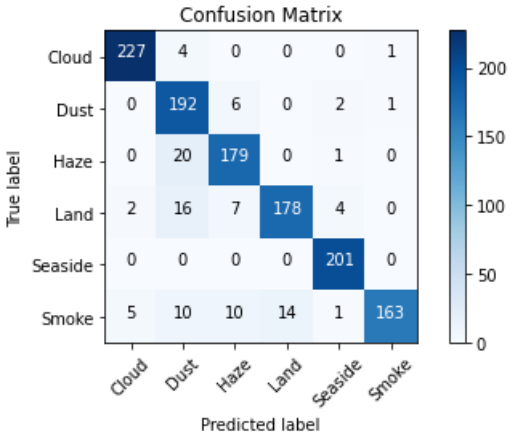
Tabel 3.17 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kedua pada model *Inception ResNet V2.*

Tabel 3.17: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kedua Inception ResNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.90	0.94	0.92	232
Dust	0.91	0.89	0.90	201
Haze	0.71	0.97	0.82	200
Land	0.88	0.86	0.87	207
Seaside	0.99	0.99	0.99	201
Smoke	1.00	0.64	0.78	203
Macro Avg	0.90	0.88	0.88	1244
Weighted Avg	0.90	0.88	0.88	1244

3.5.3.3 Evaluasi Model *Inception ResNet V2* Ketiga

Evaluasi ketiga hasil proses *training* pada model *Inception ResNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 227 dari 232 gambar. 192 dari 201 gambar dengan *class Dust*. 179 dari 200 gambar dengan *class Haze*. 178 dari 207 gambar dengan *class Land*. 201 dari 201 gambar dengan *class Seaside*. Dan 163 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi ketiga hasil proses *training* pada model *Inception ResNet V2* memiliki akurasi sebesar 91,6398%. Gambar 3.15 menunjukkan *confusion matrix* hasil dari *training* ketiga pada model *Inception ResNet V2*.



Gambar 3.15: Confusion matrix training ketiga Inception ResNet V2

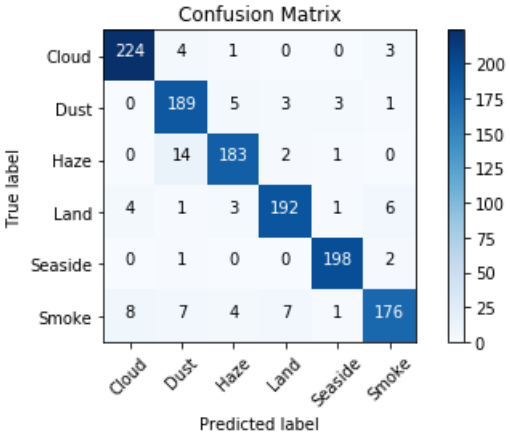
Tabel 3.18 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi ketiga pada model *Inception ResNet V2*.

Tabel 3.18: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi ketiga Inception ResNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.97	0.98	0.97	232
Dust	0.79	0.96	0.87	201
Haze	0.89	0.90	0.89	200
Land	0.93	0.86	0.89	207
Seaside	0.96	1.00	0.98	201
Smoke	0.99	0.80	0.89	203
Macro Avg	0.92	0.92	0.92	1244
Weighted Avg	0.92	0.92	0.82	1244

3.5.3.4 Evaluasi Model *Inception ResNet V2* Keempat

Evaluasi keempat hasil proses *training* pada model *Inception ResNet V2* berhasil memprediksi gambar dengan *class Cloud* sebanyak 224 dari 232 gambar. 189 dari 201 gambar dengan *class Dust*. 183 dari 200 gambar dengan *class Haze*. 192 dari 207 gambar dengan *class Land*. 198 dari 201 gambar dengan *class Seaside*. Dan 176 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi keempat hasil proses *training* pada model *Inception ResNet V2* memiliki akurasi sebesar 93,4083%. Gambar 3.16 menunjukkan *confusion matrix* hasil dari *training* keempat pada model *Inception ResNet V2*.



Gambar 3.16: Confusion matrix training keempat Inception ResNet V2

Tabel 3.19 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi keempat pada model *Inception ResNet V2.*

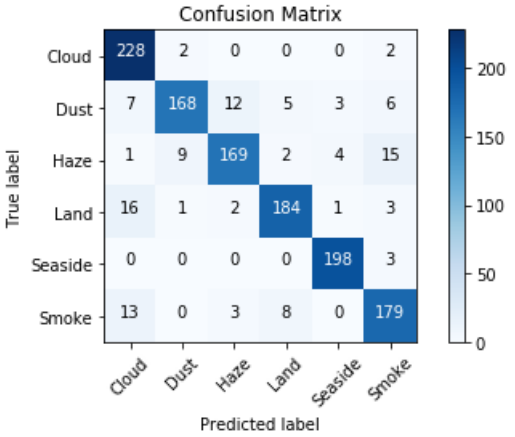
Tabel 3.19: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi keempat Inception ResNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.95	0.97	0.96	232
Dust	0.88	0.94	0.91	201
Haze	0.93	0.92	0.92	200
Land	0.94	0.93	0.93	207
Seaside	0.97	0.99	0.98	201
Smoke	0.94	0.87	0.90	203
Macro Avg	0.93	0.93	0.93	1244
Weighted Avg	0.93	0.93	0.93	1244

3.5.3.5 Evaluasi Model *Inception ResNet V2* Kelima

Evaluasi kelima hasil proses *training* pada model *Inception ResNet V2* berhasil memprediksi gambar dengan *class Cloud* se-

banyak 228 dari 232 gambar. 168 dari 201 gambar dengan *class Dust*. 169 dari 200 gambar dengan *class Haze*. 184 dari 207 gambar dengan *class Land*. 198 dari 201 gambar dengan *class Seaside*. Dan 179 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kelima hasil proses *training* pada model *Inception ResNet V2* memiliki akurasi sebesar 90,5144%. Gambar 3.17 menunjukkan *confusion matrix* hasil dari *training* kelima pada model *Inception ResNet V2*.



Gambar 3.17: Confusion matrix training kelima Inception ResNet V2

Tabel 3.20 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kelima pada model *Inception ResNet V2*.

Tabel 3.20: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kelima Inception ResNet V2

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.86	0.98	0.92	232
Dust	0.93	0.84	0.88	201
Haze	0.91	0.84	0.88	200
Land	0.92	0.89	0.91	207
Seaside	0.96	0.99	0.97	201
Smoke	0.86	0.88	0.87	203
Macro Avg	0.91	0.90	0.90	1244
Weighted Avg	0.91	0.91	0.90	1244

3.5.4 Evaluasi Model *NASNet*

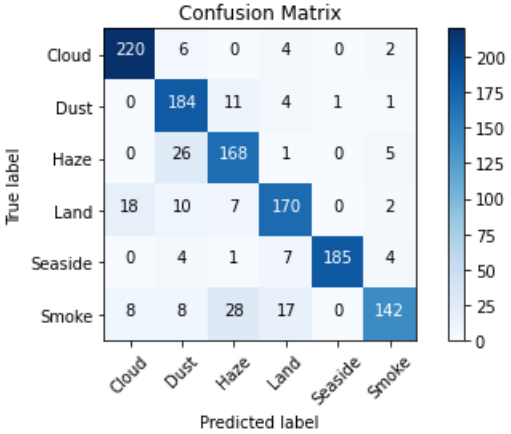
Evaluasi model *NASNet* hasil dari proses training dilakukan untuk mendapatkan performa model dalam melakukan klasifikasi gambar satelit. Tabel 3.21 menunjukkan hasil 5 kali evaluasi setelah dilakukan 5 kali proses *training* model *NASNet*.

Tabel 3.21: Hasil evaluasi model *NASNet*

No	Akurasi
1	85,9324%
2	88,3440%
3	88,9067%
4	75,0000%
5	87,6205%
Rata-rata	85,1607%

3.5.4.1 Evaluasi Model *NASNet* Pertama

Evaluasi pertama hasil proses *training* pada model *NASNet* berhasil memprediksi gambar dengan *class Cloud* sebanyak 220 dari 232 gambar. 184 dari 201 gambar dengan *class Dust*. 168 dari 200 gambar dengan *class Haze*. 170 dari 207 gambar dengan *class Land*. 185 dari 201 gambar dengan *class Seaside*. Dan 142 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi pertama hasil proses *training* pada model *NASNet* memiliki akurasi sebesar 85,9324%. Gambar 3.18 menunjukkan *confusion matrix* hasil dari *training* pertama pada model *NASNet*.



Gambar 3.18: Confusion matrix training pertama NASNet

Tabel 3.22 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi pertama pada model *NASNet*.

