TUGAS UTS

PROGRAM KLASIFIKASI BINER GEDUNG OKTAGON DENGAN MENGGUNAKAN METODE TRANSFER LEARNING INCEPTION V3, VVG16, CNN, DAN SUPPORT VECTOR MACHINE UNTUK MEMUDAHKAN TURIS DALAM MENDAPATKAN INFORMASI OKTAGON DI ITB

TF4063 Sains Data Rekayasa

oleh:

Gregory Sembiring Brahmana	13317001
Bernardus Rendy	13317041
Nicholas Biantoro	13317043
Murphy Halim	13317069
Putu Bhargo Abhimana Chrysnanda	13317084

Dosen :

Dr.Ir. Eko Mursito Budi MT

Miranti I. Mandasari, Ph.D.

Dr. Fadjar Fathurrahman

SARJANA TEKNIK FISIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI BANDUNG

2019

DAFTAR ISI

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Institut Teknologi Bandung merupakan salah satu perguruan tinggi negeri yang cukup terkenal di kalangan masyarakat dalam negeri maupun luar negeri. Hal tersebut menyebabkan ITB sering mendapatkan kunjungan baik itu dari kalangan luar negeri maupun dalam negeri. Kunjungan tersebut dapat bersifat formal maupun tidak formal. Kunjungan formal biasanya dipandu oleh instansi yang mengundang atau menerima undangan kunjungan. Misalnya saja kunjungan Unit Tenis Meja Unpar akan mendapatkan sambutan dari Unit Tenis Meja ITB begitu juga sebaliknya. Kunjungan tidak formal biasanya dilakukan oleh orang-orang yang ingin mengetahui bagian dalam ITB dan tidak dipandu oleh instansi. Hal ini menyebabkan kesulitan karena tidak ada pemandu saat ingin menelusuri Kampus Gajah ini. Saat turis ingin bertanya tentang nama dan sejarah gedung di ITB, mereka tidak tahu harus bertanya kepada siapa. Mereka merasa sungkan untuk bertanya tentang nama dan sejarah gedung kepada civitas akademik yang tidak dikenalnya. Oleh karena itu, kami berinisiatif untuk menciptakan aplikasi bernama “SITU Ganesha” yang dapat menyediakan informasi terkait sejarah dan nama gedung di ITB. Aplikasi SITU Ganesha cukup mudah untuk digunakan. Turis hanya perlu mengambil foto terkait gedung yang ingin diketahui nama dan sejarahnya, lalu aplikasi akan memberikan informasi berupa gambar gedung (dari sudut yang lebih jelas terlihat), sejarah 1 gedung, penggunaannya saat ini, dan siapa yang memberikan sponsor. Untuk membuat seluruh gedung dapat di amati akan cukup sulit. Maka dari itu, kami membatasi hanya gedung oktagon yang ditampilkan.

1.2 Tujuan

Berdasarkan latar belakang masalah di atas, dapat diturunkan tujuan yang harus di capai yaitu menciptakan aplikasi yang menyediakan menu untuk mengambil foto gedung oktagon, mengirim foto ke server, menerima informasi terkait foto tersebut, dan menampilkannya.

Selain memiliki tujuan yang harus dicapai, terdapat beberapa batasan untuk mempermudah pembuatan aplikasi. Batasan-batasan tersebut adalah:

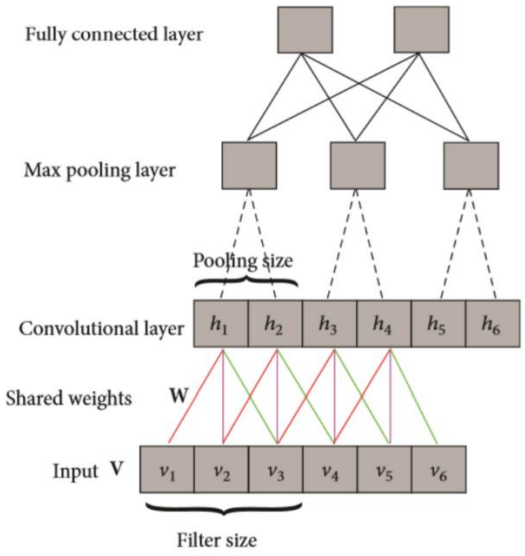
1. Obyek yang ingin ditampilkan informasinya dibatasi pada Gedung Oktagon
2. Turis hanya akan berjalan di jalur tengah ITB, antara gerbang selatan dan gerbang utara, pada pagi – sore hari (tidak malam hari), dan cuaca tidak hujan

BAB 2

TEORI DASAR

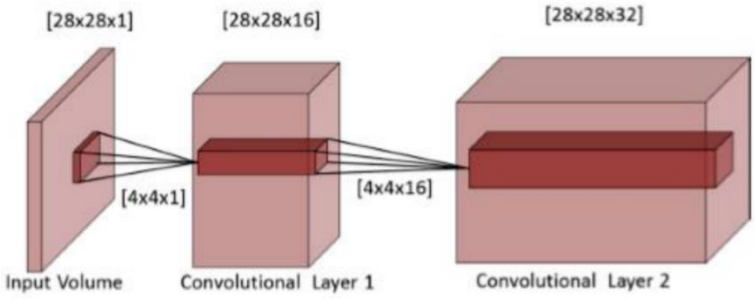
2.1 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah merupakan salah satu jenis neural network yang berisi kombinasi beberapa layer yaitu convolutional layer, pooling layer, dan fully connected layer (Hu, et al. 2015). Convolutioinal Layer memproses data dengan topologi grid (Goodfellow, et al. 2016). Convolutional Neural Network menggunakan operasi convolution pada perkalian matriks di setiap layer. Arsitektur Convolutional Neural Network (CNN) dapat dilihat pada Gambar 2.1 dimana jaringan ini terdiri dari beberapa layer, yakni convolutional layer, pooling layer dan fully-connected layer.



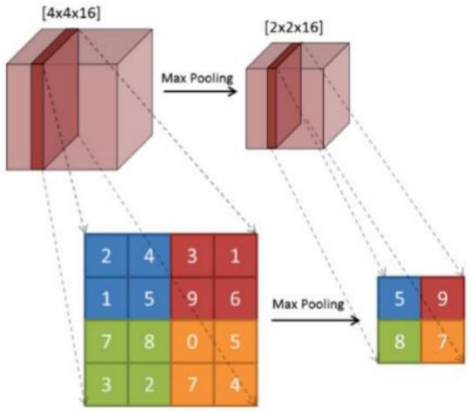
Gambar 2.1 Arsitektur Convolutional Neural Network (Hu, et al. 2015)

Convolutional Layer adalah sebuah inti utama dari CNN, dimana layer ini memiliki sebuah kumpulan filter yang dapat digunakan untuk mempelajari citra masukan. Melalui layer ini, fitur akan di ekstraksi dan kemudian di lanjutkan ke layer berikutnya dengan tujuan untuk mengekstraksi fitur yang lebih kompleks (Bui & Chang, 2016). Contoh diagram Convolutional Layer dapat dilihat pada Gambar 2.2. dimana ukuran citra masukan yang diberikan adalah 28x28 dan filter atau kernel 4x4



Gambar 2.2 Contoh Diagram Convolutional Layer (Bui & Chang, 2016)

