IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH*

LAPORAN TUGAS AKHIR

Disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana
di Jurusan Teknik Elektro Universitas Jenderal Soedirman

Disusun oleh:

Fendy Prayogi
H1A016029

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA

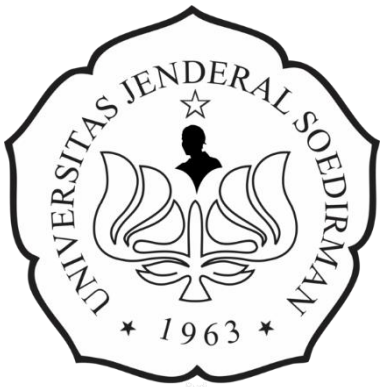
2020


UNIVERSITAS JENDERAL SOEDIRMAN
1963

IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH*

LAPORAN TUGAS AKHIR

Disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana
di Jurusan Teknik Elektro Universitas Jenderal Soedirman



Disusun oleh:

Fendy Prayogi
H1A016029

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA
2020

HALAMAN PENGESAHAN

Laporan Tugas Akhir dengan Judul:

IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK (*CNN) PADA CITRA *GOOGLE EARTH*

Disusun oleh:
Fendy Prayogi
H1A016029

Diajukan untuk memenuhi salah satu persyaratan
memperoleh gelar Sarjana Teknik pada
Jurusan/Program Studi Teknik Elektro
Fakultas Teknik
Universitas Jenderal Soedirman

Diterima dan disetujui
Pada Tanggal : _____

Pembimbing I	Pembimbing II
Imron Rosyadi, ST., M.SC	Farida Asriani, S.Si., M.T.
(NIP : 197909242003121003)	(NIP : 197502012000032005)

Mengetahui,
Dekan Fakultas Teknik

Dr. Eng. Suroso, S.T., M.Eng.
NIP . 197812242001121002

HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Laporan Tugas Akhir dengan judul "**IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH***" ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Purbalingga, 6 Februari 2020

Fendy Prayogi
NIM. H1A016029

HALAMAN MOTTO DAN PERSEMBAHAN

MOTTO

“ Bekerjalah kamu, maka Allah dan rasul Nya serta orang orang mukmin akan melihat pekerjaan mu itu dan kamu akan dikembalikan kepada Allah lalu diberitakan kepada Nya apa yang telah kamu kerjakan ” **(Q.S At-Taubah : 105)**

PERSEMBAHAN

Tugas akhir dan juga penulisan laporan tugas akhir merupakan proses yang tidak mudah maka dari itu penulis menyadari bahwa keberhasilan kegiatan tugas akhir dan proses pembuatan laporan ini tidak lepas dari bantuan berbagai pihak, oleh karena itu penulis mengucapkan terima kasih kepada :

1. Allah Subhanahu Wa Taala yang selalu melimpahkan rahmat-Nya sehingga laporan tugas akhir dapat terselesaikan.
2. Kemudian dan seterusnya nabi Muhammad Solallohu alaihi wa sallam selaku guru besar penulis sehingga penulis bisa mengerti akan pentingnya sebuah ilmu.
3. Ibu, Bapak dan saudara-saudara atas limpahan do’a, dukungan dan semangat yang tidak pernah padam.
4. Bapak Imron Rosyadi, ST., M.SC selaku Dosen Pembimbing I Tugas Akhir.
5. Ibu Farida Asriani, S.Si., M.T. selaku Pembimbing II Tugas Akhir dan Ketua Jurusan / Program Studi Teknik Elektro.
6. Semua pihak yang tidak dapat disebutkan satu persatu yang telah membantu penulis baik dalam melaksanakan.

RINGKASAN

IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH*

Fendy Prayogi

Masyarakat Indonesia umumnya dikenal sebagai masyarakat agraris, karena hampir sebagian besar dari penduduk Indonesia memiliki status pekerjaan dalam sektor pertanian sebesar 35.703.074 jiwa. Keterlibatan sektor pertanian juga menjadi kebutuhan pokok pangan bagi masyarakat Indonesia, selain dari faktor wilayahnya yang beriklim tropis. Namun identifikasi suatu area lahan pertanian masih dengan metode tradisional dilakukan dengan survei lapangan, dan memerlukan banyak waktu maupun biaya. Meskipun sudah ada lembaga statistika milik pemerintahan yang menggunakan teknologi penginderaan, namun tidak banyak kalangan yang dapat mengimplementasikannya.

Deep learning menjadi pembelajaran fitur dan pengklasifikasi suatu objek secara bersamaan, dan menggunakan data pelatihan dengan metode CNN yang dinilai efektif dalam berbagai pengenalan. Proses dalam deteksi lahan pertanian sawah pada citra *Google Earth Pro* dilakukan dengan mengidentifikasi berbagai objek yang melibatkan citra yang kompleks dan variatif. Kemudian datasheet yang diperoleh dikelola pada Google Colaboratory untuk dilatih dan didapatkan model, serta diaplikasikan hasilnya dalam pengujian citra. Terdapat dua arsitektur CNN yang digunakan *LeNet-5* dan *VGG-16Net,* sehingga akan dibandingkan hasil perbandingan dari pemilihan arsitekur tersebut.

Berdasarkan objek yang berukuran sangat kecil dengan metode CPM dapat menghasilkan potongan citra menjadi 89.100 citra yang berukuran 56x56 piksel sehingga mengoptimalkan dalam identifikasi objek geografis lahan pertanian sawah dengan akurasi yang cukup tinggi. Perbandingan kurva akurasi dan kegagalan menunjukkan kedua arsitektur dalam kondisi *goodfit*, namun asitektur *LeNet-5* Modifikasi memiliki hasil yang lebih baik dalam pelatihan dengan akurasi akhir sebesar 0.9936 dan kegagalan 0.0193 dibandingkan dengan arsitektur *VGG-16Net* yang akurasi diperoleh 0.9923 dan kegagalan 0.0229 dalam skala nilai 0 sampai 1. Pengaruh objek citra terutama pada kualitas citra dan ketinggiannya dari ketiga pulau dapat berdampak pada hasil prediksi maupun presentase akurasinya.

Kata kunci : *Identifikasi, Convolution Neural Network (CNN), Google Earth, Deep Learning*

SUMMARY

IDENTIFICATION OF PADDY FIELDS USING CONVOLUTIONAL NEURAL NETWORK (CNN) ON GOOGLE EARTH IMAGES

Fendy Prayogi

Indonesian society is known as an agrarian society, because most of the population of Indonesia has employment status in the agricultural sector of 35,703,074 people. Related to the agricultural sector is also a basic need for the people of Indonesia, apart from the factor of the region with a tropical climate. Identification of agricultural land areas that are still carried out by traditional methods carried out by field surveys, and need a lot of time. Even though there are government-owned statistical institutions that use sensing technology, not many people can implement it.

In deep learning into learning features and classifying an object together, and using training data with the CNN method that addresses effective in various introductions. The process of detecting agricultural land in Google Earth Pro imagery is carried out with various objects that involve complex and varied imagery. Then the datasheet obtained successfully in Google Collaboratory to be drilled and obtained by the model, and applied the results in image testing. There are two CNN architectures used by LeNet-5 and VGG-16Net, so they will compare the results obtained from the architect's selection.

Based on very small objects with CPM method, it can produce 89,100 image pieces with a size of 56x56 pixels, thus optimizing the identification of geographical objects of paddy fields with high accuracy. Comparison of accuracy and failure curves shows both architectures in goodfit conditions, but the LeNet-5 Modified architecture has better results in training with a final accuracy of 0.9936 and failure of 0.0193 compared to VGG-16Net architecture where accuracy is obtained by 0.9923 and failure by 0.0229 on a scale of values of 0 to 1. The influence of image objects, especially on the image quality and height of the three islands can have an impact on the prediction results and the percentage of accuracy.

Keywords : Identification, Convolution Neural Network (CNN), Paddy Fields, Google Earth, Deep Learning

PRAKATA

Puji syukur kehadirat Allah subhanahu wa taala yang telah melimpahkan berkah dan rahmat-Nya laporan Tugas Akhir "**IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH***" ini dapat disusun. Terimakasih kami sampaikan kepada seluruh pihak yang telah membantu terwujudnya laporan ini, diantaranya: Dekan FT Unsoed, Wakil Dekan Akademik FT Unsoed, Kajur Teknik Elektro Unsoed, Sekretaris Jurusan Teknik Elektro Unsoed, bapak-ibu dosen Teknik Elektro Unsoed, Teman-teman Teknik Elektro Unsoed dan pihak-pihak lain yang tidak dapat kami sebutkan satu persatu.

Purbalingga, 6 Februari 2020

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR ISTILAH DAN SINGKATAN

BPS : Badan Pusat Statistika

KSA : Kerangka Sampel Area

Deep learning: Salah satu jenis teknik pembelajaran mesin (*machine learning)* menggunakan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan

CNN : *Convolutional Neural Network*

Bit Depth : Kedalaman bit

Eye Alt : Ketinggian pada titik tertentu

IPYNB : *IPython Notebooks*

MLP : *Multi Layer Percepton*

Stride : Parameter yang menentukan berapa jumlah pergeseran filter.

Pooling Layer :Mengurangi dimensi dari *feature map* (*downsampling*), sehingga mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting*

LeNet-5 : Salah satu arsitektur paling sederhana dalam CNN yang memiliki 2 *convolution layer* dan 3 fully-connected *layer*

VGG-16Net : Arsitektur CNN yang efisien karena *convolutional layer* yang digunakan dalam arsitektur ini ada 2 jenis, yaitu *convolutional layer* dengan ukuran filter 3x3 (*conv3*) dan ukuran filter 1x1 (*conv1*).

ANN : *Artificial Neural Network*

Sliding Window : sebuah metode yang paling umum digunakan dalam pemrosesan citra digital.

CPM : *Chopping Picture Methode*

API : *Application Programming Interface*

DAFTAR SIMBOL

z_j	:	*Output unit*
w	:	*Bobot unit*
x	:	*Input unit* $(x_i, i = 1, \dots, n)$
h_j	:	Operasi aktivasi/*non-linear*
W	:	Panjang/Tinggi Input
N	:	Panjang/Tinggi Filter
P	:	*Zero Padding*
S	:	*Stride*
$s(t)$	:	Output tunggal (*Feature Map)*
x	:	Input argument pertama
t	:	*pixel*

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Masyarakat Indonesia umumnya dikenal sebagai masyarakat agraris, yang berarti perekonomian bertumpu pada sektor pertanian. Hal itu dikarenakan Indonesia memiliki sebagian besar daratan yang dilalui oleh sepertiga lautan dari luas keseluruhan wilayah negara Indonesia. Berdasarkan data yang diperoleh dari Badan Pusat Statistika (BPS), jumlah penduduk terbesar ke-empat di dunia tahun 2018 sebanyak 265.015.300 jiwa dan tercatat luas wilayah Indonesia adalah berupa daratan seluas 1.916.862,20 Km^2. Hampir sebagian besar dari penduduk Indonesia memiliki status pekerjaan dalam sektor pertanian sebesar 35.703.074 jiwa [1].

Secara kebutuhan pangan, padi menjadi salah satu jenis tanaman budidaya dan sumber daya alam yang bermanfaat bagi manusia. Padi yang ditanam oleh petani dapat menjadi beras, selanjutnya bisa dikonsumsi sebagai bahan pokok manan dalam kehidupan kita sehari-hari. Tanaman padi dapat hidup baik di daerah yang berhawa panas dan banyak mengandung uap air, serta di wilayah beriklim tropis seperti di Indonesia. Penggunaan lahan secara umum meliputi pertanian tadah hujan, pertanian beririgasi, padang rumput pengembalaan, kehutanan, daerah rekreasi dan sebagainya, sedangkan tipe penggunaan lahan adalah penggunaan lahan yang lebih detil dengan mempertimbangkan sekumpulan rincian teknis yang didasarkan pada keadaan fisik dan sosial dari satu jenis tanaman atau lebih [17].

Identifikasi dan pemetaan vegetasi di suatu area diperlukan untuk lingkup penelitian ilmu pengetahuan dan pengelolaan sumber daya alam, seperti identifikasi lahan pertanian. Manfaat lainnnya yang diperoleh mampu digunakan sebagai perhitungan suatu luasan lahan, dan perkiraan hasil panen padi terhadap luas lahan yang teridentifikasi, serta sebagai perencanaan tata ruang suatu wilayah. Namun metode yang diaplikasikan masih tradisional dilakukan dengan survei lapangan, tinjauan literatur, dan interpretasi foto manual melalui satelit menjadi tidak efektif untuk memperoleh data vegetasi karena menghabiskan banyak waktu dan seringkali memerlukan banyak biaya. Teknologi penginderaan jauh menawarkan cara praktis dan ekonomis untuk memperoleh informasi tentang tutupan vegetasi, terutama di wilayah yang luas. Karena pengamatannya yang sistematis pada berbagai skala, teknologi penginderaan jauh berpotensi memungkinkan klasifikasi dan pemetaan vegetasi pada resolusi temporal yang tinggi [2].

Di Indonesia sudah terdapat lembaga yang mengelola dan mengumpulkan data luas area vegetasi dengan peta digital lahan yang dikelola oleh Badan Pusat Statistik (BPS). Penggunaan metode estimasi produksi padi yang lebih objektif melalui Kerangka Sampel Area (KSA). KSA didefinisikan sebagai teknik pendekatan penyampelan yang menggunakan area lahan sebagai unit enumerasi. Sistem ini berbasis teknologi sistem informasi geografi (SIG), pengideraan jauh, teknologi informasi, dan statistika yang saat ini sedang diimplementasikan di Indonesia untuk perolehan data dan informasi pertanian tanaman pangan [3].

Deep learning adalah salah satu jenis teknik pembelajaran mesin (*machine learning)* menggunakan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan [4]. *Deep learning* melibatkan pembelajaran fitur dan pengklasifikasi secara bersamaan, dan menggunakan data pelatihan untuk mengkategorikan konten gambar tanpa spesifikasi dari fitur gambar. Di antara semua jaringan berbasis pembelajaran yang dalam, CNN (*convolutional neural network*) adalah yang paling populer untuk mempelajari fitur visual dalam aplikasi penglihatan komputer termasuk penginderaan jauh. CNN memiliki bobot dan beberapa lapisan tersembunyi yang disusun menjadi arsitektur. Arsitektur CNN dibagi menjadi tiga bagian, yaitu lapisan konvolusional, fungsi aktivasi (Relu), dan lapisan max pooling. Penelitian terbaru menunjukkan bahwa CNN efektif untuk beragam aplikasi [5].

Proses dalam deteksi lahan pertanian pada citra Google Earth dilakukan dengan mengidentifikasi berbagai objek yang melibatkan citra yang kompleks dan variatif, dimana terdiri dari lahan sawah, dan bukan lahan sawah, contohnya rumah, jalan, pohon, mobil, dan sebagainya. Semakin banyak jumlah objek dan variasinya membuat citra susah diprediksi dan tingkat keakuratannya semakin kecil. Selain itu, citra Google Earth diperoleh dari berdasarkan level resolusi, ketinggian gambar (*eye alt*), dan waktu pengambilannya. Dengan menggunakan metode *convolutional neural networks* (CNN) digunakan karena dapat menangani masalah yang kompleks tersebut dan memiliki kinerja yang baik dari hasil penelitian terkait.

Dengan latar belakang tersebut, penulis menyelidiki potensi untuk mengadaptasi model *deep learning* dan metode *convolutional neural network* untuk identifikasi lahan pertanian dengan citra *Google Earth*. Oleh karena itu penulis mengajukan kerja dengan judul "**IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH***".

1.2 Perumusan Masalah

Berdasarkan latar belakang maka permasalahan yang muncul adalah :

1. Bagaimana arsitektur CNN untuk melakukan identifikasi lahan pertanian pada citra *Google Earth* ?
2. Bagaimana unjuk kerja arsitektur CNN dalam melakukan pelatihan dan pengujian dalam proses identifikasi lahan pertanian ?

1.3 Batasan Masalah

Agar Tugas Akhir lebih fokus dan terarah, maka perlu adanya batasan masalah. Adapun batasan masalah dalam tugas akhir ini sebagai berikut :

1. Metode *deep learning* yang digunakan untuk lahan pertanian menggunakan CNN (*Convolutional Neural Network*).
2. Dataset gambar geografis diambil melalui layanan *Google Earth Pro*.
3. Jumlah *dataset* pada pelatihan dan validasi memiliki rasio 75:25, sedangkan pada *dataset* pengujian dilakukan hanya pada satu citra pada implementasi identifikasi lahan sawah.
4. Secara umum citra diambil melalui ketentuan, yaitu lahan sawah basah atau terlihat pada citra berwarna hijau, ketinggian 100 meter dan 300 meter,

lahan sawah dan bukan lahan sawah, resolusi dimensinya 1116x632 piksel berformat "*.jpg*" dan minimal kedalaman bit (*bit depth)* sebanyak 24 bit.

5. Pada *dataset* pelatihan dan pengujian terdapat tiga wilayah di Indonesia, berupa di pulau Jawa, Kalimantan, dan Sumatra. Lokasi spesifiknya di pulau Jawa terdapat pada Kabupaten Banyumas, di pulau Kalimantan terdapat pada Kabupaten Tanah Laut Kalimantan Selatan, dan di pulau Sumatra terdapat di Kabupaten Ogan Sumatra Selatan.
6. Pengambilan citra berdasarkan ketinggian (*eye alt*) hanya menyesuaikan ukuran dari *Google Earth* dan citra yang dikumpulkan dalam keadaan cerah maupun jelas terlihat lahan sawah tanpa menyetarakan waktu pengambilan citra di seluruh dataset.
7. Program yang dirancang menggunakan Bahasa Pemrograman *Python* dan memiliki file berformat "*.ipynb*" atau *IPython Notebooks.*
8. Antarmuka melalui *Google Colaboratory* dengan spesifikasi yang digunakan saat penelitian ini dilakukan.
9. Pembuatan model *deep learning* pada identifikasi vegetasi pertanian menggunakan Framework *Keras* dan *TensorFlow*.

1.4 Tujuan dan Manfaat

4.4.1 Tujuan

Tugas Akhir ini memiliki beberapa tujuan sebagai berikut :

1. Merancang arsitektur CNN untuk identifikasi lahan pertanian sawah.
2. Melakukan proses pelatihan dan pengujian dalam pemodelan aristektur CNN.

4.4.2 Manfaat

Manfaat yang diharapkan setelah penelitian ini dilaksanakan adalah sebagai berikut.

1. Meminimalkan kesalahan yang dilakukan oleh manusia, maupun lembaga statistika dalam mendeteksi luas lahan pertanian.
2. Mendeteksi lahan pertanian di suatu wilayah menjadi lebih akurat dengan menggunakan metode CNN dibandingkan dengan metode penelitian lain.
3. Menawarkan solusi yang lebih murah dan efiesien dalam identifikasi lahan pertanian khususnya di Indonesia, maupun area geografis di daerah lainnya.

1.5 Sistematika Penulisan

Untuk mempermudah pemahaman dan pembahasan, penulis membuat sistematika penulisan yang terdiri dari BAB I Pendahuluan, BAB II Tinjauan pustaka, BAB III Metode Penelitian, BAB IV Hasil dan Pembahasan, dan BAB V Penutup.

BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Dalam beberapa jurnal penelitian terhadulu yang membahas tentang klasifikasi objek dengan citra dapat dilihat dari beberapa jurnal sebagai berikut :

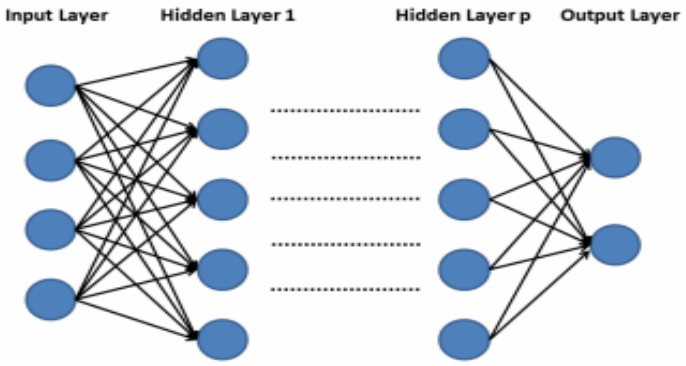
1. Dayong Wang, Aditya Khosla, Rishab Gargeya, Humayun Irshad, dan Andrew H Beck dengan penelitiannya yang berjudul Mengidentifikasi Kanker Payudara Metastik dengan *Deep Learning* membahas tentang deteksi kanker payudara yang kerangkanya terdiri dari tahap *patch-based classification* dan tahap *heatmap-based postprocessing*, dimana hasilnya menunjukan kemampuan penggunaan *deep learning* menghasilkan peningkatan yang signifikan dalam akurasi diagnosis patologis.
2. Shuntaro Watanabe, Kazuaki Sumi, dan Takeshi Ise penelitiannya yang berjudul Identifikasi Vegetasi Otomatis dengan Gambar Google Earth menggunakan metode *Convolutional Neural Network* pada Hutan Bambu di Jepang membahas tentang penggunaan deep learning dalam mengidentifikasikan vegatasi akurasi tinggi lahan hutan bamboo di beberapa wilayah Jepang, dengan klasisfikasi CNN, penelitian dilakukan dengan metode *chopped-image*.
3. Stephen Ekaputra Limantoro, Yosi Kristian, dan Devi Dwi Purwanto melakukan penelitian yang berjudul Pemanfaatan Deep Learning pada Video Dash Cam untuk Deteksi Pengendara Sepeda Motor yang membahas mengenai metode CNN yang digunakan dalam mendeteksi pengendara sepeda motor di Indonesia

dari video *dash cam* dengan *framework* yang terdiri dari *sliding window* dan *heat map*.

2.2 *Deep Learning*

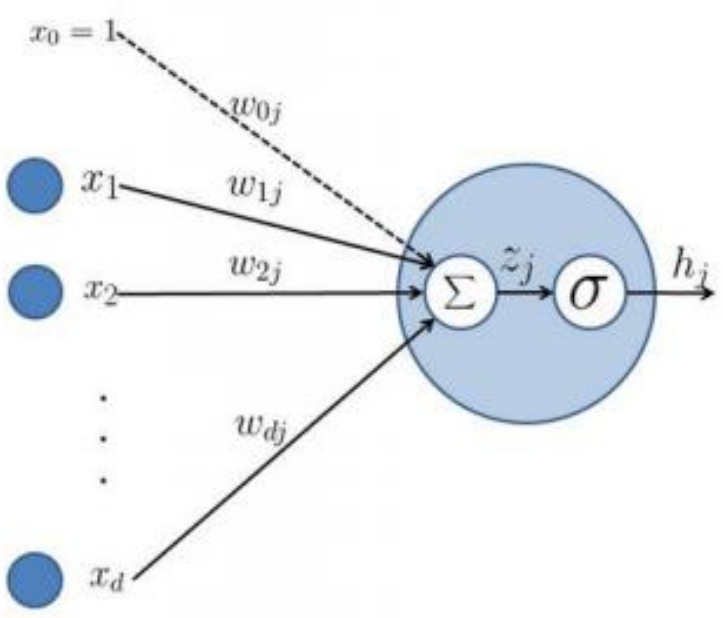
Deep Learning merupakan cabang dari Machine Learning yang terinspirasi dari kortex manusia dengan menerapkan jaringan syaraf buatan yang memiliki banyak hiden layer. Convolutional Neural Network (CNN) merupakan salah satu metode dalam *Deep Learning* yang dibuat untuk menutupi kelemahan dari metode sebelumnya. Terdapat beberapa kelemahan dalam metode sebelumnya, tetapi dengan model ini sejumlah parameter bebas dapat dikurangi dan deformasi gambar input seperti translasi, rotasi dan skala dapat ditangani [4].

Metode *deep learning* memrepresentasikan beberapa tingkat representasi, dimana representasi membentuk medan arsitektur jaringan syaraf yang berisi banyak *layer* (lapisan). Lapisan pada deep learning terdiri atas tiga bagian, yaitu input layer, hidden layer, dan output layer. Pada hidden layer dapat dibuat berlapis-lapis untuk menemukan komposisi algoritma yang tepat agar meminimalisir error pada output.



Gambar 2. 1 Layer-layer pada Deep Learning[4]

Gambar 2.1 mengilustrasikan layer-layer pada deep learning yang memiliki p + 2 *layer* (p *hidden layer*, 1 *input* dan 1 *output layer*). Bulatan berwarna biru menggambarkan neuron. Disetiap lapis *hidden layer* terdapat satu atau lebih neuron. Neuron-neuron tersebut akan terhubung langsung dengan neuron lain pada layer selanjutnya. Koneksi antar neuron hanya terjadi di antara 2 buah *layer* (input dan output) tidak ada koneksi pada layer yang sama walaupun secara teknis bisa saja dibuat dan juga *fully connected.*



Gambar 2. 2 Sebuah Perceptron Dengan 4 Buah Input[6]

Sebuah sistem yang terdiri dari sebuah neuron beserta input dan output nya disebut sebagai perceptron. Seperti yang dilihat pada gambar 2.2 sebuah neuron diperbesar disimbolkan sebagai hj yang menerima d buah input $x1 , \ldots , xd \in R$ dapat berasal dari data ataupun ouput dari lapisan sebelumnya. Sedangkan x0 tidak dianggap sebagai input (atau sebagai *dummy*) dan selalu bernilai 1. Variabel-variabel $w1j , \ldots , wdj \in R$ merupakan bobot/*weights* dari koneksi input ke neuron

hj dan w0j dinamakan sebagai bias. Bobot adalah koneksi antar lapisan yang berupa nilai yang menentukan fungsi input-output dari mesin. Bobot adalah parameter penyesuaian yang diatur oleh mesin untuk mengukur kesalahan antara nilai output dan pola nilai yang dinginkan pada pembelajaran. Sehingga bobot inilah yang diatur oleh mesin untuk mengurangi kesalahan yang terjadi. Dalam sistem deep learning kemungkinan terdapat ratusan juta bobot yang dapat diatur. Untuk menyesuaikan vektor bobot dengan benar, algoritma menghitung vektor gradien untuk setiap bobot berdasarkan jumlah kesalahan yang meningkat atau menurun jika bobot meningkat dalam jumlah kecil [6].

Operasi pada sebuah perceptron merupakan dua buah operasi terpisah yaitu operasi linear ∑ dan operasi *non-linear*/aktivasi σ (2.2). Operasi linear dengan nilai bobot yang ada, kemudian hasil komputasi dari operasi linear akan ditransformasi menggunakan operasi non-linear yang disebut sebagai fungsi aktivasi. Berikut ini dapat dilihat persamaan operasi linear ∑ [2.1] dan operasi non-linear/aktivasi σ [2.2], sebagai berikut :

$$z_j = \sum_{i=0}^{d} w_{ij}\, x_i \qquad (2.1)$$

Dengan :

$z_j =$ output unit

$w =$ bobot unit

$x =$ input unit ($x_i, i = 1, \dots, n$)

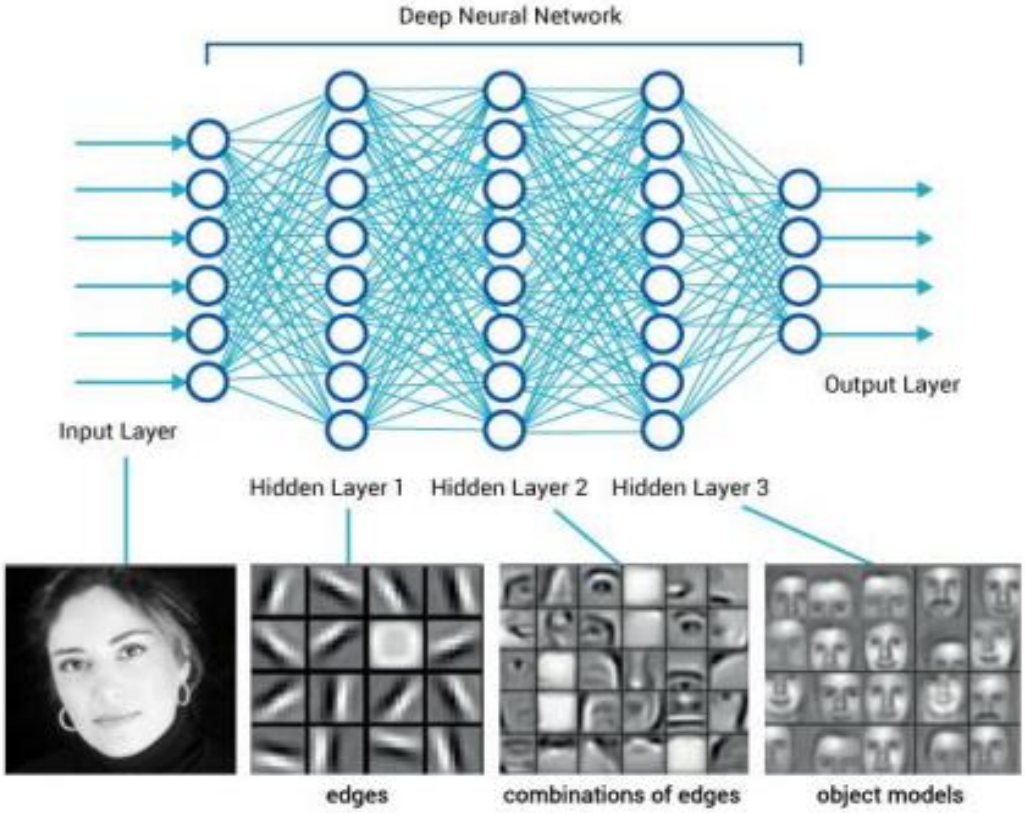
$$h_j = \sigma(z_j) \qquad (2.2)$$

Dengan :

$h_j =$ operasi aktivasi/non-linear

$z_j =$ output unit

Pada operasi aktivasi/non-linear $\sigma : R \rightarrow R$ terdapat berbagai macam fungsi aktivasi yang dapat diimplementasikan pada hidden neuron. Di dalam penelitian ini digunakan salah satu fungsi aktivasi yaitu *softmax layer*. *Softmax layer* digunakan apabila pada permasalahan *multiclass classification*, *output layer* biasanya memiliki lebih dari satu neuron.



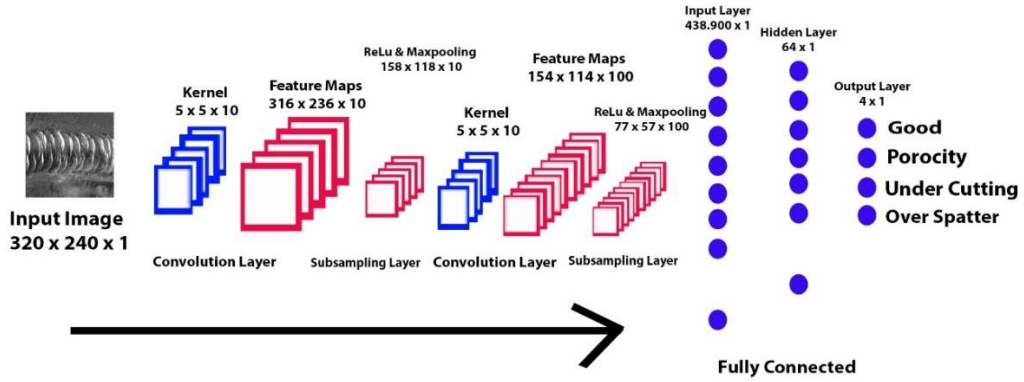
Gambar 2. 3 Visual Object Recognition Pada Deep Learning[6]

Pada Gambar 2.3 menampilkan hasil pengenalan objek secara visual pada deep learning yang terdiri dari beberapa layer yang mengkombinasikan suatu *Deep Neural Network,* yaitu terdapat *input layer*, *hidden layer*, dan *output layer*.

2.3 *Convolutional Neural Network* (*CNN*)

Convolutional Neural Network (CNN) adalah salah satu algoritma dari deep learning yang merupakan pengembangan dari *Multilayer Percepton* (MLP) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN digunakan untuk mengklasifikasi data yang terlabel dengan menggunakan metode *supervised learning*, yang mana cara kerja dari supervised learning adalah terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini adalah mengkelompokan suatu data ke data yang sudah ada.

Lapisan-lapisan CNN memiliki susunan neuron 3 dimensi (lebar, tinggi, kedalaman). Lebar dan tinggi merupakan ukuran lapisan sedangkan kedalaman mengacu pada jumlah lapisan. Sebuah CNN dapat memiliki puluhan hingga ratusan lapisan yang masing-masing belajar mendeteksi berbagai gambar. Pengolahan citra diterapkan pada setiap citra latih pada resolusi yang berbeda, dan output dari masing-masing gambar yang diolah dan digunakan sebagai input ke lapisan berikutnya. Pengolahan citra dapat dimulai sebagai fitur yang sangat sederhana, seperti kecerahan dan tepi atau meningkatkan kompleksitas pada fitur yang secara unik menentukan objek sesuai ketebalan lapisan [8].

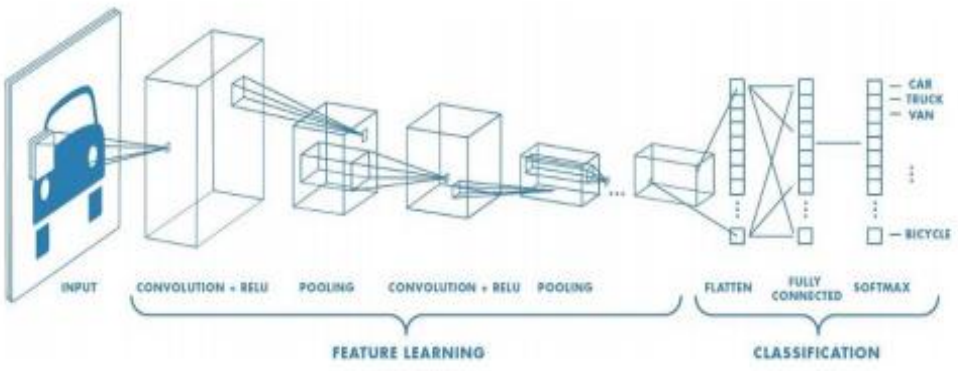


Gambar 2. 4 Contoh Jaringan CNN[8]

Pada gambar 2.4 yaitu di setiap lapisan input yang dimasukkan mempunyai susunan neuron 3 dimensi, yaitu lebar, tinggi, dan kedalaman). Lebar dan tinggi yaitu ukuran lapisan, sedangkan untuk ke dalam yaitu mengacu pada jumlah lapisan. Setiap besaran yang didapat terhantung dari hasil fitrasi dari lapisan sebelumnya dan banyaknya filter yang digunakan. Model jaringan seperti ini sudah terbukti efektif dalam menangani permasalahan klasifikasi citra. Sebuah CNN mampu memiliki puluhan hingga ratusan lapisan yang masing-masing lapusan mempelajari deteksi berbagai gambar. Pengolahan citra diterapkan pada setiap citra latih pada resolusi yang berbeda, dan output dari masing-masing data gambar yang diolah dan digunakan sebagai input ke lapisan berikutnya. Pengolahan citra dapat dimulai sebagai fitur yang sederhana, seperti ukuran kecerahan dan tepi atau meningkatkan kekompleksan pada fitur secara unik untuk menentukan objek sesuai ketebalan lapisan [8].

2.3.1 Arsitektur *Convolution Neural Network*

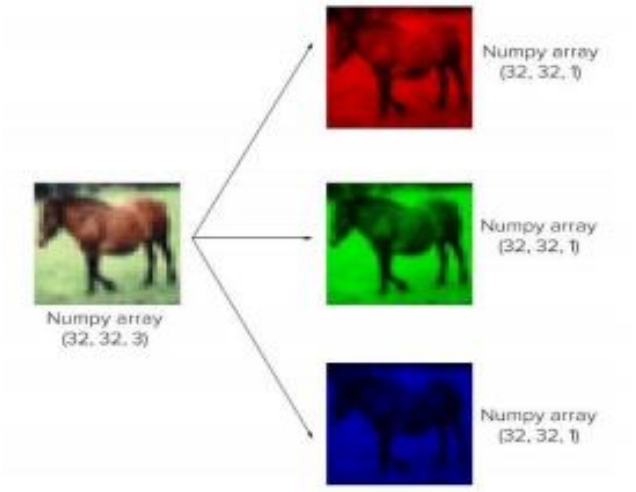
Arsitektur dari CNN dibagi menjadi 2 bagian besar, *Feature Extraction Layer* dan *Fully-Connected Layer* (MLP). Berikut ini diperlihatkan pada Gambar 2.5 yang merupakan arsitektur CNN.



Gambar 2. 5 Arsitektur CNN

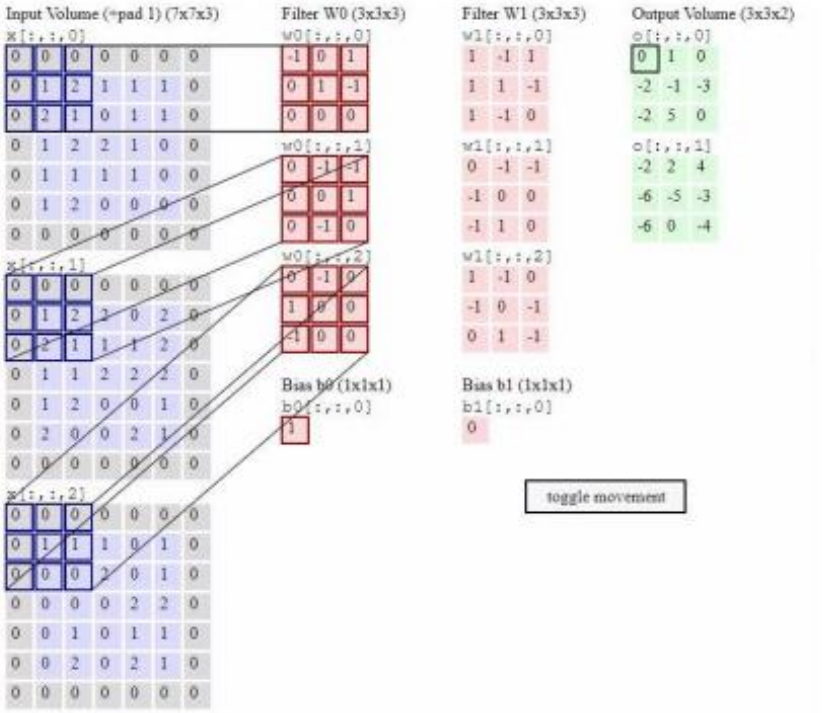
1. Feature Extraction Layer

Proses yang terjadi pada bagian ini adalah melakukan "*encoding*" dari sebuah image menjadi features yang berupa angka-angka yang merepresentasikan image tersebut (*Feature Extraction*). Feature extraction layer terdiri dari dua bagian. *Convolutional Layer* dan *Pooling Layer.* Namun kadang ada beberapa riset/paper yang tidak menggunakan pooling[9].



Gambar 2. 6 Image RGB[9]

Pada Gambar 2.6 diatas adalah RGB (*Red, Green, Blue*) *image* berukuran 32x32 pixels yang sebenarnya adalah multidimensional array dengan ukuran 32x32x3 (3 adalah jumlah channel). *Convolutional layer* terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (*pixels*). Sebagai contoh, layer pertama pada *feature extraction layer* biasanya adalah convolution layer dengan ukuran 5x5x3. Panjang 5 pixels, tinggi 5 pixels dan tebal atau jumlah 3 buah sesuai dengan channel dari image tersebut. Ketiga filter ini akan digeser keseluruh bagian dari gambar. Setiap pergeseran akan dilakukan operasi “dot” antara input dan nilai dari filter tersebut sehingga menghasilkan sebuah output atau biasa disebut sebagai *activation map* atau *feature map*.