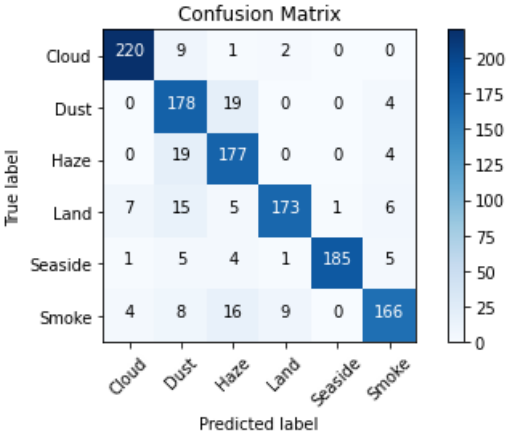
Tabel 3.22: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi pertama NASNet

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.89	0.95	0.92	232
Dust	0.77	0.92	0.84	201
Haze	0.78	0.84	0.81	200
Land	0.84	0.82	0.83	207
Seaside	0.99	0.92	0.96	201
Smoke	0.91	0.70	0.79	203
Macro Avg	0.87	0.86	0.86	1244
Weighted Avg	0.87	0.86	0.86	1244

3.5.4.2 Evaluasi Model *NASNet* Kedua

Evaluasi kedua hasil proses *training* pada model *NASNet* berhasil memprediksi gambar dengan *class Cloud* sebanyak 220 dari 232 gambar. 178 dari 201 gambar dengan *class Dust*. 177 dari 200

gambar dengan *class Haze*. 173 dari 207 gambar dengan *class Land*. 185 dari 201 gambar dengan *class Seaside*. Dan 166 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kedua hasil proses *training* pada model *NASNet* memiliki akurasi sebesar 88,3440%. Gambar 3.19 menunjukkan *confusion matrix* hasil dari *training* kedua pada model *NASNet*.



Gambar 3.19: Confusion matrix training kedua NASNet

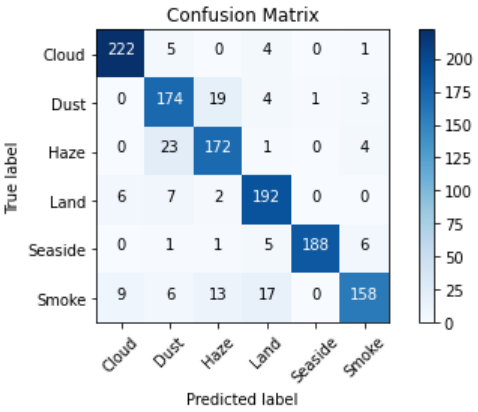
Tabel 3.23 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kedua pada model *NASNet*.

Tabel 3.23: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kedua NASNet

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.95	0.95	0.95	232
Dust	0.77	0.89	0.82	201
Haze	0.80	0.89	0.84	200
Land	0.94	0.84	0.88	207
Seaside	0.99	0.92	0.96	201
Smoke	0.90	0.82	0.86	203
Macro Avg	0.89	0.88	0.88	1244
Weighted Avg	0.89	0.88	0.88	1244

3.5.4.3 Evaluasi Model *NASNet* Ketiga

Evaluasi ketiga hasil proses *training* pada model *NASNet* berhasil memprediksi gambar dengan *class Cloud* sebanyak 222 dari 232 gambar. 174 dari 201 gambar dengan *class Dust*. 172 dari 200 gambar dengan *class Haze*. 192 dari 207 gambar dengan *class Land*. 188 dari 201 gambar dengan *class Seaside*. Dan 158 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi ketiga hasil proses *training* pada model *NASNet* memiliki akurasi sebesar 88,9067%. Gambar 3.20 menunjukkan *confusion matrix* hasil dari *training* ketiga pada model *NASNet*.



Gambar 3.20: Confusion matrix training ketiga NASNet

Tabel 3.24 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi ketiga pada model *NASNet*.

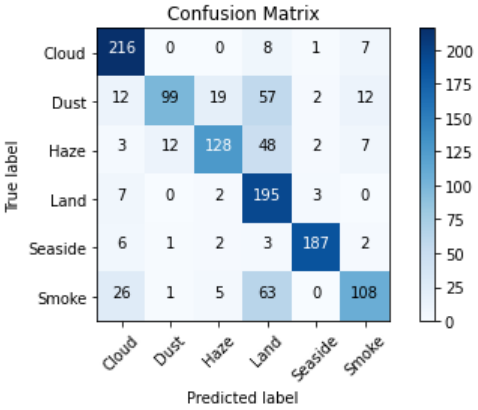
Tabel 3.24: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi ketiga NASNet

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.94	0.96	0.95	232
Dust	0.81	0.87	0.83	201
Haze	0.83	0.86	0.85	200
Land	0.86	0.93	0.89	207
Seaside	0.99	0.94	0.96	201
Smoke	0.92	0.78	0.84	203
Macro Avg	0.89	0.89	0.89	1244
Weighted Avg	0.89	0.89	0.89	1244

3.5.4.4 Evaluasi Model *NASNet* Keempat

Evaluasi keempat hasil proses *training* pada model *NASNet* berhasil memprediksi gambar dengan *class Cloud* sebanyak 216 dari 232 gambar. 99 dari 201 gambar dengan *class Dust*. 128 dari 200 gambar dengan *class Haze*. 195 dari 207 gambar dengan *class Land*.

187 dari 201 gambar dengan *class Seaside*. Dan 108 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi keempat hasil proses *training* pada model *NASNet* memiliki akurasi sebesar 75,0000%. Gambar 3.21 menunjukkan *confusion matrix* hasil dari *training* keempat pada model *NASNet*.



Gambar 3.21: Confusion matrix training keempat NASNet

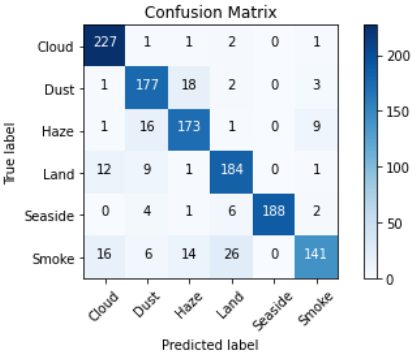
Tabel 3.25 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi keempat pada model *NASNet*.

Tabel 3.25: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi keempat NASNet

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.80	0.93	0.86	232
Dust	0.88	0.49	0.63	201
Haze	0.82	0.64	0.72	200
Land	0.52	0.94	0.67	207
Seaside	0.96	0.93	0.94	201
Smoke	0.79	0.53	0.64	203
Macro Avg	0.80	0.74	0.74	1244
Weighted Avg	0.79	0.75	0.75	1244

3.5.4.5 Evaluasi Model *NASNet* Kelima

evaluasi kelima hasil proses *training* pada model *NASNet* berhasil memprediksi gambar dengan *class Cloud* sebanyak 227 dari 232 gambar. 177 dari 201 gambar dengan *class Dust*. 173 dari 200 gambar dengan *class Haze*. 184 dari 207 gambar dengan *class Land*. 188 dari 201 gambar dengan *class Seaside*. Dan 141 dari 203 gambar dengan *class Smoke*. Jika dirata-rata, hasil dari evaluasi kelima hasil proses *training* pada model *NASNet* memiliki akurasi sebesar 87,6205%. Gambar 3.22 menunjukkan *confusion matrix* hasil dari *training* kelima pada model *NASNet*.



Gambar 3.22: Confusion matrix training kelima NASNet

Tabel 3.26 menunjukan nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* setiap *class* hasil evaluasi kelima pada model *NASNet.*

Tabel 3.26: Tabel nilai *Precision, Recall, F1-Score,* nilai rata-rata makro, serta nilai rata-rata *weight* evaluasi kelima NASNet

	Precision	***Recall***	***F1-Score***	***Support***
Cloud	0.88	0.98	0.93	232
Dust	0.83	0.88	0.86	201
Haze	0.83	0.86	0.85	200
Land	0.83	0.89	0.86	207
Seaside	1.00	0.94	0.97	201
Smoke	0.90	0.69	0.78	203
Macro Avg	0.88	0.87	0.87	1244
Weighted Avg	0.88	0.88	0.87	1244

Halaman ini sengaja dikosongkan

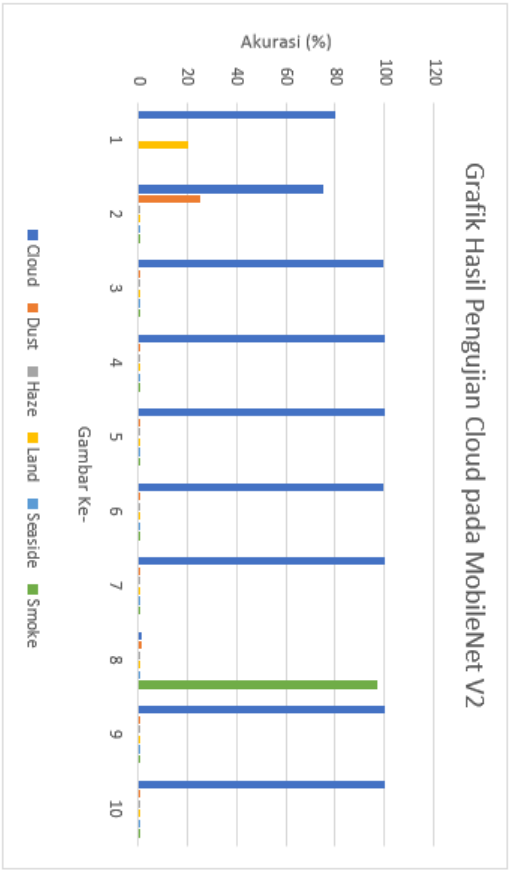
BAB 4
PENGUJIAN DAN ANALISIS

Terdapat empat macam pengujian yang dilakukan pada penelitian ini. Pengujian dilakukan dalam beberapa bagian sesuai dengan *pre-trained* model CNN yang digunakan. Data yang digunakan dalam pengujian diperoleh dari gambar yang sudah disediakan di dalam dataset USTC SmokeRS.