Pooling Layer merupakan proses resizing yaitu proses untuk mengubah ukuran citra input yang berbeda,salah satunya dengan menggunakan operasi MAX. Hal ini bertujuan untuk membantu mengurangi jumlah parameter dan waktu perhitungan yang dibutuhkan saat melatih network (Bui & Chang, 2016). Contoh diagram Pooling Layer dapat dilihat pada Gambar 2.3

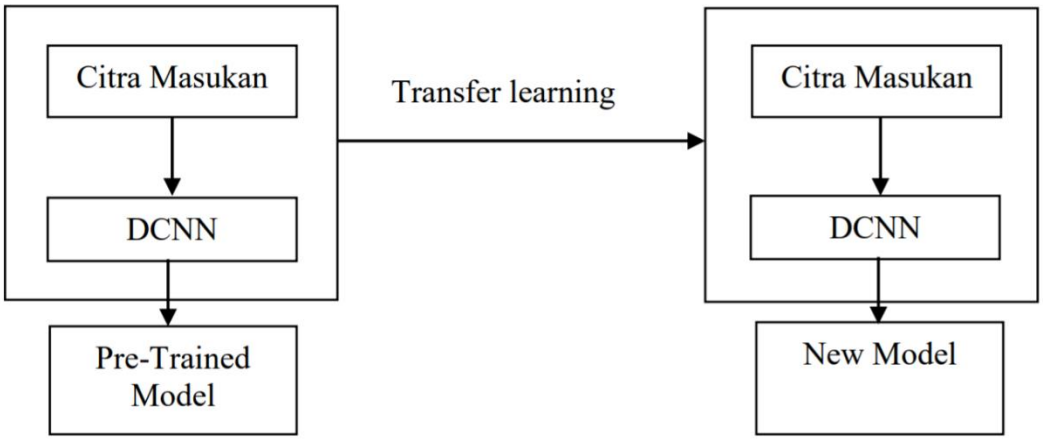


Gambar 2.3 Contoh Diagram MAX Pooling Layer (Bui & Chang, 2016)

Pada Gambar 2.3, citra yang di masukkan berukuran 4x4 kemudian di resize menjadi citra berukuran 2x2 dengan kedalaman masing-masing bernilai 16. Pada Max Pooling, untuk setiap 4 pixels akan diambil satu nilai maksimum. Terlihat pada Gambar 2.3. pada 4 pixels berwarna biru, nilai maksimum yang akan di ambil adalah 5. Pada 4 pixels berwarna merah, nilai maksimum yang akan di ambil adalah 9. Pada pixels berwarna hijau, nilai maksimum yang akan di ambil adalah 8. Pada pixels berwarna orange, nilai maksimum yang akan di ambil adalah 7. Sehingga menghasilkan sebuah citra yang telah diperkecil. Dan layer ketiga pada CNN adalah Fully Connected Layer, dimana layer ini mengambil seluruh neuron pada layer sebelumnya (Convolutional Layer dan MAX Pooling Layer) dan menghubungkannya ke setiap single neuron yang ada (Devikar, 2016).

2.2 Transfer Learning

Transfer learning adalah proses mentransfer pengetahuan dari training sebelumnya untuk dapat digunakan dalam penelitian yang baru sehingga waktu training akan lebih cepat di selesaikan (Devikar, 2016). Hal ini tentu berbeda dengan proses training pada mesin tradisional yang mempelajari input data dari awal dan membutuhkan waktu komputasi yang lama. Visualisasi transfer learning dapat dilihat pada Gambar 2.4.



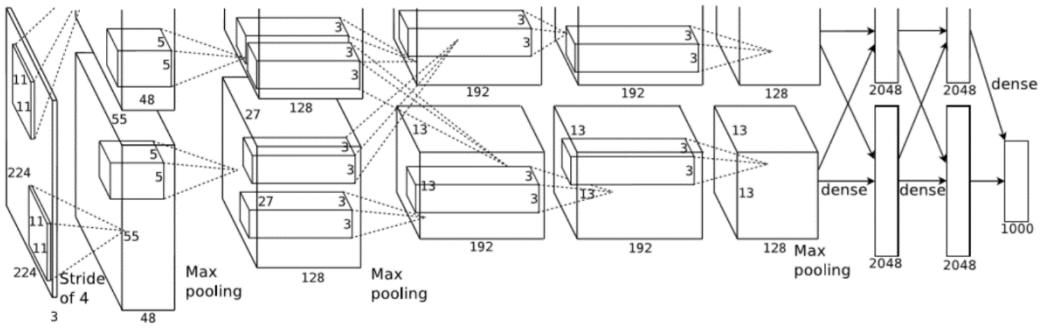
Gambar 2.4 Visualisasi Transfer Learning

2.3 Deep Convolutional Neural Network (DCNN)

Deep Convolutional Neural Network (DCNN) menunjukan performa yang luar biasa dalam bidang *image recognition*, hal ini didukung dari performa DCNN yang sangat baik dalam mengekstraksi *high-level features*. Selain itu, *Convolutional Layer* dan *MAX Pooling Layer* yang digunakan pada DCNN terbukti sangat efektif dalam mengenali bentuk yang bervariasi. DCNN mampu melakukan seluruh tahap pada pengenalan citra yaitu tahap *feature extraction* dan *classifier* secara bersamaan karena DCNN menerima *raw image* sebagai input, sehingga tidak membutuhkan tahap ekstraksi fitur dan *pre-processing* secara terpisah seperti pada *Conventional Classifier* (Kim & Xie, 2015).

Arsitektur *Deep Convolutional Neural Network* dapat dilihat pada Gambar 2.5 dimana terdapat lima *Convolutional Layer* pada layer awal dan tiga layer berikutnya adalah *Fully Connected Layer* di akhir (Krizhevsky, et al. 2012). Output dari *Fully Connected Layer* adalah 1000-way-softmax dan menghasilkan distribusi 1000 kelas label. *Convolutional Layer* yang pertama melakukan filter pada citra masukkan yang memiliki ukuran 224x224x3 dengan 96 kernel yang memiliki ukuran 11x11x3 dengan *stride* 4 pixels, yaitu jarak antara receptive *field* dari neuron tetangga dalam kernel map. Hasil dari

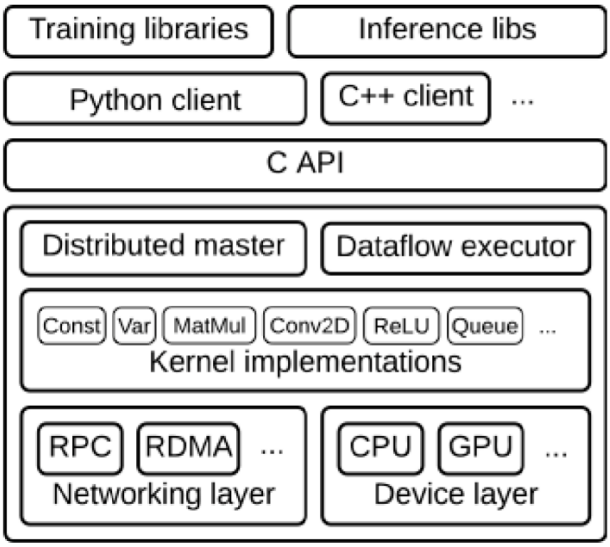
Convolutional Layer yang pertama menjadi masukkan pada *Convolutional Layer* kedua. Pada *Convolutional Layer* kedua, dilakukan filter dengan 256 kernel yang memiliki ukuran 5x5x48. *Convolutional Layer* ketiga, keempat dan kelima terhubung satu sama lain tanpa intervensi *pooling* atau normalisasi layer. *Convolutional Layer* ketiga memiliki 384 kernels berukuran 3x3x256. *Convolutional Layer* keempat memiliki 384 kernels berukuran 3x3x192. *Convolutional Layer* kelima memiliki 256 kernels berukuran 3x3x192. Setelah proses *Convolutional Layer* selesai, dihasilkan *Fully Connected Layer* yang memiliki 4096 neuron (Krizhevsky, et al. 2012).



