IMPLEMENTASI *DEEP LEARNING* UNTUK *IMAGE CLASSIFICATION* MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA WAYANG GOLEK

TUGAS AKHIR

TRIANO NURHIKMAT

14611209

PROGRAM STUDI STATISTIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2018

HALAMAN PERSETUJUAN PEMBIMBING

TUGAS AKHIR

Judul	:	Implementasi *Deep Learning* Untuk *Image Classification* Menggunakan Algoritma *Convolutional Neural Network* (CNN) Pada Citra Wayang Golek
Nama Mahasiswa	:	Triano Nurhikmat
Nomor Mahasiswa	:	14611209

TUGAS AKHIR INI TELAH DIPERIKSA DAN DISETUJUI UNTUK DIUJIKAN

Yogyakarta, 05 Mei 2018

Menyetujui,

Dosen Pembimbing

(Tuti Purwaningsih S.Stat., M.Si.)

HALAMAN PENGESAHAN

TUGAS AKHIR

IMPLEMENTASI *DEEP LEARNING* UNTUK *IMAGE CLASSIFICATION* MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA WAYANG GOLEK

Nama : Triano Nurhikmat

NIM : 14611209

TUGAS AKHIR INI TELAH DIUJIKAN

PADA TANGGAL 23 MEI 2018

Nama Penguji		**Tanda Tangan**
1 Andrie Pasca Hendradewa, S.T, M.T	:	.....
2 Muhammad Muhajir, S.Si., M.Sc	:	.....
3 Tuti Purwaningsih S.Stat., M.Si	:	.....

Mengetahui,

Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

(Drs. Allwar, M.Sc., Ph.D.)

KATA PENGANTAR

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Panyayang, Kami panjatkan puja dan puji syukur atas kehadirat-Nya, yang telah melimpahkan rahmat, hidayah, dan inayah-Nya kepada kami, sehingga kami dapat menyelesaikan makalah ilmiah tentang limbah dan manfaatnya untuk masyarakat.

Tugas Akhir ini telah saya susun dengan maksimal dan mendapatkan bantuan dari berbagai pihak sehingga dapat memperlancar pembuatan makalah ini. Oleh karena itu, dalam kesempatan ini penulis menyampaikan ucapan terima kasih kepada :

1. Nandang Sutrisno, SH., LL.M., M.Hum., Ph.D Sebagai Rektor Universitas Islam Indonesia
2. Bapak Drs. Allwar, M.Sc., Ph.D. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia.
3. Bapak Dr. RB. Fajriya Hakim, S.Si., M.Si. selaku Ketua Jurusan Statistika beserta jajarannya.
4. Tuti Purwaningsih S.Stat., M.Si., selaku dosen Pembimbing yang sudah membimbing dan memberikan dukungan kepada saya dari awal sampai akhir ini.
5. Seluruh staff pengajar Program Studi Statistika Universitas Islam Indonesia yang telah memberikan bekal ilmu kepada saya.
6. Orang tua beserta keluarga besar saya atas kasih sayang, dukungan dan doanya sehingga dapat menyelesaikan Tugas Akhir ini dengan baik.
7. Keluarga besar IKS (Ikatan Keluarga Statistika), sebagai Organisasi yang membawahi mahasiswa statistika FMIPA UII, terimakasih atas kebersamaan, kekeluargaan, kekompakan, keceriaan dan pelajaran berharga lainnya selama tiga tahun ini.

8. Mantan Inti Kadep IKS FMIPA UII 2016/2017, Anggi Prabaningrum, Hafizah Ilma, Suci Nurul insani, Rachmad Febrian, Syauqi Amri Yahya, Yusnandar, Irsyad Muhammad Firdaus, Muhammad Farhan Abdul F, yang sudah membantu saya selama di IKS selama satu periode.
9. Teman-teman bimbingan TA yang sudah sama-sama berjuang, saling mengingatkan dan memberi motivasi serta dorongan untuk menyelasaikan Tugas Akhir ini
10. Teman-teman seperjuangan yaitu Jimmy, Tiara, Sendhy, dan Hafizhan yang telah membantu saya dalam proses penelitian.
11. Teman-teman Statistika UII Angkatan 2014 yang bersama-sama menjadi pejuang gelar S.Stat dan Toga UII, terimakasih semangatnya
12. Serta semua pihak lainya yang tidak bisa dituliskan penulis satu per satu yang telah membantu selama pembuatan Tugas Akhir ini.