4.1 Pengujian Model *MobileNet V2*

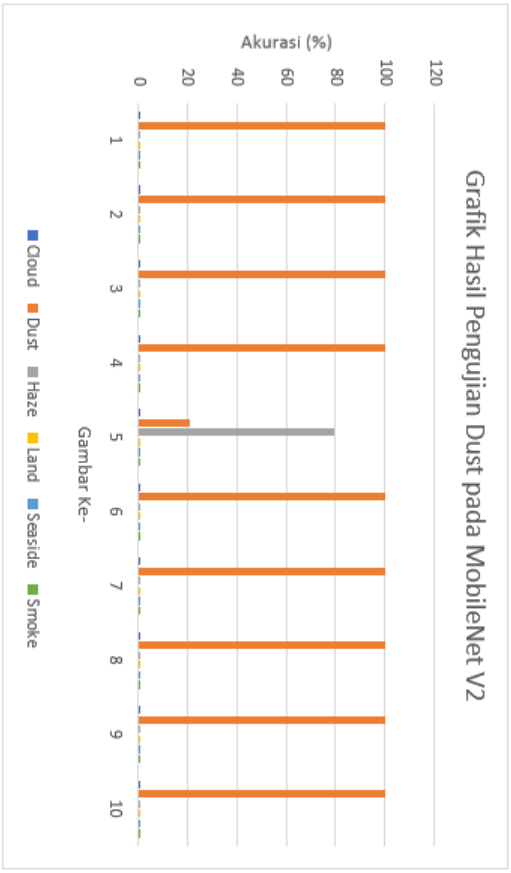
Pengujian model *MobileNet V2* menggunakan *MobileNet V2* dengan hasil training terbaik, yaitu hasil training ketiga. Berikut hasil prediksi model MobileNet V2 terhadap skenario pengujian:

Gambar 4.1 menunjukan pengujian prediksi citra satelit kelas *Cloud*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 9 ,10 dan prediksi **salah sebanyak satu kali** pada citra uji 8.



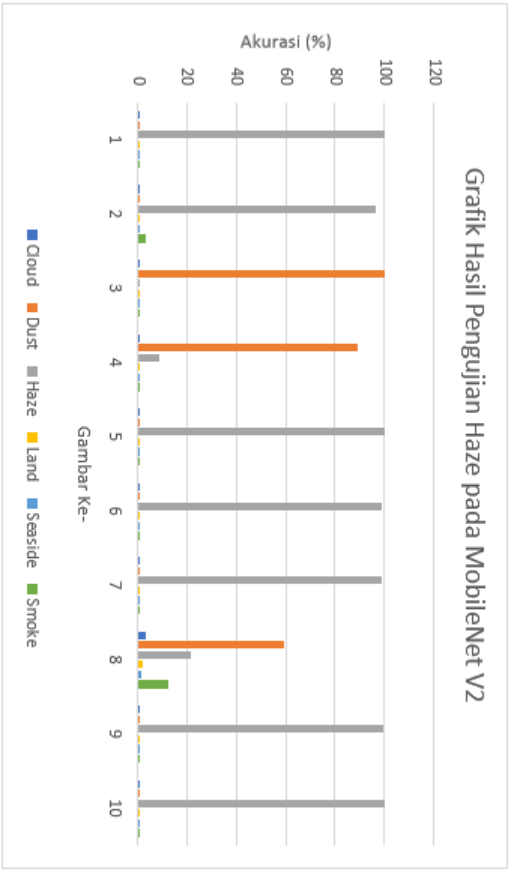
Gambar 4.1: Grafik hasil pengujian *Cloud* MobileNet V2

Gambar 4.2 menunjukan pengujian prediksi citra satelit kelas *Dust*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 6, 7, 8, 9 ,10 dan prediksi **salah sebanyak satu kali** pada citra uji 5.



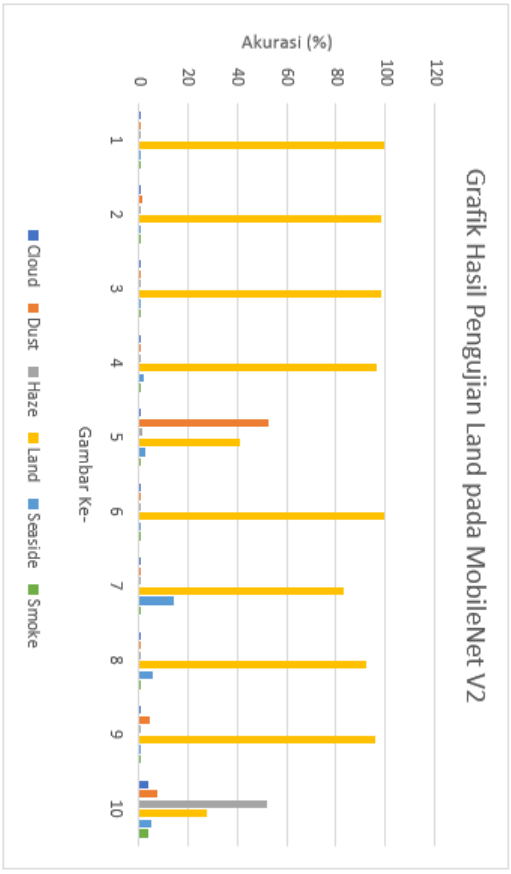
Gambar 4.2: Grafik hasil pengujian *Dust* MobileNet V2

Gambar 4.3 menunjukan pengujian prediksi citra satelit kelas *Haze*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak tujuh kali** pada citra uji 1, 2, 5, 6, 7, 9 ,10 dan prediksi **salah sebanyak tiga kali** pada citra uji 3, 4, 8.



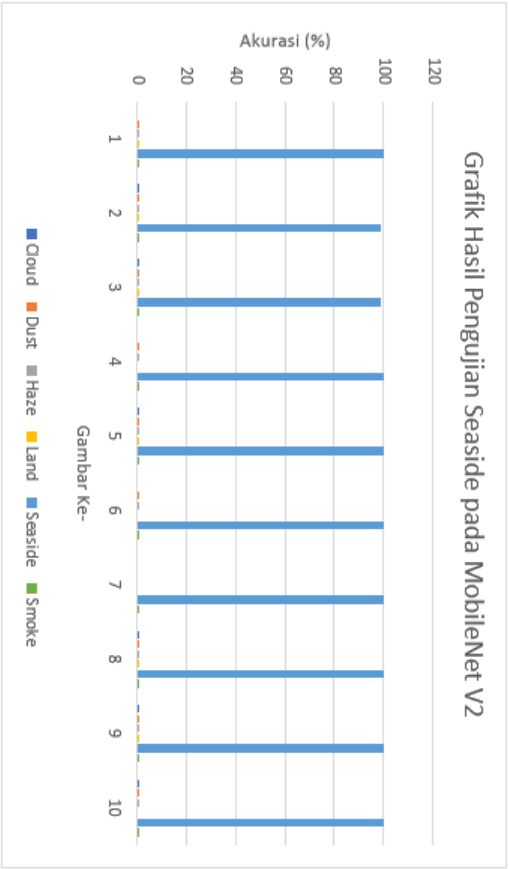
Gambar 4.3: Grafik hasil pengujian *Haze* MobileNet V2

Gambar 4.4 menunjukan pengujian prediksi citra satelit kelas *Land*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak delapan kali** pada citra uji 1, 2, 3, 4, 6, 7, 8, 9 dan prediksi **salah sebanyak dua kali** pada citra uji 5, 10.



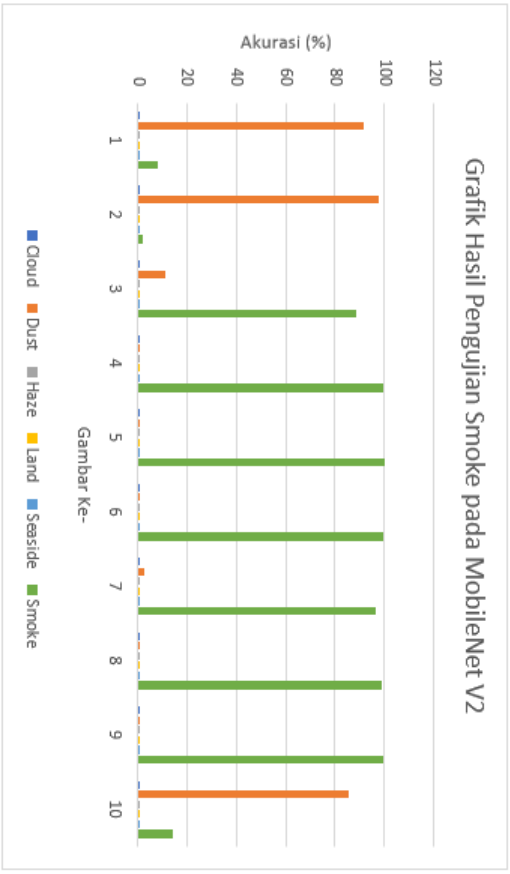
Gambar 4.4: Grafik hasil pengujian *Land* MobileNet V2

Gambar 4.5 menunjukan pengujian prediksi citra satelit kelas *Seaside*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak sepuluh kali** pada semua citra uji.