Gambar 2.5 Arsitektur Deep Convolutional Neural Network (Krizhevsky, et al. 2012)

2.4 TensorFlow

TensorFlow adalah *open source library* untuk *machine learning* yang di release oleh Google yang mendukung beberapa bahasa pemrograman (Devikar, 2016). Dalam proses *Transfer Learning, Tensorflow* berperan untuk memproses *Inception-v3 Model* untuk di training ulang menggunakan data yang baru dan kemudian menghasilkan *classifier* dengan komputasi yang cepat dan akurasi yang baik. *Tensorflow* dapat digunakan pada semua sistem operasi. Arsitektur umum dari *Tensorflow* dapat dilihat pada Gambar 2.6.



Gambar 2.6 Arsitektur Umum Tensorflow (www.tensorflow.org)

2.5 Metode Transfer Learning dengan menggunakan Inception V3

Berdasarkan yang sudah disebutkan sebelumnya, transfer learning adalah proses mentransfer pengetahuan dari training sebelumnya untuk dapat digunakan dalam penelitian. Terdapat berbagai model yang dapat digunakan sebagai *Pre-Trained Model*. Salah satunya adalah dengan menggunakan *inception v3*. *Inception v3* merupakan CNN yang sudah di latih lebih dari satu juta gambar dari ImageNet database. Model ini memiliki kedalaman sampai dengan 48 layer dan dapat mengklasifikasikan gambar sampai 1000 kategori objek seperti *keyboard, mouse,* pensil, dan lain-lainnya. Alhasil, model ini belajar berbagai *feature* yang sangat banyak untuk berbagai objek gambar. *Inception v3* biasanya menggunakan ukuran input gambar 299-by-299

2.6 Metode Transfer Learning dengan menggunakan VGG16

VGG merupakan singkatan dari Visual Geometry Group ynag dikembangkan oleh Oxford. VGG merupakan salah satu model yang menyediakan struktur dan *weights* yang sudah dilatih secara online. VGG juga memiliki 160 juta parameter yang sudah dilatih. Parameter ini sebagian besar digunakan pada *fully connected layers.* Parameter ini dapat dikategorikan cukup banyak dibandingkan dengan model yang lainnya. VGG memiliki 16 *weight layers* yang terdiri dari 13 *convolutional layers* dengan ukuran filter 3x3 dan 3 *fully connected layers* dengan ukuran filter 3x3.

2.7 Metode Support Vector Machine

Support Vector Machine (SVM) pertama kali diperkenalkan oleh Vapnik pada tahun 1992 sebagai rangkaian harmonis konsep-konsep unggulan dalam bidang pattern

recognition. Sebagai salah satu metode pattern recognition, usia SVM terbilang masih relatif muda. Walaupun demikian, evaluasi kemampuannya dalam berbagai aplikasinya menempatkannya sebagai state of the art dalam pattern recognition, dan dewasa ini merupakan salah satu tema yang berkembang dengan pesat. SVM adalah metode learning machine yang bekerja atas prinsip Structural Risk Minimization (SRM) dengan tujuan menemukan hyperplane terbaik yang memisahkan dua buah class pada input space.

BAB 3

METODOLOGI

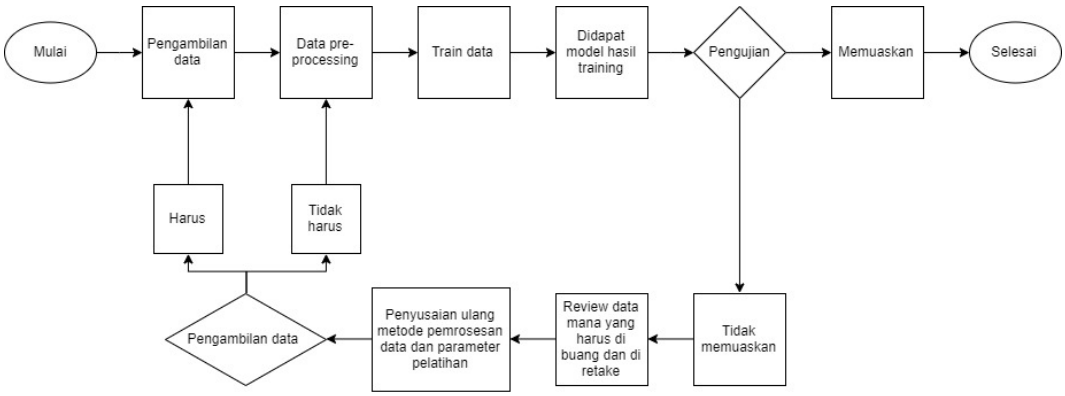
3.1 Garis Besar Aplikasi dan Pengumpulan Data

Proses pertama dalam pembuatan aplikasi “SITU Ganesha” adalah proses pengambilan data. Data yang diambil berupa foto yang berjumlah ratusan bahkan dapat mencapai ribuan foto sebagai data latih program yang akan dibuat. Proses pengambilan foto harus memperhatikan beberapa aspek agar dapat menjadi data latih yang baik. Aspek-aspek tersebut adalah :

1. Arah pengambilan. Arah pengambilan foto dapat didefinisikan sebagai sudut pengambilan kamera. Berdasarkan pengalaman di lapangan, pada setiap titik lebih baik dilakukan pengambilan sampai dengan tiga kali. Setiap pengambilan di berikan selisih sampai dengan 30 derajat.
2. Jarak pengambilan (luas gedung yang tertangkap). Lebih baik arah pengambilannya di cari yang sejauh mungkin. Alasan utamanya adalah pengambilan data pada arah yang dekat dapat di atasi dengan melakukan *zoom in* pada gambar yang di ambil dari arah jauh (sejauh mungkin bukan berarti sampai tidak terlihat).
3. Waktu pengambilan (pagi / siang / sore). Pengambilan data harus dilakukan pada tiga waktu ini. Setiap waktu memiliki intensitas penerangan yang berbeda yang nantikan akan mempengaruhi nilai setiap warnanya.
4. Cuaca dibatasi sedang tidak hujan, namun mungkin saja cerah / mendung. Pengambilan saat hujan akan memberikan bias pada data karena titik-titik air hujan akan tertangkap juga.
5. Pengambilan gambar dengan mode landscape. Alasan utama menggunakan mode landscape adalah agar data yang diambil dapat tertangkap lebih banyak dibandingkan dengan mode portrait.
6. Pengambilan gambar gedung jangan sampai terhalang terlalu banyak objek.
7. Pengambilan gambar jangan dilakukan pada malam hari.