Gambar 2. 7 Feature Map[9]

Stride adalah parameter yang menentukan berapa jumlah pergeseran filter. Jika nilai stride adalah 1, maka conv. filter akan bergeser sebanyak 1 pixels secara *horizontal* lalu *vertical*. Pada gambar 2.7 diatas, *stride* yang digunakan adalah 2. Semakin kecil stride maka akan semakin detail informasi yang kita dapatkan dari sebuah input, namun membutuhkan komputasi yang lebih jika dibandingkan dengan *stride* yang besar. Namun perlu diperhatikan bahwa dengan menggunakan *stride* yang kecil kita tidak selalu akan mendapatkan performa yang bagus. *Padding* atau *Zero Padding* adalah parameter yang menentukan jumlah *pixels* (berisi nilai 0) yang akan ditambahkan di setiap sisi dari input. Hal ini digunakan dengan tujuan untuk memanipulasi dimensi output dari *conv. layer* (*Feature Map*). Tujuan dari penggunaan padding adalah dimensi output dari *conv. layer* selalu lebih kecil dari inputnya (kecuali penggunaan 1x1 filter dengan stride

1). Output ini akan digunakan kembali sebagai input dari *conv. layer* selanjutnya, sehingga makin banyak informasi yang terbuang. Dengan menggunakan padding, kita dapat mengatur dimensi output agar tetap sama seperti dimensi input atau setidaknya tidak berkurang secara drastis. Sehingga kita bisa menggunakan *conv. layer* yang lebih dalam sehingga lebih banyak features yang berhasil di-extract. *Padding* juga mampu meningkatkan performa dari model, karena conv. filter akan fokus pada informasi yang sebenarnya yaitu yang berada diantara zero padding tersebut. Pada ilustrasi diatas, dimensi dari input sebenarnya adalah 5x5, jika dilakukan *convolution* dengan filter 3x3 dan stride sebesar 2, maka akan didapatkan feature map dengan ukuran 2x2. Namun jika kita tambahkan zero padding sebanyak 1, maka feature map yang dihasilkan berukuran 3x3 (lebih banyak informasi yang dihasilkan). Untuk menghitung dimensi dari feature map kita bisa gunakan rumus seperti dibawah ini:

$$Output = \frac{W+N+2P}{S} + 1 \qquad (2.3)$$

Dengan :

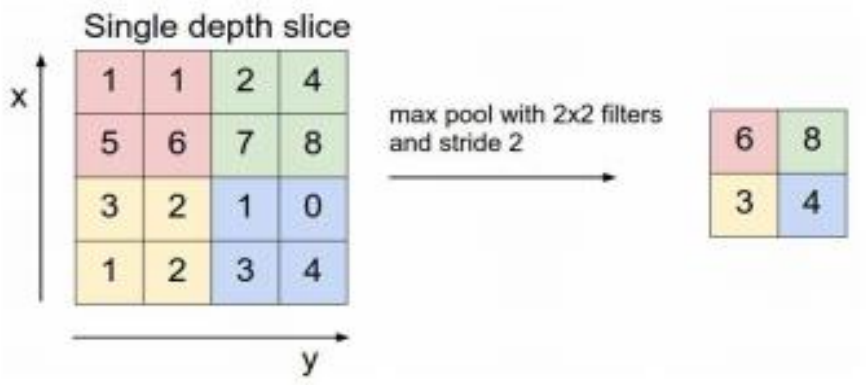
W = Panjang/Tinggi Input

N = Panjang/Tinggi Filter

P = *Zero Padding*

S = *Stride*

Pooling layer biasanya berada setelah *convolution layer*. Pada *prinsipnya pooling layer* terdiri dari sebuah filter dengan ukuran dan stride tertentu yang akan bergeser pada seluruh area feature map, seperti yang terlihat pada Gambar 2.8. Pooling yang biasa digunakan adalah *Max Pooling* dan *Average Pooling.* Sebagai contoh jika kita menggunakan *Max Pooling* 2x2 dengan stride 2, maka pada setiap pergeseran filter, nilai maximum pada area 2x2 *pixel* tersebut yang akan dipilih, sedangkan *Average Pooling* akan memilih nilai rata-ratanya.



Gambar 2. 8 Max Pooling[9]

Tujuan dari penggunaan *pooling layer* adalah mengurangi dimensi dari *feature map* (*downsampling*), sehingga mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting*[9].

2. *Fully-Connected Layer* (FC Layer)

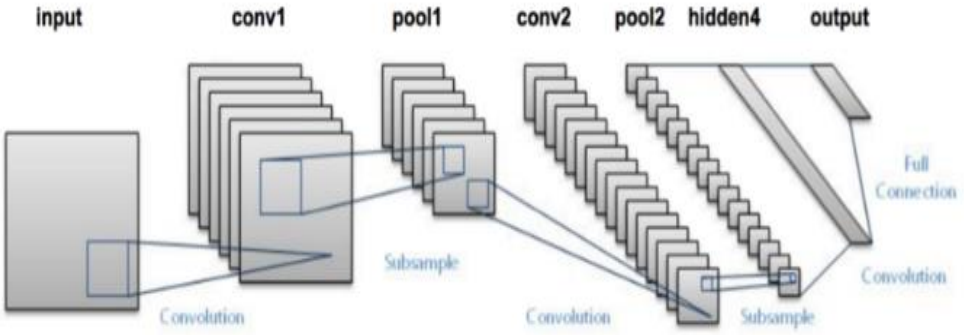
Feature map yang dihasilkan dari feature extraction layer masih berbentuk multidimensional array, sehingga harus melakukan "*flatten*" atau reshape feature map menjadi sebuah vector agar bisa digunakan sebagai input *dari fully-connected layer*[10].

Lapisan *Fully-Connected* adalah lapisan di mana semua neuron aktivasi dari lapisan sebelumnya terhubung semua dengan neuron di lapisan selanjutnya seperti halnya jaringan saraf tiruan biasa. Setiap aktivasi dari lapisan sebelumnya perlu diubah menjadi data satu dimensi sebelum dapat dihubungkan ke semua neuron di lapisan.

Lapisan *Fully-Connected* biasanya digunakan pada metode Multi lapisan Perceptron dan bertujuan untuk mengolah data sehingga bisa diklasifikasikan. Perbedaan antara lapisan *Fully-Connected* dan lapisan konvolusi biasa adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada input, sementara lapisan *Fully-Connected* memiliki neuron yang secara keseluruhan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan produk dot, sehingga fungsinya tidak begitu berbeda.

2.3.2 Arsitektur *LeNet-5*

LeNet-5 adalah salah satu arsitektur paling sederhana dalam CNN yang memiliki 2 *convolution layer* dan 3 fully-connected *layer*. Seperti halnya dengan arsitektur CNN lainnya, LeNet-5 dirancang untuk mengenali pola visual langsung dari gambar piksel dengan preprocessing minimal. Rata-rata lapisan *max pooling* seperti yang dikenali sekarang disebut lapisan *sub-sampling* dan memiliki bobot yang dapat dilatihkan. Arsitektur ini memiliki sekitar 60.000 parameter [6]. Pada Gambar 2.9 diperlihatkan arsitektur CNN dalam *LeNet-5*, seperti berikut ini.

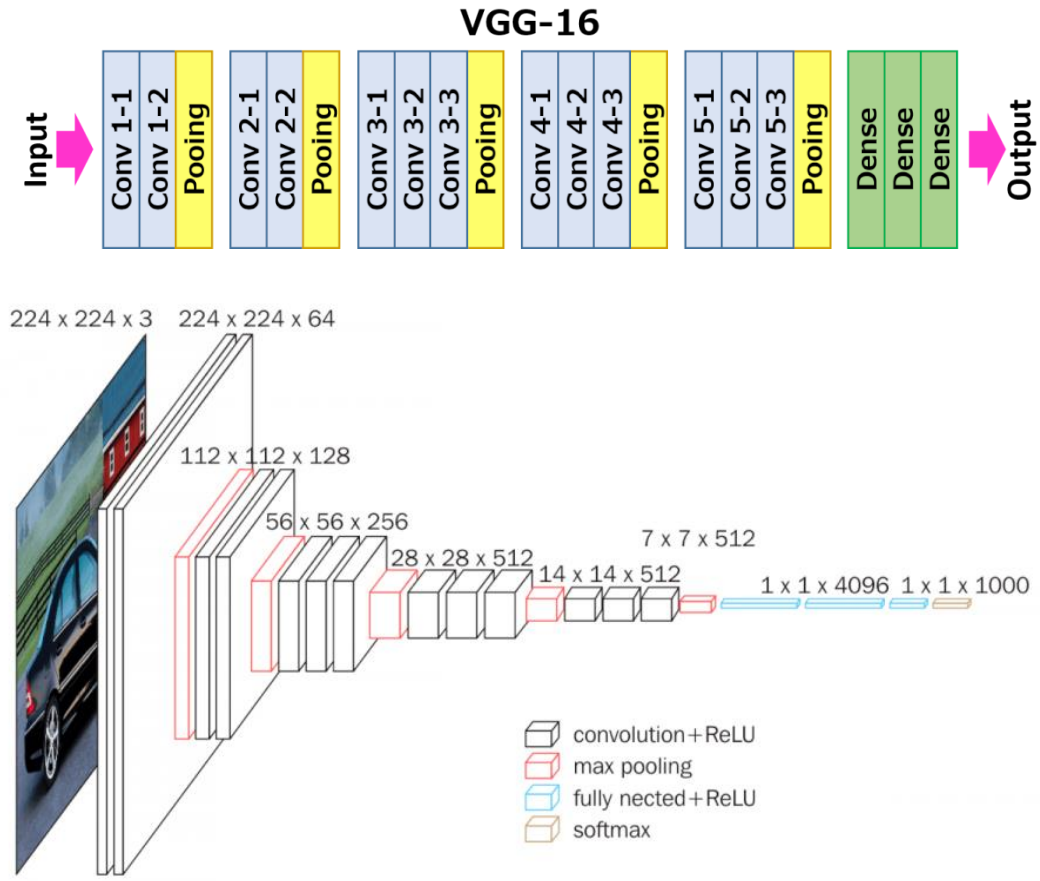


Gambar 2. 9 Arsitektur LeNet-5[6]

Aristektur *LeNet-5* dirancang oleh LeCun pada tahun 1998, yang mengklasifikasikan digit, diterapkan oleh beberapa bank untuk mengenali nomor tulisan tangan pada *checking* yang didigitalkan dalam input-input skala *grayscale* 32x32 pixel. Kemampuan untuk memproses gambar dengan resolusi lebih tinggi membutuhkan lapisan yang lebih besar dan lebih convolutional, sehingga teknik ini dibatasi oleh ketersediaan sumber daya komputasi.

2.3.3 Arsitektur *VGG-16Net*

VGG-16 Network adalah arsitektur *Convolutional Neural Network* yang dirancang oleh Karen Simonyan dan Andrew Zisserman dari *Visual Geometry Group, Department of Engineering Science, University of Oxford*. Arsitektur ini dibuat untuk mengikuti kompetisi *ImageNet Challenge* 2014 dan berhasil meraih peringkat teratas untuk *localization* dan *classification*. Input yang digunakan berupa RGB image berukuran 224 × 224 pixels. *Convolutional layer* yang digunakan dalam arsitektur ini ada 2 jenis, yaitu *convolutional layer* dengan ukuran filter 3x3 (*conv3*) dan ukuran filter 1x1 (*conv1*). Ukuran *convolutional layer* yang digunakan bermacam-macam, yaitu 64x64, 128x128, 256x256, dan 512x512[20].



Gambar 2. 10 Arsitektur VGG-16Net[20]

Masukan ke lapisan *conv1* berukuran tetap 224x224 gambar RGB. Gambar dilewatkan melalui tumpukan lapisan konvolusional (*conv*.), dimana *filter* digunakan dengan bidang reseptif yang sangat kecil sebesar 3×3. Dalam salah satu konfigurasi, ia juga menggunakan *filter* konvolusi 1×1, yang dapat dilihat sebagai transformasi linear dari saluran masukan (diikuti oleh non-linearitas). Langkah (*stride*) konvolusi ditetapkan pada 1 piksel, spasial *padding* dari masukan lapisan konvolusi sedemikian rupa sehingga resolusi spasial/ukuran tersebut dipertahankan setelah konvolusi, yaitu *padding* sebesar 1 piksel untuk 3×3 lapisan konvolusi. Spasial *pooling* dilakukan oleh lima lapisan *max-pooling*, yang mengikuti beberapa

lapisan konvolusi (tidak semua lapisan konv diikuti oleh *max-pooling*). *Max-pooling* dilakukan dengan ukuran piksel 2×2, dan 2 *stride*.

Pada tiga lapisan *Fully-Connected* (FC) mengikuti tumpukan lapisan konvolusional (yang memiliki kedalaman berbeda dalam arsitektur yang berbeda), yaitu dua yang pertama memiliki ukuran masing-masing 4096 kanal, yang ketiga melakukan klasifikasi ILSVRC 1000 keluaran dan dengan demikian memuat 1000 saluran (satu untuk setiap kelas). Lapisan terakhir adalah lapisan *soft-max*. Konfigurasi *fully connected layers* adalah sama di semua jaringan.

Setiap *hidden layers* dilengkapi dengan fungsi aktivasi ReLU. Juga dicatat bahwa tidak ada jaringan (kecuali satu) yang berisi Normalisasi Respon Lokal (LRN), normalisasi seperti itu tidak meningkatkan kinerja pada *dataset* ILSVRC, tetapi mengarah pada peningkatan konsumsi memori dan waktu komputasi.

2.3.4 Operasi Konvolusi

Operasi konvolusi adalah operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi output sebagai *Feature Map* dari input citra. Input dan output ini dapat dilihat sebagai dua argumen bernilai riil. Secara formal operasi konvolusi dapat ditulis dengan rumus pada persamaan [2.4] berikut ini :

$$s(t) = (x * w)(t) \qquad (2.4)$$

Dengan :

s(t) = output tunggal (*Feature Map)*

x = input argument pertama

w = kernel atau filter argumen kedua

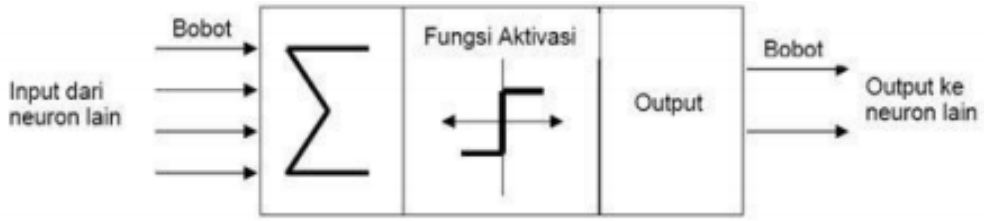
t = *pixel*

Sebagai alternatif, operasi konvolusi dapat dilihat sebagai 30 perkalian matriks antara citra masukan dan kernel dimana keluarannya dapat dihitung dengan *dot product* [11].

2.4 Konsep *Neural Network*

4.4.1 Struktur *Neural Network*

Ide mendasar dari *Artificial Neural Network* (ANN) adalah mengadopsi mekanisme berpikir sebuah sistem atau aplikasi yang menyerupai otak manusia, baik untuk pemrosesan berbagai sinyal elemen yang diterima, toleransi terhadap kesalahan/error, dan juga *parallel processing*[12].



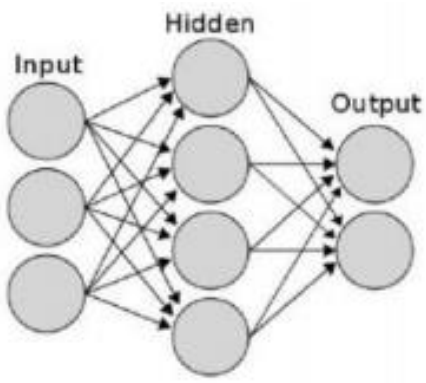
Gambar 2. 11 Struktur Artificial Neural Network[12]

Pada Gambar 2.11 merupakan struktur ANN, dimana proses pada ANN dimulai dari input yang diterima oleh neuron beserta dengan nilai bobot dari tiap-tiap input yang ada. Setelah masuk ke dalam neuron, nilai input yang ada akan dijumlahkan oleh suatu fungsi perambatan (summing function), yang bisa dilihat seperti pada di gambar dengan lambang sigma (∑). Hasil penjumlahan akan diproses oleh fungsi aktivasi setiap neuron, disini akan dibandingkan hasil penjumlahan dengan *threshold* (nilai ambang) tertentu. Jika nilai melebihi

threshold, maka aktivasi neuron akan dibatalkan, sebaliknya, jika masih dibawah nilai *threshold*, neuron akan diaktifkan. Setelah aktif, neuron akan mengirimkan nilai output melalui bobot-bobot outputnya ke semua neuron yang berhubungan dengannya. Proses ini akan terus berulang pada input-input selanjutnya.

4.4.2 Arsitektur Jaringan Syaraf Tiruan

Secara umum, Arsitektur JST terdiri atas beberapa lapisan, yaitu lapisanmasukan (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan keluaran (*output layer*). Masing-masing lapisan mempunyai jumlah node atau neuron yang berbeda-beda. Arsitektur JST tersebut dapat diilustrasikan pada Gambar 2.12 berikut ini :



Gambar 2. 12 Arsitektur Jaringan Syaraf Tiruan[12]

1. Lapisan Masukan (*input layer*)

   Lapisan masukan merupakan lapisan yang terdiri dari beberapa neuron yang akan menerima sinyal dari luar dan kemudian meneruskan ke neuron-

neuron lain dalam jaringan. Lapisan ini dillhami berdasarkan cirri-ciri dancara kerja sel-sel saraf sensori pada jaringan saraf biologi. Linear Function

2. Lapisan tersembunyi (*hidden layer*)

   Lapisan tersembunyi berfungsi meningkatkan kemampuan jaringan dalam memecahkan masalah. Konsekuensi dari adanya lapisan ini adalah pelatihan menjadi makin sulit atau lama.

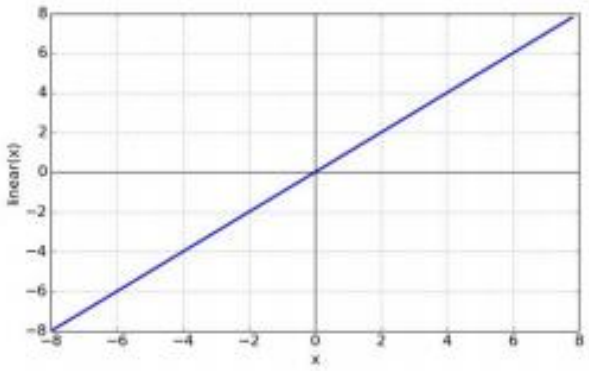
3. Lapisan keluaran (*output layer*)

   Lapisan keluaran berfungsi menyalurkan sinyal-sinyal keluaran hasil pemrosesan jaringan. Lapisan ini juga terdiri dair sejumlah neuron. Lapisan keluaran merupakan tiruan dari sel saraf motor pada jaringan saraf biologis.

4.4.3 Fungsi Aktivasi (*Function Activation*)

Sesuai dengan namanya, activation function befungsi untuk menentukan apakah neuron tersebut harus “aktif” atau tidak berdasarkan dari weighted sum dari input. Secara umum terdapat 2 jenis activation function, Linear dan Non-Linear Activation function [13].

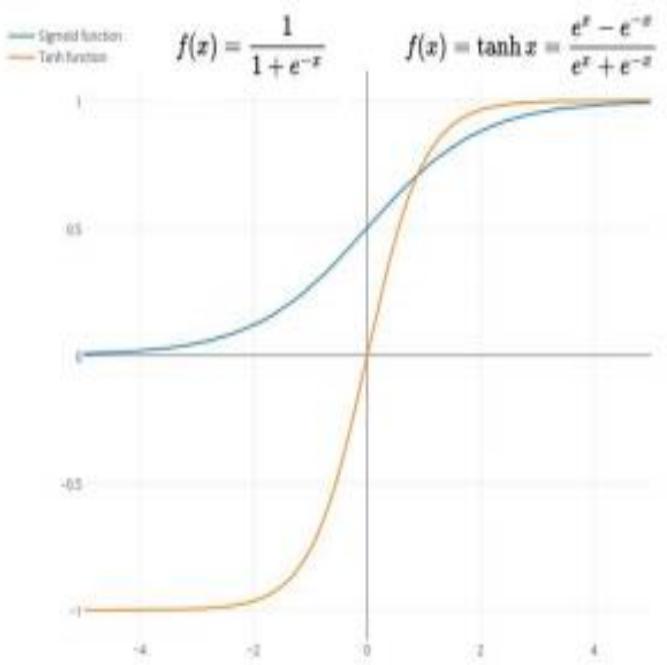
1. Linear Function



Gambar 2. 13 Linear Function[13]

Pada Gambar 2.13 merupakan fungsi aktifasi yang bisa dikatakan secara "*default*" activation function dari sebuah neuron adalah Linear. Jika sebuah neuron menggunakan linear function, maka keluaran dari neuron tersebut adalah *weighted sum* dari input + bias.

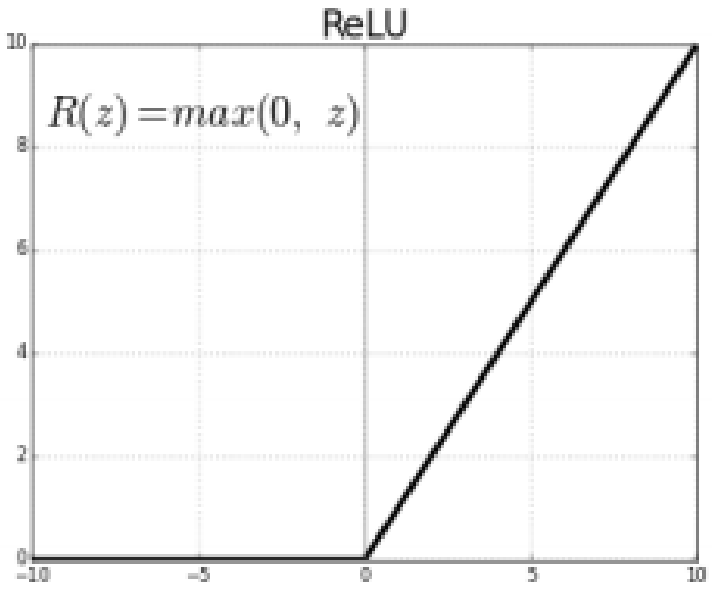
2. *Sigmoid and Tanh Function* (*Non-Linear*)



Gambar 2. 14 Sigmoid and Tanh Function[13]

Sigmoid function mempunyai rentang antara 0 hingga 1 sedangkan rentang dari Tanh adalah -1 hingga 1, seperti yang ditunjukan pada Gambar 2.14. Kedua fungsi ini biasanya digunakan untuk klasifikasi 2 class atau kelompok data.

3. ReLU (*Non-Linear)*



Gambar 2. 15 ReLU (Non-Linear)[13]

Pada Gambar 2.15 merupakan fungsi aktifasi pada ReLU melakukan "*treshold*" dari 0 hingga infinity. ReLU juga dapat menutupi kelemahan yang dimiliki oleh *Sigmoid dan Tanh*. Sebenarnya masih banyak activation function yang lain, namun beberapa fungsi yang disebutkan diatas merupakan fungsi yang sering digunakan [13].

2.5 Metode *Sliding Window*

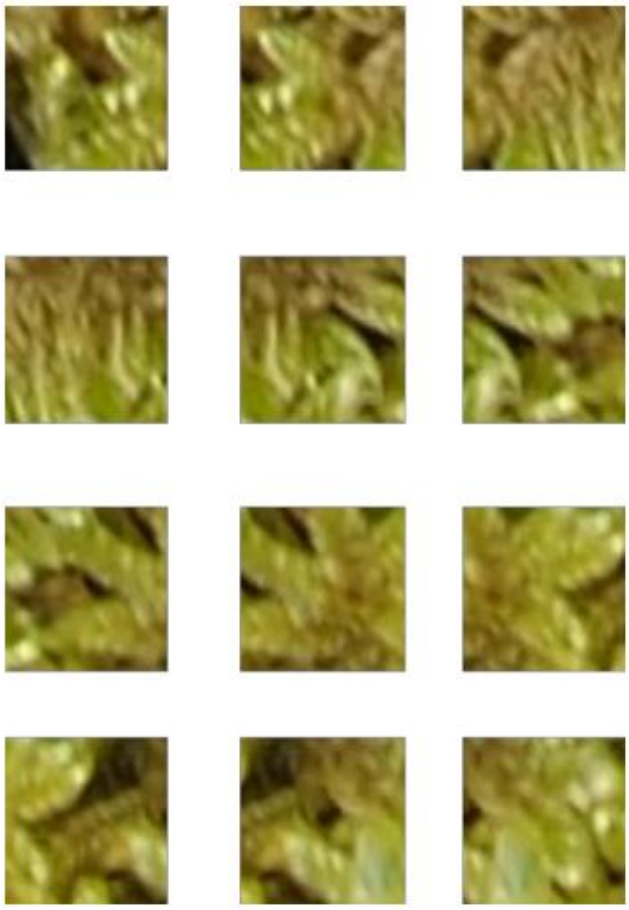
Sliding Window merupakan sebuah metode yang paling umum digunakan dalam pemrosesan citra digital. Proses ini digunakan untuk mempermudah dalam mengestraksi sebuah fitur secara lokal maupun untuk membentuk sebuah informasi. Dengan kata lain, *sliding window* adalah sebuah metode yang menggunakan window yang bergeser sebesar ukuran perpindahannya untuk mempermudah atau memperkecil komputasi yang dilakukan. *Sliding window* ini terdiri dari dua

parameter yakni ukuran *window* atau *window size* dan *displacement size* atau besara perpindahan. Keduanya saling berpengaruh satu sama lain. Jika ukuran perpindahan kurang dari ukuran window maka akan terjadi tumpang-tindih antar *window*. Hal ini biasa disebut dengan *overlapping sliding window*. Terkadang *overlapping sliding window* ini dapat menghasilkan fitur yang lebih kuat dibanding dengan yang tidak tumpang-tindih [18].

2.6 Metode Pencacahan Citra (*Chopping Picture Methode*)

Metode pencacahan citra atau yang disebut *chopping picture methode* (CPM) merupakan metode yang digunakan untuk mengidentifikasi objek yang tergolong berukuran kecil, dan secara umum tidak dikenali melalui pengelihatan manusia. Berbeda halnya dengan pengaplikasian *deep learning* sangat hanya efektif untuk mendeteksi dan identifikasi objek yang sebagian besar diterapkan untuk objek yang khas dan deterministik seperti wajah manusia, tubuh manusia, kucing, anjing, mobil, dan sebaginya. Metode CPM dapat diterapkan untuk pengamatan visual tanaman dalam berbagai skala, seperti fotografi lapangan (dalam penelitian ini), fotografi drone, fotografi pesawat terbang seperti Google Earth, dan gambar satelit [19].

Metode ini dirancang oleh peneliti asal Jepang yang membahas mengenai, lumut (*bryophytes*), sejenis tanaman hijau yang menjadi objek sasaran untuk identifikasi. Dengan cara membedah 60 gambar menjadi banyak kotak kecil dan efisien menghasilkan gambar pelatihan. Pada metode ini peneliti menyatakan dengan benar mengklasifikasikan tiga spesies lumut dan objek "non-62 lumut" dalam gambar uji dengan akurasi lebih dari 90%.

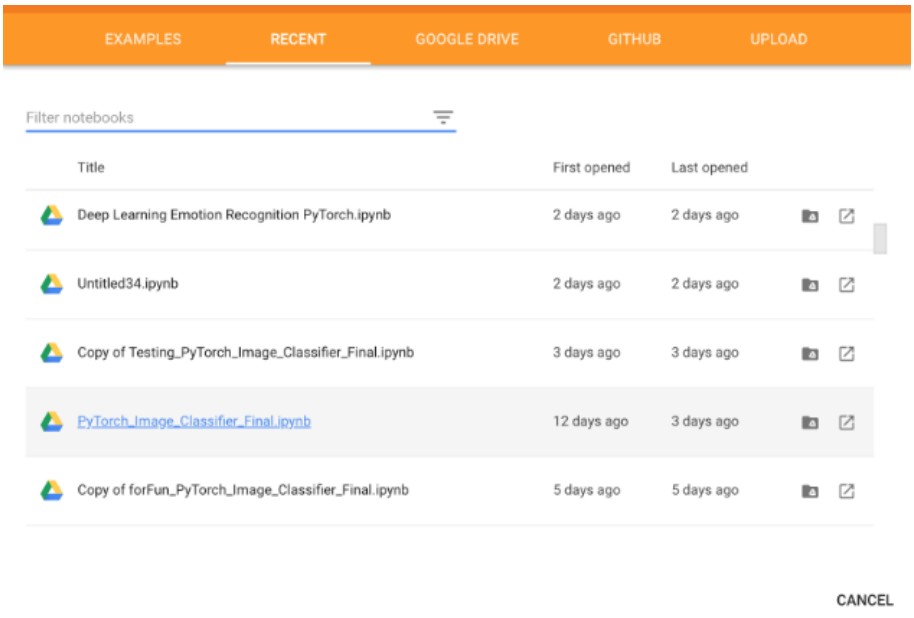


Gambar 2. 16 Hasil CPM beresolusi 56x56 piksel[19]

Pada Gambar 2.16 diatas merupakan hasil pemotongan citra menjadi kotak kecil beresolusi 56 × 56 piksel dengan 50% tumpang tindih baik secara vertikal maupun horizontal, dimana ukuran awal memiliki ukuran piksel 4608 × 3456. Dengan metode ini, itu sangat mudah untuk mendapatkan banyak citra yang lebih spesifik karena kita hanya perlu beberapa foto digital induknya menjadi citra.

2.7 *Google Colaboratory*

Google Colaboratory atau yang dikenal *Colaboratory* adalah sebuah layanan gratis dalam lingkup Jupyter notebook yang tidak memerlukan pengaturan dan sepenuhnya berjalan di cloud [14]. Dengan Colaboratory pengguna dapat menulis dan mengeksekusi kode, menyimpan dan membagikan analisis, serta mengakses sumber daya komputasi yang kuat, dimana semuanya gratis dari browser pengguna. Dalam sebagian besar hal, penggunaan *Colaboratory* hampir sama seperti yang dilakukan pada instalasi *desktop Jupyter Notebook* dan dihosting di Colaboratory. Ini bukan halaman statis, tetapi lingkungan interaktif yang memungkinkan Anda menulis dan mengeksekusi kode dalam Python dan bahasa lain.



Gambar 2. 17 Tampilan dalam Membuka file[14]

Salah satu fitur dalam *Colaboratory* ini, dapat membuka file terbaru, file dari Google Drive Anda, file GitHub, dan dapat mengunggah notebook di halaman colab, seperti yang terlihat pada Gambar 2.17 diatas.

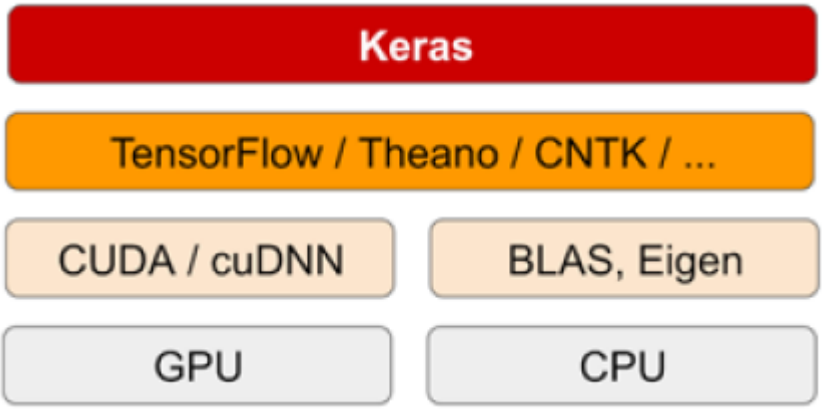
2.8 Keras dan *TensorFlow*

Keras adalah jaringan saraf tingkat tinggi memiliki *Application Programming Interface* (API), dalam bahasa pemrograman *Python* dan mampu berjalan di atas TensorFlow, CNTK, atau Theano. *Framework* ini dikembangkan dengan fokus pada memungkinkan eksperimen cepat. Mampu beralih dari ide ke hasil dengan penundaan sesedikit mungkin menjadi kunci untuk melakukan penelitian yang baik [15].

Keras digunakan apabila pengguna membutuhkan *deep learning library*, yaitu :

- Memungkinkan pembuatan prototipe yang mudah dan cepat (melalui keramahan pengguna, modularitas, dan ekstensibilitas).
- Mendukung jaringan convolutional dan jaringan berulang, serta kombinasi keduanya.
- Mampu berjalan lancar di CPU dan GPU.

Keras didistribusikan di bawah lisensi MIT permisif, yang berarti dapat digunakan secara bebas dalam proyek komersial. *Framework* ini kompatibel dengan versi *Python* apa pun mulai 2,7 hingga 3,6.



Gambar 2. 18 Susunan perangkat lunak dan perangkat keras deep learning[15]

Pada Gambar 2.18 merupakan susunan perangkat lunak dan perangkat keras dari *deep learning*. Melalui *TensorFlow* (atau *Theano*, atau CNTK), *Keras* dapat berjalan dengan mulus di CPU dan GPU. Saat berjalan pada CPU, *TensorFlow* sendiri melibatkan *library* tingkat rendah untuk operasi *tensor* yang disebut *Eigen*. Pada GPU, *TensorFlow* melibatkan *library* operasi *deep learning* yang dioptimalkan dengan baik yang disebut *Library NVIDIA CUDA Deep Neural Network* (cuDNN).

2.9 Google Earth Pro

Google Earth Pro adalah alat yang sangat baik untuk membuat dan mendistribusikan visualisasi data geografis[16]. Pengguna akan belajar cara menggunakan *Google Earth* untuk melampirkan informasi penting ke fitur pengguna termasuk foto, video, tautan ke file PDF dan situs web, teks, dan banyak lagi. Kemudian juga akan belajar cara membuat peta tematik, peta 3D, alamat *geocode*, membuat profil elevasi dan *viewsheds*, menambahkan citra sendiri, menggunakan citra historis dari berbagai lokasi, dan lain-lain.

Gambar 2. 19 Tampilan Software Google Earth Pro[16]

Pada Gambar 2.19 memperlihatkan tampilan utama pada *Google Earth Pro*. Ada banyak alasan mengapa ingin memvisualisasikan data GIS yang ada di *Google Earth.* Mungkin pengguna menyukai kenyamanan menggunakan peta dasar bawaan dan foto udara yang disediakan oleh *Google Earth*. Ini dapat dengan mudah dilakukan dengan menempatkan file format *Google Earth* di situs web yang dapat diakses publik dan memungkinkan Google untuk memasukkannya dalam hasil pencarian. Karena semua data pengguna untuk proyek *Google Earth* tertentu dapat dimuat dalam satu file, juga mudah untuk didistribusikan ke banyak pengguna. Kami akan mengeksplorasi masing-masing alasan ini secara lebih rinci di bagian kursus ini dan pengguna akan belajar tentang sejumlah alat yang dapat digunakan untuk dengan mudah memindahkan data *user* dari ArcGIS ke *Google Earth*. Ekstensi *Arc2Earth* adalah alat yang luar biasa untuk menyelesaikan tugas ini sehingga akan membahas fungsionalitas ArcGIS ini secara terperinci.

Google Earth dan *Google Earth Pro* juga dapat digunakan untuk dengan mudah menambahkan semua data pengguna ke peta dan melampirkan informasi penting termasuk foto, video, tautan ke file PDF eksternal dan presentasi, dan banyak lagi. *Google Earth* juga dapat digunakan untuk memperoleh citra historis suatu situs, memperoleh informasi terain termasuk *viewsheds* dan profil ketinggian, membuat bidang tanah, mendapatkan informasi pengukuran, dan banyak lagi.

BAB 3
METODOLOGI PENELITIAN

3.1 Tempat Penelitian

Penelitian dilaksanakan dalam masa waktu 4 bulan yang dimulai pada bulan Oktober 2019 sampai dengan Januari 2020 dan bertempat di Kampus Teknik Kabupaten Purbalingga, Universitas Jenderal Soedirman.

3.2 Alat dan Bahan Tempat Penelitian

Daftar alat dan bahan yang digunakan selama penelitian ini adalah sebagai berikut.

1. Perangkat Keras :

a. Sebuah Laptop Asus *Vivobook* X442UR dengan spresifikasi prosesor Intel Core i5-7200U CPU @2.50 Hz, RAM 4 GB, dan Harddisk 1 TB, serta VGA Nvidia GeForce 930MX.

b. Komputer virtual *Google Colaboratory* dengan spesifikasi GPU: 1x Tesla K80, komputasi 3.7, mempunyai inti CUDA 2496 , 12 GB GDDR5 VRAM, CPU: 1x *single core hyper threaded Xeon Processors* @2.3 Ghz, RAM: ~12.6 GB, *Disk*: ~33 GB.

2. Perangkat Lunak :

a. Sistem Operasi Windows 10 64 bit.

b. *Google Earth Pro* versi 7.3.2.5776 64 bit.

c. Peramban Web *Google Chrome* versi 78.0.3904.97 64 bit.

d. *Google Colaboratory* (*Jupyter Notebook versi cloud).*

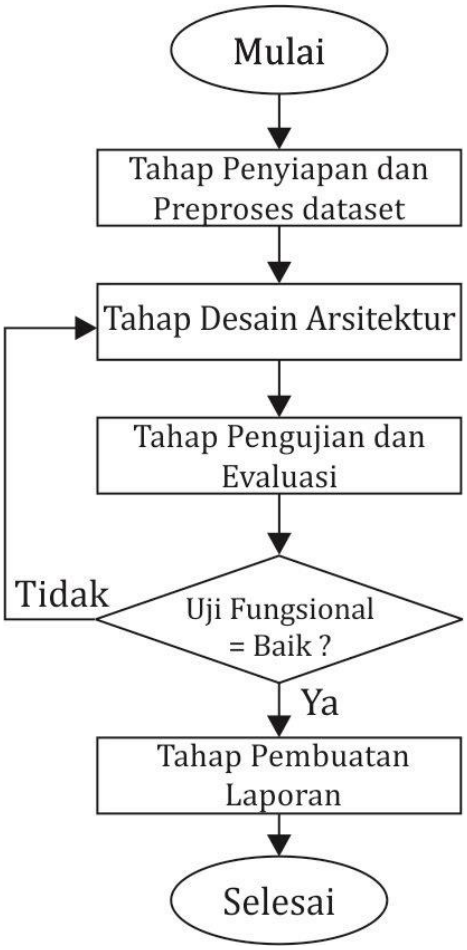
e. Layanan Repositori Web *Development* pada *Platform Github*.

3. Alat pengukur panjang, berupa Meteran digunakan untuk mengukur panjang lahan dan sebagai pembukti validitas meter dalam piksel.
4. Dataset yang digunakan untuk pelatihan dan pengujian berupa sampel citra geografis lahan pertanian yang diambil dari *Google Earth* memiliki ketinggian 100 meter dan 300 meter, serta objek lahan sawah dan bukan lahan sawah dengan resolusi dimensinya 1116x632 piksel berformat "*.jpg*" dan kedalaman bit (*bit depth)* sebanyak 24 bit.
5. Program yang dibuat memikiki *sourcecode* berbahasa pemrograman *python* dan memiliki format "*.ipynb*" atau *IPython Notebooks*, sehingga dapat diakses maupun melakukan eksekusi di dalam browser secara langsung.

3.3 Metode dan Alur Penelitian

Metode penelitian dalam data mining merupakan tahapan atau algoritma perencanaan dengan bantuan beberapa metode/cara, teknik, alat dan dokumentasi untuk membantu penulis dalam meminimalkan resiko kegagalan dan menekankan pada proses/sasaran penelitian.

Untuk penjelasan dalam alur penelitian ini dilakukan melalui beberapa tahapan yang dimulai dari tahap penyiapan dan preproses dataset, dilanjutkan dengan tahap desain arsitektur dataset. Kemudian dilakukan tahap pengujian dan evaluasi sistem. Pada tahap terakhir yaitu tahapan pembuatan laporan penelitian. Berikut diagram alirnya yang disajikan pada Gambar 3.1.



Gambar 3. 1 Diagram Alir Penelitian

Penelitian ini dilaksanakan melalui beberapa tahapan sebagai berikut.

3.3.1 Tahap Penyiapan dan Preproses Dataset

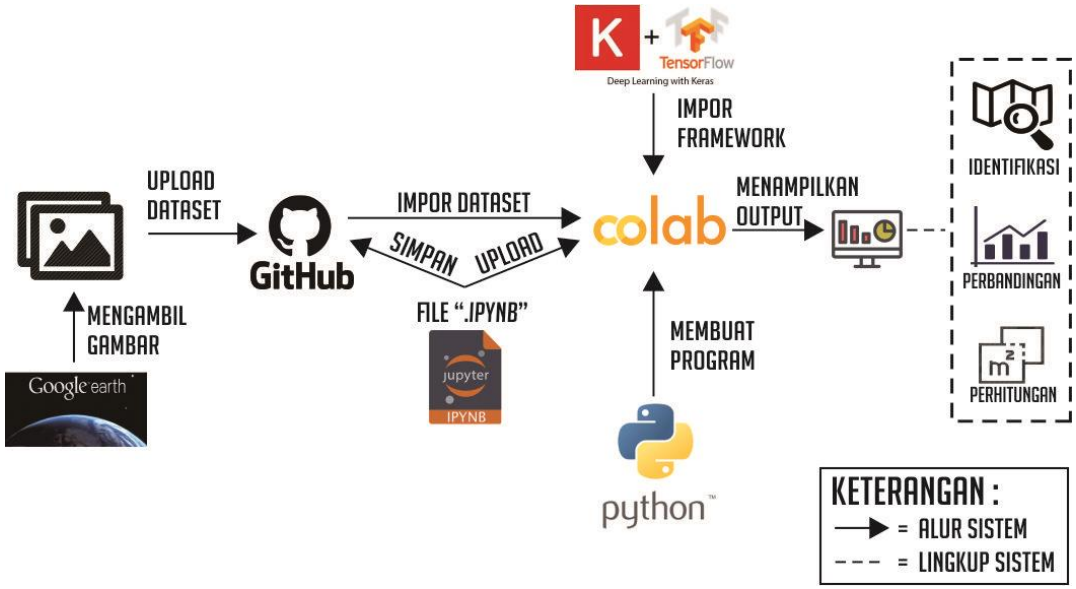
Pada tahap penyiapan ini digunakan untuk menyiapkan alat dan bahan yang diperlukan selama penelitian. Alat dan bahan tersebut berupa perangkat keras laptop dan software *Google Earth Pro*, serta peramban web *Google Chrome* yang digunakan untuk akses masuk pengolahan data berbasis cloud pada *Google Colaboratory*. Selain itu, penulis juga mengumpulkan berbagai informasi melalui studi pustaka. Studi pustaka yang dilakukan oleh penulis, yaitu membaca penelitian sebelumnya, jurnal, buku dan artikel ilmiah yang bersumber dari internet terkait

dengan topik penelitian. Teori-teori yang diperoleh dari studi pustaka ini akan menunjang penulis dalam mengerjakan tugas akhir.