Gambar 4.5: Grafik hasil pengujian *Seaside* MobileNet V2

Gambar 4.6 menunjukan pengujian prediksi citra satelit kelas *Smoke*. Dari sepuluh kali percobaan menggunakan data tes, model MobileNet V2 berhasil melakukan prediksi **benar sebanyak tujuh kali** pada citra uji 3, 4, 5, 6, 7, 8, 9 dan prediksi **salah sebanyak tiga kali** pada citra uji 1, 2, 10.



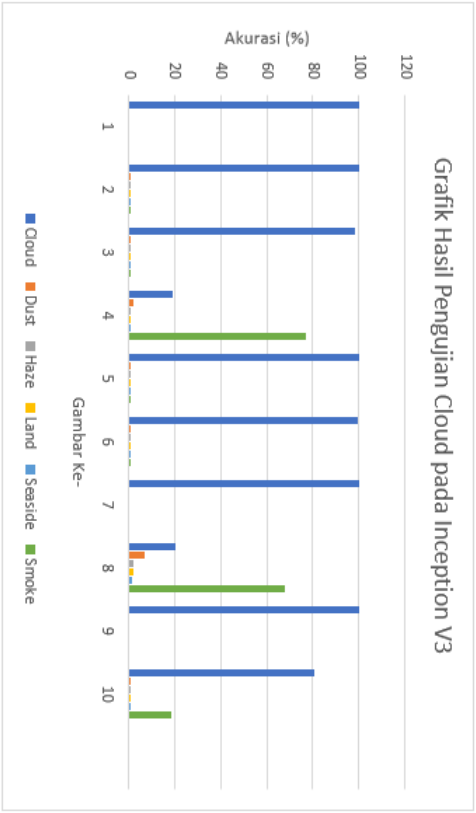
Gambar 4.6: Grafik hasil pengujian *Smoke* MobileNet V2

4.2 Pengujian Model *Inception V3*

Pengujian model *Inception V3* menggunakan *Inception V3* dengan hasil training terbaik, yaitu hasil training kelima. Berikut hasil prediksi model *Inception V3* terhadap skenario pengujian:

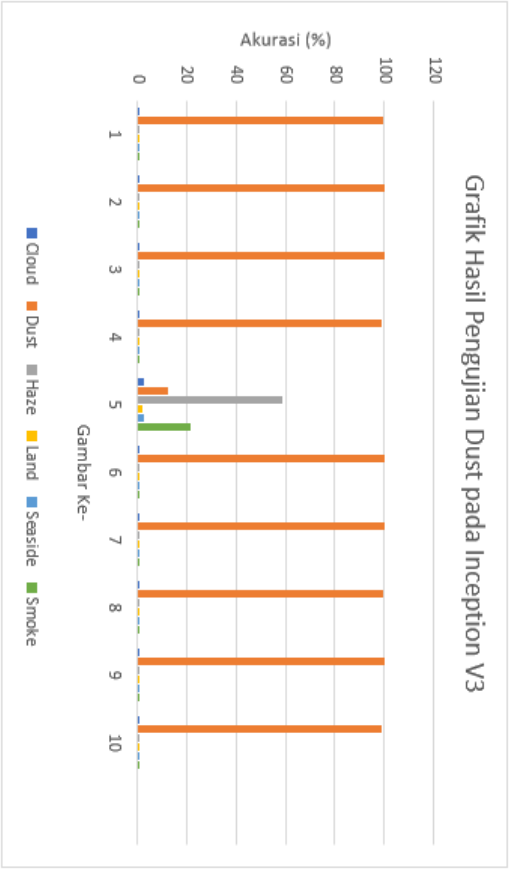
Gambar 4.7 menunjukan pengujian prediksi citra satelit kelas *Cloud*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak de-**

lapan kali pada citra uji 1, 2, 3, 5, 6, 7, 9 ,10 dan prediksi **salah sebanyak dua kali** pada citra uji 4, 8.



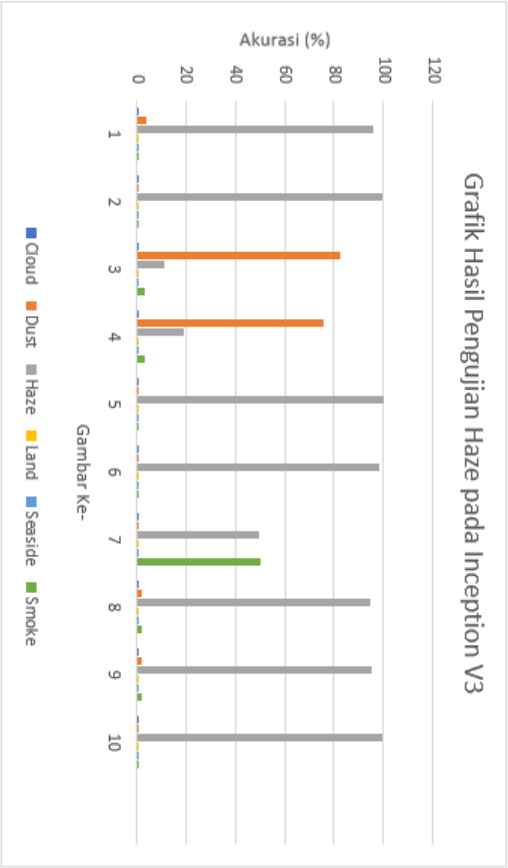
Gambar 4.7: Grafik hasil pengujian *Cloud* Inception V3

Gambar 4.8 menunjukan pengujian prediksi citra satelit kelas *Dust*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 6, 7, 8, 9,10 dan prediksi **salah sebanyak satu kali** pada citra uji 5.



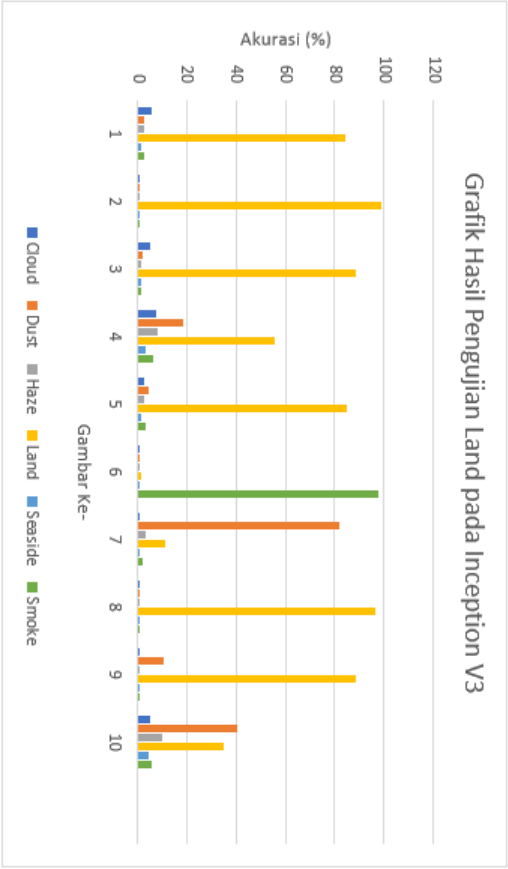
Gambar 4.8: Grafik hasil pengujian *Dust* Inception V3

Gambar 4.9 menunjukan pengujian prediksi citra satelit kelas *Haze*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak tujuh kali** pada citra uji 1, 2, 5, 7, 8, 9,10 dan prediksi **salah sebanyak tiga kali** pada citra uji 3, 4, 6.



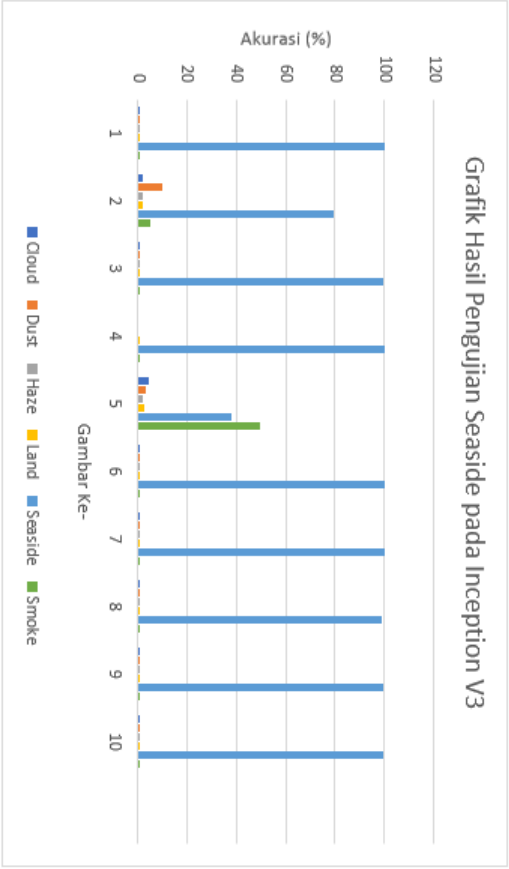
Gambar 4.9: Grafik hasil pengujian *Haze* Inception V3

Gambar 4.10 menunjukan pengujian prediksi citra satelit kelas *Land*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak delapan kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8, 9 dan prediksi **salah sebanyak dua kali** pada citra uji 6, 10.