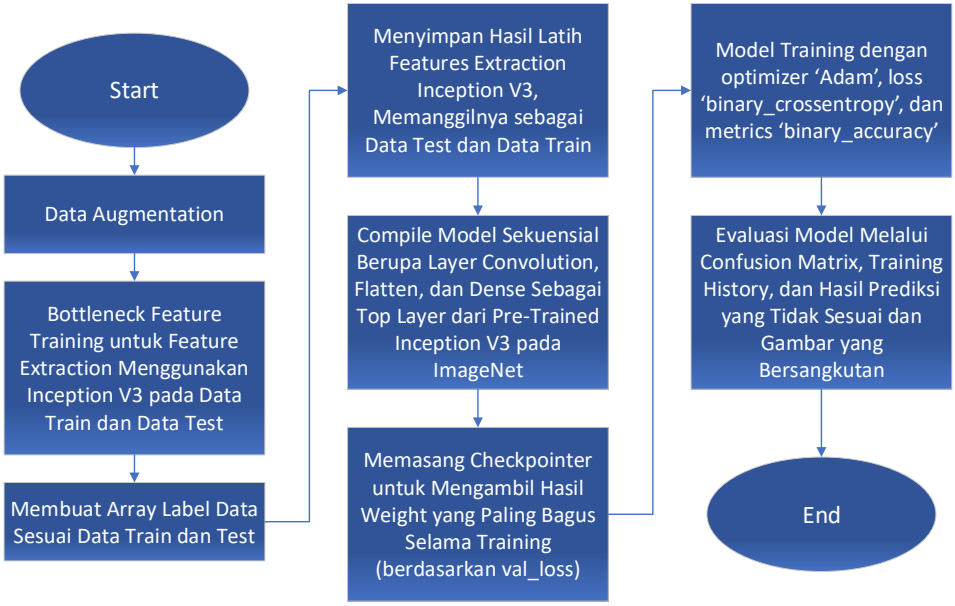
Selanjutnya proses data pre-processing. Proses ini bertujuan untuk mendapatkan data yang sesuai dengan kebutuhan untuk di olah dilangkah selanjutnya. Misalnya data yang harus dimasukkan adalah matriks 299 x 299, tetapi data yang diperoleh saat melakukan pengambilan foto adalah 2000 x 2000. Data *pre-processing* melakukan *resize* agar data dapat masuk ke dalam matriks yang dikehendaki. Data yang sudah siap untuk di olah akan

dilakukan *training* untuk mendapatkan model yang sesuai dengan yang diharapkan. Untuk mengetahui hasil sesuai harapan atau tidak, data harus dilakukan pengujian agar. Pengujian akan memberikan hasil berupa besar akurasi yang dibaca oleh model yang sudah dilatih. Biasanya, model yang memberikan hasil memuaskan akan menampilkan persen akurasi sebesar 90% - 99%. Jika hasil yang diberikan saat pengujian tidak sesuai harapan, maka harus dilakukan penulusuran ulang terkait data yang dimiliki. Apakah data yang digunakan untuk pelatihan sudah benar atau tidak. Jika belum benar, maka data harus di sortir kembali mana data yang tidak dapat digunakan untuk *training*. Setelah data disortir, data akan ditentukan lagi parameter dan metode pemrosesannya. Hasil nya akan menentukan apakah data harus dilakukan pengambilan ulang atau tidak. Siklus ini akan terus berlanjut sampai didapati tingkat akurasi yang diinginkan.



Gambar 2.7 Flowchart pembuatan aplikasi

3.2 *Transfer Learning* dengan menggunakan *Inception V3*



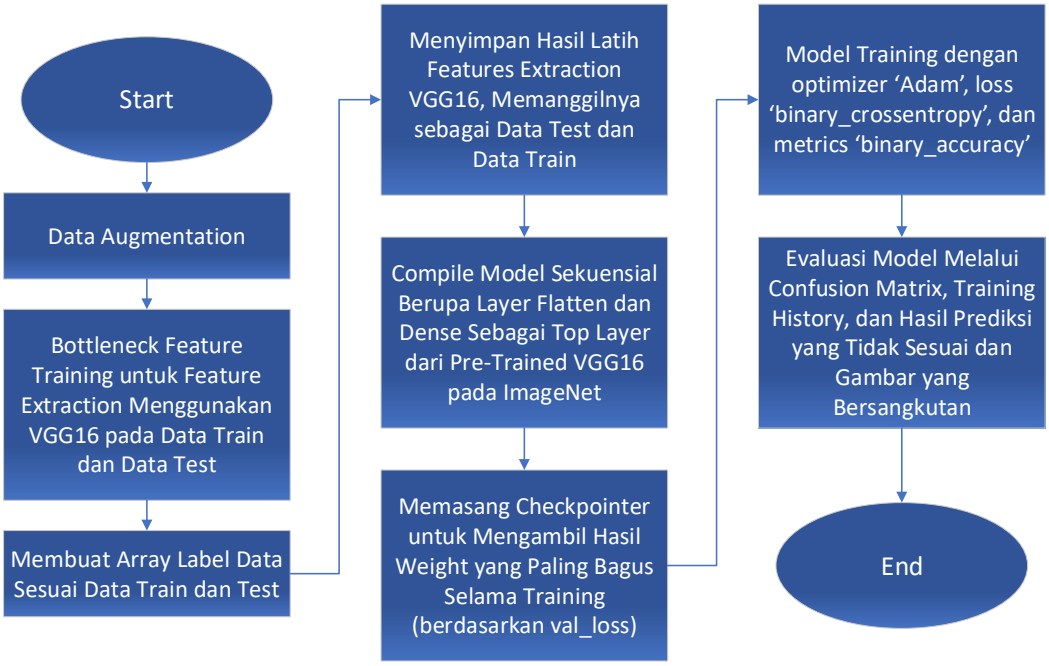
Gambar 3.1 *Flowchart* metodologi *Transfer Learning* dengan menggunakan *Inception V3*

Kode sumber asli dimana metode ini dipelajari tertera pada referensi (terlampir di daftar pustaka). Pada sumber, kode menggunakan metode transfer learning pada layer telah dilatih sebelumnya yaitu Inception V3 dengan weights dari hasil latih ImageNet. Secara umum, algoritma dalam program yang dibuat adalah pada gambar diatas. Program mengambil data dari suatu subdirektori dalam folder dimana program diletakkan, lalu melakukan augmentasi pada data karena jumlah data yang kurang banyak. Augmentasi data adalah proses penambahan jumlah dan keberagaman data latih dengan melakukan transformasi umum seperti penambahan noise, rotasi, scaling, zooming, dan flip.

Selanjutnya dilakukan bottleneck feature training dimana bottleneck feature adalah feature yang dihasilkan oleh Inception V3 sebagai input untuk training model sekuensial yang selanjutnya di buat berupa layer convolution, flatten, dan dense dengan luaran sigmoid untuk klasifikasi biner. Pelatihan layer tersebut dilakukan dengan memasang checkpointer untuk menyimpan nilai weights pada layer convolution, flatten, dan dense yang telah dibuat (perhatikan bahwa pelatihan dilakukan hanya pada layer yang baru dibuat untuk menghemat waktu dan biaya komputasi, memanfaatkan feature extraction yang dilakukan oleh Inception V3). Kemudian model dilatih dengan 50 epoch dan batch_size 32 serta dilakukan pencatatan proses pelatihan. Setelah pelatihan, model dievaluasi secara manual dengan memasukkan data seluruh data uji, melakukan prediksi menggunakan model, dan memetakannya dalam confusion matrix. Evaluasi juga dilakukan dengan menampilkan foto dimana model salah memprediksi dan mencari kelemahan model. Training history juga memberikan tambahan pengetahuan tentang proses pelatihan dan tingkat fitness apakah overfit atau underfit. Dari proses ini didapatkan weights optimal dalam model yang dapat digunakan untuk melakukan klasifikasi atas data asing.