Demikian Tugas Akhir ini, penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan bantuan baik moril maupun materil sehingga tugas akhir ini dapat diselesaikan. Penulis menyadari bahwa tugas akhir ini masih jauh dari kata sempurna dan masih banyak kekurangan. Hal tersebut dikarenakan keterbatasan ilmu dan pengetahuan yang dimiliki penulis semata. Oleh karena itu penulis mengharapkan kritik dan saran dari pembaca untuk menyempurnakan penulisan tugas akhir ini. Semoga Tugas Akhir ini dapat memberikan manfaat bagi penulis khususnya dan umumnya bagi semua pihak yang membutuhkan. Akhir kata, semoga Allah SWT senantiasa melimpahkan rahmat serta hidayah-Nya kepada kita semua, Amin amin ya robbal ‘alamiin.

Yogyakarta, Mei 2018

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR ISTILAH

Batch Size	:	Jumlah sampel data yang disebarkan ke Neural network atau ukuran dari satuan kecil *Epoch* yang dimasukkan ke dalam *computer*
Class/Label	:	Variable atau atribut yang digunakan dalam penelitian
Stride	:	Parameter yang digunakan untuk menentukan jumlah pergeseran filter/kernel.
Convolution	:	Proses dimana perhitungan *dot product* nilai matriks dari image dengan nilai matriks dari kernel/filternya.
Pooling	:	Proses Mengurangi dimensi dari feature map (downsampling).
Kernel/Filter	:	Matriks untuk menghitung dan mendeteksi suatu pola yang digunakan pada saat proses *convolution*
Step	:	Sejumlah langkah yang mendefinisikan pada konfigurasi *pipline* untuk proses pelatihan yang menentukan tingkat keberhasilan pelatihan *Neural Networks*.
Padding	:	Parameter jumlah piksel yang berisi nilai nol yang ditambahkan disetiap sisi input.
Dropout	:	Teknik regulasi jaringan sarag dimana beberapa neuron akan dipilih secara acak dan tidak dipakai selama proses pelatihan.
Epoch	:	Ketika seluruh dataset sudah melalui proses pelatihan pada *Neural Network* sampai dikembalikan keawal untuk sekali putaran
Learning Rate	:	Prameter dari *Gradient Descent*
Loss Function	:	Nilai Kerugian yang diperoleh pada proses pelatihan
Iterations	:	Jumlah *batch* yang diperlukan untuk menyelesaikan satu *Epoch*

Gradient Descent	:	Algoritma untuk mengoptimalkan iterasi yang digunakan pada *Machine Learning* untuk menemukan hasil yang terbaik

DAFTAR LAMPIRAN

PERNYATAAN

Dengan ini kami menyatakan bahwa dalam Tugas Akhir ini tidak terdapat karya yang sebelumnya pernah diajukan untuk tugas akhir. Tugas akhir ini diajukan untuk memperoleh gelar sarjana di suatu Perguruan Tinggi dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 05 Mei 2018

Penulis

IMPLEMENTASI *DEEP LEARNING* UNTUK *IMAGE CLASSIFICATION* MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (CNN) PADA CITRA WAYANG GOLEK

Triano Nurhikmat
Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Islam Indonesia