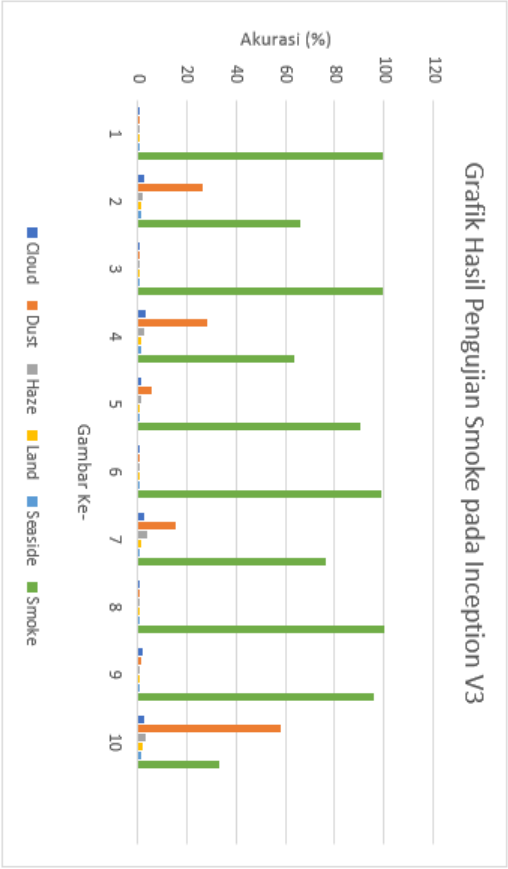
Gambar 4.10: Grafik hasil pengujian *Land* Inception V3

Gambar 4.11 menunjukan pengujian prediksi citra satelit kelas *Seaside*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 6, 7, 8, 9, 10 dan prediksi **salah sebanyak satu kali** pada citra uji 5.



Gambar 4.11: Grafik hasil pengujian *Seaside* Inception V3

Gambar 4.12 menunjukan pengujian prediksi citra satelit kelas *Smoke*. Dari sepuluh kali percobaan menggunakan data tes, model Inception V3 berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8, 9 dan prediksi **salah sebanyak satu kali** pada citra uji 10.

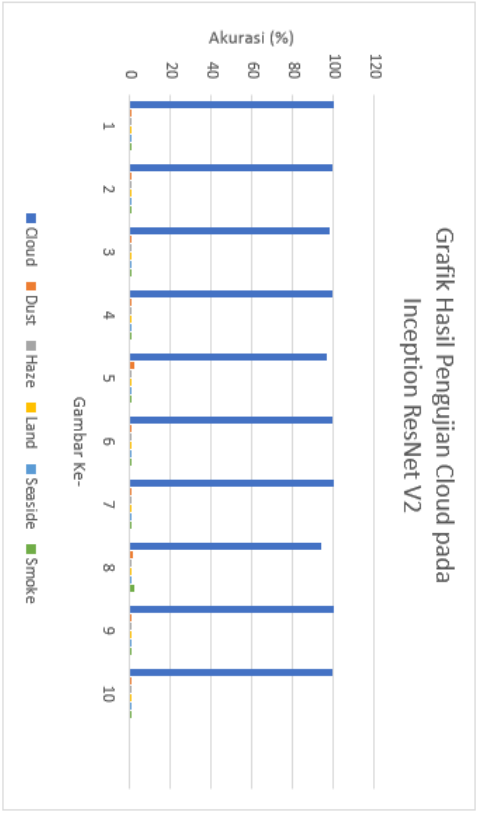


Gambar 4.12: Grafik hasil pengujian *Smoke* Inception V3

4.3 Pengujian Model *Inception ResNet V2*

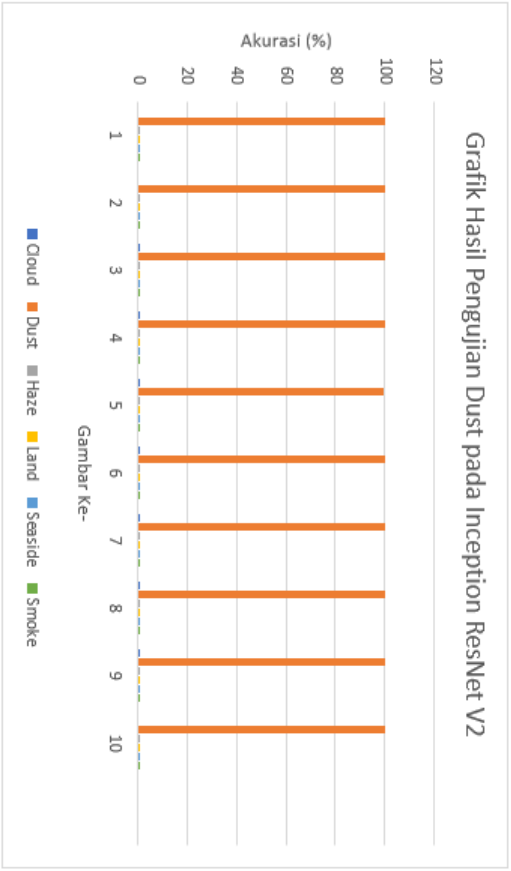
Pengujian model *Inception ResNet V2* menggunakan *Inception ResNet V2* dengan hasil training terbaik, yaitu hasil training keempat. Berikut hasil prediksi model *Inception ResNet V2* terhadap skenario pengujian:

Gambar 4.13 menunjukan pengujian prediksi citra satelit kelas *Cloud*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak sepuluh kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.



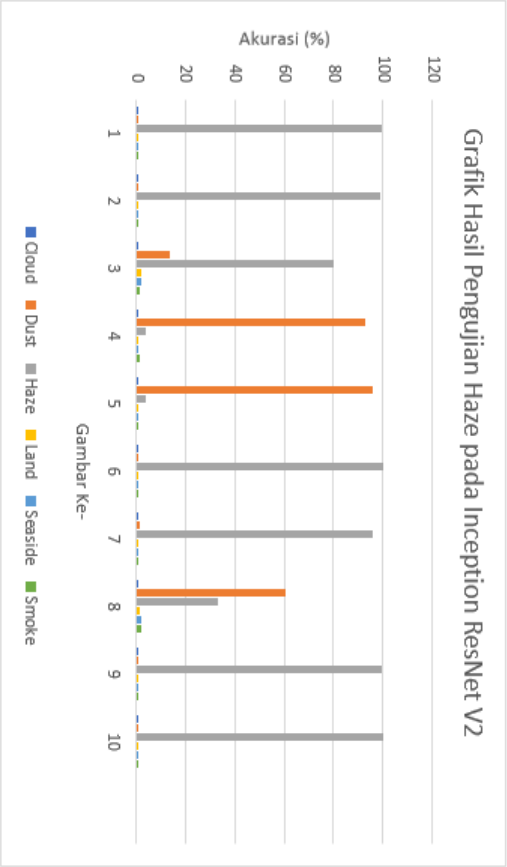
Gambar 4.13: Grafik hasil pengujian *Cloud* Inception ResNet V2

Gambar 4.14 menunjukan pengujian prediksi citra satelit kelas *Dust*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak sepuluh kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.



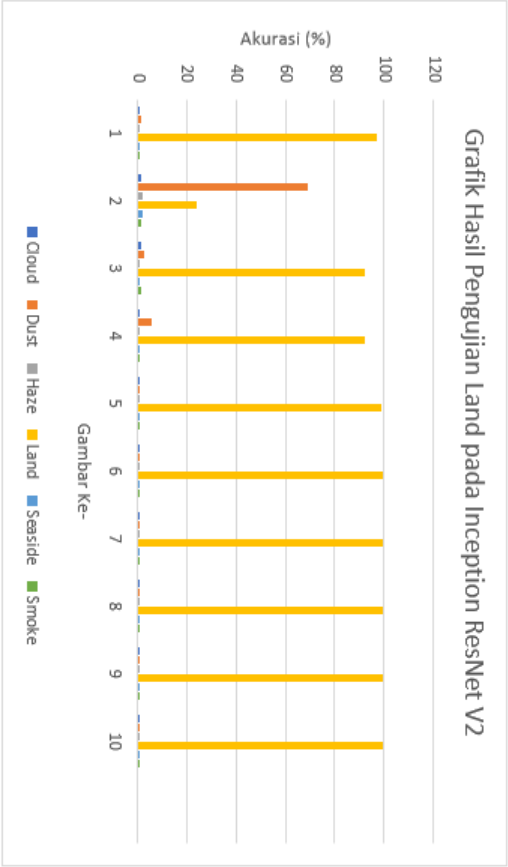
Gambar 4.14: Grafik hasil pengujian *Dust* Inception ResNet V2

Gambar 4.15 menunjukan pengujian prediksi citra satelit kelas *Haze*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak tujuh kali** pada citra uji 1, 2, 5, 7, 8, 9, 10 dan prediksi **salah sebanyak tiga kali** pada citra uji 3, 4, 6.