Pada implementasi metode berbasis transfer learning dengan menggunakan Inception V3, kami melakukan beberapa modifikasi. Modifikasi tambahan kami lakukan pada augmentasi data, kami menambahkan beberapa metode augmentasi karena mungkin didapatkan dalam kehidupan nyata dan kebutuhan data latih yang lebih banyak. Kami juga menambahkan berbagai macam kemungkinan untuk pengecekan hasil prediksi dan secara khusus menambahkan prediksi terhadap error akibat kemiripan visual TVST dengan Oktagon sehingga memudahkan evaluasi terhadap false positive TVST. Kami juga berusaha menambahkan fine tuning dengan membuka block layer terakhir Inception V3, namun karena keterbatasan waktu dan kekuatan komputasi, hal tersebut tidak memungkinkan setelah dicoba dilakukan.

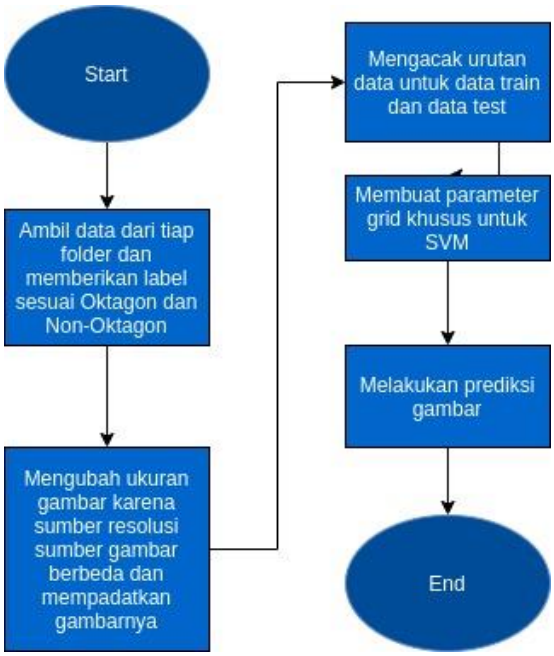
3.3 Metodologi *Transfer Learning* dengan menggunakan VGG16



Gambar 3.2 Metodologi *Transfer Learning* dengan menggunakan VGG16

Kode sumber asli tertera pada referensi. Pada sumber, kode menggunakan transfer learning yang sama seperti pada metode sebelumnya, namun kali ini menggunakan VGG16 dan top layer yang berbeda yaitu hanya flatten dan dense yang dilatih.

3.4 Metodologi *Support Vector Machine*



Gambar 3.3 Metodologi *Support Vector Machine*

Pada metode *support vector machine*, modifikasi yang dilakukan adalah mengimport data dari *google drive*, melakukan penyimpanan model dengan *pickle*, dan membuat classify.py untuk load model dan prediksi

BAB 4

HASIL DAN ANALISA

4.1 Prosedur Pengujian

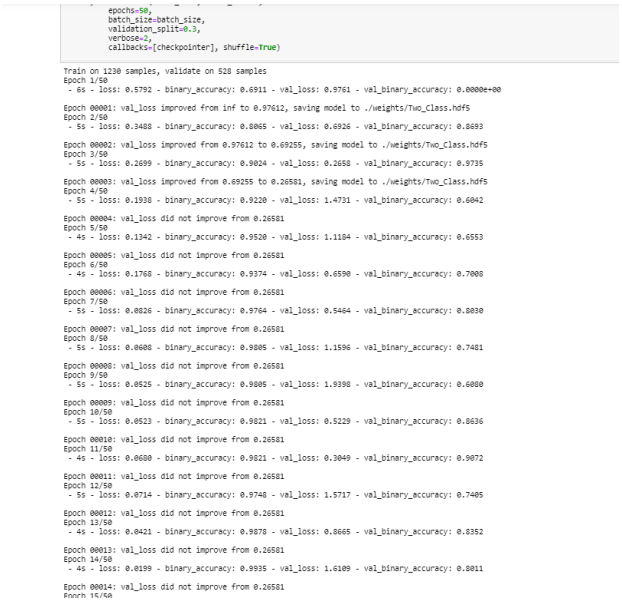
Secara garis besar, pengujian dilakukan dengan 2 tahap, yaitu melatih mesin untuk dapat belajar mengenali fitur-fitur dari bangunan yang ingin dipelajari dan menguji hasil pembelajaran mesin menggunakan 1 foto bebas. Pelatihan mesin dilakukan dengan menggunakan sebuah file *train.py* (yang berisikan kode-kode untuk melatih mesin mengenali fitur) dan diuji menggunakan program *classify.py* (yang berisikan kode-kode untuk mengambil 1 foto bebas tersebut kemudian ditentukan apakah bangunan tersebut merupakan bangunan yang sesuai dengan yang dipelajari oleh mesin atau tidak serta mengukur tingkat kesesuaian gambar yang diuji dengan hasil latihan mesin).

Pengujian dilakukan dengan berbagai jenis gambar. Pertama, pengujian dilakukan dengan foto gedung yang merupakan hasil pembelajaran mesin (dalam hal ini, Oktagon). Pengujian dilakukan untuk melihat apakah mesin mengklasifikasi gambar tersebut sebagai gambar gedung Oktagon atau tidak serta akurasi/ketepatan foto dengan hasil pembelajaran. Kedua, pengujian dilakukan dengan foto gedung yang bukan merupakan hasil pembelajarn mesin (dalam hal ini, selain gedung Oktagon). Pengujian dilakukan untuk melihat apakah mesin mengklasifikasi gambar tersebut sebagai bukan gambar gedung Oktagon serta menunjukkan alasannya dengan memperlihatkan akurasinya. Pengujian juga bisa dilakukan dengan menggunakan gambar objek apapun (dan dianggap sebagai pengujian ketiga). Tujuannya adalah untuk mengecek apakah mesin telah melakukan pembelajaran dengan baik dan tidak mengalami *overfitting* dalam proses pembelajarannya. Hasil yang diharapkan dari pengujian ketiga ini tentunya tidak terdeteksi sebagai gambar gedung Oktagon (akurasi diharapkan bisa dibawah 5%).