INTISARI

Indonesia merupakan bangsa yang terdiri dari berbagai etnik dan memiliki latar belakang budaya yang beraneka ragam. Salah satu hasil kebudayaan masyarakat Indonesia adalah Wayang. Wayang merupakan seni tradisional yang bekembang di indonesia terutama di pulau Jawa dan Bali. Di dunia internasional wayang kini telah tercatat sebagai karya seni budaya adiluhung, yaitu oleh UNESCO, sebuah lembaga di bawah PBB yang menangani masalah pendidikan, ilmu pengetahuan, dan kebudayaan. Melihat penghargaan tersebut sudah seharusnya masyarakat Indonesia menjaga dan melestarikannya. Akan tetapi, diera sekarang ini dunia teknologi sudah semakin berkembang, sehingga banyak masyarakat yang melupakan akan kebudayaan tradisional ini, tertuama dikalangan remaja. Hasil survey berdasarkan citra digital toko-tokoh pewayangan menunjukan sebanyak 71 % dari 60 orang tidak mengenalinya. Ini bertujuan untuk membatu mengklasifikasi objek tokoh-tokoh pewayangan berdasarkan citra digital. Sehingga, dibutuhkan suatu pendekatan dalam penyelesaian permasalan ini. Salah satu pendekatan dalam pengenalan suatu gambar adalah menggunakan metode Convolutional Neural Network. Metode ini salah satu metode Deep learning yang dapat digunakan untuk mengenali dan mengklasifikasi sebuah objek pada sebuah citra digital. Berdasarkan hasil pembahasan didapatkan tingkat akurasi sebesar 95% pada proses training dan 90 % pada proses testing. Kemudian penelitian ini menggunakan data baru untuk menguji model yang telah dibuat. Tingkat akurasi yang dihasilkan menggunakan data baru sebesar 93 % dalam mengklasifikasikan gambar wayang golek. Sehingga, performa dari model yang dibuat pada penelitian ini dapat dikatakan optimal dalam mengklasifikasikan gambar wayang golek.

Kata Kunci : *Deep Learning, Image Classification, Wayang Golek*

IMPLEMENTATION OF DEEP LEARNING FOR IMAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK ALGORITHM ON THE IMAGE WAYANG GOLEK

Triano Nurhikmat
Department of Statictics, Faculty of Mathematics and Natural Sciences
Islamic University of Indonesia

ABSTRACT

Indonesia is a nation that consists of various ethnic and has diverse cultural background. One of the cultural results of Indonesian society is Wayang. Wayang is a traditional art that develops in Indonesia, especially in Java and Bali. In the international world of wayang has now been recorded as a masterpiece of cultural art of keduhung, namely by UNESCO, an institution under the United Nations dealing with the problems of education, science, and culture. Seeing the award should be the people of Indonesia to maintain and preserve it. However, nowadays the world of technology has been growing, so many people who forget about this traditional culture, tertuama among teenagers. The survey results based on the digital image of the shop-figure puppet show as many as 71% of 60 people do not recognize it. It aims to help to classify objects of wayang figures based on digital imagery. Thus, an approach is needed in the settlement of this problem. One approach in the introduction of an image is to use the method of Convolutional Neural Network. This method is one method Deep learning that can be used to recognize and classify an object on a digital image. Based on the results obtained an accuracy of 95% on the training process and 90% in the testing process. Then this research uses new data to test the model that has been made. The resulting accuracy rate using new data is 93% in classifying wayang golek images. Thus, the performance of the model made in this study can be said to be optimal in classifying images of wayang golek.

Keyword : *Deep Learning, Image Classification, Wayang Golek*

BAB 1
PENDAHULUAN

1.1. Latar Belakang

Indonesia merupakan bangsa yang terdiri dari berbagai etnik dan memiliki latar belakang budaya yang beraneka ragam. Budaya adalah hasil budi dan daya yang berupa cipta, karsa, dan rasa yang didalamnya mengandung kebiasaan manusia sebagai anggota masyarakat. Menurut Bronislow Malinowsky dalam buku M. Munandar Sulaeman, kebudayaan di dunia memiliki tujuh unsur universal, yaitu bahasa, religi, sistem pengetahuan, sistem mata pencaharian, organisasi sosial, sistem teknologi dan kesenian (Sulaeman, 1998). Salah satu unsur dari kebudayaan yang menarik dimata masyarakat adalah kesenian. Hal ini dikarekan kesenian memiliki bobot besar dalam kebudayaan, kesenian sarat dengan kandungan nilai-nilai budaya, bahkan menjadi wujud dan ekspresi yang menonjol dari nilai-nilai budaya.