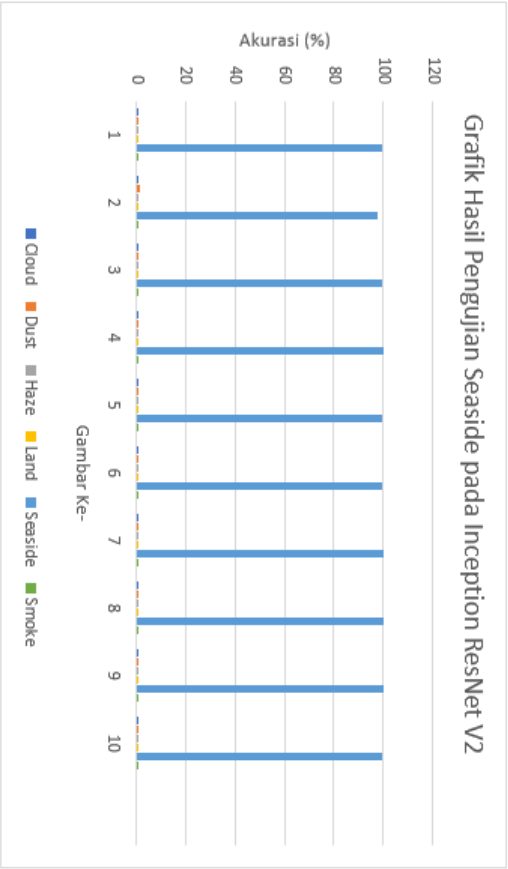
Gambar 4.15: Grafik hasil pengujian *Haze* Inception ResNet V2

Gambar 4.16 menunjukan pengujian prediksi citra satelit kelas *Land*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak delapan kali** pada citra uji 1, 2, 3, 4, 5, 7, 8, 9, 10 dan prediksi **salah sebanyak tiga kali** pada citra uji 6, 10.



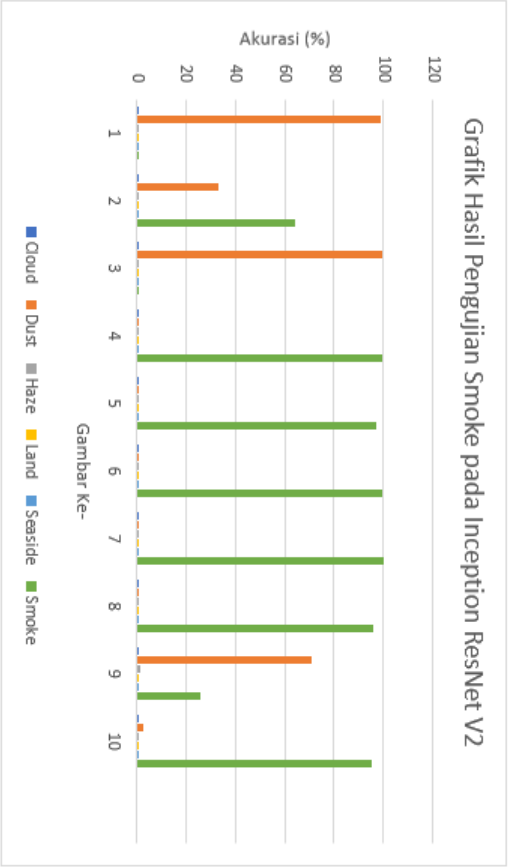
Gambar 4.16: Grafik hasil pengujian *Land* Inception ResNet V2

Gambar 4.17 menunjukan pengujian prediksi citra satelit kelas *Seaside*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak sepuluh kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.



Gambar 4.17: Grafik hasil pengujian *Seaside* Inception ResNet V2

Gambar 4.18 menunjukan pengujian prediksi citra satelit kelas *Smoke*. Dari sepuluh kali percobaan menggunakan data tes, model Inception ResNet V2 berhasil melakukan prediksi **benar sebanyak tujuh kali** pada citra uji 2, 4, 5, 6, 7, 8, 10 dan prediksi **salah sebanyak tiga kali** pada citra uji 1, 3, 9.

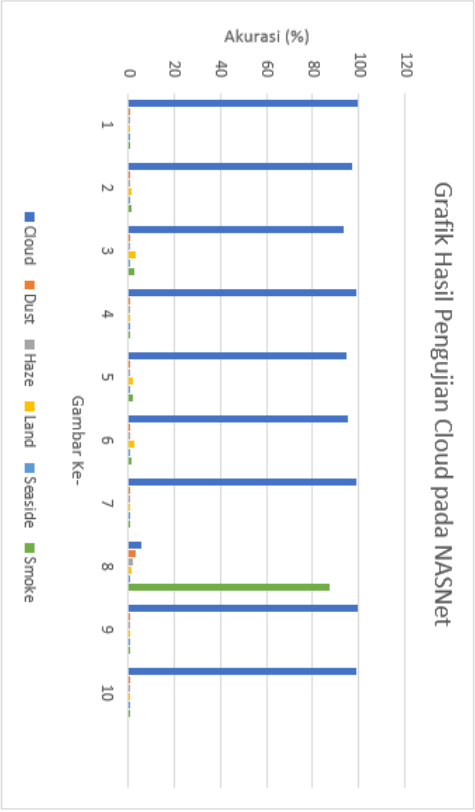


Gambar 4.18: Grafik hasil pengujian *Smoke* Inception ResNet V2

4.4 Pengujian Model *NASNet*

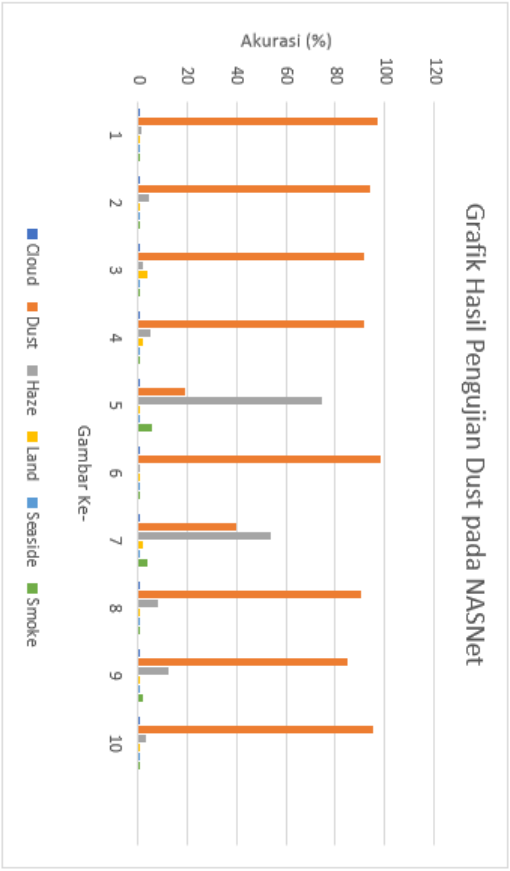
Pengujian model *NASNet* menggunakan *NASNet* dengan hasil training terbaik, yaitu hasil training ketiga. Berikut hasil prediksi model *NASNet* terhadap skenario pengujian:

Gambar 4.19 menunjukan pengujian prediksi citra satelit kelas *Cloud*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 9 ,10 dan prediksi **salah sebanyak satu kali** pada citra uji 8.



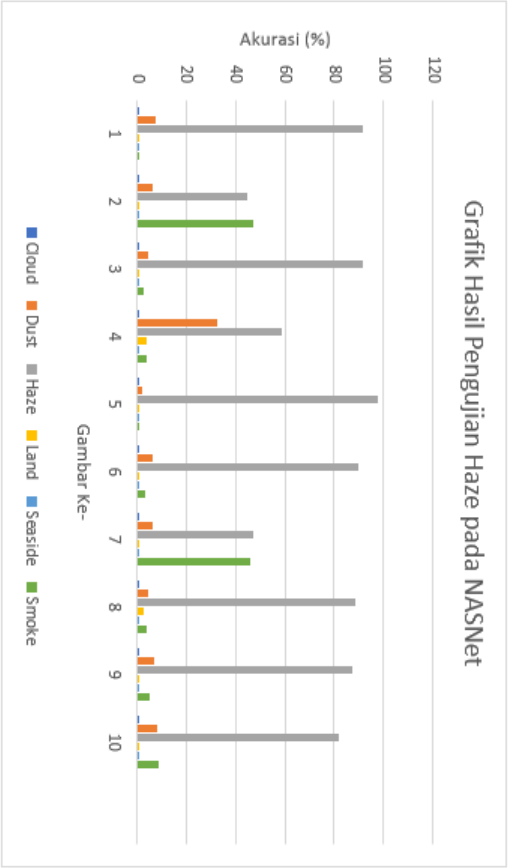
Gambar 4.19: Grafik hasil pengujian *Cloud* NASNet

Gambar 4.20 menunjukan pengujian prediksi citra satelit kelas *Dust*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak delapan kali** pada citra uji 1, 2, 3, 4, 6, 8, 9 ,10 dan prediksi **salah sebanyak dua kali** pada citra uji 5, 7.