4.2 Hasil Pengujian

4.2.1 Pengujian *Transfer Learning* dengan Inception V3

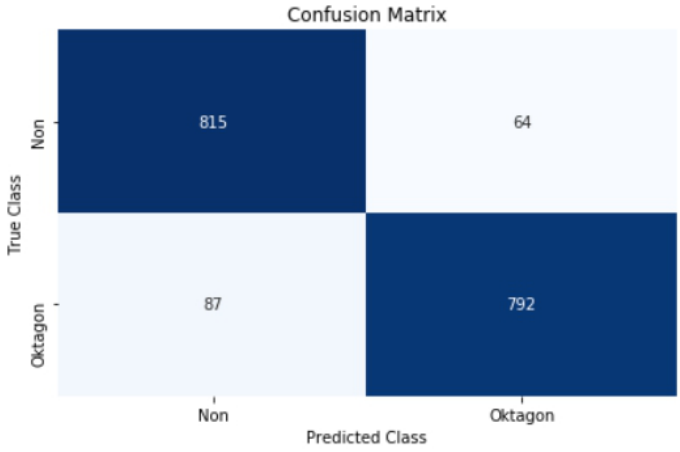
Pengujian dilakukan pada data *random* yang berasal dari kumpulan data uji dan data latih yang terdapat di kumpulan file hasil pengambilan data mahasiswa S1 dan S2 yang menjadi peserta kuliah TF4063 Sains Data Rekayasa sejumlah 1758 data uji yang berisi 879 data bangunan bukan oktagon dan 879 data bangunan oktagon.



```
        epochs=50,
        batch_size=batch_size,
        validation_split=0.3,
        verbose=2,
        callbacks=[checkpointer], shuffle=True)

Train on 1230 samples, validate on 528 samples
Epoch 1/50
 - 6s - loss: 0.5792 - binary_accuracy: 0.6911 - val_loss: 0.9761 - val_binary_accuracy: 0.0000e+00

Epoch 00001: val_loss improved from inf to 0.97612, saving model to ./weights/Two_Class.hdf5
Epoch 2/50
 - 5s - loss: 0.3488 - binary_accuracy: 0.8065 - val_loss: 0.6926 - val_binary_accuracy: 0.8693

Epoch 00002: val_loss improved from 0.97612 to 0.69255, saving model to ./weights/Two_Class.hdf5
Epoch 3/50
 - 5s - loss: 0.2699 - binary_accuracy: 0.9024 - val_loss: 0.2658 - val_binary_accuracy: 0.9735

Epoch 00003: val_loss improved from 0.69255 to 0.26581, saving model to ./weights/Two_Class.hdf5
Epoch 4/50
 - 5s - loss: 0.1938 - binary_accuracy: 0.9220 - val_loss: 1.4731 - val_binary_accuracy: 0.6042

Epoch 00004: val_loss did not improve from 0.26581
Epoch 5/50
 - 4s - loss: 0.1342 - binary_accuracy: 0.9520 - val_loss: 1.1184 - val_binary_accuracy: 0.6553

Epoch 00005: val_loss did not improve from 0.26581
Epoch 6/50
 - 4s - loss: 0.1768 - binary_accuracy: 0.9374 - val_loss: 0.6590 - val_binary_accuracy: 0.7008

Epoch 00006: val_loss did not improve from 0.26581
Epoch 7/50
 - 5s - loss: 0.0826 - binary_accuracy: 0.9764 - val_loss: 0.5464 - val_binary_accuracy: 0.8030

Epoch 00007: val_loss did not improve from 0.26581
Epoch 8/50
 - 5s - loss: 0.0608 - binary_accuracy: 0.9805 - val_loss: 1.1596 - val_binary_accuracy: 0.7481

Epoch 00008: val_loss did not improve from 0.26581
Epoch 9/50
 - 5s - loss: 0.0525 - binary_accuracy: 0.9805 - val_loss: 1.9398 - val_binary_accuracy: 0.6080

Epoch 00009: val_loss did not improve from 0.26581
Epoch 10/50
 - 5s - loss: 0.0523 - binary_accuracy: 0.9821 - val_loss: 0.5229 - val_binary_accuracy: 0.8636

Epoch 00010: val_loss did not improve from 0.26581
Epoch 11/50
 - 4s - loss: 0.0600 - binary_accuracy: 0.9821 - val_loss: 0.3049 - val_binary_accuracy: 0.9072

Epoch 00011: val_loss did not improve from 0.26581
Epoch 12/50
 - 5s - loss: 0.0714 - binary_accuracy: 0.9748 - val_loss: 1.5717 - val_binary_accuracy: 0.7405

Epoch 00012: val_loss did not improve from 0.26581
Epoch 13/50
 - 4s - loss: 0.0421 - binary_accuracy: 0.9878 - val_loss: 0.8665 - val_binary_accuracy: 0.8352

Epoch 00013: val_loss did not improve from 0.26581
Epoch 14/50
 - 4s - loss: 0.0199 - binary_accuracy: 0.9935 - val_loss: 1.6109 - val_binary_accuracy: 0.8011

Epoch 00014: val_loss did not improve from 0.26581
Epoch 15/50
```

Gambar 4.1 Proses Pengujian dalam Pelatihan Model *Transfer Learning* InceptionV3

Selain penggunaan pengujian saat pelatihan, model diuji kembali dengan data random yang sama untuk mendefinisikan bukan hanya sebagian seperti pada saat pelatihan, namun seluruh data uji. Hasilnya adalah sebagai berikut

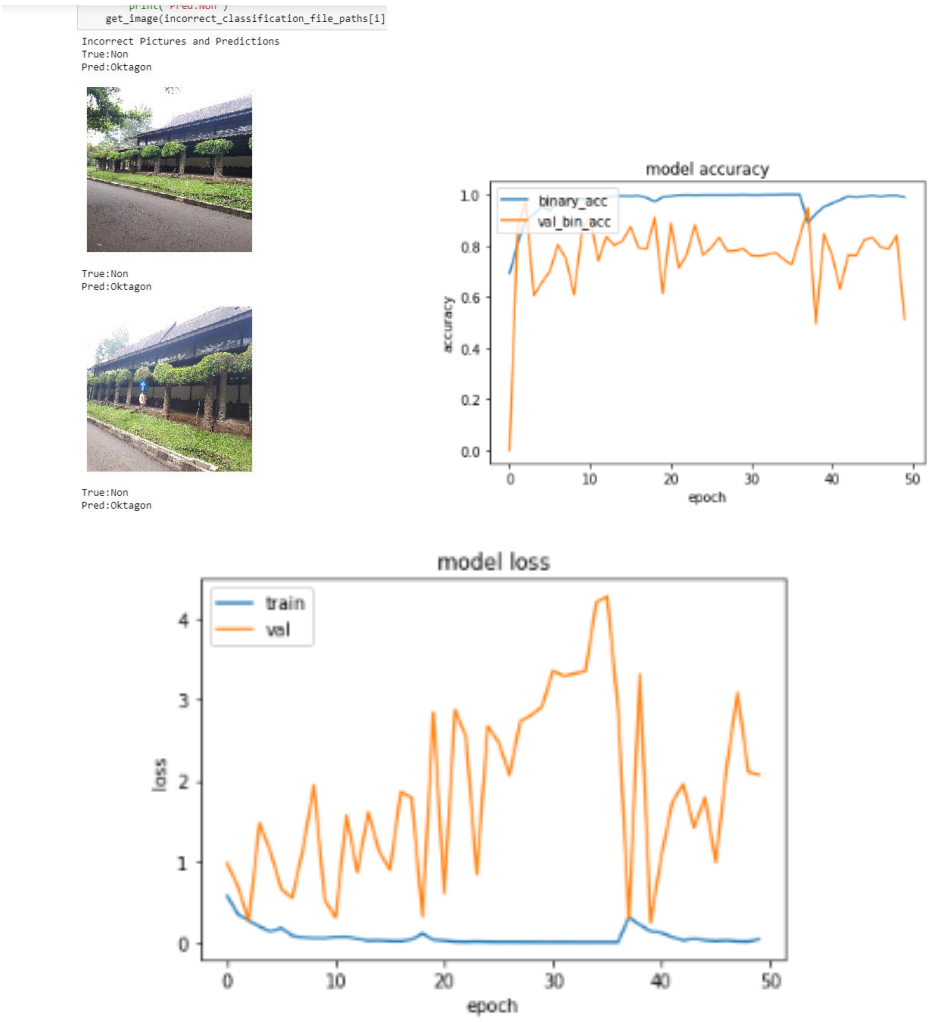


Gambar 4.2 Hasil Pengujian False Positive dan False Negative Berupa Confusion Matrix Model *Transfer Learning* InceptionV3

Dapat diamati bahwa model memiliki total akurasi tervalidasi 90.7%. Namun saat dilakukan pengulangan komputasi fitur pada data uji menggunakan InceptionV3, model dapat mengalami penurunan maupun peningkatan akurasi dengan rata-rata 90% dan span 88%-92%.