Sedangkan pada tahap preproses dataset dilakukan dengan mengambil citra geografis wilayah di tiga lokasi pulau yang terdapat lahan persawahan, yaitu Jawa, Kalimantan, dan Sumatra dari *Google Earth Pro* diambil pada ketinggian 100 meter dan 300 meter. Selain itu juga diambil citra dengan objek bukan lahan sawah, misal rumah, jalanan, kendaraan, dan sebagainya. Citra memiliki resolusi dimensi sebesar 1136x632 piksel dan memiliki format citra berupa ".jpg", serta bit depthnya sebanyak 24 bit. Jumlah dataset yang dikumpulkan tidak dibatasi pada penelitian ini, akan tetapi penulis telah mengumpulkan dataset untuk pelatihan dan pengujian ini sebanyak 220 citra, dengan rincian citra pelatihan sebanyak 75 lahan persawahan dan 75 area tanpa sawah, serta dataset untuk pengujian sebanyak 70 citra. Pada dataset pengujian terdapat citra masing-masing setiap lokasi sebanyak 20 citra dan pengujian lahan campuran sebanyak 10 citra. Kemudian datasheet ini akan dikelola pada proses identifikasi lahan pertanian melalui *Google Colaboratory*.

3.3.2 Tahap Desain Arsitektur

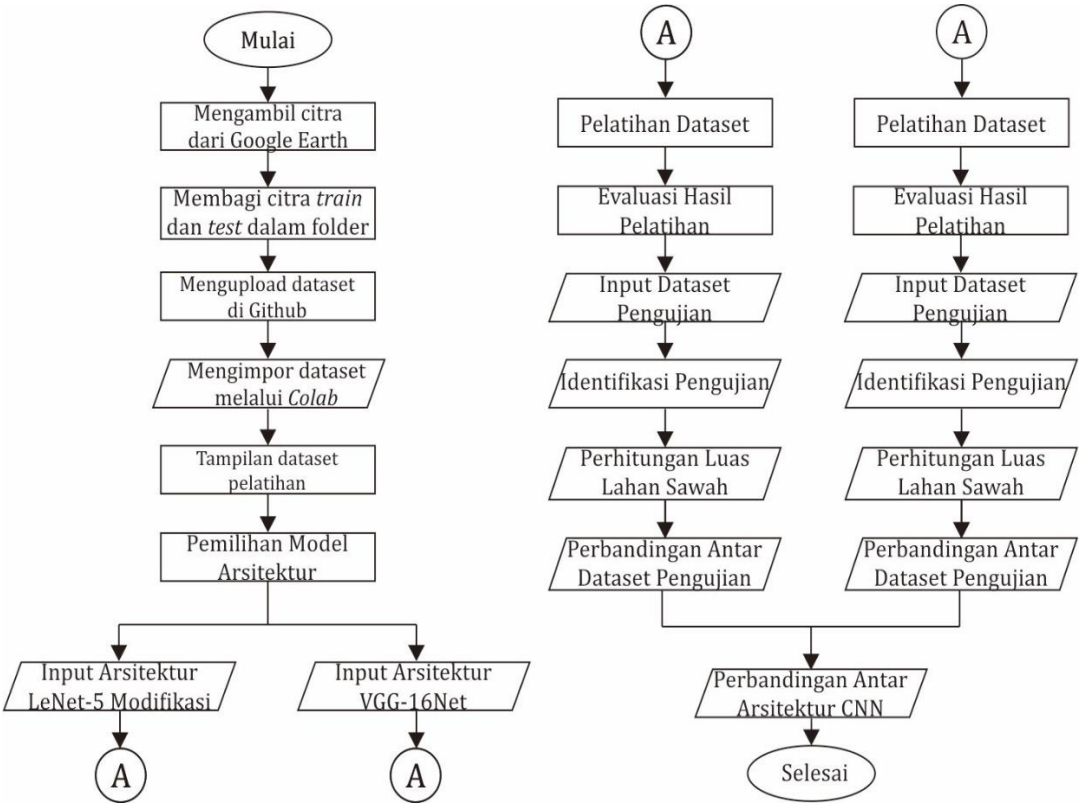
Tahap ini berfungsi untuk menunjukan gambaran kerja dari sistem yang dibuat oleh penulis berdasarkan hasil studi pustaka dan diskusi dengan dosen pembimbing. Untuk memudahkan pemahaman konsepnya, penulis membuat perencanaan desain infrastruktur penulis untuk mengidentifikasi aktifitas yang terjadi pada sistem. Hasil perencanaan desain sistem yang diusulkan pada penelitian ini dapat dilihat pada Gambar 3.2 berikut.



Gambar 3. 2 Desain Infrastruktur Sistem

Setelah menyiapkan alat dan bahan yang diperlukan selama penelitian, serta diperoleh *datasheet* untuk pelatihan dan pengujian pada *Google Colaboratory*. Proses awal yang dilakukan dengan membuat satu folder yang berisikan datasheet, lalu diupload pada situs *Github* melalui repositori akun user. Selain itu juga terdapat file ".ipynb" sebagai dokumen proyek file ini yang berisi artikel dan proses pemrograman dalam bahasa *python* yang akan dieksekusi di dalam web browser secara langsung. Pada *Google Colaboratory* dapat dibuka file proyek tersebut melalui *Github*, lalu dilakukan tahap eksekusi file. Selanjutnya pengolahan data dari metode *deep learning* ini akan secara langsung menampilkan hasil/output pada *Google Colaboratory*.

Kemudian dijelaskan juga pemahaman kerja sistem dalam merepresentasikan ke dalam bentuk diagram alir sebagaimana terlihat pada Gambar 3.3.



Gambar 3. 3 Diagram Alir Sistem

Pada *Google Colaboratory* dapat dibuka file proyek tersebut melalui *Github*, lalu dilakukan tahap eksekusi file. Kemudian pada dataset akan tersimpan dan dilatih menggunakan metode *deep learning* CNN sampai dengan panjangnya *epoch* yang ditentukan. Arsitektur CNN yang digunakan yaitu *LeNet-5* dan *VGG-16Net*. Sesudah itu dataset pelatihan diekstrak ke dalam bentuk *array* dan dibandingkan hasilnya dengan data pengujian untuk melakukan perbandingan area yang terdapat lahan persawahannya, serta dapat diketahui luas lahannya. Karena pada dataset pengujian terdapat 3 jenis lokasi sawah yang berbeda, kemudian ketiga lokasi tersebut dibandingan dengan parameter yang ditentukan menjadi sebuah grafik. Begitu pun hasil perbandingan dari pemilihan arsitekur yang berbeda akan

dibandingan seberapa efisien penggunaan aristekurnya. Kemudian jika nilai akurasi masih rendah dan nilai kesalahan masih tinggi maka perlu diubah parameter citra dan modelnya.

3.3.3 Tahap Pengujian dan Evaluasi

Setelah sistem telah dirancang, tahap selanjutnya adalah pengujian dan evaluasi. Tahap ini penulis melakukan pengujian keseluruhan terhadap sistem yang telah dirancang untuk mendapatkan hasil uji berdasarkan metode CNN dari pengolahan data pelatihan. Sistem akan diuji apakah sudah sesuai yang diharapkan atau belum. Pengujian ini akan menghasilkan output berupa identifikasi lahan pertanian yang diujikan, dan perbandingan hasil pengujian terhadap tiga lokasi sawah yang berbeda serta nilai luas lahannya.

Setelah tahap pengujian selesai, hasil pengujian digunakan sebagai bahan evaluasi terhadap sistem yang dibangun. Sistem dianggap sudah baik jika nilai dari kesalahan atau error (*loss*) semakin mendekati 0, nilai dari akurasi pelatihan semakin mendekati 1, dan nilai sebagian besar hasil prediksi mendekati data validnya. Jika tidak sesuai maka penulis melakukan pembenahan dan pengkajian ulang terhadap berbagai hal yang timbul selama pengujian.

3.3.4 Tahapan Pembuatan Laporan

Tahap laporan adalah tahap terakhir dalam penelitian ini. Setelah sistem menghasilkan keluaran sesuai dengan yang diharapkan, maka disusunlah laporan yang berjudul “IDENTIFIKASI LAHAN PERTANIAN MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA *GOOGLE EARTH*”.

Penulisan laporan dimulai dengan mengolah data hasil pengujian, kemudian menganalisis hasil dalam penelitian ini. Setelah itu, memaparkan hasilnya ke dalam bentuk presentasi dan pembukuan laporan.

3.4 Waktu dan Jadwal Penelitian

Penelitian dilaksanakan dalam waktu 4 bulan dimulai dari bulan Oktober 2019 sampai dengan bulan Januari 2020 dengan rincian jadwal kegiatan sebagai berikut pada tabel 3.1.

Tabel 3. 1 Waktu dan Jadwal Penelitian

No.	Kegiatan	Bulan 1				Bulan 2				Bulan 3				Bulan 4			
		I	II	III	IV	I	II	III	IV	I	II	III	IV	I	II	III	IV
1.	Studi Pustaka																
2.	Pengumpulan data																
3.	Perancangan sistem																
4.	Pengujian dan evaluasi sistem																
5.	Pembuatan laporan																

BAB 4
PEMBAHASAN

4.1 Area dan Dataset Penelitian

Penelitian ini memiliki *dataset* pada pelatihan dan pengujian dengan objek citra area geografis berupa lahan pertanian sawah pada tiga pulau di Indonesia, yaitu Jawa, Kalimantan, dan Sumatra. Citra tersebut diambil melalui layanan *Google Earth Pro*, dimana pengguna dapat mencari, melihat dan menyimpan objek suatu area geografis dalam bentuk citra berformat "*.jpg*". Dataset yang telah disusun dalam bentuk file, kemudian diupload pada situs *Github* melalui repositori akun pengguna.

4.1.1 Area Penelitian

Area/wilayah yang dijadikan sebagai objek observasi lahan pertanian sawah telah dilakukan riset terlebih dahulu melalui informasi/berita pada internet dengan pokok pembahasan kuantitas lahan pertanian sawah di Indonesia, khususnya pulau Jawa, Kalimantan, dan Sumatra. Dari ketiga pulau tersebut yang dijadikan sebagai objek pengambilan citra lahan sawah pada area/wilayah berfokus hanya di satu kabupaten pada masing-masing pulau. Sehingga peneliti dapat menjangkau area yang memiliki kualitas pengambilan citra yang baik dari ketiga pulau yang sudah ditentukan. Berikut ini ditampilkan tabel 4.1 dibawah ini mengenai ketiga pulau yang peneliti telah tentukan, sebagai berikut :

Tabel 4. 1 Area/Wilayah Penelitian

No.	Area Penelitian	Lokasi spesifik	Koordinat Geografis	Waktu Pengambilan Citra	Keterangan Lainnya
1.	Pulau Jawa	Kabupaten Banyumas, Provinsi Jawa Tengah	7°25'31.66"S 109°20'22.26" E	26-02-2019	Memiliki luas lahan sawah total dari jenis pengairan sebesar 70.810 Ha [1]
2.	Pulau Kalimantan	Kabupaten Tanah Laut, Provinsi Kalimantan Selatan	3°46'20.79"S 114°49'28.12" E	02-03-2018	Memiliki luas lahan sawah total dari jenis pengairan sebesar 74.061 Ha [1]
3.	Pulau Sumatra	Kabupaten Ogan Komering Ulu Timur, Provinsi Sumatra Selatan	4°10'25.59"S 104°38'59.84" E	26-07-2016	Memiliki luas lahan sawah total dari jenis pengairan sebesar 176.401 Ha [1]

Berdasarkan tabel 4.1 diatas terdapat perbedaan kurun waktu pengambilan citra dataset dari ketiga pulau, karena satelit yang dimiliki oleh pihak *Google Earth* tidak sepenuhnya berada pada posisi statis pada satu tempat. Pengambilan citra geografis pada Kabupaten Banyumas di Pulau Jawa memiliki intensitas kurun waktu pengambilan yang lebih sering oleh satelit, dibandingkan kedua area lainnya. Oleh karena itu, peneliti tidak membatasi waktu pengambilan citra yang ketiganya diharuskan sama. Selain itu terdapat faktor lain yang peneliti perhatikan, seperti

kualitas citra suatu area yang terlihat cerah dan jelas, serta lahan sawah yang diambil dalam kondisi berwarna hijau atau bukan sawah kering. Sehingga beberapa ketentuan tersebut diperoleh citra lahan pertanian sawah yang digunakan menjadi dataset-dataset dalam penelitian ini.

4.1.2 Dataset

Dataset pada penelitian ini meliputi kumpulan citra yang telah diperoleh pada pengambilan citra geografis ketiga area yang diteliti dan dibagi menjadi beberapa folder yang berisi citra. Citra-citra tersebut dibagi menjadi 3 jenis objek utama, yaitu citra lahan sawah, citra bukan/atau selain sawah, dan citra campuran/perpaduan antara sawah maupun bukan sawah. Kemudian dibuat folder yang digunakan sebagai dataset dengan nama, berupa folder *train*, folder *chopped*, dan folder *test*.

Dataset yang digunakan pada identifikasi lahan pertahanian sawah ini terdapat tiga macam, yaitu *dataset* pelatihan, *dataset* validasi, dan *dataset* pengujian. Seluruh *dataset* yang diteliti oleh peneliti memiliki resolusi dimensi sebesar 1136x632 piksel dan memiliki format citra berupa "*.jpg*", serta bit depthnya sebanyak 24 bit. Peneliti melakukan pelatihan *dataset* yang dimiliki dengan menggunakan metode *Chopping Picture Method* (CPM). Sehingga dilakukan pemotongan setiap citra pada *dataset* pelatihan menjadi citra dengan sangat kecil, serta ukurannya menjadi 56x56 piksel. Kemudian tersimpan *dataset* pelatihan baru yang telah dipotong-potong pada folder *chopped*. Pada dataset pelatihan dan *dataset* validasi merupakan dataset yang akan digunakan untuk memperoleh suatu model dari objek identifikasi penelitian ini. Peneliti menggunakan 75% citra yang telah

dipotong-potong sebagai *dataset* pelatihan dan sisanya 25% sebagai *dataset* validasi. Sedangkan pada dataset pengujian digunakan untuk memperoleh hasil probabilitas/prediksi dan perbandingan dari model CNN pada pelatihan. Pada tabel 4.2 diperlihatkan *dataset* pelatihan dan setelah dilakukan metode CPM pada pengelompokkan CNN.

Tabel 4. 2 Dataset Pelatihan dan Dataset Validasi Pada Folder Chopped

Class	**Objek Citra**		**CNN**	**Klasifikasi CNN**	
	Area Penelitian (citra)		**Folder *Train***	**Folder *Chopped***	
			***Dataset* Pelatihan**	**Pelatihan**	**Validasi**
Sawah	Jawa	25	75 citra	33.412 / 33.413 citra	11.137 / 11.138 citra
	Kalimantan	25			
	Sumatra	25			
Bukan Sawah	Jawa	25	75 citra	33.412 / 33.413 citra	11.137 / 11.138 citra
	Kalimantan	25			
	Sumatra	25			
Total Citra		150	150 citra	66.825 citra	22.275 citra
				89100 citra	

Berdasarkan tabel 4.2 menjelaskan class, objek citra dan folder *train* yang memiliki jumlah citra pada *dataset* pelatihan sebelum dilakukan metode CPM. Sehingga terdapat perubahan jumlah *dataset* pelatihan pada folder *chopped* menjadi puluhan ribu citra. Berikut ini *dataset-dataset* yang digunakan pada identifikasi lahan pertahanian sawah ini terdapat tiga macam, yaitu *dataset* pelatihan, *dataset* validasi, dan *dataset* pengujian.

1. Jumlah *Dataset* Pelatihan

Pada penelitian ini terdapat *dataset* pelatihan sebanyak 150 citra memiliki format citra berupa "*.jpg*". *Dataset* pelatihan ini dibagi menjadi 2 *class*, yaitu citra sawah dan citra bukan/atau selain sawah. Kemudian setiap *class* memiliki tiga area yang berbeda dan masing-masing area berjumlah 25 citra, sehingga apabila dijumlahkan satu *class* terdapat total sebanyak 75 citra. Karena pada *dataset* pelatihan ini menggunakan metode CPM yang memotong setiap satu citra dalam ukuran 56x56 piksel, maka jumlah citra menjadi bertambah. Sehingga memiliki jumlah dari kedua *class* sebanyak 89.100 citra. Selanjutnya *dataset* pelatihan tersimpan pada folder *chopped* dan setiap *class* maupun area wilayah berada dalam satu folder. *Dataset* pelatihan ini diambil sebanyak 75% secara acak dari folder *chopped* yang berjumlah 66.825 citra meliputi setiap *class* diambil 33.412 citra dan 33.413 citra secara acak.

2. Jumlah *Dataset* Validasi

Dataset validasi yang digunakan sebanyak 25% dari folder *chopped* dengan jumlah citra 22.275 citra. Sehingga jumlahnya sebanyak 11.137 citra atau 11.138 citra pada setiap *class.*

3. Jumlah *Dataset* Pengujian

Jumlah *dataset* pengujian yang digunakan pada folder *test* sebanyak 70 citra dengan terdapat folder-folder yang terpisah didalamnya, yaitu folder Jawa, folder Kalimantan, folder Sumatra, dan folder *test image* seperti yang terlihat pada tabel 4.3. Selain folder *test image*, di setiap folder memiliki jumlah citra sebanyak 20 citra dengan dibagi menjadi dua *class*. Sehingga setiap folder pulau memiliki jumlah sebanyak 10 citra di setiap *class.* Sedangkan pada folder *test image* terdapat

citra-citra perpaduan/campuran antara objek sawah dan bukan sawah dan berjumlah total 10 citra. Pada folder *test image* ini digunakan untuk melakukan uji implementasi identifikasi dan perhitungan lahan pertanian sawah terhadap jaringan CNN yang digunakan. Pada pulau Jawa diambil satu citra dari *dataset* pengujian citra yang memiliki objek perpaduan sawah dan/atau selain untuk dijadikan sampel pengujian. Kemudian akan digunakan sebagai perbandingan terhadap ketinggian yang berbeda dan arsituktur yang digunakan.

Tabel 4. 3 Dataset Pengujian pada Folder Test

Objek Citra **Folder *Test*** ***Dataset* Pengujian (citra)**		**Class**	
Jawa	20	Sawah	10 citra
		Bukan Sawah	10 citra
Kalimantan	20	Sawah	10 citra
		Bukan Sawah	10 citra
Sumatra	20	Sawah	10 citra
		Bukan Sawah	10 citra
Test Image	10		
Total Citra	70	60 citra	

4.1.3 Pengambilan Dataset

Dataset diperoleh dengan mengambil objek citra geografis wilayah di tiga lokasi pulau yang terdapat lahan pertanian sawah, yaitu Jawa, Kalimantan, dan Sumatra dari perangkat lunak *Google Earth Pro*. Terdapat dua kriteria pengambilan citra, yaitu diambil pada ketinggian 100 meter dan 300 meter. Selain itu juga diambil citra dengan objek bukan lahan sawah, misal rumah, jalanan, kendaraan,

dan sebagainya. Untuk mencari area yang akan diteliti dapat dicari melalui panel pencarian, kemudian klik tombol “OK”. Karena pada fitur *Google Earth Pro* tidak terdapat cara yang dapat mengatur ketinggian dengan menginputnya, maka peneliti melakukannya melalui penganturan *zoom*/memperbesar citra dengan kursor hingga tertampilkan ketinggian yang akan digunakan. Setelah menentukan area yang akan diambil, terdapat fitur pada *Google Earth Pro* yang dapat menyimpan citra pada folder di komputer dengan dapat memilih resolusi citra yang akan diambil. Peneliti menyimpan citra yang memiliki resolusi dimensi sebesar 1136x632 piksel atau standar dan memiliki format gambar berupa “*.jpg*”, serta *bit depth*nya sebanyak 24 bit.

4.2 Pelatihan dan Pengujian pada Google Colaboratory

Setelah diperoleh *dataset* yang diperlukan selama penelitian, maka dilakukan tahapan pelatihan dan pengujian pada *Google Colaboratory*. Pelatihan dan pengujian pada *Google Colaboratory* digunakan untuk melatih *dataset* dan memperoleh model, serta diaplikasikan hasilnya dalam pengujian citra. Proses awal yang dilakukan pada *Google Colaboratory* dengan membuka file proyek penelitian melalui *Github* dengan format “*.ipynb*” sebagai dokumen proyek file ini yang berisi artikel dan proses pemrograman dalam bahasa *python*, lalu dilakukan tahap eksekusi file. Kemudian pada dataset akan tersimpan dan dilatih menggunakan metode *deep learning* CNN sampai dengan panjangnya *epoch* yang ditentukan. Parameter model yang diimplemetasikan dalam penelitian ini meliputi jumlah pelatihan *epoch* sebanyak 50 *epoch*, batch size pada pelatihan sebanyak 200 citra. Arsitektur CNN yang digunakan yaitu *LeNet-5* Modifikasi

dan *VGG-16Net*. Sesudah itu dataset pelatihan diekstrak ke dalam bentuk *array* dan dibandingkan hasilnya dengan data pengujian untuk melakukan perbandingan antar *class* area yang terdapat lahan persawahannya dan bukan lahan sawah, serta citra perpaduan/campuran antara sawah dan bukan sawah. Karena pada dataset pengujian terdapat 3 jenis lokasi sawah yang berbeda, kemudian ketiga lokasi tersebut dibandingan dengan parameter yang ditentukan menjadi sebuah grafik. Begitu pun hasil perbandingan dari pemilihan arsitekur yang berbeda akan dibandingan seberapa efisien penggunaan aristekurnya. Kemudian jika nilai akurasi masih rendah dan nilai kesalahan masih tinggi maka perlu diubah parameter citra dan modelnya.

4.2.1 Dataset Pelatihan dan Pengujian dari Ketiga Pulau

Pada *dataset* pelatihan memiliki kumpulan citra random yang ditempatkan pada satu folder dari ketiga area yang dilatih, yaitu pulau Jawa, pulau Kalimantan, dan pulau Sumatra. Perbedaan dari ketiganya hanya dibatasi pada *class*, dimana terdapat citra sawah dan citra yang bukan/atau selain sawah. *Dataset* pelatihan ini akan diambil secara acak dengan jumlah mencapai 89.000 citra hasil dari metode CPM, kemudian digunakan sebagai *dataset* yang dilatih.

Sedangkan pada *dataset* pengujian di setiap citra pada masing-masing area dibuat folder dengan nama pulaunya. Ketiga area tersebut tidak dilakukan metode CPM dan hanya sebagai citra yang masih utuh. Seperti halnya *dataset* pelatihan, dimana perbedaan di setiap pulau hanya dibatasi pada *class* juga. Selain itu terdapat satu folder pengujian lain yang terpisah, serta digunakan sebagai objek pengujian implementasi dalam identifikasi dan perhitungan luas lawah pertanian sawah. Citra

yang berada didalamnya berisikan citra perpaduan/gabungan antara sawah dan bukan sawah.

4.2.2 Ragam Dataset Berdasarkan Skala Ketinggian

Dataset yang diambil pada identifikasi lahan pertanian sawah terdapat dua kondisi, yaitu citra pada ketinggian 100 meter dan citra pada ketinggia 300 meter. Ketinggian tesebut didasari oleh pengukuran yang tertera pada keterangan ketinggian di suatu area setiap pulau pada *Google Earth Pro*. Kemudian kedua kondisi ini memiliki file repositori penelitian yang dipisah satu sama lain dan *dataset* nya pun juga dibedakan. Dalam melakukan proses pelatihan dan pengujian pada *Google Colaboratory* untuk memperoleh hasil identifikasi lahan pertanian sawah diperlukan untuk melakukannya secara masing-masing.

4.2.3 Arsitektur yang Digunakan

Arsitektur CNN yang digunakan pada pengujian identifikasi lahan pertanian ini, yaitu arsitektur *LeNet-5* yang dimodifikasi dan *VGG-16Net*. Kedua arsitektur ini masing-masing akan dibandingkan performa dalam melakukan CNN, sehingga dapat diperoleh hasil yang terbaik dan melakuan pembelajaran terhadap gambar yang telah dipotong-potong dengan berhasil.

4.2.4 Hasil Fluktuasi *Accuarcy* dan *Loss* Selama Pembelajaran *Epoch*

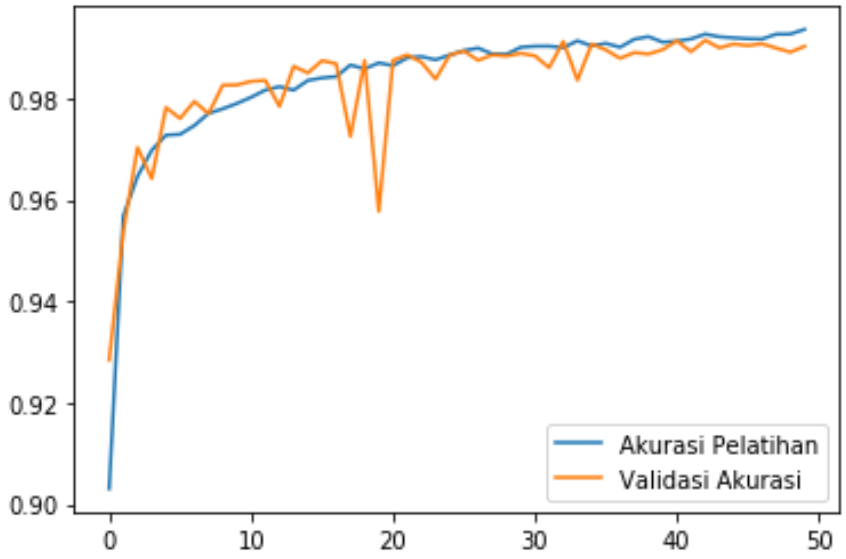
Pada penelitian ini pengujian yang dilakukan pada *Google Colaboratory* terdapat empat macam kondisi atau file yang berbeda, yaitu *dataset* pelatihan dan pengujian dengan ketinggian citra 100 meter menggunakan arsitektur *LeNet-5* modifikasi, *dataset* pelatihan dan pengujian dengan ketinggian citra 100 meter menggunakan arsitektur *VGG-16Net*, *dataset* pelatihan dan pengujian dengan ketinggian citra 300 meter menggunakan arsitektur *LeNet-5* modifikasi, dan *dataset*

pelatihan dan pengujian dengan ketinggian citra 300 meter menggunakan arsitektur *VGG-16Net*. Tujuannya untuk menguji dan membandingkan performa kedua arsitektur melakukan CNN, sehingga dapat diperoleh hasil yang terbaik dan melakuan pembelajaran terhadap gambar yang telah dipotong-potong dengan berhasil. Kemudian mengamati pengaruh kondisi *dataset* dari perbedaan faktor ketinggian terhadap tingkat akurasi keberasilan dan mengidentifikasi lahan pertanian sawah.

Pada pengujian ini memiliki *epoch* dalam pemodelan sebanyak 20. Akurasi diperoleh ketika keakuratan suatu model dapat mengklasifikasi citra validasi dan *loss* menandakan ketidakakuratan atau kegagalan dalam memprediksi suatu model. Jika pemodelan berhasil, maka *loss* pada validasi mencapai nilai yang sangat kecil umumnya mendekati nilai 0 dan pada akurasinya mencapai nilai yang mendekati nilau 1, serta umumnya disebut sebagai kondisi *goodfit*. Namun ketika *loss* pada validasi menjadi tinggi dan terdapat variasi biasnya selama pembelajaran, maka diindikasikan terjadinya *overfitting*. Sebaliknya apabila *loss* pada validasi menjadi rendah dibandingkan dengan loss pada pelatihan, maka terjadi *underfitting*.

1. Hasil pelatihan dan pengujian pada arsitektur *LeNet-5* modifikasi dengan ketinggian 100 meter

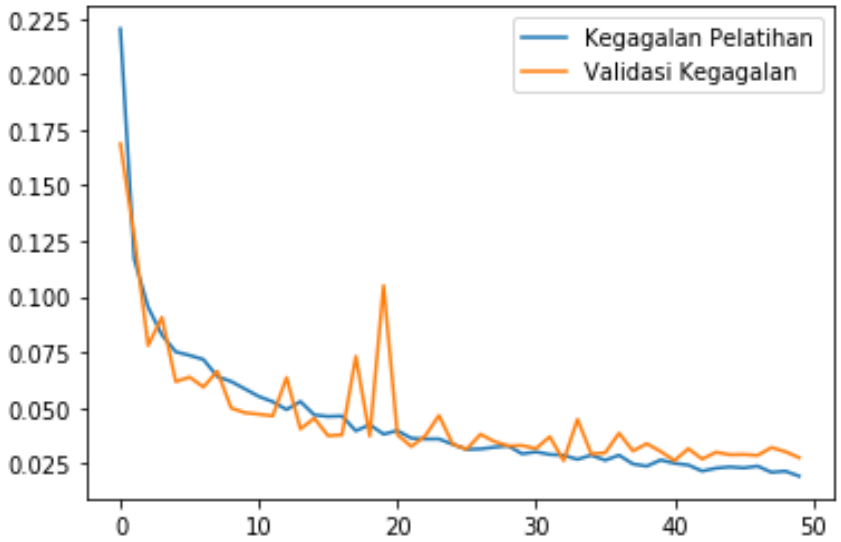
Berikut ini ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam akurasinya pada Gambar 4.1.



Gambar 4. 1 Kurva Akurasi Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur LeNet-5 Modifikasi

Pada gambar 4.1 diatas memperlihatkan kondisi kurva akurasi pelatihan yang mendekati nilai 1 di seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Sehingga diperoleh nilai akurasi pada *epoch* terakhir diantara akurasi pelatihan dengan akurasi validasi masing-masing adalah 0.9936 dan 0.9903. Diketahui bahwa garis kurva yang berwarrna biru merupakan akurasi pelatihan, sedangkan yang berwarna oranye merupakan akurasi validasi. Karena pada kurva dapat mencapai akurasi mendekati nilai 1, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya, lalu tanpa adanya fluktuasi yang signifikan. Oleh karena itu, pada kurva akurasi pelatihan pada dataset pelatihan dan validasi dengan aristektur *LeNet-5* Modifikasi disebut dengan kondisi *goodfit*.

Adapun ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam kegagalannya pada Gambar 4.2 dibawah ini.



Gambar 4. 2 Kurva Kegagalan Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur Lenet-5 Modifikasi

Pada gambar 4.2 diatas memperlihatkan kondisi kurva kegagalan/kesalahan pelatihan yang mendekati nilai 0 seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Nilai kegagalan pada *epoch* terakhir pada kegagalan pelatihan sebesar 0.0193, sedangkan pada kegagalan validasi sebesar 0.0277. Perbedaan kegagalan antara keduanya dibedakan pada warna garis kurva seperti yang tertera pada gambar. Karena pada kurva dapat mencapai kegagalan mendekati nilai 0, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi dengan baik. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya hingga akhir *epoch* yang ditentukan, meskipun kurva kegagalan validasi sempat berada pada kondisi fluktuasi diawal *epoch*. Oleh karena itu, pada kurva kegagalan/kesalahan pelatihan pada dataset di

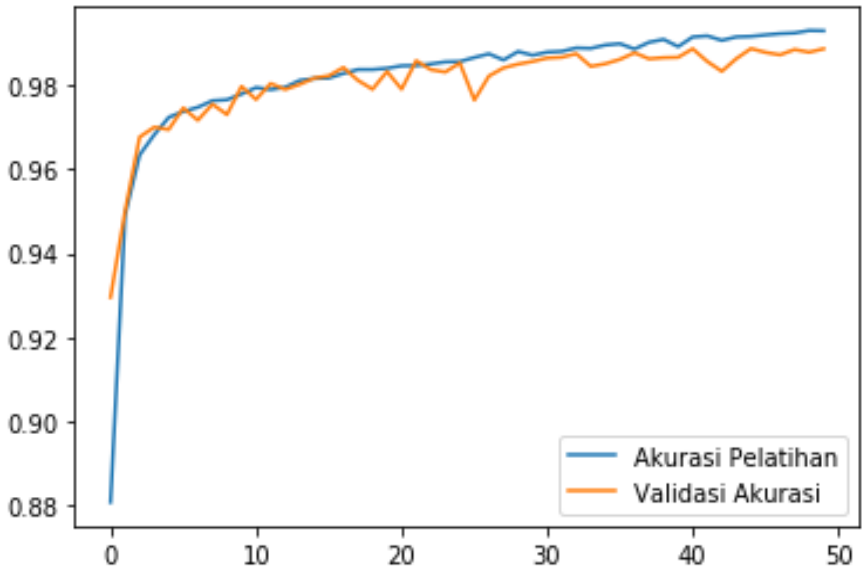
ketinggian 100 meter pelatihan dan validasi dengan aristektur LeNet-5 Modifikasi disebut dengan kondisi *goodfit.*

Setelah dilakukan hasil pelatihan pada *dataset* pelatihan dan validasi yang mendapatkan masing-masing nilai akurasi dan kegagalannya. Maka selanjutnya hasil tesebut diujikan dan dilakukan evaluasi model pada *dataset* pengujian di ketiga area pulau. Kemudian diperoleh hasil pengujian akurasi dan kegagalannya, yaitu pada pulau Jawa diperoleh akurasi sebesar 0.8999 dan kegagalan yang cukup tinggi dibandingkan kedua area lainnya sebesar 0.5691, pada pulau Kalimantan didapatkan akurasi sebesar 0.8999 dan kegagalan pengujiannya sebesar 0.2153, dan pengujian pada pulau Sumatra yang mendapatkan hasil pengujian yang sangat baik pada akurasinya sebesar 0.9499 dan kegagalan yang cukup rendah sebesar 0.1861.

Hasil akurasi dan kegagalan yang diujikan pada *dataset* pengujian pada ketinggian 100 meter tersebut masih dinyatakan cukup baik dalam mengidentifikasi suatu objek citra area di ketiga pulau. Meski terdapat kegagalan yang masih sedikit tinggi pada pulau Jawa, hal tersebut dikarenakan dengan kemungkinan pengambilan *dataset* pelatihan dan validasi yang random dan jumlah yang sangat banyak dalam proses pelatihan. Sehingga ketika diujikan terhadap ketiga pula, model masih terdapat citra yang belum mengenali citra pada pulau Jawa.

2. Hasil pelatihan dan pengujian pada arsitektur *LeNet-5* modifikasi dengan ketinggian 300 meter

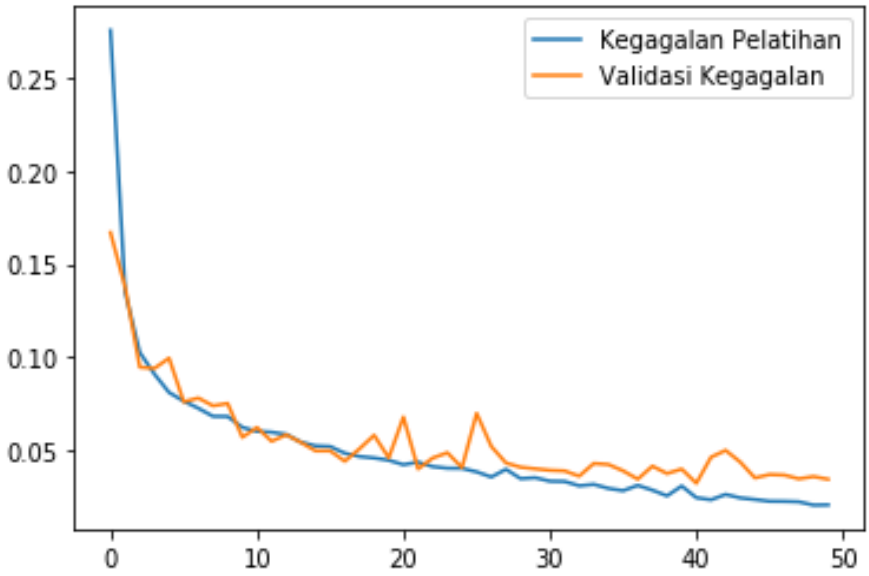
Berikut ini ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam akurasinya pada Gambar 4.3.



Gambar 4. 3 Kurva Akurasi Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur Lenet-5 Modifikasi

Pada gambar 4.3 diatas memperlihatkan kondisi kurva akurasi pelatihan pada *dataset* pelatihan maupun validasi yang mendekati nilai 1 di seiring bertambahnya *epoch*. Sehingga diperoleh nilai akurasi pada *epoch* terakhir diantara akurasi pelatihan dengan akurasi validasi masing-masing adalah 0.9928 dan 0.9986. Karena pada kurva dapat mencapai akurasi mendekati nilai 1, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya, lalu tanpa adanya fluktuasi yang signifikan. Oleh karena itu, pada kurva akurasi pelatihan pada dataset di ketinggian 300 meter pelatihan dan validasi dengan aristektur *LeNet-5* Modifikasi disebut dengan kondisi *goodfit*.

Adapun ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam kegagalannya pada Gambar 4.4 dibawah ini.



Gambar 4. 4 Kurva Kegagalan Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur Lenet-5 Modifikasi

Pada gambar 4.4 diatas memperlihatkan kondisi kurva kegagalan/kesalahan pelatihan pada dataset pelatihan maupun validasi yang mendekati nilai 0 seiring bertambahnya *epoch.* Nilai kegagalan pada *epoch* terakhir pada kegagalan pelatihan sebesar 0.0209, sedangkan pada kegagalan validasi sebesar 0.0346. Karena pada kurva dapat mencapai kegagalan mendekati nilai 0, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi dengan baik. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya hingga akhir *epoch* yang ditentukan. Oleh karena itu, pada kurva kegagalan/kesalahan pelatihan pada dataset pelatihan dan validasi dengan aristektur LeNet-5 Modifikasi disebut dengan kondisi *goodfit.*

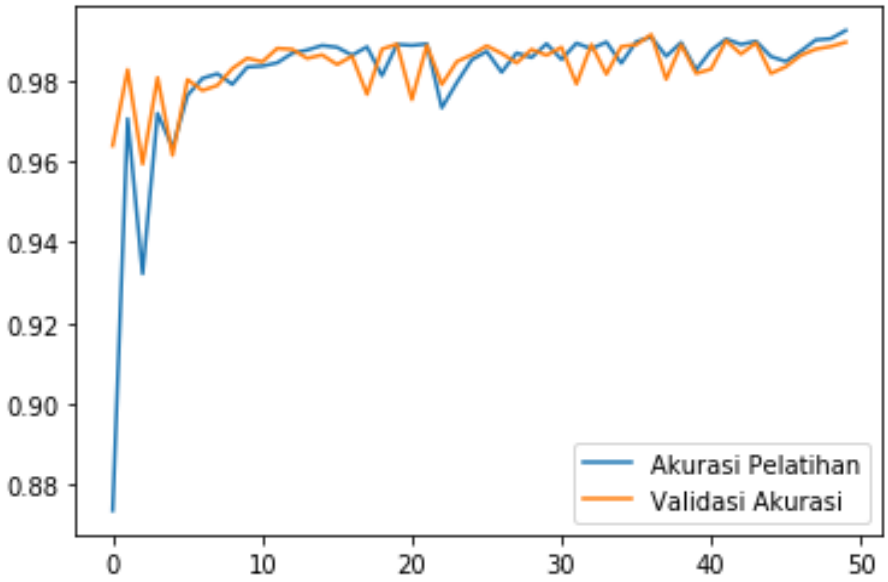
Setelah dilakukan hasil pelatihan pada *dataset* pelatihan dan validasi yang mendapatkan masing-masing nilai akurasi dan kegagalannya. Maka selanjutnya

hasil tesebut diujikan dan dilakukan evaluasi model pada *dataset* pengujian di ketiga area pulau. Kemudian diperoleh hasil pengujian akurasi dan kegagalannya, yaitu pada pulau Jawa diperoleh akurasi yang sangat baik sebesar 0.9499 dan kegagalan sebesar 0.5579, pada pulau Kalimantan didapatkan akurasi sebesar 0.8999 dan kegagalan pengujiannya sebesar 0.4069, dan pengujian pada pulau Sumatra yang akurasinya sebesar 0.8999 dan kegagalan yang cukup rendah sebesar 0.3498.