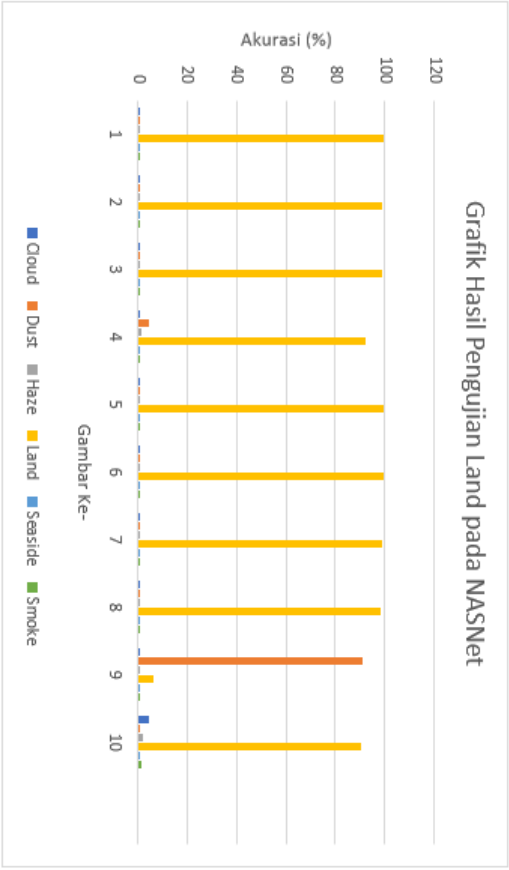
Gambar 4.20: Grafik hasil pengujian *Dust* NASNet

Gambar 4.21 menunjukan pengujian prediksi citra satelit kelas *Haze*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 3, 4, 5, 6, 7, 8, 9 ,10 dan prediksi **salah sebanyak satu kali** pada citra uji 2.



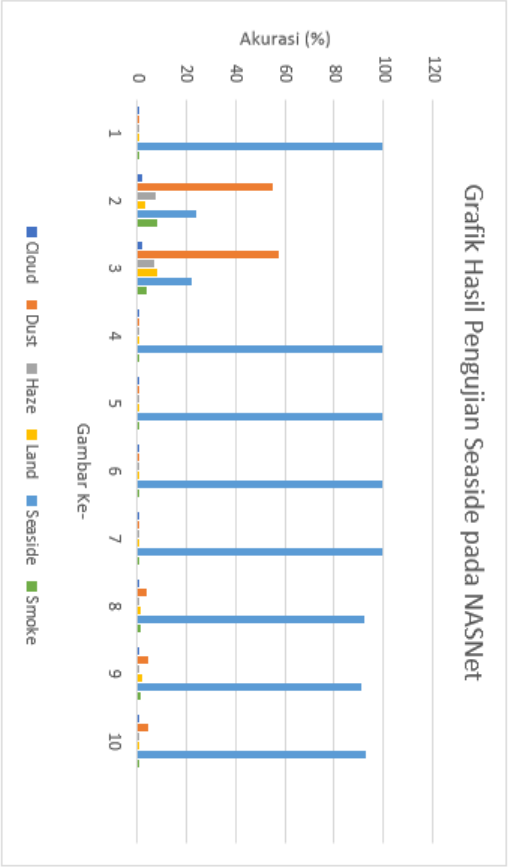
Gambar 4.21: Grafik hasil pengujian *Haze* NASNet

Gambar 4.22 menunjukan pengujian prediksi citra satelit kelas *Land*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 1, 2, 3, 4, 5, 6, 7, 8 ,10 dan prediksi **salah sebanyak satu kali** pada citra uji 9.



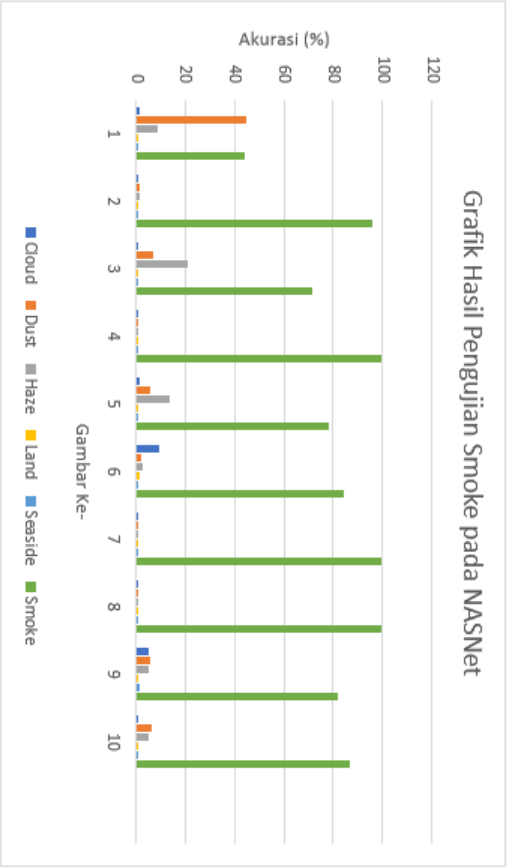
Gambar 4.22: Grafik hasil pengujian *Land* NASNet

Gambar 4.23 menunjukan pengujian prediksi citra satelit kelas *seaside*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak delapan kali** pada 1, 4, 5, 6, 7, 8, 9, 10 dan prediksi **salah sebanyak dua kali** pada citra uji 2, 3.



Gambar 4.23: Grafik hasil pengujian *Seaside* NASNet

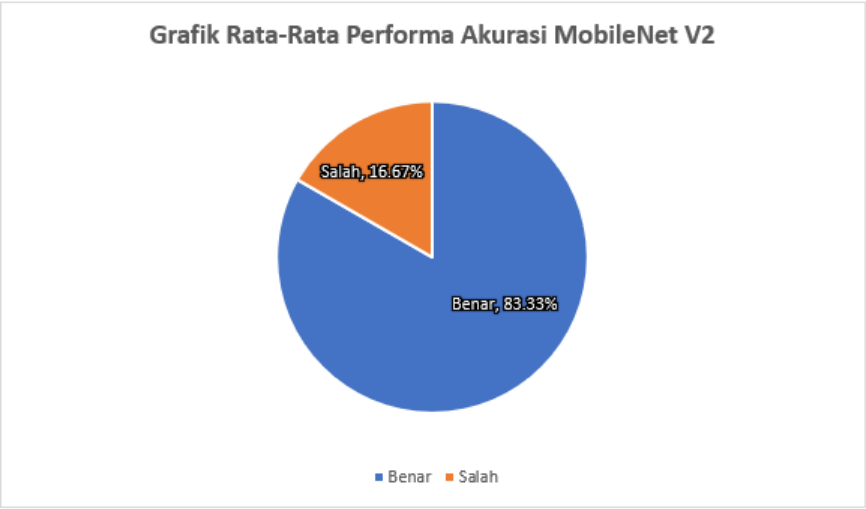
Gambar 4.24 menunjukan pengujian prediksi citra satelit kelas *Smoke*. Dari sepuluh kali percobaan menggunakan data tes, model NASNet berhasil melakukan prediksi **benar sebanyak sembilan kali** pada citra uji 2, 3, 4, 5, 6, 7, 8, 9, 10 dan prediksi **salah sebanyak satu kali** pada citra uji 1.



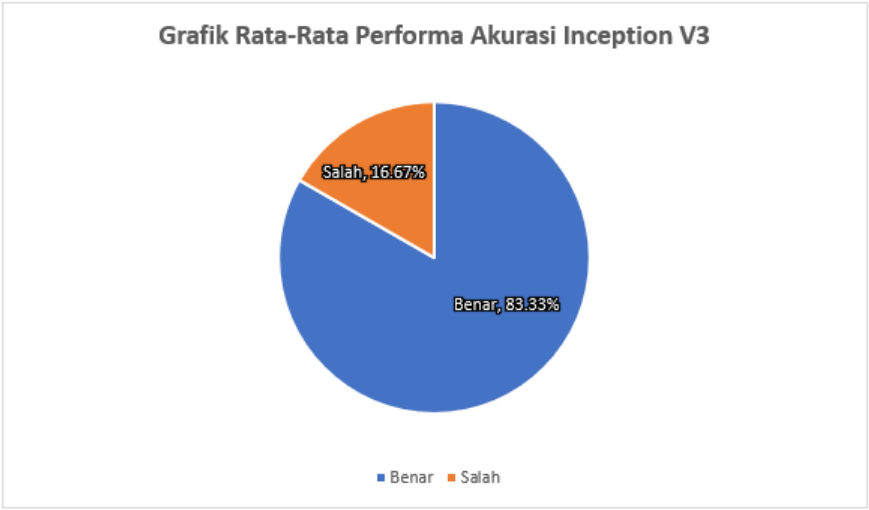
Gambar 4.24: Grafik hasil pengujian *Smoke* NASNet

4.5 Analisis Perbandingan Hasil Pengujian

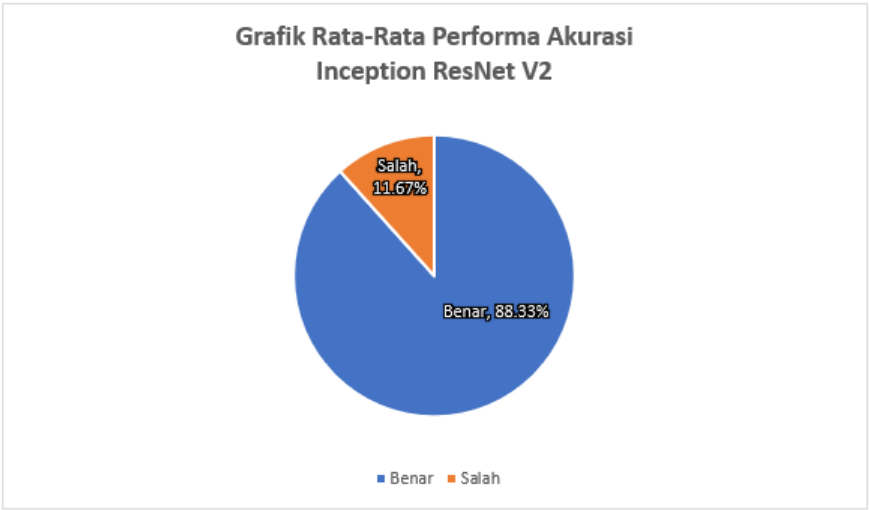
Setelah dilakukan uji performa prediksi didapatkan nilai rata-rata performa model yang dapat dilihat pada Gambar 4.25, Gambar 4.26, Gambar 4.27 dan Gambar 4.28. Akurasi pada model MobileNet V2 sebesar 83,33%. 83,33% pada Inception V3. 88,33% pada Inception ResNet V2. 86,67% pada NASNet. Sehingga dapat disimpulkan bahwa hasil model Inception ResNet V2 memiliki performa dalam prediksi citra input yang lebih baik jika dibandingkan performa akurasi model lainnya.