Salah satu hasil kebudayaan masyarakat Indonesia adalah Wayang. Wayang merupakan seni tradisional yang bekembang di indonesia terutama di pulau Jawa dan Bali. Menurut para ahli wayang dikenal oleh bangsa Indonesia sejak tahun 1500 SM karena nenek moyang percaya bahwa setiap benda mati mempunyai roh yang baik dan jahat,agar tidak diganggu oleh roh jahat maka roh-roh tersebut dilukis dalam bentuk gambaran atau bayangan (wewayangan atau wayang) dan disembah serta diberi sesajen kepercayaan ini dikenal dengan animisme, kepercayaan ini berlangsung lama namun setelah kedatangan agama Hindu maka gambaran roh berubah fungsinya menjadi alat peraga untuk menyampaikan ajaran-ajaran agama dan kini menjadi tontonan serta tuntunan (Pasha, 2011). Terdapat 2 versi jenis wayang yaitu wayang orang yang di mainkan langsung oleh beberapa orang dan wayang yang berwujud boneka yang dimainkan oleh dalang. Salah satu wayang berwujud boneka adalah Wayang Golek.

Wayang golek merupakan suatu seni pertunjukan wayang yang terbuat dari boneka kayu yang berasal dari Jawa Barat. Wayang golek pada umumnya

kebanyakan ceritanya diambil dari cerita *Ramayana* dan *Mahabarata* dengan menggunakan bahasa Sunda. Pertunjukan Wayang golek memiliki nilai-nilai kebajikan dan falsafah hidup seperti sebagai media pendidikan, media dakwah islamiyah. Sebagai sebuah seni kreatif bermutu tinggi, wayang tidak hanya sekedar tontonan hiburan, tetapi juga sebagai tuntunan hidup yang memberikan pelajaran untuk memahami alam semesta dan sekaligus sebagai kerangka acuan untuk menyeimbangkan ekspresi moral, seni religiusitas. Menurut Sedyatmanto dalam (Effendi, 2009), wayang berguna tidak hanya sebagai pertunjukan dan hiburan, tetapi juga untuk membentuk watak dan karakter. Pertunjukan wayang juga menampilkan tokoh-tokoh wayang dan menunjukan bagaiamana setiap peran itu harus dijalankan. Contohnya tokoh wayang punakawan yang terdiri dari semar, gareng, petruk, dan bagong memiliki watak yang berbeda-beda. Setiap tokoh wayang memiliki karakter yang jelas dan dapat diketahui sikap dan tindakan mana yang dapat diharapkan dari tokohtokoh tersebut (Suseno, 1991).

Di dunia internasional wayang kini telah tercatat sebagai karya seni budaya *adiluhung*, yaitu oleh UNESCO, sebuah lembaga di bawah PBB yang menangani masalah pendidikan, ilmu pengetahuan, dan kebudayaan. Pada tanggal 7 November 2003 wayang Indonesia diumumkan oleh UNESCO sebagai karya agung dunia di Paris. Hal ini menunjukan bahwa wayang sebagai salah satu warisan budaya tradisional, telah diakui dunia internasional sebagai sebagai sebuah warisan budaya sarat nilai yang berperan besar dalam pembentukan dan pengembangan jatidiri bangsa. Sebagaimana dikemukakan oleh direktur UNESCO 2004 (Koitchiro Matsuura), karena wayang telah diakui sebagai salah satu warisan budaya dunia, ia harus dilestarikan dan itu menjadi tugas seluruh bangsa, terutama bangsa Indonesia yang memiliki produk yang sedemikian luhur ini. Jadi, bangsa Indonesia kini memiliki tugas berat untuk menyelamatkan dan melestarikan produk budayanya ini (Sudarwo, Sumari, Undung Wijaya, 2010).