Hasil akurasi dan kegagalan yang diujikan pada *dataset* pengujian pada ketinggian 300 meter tersebut tergolong belum memuaskan dan tentunya belum sepenuhnya akurat dalam melakukan prediksi maupun mengidentifikasi citra pengujian di ketiga area terutama pada kegagalan yang masih cukup tinggi dikarena tidak mencapai nilai 0. Sedangkan akurasinya diperoleh cukup baik di ketiga area pulau yang diujikan. Hal ini karena citra yang diambil dalam ketinggian 300 meter mengakibatkan objek-objek pada citra lahan sawah dan yang bukan lahan sawah menjadi tidak sepenuhnya dapat dikenali dengan baik. Sehingga terdapat kegagalan yang cukup tinggi dalam pengenalan model terhadap *dataset* pengujian.

3. Hasil pelatihan dan pengujian pada arsitektur *VGG-16Net* modifikasi dengan ketinggian 100 meter

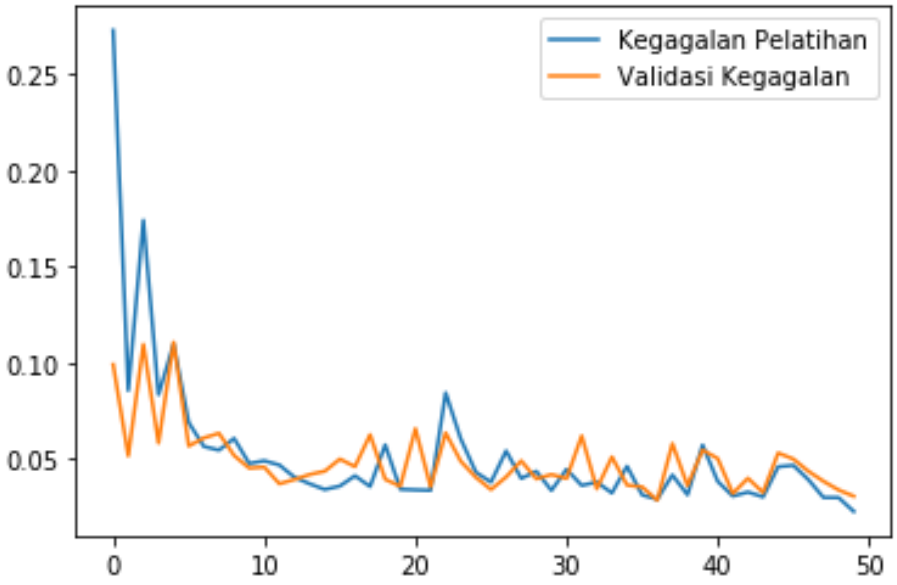
Berikut ini ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam akurasinya pada Gambar 4.5.



Gambar 4. 5 Kurva Akurasi Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur VGG-16Net

Pada gambar 4.5 diatas memperlihatkan kondisi kurva akurasi pelatihan yang mendekati nilai 1 seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Sehingga diperoleh nilai akurasi pada *epoch* terakhir pada akurasi pelatihan sebesar 0.9923, sedangkan pada akurasi validasi sebesar 0.9894. Karena pada kurva dapat mencapai akurasi mendekati nilai 1, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi dengan baik. Selain itu akurasi pelatihan dan validasi berada pada posisi yang relatif sama atau beriringan diantara keduanya, lalu terdapat adanya fluktuasi yang cenderung tidak terlalu bervariasi perubahannya. Oleh karena itu, pada kurva akurasi pelatihan pada dataset pelatihan dan validasi dengan aristektur *VGG-16Net* disebut dengan kondisi *goodfit*.

Adapun ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam kegagalannya pada Gambar 4.6 dibawah ini.



Gambar 4. 6 Kurva Kegagalan Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur VGG-16Net

Pada gambar 4.6 diatas memperlihatkan kondisi kurva kegagalan/kesalahan pelatihan yang mendekati nilai 0 di seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Nilai kegagalan pada *epoch* terakhir pada kegagalan pelatihan sebesar 0.0229, sedangkan pada kegagalan validasi sebesar 0.0309. Perbedaan kegagalan antara keduanya dibedakan pada warna garis kurva seperti yang tertera pada gambar. Karena pada kurva dapat mencapai kegagalan mendekati nilai 0, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya hingga akhir *epoch* yang ditentukan, terdapat adanya fluktuasi yang cenderung tidak terlalu bervariasi perubahannya. Oleh

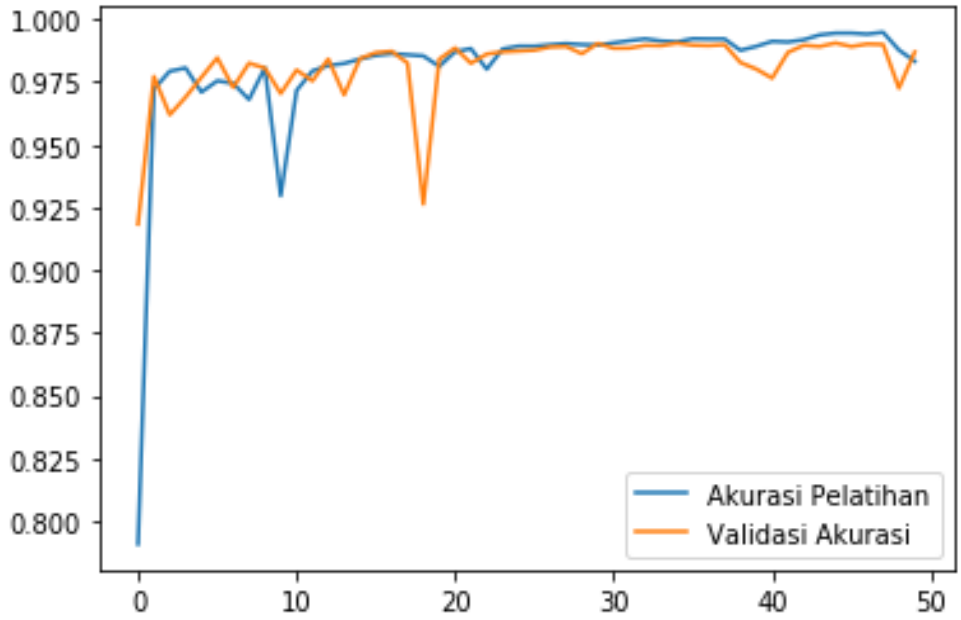
karena itu, pada kurva kegagalan/kesalahan pelatihan pada dataset pelatihan dan validasi dengan aristektur *VGG-16Net* Modifikasi disebut dengan kondisi *goodfit.*

Setelah didapatkan hasil nilai akurasi dan kegagalan pada pelatihan, lalu hasil tesebut diujikan dan dilakukan evaluasi model pada *dataset* pengujian di ketiga area pulau. Kemudian diperoleh hasil pengujian akurasi dan kegagalannya, yaitu pada pulau Jawa diperoleh akurasi yang cukup baik mencapai nilai sebesar 0.95 dan kegagalan yang cukup rendah sebesar 0.1747, pada pulau Kalimantan didapatkan akurasi sebesar 0.85 dan kegagalan pengujiannya sebesar 0.2945, dan pengujian pada pulau Sumatra yang mendapatkan hasil pengujian yang sangat baik mencapai nilai sebesar 1 pada akurasi maupun kegagalannya yang sangat rendah sebesar 0.0326.

Hasil akurasi dan kegagalan yang diujikan pada *dataset* pengujian pada ketinggian 100 meter tersebut dinyatakan sangat baik dalam mengidentifikasi suatu objek citra area di ketiga pulau. Karena dapat mencapai tingkat akurasi yang sangat tinggi dan kegagalan yang cukup rendah di ketiga pulau yang diujikan. Pengambilan *dataset* pelatihan dan validasi yang random dan jumlah yang sangat banyak umumnya dapat mempengaruhi hasil akurasi dan kegagalan terlepas dari dataset pelatihan yang memiliki citra yang telah dipotong-potong dalam proses pelatihan. Namun karena arsitektur *VGG-16Net* ini dapat memberikan hasil evaluasi yang lebih optimal dibandingkan dengan arsitektur LeNet-5 Modifikasi pada ketinggian 100 meter.

4. Hasil pelatihan dan pengujian pada arsitektur *VGG-16Net* modifikasi dengan ketinggian 300 meter

Berikut ini ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam akurasinya pada Gambar 4.7.

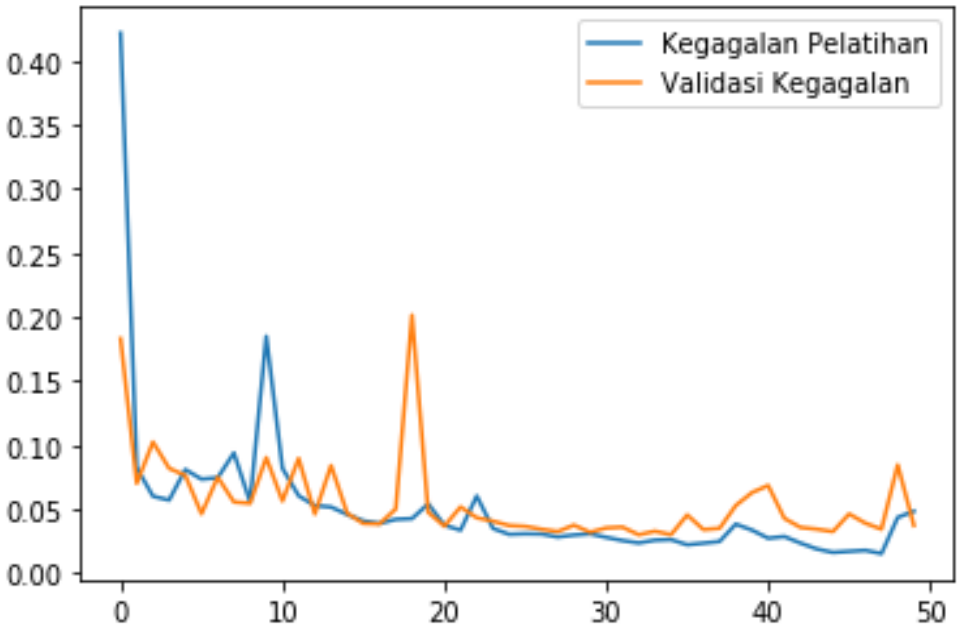


Gambar 4. 7 Kurva Akurasi Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur VGG-16Net

Pada gambar 4.7 diatas memperlihatkan kondisi kurva akurasi pelatihan yang mendekati nilai 1 di seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Sehingga diperoleh nilai akurasi pada *epoch* terakhir diantara akurasi pelatihan dengan akurasi validasi masing-masing adalah 0.9832 dan 0.9869. Karena pada kurva dapat mencapai akurasi mendekati nilai 1, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya, meskipun terdapat adanya sedikit fluktuasi pada akurasi validasi diakhir

bertambahnya *epoch*. Hal ini dinyatakan bahwa kurva akurasi pelatihan pada dataset pelatihan dan validasi dengan aristektur *VGG-16Net* termasuk dalam kondisi *goodfit*.

Adapun ditampilkan hasil pelatihan pada dataset pelatihan dan validasi dalam kegagalannya pada Gambar 4.8 dibawah ini.



Gambar 4. 8 Kurva Kegagalan Pelatihan pada Dataset Pelatihan dan Validasi dengan Arsitektur VGG-16Net

Pada gambar 4.8 diatas memperlihatkan kondisi kurva kegagalan/kesalahan pelatihan yang mendekati nilai 0 di seiring bertambahnya *epoch*, baik pada dataset pelatihan maupun validasinya. Nilai kegagalan pada *epoch* terakhir pada kegagalan pelatihan sebesar 0.0482, sedangkan pada kegagalan validasi sebesar 0.0372. Perbedaan kegagalan antara keduanya dibedakan pada warna garis kurva seperti yang tertera pada gambar. Karena pada kurva dapat mencapai kegagalan mendekati

nilai 0, maka proses pelatihan dan validasi dinyatakan berhasil dalam melakukan klasifikasi dengan baik. Selain itu akurasi pelatihan dan validasi berada pada posisi yang sama atau beriringan diantara keduanya hingga akhir *epoch* yang ditentukan, meskipun kurva kegagalan validasi sempat berada pada kondisi fluktuasi diawal maupun diakhir *epoch*. Oleh karena itu, pada kurva kegagalan/kesalahan pelatihan pada dataset pelatihan dan validasi dengan aristektur LeNet-5 Modifikasi disebut dengan kondisi *goodfit*.

Setelah dilakukan hasil pelatihan pada *dataset* pelatihan dan validasi yang mendapatkan masing-masing nilai akurasi dan kegagalannya. Maka selanjutnya hasil tesebut diujikan dan dilakukan evaluasi model pada *dataset* pengujian di ketiga area pulau. Kemudian diperoleh hasil pengujian akurasi dan kegagalannya, yaitu pada pulau Jawa diperoleh akurasi sebesar 0.9 dan kegagalan yang cukup tinggi dibandingkan kedua area lainnya sebesar 0.7478, pada pulau Kalimantan didapatkan akurasi sebesar 0.85 dan kegagalan pengujiannya juga cukup tinggi sebesar 0.5159, dan pengujian pada pulau Sumatra yang mendapatkan hasil pengujian yang sangat baik pada akurasinya sebesar 0.95 dan kegagalan yang cukup rendah sebesar 0.1010.

Hasil akurasi dan kegagalan yang diujikan pada *dataset* pengujian pada ketinggian 300 meter tersebut tergolong belum memuaskan terutama pada kegagalan yang masih cukup tinggi dikarena tidak mencapai nilai 0. Pada pengujian di area pulau Jawa dan Kalimantan memperoleh hasil kegagalan yang buruk, sehingga tentunya belum sepenuhnya akurat dalam melakukan prediksi maupun mengidentifikasi citra pengujian di kedua area tersebut. Sedangkan akurasinya

diperoleh cukup baik di ketiga area pulau yang diujikan. Hal ini karena citra yang diambil dalam ketinggian 300 meter mengakibatkan objek-objek pada citra lahan sawah dan yang bukan lahan sawah menjadi tidak sepenuhnya dapat dikenali dengan baik. Sehingga terdapat kegagalan yang cukup tinggi dalam pengenalan model terhadap *dataset* pengujian.

4.3 Hasil Evaluasi Dataset

Pada hasil evaluasi ini dilakukan dengan menggunakan model CNN yang dimiliki terhadap *dataset* pengujian dari ketiga area pulau. Tujuannya dengan melakukan prediksi terhadap kedua *class,* yaitu lahan sawah dan bukan lahan sawah. Sehingga dapat dilakukan perbandingan *class* diantara *dataset* pengujian dengan hasil prediksi dari model. Perbandingan ini dinyatakan dalam bentuk nilai 0 dan 1 yang menandakan identitas *class* pada citra sejumlah *dataset* pengujian yang sebanyak 20 citra di ketiga area.

4.3.1 Evaluasi Performa Model

Dalam mengevaluasi perfoma model yang dimiliki, dapat dilakukan dengan menggolongkan hasil klasifikasi pada perbandingan *class* antara *dataset* pengujian aktual dengan hasil prediksi dari model kedalam empat kategori, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Kemudian dari empat kategori tersebut digunakan sebagai parameter pengukuran kinerja dalam metode perhitungan akurasi atau sistem pendukung pengambilan keputusan yang disebut *confusion matrix*. Pada tabel 4.4 dibawah ini memperlihatkan bentuk *confusion matrix* pada *dataset* pengujian.

Tabel 4. 4 Confusion Matrix Terhadap Model pada Dataset Pengujian

n = Jumlah Citra	**Dataset Pengujian**	
	Prediksi	**Prediksi**
	Negatif (Bukan Sawah)	**Positif (Sawah)**
Aktual	*True Negative* (TN)	*False Positive* (FP)
Negatif (Bukan Sawah)		
Aktual	*False Negative* (FN)	*True Positive* (FN)
Positif (Sawah)		

Untuk mengklasifikasikan data evaluasi dan komparasi pada *confusion matrix*, peneliti melakukan pengujian yang diperoleh pada empat perhitungan klasifikasi. Klasifikasi tersebut, yaitu klasifikasi akurasi (*classification accuracy*), presisi (*precision rate*), *recall rate*, dan F1 *score*. Berikut ini merupakan evaluasi perorma model terhadap setiap *dataset* dengan ketinggian yang berbeda dan arsitektur yang ditentukan.

1. Evaluasi perfoma model terhadap *dataset* pengujian pada ketinggian 100 meter menggunakan arsitektur *LeNet-5* Modifikasi

Hasil prediksi pada *dataset* pengujian diperoleh dengan melakukan pengujian pada model terhadap prediksi *class* antara citra sawah dan bukan sawah. Dari 20 citra pada masing-masing ketiga area pulau yang diperoleh prediksinya, kemudian dibandingkan dengan *dataset* pengujian aktual menjadi runtun baris terhadap *class*, yaitu 0 dan 1. Sehingga dapat dilakukan *confusion matrix* yang memiliki pengukuran empat kategori parameter untuk menguji performa model. Pada tabel 4.5 merupakan hasil pengukuran parameter terhadap perbandingan *dataset* pengujian aktual dengan hasil prediksinya.

Tabel 4. 5 Hasil Pengukuran Parameter pada Confusion Matrix

No.	**Objek Citra Ketinggian 100 Meter**	***Confusion Matrix***				**Jumlah Citra**	***Class***	
	Area Penelitian (citra)	**TP**	**TN**	**FP**	**FN**		**Bukan Sawah**	**Sawah**
1.	Pulau Jawa	8	10	0	2	20	10	10
2.	Pulau Kalimantan	8	10	0	2	20	10	10
3.	Pulau Sumatra	10	9	1	0	20	10	10

Berdasarkan tabel 4.5 diatas bahwa hasil prediksi yang salah pada *false positive* (FP) maupun *false negative* (FN) di ketiga area penelitian cukup kecil. Hal ini menandakan model dapat memprediksi dengan baik dan benar diantara *dataset* pengujian aktual dengan hasil prediksi terhadap dua *class* yang diujikan yaitu citra sawah dan citra bukan sawah.

Adapun klasifikasikan data evaluasi dan komparasi pada *confusion matrix* dalam melakukan pengujian, sehingga diperoleh empat perhitungan klasifikasi, yaitu klasifikasi akurasi (*classification accuracy*), presisi (*precision rate*), *recall rate*, dan F1 *score* sebagai berikut ini :

a. Klasifikasi akurasi (*classification accuracy*)

   Klasifisikasi akurasi merupakan rasio prediksi yang benar (positif maupun negative) yaitu TP dan TN dengan keseluruhan data. Perhitungan secara matematis dapat dilihat pada rumus 4.1 berikut :

$$Classification\ Accuracy = (TP\ +\ TN)\ /\ (TP\ +\ TN\ +\ FP\ +\ FN) \qquad (4.1)$$

Apabila kita lakukan perhitungan akurasinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Akurasi pada Pulau Jawa

$$Classification\ Accuracy = \frac{(8+10)}{(8+10+0+2)} = \frac{18}{20} = 0.9$$

$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

(2) Akurasi pada Pulau Kalimantan

$$Classification\ Accuracy = \frac{(8+10)}{(8+10+0+2)} = \frac{18}{20} = 0.9$$

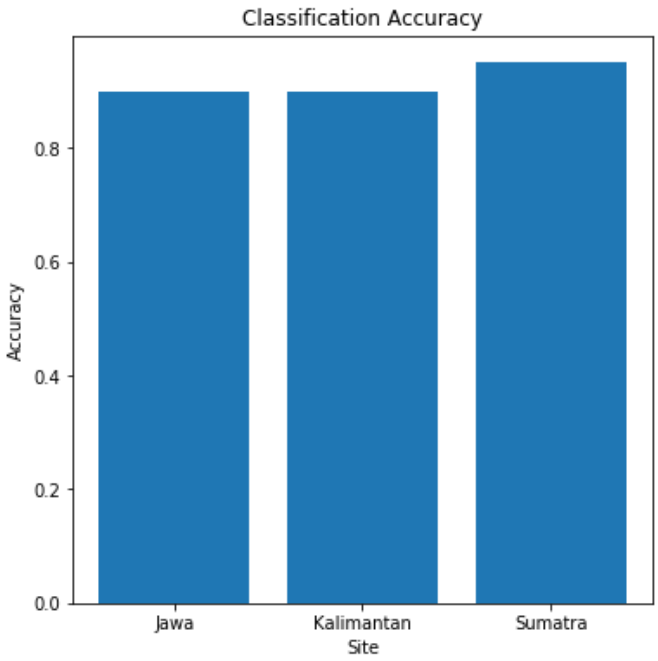
$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

(3) Akurasi pada Pulau Sumatra

$$Classification\ Accuracy = \frac{(10+9)}{(10+9+1+0)} = \frac{19}{20} = 0.95$$

$$\%\ Classification\ Accuracy = 0.95\ x\ 100\ \% = 95\ \%$$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik akurasi pada Gambar 4.9 berikut ini :



Gambar 4. 9 Grafik Klasifikasi Akurasi Antar Ketiga Pulau

Pada Gambar 4.9 menunjukan akurasi ketiga pulau sangat baik karena mendekati nilai 1 dan pada akurasi pada pulau Sumatra yang tertinggi dengan akurasi mencapai 95%. Hal ini karena pada pulau sumatera hanya memiliki kesalahan prediksi berjumlah satu citra.

b. Presisi (*precision rate*)

Presisi merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif, sebaliknya jika presisi yang diuji dengan citra bukan sawah maka prediksi benar negative dibandingkan dengan keseluruhan hasil yang diprediksi negative juga. Berikut ini merupakan perhitungan pada presisi antara citra sawah dan bukan sawah pada rumus (4.2) dan (4.3).

$Precision\ Positive = TP\ /\ (TP\ +\ FP)$ (4.2)

$Precision\ Negative = TN\ /\ (TN\ +\ FN)$ (4.3)

Apabila kita lakukan perhitungan presisinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Presisi pada Pulau Jawa

- $Presisi\ Citra\ Sawah = \frac{(8)}{(8+0)} = \frac{8}{8} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+2)} = \frac{10}{12} = 0.833$

  $\%\ Presisi\ Citra\ Sawah =\ 0.833\ x\ 100\ \% = 83.3\ \%$

(2) Presisi pada Pulau Kalimantan

- $Presisi\ Citra\ Sawah = \frac{(8)}{(8+0)} = \frac{8}{8} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+2)} = \frac{10}{12} = 0.833$

  $\%\ Presisi\ Citra\ Sawah =\ 0.833\ x\ 100\ \% = 83.3\ \%$

(3) Presisi pada Pulau Sumatra

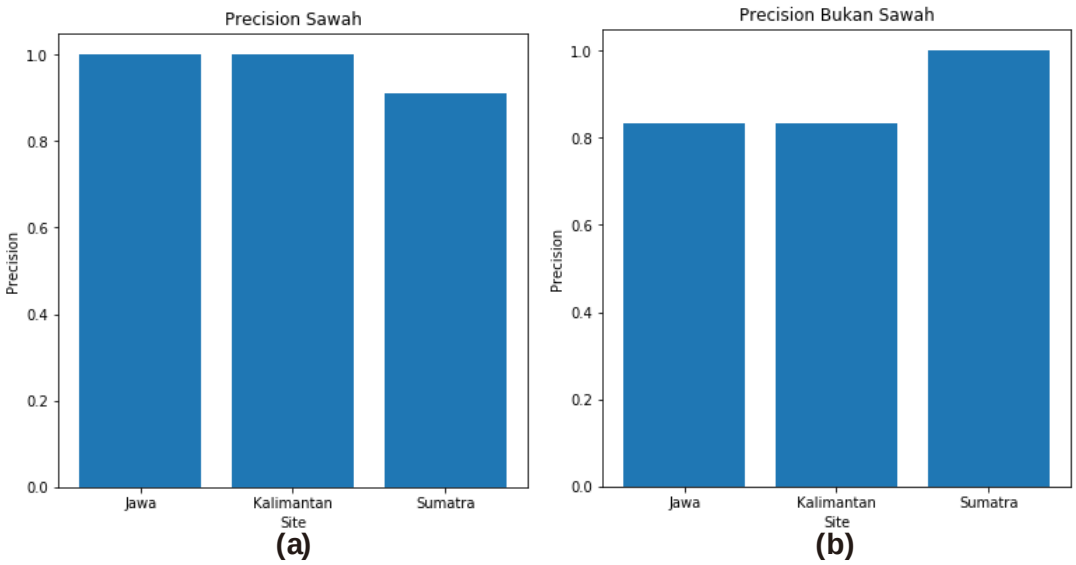
- $Presisi\ Citra\ Sawah = \frac{(10)}{(10+1)} = \frac{10}{11} = 0.909$

  $\%\ Presisi\ Citra\ Sawah = 0.909\ x\ 100\ \% = 90.9\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+0)} = \frac{9}{9} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik presisi pada Gambar 4.10 berikut ini :



Gambar 4. 10 Grafik Presisi Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.10 menunjukan presisi ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1 dan pada presisi dalam memprediksi citra sawah di pulau Jawa dan Kalimantan didapatkan hasil yang 100% baik. Namun kedua area tersebut memiliki presisi yang menurun ketika memprediksi citra bukan sawah dimana hanya mencapai 88.3%.

c. *Recall rate*

Recall rate merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif dan prediksi salah yang negatif, sebaliknya jika *recall rate* yang diuji prediksinya dengan citra bukan sawah maka prediksi benar negatif

dibandingkan dengan keseluruhan hasil yang diprediksi negative dan prediksi salah yang positif. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.4) dan (4.5).

$$Recall\ Rate\ Positive = TP\ /\ (TP\ +\ FN) \qquad (4.4)$$

$$Recall\ Rate\ Negative = TN\ /\ (TN\ +\ FP) \qquad (4.5)$$

Apabila kita lakukan perhitungan *recall rate* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *Recall rate* pada Pulau Jawa

- $Recall\ rate\ Citra\ Sawah = \frac{(8)}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.8\ x\ 100\ \% = 80\ \%$

- $Recall\ rateBukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(2) *Recall rate* pada Pulau Kalimantan

- $Recall\ rate\ Citra\ Sawah = \frac{(8\ )}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.8\ x\ 100\ \% = 80\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(3) *Recall rate* pada Pulau Sumatra

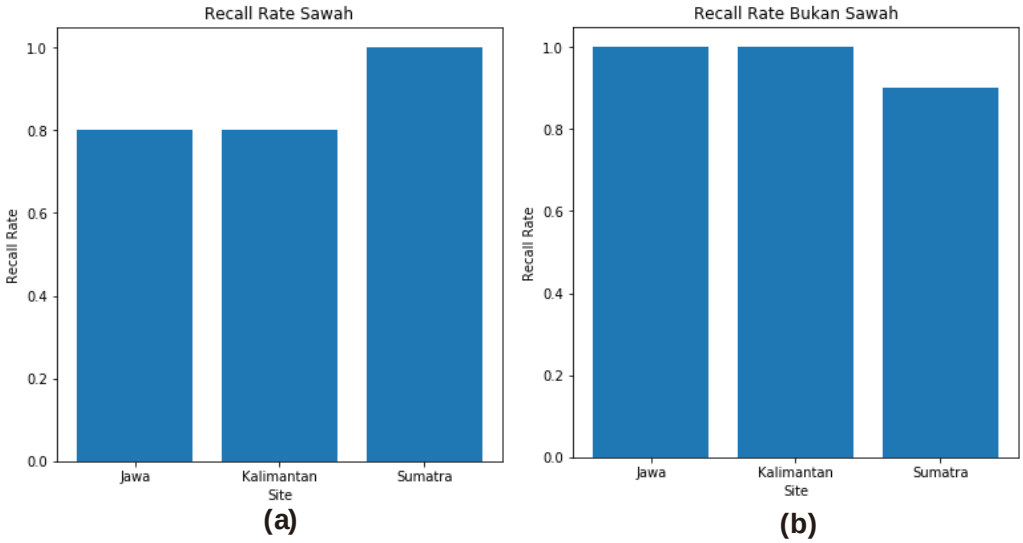
- $Recall\ rate\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+1)} = \frac{9}{10} = 0.9$

  $\%\ Recall\ rate\ Citra\ Sawah = 1\ x\ 0.9\ \% = 90\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.11 berikut ini :



Gambar 4. 11 Grafik Recall Rate Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.11 menunjukan grafik ketiga pulau cukup baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1 dan pada *recall rate* pada pulau Sumatra dengan citra sawah yang tertinggi mencapai 100% sedangkan *recall rate* pada citra bukan sawah yang tertinggi terdapat pada pulau Jawa dan Kalimantan mencapai 100%.

d. F1 *score*

F1 score merupakan evaluasi keseimbangan dan perbandingan rata-rata presisi dan *recall rate*. Pada F1 *score* ini memiliki dua jenis yaitu pada

citra sawah dan bukan sawah. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.6).

$$F1\ Score\ = (2\ x\ (RR\ x\ Precission)\ /(RR +\ Precission)) \qquad (4.6)$$

Apabila kita lakukan perhitungan *F1 score* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *F1 score* pada Pulau Jawa

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 1}{0.8\ +1}\right) = 0.88$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.88\ x\ 100\ \% = 88\%$

- $F1\ score\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.833}{1\ +0.833}\right) = 0.908$

  $\%\ F1\ score\ Citra\ Sawah =\ 0.908\ x\ 100\ \% = 90.8\ \%$

(2) *F1 score* pada Pulau Kalimantan

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 1}{0.8\ +1}\right) = 0.88$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.88\ x\ 100\ \% = 88\%$

  $F1\ scoreCitra\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.833}{1\ +0.833}\right) = 0.908$

  $\%\ F1\ score\ \ Citra\ Sawah =\ 0.908\ x\ 100\ \% = 90.8\ \%$

(3) F1 *score* pada Pulau Sumatra

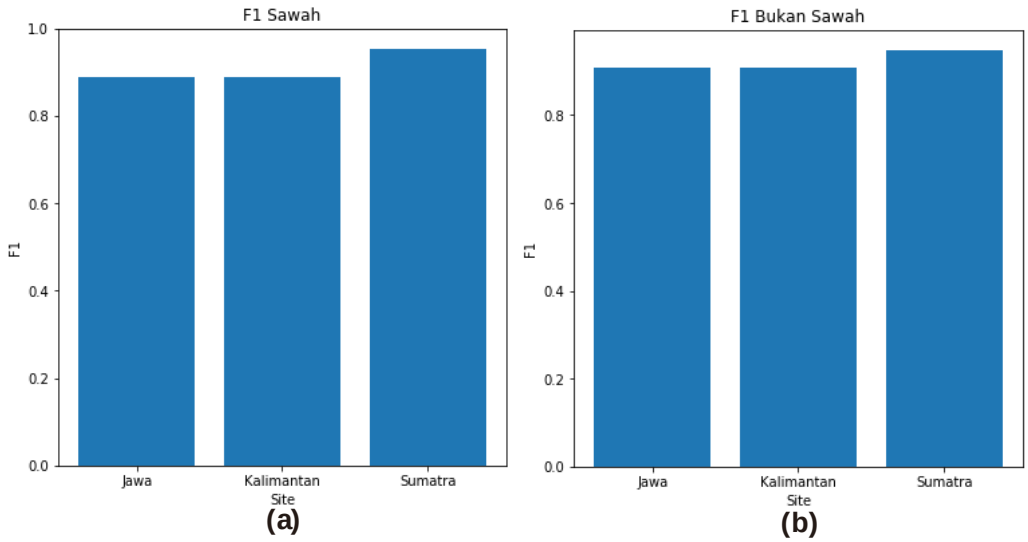
- $F1\ score\ Citra\ Sawah = 2\left(\frac{1\ x\ 0.909}{1\ +\ 0.909}\right) = 0.952$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.952\ x\ 100\ \% = 95.2\%$

  $F1\ score\ \ Citra\ Bukan\ Sawah = 2\left(\frac{0.9\ \ x\ 1}{0.9\ +1}\right) = 0.947$

  $\%F1\ score\ \ Citra\ Sawah = 0.947\ x\ 100\ \% = 94.7\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.12 berikut ini :



Gambar 4. 12 Grafik F1 Score Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.12 menunjukan grafik F1 *score* ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1. Terutama pada hasil F1 *score* dalam prediksi di pulau Sumatra diantara citra sawah maupun bukan sawah memiliki persentase diatas 90%.

2. Evaluasi perfoma model terhadap *dataset* pengujian pada ketinggian 300 meter menggunakan arsitektur *LeNet-5* Modifikasi

Pada *dataset* pengujian ketinggian 300 meter dilakukan evaluasi seperti pada pengujian sebelumnya. Hasil prediksi pada *dataset* pengujian diperoleh dengan melakukan pengujian pada model terhadap prediksi *class* antara citra sawah dan bukan sawah.

Pada tabel 4.6 menunjukkan *confusion matrix* jumlah *dataset* pada pengujian dengan hasil prediksi yang telah didapatkan empat pengukuran parameter sebanyak 20 citra di setiap area yang ditentukan.

Tabel 4. 6 Hasil Pengukuran Parameter pada Confusion Matrix

No.	**Objek Citra Ketinggian 300 Meter**	***Confusion Matrix***				**Jumlah Citra**	***Class***	
	Area Penelitian (citra)	**TP**	**TN**	**FP**	**FN**		**Bukan Sawah**	**Sawah**
1.	Pulau Jawa	9	10	0	1	20	10	10
2.	Pulau Kalimantan	10	8	2	0	20	10	10
3.	Pulau Sumatra	10	8	2	0	20	10	10

Berdasarkan tabel 4.6 diatas bahwa hasil prediksi yang salah pada *false positive* (FP) maupun *false negative* (FN) di ketiga area penelitian cukup kecil. Hal ini menandakan model dapat memprediksi dengan baik dan benar diantara *dataset* pengujian aktual dengan hasil prediksi terhadap dua *class* yang diujikan yaitu citra sawah dan citra bukan sawah.

Adapun klasifikasikan data evaluasi dan komparasi pada *confusion matrix* dalam melakukan pengujian, sehingga diperoleh empat perhitungan klasifikasi, yaitu klasifikasi akurasi (*classification accuracy*), presisi (*precision rate*), *recall rate*, dan F1 *score* sebagai berikut ini :

a. Klasifikasi akurasi (*classification accuracy*)

Klasifisikasi akurasi merupakan rasio prediksi yang benar (positif maupun negative) yaitu TP dan TN dengan keseluruhan data. Perhitungan secara matematis dapat dilihat pada rumus 4.1 berikut :

$$Classification\ Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (4.1)$$

Apabila kita lakukan perhitungan akurasinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Akurasi pada Pulau Jawa

$$Classification\ Accuracy = \frac{(9+10)}{(9+10+0+1)} = \frac{19}{20} = 0.95$$

$$\%\ Classification\ Accuracy = 0.95\ x\ 100\ \% = 95\ \%$$

(2) Akurasi pada Pulau Kalimantan

$$Classification\ Accuracy = \frac{(10+8)}{(10+8+2+0)} = \frac{18}{20} = 0.9$$

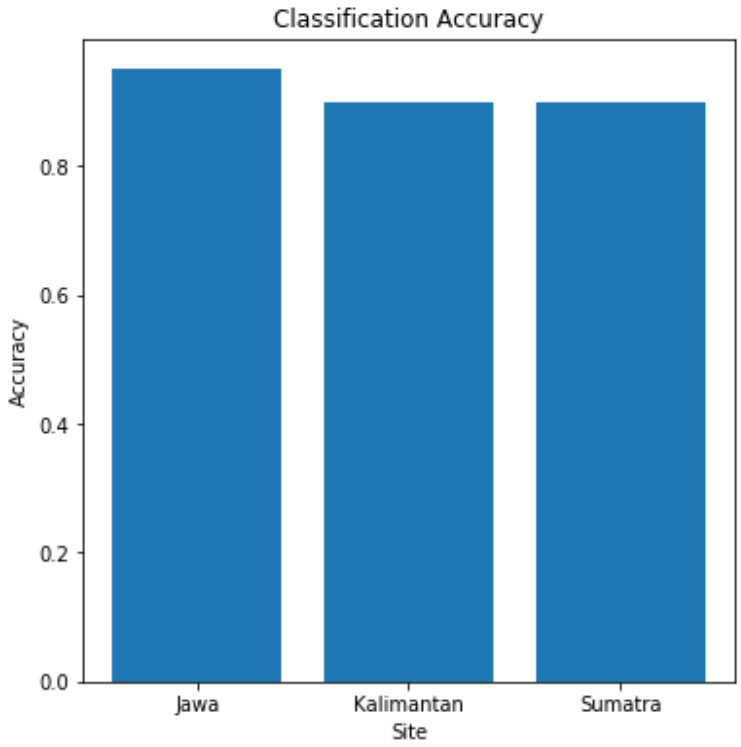
$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

(3) Akurasi pada Pulau Sumatra

$$Classification\ Accuracy = \frac{(10+8)}{(10+8+2+0)} = \frac{18}{20} = 0.9$$

$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik akurasi pada Gambar 4.13 berikut ini :



Gambar 4. 13 Grafik Klasifikasi Akurasi Antar Ketiga Pulau

Pada gambar 4.13 diatas menunjukan ketiga area memiliki akurasi yang sangat baik yang mana dapat mendekati nilai 1 atau mencapai persentase diatas 90% dari 20 *dataset* pengujian aktual yang diprediksikan pada masing-masing objek area citra. Sehingga pada *dataset* pelatihan dan pengujian yang memiliki ketinggian ini dapat dinyatakan evaluasi model dan klasifikasinya berhasil.

b. Presisi

Presisi merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif, sebaliknya jika presisi yang diuji dengan citra bukan sawah maka prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negatif juga.

Berikut ini merupakan perhitungan pada presisi antara citra sawah dan bukan sawah pada rumus (4.2) dan (4.3).