Hasil pengujian ketiga adalah ketika dihadapkan dengan foto satu gambar. Model dapat mengulangi dengan konsisten hasil pembelajarannya (namun juga salah di tempat yang sama dimana dia salah pada saat pembelajaran) meskipun dilakukan *feature extraction* setiap satu gambar baru ingin ditentukan. Waktu penentuan relatif cukup lama dibandingkan langsung tanpa ekstraksi fitur.

Hasil pengujian yang tidak kalah penting adalah bagaimana model divalidasi aman dan tidak *overfitting*, pertama adalah pengujian dengan mengamati perubahan besar *loss* validasi dan *binary accuracy*, berikut hasilnya:



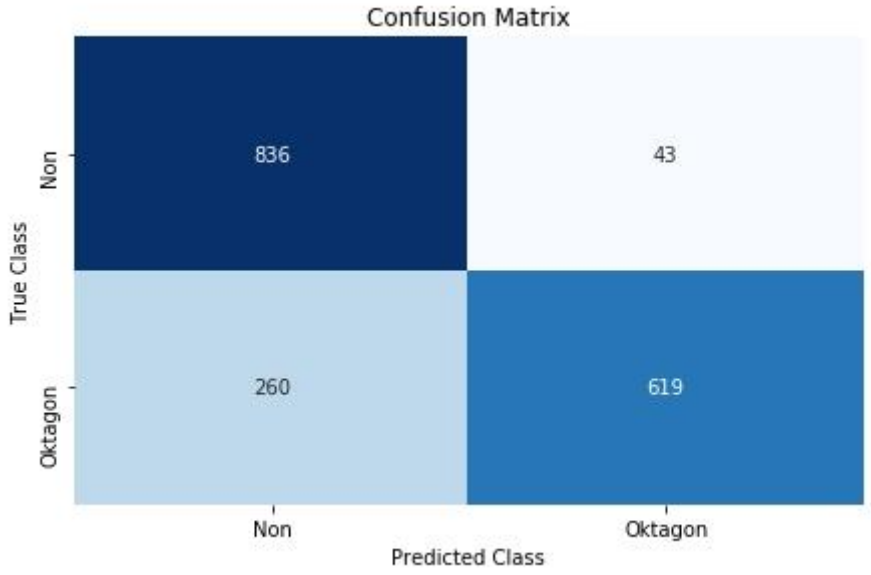
Gambar 4.3 Hasil perubahan akurasi dan *loss* serta penampilan kegagalan model

4.2.2 Pengujian *Transfer Learning* dengan VVG16

Pengujian dilakukan pada data *random* yang berasal dari kumpulan data uji dan data latih yang terdapat di kumpulan file hasil pengambilan data mahasiswa S1 dan S2 yang

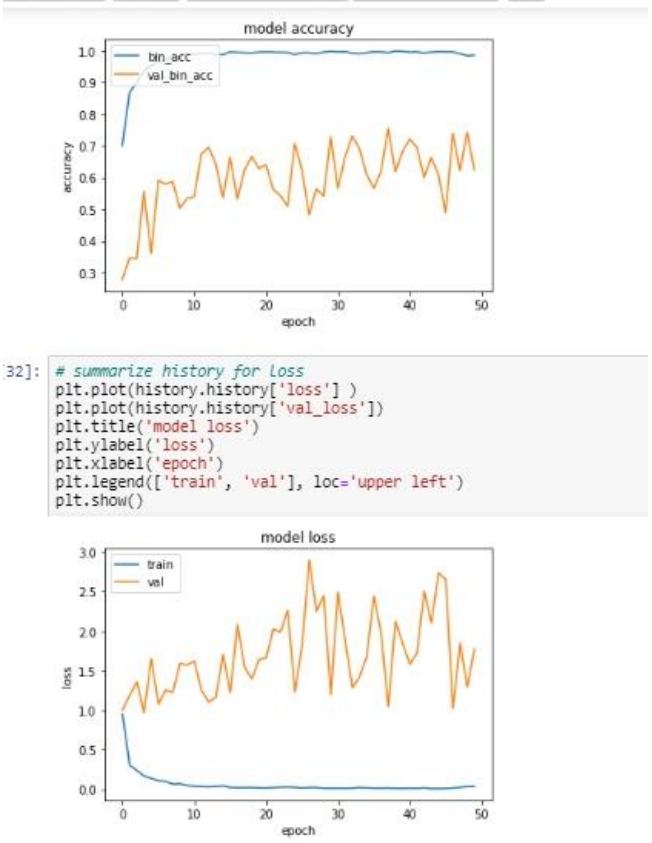
menjadi peserta kuliah TF4063 Sains Data Rekayasa sejumlah 1758 data uji yang berisi 879 data bangunan bukan oktagon dan 879 data bangunan oktagon.

Selain penggunaan pengujian saat pelatihan, model diuji kembali dengan data random yang sama untuk mendefinisikan bukan hanya sebagian seperti pada saat pelatihan, namun seluruh data uji. Hasilnya adalah sebagai berikut



Gambar 4.4 Hasil pengujian *false positive* dan *false negative* berupa *confusion matrix model transfer learning* VVG16

Hasil pengujian memberikan nilai akurasi sebesar 82.7 % yang memiliki nilai lebih rendah dibandingkan dengan model *interception v3.* Selain itu, model juga gagal mirip dengan *interception v3* namun lebih sedikit kegagalan false positivenya.



Gambar 4.5 Model *loss* dan model *accuracy* pada model VVG16

4.2.3 Pengujian *Support Vector Machine*

Pengujian dilakukan pada data *random* yang berasal dari kumpulan data uji dan data latih yang terdapat di kumpulan file hasil pengambilan data mahasiswa S1 dan S2 yang menjadi peserta kuliah TF4063 Sains Data Rekayasa sejumlah 1758 data uji yang berisi 879 data bangunan bukan oktagon dan 879 data bangunan oktagon.