Melihat penghargaan tersebut, sebagai warga negara yang yang berintegritas sudah seharusnya menjaga dan melestarikan kebudayaan ini. Akan tetapi, seiring dengan perkembangan zaman dengan kemajuan teknologi yang semakin canggih, pertunjukan wayang golek yang dulu menjadi bagian dari sarana hiburan

masyarakat kini semakin tersisih dari percaturan dunia panggung hiburan. Teknologi seperti, televisi, *handphone* VCD, DVD, komputer dan sebagainya, memudahkan masyarakat untuk mendapatkan sarana hiburan yang lebih moderenisasi. Sehingga masyarakat lebih memilih menggunakan teknologi yang sudah ada dibanding dengan hiburan-hiburan yang bersifat tradisional, khususnya dikalangan generasi muda sekarang. Pengetahuan akan kesenian wayang golek ini dikalangan remaja sudah semakin berkurang, salah satunya ketidaktahuan akan tokoh-tokoh pewayangan. hal ini di karenakan banyaknya tokoh pewayangan memiliki bentuk dan karakter yang berbeda-beda. Hasil survey berdasarkan citra digital toko-tokoh wayang berdasarkan karakternya menunjukan sebanyak 71 % dari 60 orang remaja tidak mengenalinya. Survey ini bertujuan untuk mengetahui apakah remaja saat ini masih mengenal tokoh-tokoh wayang berdasarkan karakter gambar wayang golek.

Seiring dengan kemajuan zaman, klasifikasi citra digital sangat dibutuhkan diberbagai macam bidang, seperti : informatika, kedokteran, kelautan, pertanian, dan bisnis. Beberapa penelitian yang telah dilakukan misalnya klasifikasi buku (Lukman, 2012) dan klasifikasi pada daging sapi (Budianita & Jasril, 2015). Tujuan dari klasifikasi citra adalah mengklasifikasikan masukkan citra kedalam beberapa kategori tertentu. Klasifikasi citra saat ini menjadi salah satu problem yang telah lama dicari solusinya dalam *computer vision.* Bagaimana menduplikasikan kemampuan manusia dalam memahami informasi citra digital, supaya komputer dapat mengenali objek pada citra selayaknya manusia. Proses *feature engineering* yang digunakan pada umumnya sangat terbatas dimana hanya dapat berlaku pada dataset tertentu saja tanpa kemampuan generalisasi apapun. Hal ini dikarenakan berbagai perbedaan antar citra antara lain perbedaan sudut pandang, perbedaan skala, perbedaan kondisi pencahayaan, deformasi objek, dan sebagainya.

Kalangan akademisi telah banyak bergelut dalam problem ini. Salah satu pendekatan yang berhasil digunakan dengan menggunakan Jaringan Syaraf Tiruan (*Artificial Neural Network,* ANN). ANN adalah salah satu bentuk kecerdasan buatan yang mempunyai kemampuan untuk belajar dari data dan tidak membutuhkan waktu lama dalam pembuatan model (Setiawan dan Rudiyanto,

2004). Keuntungan dari penggunaan ANN adalah kemampuannya untuk mempelajari hubungan yang tidak diketahui yang sudah ada sebelumnya antara data input dan *output* dari setiap sistim. Selain itu pemodelan dengan ANN memiliki atribut yang diinginkan dan kemampuan belajar dari contoh-contoh tanpa memerlukan data fisik secara eksplisit. ANN merupakan bagian dari *Mechine Learning* (ML). *Mechine Learning* adalah kecerdasan buatan yang bertujuan untuk mengoptimalkan performa dari suatu sistem dengan mempelajari data sampel atau data histori (Alpaydin, 2009). ANN banyak diterapkan untuk menyelesaikan permasalahan mengenai pengenalan pola, pengenalan suara, pengenalan karakter untuk pembacaan dokumen, pengenalan sinyal, penentuan pola gizi, dan pengolahan citra maupun permasalahan lainya. Jenis model ANN yang terdiri dari banyak lapisan disebut sebagai *Multi-Layer Perceptron* (MLP) yang berfungsi menghubungkan penuh diantara neuronnya. Kemampuan dari MLP ini dapat mmengklasifikasikan secara *powerfull*. Namun teknik klasifikasi menggunakan MLP ini memiliki kelemahan ketika input yang dimasukan berupa gambar. Gambar yang harus dilakukan pre-processing, segmentasi, dan di ekstrak untuk medapatkan kinerja yang optimal. Pengembangan lain dari MLP yang dapat mengatasi permasalahan ini adalah *Convolutional Neural Network* (CNN).