$Precision\ Positive = TP\ /\ (TP\ +\ FP)$ (4.2)

$Precision\ Negative = TN\ /\ (TN\ +\ FN)$ (4.3)

Apabila kita lakukan perhitungan presisinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Presisi pada Pulau Jawa

- $Presisi\ Citra\ Sawah = \frac{(9)}{(9+0)} = \frac{9}{9} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+1)} = \frac{10}{11} = 0.909$

  $\%\ Presisi\ Citra\ Sawah = \ 0.909\ x\ 100\ \% = 90.9\ \%$

(2) Presisi pada Pulau Kalimantan

- $Presisi\ Citra\ Sawah = \frac{(10)}{(10+2)} = \frac{10}{12} = 0.833$

  $\%\ Presisi\ Citra\ Sawah = \ 0.833\ x\ 100\ \% = 83.3\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(8)}{(8+0)} = \frac{8}{8} = 1$

  $\%\ Presisi\ Citra\ Sawah = \ 1\ x\ 100\ \% = 100\ \%$

(3) Presisi pada Pulau Sumatra

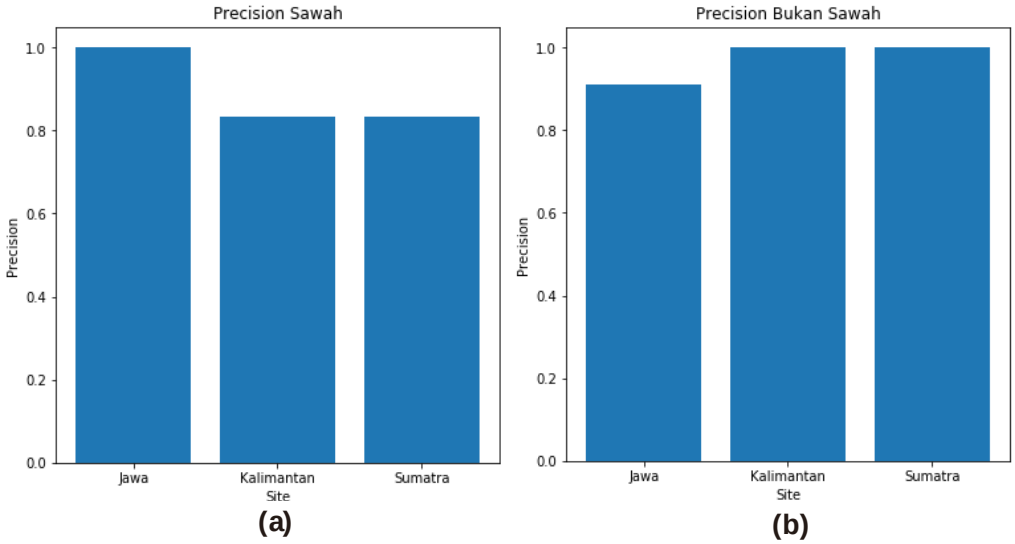
- $Presisi\ Citra\ Sawah = \frac{(10)}{(10+2)} = \frac{10}{12} = 0.833$

  $\%\ Presisi\ Citra\ Sawah = \ 0.833\ x\ 100\ \% = 83.3\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(8)}{(8+0)} = \frac{8}{8} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik presisi pada Gambar 4.14 berikut ini :



Gambar 4. 14 Grafik Presisi Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada gambar 4.14 diatas menunjukkan tingkat presisi terhadap prediksi citra sawah dan citra bukan sawah yang cukup baik. Ketiga area pulau yang diujikan memperoleh hasil presisi persentase diatas 80%. Pada prediksi yang benar terhadap citra sawah dapat mencapai 100% pada area pulau Jawa, sedangkan presentase yang sempurna didapatkan juga pada area pulau Kalimantan dan Sumatra pada citra bukan sawah.

c. *Recall rate*

Recall rate merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif dan

prediksi salah yang negatif, sebaliknya jika *recall rate* yang diuji prediksinya dengan citra bukan sawah maka prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negative dan prediksi salah yang positif. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.4) dan (4.5).

$$Recall\ Rate\ Positive = TP\ /\ (TP\ +\ FN) \qquad (4.4)$$

$$Recall\ Rate\ Negative = TN\ /\ (TN\ +\ FP) \qquad (4.5)$$

Apabila kita lakukan perhitungan *recall rate* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *Recall rate* pada Pulau Jawa

- $Recall\ rate\ Citra\ Sawah = \frac{(9)}{(9+1)} = \frac{9}{10} = 0.9$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.9\ x\ 100\ \% = 90\ \%$

- $Recall\ rate\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(2) *Recall rate* pada Pulau Kalimantan

- $Recall\ rate\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(8)}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 0.8\ x\ 100\ \% = 80\ \%$

(3) *Recall rate* pada Pulau Sumatra

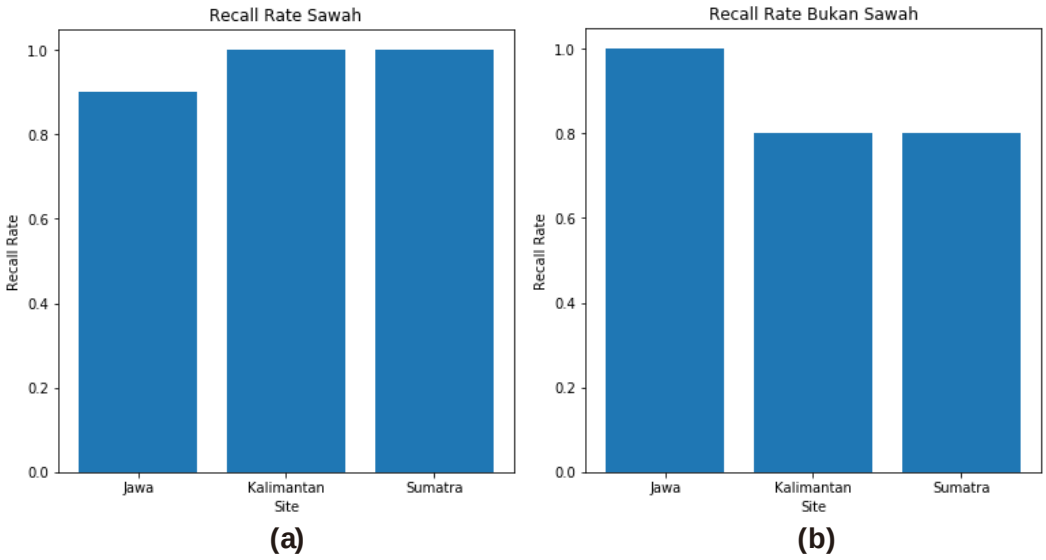
- $Recall\ rate\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rateCitra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(8)}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.8\ x\ 100\ \% = 80\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.15 berikut ini :



Gambar 4. 15 Grafik Recall Rate Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Berdasarkan gambar 4.15 diperlihatkan hasil grafik perbandingan dari ketiga pulau di masing-masing citra. Hasil prediksi benar pada *recall rate* citra sawah memiliki tingkat prediksi yang tinggi yang mana diperoleh persentase 100% di pulau Kalimantan dan Sumatra. Sedangkan pada *recall*

rate citra bukan sawah yang mencapai persentase tertinggi terdapat pada pulau Jawa.

d. F1 *score*

F1 score merupakan evaluasi keseimbangan dan perbandingan rata-rata presisi dan *recall rate*. Pada F1 *score* ini memiliki dua jenis yaitu pada citra sawah dan bukan sawah. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.6).

$$F1\ Score\ = (2\ x\ (RR\ x\ Precission)\ /(RR +\ Precission)) \qquad (4.6)$$

Apabila kita lakukan perhitungan *F1 score* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *F1 score* pada Pulau Jawa

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 1}{0.8\ +1}\right) = 0.947$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.947\ x\ 100\% = 94.7\%$

- $F1\ score\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.909}{1\ +0.909}\right) = 0.952$

  $\%\ F1\ score\ Citra\ Sawah =\ 0.952\ x\ 100\ \% = 95.2\ \%$

(2) *F1 score* pada Pulau Kalimantan

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 0.833}{0.8\ +\ 0.833}\right) = 0.909$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.909\ x\ 100\ \% = 90.9\%$

  $F1\ score\ Citra\ Bukan\ Sawah = 2\left(\frac{0.8\ \ x\ 1}{0.8\ +1}\right) = 0.888$

  $\%\ F1\ score\ \ Citra\ Sawah =\ 0.888\ x\ 100\ \% = 88.8\ \%$

(3) *F1 score* pada Pulau Sumatra

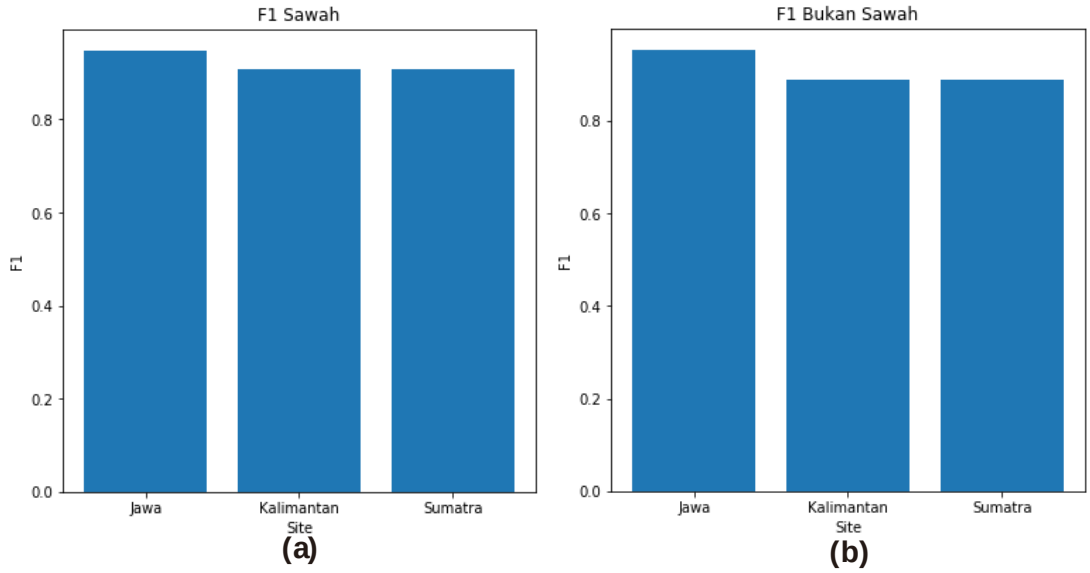
- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 0.833}{0.8 + 0.833}\right) = 0.952$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.952\ x\ 100\ \% = 95.2\%$

  $F1\ score\ \ Citra\ Bukan\ Sawah = 2\left(\frac{0.8\ \ x\ 1}{0.8 + 1}\right) = 0.888$

  $\%F1\ score\ \ Citra\ Sawah = 0.888\ x\ 100\ \% = 88.8\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.16 berikut ini :



Gambar 4. 16 Grafik F1 Score Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.16 menunjukan grafik F1 *score* ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1. Terutama pada pulau Jawa yang mendapatkan hasil F1 *score* citra sawah dan bukan sawah yang memiliki persentase 100%.

3. Evaluasi perfoma model terhadap *dataset* pengujian pada ketinggian 100 meter menggunakan arsitektur *VGG-16Net*

Pada *dataset* pengujian ketinggian 100 meter dilakukan evaluasi seperti pada pengujian sebelumnya. Hasil prediksi pada *dataset* pengujian diperoleh dengan melakukan pengujian pada model terhadap prediksi *class* antara citra sawah dan bukan sawah.

Pada tabel 4.7 menunjukkan *confusion matrix* jumlah *dataset* pada pengujian dengan hasil prediksi yang telah didapatkan empat pengukuran parameter sebanyak 20 citra di setiap area yang ditentukan.

Tabel 4. 7 Hasil Pengukuran Parameter pada Confusion Matrix

No.	**Objek Citra Ketinggian 100 Meter**	***Confusion Matrix***				**Jumlah Citra**	***Class***	
	Area Penelitian (citra)	**TP**	**TN**	**FP**	**FN**		**Bukan Sawah**	**Sawah**
1.	Pulau Jawa	9	10	0	1	20	10	10
2.	Pulau Kalimantan	7	10	0	3	20	10	10
3.	Pulau Sumatra	10	10	0	0	20	10	10

Pada tabel 4.7 diatas menunjukkan hasil prediksi yang salah pada *false positive* (FP) maupun *false negative* (FN) di ketiga area penelitian cukup kecil. Terutama pada objek citra di pulau Sumatra yang memperoleh yang 100% akurat antara citra sawah dan bukan sawah. Meskipun terdapat prediksi yang salah terhadap pengenalan citra bukan sawah pada area pulau Kalimantan yang memiliki *false negative* sebanyak tiga citra. Namun secara keseluruhan model masih dapat

dikatakan memprediksi dengan baik dan benar diantara *dataset* pengujian aktual dengan hasil prediksi terhadap dua *class* yang diujikan yaitu citra sawah dan citra bukan sawah.

Adapun klasifikasikan data evaluasi dan komparasi pada *confusion matrix* dalam melakukan pengujian, sehingga diperoleh empat perhitungan klasifikasi, yaitu klasifikasi akurasi (*classification accuracy*), presisi (*precision rate*), *recall rate*, dan F1 *score* sebagai berikut ini :

a. Klasifikasi akurasi (*classification accuracy*)

Klasifisikasi akurasi merupakan rasio prediksi yang benar (positif maupun negatif) yaitu TP dan TN dengan keseluruhan data. Perhitungan secara matematis dapat dilihat pada rumus 4.1 berikut :

$$Classification\ Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (4.1)$$

Apabila kita lakukan perhitungan akurasinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Akurasi pada Pulau Jawa

$$Classification\ Accuracy = \frac{(9+10)}{(9+10+0+21)} = \frac{19}{20} = 0.95$$

$$\%\ Classification\ Accuracy = 0.95\ x\ 100\ \% = 95\ \%$$

(2) Akurasi pada Pulau Kalimantan

$$Classification\ Accuracy = \frac{(7+10)}{(7+10+0+3)} = \frac{18}{20} = 0.9$$

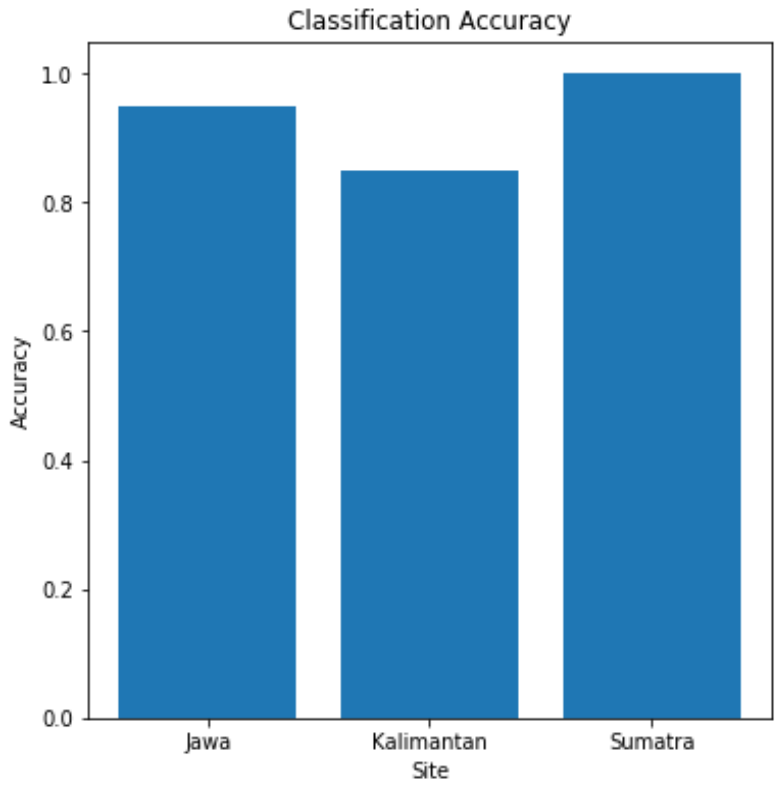
$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

(3) Akurasi pada Pulau Sumatra

$$Classification\ Accuracy = \frac{(10+10)}{(10+10+0+0)} = \frac{20}{20} = 1$$

$$\%\ Classification\ Accuracy = 1\ x\ 100\ \% = 100\ \%$$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik akurasi pada Gambar 4.17 berikut ini :



Gambar 4. 17 Grafik Klasifikasi Akurasi Antar Ketiga Pulau

Pada Gambar 4.17 menunjukkan hasil akurasi yang sangat baik diketiga area pulau yang diujikan. Masing-masing area dapat mencapai akurasi sebesar 90% dalam presentasenya. Akurasi tertinggi dicapai pada pulau Sumatra diperoleh presentase 100%, karena tidak terdapat prediksi yang salah pada *confusion matrix.*

b. Presisi

Presisi pada penelitian ini memiliki dua perediksi pada citra sawah dan citra bukan sawah. Pada citra sawah merupakan rasio prediksi benar positif yang dibandingkan dengan keseluruhan hasil yang diprediksi positif, sebaliknya jika presisi yang diuji dengan citra bukan sawah maka prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negatif juga. Berikut ini merupakan perhitungan pada presisi antara citra sawah dan bukan sawah pada rumus (4.2) dan (4.3).

$$Precision\ Positive = TP\ /\ (TP\ +\ FP) \qquad (4.2)$$

$$Precision\ Negative = TN\ /\ (TN\ +\ FN) \qquad (4.3)$$

Apabila kita lakukan perhitungan presisinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Presisi pada Pulau Jawa

- $Presisi\ Citra\ Sawah = \frac{(9)}{(9+0)} = \frac{9}{9} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+1)} = \frac{10}{11} = 0.909$

  $\%\ Presisi\ Citra\ Sawah = \ 0.909\ x\ 100\ \% = 90.9\ \%$

(2) Presisi pada Pulau Kalimantan

- $Presisi\ Citra\ Sawah = \frac{(7\ )}{(7+0)} = \frac{7}{7} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+3)} = \frac{10}{13} = 0.769$

  $\%\ Presisi\ Citra\ Sawah = \ 76.9\ x\ 100\ \% = 76.9\ \%$

(3) Presisi pada Pulau Sumatra

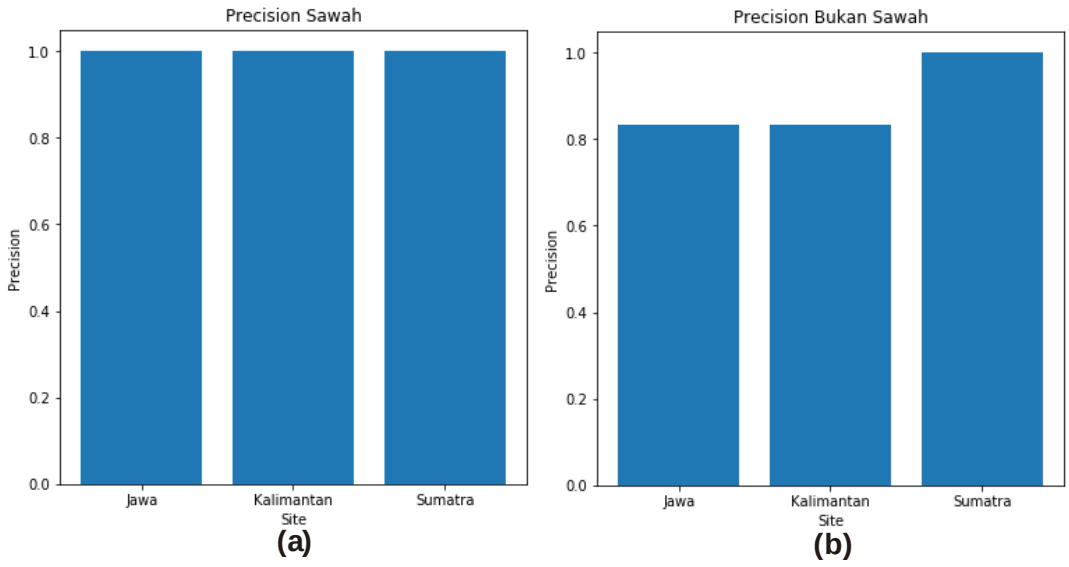
- $Presisi\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik presisi pada Gambar 4.18 berikut ini :



Gambar 4. 18 Grafik Presisi Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.18 menunjukkan presisi pada citra sawah dan bukan sawah. Presisi sawah mendapatkan prediksi yang sangat tinggi dengan presentasi ketiganya mencapai 100%. Sedangkan pada citra bukan sawah

presisi dengan presentase tertinggi diperoleh pada pulau Sumatra dengan presentase 100%. Karena pada pulau Sumatra tidak terdapat prediksi salah pada citra bukan sawah.

c. *Recall rate*

Recall rate merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif dan prediksi salah yang negatif, sebaliknya jika *recall rate* yang diuji prediksinya dengan citra bukan sawah maka prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negatif dan prediksi salah yang positif. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.4) dan (4.5).

$$Recall\ Rate\ Positive = TP\ /\ (TP\ +\ FN) \qquad (4.4)$$

$$Recall\ Rate\ Negative = TN\ /\ (TN\ +\ FP) \qquad (4.5)$$

Apabila kita lakukan perhitungan *recall rate* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *Recall rate* pada Pulau Jawa

- $Recall\ rate\ Citra\ Sawah = \frac{(9)}{(9+1)} = \frac{9}{10} = 0.9$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.9\ x\ 100\ \% = 90\ \%$

- $Recall\ rate\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(2) *Recall rate* pada Pulau Kalimantan

- $Recall\ rate\ Citra\ Sawah = \frac{(7)}{(7+3)} = \frac{7}{10} = 0.7$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.7\ x\ 100\ \% = 70\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(3) *Recall rate* pada Pulau Sumatra

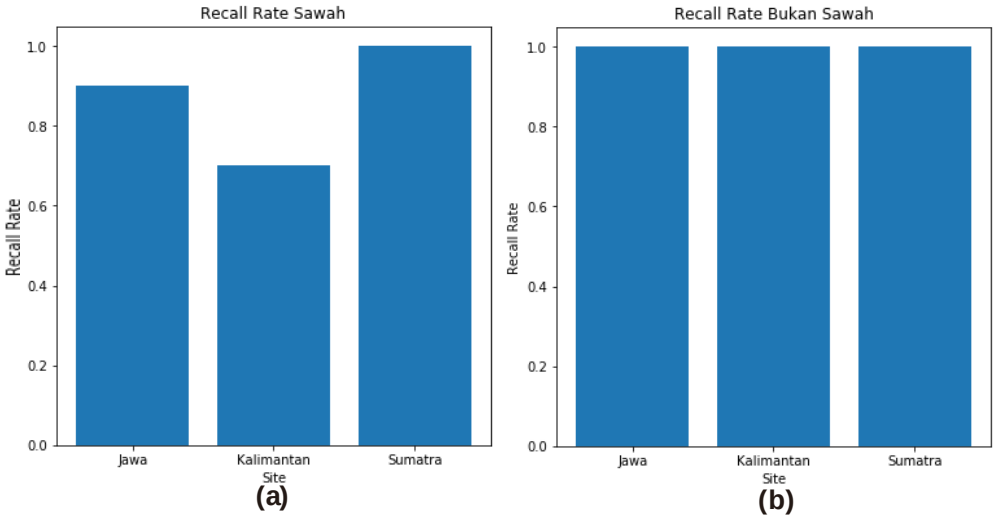
- $Recall\ rate\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rateCitra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.19 berikut ini :



Gambar 4. 19 Grafik Recall Rate Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Berdasarkan gambar 4.19 diperlihatkan hasil grafik perbandingan dari ketiga pulau di masing-masing citra. Hasil prediksi benar pada *recall rate* citra sawah memiliki tingkat prediksi yang tinggi yang mana diperoleh persentase 100% di pulau Sumatra, sedangkan presentase terendah terdapat pada pulau Kalimantan. Pada *recall rate* citra bukan sawah yang mencapai persentase tertinggi di ketiga pulau karena dapat dapat mencapai presentase 100%.

d. F1 *score*

F1 score merupakan evaluasi keseimbangan dan perbandingan rata-rata presisi dan *recall rate*. Pada F1 *score* ini memiliki dua jenis yaitu pada citra sawah dan bukan sawah. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.6).

$$F1\ Score\ = (2\ x\ (RR\ x\ Precission)\ /(RR +\ Precission)) \qquad (4.6)$$

Apabila kita lakukan perhitungan *F1 score* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *F1 score* pada Pulau Jawa

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.9x\ 1}{0.9+1}\right) = 0.88$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.88\ x\ 100\ \% = 88\%$

- $F1\ score\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.909}{1\ +0.909}\right) = 0.952$

  $\%\ F1\ score\ Citra\ Sawah =\ 0.952\ x\ 100\ \% = 95.2\ \%$

(2) *F1 score* pada Pulau Kalimantan

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.7\ x\ 1}{0.7+1}\right) = 0.88$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.88\ x\ 100\ \% = 88\%$

  $F1\ score\ Citra\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.769}{1\ +0.769}\right) = 0.869$

  $\%\ F1\ score\ \ Citra\ Sawah = \ 0.869\ x\ 100\ \% = 86.9\%$

(3) *F1 score* pada Pulau Sumatra

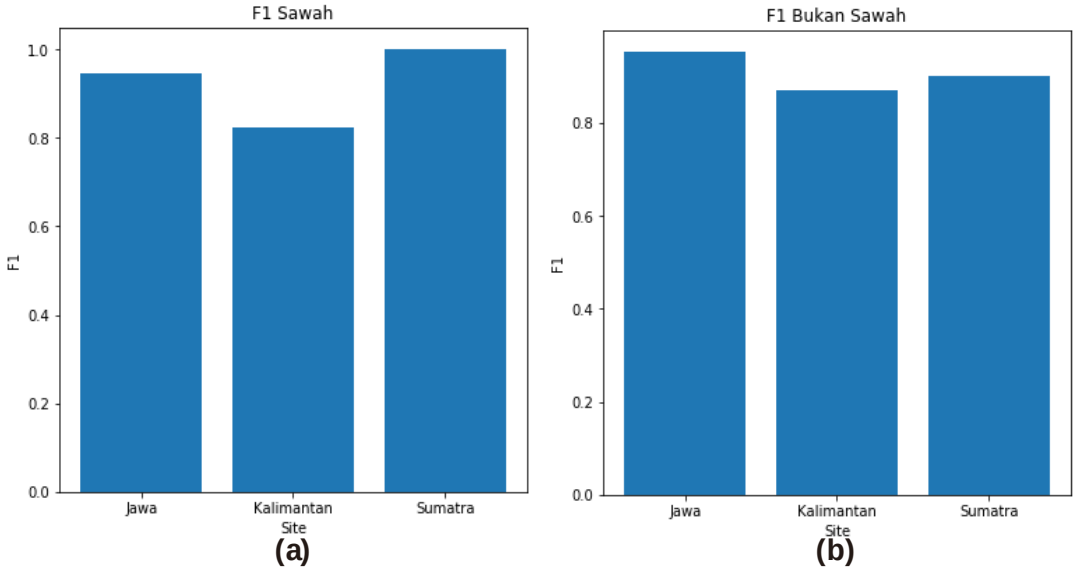
- $F1\ score\ Citra\ Sawah = 2\left(\frac{1\ x\ 1}{1+1}\right) = 1$

  $\%\ F1\ score\ \ Citra\ Sawah = 1\ x\ 100\ \% = 100\%$

  $F1\ score\ \ Citra\ Bukan\ Sawah = 2\left(\frac{1\ x\ 1}{1\ +1}\right) = 0.9$

  $\%F1\ score\ \ Citra\ Sawah = 0.9\ x\ 100\ \% = 90\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.20 berikut ini :



Gambar 4. 20 Grafik F1 Score Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.20 menunjukan grafik F1 *score* ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1 atau presentase 100%. Pada citra sawah memiliki presentase tertinggi pada pulau Sumatra, sedangkan pada F1 *score* citra bukan sawah mencapai 100% di pulau Jawa.

4. Evaluasi perfoma model terhadap *dataset* pengujian pada ketinggian 300 meter menggunakan arsitektur *VGG-16Net*

Pada *dataset* pengujian terakhir dalam evaluasi performa model dilakukan dengan kumpulan citra pada ketinggian 300 meter. Hasil prediksi pada *dataset* pengujian diperoleh dengan melakukan pengujian pada model terhadap prediksi *class* antara citra sawah dan bukan sawah.

Pada tabel 4.8 menunjukkan *confusion matrix* jumlah *dataset* pada pengujian dengan hasil prediksi yang telah didapatkan empat pengukuran parameter sebanyak 20 citra di setiap area yang ditentukan.

Tabel 4. 8 Hasil Pengukuran Parameter pada Confusion Matrix

No.	**Objek Citra Ketinggian 300 Meter**	***Confusion Matrix***				**Jumlah Citra**	***Class***	
	Area Penelitian (citra)	**TP**	**TN**	**FP**	**FN**		**Bukan Sawah**	**Sawah**
1.	Pulau Jawa	8	10	0	2	20	10	10
2.	Pulau Kalimantan	8	9	1	2	20	10	10
3.	Pulau Sumatra	10	9	1	0	20	10	10

Berdasarkan tabel 4.8 diatas diketahui bahwa hasil prediksi pada *confusion matrix* memiliki tingkat prediksi benar yang cukup tinggi dan dapat mengklasifikasinya dengan baik. Karena pada parameter yang menandakan prediksi yang salah berupa *false positive* (FP) dan *false negative* (FN) sangat sedikit. Meskipun pada hasil prediksi pada pulau Kalimantan terdapat tiga prediksi yang salah. Namun model dapat memprediksi dengan baik dan benar diantara *dataset* pengujian aktual dengan hasil prediksi terhadap dua *class* yang diujikan yaitu citra sawah dan citra bukan sawah.

Adapun klasifikasikan data evaluasi dan komparasi pada *confusion matrix* dalam melakukan pengujian, sehingga diperoleh empat perhitungan klasifikasi, yaitu klasifikasi akurasi (*classification accuracy*), presisi (*precision rate*), *recall rate*, dan F1 *score* sebagai berikut ini :

a. Klasifikasi akurasi (*classification accuracy*)

Klasifisikasi akurasi merupakan rasio prediksi yang benar (positif maupun negative) yaitu TP dan TN dengan keseluruhan data. Perhitungan secara matematis dapat dilihat pada rumus 4.1 berikut :

$$Classification\ Accuracy = (TP + TN) / (TP + TN + FP + FN) \qquad (4.1)$$

Apabila kita lakukan perhitungan akurasinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Akurasi pada Pulau Jawa

$$Classification\ Accuracy = \frac{(8+10)}{(8+10+0+2)} = \frac{18}{20} = 0.9$$

$$\%\ Classification\ Accuracy = 0.9\ x\ 100\ \% = 90\ \%$$

(2) Akurasi pada Pulau Kalimantan

$$Classification\ Accuracy = \frac{(8+9)}{(8+9+1+2)} = \frac{17}{20} = 0.85$$

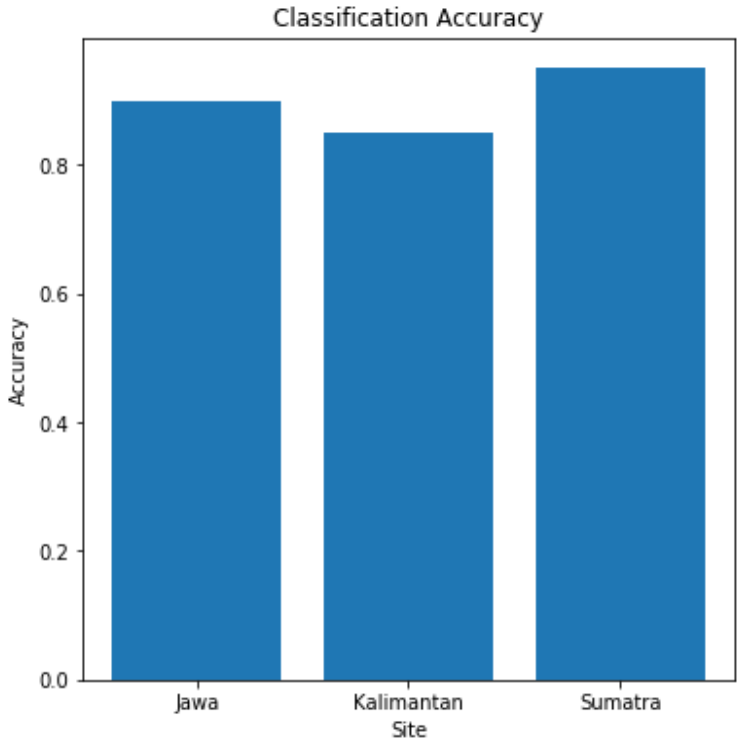
$$\%\ Classification\ Accuracy = 0.85\ x\ 100\ \% = 85\ \%$$

(3) Akurasi pada Pulau Sumatra

$$Classification\ Accuracy = \frac{(10+9)}{(10+9+1+0)} = \frac{19}{20} = 0.95$$

$$\%\ Classification\ Accuracy = 0.95\ x\ 100\ \% = 95\ \%$$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik akurasi pada Gambar 4.21 berikut ini :



Gambar 4. 21 Grafik Klasifikasi Akurasi Antar Ketiga Pulau

Pada Gambar 4.21 menunjukan akurasi ketiga pulau sangat baik karena mendekati nilai 1 dan pada akurasi pada pulau Sumatra yang tertinggi dengan akurasi mencapai 95%. Hal ini karena pada pulau sumatera hanya memiliki kesalahan prediksi berjumlah satu citra.

b. Presisi

Presisi merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif, sebaliknya jika presisi yang diuji dengan citra bukan sawah maka prediksi benar negative dibandingkan dengan keseluruhan hasil yang diprediksi negative juga. Berikut ini merupakan perhitungan pada presisi antara citra sawah dan bukan sawah pada rumus (4.2) dan (4.3).

$$Precision\ Positive = TP\ /\ (TP\ +\ FP) \qquad (4.2)$$

$$Precision\ Negative = TN\ /\ (TN\ +\ FN) \qquad (4.3)$$

Apabila kita lakukan perhitungan presisinya terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) Presisi pada Pulau Jawa

- $Presisi\ Citra\ Sawah = \frac{(8\ )}{(8+0)} = \frac{8}{8} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(10)}{(10+2)} = \frac{10}{12} = 0.833$

  $\%\ Presisi\ Citra\ Sawah =\ 0.833\ x\ 100\ \% = 83.3\ \%$

(2) Presisi pada Pulau Kalimantan

- $Presisi\ Citra\ Sawah = \frac{(8)}{(8+1)} = \frac{8}{9} = 0.888$

  $\%\ Presisi\ Citra\ Sawah = 0.888\ x\ 100\ \% = 88.8\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+2)} = \frac{9}{11} = 0.818$

  $\%\ Presisi\ Citra\ Sawah = \ 0.818\ x\ 100\ \% = 81.8\ \%$

(3) Presisi pada Pulau Sumatra

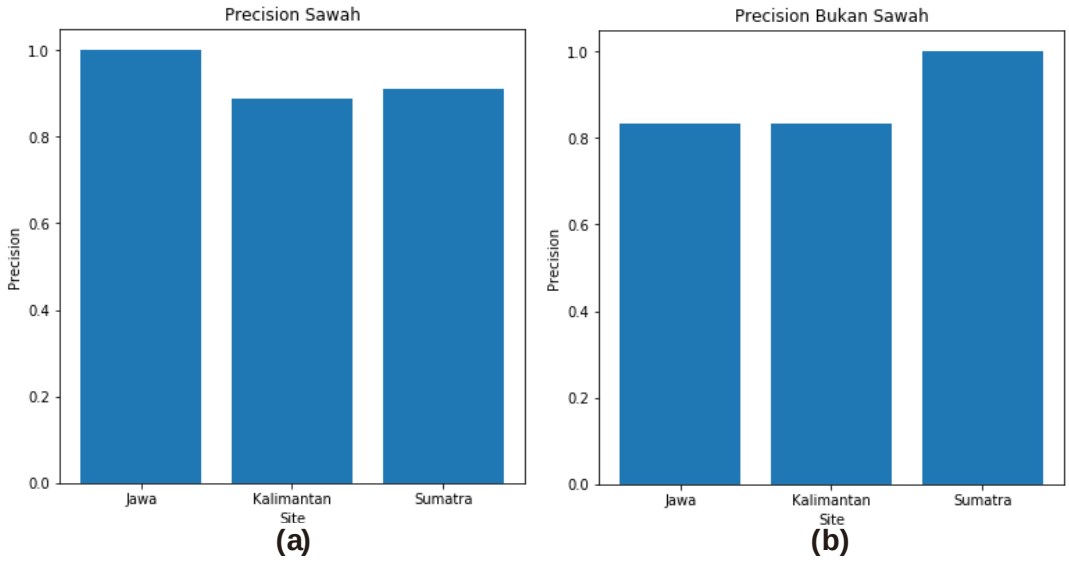
- $Presisi\ Citra\ Sawah = \frac{(10)}{(10+1)} = \frac{10}{11} = 0.909$

  $\%\ Presisi\ Citra\ Sawah = 0.909\ x\ 100\ \% = 90.9\ \%$

- $Presisi\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+0)} = \frac{9}{9} = 1$

  $\%\ Presisi\ Citra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik presisi pada Gambar 4.22 berikut ini :



Gambar 4. 22 Grafik Presisi Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.22 menunjukan presisi ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1 dan pada presisi pada pulau Jawa yang tertinggi dengan presisi citra sawah mencapai 100% sedangkan pada citra bukan sawah yang terdapat dipulau Sumatra mencapai 100%.

c. *Recall rate*

Recall rate merupakan rasio prediksi benar positif jika pada citra sawah dibandingkan dengan keseluruhan hasil yang diprediksi positif dan prediksi salah yang negatif, sebaliknya jika *recall rate* yang diuji prediksinya dengan citra bukan sawah maka prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negative dan prediksi salah yang positif. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.4) dan (4.5).

$$Recall\ Rate\ Positive = TP\ /\ (TP\ +\ FN) \qquad (4.4)$$

$$Recall\ Rate\ Negative = TN\ /\ (TN\ +\ FP) \qquad (4.5)$$

Apabila kita lakukan perhitungan *recall rate* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *Recall rate* pada Pulau Jawa

- $Recall\ rate\ Citra\ Sawah = \frac{(8)}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.8\ x\ 100\ \% = 80\ \%$

- $Recall\ rate\ Bukan\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rate\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(2) *Recall rate* pada Pulau Kalimantan

- $Recall\ rate\ Citra\ Sawah = \frac{(8)}{(8+2)} = \frac{8}{10} = 0.8$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.8\ x\ 100\ \% = 80\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+1)} = \frac{9}{10} = 0.9$

  $\%\ Recall\ rate\ Citra\ Sawah = 0.9\ x\ 100\ \% = 90\ \%$

(3) *Recall rate* pada Pulau Sumatra

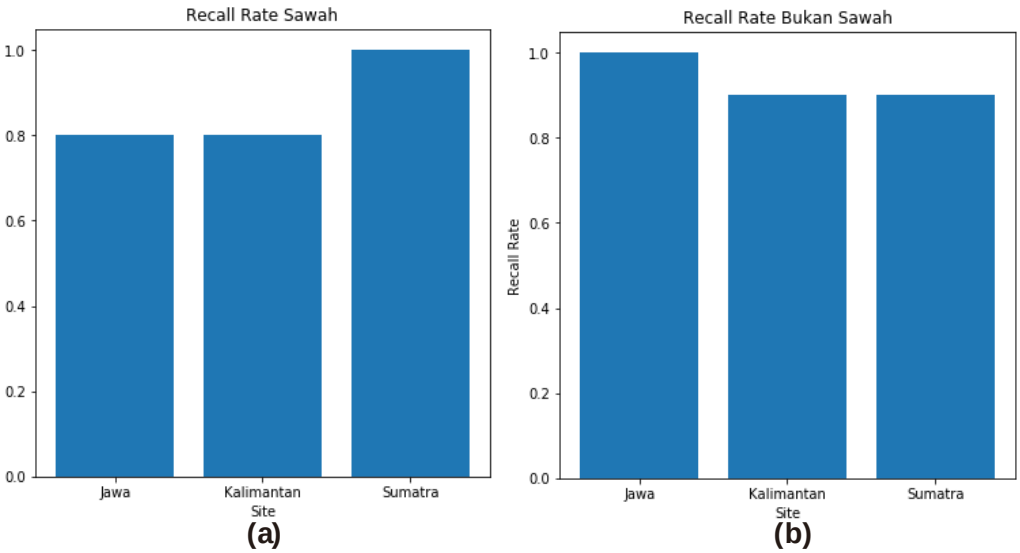
- $Recall\ rate\ Citra\ Sawah = \frac{(10)}{(10+0)} = \frac{10}{10} = 1$

  $\%\ Recall\ rateCitra\ Sawah = 1\ x\ 100\ \% = 100\ \%$

- $Recall\ rate\ Citra\ Bukan\ Sawah = \frac{(9)}{(9+1)} = \frac{9}{10} = 0.9$

  $\%\ Recall\ rate\ Citra\ Sawah = 1\ x\ 0.9\ \% = 90\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.23 berikut ini :



Gambar 4. 23 Grafik Recall Rate Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.23 menunjukan perbandingan grafik presisi ketiga pulau cukup baik antara citra sawah dan yang bukan sawah. Namun recall rate citra sawah yang terdapat pada pulau Jawa dan Kalimantan hanya dapat mencapai prediksi benar mencapai 80%. Sedangkan pada pulau Kalimantan dan Sumatra diperoleh presentase *recall rate* citra bukan sawah sebesar 90%.

d. F1 *Score*

F1 score merupakan evaluasi keseimbangan dan perbandingan rata-rata presisi dan *recall rate*. Pada F1 *score* ini memiliki dua jenis yaitu pada citra sawah dan bukan sawah. Berikut ini merupakan perhitungan pada *recall rate* antara citra sawah dan bukan sawah pada rumus (4.6).

$$F1\ Score\ = (2\ x\ (RR\ x\ Precission)\ /(RR +\ Precission)) \qquad (4.6)$$

Apabila kita lakukan perhitungan *F1 score* terhadap parameter yang didapatkan, maka diperoleh hasil sebagai berikut :

(1) *F1 score* pada Pulau Jawa

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 1}{0.8\ +1}\right) = 0.88$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.88\ x\ 100\ \% = 88\%$