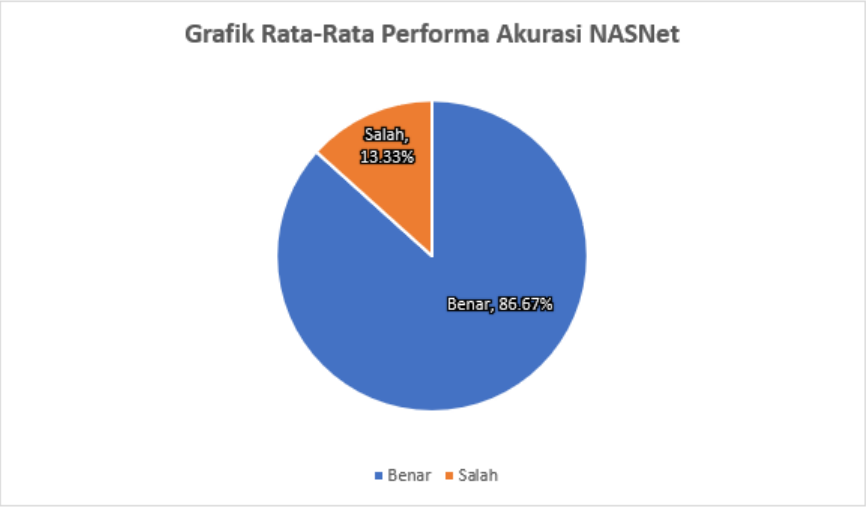
Gambar 4.25: Grafik rata-rata performa prediksi MobileNet V2(%)



Gambar 4.26: Grafik rata-rata performa prediksi Inception V3(%)



Gambar 4.27: Grafik rata-rata performaprediksi Inception ResNet V2(%)



Gambar 4.28: Grafik rata-rata performa prediksi NASNet(%)

BAB 5
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan, dapat ditarik beberapa kesimpulan sebagai berikut:

1. Berdasarkan hasil evaluasi, model *Inception ResNet V2* memiliki rata-rata akurasi yang terbaik. Model tersebut memiliki rata-rata akurasi sebesar 90,7877%.
2. Jumlah *epoch* akan mempengaruhi nilai akurasi, *pre-trained* model *Inception V3* yang di train dengan menggunakan 30 *epoch* memiliki akurasi lebih besar dari *NASNet* yang hanya di train dengan menggunakan 20 *epoch*. Walaupun *NASNet* memiliki jumlah *layer* yang lebih banyak.
3. Berdasarkan hasil pengujian, citra dengan *class Seaside* memiliki tingkat prediksi yang tertinggi dibandingkan dengan *class* lainnya.
4. Berdasarkan hasil pengujian, citra dengan *class Haze* memiliki tingkat prediksi yang terendah dibandingkan dengan *class* lainnya.

5.2 Saran

Demi pengembangan lebih lanjut mengenai penelitian pada tugas akhir ini, disarankan beberapa langkah lanjutan sebagai berikut:

1. Memperbanyak data citra satelit untuk meningkatkan performa klasifikasi model *Convolutional Neural Network*.
2. Memperbanyak jumlah *epoch* yang digunakan pada proses *training*.
3. Melakukan *training* dengan menggunakan *pre-trained model* CNN lain yang belum digunakan pada penelitian ini.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] S. Bahri, "Kajian penyebaran kabut asap kebakaran hutan dan lahan di wilayah sumatera bagian utara dan kemungkinan mengatasinya dengan tmc," Jurnal Sains & Teknologi Modifikasi Cuaca. (Dikutip pada halaman 1).

[2] G. Team, "Giss surface temperature analysis (gistemp), version 4," 2019. (Dikutip pada halaman 1).

[3] N. Lenssen, G. Schmidt, J. Hansen, M. Menne, A. Persin, R. Ruedy, and D. Zyss, "Improvements in the gistemp uncertainty model," J. Geophys. Res. Atmos., vol. 124, no. 12, pp. 6307–6326, 2019. (Dikutip pada halaman 1).

[4] K. L. H. dan Kehutanan, "Rekapitulasi luas kebakaran hutan dan lahan (ha) per provinsi di indonesia tahun 2014-2019," 2019. (Dikutip pada halaman 1).

[5] I. Putra, Klasifikasi citra menggunakan convolutional neural network (CNN) pada caltech 101. PhD thesis, Institut Teknologi Sepuluh Nopember, 2016. (Dikutip pada halaman 2).

[6] J. Wang, Q. Qin, J. Zhao, X. Ye, X. Qin, X. Yang, J. Wang, X. Zheng, and Y. Sun, "A knowledge-based method for road damage detection using high-resolution remote sensing image," 2015. (Dikutip pada halaman 5).

[7] L. Somantri, "Teknologi penginderaan jauh (remote sensing)," 2009. (Dikutip pada halaman 5).

[8] A. Ahmad Hania, "Mengenal artificial intelligence, machine learning, neural network, dan deep learning," Jurnal Teknologi Indonesia, 2017. (Dikutip pada halaman 5).

[9] M. Zufar and B. Setiyono, "Convolutional neural networks untuk pengenalan wajah secara real-time," Jurnal Sains dan Seni ITS, vol. 5, no. 2, 2016. (Dikutip pada halaman 6).

[10] S. Sena, "Pengenalan deep learning part 7 : Convolutional neural network (cnn)," November 2017. [Online; posted 13-November-2017]. (Dikutip pada halaman 6, 7, 8, 9, 10, 14).

[11] T. Nurhikmat et al., "Implementasi deep learning untuk image classification menggunakan algoritma convolutional neural network (cnn) pada citra wayang golek," 2018. (Dikutip pada halaman 7, 9, 10).

[12] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015. (Dikutip pada halaman 10, 11).

[13] F. Syahrian, "Perbandingan metode optimasi stochastic gradient descent, adadelta, dan adam pada jaringan saraf tiruan dalam klasifikasi data aritmia," (Dikutip pada halaman 14).

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017. (Dikutip pada halaman 14, 15).

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–9, 2015. (Dikutip pada halaman 15).

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," CoRR, vol. abs/1512.00567, 2015. (Dikutip pada halaman 16).

[17] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," CoRR, vol. abs/1707.07012, 2017. (Dikutip pada halaman 16, 17).

[18] S. S. Girija, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," Software available from tensorflow. org, 2016. (Dikutip pada halaman 19, 20).

[19] R. Ba, C. Chen, J. Yuan, W. Song, and S. Lo, "Smokenet: Satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention," Remote Sensing, vol. 11, no. 14, p. 1702, 2019. (Dikutip pada halaman 22).

[20] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," Scientific data, vol. 5, p. 180161, 2018. (Dikutip pada halaman 23).

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014. (Dikutip pada halaman 26).

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

Halaman ini sengaja dikosongkan

BIOGRAFI PENULIS

Kentani Langgalih Prioiko, lahir pada 15 April 1998 di Bontang, Provinsi Kalimantan Timur. Penulis merupakan anak kedua dari tiga bersaudara. Saat ini penulis tinggal di Perumahan Graha Mutiara, Kelurahan Pengasinan, Kecamatan Rawalumbu, Kota Bekasi. Pada tahun 2009 menyelesaikan pendidikan di SDIT Thariq Bin Ziyad. Tahun 2012 lulus dari SMP Negeri 1 Bekasi serta pada tahun 2015 lulus dari SMA Negeri 1 Bekasi. Penulis diterima di Program Studi S-1 Departemen Teknik Komputer Fakultas Teknologi Elektro ITS. Penulis berhasil menyelesaikan tugas akhir dengan judul ***Convolutional Neural Network* Untuk Klasifikasi Citra Asap Pada Gambar Satelit**. Bagi pembaca yang memiliki kritik, saran atau pertanyaan mengenai tugas akhir ini dapat menghubungi penulis melalui email kenta.prioko@gmail.com.

Halaman ini sengaja dikosongkan