Selain penggunaan pengujian saat pelatihan, model diuji kembali dengan data random yang sama untuk mendefinisikan bukan hanya sebagian seperti pada saat pelatihan, namun seluruh data uji. Hasilnya adalah sebagai berikut

```
Classification report for -
GridSearchCV(cv='warn', error_score='raise-deprecating',
             estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='auto_deprecated', kernel='rbf', max_iter=-1,
                           probability=False, random_state=None, shrinking=True,
                           tol=0.001, verbose=False),
             iid='warn', n_jobs=None,
             param_grid=[{'C': [1, 10, 100, 1000], 'kernel': ['linear']},
                         {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001],
                          'kernel': ['rbf']}],
             pre_dispatch='2*n_jobs', refit=True, return_train score=False,
             scoring=None, verbose=0):
              precision    recall  f1-score   support

           0       0.85      0.91      0.88       264
           1       0.90      0.84      0.87       264

    accuracy                           0.87       528
   macro avg       0.87      0.87      0.87       528
weighted avg       0.87      0.87      0.87       528
```

Gambar 4.6 Hasil pengujian *support vector machine*

Akurasi yang dihasilkan bernilai 87% berdasarkan hasil pengujian dari library scikit learning. Selain itu, proses pengujian dilakukan dengan pengacakan urutan data dengan cross validation dari data latih

4.2.4 *Convolutional Neural Network* Dasar

Pertama-tama classifier diuji dengan menggunakan dataset yang dengan 733 gambar yang terbagi atas 324 gambar bangunan non Oktagon dan 409 gambar bangunan Oktagon. Dari pengujian ini diperoleh data sebagai berikut :

```
              precision    recall  f1-score   support

         Non       0.44      1.00      0.61       324
     Oktagon       0.00      0.00      0.00       409

    accuracy                           0.44       733
   macro avg       0.22      0.50      0.31       733
weighted avg       0.20      0.44      0.27       733
```

Gambar 4.7 Hasil pengujian CNN

Terlihat bahwa performa dari model ini masih sangat buruk. Precision bernilai 1 untuk Non-Oktagon yang berarti seluruh data Non octagon berhasil ditebak. Namun terlihat juga bahwa classifier ini tidak mampu sama sekali untuk mengenali octagon dari data test yang digunakan. Hal ini dapat disebabkan karena metode ini masih kurang efektif dalam mengenali fitur-fitur yang dimiliki Oktagon.

4.3 Analisa Pengujian

Hasil dari berbagai metode machine learning yang digunakan beragam. Dapat dilihat bahwa metode dengan dasar neural network seperti *transfer learning* dengan Inception V3 maupun VGG16 serta CNN dasar memiliki kekonsistenan terhadap data yang banyak meskipun secara akurasi terakadang kalah dengan metode berbasis SVM. Dari data hasil pengujian, terlihat pula kekonsistenan metode dengan dasar neural network lebih konsisten dibanding SVM dengan *loss* dan akurasi yang lebih stabil. Dapat dilihat dari data juga bahwa akurasi tertinggi dengan deviasi terendah adalah model yang dibuat dengan metode *transfer learning* berbasis Inception V3. Hal ini karena jumlah data yang relatif sedikit namun cukup banyak dapat diolah dengan baik menggunakan metode *transfer learning*. Model Inception V3 lebih baik sebagai dasar untuk pembuatan model dengan jumlah data yang sedikit tanpa terkena bias yang tinggi. Model dasar CNN mudah dilatih dan membutuhkan waktu yang cukup singkat relative terhadap Inception V3 ataupun VGG16, namun model CNN menghasilkan *false negative and false positive test* yang sangat buruk karena ketidakmampuan mengenali fitur seperti kembaran *deep learning*nya.

Dari total 1758 foto latih, model Inception V3 dapat mengenali rata-rata sebesar 90% (berdasarkan 10 pengulangan pembuatan ekstraksi fitur untuk 1758 foto latih) dengan *range* 88%-92%. Hal ini cukup konsisten mengingat dinamika ragam foto yang besar dari sisi pandang (perspektif), cuaca, waktu (pagi-siang-sore), jenis bangunan yang merupakan bukan oktagon, hingga foto lainnya seperti pohon ataupun batu yang diadakan sebagai data uji. Serta kemampuan model untuk mampu mengenali Gedung TVST yang merupakan kembaran oktagon sebanyak 56% sebagai Gedung yang bukan Oktagon, hal ini menarik karena arsitektur kedua Gedung yang sangat mirip membuat mereka sulit sekali dibedakan secara visual. Total 20% kesalahan dari model adalah Gedung TVST dengan jumlah data uji 36 dan kesalahan 16. Hal ini menunjukkan bahwa model masih bias dan mengenali fitur Oktagon yang mirip TVST sebagai acuan. Pengembangan lebih lanjut dari model ini adalah *ensemble method* dengan menggabungkan beberapa metode machine learning untuk melakukan keputusan, secara spesifik membuat beberapa neural network hasil *deep learning* dan *transfer learning* bekerja bersama. Metode lain tersebut harus dapat membedakan TVST dan Oktagon secara spesifik. Foto lain yang sulit dibedakan adalah Aula Barat, hal ini menarik mengingat fitur yang sangat sedikit kaitannya dengan Oktagon. Berdasarkan analisis kami, hal ini terjadi karena fitur Aula Barat yang jauh berbeda dari fitur non-oktagon lainnya dan kekurangan dataset tentang bagian Aula Barat yang salah diidentifikasi oleh model.

BAB 5

KESIMPULAN

Metode *machine learning* terbaik untuk klasifikasi dua kelas “Oktagon” dan “Bukan Oktagon” dalam mencapai tujuan rekognisi foto turis dalam melalui jalur tengah kampus ITB adalah *transfer learning* dengan arsitektur Inception V3 sebagai layer dasar dan *convolution, flatten,* dan *dense* sebagai *top layer*. Jumlah dataset sebanyak 1758 buah terdiri dari 879 “Bukan Oktagon” dan 879 “Oktagon” mencukupi untuk menghasilkan akurasi sebesar 90% dengan rentang akurasi 88%-92%. Metode telah menunjukkan superioritas dibandingkan dengan SVM, CNN dasar, dan *transfer learning* VGG16.

DAFTAR PUSTAKA

1. http://repositori.usu.ac.id/bitstream/handle/123456789/2412/121402064.pdf?sequence=1&isAllowed=y (diakses pada 6 Desember 2019)
2. http://asnugroho.net/papers/ikcsvm.pdf (diakses pada 6 Desember 2019)
3. https://www.quora.com/What-is-the-VGG-neural-network (diakses pada 6 Desember 2019)
4. https://towardsdatascience.com/building-the-hotdog-not-hotdog-classifier-from-hbos-silicon-valley-c0cb2317711f (diakses pada 6 Desember 2019 (referensi metode transfer learning dengan inception V3 dan VVG16))
5. https://github.com/J-Yash/Hotdog-Not Hotdog/blob/master/Hotdog_classifier_transfer_learning.ipynb (diakses pada 6 Desember 2019 (referensi metode transfer learning dengan inception V3 dan VVG16))
6. https://github.com/whimian/SVM-Image-Classification/blob/master/Image%20Classification%20using%20scikit-learn.ipynb (diakses pada 6 Desember 2019 (referensi metode support vector machine))
7. https://py2py.com/cnn-part-3-setting-up-google-colab-and-training-model-using-tensorflow-and-keras/ (diakses pada 6 Desember 2019 (referensi metode support vector machine))
8. https://machinelearningmastery.com/save-load-machine-learning-models-python-scikit-learn (diakses pada 6 Desember 2019 (referensi metode support vector machine))

LAMPIRAN

No	NIM	Nama	Peran
1	13317001	Gregory Sembiring Brahmana	***Researcher*** **Metode dan Membantu** ***Programming***
2	13317041	Bernardus Rendy	**Programmer**
3	13317043	Nicholas Biantoro	**Programmer**
4	13317069	Murphy Halim	***Researcher*** **Metode dan Laporan**
5	13317084	Putu Bhargo Abhimana Chrysnanda	**Programmer**