- $F1\ score\ Bukan\ Sawah = 2\left(\frac{1\ x\ 0.833}{1\ +0.833}\right) = 1$

  $\%\ F1\ score\ Citra\ Sawah =\ 1\ x\ 100\ \% = 100\ \%$

(2) *F1 score* pada Pulau Kalimantan

- $F1\ score\ Citra\ Sawah = 2\left(\frac{0.8\ x\ 0.8}{0.8\ +\ 0.8}\right) = 0.842$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.842\ x\ 100\ \% = 84.2\%$

$$F1\ score\ Citra\ Bukan\ Sawah = 2\left(\frac{0.9\ x\ 0.818}{0.9 + 0.818}\right) = 0.857$$

$$\%\ F1\ score\ \ Citra\ Sawah = \ 0.857\ x\ 100\ \% = 85.7\ \%$$

(3) *F1 score* pada Pulau Sumatra

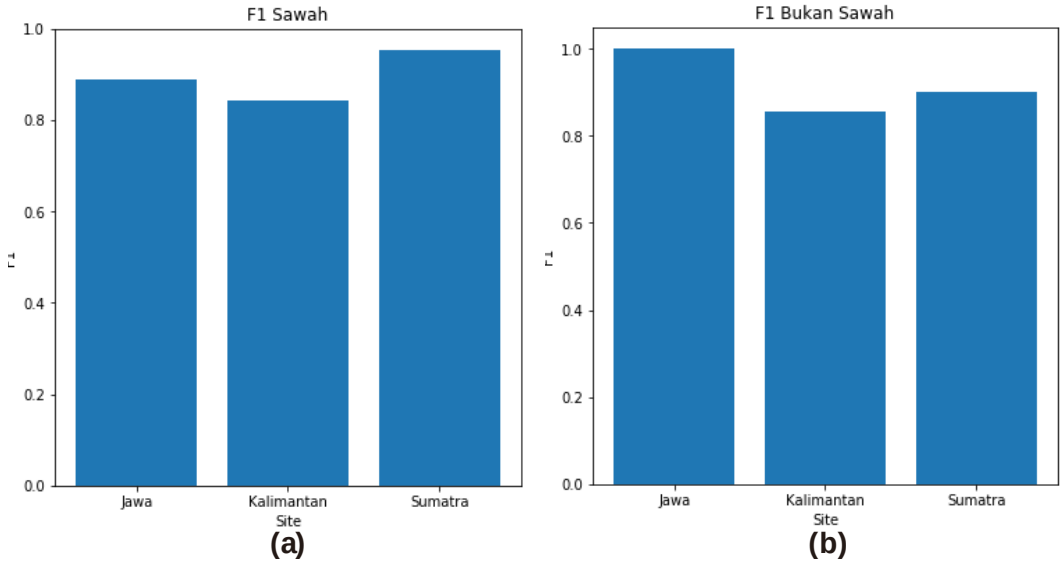
- $F1\ score\ Citra\ Sawah = 2\left(\frac{1\ x\ 0.909}{1 + 0.909}\right) = 0.952$

  $\%\ F1\ score\ \ Citra\ Sawah = 0.952\ x\ 100\ \% = 95.2\%$

  $F1\ score\ \ Citra\ Bukan\ Sawah = 2\left(\frac{0.9\ \ x\ 1}{0.9 + 1}\right) = 0.9$

  $\%F1\ score\ \ Citra\ Sawah = 0.9\ x\ 100\ \% = 90\ \%$

Adapun hasil perbandingan antar ketiga pulau dalam bentuk grafik *recall rate* pada Gambar 4.24 berikut ini :

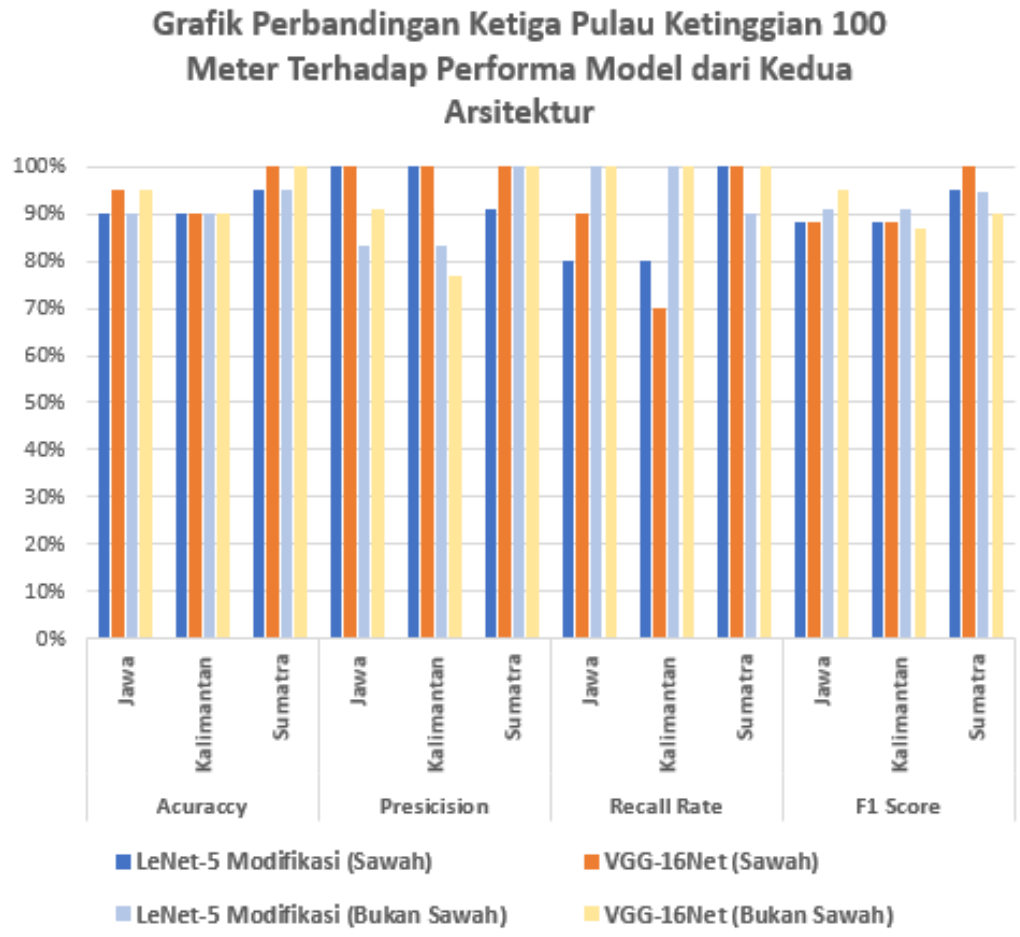


Gambar 4. 24 Grafik F1 Score Citra Sawah dan Bukan Sawah Antar Ketiga Pulau

Pada Gambar 4.24 menunjukan grafik F1 *score* ketiga pulau sangat baik antara citra sawah dan yang bukan sawah karena keduanya mendekati nilai 1 atau presentase 100%. Pada citra sawah memiliki presentase tertinggi

pada pulau Sumatra sebesar 90%, sedangkan pada F1 *score* citra bukan sawah mencapai 100% di pulau Jawa.

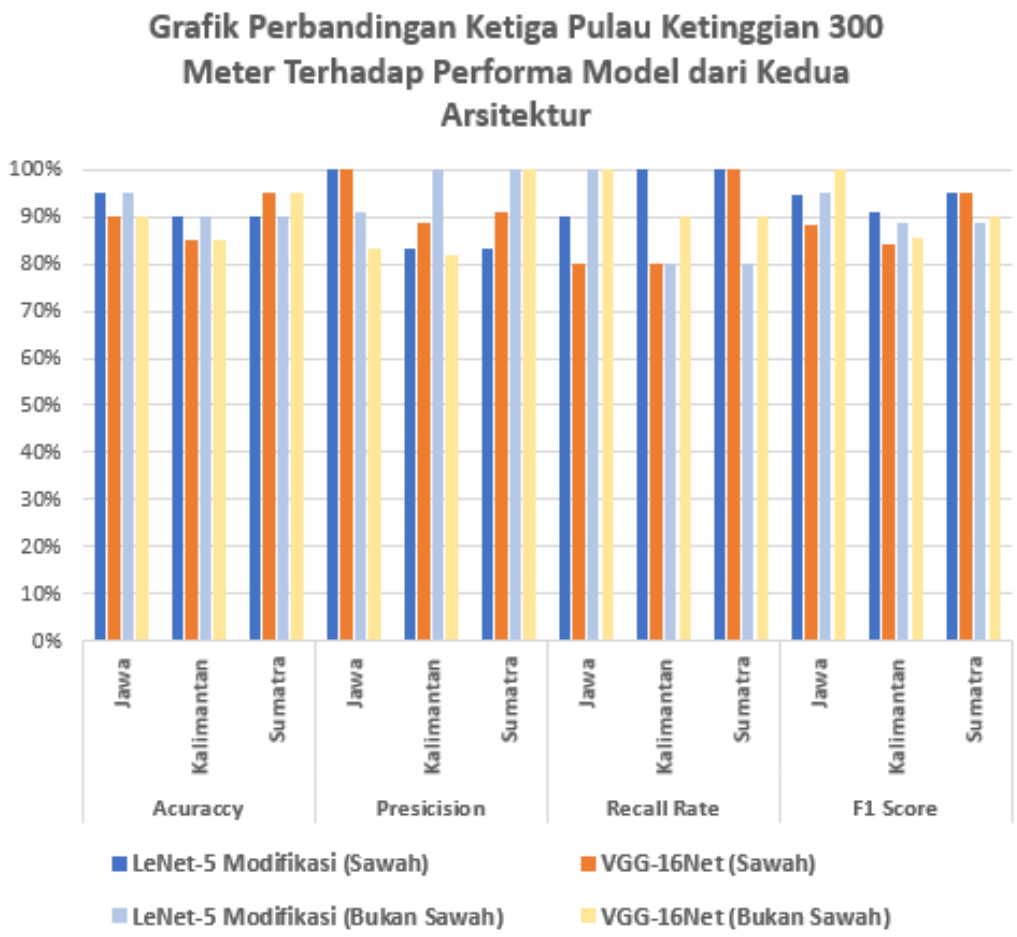
Sehingga berdasarkan dari keempat evaluasi performa model terhadap ketinggian dan arsitektur yang berbeda didapatkan empat hasil perhitungan performa model dalam klasifikasi dan prediksinya. Kemudian dapat dibandingkan antar evaluasi menjadi suatu grafik seperti Gambar 4.25 dan Gambar 4.26 berikut ini :



Gambar 4. 25 Grafik Perbandingan Ketiga Pulau Ketinggian 100 Meter Terhadap Performa Model dari Kedua Arsitektur

Pada Gambar 4.25 diatas menunjukkan grafik perbandingan antara arsitektur *LeNet-5* Modifikasi dan *VGG-16Net* terhadap ketiga pulau dalam ketinggian 100 meter. Dapat diketahui bahwa hasil pada pulau Sumatra dengan arsitektur *VGG-16Net* yang digunakan dalam perhitungan klasifikasi dan prediksi tersebut mampu mencapai persentase keseluruhan 100% pada dikedua *class* citra sawah maupun citra bukan sawah. Meskipun demikian arsitektur *VGG-16Net* menunjukkan hasil yang kurang optimal pada pulau Kalimantan citra bukan sawah dengan perhitungan presisi yang mencapai 76,9%. Begitu pun dengan penurunan signifikan pada *recall rate* dengan arsitektur *VGG-16Net* pada pulau kalimantan dengan citra sawah yang mencapai 70%. Sedangkan pada arsitektur *LeNet-5* Modifikasi dapat dikatakan lebih stabil meski keseluruhan perhitungan hanya mencapai 90%. Hal ini menandakan kinerja model pada citra ketinggian 100 meter dengan arsitektur *LeNet-5* Modifikasi jauh lebih baik dalam pengklasifikasian maupun prediksi dalam perhitungan performa model terhadap *dataset* pengujian yang dimiliki.

Kemudian didapatkan juga hasil performa model pada perbandingan grafik di ketinggian 300 meter seperti yang terlihat pada Gambar 4.26 sebagai berikut ini.



Gambar 4. 26 Grafik Perbandingan Ketiga Pulau Ketinggian 300 Meter Terhadap Performa Model dari Kedua Arsitektur

Pada Gambar 4.26 diatas menunjukkan grafik perbandingan antara arsitektur *LeNet-5* Modifikasi dan *VGG-16Net* terhadap ketiga pulau dalam ketinggian 300 meter. Dapat diketahui bahwa hasil perhitungan akursi pada pulau Jawa dan Sumatra dikedua arsitektur menunjukkan hasil yang sangat tinggi mencapai diatas dari 90% diantara dua *class* citra dibandingkan dengan pulau Kalimantan. Sedangkan pada presisi maupun *recall rate* didapatkan persentase tertinggi pada citra sawah pulau Jawa dan Sumatra yang mencapai 100% dikedua arsitektur. Sehigga diperoleh rasio keseimbangan dan perbandingan rata-arat presisi

dengan *recall rate* yang dikenal dengan F1 *score*, didapatkan angka tertinggi pada Pulau Jawa dengan citra bukan sawah yang menggunakan arsitektur *VGG-16Net* yang mencapai 100% dan arsitektur *LeNet-5* Modifikasi pada kedua *class* citra yang mencapai nilai persentase 95,2%. Hal ini menandakan kinerja model pada citra ketinggian 300 meter pun dengan arsitektur *LeNet-5* Modifikasi jauh lebih baik dalam pengklasifikasian maupun prediksi dalam perhitungan performa model terhadap *dataset* pengujian yang dimiliki.

4.3.2 Evaluasi Akurasi Dataset Pengujian dalam Piksel

Setelah peneliti melakukan pengujian dari *dataset* pengujian pada dua *class*, yaitu citra sawah dan bukan sawah dengan hasil prediksi pengujian dalam model CNN yang sudah diperoleh. Sehingga didapatkan hasil evaluasi dan perbandingan diantara dua dataset tersebut di setiap ketiga area pulau, berupa klasifikasi akurasi, presisi, *recal rate*, dan F1 *score.*

Kemudian peneliti melakukan pengujian lainnya dari hasil performa model CNN tersebut pada *dataset* pengujian yang masih menggunakan citra dari folder *test*. Akan tetapi pengujian dilakukan dengan metode perhitungan akurasi pada jumlah piksel. Kedua *class* yang terdiri dari citra sawah dan citra bukan sawah masing-masing dihitung akurasi yang dimiliki pada pikselnya. Terdapat nilai level yang menandakan suatu citra dinyatakan terhitung sebagai citra sawah maupun yang bukan sawah. Dalam penentuan ini citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari 0.5 disebut citra bukan sawah.

Berbeda dari evaluasi sebelumnya yang melakukannya pada 20 citra di masing-masing dari ketiga area pulau. Peneliti hanya melakukan perhitungan dan menampilkan visualisasi akurasi citra pada 1 citra dengan dua *class* saja pada area pulau Jawa. Dan melakukan perhitungan rata-rata akurasi dari 20 citra dalam bentuk tabel 4.10.

Berikut ini diperlihatkan salah satu contoh hasil akurasi dalam piksel di pulau Jawa diantara pengujian pada setiap ketinggian dan arsitektur yang digunakan pada tabel 4.9.

Tabel 4. 9 Hasil Akurasi dalam Piksel

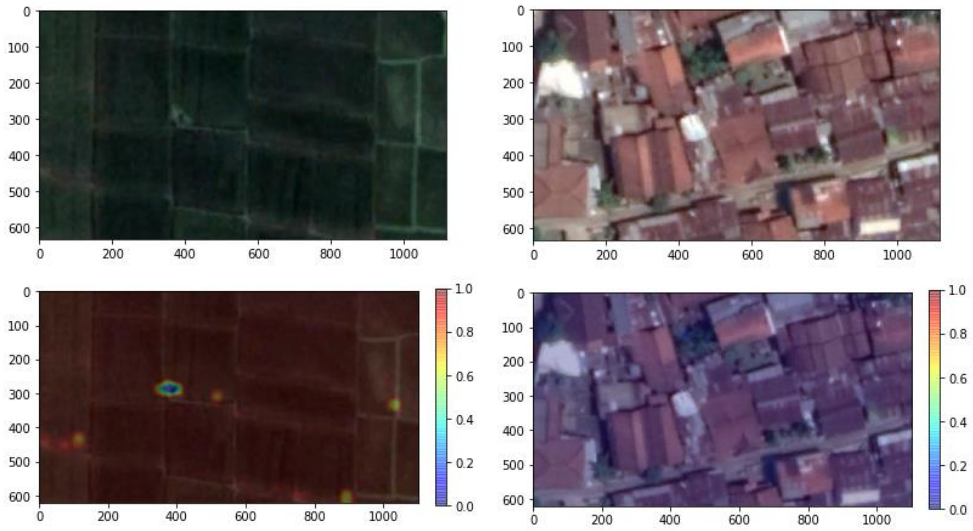
No.	**Keting-gian Objek**	**Arsitektur**	**Dataset Pengujian**				
			Nama Objek	***Class***	**Total Piksel**	**Jumlah Piksel Terdeteksi**	**Akurasi (%)**
1.	100 Meter	*LeNet-5* Modifikasi	b151	Sawah	685584 piksel	682266	99.516%
			o153	Bukan Sawah		684874	99.896%
		VGG-16Net	b151	Sawah		680403	99.244%
			o153	Bukan Sawah		683102	99.638%
2.	300 Meter	*LeNet-5* Modifikasi	b151	Sawah		680650	99.280%
			o153	Bukan Sawah		684875	99.896%
		VGG-16Net	b151	Sawah		667208	97.319%
			o153	Bukan Sawah		685175	99.940%

Pada tabel 4.9 diatas menunjukkan hasil akurasi dalam piksel terhadap masing-masing objek ketinggian dan arsitektur yang digunakan. Pada evaluasi penelitian ini diambil *dataset* pengujian hanya pada 2 citra di pulau Jawa dan

dibandingkan hasil akurasinya dalam kedua arsitekturnya. Akan terlihat kinerja performa model yang lebih baik dalam melakukan akurasi dari kedua arsitektur. Total piksel yang dimiliki pada masing-masing citra sawah dan bukan sawah adalah 685584 piksel. Sehingga setelah dilakukan metode perhitungan akurasi pada jumlah piksel diperoleh hasil akurasi tertinggi pada citra bukan sawah di ketinggian 300 meter menggunakan arsitektur *VGG-16Net* sebesar 99.940%. Sedangkan dari hasil akurasi keseluruhan pada citra sawah dan citra bukan sawah pada setiap ketinggian yang ditentukan, hasil akurasi yang lebih merata didapatkan dengan menggunakan arsitektur *LeNet-5* Modifikasi. Perlu diperhatikan bahwa perbandingan antar arsitektur dibandingkan pada ketinggian yang sama. Karena masing-masing arsitektur memiliki tingkat level perbesaran citra yang berbeda dan pada pengujian ini lokasi citra antar ketinggian juga berbeda satu sama lain, meskipun di lokasi yang sama di daerah banyumas pada pulau Jawa.

1. Evaluasi akurasi dalam piksel pada *dataset* pengujian di ketinggian 100 Meter dengan arsitektur *LeNet-5Net* Modifikasi

   Pada tabel 4.9 dapat dilihat hasil akurasi piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Kedua citra yang diujikan memiliki dimensi piksel sebesar 1116x632 piksel dan jumlah pikselnya adalah 685584 piksel. Dengan ketentuan level nilai diantara kedua *class* ini berupa citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari sama dengan 0.5 disebut citra bukan sawah. Sehingga apabila dievaluasikan model yang sudah didapatkan pada *dataset* pengujian dengan kedua *class* citra dapat dilihat hasilnya pada Gambar 4.27 dibawah ini.



Gambar 4. 27 Identifikasi level nilai citra sawah dan bukan sawah

Berdasarkan Gambar 4.27 diatas menunjukkan level citra yang sawah dan citra bukan sawah. Pada citra sawah didapatkan jumlah piksel sebanyak 682266 piksel, sedangkan citra yang bukan sawah diperoleh sebanyak 684874 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.7) dan (4.8).

$$\%Akurasi\ Piksel\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.7)$$

$$\%Akurasi\ Piksel\ Bukan\ Sawah = (Jumlah\ piksel \leq 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.8)$$

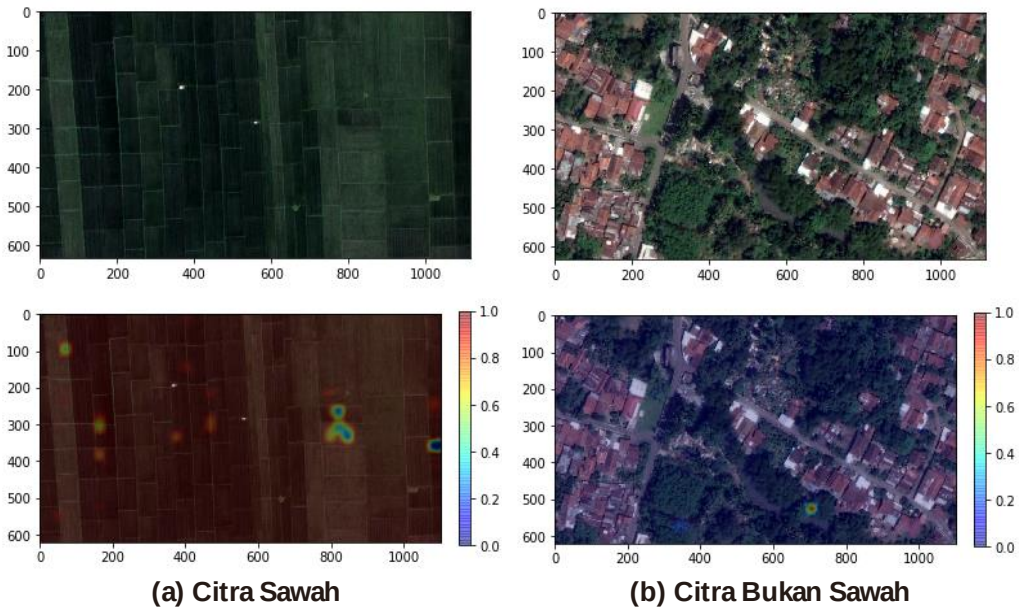
Apabila dilakukan perhitungan didapatkan hasil presentase akurasi sebagai berikut :

- $\%Akurasi\ Piksel\ Sawah = {}^{(682266}/_{685584}\ ) \ x\ 100\% = 99{,}516\%$
- $\%Akurasi\ Piksel\ Bukan\ Sawah = {}^{(684874}/_{685584}\ ) \ x\ 100\% = 99{,}896\%$

Kedua presentase akurasi piksel sawah diantaranya memiliki hasil yang mendekati 100%. Pada presentase akurasi citra sawah mencapai 99,516%, sedangkan presentase akurasi citra bukan sawah memeproleh 99,896%. Hal ini menunjukkan model pelatihan yang sebelumnya didapatkan dapat mengidentifikasi dengan baik citra sawah dan bukan sawah.

2. Evaluasi akurasi dalam piksel pada *dataset* pengujian di ketinggian 300 Meter dengan arsitektur *LeNet-5Net* Modifikasi

Pada tabel 4.9 dapat dilihat hasil akurasi piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Ketentuan metode perhitungan akurasi dalam piksel dilakukan dengan mengetahui level nilai diantara kedua *class* ini berupa citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari sama dengan 0.5 disebut citra bukan sawah. Sehingga apabila dievaluasikan model yang sudah didapatkan pada *dataset* pengujian dengan kedua *class* citra dapat dilihat hasilnya pada Gambar 4.28 dibawah ini.



Gambar 4. 28 Identifikasi level nilai citra sawah dan bukan sawah

Berdasarkan Gambar 4.28 diatas menunjukkan level citra yang sawah dan citra bukan sawah. Kedua *class* berupa citra sawah dan citra bukan sawah memiliki jumlah piksel sebanyak 680650 piksel dan 684875 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.7) dan (4.8).

$$\%Akurasi\ Piksel\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.7)$$

$$\%Akurasi\ Piksel\ Bukan\ Sawah = (Jumlah\ piksel \leq 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.8)$$

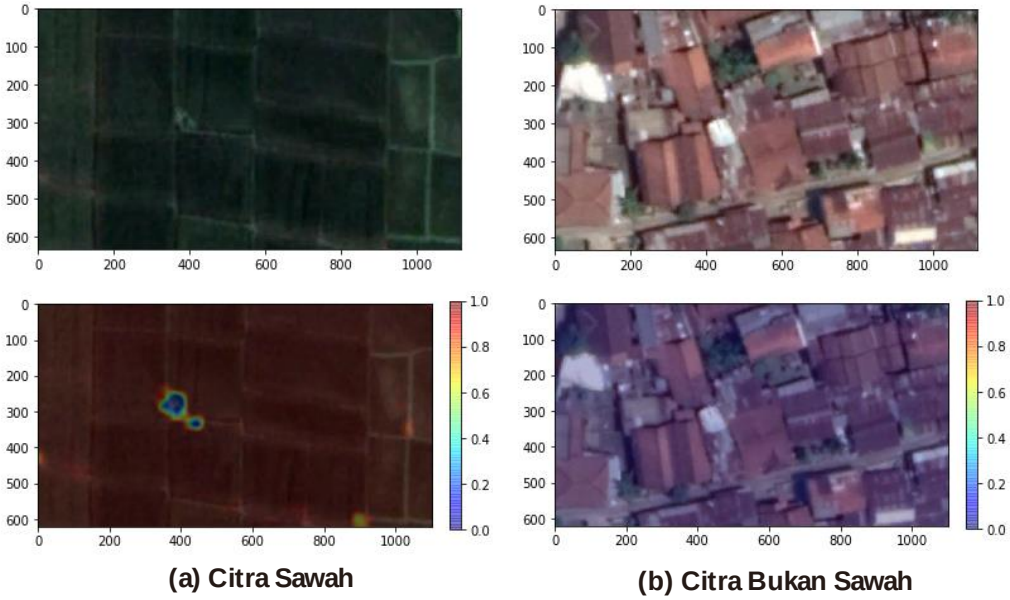
Apabila dilakukan perhitungan didapatkan hasil presentase akurasi sebagai berikut :

- $\%Akurasi\ Piksel\ Sawah = {}^{(680650}/_{685584}\ ) \ x\ 100\% = 99{,}280\%$
- $\%Akurasi\ Piksel\ Bukan\ Sawah = {}^{(684875}/_{685584}\ ) \ x\ 100\% = 99{,}896\%$

Kedua presentase menunjukkan bahwa pada model pelatihan yang sebelumnya didapatkan dapat mengidentifikasi dengan baik citra sawah dan bukan sawah. Karena pada citra sawah didapatkan hasil akurasi piksel dengan presentase 99,280% , sedangkan citra yang bukan sawah diperoleh presentase dengan akurasi sebesar 99,896%. Meskipun masih terdapat sedikit bintik pada kedua citra.

3. Evaluasi akurasi dalam piksel pada *dataset* pengujian di ketinggian 100 Meter dengan arsitektur *VGG-16Net*

Pada tabel 4.9 dapat dilihat hasil akurasi piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Kedua citra yang diujikan memiliki dimensi piksel sebesar 1116x632 piksel dan jumlah pikselnya adalah 685584 piksel. Dengan ketentuan level nilai diantara kedua *class* ini berupa citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari sama dengan 0.5 disebut citra bukan sawah. Sehingga apabila dievaluasikan model yang sudah didapatkan pada *dataset* pengujian dengan kedua *class* citra dapat dilihat hasilnya pada Gambar 4.29 dibawah ini.



Gambar 4. 29 Identifikasi level nilai citra sawah dan bukan sawah

Berdasarkan Gambar 4.29 diatas menunjukkan level citra yang sawah dan citra bukan sawah. Pada citra sawah didapatkan jumlah piksel sebanyak 680403 piksel, sedangkan citra yang bukan sawah diperoleh sebanyak 683102 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.7) dan (4.8).

$$\%Akurasi\ Piksel\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.7)$$

$$\%Akurasi\ Piksel\ Bukan\ Sawah = (Jumlah\ piksel \leq 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.8)$$

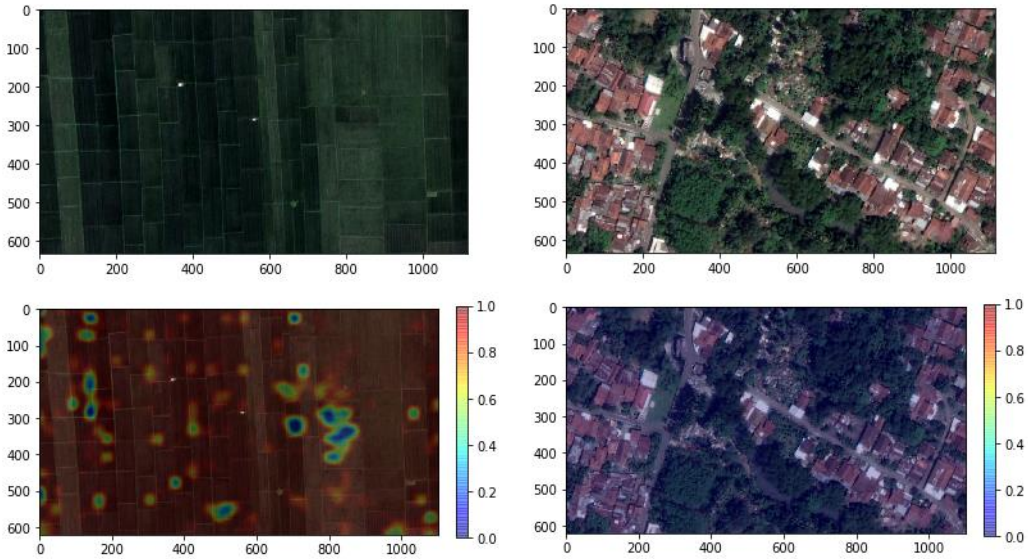
Apabila dilakukan perhitungan didapatkan hasil presentase akurasi sebagai berikut :

- $\%Akurasi\ Piksel\ Sawah = {}^{(680403}/_{685584}\ )\ x\ 100\% = 99{,}244\%$
- $\%Akurasi\ Piksel\ Bukan\ Sawah = {}^{(683102}/_{685584}\ )\ x\ 100\% = 99{,}638\%$

Kedua presentase akurasi piksel sawah diantaranya memiliki hasil yang mendekati 100%. Pada presentase akurasi citra sawah mencapai 99,244%, sedangkan presentase akurasi citra bukan sawah memeproleh 99,638%. Hal ini menunjukkan model pelatihan yang sebelumnya didapatkan dapat mengidentifikasi dengan baik citra sawah dan bukan sawah.

4. Evaluasi akurasi dalam piksel pada *dataset* pengujian di ketinggian 300 Meter dengan arsitektur *VGG-16Net*

Pada tabel 4.9 dapat dilihat hasil akurasi piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Ketentuan metode perhitungan akurasi dalam piksel dilakukan dengan mengetahui level nilai diantara kedua *class* ini berupa citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari sama dengan 0.5 disebut citra bukan sawah. Sehingga apabila dievaluasikan model yang sudah didapatkan pada *dataset* pengujian dengan kedua *class* citra dapat dilihat hasilnya pada Gambar 4.30 dibawah ini.



Gambar 4. 30 Identifikasi level nilai citra sawah dan bukan sawah

Berdasarkan Gambar 4.30 diatas menunjukkan level citra yang sawah dan citra bukan sawah. Kedua *class* berupa citra sawah dan citra bukan sawah memiliki jumlah piksel sebanyak 667208 piksel dan 685175 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.7) dan (4.8).

$$\%Akurasi\ Piksel\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.7)$$

$$\%Akurasi\ Piksel\ Bukan\ Sawah = (Jumlah\ piksel \leq 0.5)\ /\ (Total\ Piksel \geq 1) \quad (4.8)$$

Apabila dilakukan perhitungan didapatkan hasil presentase akurasi sebagai berikut :

- $\%Akurasi\ Piksel\ Sawah = {}^{(667208}/_{685584}\ ) \ x\ 100\% = 97{,}319\%$
- $\%Akurasi\ Piksel\ Bukan\ Sawah = {}^{(685175}/_{685584}\ ) \ x\ 100\% = 99{,}940\%$

Kedua presentase menunjukkan bahwa pada model pelatihan yang sebelumnya didapatkan dapat mengidentifikasi dengan baik citra sawah dan bukan

sawah. Karena pada citra sawah didapatkan hasil akurasi piksel dengan presentase 97,319% , sedangkan citra yang bukan sawah diperoleh presentase dengan akurasi sebesar 99,940%. Meskipun masih terdapat banyak bintik pada citra sawah yang mengurangi 100% ketidakakurasiannya.

Berikut ini perbandingan akurasi ketiga citra dari ketinggian dan arsitektur yang berbeda, maka didapatkan hasilnya pada tabel 4.10.

Tabel 4. 10 Hasil Perbandingan Akurasi Ketiga Citra Terhadap Arsitektur

Ketinggian	**Objek Area Penelitian**	**Arsitektur CNN**	**Jumlah Citra**	**Hasil Perbandingan Akurasi Ketiga Citra terhadap Arsitektur**	
				Rata-rata Akurasi dalam Piksel (%)	
				Sawah	**Bukan Sawah**
100 Meter	Pulau Jawa	*LeNet-5 Modif*	20	99,70%	99,37%
		VGG-16Net	20	99,66%	96,44%
	Pulau Kalimantan	*LeNet-5 Modif*	20	98,01%	97,42%
		VGG-16Net	20	97,25%	95,63%
	Pulau Sumatra	*LeNet-5 Modif*	20	99,35%	98,56%
		VGG-16Net	20	99,19%	98,93%
300 Meter	Pulau Jawa	*LeNet-5 Modif*	20	99,37%	99,70%
		VGG-16Net	20	98,13%	98,73%
	Pulau Kalimantan	*LeNet-5 Modif*	20	91,75%	98,06%
		VGG-16Net	20	92,51%	94,97%
	Pulau Sumatra	*LeNet-5 Modif*	20	97,03%	99,35%
		VGG-16Net	20	96,94%	99,18%
Hasil Perbandingan Ketiga Akurasi					
Ketinggian 100 Meter		*LeNet-5 Modif*	60	99,02%	97,85%
		VGG-16Net	60	98,7%	97%
Ketinggian 300 Meter		*LeNet-5 Modif*	60	96.05%	98,82%
		VGG-16Net	60	95,86%	97,62%

Pada tabel 4.10 diatas menunjukkan hasil perbandingan akurasi kedua arsitektur dalam ketinggian yang berbeda dalam piksel. Diketahui bahwa perbandingan yang didapatkan khususnya antar ketiga pulau dengan ketinggian yang berbeda, diperoleh nilai persentase yang jauh lebih unggul dengan menggunakan *LeNet-5* Modifikasi. Akurasi piksel tertinggi diperoleh pada pulau Jawa pada kedua *class* citra sawah dan bukan sawah yaitu didapatkan persentase 99,70% dan 99,37%. Sedangkan pada keseluruhan umum pada masing-masing ketinggiannya didapatkan perbandingan anatar kedua arsitektur yang mana pada *LeNet-5* Modifikasi dengan ketinggian 100 meter diperoleh 99,02% pada citra sawah dan 97,85% pada citra buka sawah. Sedangkan pada ketinggian 300 meter masing-masing citra pada arsitektur *LeNet-5* diperoleh hasil persentase sebesar 96,05% dan 97,85%. Hal ini menunjukkan penggunaan arsitektur *LeNet-5* Modifikasi jauh lebih akurat dan efisien dalam melakukan klasifikasi maupun identifikasi seperti kasus vegetasi geografis yang mengenali objek berukuran kecil.

4.4 Implementasi Terhadap Luasan Sawah

Pada penelitian terakhir ini, peneliti melakukan pengujian lainnya dari hasil performa model CNN tersebut pada *dataset* pengujian pada folder *test image*. Pengujian ini diimplementasikan untuk melakukan persentase hasil area yang terdapat lahan pertanian sawah, serta melakukan perhitunagn luas lahan dalam skala hektar. Pada folder *test image* terdapat kumpulan citra yang berisi objek campuran/perpaduan antara objek sawah dan bukan/atau selain sawah. Terdapat citra-citra yang berjumlah 10 citra pengujian yang mana terdiri dari masing-masing

tiga citra pada setiap ketiga pulau dan satu citra yang acak dari ketiga pulau yang ditentukan.

Namun peneliti hanya mengambil satu sampel citra dari 10 citra yang terdapat pada folder *test image*. Sampel yang diujikan dalam implementasi ini berlokasi di pulau Jawa dengan nama gambar "*test_image1*". Selain itu peneliti juga melakukan validasi dilapangan yang bertujuan untuk mengukur panjangnya dalam meter yang dibandingan panjang meter tersebut pada objek citra geografis yang sama..

4.4.1 Persentase Luas Lahan Sawah pada Sampel Pengujian

Pada pengujian ini serupa dengan pengujian pada akurasi piksel yang mana terdapat nilai level yang menandakan suatu citra dinyatakan terhitung sebagai citra sawah maupun yang bukan sawah. Dalam penentuan ini citra memiliki level benilai lebih dari 0.5 disebut citra sawah dan citra yang levelnya bernilai kurang dari 0.5 disebut citra bukan sawah. Berikut ini pada tabel 4.10 merupakan hasil persentase luasan sawah yang diimplemantasikan pada *dataset* pengujian dengan citra perpaduan objek antara sawah dan bukan/selain sawah pada pada sampel objek dengan nama "*test_image1*".

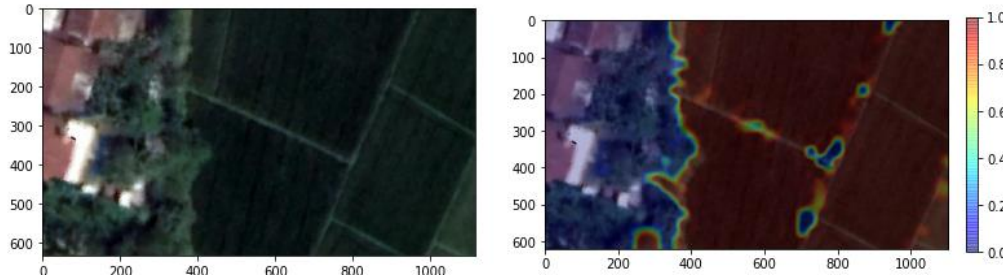
Tabel 4. 11 Hasil Persentase Luasan Sawah pada Sampel

No.	Ketinggian Objek	Arsitektur	Dataset Pengujian			
			Nama Objek	Total Piksel	Jumlah Piksel Sawah	Persentase (%)
1.	100 Meter	*LeNet-5* Modifikasi	test_image1	685584 piksel	454237	66.255%
		VGG-16Net	test_image1		442656	64.566%
2.	300 Meter	*LeNet-5* Modifikasi	test_image1		359651	52.459%
		VGG-16Net	test_image1		339664	49.543%

Berdasarkan tabel 4.10 diatas menunjukkan hasil persentase luasan sawah yang memiliki hasil berupa jumlah piksel yang terdeteksi dan presentase pada kedua arstitektur maupun ketinggiannya. Hasil persentase ini dilakukan perbandingan antara kinerja/performa arsitektur dapat dilihat dari hasil yang diperoleh anatar masing-masing ketinggian. Karena setiap ketinggian yang ditentukan memiliki gambar yang berbeda dan tingkat perbesaran citranya. Dapat diketahui bahwa citra pada ketinggian 100 meter menggunakan arsitektur *LeNet-5* Modifikasi mempunyai persentase yang tinggi dibandingkan arsitekur *VGG-16Net* yaitu sebesar 66,255%. Sedangkan pada ketinggian 300m, hasil persentase tertinggi pun dimiliki oleh arsitektur *LeNet-5* Modifikasi sebesar 52,429%.

1. Hasil persentase dan jumlah piksel pada *dataset* pengujian di ketinggian 100 Meter dengan arsitektur *LeNet-5Net* Modifikasi

Pada tabel 4.10 dapat dilihat hasil presentase dan jumlah piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Pada *dataset* pengujian memiliki dimensi piksel sebesar 1116x632 piksel dan jumlah pikselnya adalah 454237 piksel. Apabila diimplentasikan model, maka hasil identifikasinya dapat dilihat hasilnya pada Gambar 4.31 dibawah ini.



(a) Citra Perpaduan Sawah dan Bukan Sawah

(b) Citra yang Terdeteksi Sawah

Gambar 4. 31 Hasil Identifikasi Lahan Sawah pada Ketinggian 100 Meter dengan LeNet-5 Modifikasi

Berdasarkan Gambar 4.31 diatas menunjukkan level citra yang berwarna merah merupakan yang terdeteksi sebagai lahan sawah dan citra yang berwarna biru merupakan citra bukan sawah. Pada citra sawah yang terdeteksi didapatkan jumlah piksel sebanyak 454237 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.9), berikut ini.

$$\%Lahan\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \qquad (4.9)$$

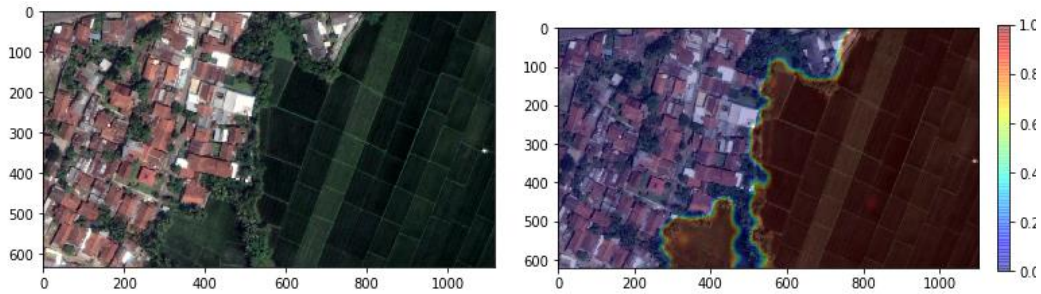
Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $\%Lahan\ Sawah = {}^{454237}/_{685584}\ x\ 100\% = 66.255\%$

Hasil presentasi lahan sawah menunjukkan nilai sebesar 66.255%. Apabila dilihat secara langsung dengan pengelihatan, memang bagian yang berwarna merah menunjukkan identifikasi lahan pertanian sawah. Meksipun masih terdapat sedikit bagian yang belum teridentifikasi keselurahnnya. Namun secara umum dapat dinyatakan model melakukan evaluasi dan klasifikasi dengan baik dan benar.

2. Hasil persentase dan jumlah piksel pada *dataset* pengujian di ketinggian 300 Meter dengan arsitektur *LeNet-5Net* Modifikasi

Pada tabel 4.10 dapat dilihat hasil presentase dan jumlah piksel pada *dataset* pengujian di ketinggian 300 meter dengan arsitektur *LeNet-5Net* Modifikasi. Apabila diimplentasikan model, maka hasil identifikasinya dapat dilihat hasilnya pada Gambar 4.32 dibawah ini.



Gambar 4. 32 Hasil Identifikasi Lahan Sawah pada Ketinggian 300 Meter dengan LeNet-5 Modifikasi

Berdasarkan Gambar 4.32 diatas menunjukkan level citra yang berwarna merah merupakan yang terdeteksi sebagai lahan sawah dan citra yang berwarna biru merupakan citra bukan sawah. Pada citra sawah yang terdeteksi didapatkan jumlah

piksel sebanyak 359651 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.9), berikut ini.

$$\%Lahan\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \qquad (4.9)$$

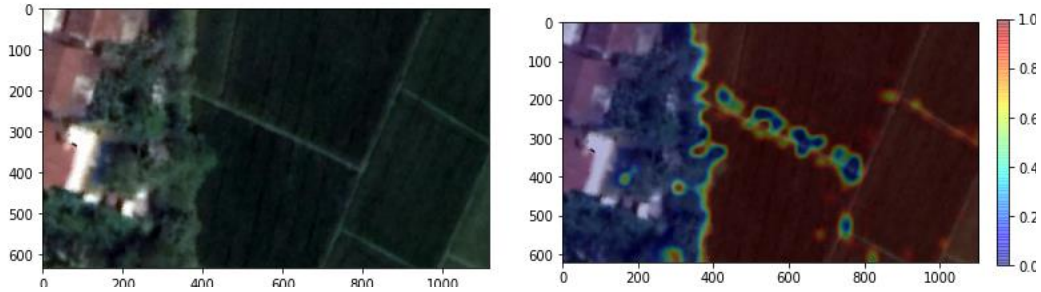
Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $\%Lahan\ Sawah = {}^{359651}/_{685584}\ x\ 100\% = 52.459\%$

Hasil presentasi lahan sawah menunjukkan nilai sebesar 52.459%. Apabila dilihat secara langsung dengan pengelihatan, memang bagian yang berwarna merah menunjukkan identifikasi lahan pertanian sawah. Oleh karena itu, dapat dinyatakan model melakukan evaluasi dan klasifikasi dengan baik dan benar.

3. Hasil persentase dan jumlah piksel pada *dataset* pengujian di ketinggian 100 Meter dengan arsitektur *VGG-16Net*

Pada tabel 4.10 dapat dilihat hasil presentase dan jumlah piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Pada *dataset* pengujian memiliki dimensi piksel sebesar 1116x632 piksel dan jumlah pikselnya adalah 442656 piksel. Apabila diimplentasikan model, maka hasil identifikasinya dapat dilihat hasilnya pada Gambar 4.33 dibawah ini.



Gambar 4. 33 Hasil Identifikasi Lahan Sawah pada Ketinggian 100 Meter dengan VGG-16Net

Berdasarkan Gambar 4.33 diatas menunjukkan level citra yang berwarna merah merupakan yang terdeteksi sebagai lahan sawah dan citra yang berwarna biru merupakan citra bukan sawah. Pada citra sawah yang terdeteksi didapatkan jumlah piksel sebanyak 442656 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.9), berikut ini.

$$\%Lahan\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \qquad (4.9)$$

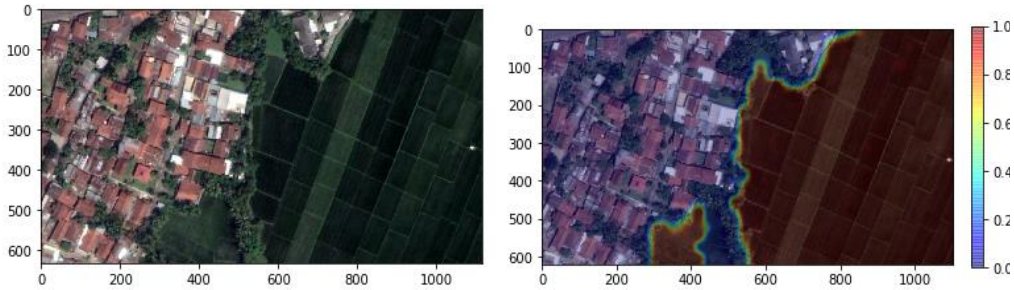
Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $\%Lahan\ Sawah = {}^{442656}/_{685584}\ x\ 100\% = 64.566\%$

Hasil presentasi lahan sawah menunjukkan nilai sebesar 64.566%. Apabila dilihat secara langsung dengan pengelihatan, memang bagian yang berwarna merah menunjukkan identifikasi lahan pertanian sawah. Oleh karena itu, dapat dinyatakan model melakukan evaluasi dan klasifikasi dengan baik dan benar.

4. Hasil persentase dan jumlah piksel pada *dataset* pengujian di ketinggian 300 Meter dengan arsitektur *VGG-16Net*

Pada tabel 4.10 dapat dilihat hasil presentase dan jumlah piksel pada *dataset* pengujian di ketinggian 100 meter dengan arsitektur *LeNet-5Net* Modifikasi. Apabila diimplentasikan model, maka hasil identifikasinya dapat dilihat hasilnya pada Gambar 4.34 dibawah ini.



Gambar 4. 34 Hasil Identifikasi Lahan Sawah pada Ketinggian 300 Meter dengan VGG-16Net

Berdasarkan Gambar 4.34 diatas menunjukkan level citra yang berwarna merah merupakan yang terdeteksi sebagai lahan sawah dan citra yang berwarna biru merupakan citra bukan sawah. Pada citra sawah yang terdeteksi didapatkan jumlah piksel sebanyak 339664 piksel. Sehingga untuk mendapatkan persentase akurasi pada kedua *class* citra dapat dilakukan dengan perhitungan dengan rumus (4.9), berikut ini.

$$\%Lahan\ Sawah = (Jumlah\ piksel > 0.5)\ /\ (Total\ Piksel \geq 1) \qquad (4.9)$$

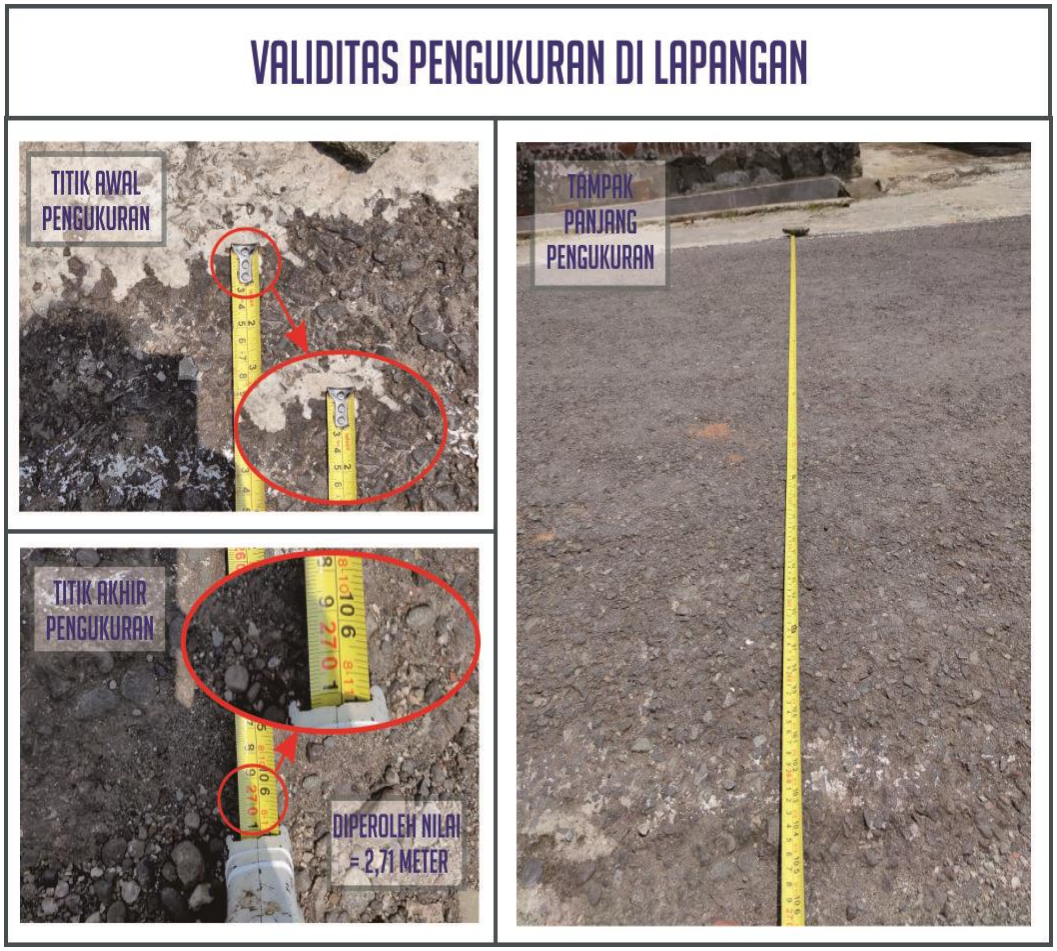
Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $\%Lahan\ Sawah = {}^{339664}/_{685584}\ x\ 100\% =\ 49.543\%$

Hasil presentasi lahan sawah menunjukkan nilai sebesar 49.543%. Apabila dilihat secara langsung dengan pengelihatan, memang bagian yang berwarna merah menunjukkan identifikasi lahan pertanian sawah. Oleh karena itu, dapat dinyatakan model melakukan evaluasi dan klasifikasi dengan baik dan benar.

4.4.2 Perhitungan Luas Lahan Sawah pada Sampel Pengujian

Pada perhitungan luas lahan sawah ini, peneliti juga melakukan validasi dilapangan yang bertujuan untuk mengukur panjangnya dalam meter yang dibandingan panjang meter tersebut pada objek citra geografis yang sama. Untuk validasi/keakuratan perhitungan konversi nilai piksel pada meter, peneliti mengambil objek yang digunakan sebagai pengukuran tersebut pada lebar jalan di depan rumah. Lokasi terletak di Dusun II RT 02/RW 07, Sokaraja Tengah, Kecamatan Sokaraja, Kabupaten Banyumas, Provinsi Jawa Tengah . Sehingga didapat hasil pengukurannya terlihat pada Gambar 4.35 dibawah ini :



Gambar 4. 35 Validasi pengukuran di lapangan

Pada Gambar 4.35 menunjukkan validasi pengukuran langsung dilapangan dengan menggunakan alat meteran. Sehingga didapat hasil pengukuran sebesar 2,71 meter. Untuk metode perhitungan konversi meter per piksel dapat dilakukan mengambil objek yang sama pada lokasi pengukuran validasinya melalui fitur pengukuran panjang pada *Google Earth Pro* dan hingga mendapatkan panjang yang serupa dengan data validasi sebesar 2,71 meter. Kemudian gambar tesebut disimpan dan memperbesar citra mencapai batas maksimum dapat dilakukan pada perangkat lunak desain, dalam hal ini peneliti menggunakan *CorelDraw* X7. Sehingga akan didaptatkan satu piksel dalam panjang meter objek yang kita validasikan

dilapangan. Perlu diperhatikan pada pengujian ini kita memiliki dua ketinggian yang berbeda, yaitu ketinggian 100 meter dan 300 meter. Oleh karena itu, akan dimiliki dua hasil perhitungan konversi meter per piksel dan hanya dapat digunakan pada ketinggian yang sama. Berikut ini ditampilkan hasil perhitungan luas lahan sawah pada sampel pengujian

Tabel 4. 12 Hasil Perhitungan Luas Lahan Sawah pada Sampel Pengujian

No.	**Keting-gian Objek (meter)**	**Arsitektur**	**Sampel Pengujian Citra (*test_image1*)**				
			Jumlah Piksel Sawah (piksel)	**Validasi Pengu-kuran (meter)**	**Piksel Pengu-Kuran (piksel)**	**Konversi Meter per Piksel (m/p)**	**Luas Lahan Sawah (Ha)**
1.	100 Meter	*LeNet-5* Modifikasi	454237	2,71	40	0,068	0.21
		VGG-16Net	442656	2,71	40	0,068	0.204
2.	300 Meter	*LeNet-5* Modifikasi	359651	2,71	10	0,271	2.64
		VGG-16Net	339664	2,71	10	0,271	2.49

Pada tabel 4.11 menunjukkan beberapa nilai yang digunakan dalam melakukan perhitungan konversi meter per piksel dan perhitungan luas lahan sawah. Terdapat rumus yang dilakukan dalam melakukan konversi meter per piksel seperti pada rumus (4.10), berikut ini :

$$Konversi\ M\ per\ P = (Validasi\ (m)) \ / \ (piksel\ (p)) \qquad (4.10)$$

Sehingga didapatkan pengukuran piksel pada ketinggian 100 meter dan 300 meter seperti ini :

- $Konversi\ 100\ meter\ (m/p) = 2{,}71 \ / \ 40 \ = \ 0{,}068\ (m/p)$

- $Konversi\ 300\ meter\ (m/p) = 2{,}71\ /\ 10\ =\ \ 0{,}271\ \ (m/p)$

Kemudian didapatkan hasil perhitungan konversi pada masing-masing ketinggian, yang mana pada ketinggian 100 meter diperoleh nilai 0,068 satuan meter per piksel. Sedangkan pada ketinggian 300 meter diperoleh nilai 0,271 satuan meter per piksel.

1. Hasil perhitungan luas lahan pada sampel pengujian ketinggian 100 Meter dengan arsitektur *LeNet-5Net* Modifikasi

Pada sampel pengujian yang terdeteksi citra lahan sawah didapatkan jumlah piksel sebanyak 454237 piksel. Sehingga untuk mendapatkan luas lahan pertanian sawah dapat dilakukan dengan perhitungan seperti pada rumus (4.11) berikut ini :

$$Luas\ Lahan = (konversi\ m\ per\ p)^2\ x\ (jumlah\ piksel\ yang\ terdeteksi) \qquad (4.11)$$

Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $Luas\ Lahan = (0{,}068)^2\ x\ (454237) =\ 2.100{,}391\ meter2$
- $Luas\ Lahan\ Hektar\ (Ha) =\ 2.100{,}391\ x\ 10.000 =\ 0{,}21\ hektar$

Hasil perhitungan luas lahan sawah pada sampel pengujian di ketinggian 100 meter dengan menggunakan arsitektur *LeNet-5* modifikasi diperoleh luas sebesar 0.21 hektar.

2. Hasil perhitungan luas lahan pada sampel pengujian ketinggian 300 Meter dengan arsitektur *LeNet-5Net* Modifikasi

Kemudian pada sampel pengujian ketinggian 300 meter yang terdeteksi citra lahan sawah didapatkan jumlah piksel sebanyak 359651 piksel. Sehingga untuk mendapatkan luas lahan pertanian sawah dapat dilakukan dengan perhitungan seperti pada rumus (4.10) berikut ini :

$$Luas\ Lahan = (konversi\ m\ per\ p)^2\ x\ (jumlah\ piksel\ yang\ terdeteksi) \quad (4.11)$$

Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $Luas\ Lahan = (0{,}271)^2\ x\ (359651\ ) =\ 26.413{,}129\ meter2$
- $Luas\ Lahan\ Hektar\ (Ha) =\ 26.413{,}129\ x\ 10.000 =\ 2{,}64\ hektar$

Hasil perhitungan luas lahan sawah pada sampel pengujian di ketinggian 300 meter dengan menggunakan arsitektur *LeNet-5* modifikasi diperoleh luas sebesar 26.413,129$m2$ atau sebesar 2.64 hektar.

3. Hasil perhitungan luas lahan pada sampel pengujian ketinggian 100 Meter dengan arsitektur *VGG-16Net*

Pada sampel pengujian yang terdeteksi citra lahan sawah didapatkan jumlah piksel sebanyak 454237 piksel. Sehingga untuk mendapatkan luas lahan pertanian sawah dapat dilakukan dengan perhitungan seperti pada rumus (4.11) berikut ini :

$$Luas\ Lahan = (konversi\ m\ per\ p)^2\ x\ (jumlah\ piksel\ yang\ terdeteksi) \quad (4.11)$$

Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut :

- $Luas\ Lahan = (0{,}068)^2\ x\ (454237) =\ 2.046.84\ meter2$
- $Luas\ Lahan\ Hektar\ (Ha) =\ 2.046.84\ x\ 10.000 =\ 0{,}204\ hektar$

Hasil perhitungan luas lahan sawah pada sampel pengujian di ketinggian 100 meter dengan menggunakan arsitektur *VGG-16Net* modifikasi diperoleh luas sebesar 0.204 hektar.

4. Hasil perhitungan luas lahan pada sampel pengujian ketinggian 300 Meter dengan arsitektur *VGG-16Net*

Pada sampel pengujian terakhir yaitu pada ketinggian 300 meter yang terdeteksi citra lahan sawah didapatkan jumlah piksel sebanyak 339664 piksel. Sehingga untuk mendapatkan luas lahan pertanian sawah dapat dilakukan dengan perhitungan seperti pada rumus (4.10) berikut ini :

$$Luas\ Lahan = (konversi\ m\ per\ p)^2\ x\ (jumlah\ piksel\ yang\ terdeteksi) \quad (4.11)$$

Apabila dilakukan perhitungan didapatkan hasil presentasenya sebagai berikut : 24945.26

- $Luas\ Lahan = (0{,}271)^2\ x\ (339664\ ) = 24.945{,}26\ meter2$
- $Luas\ Lahan\ Hektar\ (Ha) =\ 24.945{,}26\ \ x\ 10.000 =\ 2{,}49\ hektar$

Hasil perhitungan luas lahan sawah pada sampel pengujian di ketinggian 300 meter dengan menggunakan arsitektur *VGG-16Net* modifikasi diperoleh luas sebesar 26.413,129$m2$ atau sebesar 2.64 hektar.

Agar hasil perhitungan luas lahan pada penelitian ini bisa dikatakan akurat, maka peneliti juga membandingkan hasilnya pada fitur pengukuran luas suatu daerah pada *Google Earth Pro*, disertakan pada lampiran. Kemudian mengukur luas lahannya yang sama objek pengambilannya dengan *dataset* pengujian citra yang telah didapatkan hasilnya juga.

BAB 5
PENUTUP

5.1 Kesimpulan

Kesimpulan dalam penelitian ini adalah sebagai berikut.

1. Objek yang berukuran sangat kecil dengan metode CPM dapat menghasilkan potongan citra menjadi 89.100 citra yang berukuran 56x56 piksel sehingga mengoptimalkan dalam identifikasi objek geografis lahan pertanian sawah dengan akurasi yang cukup tinggi.

2. Arsitektur *VGG-16Net* memerlukan waktu lebih lama sekitar 40 detik dibandingkan dengan arsitektur *LeNet-5* Modifikasi hanya 10 detik dalam melakukan pelatihan sepanjang pembelajaran *epoch.*

3. Perbandingan kurva akurasi dan kegagalan menunjukkan kedua arsitektur dalam kondisi *goodfit*, namun asitektur *LeNet-5* Modifikasi memiliki hasil yang lebih baik dalam pelatihan dengan akurasi akhir sebesar 0.9936 dan kegagalan 0.0193 dibandingkan dengan arsitektur *VGG-16Net* yang akurasi diperoleh 0.9923 dan kegagalan 0.0229 dalam skala nilai 0 sampai 1.

4. Hasil evaluasi performa model dari keempat perhitungan klasifikasi yang diperoleh dapat disimpulkan bahwa pada pulau Sumatra ketinggian 100 meter dengan arsitektur *VGG-16Net* yang digunakan mampu mencapai persentase keseluruhan 100% pada dikedua *class* citra sawah maupun citra bukan sawah, sedangkan pada ketinggian 300 meter tergolong merata penggunaan kedua arsitektur terhadap ketiga pulau yang diujikan. Hal ini

dikarenakan pengaruh lebih pada faktor kualias citra, objek sawah dan ketinggian pada ketiga pulau yang diambil.

5. Pada keseluruhan umum pada masing-masing ketinggiannya didapatkan perbandingan akurasi dalam piksel antar kedua arsitektur yang mana pada *LeNet-5* Modifikasi dengan ketinggian 100 meter diperoleh 99,02% pada citra sawah dan 97,85% pada citra bukan sawah. Sedangkan pada ketinggian 300 meter masing-masing citra pada arsitektur *LeNet-5* diperoleh hasil persentase sebesar 96,05% dan 97,85%.

6. Tingkat persentase dalam perbandingan luas lahan dengan satu citra dari ketinggian dengan arsitektur yang berbeda, diperoleh persentase tertinggi pada *LeNet-5* Modifikasi mencapai 66.255% pada ketinggian 100 meter, sedangkan pada ketinggian 300 meter mencapai 52.459%.

5.2 Saran

Berdasarkan penelitian yang telah dikerjakan penulis, penulis memberikan dapat memberikan beberapa saran guna pengembangan sistem selanjutnya.

1. Penggunaan *deep learning* dan metode CPM dapat juga diimplentasikan dalam pemetaan vegatasi komoditas pertanian lainnya, karena sangat baik dan mampu mendeteksi gambar dengan objek kecil, serta menawarkan penggunaannya yang murah dan efisien.
2. Untuk penelitian selanjutnya, dapat dikembangkan untuk identifikasi jenis padi pada lahan sawah dan pengenalan terhadap lahan sawah kering, serta diambil *dataset* dari pulau lainnya.

3. Dapat melakukan penelitian pada pelatihan dengan pembanding tidak hanya *dataset* pelatihan yang random, tetapi bagaimana setiap lahan pertanian sawah dapat diujikan satu sama lain.

DAFTAR PUSTAKA

[1] BPS Indonesia, “Statistika Indonesia 2019,” Badan Pusat Statistika. 2019.

[2] W. Shuntaro, Kazuaki Sumi, dan Takeshi Ise, “Automatic vegetation identification in Google Earth images using a convolutional neural network: A case study for Japanese bamboo forests,” hlm. 3.

[3] Badan Pusat Statistika, “PENGANTAR KERANGKA SAMPLE AREA,” 2017. [Daring]. Tersedia pada: https://ksa-nasional.info/a_pengantar.php [Diakses:18-Nov-2019].

[4] Goodfellow, I. Bengio, dan Courville A, “Deep Learning, ” MIT press. 2016.

[5] Yosinski. J, Clune, I. Bengio, dan Lipson H, “How transferable are feature features in deep neural networks?,” Advances in Neural Information Processing Systems (Montreal), 2016, hlm. 3320–3328

[6] LeCun, Y., Bottou, L., Bengio, Y., dan Haffner, P. “Gradient-based learning applied to document recognition. Proceedings of the IEEE”. 1998. 86(11):2278- 232.

[7] LeCun, Y., Bengio, Y., dan Hinton. G, “Deep Learning“. Nature, 2016, hlm 436–444

[8] MathWorks. “Graycoprops”. 2017. [Daring]. Tersedia pada https://www.mathworks.com/help/images/ref/graycoprops.html?search Hig hlight= [Diakses; 19-Nov-2019]

[9] Medium, “Neural Network,” 2004. [Daring]. Tersedia pada : https://medium.com/@samuelsena/pengenalan-deep-learning-8fbb7d8028ac [Diakses; 18-Nov-2019]

[10] Danukusumo, Kevin Pudi, “Implementasi Deep Learning Menggunakan Convolutional Neural Network untuk Klasifikasi Citra Candi Berbasis GPU”. Program Studi Teknik Informatika Fakultas Teknologi Industri : Universitas Atma Jaya Yogyakarta. 2017.

[11] Rismiyati, ”Implementasi Convolution Neural Network Untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital,” Universitas Gajah Mada, Yogyakarta. 2016.

[12] School of Computer Science Binus, “Dasar Pemahaman Neural Network,” [Daring]. Tersedia pada : https://socs.binus.ac.id/2012/07/26/konsep-neural-network/ [Diakses:18-Jan-2019]

[13] Medium, “Convolution Neural Network (CNN), ” [Daring]. Tersedia pada : https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutionalneural-network-cnn-b003b477dc94 [Diakses:11-Nov-2019]

[14] John Paul Mueller dan Luca Massaron. “*What is Google Colaboratory?*”. [Daring]. Tersedia pada: https://www.dummies.com/programming/python/ what-is-google-colaboratory/. [Diakses: 17-Nov-2019].

[15] Chollet, François. 2018. *Deep Learning with Python*. Shelter Island: Manning Publications Co.
[16] Geospatial Training Services. “Fundamentals Of Google Earth Pro”. 2016. [Daring]. Tersedia pada: http://geospatialtraining.com/fundamentals-of-google-earth-pro/ [Diakses: 17-Nov-2019].
[17] Mahi, A K. 2001. “Tata Ruang dan Pengelolaan Lingkungan Hidup”. PSL : Universitas Lampung. 2001.
[18] Abdurrohman, Harits. “Sliding Window”. 2016. [Daring]. Tersedia pada: https://medium.com/otakbeku/imgcv-2-sliding-window-2dda64723df0 [Diakses: 28-Nov-2019]
[19] T. Ise, M. Minagawa, & M. Onishi . “Classifying 3 moss species by deep learning using the chopped picture method”. Open Journal of Ecology. 2018. 166–173.
[20] Neurohive. “VGG16 – Convolutional Network for Classification and Detection”. [Daring]. Tersedia pada: https://neurohive.io/en/popular-networks/vgg16/. [Diakses: 18-Jan-2019].

LAMPIRAN

Lampiran 1. *Sourcecode* Identifikasi Lahan Pertanian Sawah dengan Menggunakan Arsitektur *LeNet-5* Modifikasi

```
#Mengimpor library
!pip install slidingwindow
import slidingwindow as sw
import numpy as np
import os
import keras.backend as K
import keras
import pandas as pd
from keras.models import Sequential
from keras import layers
from keras.preprocessing.image import load_img, img_to_array
import matplotlib.pyplot as plt
from skimage.io import imread, imsave
from skimage.transform import resize
from sklearn.utils.class_weight import compute_class_weight

from sklearn import svm, datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.utils.multiclass import unique_labels

#Mengambil dataset pada github yang ketinggian 100m atau 300m
!apt-get install subversion > /dev/null
!svn export https://github.com/Gio1709/paddy_100m/trunk/data/
> /dev/null

!apt-get install subversion > /dev/null
!svn export https://github.com/Gio1709/paddy_300m/trunk/data/
> /dev/null

#Perintah untuk melihat list bagian dari file yang sudah diund
uh dari Github
!ls data/paddy/train
#gambar dengan nama awalan dengan "b" adalah gambar yang terda
pat padi"
#gambar dengan nama awalan dengan "o" adalah gambar yang tidak
terdapat padi"

#Gambar yang terdapat lahan sawah padi
plt.title("Gambar yang terdapat lahan sawah padi") plt.imshow
(imread("data/paddy/train/b58.jpg")) plt.show()
#Gambar yang terdapat tidak lahan sawah padi plt.title("Gambar
yang tidak terdapat lahan sawah padi") plt.imshow(imread("data
/paddy/train/o125.jpg")) plt.show()
```

```
#Membuat patches dengan metode sliding windows
def save_to_folder(img_path,directory,size=56,overlap=0.4):
    #Membaca gambar path penyimpanan window ke direktori/file
yang spesifik
    img = imread(img_path)
    basename = os.path.basename(img_path)
    fname = os.path.splitext(basename)[0]
    windows = sw.generate(img, sw.DimOrder.HeightWidthChannel,
size, overlap)
    for i,window in enumerate(windows):
        ii = str("{0:05d}".format(i))
        _fname = fname + "_" + ii + ".jpg"
        _img = img[window.indices()]
        if _img.shape !=(size,size,3):
            continue
        img_target_path = os.path.join(directory,_fname)

        imsave(img_target_path,_img, check_contrast=False)

files = os.listdir("data/paddy/train/")
files = [x for x in files if x.endswith(".jpg")]

#Menyimpan gambar yang telah dicacah pada folder chopped
for file in files:
    save_directory = "data/paddy/chopped/"
    file = os.path.join("data/paddy/train/",file)
    save_to_folder(file,save_directory)

#Mengembalikan daftar gambar yang telah diacak pada direktori
#Data yang dilakukan pada model evaluasi

#Dataset Pulau Jawa
files = os.listdir("data/paddy/chopped/")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset Pelatihan
x = []
y = []

for file in files:
    if file.startswith("b"):
        label = 1
    else:
        label = 0

    img_path = os.path.join("data/paddy/chopped/",file)
    img = load_img(img_path)
    img = img_to_array(img)/255
    x.append(img)
    y.append(label)

X = np.array(x)
Y = np.array(y)
```

```
print("Bentuk array dari dataset train (pelatihan) adalah :",
(X.shape,Y.shape))

files = os.listdir("data/paddy/test/Jawa")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset pengujian

X_test1 = []
Y_test1 = []

for file in files:
    if file.startswith("b"):
        label1 = 1
    else:
        label1 = 0

    img_path1 = os.path.join("data/paddy/test/Jawa",file)
    img1 = load_img(img_path1)
    img1 = img_to_array(img1)/255.
    img1 = resize(img1,(56,56))
    X_test1.append(img1)
    Y_test1.append(label1)

x_test1 = np.array(X_test1)
y_test1 = np.array(Y_test1)
print("Bentuk array dari dataset test pulau Jawa (pengujian) a
dalah :", (x_test1.shape,y_test1.shape))

files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

files = os.listdir("data/paddy/test/Kalimantan")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset pengujian

X_test2 = []
Y_test2 = []

for file in files:
    if file.startswith("b"):
        label2 = 1
    else:
        label2 = 0

    img_path2 = os.path.join("data/paddy/test/Kalimantan",fil
e)
    img2 = load_img(img_path2)
    img2 = img_to_array(img2)/255.
    img2 = resize(img2,(56,56))
    X_test2.append(img2)
    Y_test2.append(label2)
```

```
x_test2 = np.array(X_test2)
y_test2 = np.array(Y_test2)
print("Bentuk array dari dataset test pulau Kalimantan (penguj
ian) adalah :", (x_test2.shape,y_test2.shape))

files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

files = os.listdir("data/paddy/test/Sumatra")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset pengujian

X_test3 = []
Y_test3 = []

for file in files:
    if file.startswith("b"):
        label3 = 1
    else:
        label3 = 0

    img_path3 = os.path.join("data/paddy/test/Sumatra",file)
    img3 = load_img(img_path3)
    img3 = img_to_array(img3)/255.
    img3 = resize(img3,(56,56))
    X_test3.append(img3)
    Y_test3.append(label3)

x_test3 = np.array(X_test3)
y_test3 = np.array(Y_test3)
print("Bentuk array dari dataset test pulau Sumatra (pengujia
n) adalah :", (x_test3.shape,y_test3.shape))

files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

#Membagi dataset menjadi dua yaitu pelatihan dan validasi
from sklearn.model_selection import train_test_split

train_x, valid_x, train_y, valid_y = train_test_split(X,Y,test
_size=0.25,stratify = y)

print("Bentuk array dari dataset train (pelatihan) adalah:",tr
ain_x.shape,train_y.shape)
print("Bentuk array dari dataset validation adalah:",valid_x.s
hape,valid_y.shape)

#Mengkomputasi kelas bobotnya
print(np.unique(train_y,return_counts=True))
print(np.unique(valid_y,return_counts=True))
```

```
cw = compute_class_weight("balanced",np.unique(train_y),train_
y)
print(cw)

#Pembuatan Model CNN
model = Sequential([
    layers.Conv2D(16,(3,3),name="conv1",input_shape=(56,56,3),
activation="relu",padding="same"),
    layers.MaxPool2D((2,2),name="pool1"),
    #layers.Dropout(0.05),
    layers.Conv2D(32,(3,3),name="conv2",padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D((2,2),name="pool2"),
    #layers.Dropout(0.05),
    layers.Conv2D(32,(3,3),name="conv3",padding="same"),
    layers.Activation("relu"),
    layers.MaxPool2D((2,2),name="pool3"),
    #layers.Dropout(0.05),
    layers.Flatten(),
    layers.Dropout(0.2),
    layers.Dense(32,activation="relu"),
    layers.Dense(1),
    layers.Activation("sigmoid",name="prediction")
]
)

model.compile(optimizer="adam",loss="binary_crossentropy",metr
ics=["acc"])

model.summary()

from keras.callbacks import EarlyStopping, ModelCheckpoint

#Menyimpan model bobot yang terbaik selama training
ckpt = ModelCheckpoint("paddy.h5", monitor='val_loss', verbose
=1, save_best_only=True, save_weights_only=False, mode='auto',
period=1)

#Menampilkan kurva akurasi dan kegagalan pada model
history = model.fit(train_x,train_y,batch_size=200,epochs=50,v
alidation_data=(valid_x,valid_y),class_weight=cw, callbacks =
[ckpt])

plt.plot(history.history["acc"],label="Akurasi Pelatihan")
plt.plot(history.history["val_acc"],label="Validasi Akurasi")
plt.legend()
plt.show()

plt.plot(history.history["loss"],label="Kegagalan Pelatihan")
plt.plot(history.history["val_loss"],label="Validasi Kegagalan
")
plt.legend()
plt.show()
```

```
#Memeriksa matriks model
print(model.metrics_names)
#Evaluasi data training
print(model.evaluate(train_x,train_y))
#Evaluasi validasi data
print(model.evaluate(valid_x,valid_y))
#Evaluasi test data
print('[Loss, Acc] Test Pulau Jawa', model.evaluate(x= x_test
1, y = y_test1))
print('[Loss, Acc] Test Pulau Kalimantan', model.evaluate(x= x
_test2, y = y_test2))
print('[Loss, Acc] Test Pulau Sumatra', model.evaluate(x= x_te
st3, y = y_test3))

y_pred_class1 = model.predict_classes(x_test1)
y_pred_class2 = model.predict_classes(x_test2)
y_pred_class3 = model.predict_classes(x_test3)

#Jumlah data prediksi dari data test
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Jawa :', y_pred_class1.transpose())
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Kalimantan :', y_pred_class2.transpose())
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Sumatra :', y_pred_class3.transpose())

from sklearn import metrics
print("Akurasi pada data Pulau Jawa:",metrics.accuracy_score(y
_test1, y_pred_class1))
print("Akurasi pada data Pulau Kalimantan:",metrics.accuracy_s
core(y_test2, y_pred_class2))
print("Akurasi pada data Pulau Sumatra:",metrics.accuracy_scor
e(y_test3, y_pred_class3))

# Mengetahui class berdasarkan gambar sawah dan bukan sawah
# Apabila 1 menandakan gambar sawah dan 0 bukan sawah
Y_test1 = pd.Index(y_test1)
print("Class dan jumlah data Pulau Jawa:\n Class Jumlah \n",Y_
test1.value_counts())
print("\n")
Y_test2 = pd.Index(y_test2)
print("Class dan jumlah data Pulau Kalimantan:\n Class Jumlah
\n",Y_test2.value_counts())
print("\n")
Y_test3 = pd.Index(y_test3)
print("Class dan jumlah data Pulau Sumatra:\n Class Jumlah \n
",Y_test3.value_counts())
print("\n")

# Menampilkan 20 data pertama/awal true atau pun prediksi
print("Dataset Pulau Jawa:")
print('True:', Y_test1.values[0:20])
print('False:', y_pred_class1.transpose()[0:20])
print("\n")
```

```
print("Dataset Pulau Kalimantan:")
print('True:', Y_test2.values[0:20])
print('False:', y_pred_class2.transpose()[0:20])
print("\n")
print("Dataset Pulau Sumatra:")
print('True:', Y_test3.values[0:20])
print('False:', y_pred_class3.transpose()[0:20])


# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion = metrics.confusion_matrix(Y_test1, y_pred_class1)
 print("Matriks hasil clasification Pulau Jawa:\n",confusion)
#[row, column]
TP = confusion[1, 1]
TN = confusion[0, 0]
FP = confusion[0, 1]
FN = confusion[1, 0]
# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion1 = metrics.confusion_matrix(Y_test2, y_pred_class2)
print("Matriks hasil clasification Pulau Kalimantan:\n",confus
ion1)
#[row, column]
TP1 = confusion1[1, 1]
TN1 = confusion1[0, 0]
FP1 = confusion1[0, 1]
FN1 = confusion1[1, 0]
# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion2 = metrics.confusion_matrix(Y_test3, y_pred_class3)
print("Matriks hasil clasification Pulau Sumatra:\n",confusion
2)
#[row, column]
TP2 = confusion2[1, 1]
TN2 = confusion2[0, 0]
FP2 = confusion2[0, 1]
FN2 = confusion2[1, 0]

# Classification accuracy dari ketiga pulau
#print((TP + TN) / float(TP + TN + FP + FN))
print("Classification accuracy Pulau Jawa:",metrics.accuracy_s
core(Y_test1, y_pred_class1))

#print((TP1 + TN1) / float(TP1 + TN1 + FP1 + FN1))
print("Classification accuracy Pulau Kalimantan:",metrics.accu
racy_score(Y_test2, y_pred_class2))

#print((TP2 + TN2) / float(TP2 + TN2 + FP2 + FN2))
print("Classification accuracy Pulau Sumatra:",metrics.accurac
y_score(Y_test3, y_pred_class3))
```

```
import matplotlib.pyplot as plt

data = {'Jawa': metrics.accuracy_score(Y_test1, y_pred_class
1), 'Kalimantan': metrics.accuracy_score(Y_test2, y_pred_class
2), 'Sumatra': metrics.accuracy_score(Y_test3, y_pred_class3)}
names = list(data.keys())
values = list(data.values())

fig, axs = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
axs.bar(names, values)

plt.title('Classification Accuracy')
plt.xlabel('Site')
plt.ylabel('Accuracy')

recall_rate_sawah = TP / float(FN + TP)

print("Recall Rate Sawah Pulau Jawa:",recall_rate_sawah)
#print(metrics.recall_score(test_y, y_pred_class))

recall_rate_sawah1 = TP1 / float(FN1 + TP1)

print("Recall Rate Sawah Pulau Kalimantan:",recall_rate_sawah
1)
#print(metrics.recall_score(test_y1, y_pred_class1))

recall_rate_sawah2 = TP2 / float(FN2 + TP2)

print("Recall Rate Sawah Pulau Sumatra:",recall_rate_sawah2)
#print(metrics.recall_score(test_y2, y_pred_class2))

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': recall_rate_sawah, 'Kalimantan': recall_rate_s
awah1, 'Sumatra': recall_rate_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Recall Rate Sawah')
plt.xlabel('Site')
plt.ylabel('Recall Rate')

recall_rate_bukan_sawah = TN / (TN + FP)
print("Recall Rate Bukan Sawah Pulau Jawa:",recall_rate_bukan_
sawah)

recall_rate_bukan_sawah1 = TN1 / (TN1 + FP1)
print("Recall Rate Bukan Sawah Pulau Kalimantan:",recall_rate_
bukan_sawah1)
```

```
recall_rate_bukan_sawah2 = TN2 / (TN2 + FP2)
print("Recall Rate Bukan Sawah Pulau Sumatra:",recall_rate_buk
an_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': recall_rate_bukan_sawah, 'Kalimantan': recall_
rate_bukan_sawah1, 'Sumatra': recall_rate_bukan_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Recall Rate Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('Recall Rate')

precision_sawah = TP / float(TP + FP)
print("Precision Sawah Pulau Jawa:",precision_sawah)

precision_sawah1 = TP1 / float(TP1 + FP1)
print("Precision Sawah Pulau Kalimantan:",precision_sawah1)

precision_sawah2 = TP2 / float(TP2 + FP2)
print("Precision Sawah Pulau Sumatra:",precision_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': precision_sawah, 'Kalimantan': precision_sawah
1, 'Sumatra': precision_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Precision Sawah')
plt.xlabel('Site')
plt.ylabel('Precision')

precision_bukan_sawah = TN / float (FP + TN)
print("Precision Bukan Sawah Pulau Jawa:",precision_bukan_sawa
h)

precision_bukan_sawah1 = TN1 / float(FP1 + TN1)
print("Precision Bukan Sawah Pulau Kalimantan:",precision_buka
n_sawah1)

precision_bukan_sawah2 = TN2 / float(FP2 + TN2)
```

```
print("Precision Bukan Sawah Pulau Sumatra:",precision_bukan_s
awah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': precision_bukan_sawah, 'Kalimantan': precision
_bukan_sawah1, 'Sumatra': precision_bukan_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Precision Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('Precision')


F1_sawah = (2*(precision_sawah*recall_rate_sawah)/(precision_s
awah + recall_rate_sawah))
print("F1 Sawah Pulau Jawa:",F1_sawah)
F1_sawah1 = (2*(precision_sawah1*recall_rate_sawah1)/(precisio
n_sawah1 + recall_rate_sawah1))
print("F1 Sawah Pulau Kalimantan:",F1_sawah1) F1_sawah2 = (2*
(precision_sawah2*recall_rate_sawah2)/(precision_sawah2 + reca
ll_rate_sawah2))
print("F1 Sawah Pulau Sumatra:",F1_sawah2)
import matplotlib.pyplot as plt import decimal
data = {'Jawa': F1_sawah, 'Kalimantan': F1_sawah1, 'Sumatra':
F1_sawah2}
names = list(data.keys())
values = list(data.values())
fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values) plt.title('F1 Sawah') plt.xlabel('Site')
plt.ylabel('F1')

F1_bukan_sawah = (2*(precision_bukan_sawah*recall_rate_bukan_s
awah)/(precision_bukan_sawah + recall_rate_bukan_sawah))
print("F1 Bukan Sawah Pulau Jawa:",F1_bukan_sawah)

F1_bukan_sawah1 = (2*(precision_bukan_sawah1*recall_rate_bukan
_sawah1)/(precision_bukan_sawah1 + recall_rate_bukan_sawah1))
print("F1 Bukan Sawah Pulau Kalimantan:",F1_bukan_sawah1)

F1_bukan_sawah2 = (2*(precision_bukan_sawah2*recall_rate_bukan
_sawah2)/(precision_bukan_sawah2 + recall_rate_bukan_sawah2))
print("F1 Bukan Sawah Pulau Sumatra:",F1_bukan_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': F1_bukan_sawah, 'Kalimantan': F1_bukan_sawah1,
'Sumatra': F1_bukan_sawah2}
```

```
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('F1 Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('F1')

#Evaluasi dataset pelatihan pada pengujian piksel
#Membaca Gambar Test Data dan Menampilkannya
img_path = "data/paddy/test/Jawa/b151.jpg"
img = load_img(img_path)
img = img_to_array(img)/255
plt.imshow(img)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches = []
windows = sw.generate(img, sw.DimOrder.HeightWidthChannel, 56,
0.6)
for i,window in enumerate(windows):
    _img = img[window.indices()]
    patches.append(_img)
patches = np.array(patches)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total = len(windows)
_x = 0
for i,window in enumerate(windows):
    if _x != window.x:
        n_x = i
        print(n_x)
        break
    _x = window.x


print(n_total,n_x,n_total//n_x)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions = model.predict(patches)
#print(predictions.shape)
#print(predictions)
paddy = np.reshape(predictions,(n_total//n_x,n_x))
paddy = np.rot90(paddy)
paddy = np.flip(paddy,axis=0)
#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor = img.shape[0]//paddy.shape[0]
w_factor = img.shape[1]//paddy.shape[1]
```

```
#print(h_factor)

from scipy.ndimage import zoom
zoomed = zoom(paddy,(h_factor,w_factor))
#print(zoomed.shape)

#Menampilkan hasil gambar level tertentu
plt.imshow(img)
plt.show()

plt.imshow(img)
plt.imshow(zoomed,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Memperoleh Nilai Akurasi data pengujian dalam piksel
print("Minimal level zoomed :", np.min(zoomed))
print("Maximal level zoomed :", np.max(zoomed))
#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized = (zoomed-np.min(zoomed))/(np.max(zoomed)-np.min(zo
omed))
print("Minimal level ternomalisasi:", np.min(normalized))
print("Maximal level ternomalisasi:", np.max(normalized))
print('')
#Menghitung banyaknya piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
banyakpiksel=(normalized>0.5).sum()
akurasi=((normalized>0.5).sum()*100/(normalized<1).sum())
print("Banyaknya piksel yang terlihat sawah :", (banyakpikse
l))
print("Akurasi dalam piksel (100%) :", (akurasi))

#Membaca Gambar Test Data dan Menampilkannya
img_path11 = "data/paddy/test/Jawa/o153.jpg"
img11 = load_img(img_path11)
img11 = img_to_array(img11)/255
plt.imshow(img11)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches11 = []
windows11 = sw.generate(img11, sw.DimOrder.HeightWidthChannel,
56, 0.6)
for i,window in enumerate(windows11):
    _img11 = img11[window.indices()]
    patches11.append(_img11)
patches11 = np.array(patches11)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total11 = len(windows11)
```

```
_x11 = 0
for i,window in enumerate(windows11):
    if _x != window.x:
        n_x11 = i
        print(n_x11)
        break
    _x = window.x


print(n_total11,n_x11,n_total11//n_x11)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions11 = model.predict(patches11)
#print(predictions.shape)
#print(predictions)
paddy11 = np.reshape(predictions11,(n_total11//n_x11,n_x11))
paddy11 = np.rot90(paddy11)
paddy11 = np.flip(paddy11,axis=0)
#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor11 = img11.shape[0]//paddy11.shape[0]
w_factor11 = img11.shape[1]//paddy11.shape[1]
#print(h_factor)

from scipy.ndimage import zoom
zoomed11 = zoom(paddy11,(h_factor11,w_factor11))
#print(zoomed.shape)

#Menampilkan hasil gambar level tertentu
plt.imshow(img11)
plt.show()

plt.imshow(img11)
plt.imshow(zoomed11,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Memperoleh Nilai Akurasi data pengujian dalam piksel
print("Minimal level zoomed :", np.min(zoomed11))
print("Maximal level zoomed :", np.max(zoomed11))
#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized11 = (zoomed11-np.min(zoomed11))/(np.max(zoomed11)-n
p.min(zoomed11))
print("Minimal level ternomalisasi:", np.min(normalized11))
print("Maximal level ternomalisasi:", np.max(normalized11))
print('')
#Menghitung banyaknya piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
banyakpiksel11=(normalized11<0.5).sum()
akurasi11=((normalized11<0.5).sum()*100/(normalized11<1).sum
())
```

```
print("Banyaknya piksel yang terlihat bukan sawah :", (banyakp
iksel11))
print("Akurasi dalam piksel (100%) :", (akurasi11))

#Evaluasi penguajian luasan lahan sawah
#Membaca Gambar Test Data dan Menampilkannya
img_path22 = "data/paddy/test/test_image/test_image1.jpg"
img22 = load_img(img_path22)
img22 = img_to_array(img22)/255
plt.imshow(img22)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches22 = []
windows22 = sw.generate(img22, sw.DimOrder.HeightWidthChannel,
56, 0.6)
for i,window in enumerate(windows22):
    _img22 = img22[window.indices()]
    patches22.append(_img22)
patches22 = np.array(patches22)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total22 = len(windows22)
_x = 0
for i,window in enumerate(windows22):
    if _x != window.x:
        n_x22 = i
        print(n_x22)
        break
    _x = window.x


print(n_total22,n_x22,n_total22//n_x22)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions22 = model.predict(patches22)
print(predictions22.shape)
print(predictions22)

paddy22 = np.reshape(predictions22,(n_total22//n_x22,n_x22))
paddy22 = np.rot90(paddy22)
paddy22 = np.flip(paddy22,axis=0)
plt.imshow(paddy22,cmap="jet")

#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor22 = img22.shape[0]//paddy22.shape[0]
w_factor22 = img22.shape[1]//paddy22.shape[1]
print(h_factor22)

from scipy.ndimage import zoom
```

```
zoomed22 = zoom(paddy22,(h_factor22,w_factor22))

print(zoomed22.shape)

#Menampilkan hasil gambar level tertentu
plt.imshow(img22)
plt.show()

plt.imshow(img22)
plt.imshow(zoomed22,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Menghitung luasan sawah
#Menampilkan nilai level zoomed dalam range acak
print("Minimal level zoomed :", np.min(zoomed22))
print("Maximal level zoomed :", np.max(zoomed22))

#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized22 = (zoomed22-np.min(zoomed22))/(np.max(zoomed22)-n
p.min(zoomed22))
print("Minimal level ternomalisasi:", np.min(normalized22))
print("Maximal level ternomalisasi:", np.max(normalized22))

#Menghitung luas lahan sawah dalam piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
luaspiksel22=(normalized22>0.5).sum()
akurasi22=((normalized22>0.5).sum()*100/(normalized22<1).sum
())
print("Luas lahan sawah dalam piksel :", (luaspiksel22))
print("Persentase lahan sawah dalam piksel (100%) :", (akurasi
22))

#Setelah diketahui luas sawah dalam piksel, kemudian menkonver
si dalam meter dan hektar
#Untuk menghitung luas maka dilakukan dengan 0,3 x 0,3 = 0,09
luasmtr = 0.004624*(luaspiksel22)
luasha = (luasmtr)/10000
print("Luas lahan sawah dalam meter persegi :", (luasmtr))
print("Luas lahan sawah dalam hektar :", (luasha))
```

Lampiran 2. Sourcecode Identifikasi Lahan Pertanian Sawah dengan Menggunakan Arsitektur *VGG-16Net*

```
#Mengimpor library
!pip install slidingwindow
import slidingwindow as sw
import numpy as np
import os
import keras.backend as K
import keras
import pandas as pd
from keras.models import Sequential
from keras import layers
from keras.preprocessing.image import load_img, img_to_array
import matplotlib.pyplot as plt
from skimage.io import imread, imsave
from skimage.transform import resize
from sklearn.utils.class_weight import compute_class_weight

from sklearn import svm, datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.utils.multiclass import unique_labels

#Mengambil dataset pada github yang ketinggian 100m atau 300m
!apt-get install subversion > /dev/null
!svn export https://github.com/Gio1709/paddy_100m/trunk/data/
> /dev/null

!apt-get install subversion > /dev/null
!svn export https://github.com/Gio1709/paddy_300m/trunk/data/
> /dev/null

#Perintah untuk melihat list bagian dari file yang sudah diund
uh dari Github
!ls data/paddy/train
#gambar dengan nama awalan dengan "b" adalah gambar yang terda
pat padi"
#gambar dengan nama awalan dengan "o" adalah gambar yang tidak
terdapat padi"

#Gambar yang terdapat lahan sawah padi
plt.title("Gambar yang terdapat lahan sawah padi") plt.imshow
(imread("data/paddy/train/b58.jpg")) plt.show()
#Gambar yang terdapat tidak lahan sawah padi plt.title("Gambar
yang tidak terdapat lahan sawah padi") plt.imshow(imread("data
/paddy/train/o125.jpg")) plt.show()


#Membuat patches dengan metode sliding windows
def save_to_folder(img_path,directory,size=56,overlap=0.4):
    #Membaca gambar path penyimpanan window ke direktori/file
yang spesifik
```

```
    img = imread(img_path)
    basename = os.path.basename(img_path)
    fname = os.path.splitext(basename)[0]
    windows = sw.generate(img, sw.DimOrder.HeightWidthChannel,
size, overlap)
    for i,window in enumerate(windows):
        ii = str("{0:05d}".format(i))
        _fname = fname + "_" + ii + ".jpg"
        _img = img[window.indices()]
        if _img.shape !=(size,size,3):
            continue
        img_target_path = os.path.join(directory,_fname)

        imsave(img_target_path,_img, check_contrast=False)

files = os.listdir("data/paddy/train/")
files = [x for x in files if x.endswith(".jpg")]

#Menyimpan gambar yang telah dicacah pada folder chopped
for file in files:
    save_directory = "data/paddy/chopped/"
    file = os.path.join("data/paddy/train/",file)
    save_to_folder(file,save_directory)

#Mengembalikan daftar gambar yang telah diacak pada direktori
#Data yang dilakukan pada model evaluasi

#Dataset Pulau Jawa
files = os.listdir("data/paddy/chopped/")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset Pelatihan
x = []
y = []

for file in files:
    if file.startswith("b"):
        label = 1
    else:
        label = 0

    img_path = os.path.join("data/paddy/chopped/",file)
    img = load_img(img_path)
    img = img_to_array(img)/255
    x.append(img)
    y.append(label)

X = np.array(x)
Y = np.array(y)
print("Bentuk array dari dataset train (pelatihan) adalah :",
(X.shape,Y.shape))

files = os.listdir("data/paddy/test/Jawa")
files = [x for x in files if x.endswith(".jpg")]
```

```
#Memuat dataset pengujian

X_test1 = []
Y_test1 = []

for file in files:
    if file.startswith("b"):
        label1 = 1
    else:
        label1 = 0

    img_path1 = os.path.join("data/paddy/test/Jawa",file)
    img1 = load_img(img_path1)
    img1 = img_to_array(img1)/255.
    img1 = resize(img1,(56,56))
    X_test1.append(img1)
    Y_test1.append(label1)

x_test1 = np.array(X_test1)
y_test1 = np.array(Y_test1)
print("Bentuk array dari dataset test pulau Jawa (pengujian) a
dalah :", (x_test1.shape,y_test1.shape))

files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

files = os.listdir("data/paddy/test/Kalimantan")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset pengujian

X_test2 = []
Y_test2 = []

for file in files:
    if file.startswith("b"):
        label2 = 1
    else:
        label2 = 0

    img_path2 = os.path.join("data/paddy/test/Kalimantan",fil
e)
    img2 = load_img(img_path2)
    img2 = img_to_array(img2)/255.
    img2 = resize(img2,(56,56))
    X_test2.append(img2)
    Y_test2.append(label2)

x_test2 = np.array(X_test2)
y_test2 = np.array(Y_test2)
print("Bentuk array dari dataset test pulau Kalimantan (penguj
ian) adalah :", (x_test2.shape,y_test2.shape))
```

```
files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

files = os.listdir("data/paddy/test/Sumatra")
files = [x for x in files if x.endswith(".jpg")]

#Memuat dataset pengujian

X_test3 = []
Y_test3 = []

for file in files:
    if file.startswith("b"):
        label3 = 1
    else:
        label3 = 0

    img_path3 = os.path.join("data/paddy/test/Sumatra",file)
    img3 = load_img(img_path3)
    img3 = img_to_array(img3)/255.
    img3 = resize(img3,(56,56))
    X_test3.append(img3)
    Y_test3.append(label3)

x_test3 = np.array(X_test3)
y_test3 = np.array(Y_test3)
print("Bentuk array dari dataset test pulau Sumatra (pengujia
n) adalah :", (x_test3.shape,y_test3.shape))

files= pd.Index(files)
print("Tampilan list gambar yang telah dichop :\n",files.value
_counts())

#Membagi dataset menjadi dua yaitu pelatihan dan validasi
from sklearn.model_selection import train_test_split

train_x, valid_x, train_y, valid_y = train_test_split(X,Y,test
_size=0.25,stratify = y)

print("Bentuk array dari dataset train (pelatihan) adalah:",tr
ain_x.shape,train_y.shape)
print("Bentuk array dari dataset validation adalah:",valid_x.s
hape,valid_y.shape)

#Mengkomputasi kelas bobotnya
print(np.unique(train_y,return_counts=True))
print(np.unique(valid_y,return_counts=True))

cw = compute_class_weight("balanced",np.unique(train_y),train_
y)
print(cw)
```

```
#Pembuatan Model CNN
import tensorflow as tf
from sklearn import svm, datasets
from sklearn.metrics import confusion_matrix
import pandas as pd

IMG_SHAPE = (56, 56, 3)
# Membuat model dasar (base model) dari pre-trained model Mobi
leNet
base_model = tf.keras.applications.VGG16(input_shape=IMG_SHAP
E,include_top=False,weights='imagenet')

base_model.trainable = True
base_model.summary()

model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32,activation="relu"),
    tf.keras.layers.Dense(1),
    tf.keras.layers.Activation("sigmoid",name="prediction")
]
)

model.compile(optimizer="adam",loss="binary_crossentropy",metr
ics=["acc"])

model.summary()


from keras.callbacks import EarlyStopping, ModelCheckpoint

#Menyimpan model bobot yang terbaik selama training
ckpt = ModelCheckpoint("paddy.h5", monitor='val_loss', verbose
=1, save_best_only=True, save_weights_only=False, mode='auto',
period=1)

#Menampilkan kurva akurasi dan kegagalan pada model
history = model.fit(train_x,train_y,batch_size=200,epochs=50,v
alidation_data=(valid_x,valid_y),class_weight=cw, callbacks =
[ckpt])

plt.plot(history.history["acc"],label="Akurasi Pelatihan")
plt.plot(history.history["val_acc"],label="Validasi Akurasi")
plt.legend()
plt.show()

plt.plot(history.history["loss"],label="Kegagalan Pelatihan")
plt.plot(history.history["val_loss"],label="Validasi Kegagalan
")
plt.legend()
plt.show()
#Memeriksa matriks model
```

```
print(model.metrics_names)
#Evaluasi data training
print(model.evaluate(train_x,train_y))
#Evaluasi validasi data
print(model.evaluate(valid_x,valid_y))
#Evaluasi test data
print('[Loss, Acc] Test Pulau Jawa', model.evaluate(x= x_test
1, y = y_test1))
print('[Loss, Acc] Test Pulau Kalimantan', model.evaluate(x= x
_test2, y = y_test2))
print('[Loss, Acc] Test Pulau Sumatra', model.evaluate(x= x_te
st3, y = y_test3))

y_pred_class1 = model.predict_classes(x_test1)
y_pred_class2 = model.predict_classes(x_test2)
y_pred_class3 = model.predict_classes(x_test3)

#Jumlah data prediksi dari data test
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Jawa :', y_pred_class1.transpose())
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Kalimantan :', y_pred_class2.transpose())
print('Jumlah data prediksi hasil dari dataset pengujian pada
Pulau Sumatra :', y_pred_class3.transpose())

from sklearn import metrics
print("Akurasi pada data Pulau Jawa:",metrics.accuracy_score(y
_test1, y_pred_class1))
print("Akurasi pada data Pulau Kalimantan:",metrics.accuracy_s
core(y_test2, y_pred_class2))
print("Akurasi pada data Pulau Sumatra:",metrics.accuracy_scor
e(y_test3, y_pred_class3))

# Mengetahui class berdasarkan gambar sawah dan bukan sawah
# Apabila 1 menandakan gambar sawah dan 0 bukan sawah
Y_test1 = pd.Index(y_test1)
print("Class dan jumlah data Pulau Jawa:\n Class Jumlah \n",Y_
test1.value_counts())
print("\n")
Y_test2 = pd.Index(y_test2)
print("Class dan jumlah data Pulau Kalimantan:\n Class Jumlah
\n",Y_test2.value_counts())
print("\n")
Y_test3 = pd.Index(y_test3)
print("Class dan jumlah data Pulau Sumatra:\n Class Jumlah \n
",Y_test3.value_counts())
print("\n")

# Menampilkan 20 data pertama/awal true atau pun prediksi
print("Dataset Pulau Jawa:")
print('True:', Y_test1.values[0:20])
print('False:', y_pred_class1.transpose()[0:20])
print("\n")
print("Dataset Pulau Kalimantan:")
```

```
print('True:', Y_test2.values[0:20])
print('False:', y_pred_class2.transpose()[0:20])
print("\n")
print("Dataset Pulau Sumatra:")
print('True:', Y_test3.values[0:20])
print('False:', y_pred_class3.transpose()[0:20])

# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion = metrics.confusion_matrix(Y_test1, y_pred_class1)
 print("Matriks hasil clasification Pulau Jawa:\n",confusion)
#[row, column]
TP = confusion[1, 1]
TN = confusion[0, 0]
FP = confusion[0, 1]
FN = confusion[1, 0]
# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion1 = metrics.confusion_matrix(Y_test2, y_pred_class2)
print("Matriks hasil clasification Pulau Kalimantan:\n",confus
ion1)
#[row, column]
TP1 = confusion1[1, 1]
TN1 = confusion1[0, 0]
FP1 = confusion1[0, 1]
FN1 = confusion1[1, 0]
# IMPORTANT: first argument is true values, second argument is
predicted values # this produces a 2x2 numpy array (matrix)
# save confusion matrix and slice into four pieces
confusion2 = metrics.confusion_matrix(Y_test3, y_pred_class3)
print("Matriks hasil clasification Pulau Sumatra:\n",confusion
2)
#[row, column]
TP2 = confusion2[1, 1]
TN2 = confusion2[0, 0]
FP2 = confusion2[0, 1]
FN2 = confusion2[1, 0]

# Classification accuracy dari ketiga pulau
#print((TP + TN) / float(TP + TN + FP + FN))
print("Classification accuracy Pulau Jawa:",metrics.accuracy_s
core(Y_test1, y_pred_class1))

#print((TP1 + TN1) / float(TP1 + TN1 + FP1 + FN1))
print("Classification accuracy Pulau Kalimantan:",metrics.accu
racy_score(Y_test2, y_pred_class2))

#print((TP2 + TN2) / float(TP2 + TN2 + FP2 + FN2))
print("Classification accuracy Pulau Sumatra:",metrics.accurac
y_score(Y_test3, y_pred_class3))

import matplotlib.pyplot as plt
```

```
data = {'Jawa': metrics.accuracy_score(Y_test1, y_pred_class
1), 'Kalimantan': metrics.accuracy_score(Y_test2, y_pred_class
2), 'Sumatra': metrics.accuracy_score(Y_test3, y_pred_class3)}
names = list(data.keys())
values = list(data.values())

fig, axs = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
axs.bar(names, values)

plt.title('Classification Accuracy')
plt.xlabel('Site')
plt.ylabel('Accuracy')

recall_rate_sawah = TP / float(FN + TP)

print("Recall Rate Sawah Pulau Jawa:",recall_rate_sawah)
#print(metrics.recall_score(test_y, y_pred_class))

recall_rate_sawah1 = TP1 / float(FN1 + TP1)

print("Recall Rate Sawah Pulau Kalimantan:",recall_rate_sawah
1)
#print(metrics.recall_score(test_y1, y_pred_class1))

recall_rate_sawah2 = TP2 / float(FN2 + TP2)

print("Recall Rate Sawah Pulau Sumatra:",recall_rate_sawah2)
#print(metrics.recall_score(test_y2, y_pred_class2))

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': recall_rate_sawah, 'Kalimantan': recall_rate_s
awah1, 'Sumatra': recall_rate_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Recall Rate Sawah')
plt.xlabel('Site')
plt.ylabel('Recall Rate')

recall_rate_bukan_sawah = TN / (TN + FP)
print("Recall Rate Bukan Sawah Pulau Jawa:",recall_rate_bukan_
sawah)

recall_rate_bukan_sawah1 = TN1 / (TN1 + FP1)
print("Recall Rate Bukan Sawah Pulau Kalimantan:",recall_rate_
bukan_sawah1)

recall_rate_bukan_sawah2 = TN2 / (TN2 + FP2)
```

```
print("Recall Rate Bukan Sawah Pulau Sumatra:",recall_rate_buk
an_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': recall_rate_bukan_sawah, 'Kalimantan': recall_
rate_bukan_sawah1, 'Sumatra': recall_rate_bukan_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Recall Rate Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('Recall Rate')

precision_sawah = TP / float(TP + FP)
print("Precision Sawah Pulau Jawa:",precision_sawah)

precision_sawah1 = TP1 / float(TP1 + FP1)
print("Precision Sawah Pulau Kalimantan:",precision_sawah1)

precision_sawah2 = TP2 / float(TP2 + FP2)
print("Precision Sawah Pulau Sumatra:",precision_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': precision_sawah, 'Kalimantan': precision_sawah
1, 'Sumatra': precision_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Precision Sawah')
plt.xlabel('Site')
plt.ylabel('Precision')

precision_bukan_sawah = TN / float (FP + TN)
print("Precision Bukan Sawah Pulau Jawa:",precision_bukan_sawa
h)

precision_bukan_sawah1 = TN1 / float(FP1 + TN1)
print("Precision Bukan Sawah Pulau Kalimantan:",precision_buka
n_sawah1)

precision_bukan_sawah2 = TN2 / float(FP2 + TN2)
print("Precision Bukan Sawah Pulau Sumatra:",precision_bukan_s
awah2)
```

```
import matplotlib.pyplot as plt
import decimal

data = {'Jawa': precision_bukan_sawah, 'Kalimantan': precision
_bukan_sawah1, 'Sumatra': precision_bukan_sawah2}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('Precision Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('Precision')


F1_sawah = (2*(precision_sawah*recall_rate_sawah)/(precision_s
awah + recall_rate_sawah))
print("F1 Sawah Pulau Jawa:",F1_sawah)
F1_sawah1 = (2*(precision_sawah1*recall_rate_sawah1)/(precisio
n_sawah1 + recall_rate_sawah1))
print("F1 Sawah Pulau Kalimantan:",F1_sawah1) F1_sawah2 = (2*
(precision_sawah2*recall_rate_sawah2)/(precision_sawah2 + reca
ll_rate_sawah2))
print("F1 Sawah Pulau Sumatra:",F1_sawah2)
import matplotlib.pyplot as plt import decimal
data = {'Jawa': F1_sawah, 'Kalimantan': F1_sawah1, 'Sumatra':
F1_sawah2}
names = list(data.keys())
values = list(data.values())
fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values) plt.title('F1 Sawah') plt.xlabel('Site')
plt.ylabel('F1')

F1_bukan_sawah = (2*(precision_bukan_sawah*recall_rate_bukan_s
awah)/(precision_bukan_sawah + recall_rate_bukan_sawah))
print("F1 Bukan Sawah Pulau Jawa:",F1_bukan_sawah)

F1_bukan_sawah1 = (2*(precision_bukan_sawah1*recall_rate_bukan
_sawah1)/(precision_bukan_sawah1 + recall_rate_bukan_sawah1))
print("F1 Bukan Sawah Pulau Kalimantan:",F1_bukan_sawah1)

F1_bukan_sawah2 = (2*(precision_bukan_sawah2*recall_rate_bukan
_sawah2)/(precision_bukan_sawah2 + recall_rate_bukan_sawah2))
print("F1 Bukan Sawah Pulau Sumatra:",F1_bukan_sawah2)

import matplotlib.pyplot as plt
import decimal

data = {'Jawa': F1_bukan_sawah, 'Kalimantan': F1_bukan_sawah1,
'Sumatra': F1_bukan_sawah2}
names = list(data.keys())
values = list(data.values())
```

```
fig, ax = plt.subplots(1, 1, figsize=(6, 6), sharey=True)
ax.bar(names, values)

plt.title('F1 Bukan Sawah')
plt.xlabel('Site')
plt.ylabel('F1')

#Evaluasi dataset pelatihan pada pengujian piksel
#Membaca Gambar Test Data dan Menampilkannya
img_path = "data/paddy/test/Jawa/b151.jpg"
img = load_img(img_path)
img = img_to_array(img)/255
plt.imshow(img)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches = []
windows = sw.generate(img, sw.DimOrder.HeightWidthChannel, 56,
0.6)
for i,window in enumerate(windows):
    _img = img[window.indices()]
    patches.append(_img)
patches = np.array(patches)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total = len(windows)
_x = 0
for i,window in enumerate(windows):
    if _x != window.x:
        n_x = i
        print(n_x)
        break
    _x = window.x


print(n_total,n_x,n_total//n_x)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions = model.predict(patches)
#print(predictions.shape)
#print(predictions)
paddy = np.reshape(predictions,(n_total//n_x,n_x))
paddy = np.rot90(paddy)
paddy = np.flip(paddy,axis=0)
#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor = img.shape[0]//paddy.shape[0]
w_factor = img.shape[1]//paddy.shape[1]
#print(h_factor)

from scipy.ndimage import zoom
```

```
zoomed = zoom(paddy,(h_factor,w_factor))
#print(zoomed.shape)

#Menampilkan hasil gambar level tertentu
plt.imshow(img)
plt.show()

plt.imshow(img)
plt.imshow(zoomed,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Memperoleh Nilai Akurasi data pengujian dalam piksel
print("Minimal level zoomed :", np.min(zoomed))
print("Maximal level zoomed :", np.max(zoomed))
#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized = (zoomed-np.min(zoomed))/(np.max(zoomed)-np.min(zo
omed))
print("Minimal level ternomalisasi:", np.min(normalized))
print("Maximal level ternomalisasi:", np.max(normalized))
print('')
#Menghitung banyaknya piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
banyakpiksel=(normalized>0.5).sum()
akurasi=((normalized>0.5).sum()*100/(normalized<1).sum())
print("Banyaknya piksel yang terlihat sawah :", (banyakpikse
l))
print("Akurasi dalam piksel (100%) :", (akurasi))

#Membaca Gambar Test Data dan Menampilkannya
img_path11 = "data/paddy/test/Jawa/o153.jpg"
img11 = load_img(img_path11)
img11 = img_to_array(img11)/255
plt.imshow(img11)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches11 = []
windows11 = sw.generate(img11, sw.DimOrder.HeightWidthChannel,
56, 0.6)
for i,window in enumerate(windows11):
    _img11 = img11[window.indices()]
    patches11.append(_img11)
patches11 = np.array(patches11)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total11 = len(windows11)
_x11 = 0
for i,window in enumerate(windows11):
    if _x != window.x:
```

```
            n_x11 = i
            print(n_x11)
            break
        _x = window.x


print(n_total11,n_x11,n_total11//n_x11)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions11 = model.predict(patches11)
#print(predictions.shape)
#print(predictions)
paddy11 = np.reshape(predictions11,(n_total11//n_x11,n_x11))
paddy11 = np.rot90(paddy11)
paddy11 = np.flip(paddy11,axis=0)
#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor11 = img11.shape[0]//paddy11.shape[0]
w_factor11 = img11.shape[1]//paddy11.shape[1]
#print(h_factor)

from scipy.ndimage import zoom
zoomed11 = zoom(paddy11,(h_factor11,w_factor11))
#print(zoomed.shape)

#Menampilkan hasil gambar level tertentu
plt.imshow(img11)
plt.show()

plt.imshow(img11)
plt.imshow(zoomed11,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Memperoleh Nilai Akurasi data pengujian dalam piksel
print("Minimal level zoomed :", np.min(zoomed11))
print("Maximal level zoomed :", np.max(zoomed11))
#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized11 = (zoomed11-np.min(zoomed11))/(np.max(zoomed11)-n
p.min(zoomed11))
print("Minimal level ternomalisasi:", np.min(normalized11))
print("Maximal level ternomalisasi:", np.max(normalized11))
print('')
#Menghitung banyaknya piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
banyakpiksel11=(normalized11<0.5).sum()
akurasi11=((normalized11<0.5).sum()*100/(normalized11<1).sum
())
print("Banyaknya piksel yang terlihat bukan sawah :", (banyakp
iksel11))
print("Akurasi dalam piksel (100%) :", (akurasi11))
```

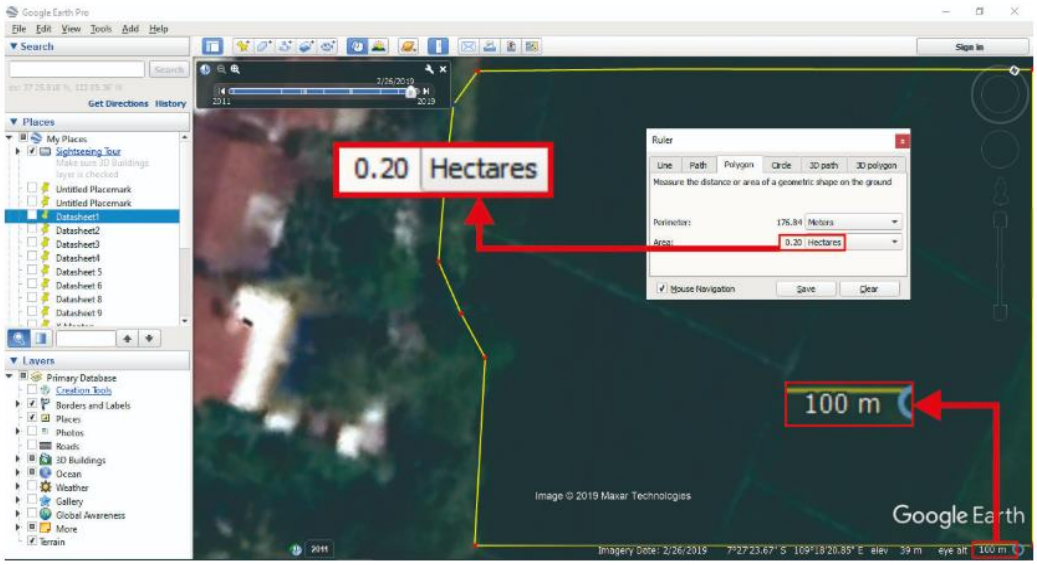
```
#Evaluasi penguajian luasan lahan sawah
#Membaca Gambar Test Data dan Menampilkannya
img_path22 = "data/paddy/test/test_image/test_image1.jpg"
img22 = load_img(img_path22)
img22 = img_to_array(img22)/255
plt.imshow(img22)
plt.show()

#Membuat subimages dengan patches menggunkan metode sliding wi
ndows
patches22 = []
windows22 = sw.generate(img22, sw.DimOrder.HeightWidthChannel,
56, 0.6)
for i,window in enumerate(windows22):
    _img22 = img22[window.indices()]
    patches22.append(_img22)
patches22 = np.array(patches22)

#Mengecek total gambar dan banyaknya gambar yang terpasang pad
a height direction, maupun width direction
n_total22 = len(windows22)
_x = 0
for i,window in enumerate(windows22):
    if _x != window.x:
        n_x22 = i
        print(n_x22)
        break
    _x = window.x


print(n_total22,n_x22,n_total22//n_x22)

#Memprediksi probalititas keberadaaan lahan sawah padi terhada
p gambar yang telah dicacah
#dan kembali pada gambar original
predictions22 = model.predict(patches22)
print(predictions22.shape)
print(predictions22)

paddy22 = np.reshape(predictions22,(n_total22//n_x22,n_x22))
paddy22 = np.rot90(paddy22)
paddy22 = np.flip(paddy22,axis=0)
plt.imshow(paddy22,cmap="jet")

#Array pada gambar diperbesar menggunakan interpolasi spline
h_factor22 = img22.shape[0]//paddy22.shape[0]
w_factor22 = img22.shape[1]//paddy22.shape[1]
print(h_factor22)

from scipy.ndimage import zoom
zoomed22 = zoom(paddy22,(h_factor22,w_factor22))

print(zoomed22.shape)
```

```
#Menampilkan hasil gambar level tertentu
plt.imshow(img22)
plt.show()

plt.imshow(img22)
plt.imshow(zoomed22,alpha=0.3,vmax=1.0,vmin=0,cmap="jet")

plt.colorbar(fraction=0.027, pad=0.04,alpha=0)
plt.show()

#Menghitung luasan sawah
#Menampilkan nilai level zoomed dalam range acak
print("Minimal level zoomed :", np.min(zoomed22))
print("Maximal level zoomed :", np.max(zoomed22))

#Menampilkan nilai level zoomed dalam range 0 sampai 1
normalized22 = (zoomed22-np.min(zoomed22))/(np.max(zoomed22)-n
p.min(zoomed22))
print("Minimal level ternomalisasi:", np.min(normalized22))
print("Maximal level ternomalisasi:", np.max(normalized22))

#Menghitung luas lahan sawah dalam piksel
#Tingkatan level warna dari 0 sampai 1 yang mencapai bentuk sa
wah adalah 0.5 (Nilai diantara 0 dan 1)
luaspiksel22=(normalized22>0.5).sum()
akurasi22=((normalized22>0.5).sum()*100/(normalized22<1).sum
())
print("Luas lahan sawah dalam piksel :", (luaspiksel22))
print("Persentase lahan sawah dalam piksel (100%) :", (akurasi
22))

#Setelah diketahui luas sawah dalam piksel, kemudian menkonver
si dalam meter dan hektar
#Untuk menghitung luas maka dilakukan dengan 0,3 x 0,3 = 0,09
luasmtr = 0.004624*(luaspiksel22)
luasha = (luasmtr)/10000
print("Luas lahan sawah dalam meter persegi :", (luasmtr))
print("Luas lahan sawah dalam hektar :", (luasha))
```

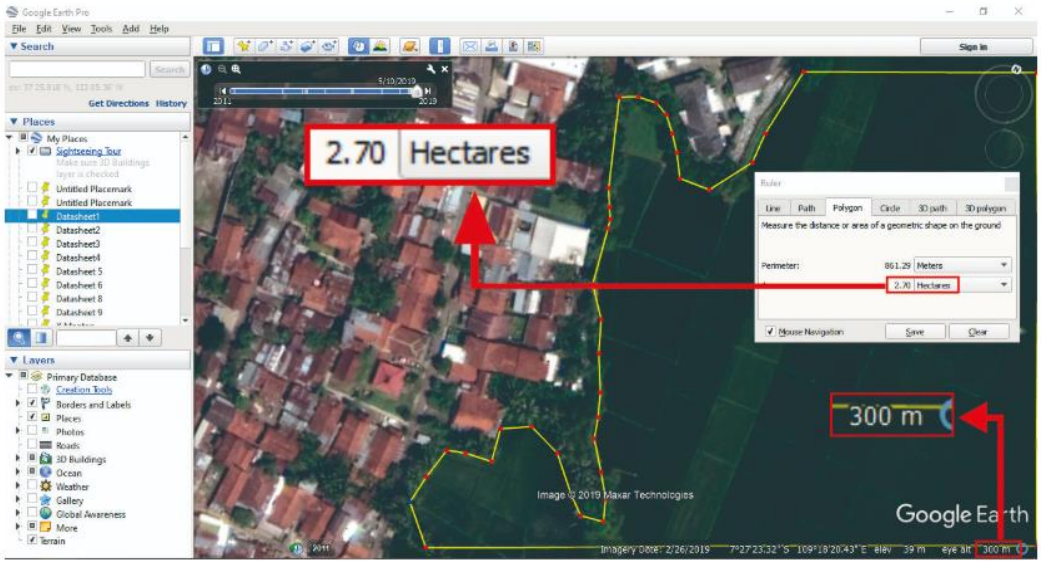
***Lampiran 3*. Jumlah Piksel pada Sampel Pengujian Ketinggian 100 Meter**

***Lampiran 4*. Jumlah Piksel pada Sampel Pengujian Ketinggian 300 Meter**

Lampiran 5*. Hasil Pengukuran Piksel dalam Sampel Pengujian Ketinggian 100 Meter dengan Menggunakan Fitur pada *Google Earth Pro



Lampiran 6*. Hasil Pengukuran Piksel dalam Sampel Pengujian Ketinggian 300 Meter dengan Menggunakan Fitur pada *Google Earth Pro



BIODATA PENULIS

Biodata penulis berisi terkait dengan identitas penulis (nama, kontak email), riwayat akademis (pendidikan) penulis ditulis dari yang paling , skill, serta prestasi penulis.

A. Identitas

Nama	:	Fendy Prayogi
NIM	:	H1A016029
Tempat, tanggal lahir	:	Jakarta, 18 Mei 1998
Alamat	:	Jl. Pinus 1. Perum Green Garden, Blok B 12 No.18 010/002. Kec.Cilincing. Kota. Jakarta Utara. Prov. DKI Jakarta
No. Telp.	:	082213871714
Alamat e-mail	:	prayogifendy@gmail.com

B. Riwayat Pendidikan Akademik

Periode	Jenjang	Institusi
2016 – 2020	S1	Teknik Elektro Universitas Jenderal Soedirman
2013 – 2016	SMA	SMA Negeri 102 Jakarta
2010 – 2013	SMP	SMP Negeri 200 Jakarta

C. Riwayat Pendidikan Non Formal (jika ada)

Tahun	Keahlian	Penyelenggara	Kota

D. Prestasi

Tahun	Tingkat	Prestasi

E. Keahlian (tuliskan secara diskriptif)

Memiliki minat di bidang otomasi industri, elektronika, dan pengolahan citra digital. Menjadi pengurus aktif dalam UKM Salman MM Teknik selama 1 tahun . Dapat mengoperasikan *cx-programmer*, *Simatic Manager*, *proteus isis*, ETAP, *corel draw*, dan *adobe photoshop*. Menjadi asisten praktikum dengan mata kuliah Sistem Kendali.