Convolutional Neural Network (CNN) merupakan salah satu metode *Deep learning* (DL) yang dapat digunakan untuk mendeteksi dan mengenali sebuah objek pada sebuah citra digital. *Deep Learning* merupakan salah satu sub bidang dari *Mechine Learning.* Pada dasarnya *Deep Learning* adalah implementasi konsep dasar dari *Mechine Learning* yang menerapkan algoritma ANN dengan lapisan yang lebih banyak. Banyaknya lapisan tersembunyi yang digunakan antara lapisan masukan dan lapisan keluaran, maka jaringan ini dapat dikatakan *deep neural net.* Beberapa tahun terakhir *Deep Learning* telah menunjukan performa yang luar biasa. Hal ini sebagain besar dipengaruhi faktor komputasi yang lebih kuat, data set yang besar dan teknik untuk melatih jaringan yang lebih dalam (Goodfellow, Bengio, Y, dan Courville, A., 2016). Kemampuan CNN di klaim sebagai model terbaik untuk memecahkan permasalahan *object detection* dan *object recognition*. Pada tahun 2012, Penelitian tentang CNN dapat melakukan pengenalan citra digital

dengan akurasi yang menyaingi manusia pada dataset tertentu. (A. Coates, H.Lee, A.Y. Ng, 2011). Namun dalam CNN, seperti model *deep learning* lainnya, memiliki kelemahan yaitu proses pelatihan model yang cukup lama. Tetapi dengan perkembangan *hardware* yang semakin pesat, hal tersebut dapat diatasi menggunakan teknologi *Graphical Procesing Unit* (GPU) dan PC yang memiliki spesifikasi tinggi. Berdasarkan latar belakang di atas, penelitian ini menerapkan implementasi dari metode *deep learning* menggunakan CNN untuk membantu mengenali tokoh-tokoh pewayangan. Penelitian ini berfokus terhadap bagaimana mengklasifikasikan citra wayang kedalam tokoh-tokoh wayang golek.

1.2. Rumusan Masalah

Adapun rumusan masalah dari penelitian ini adalah :

1. Bagaimana implementasi metode *Deep Learning* menggunakan CNN untuk mengklasifikasikan citra wayang berdasarkan tokoh-tokoh wayang golek ?
2. Bagaimana tingkat akurasi yang didapatkan dari hasil klasifikasi menggunakan CNN ?

1.3. Batasan Masalah

Adapun batasan masalah yang digunakan dalam peneliti ini adalah :

1. Data yang digunakan adalah citra wayang yang didapatkan dari hasil teknik *crawling*.
2. Citra yang digunakan memiliki ukuran *pixel* 64x64.
3. Klasifikasi citra ini hanya mencakup 3 tokoh wayang.

1.4. Tujuan Penelitian

Adapun tujuan penelitian yang digunakan dalam peneliti ini adalah :

1. Mengetahui implementasi metode *Deep Learning* menggunakan CNN untuk mengklasifikasikan citra wayang berdasarkan tokoh-tokoh wayang golek.
2. Mengetahui tingkat akurasi yang didapatkan dari hasil klasifikasi menggunakan CNN.

1.5. Manfaat Penelitian

Adapun manfaat yang diberikan dalam penelitian ini adalah :

1. Memberikan pengetahuan mengenai implementasi deep learning menggunakan *Convolutional Neural Network* untuk klasifikasi citra tokoh-tokoh wayang golek.
2. Mengetahui tingkat akurasi dari implementasi *Convolutioanl Neural Network* (CNN).
3. Mengklasifikasikan wayang berdasarkan tokoh dan karekter wayang golek.
4. Membantu para pecinta wayang dalam mengenali tokoh-tokoh wayang golek.

1.6. Sistematika penulisan

Sistematika penulisan yang dipergunakan dalam penulisan tugas akhir ini dapat diuraikan sebagai berikut :

BAB I PENDAHULUAN

Pada bab ini akan dibahas tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini memaparkan penelitian-penelitian terdahulu yang berhubungan dengan permasalahan yang diteliti dan menjadi acuan konseptual.

BAB III LANDASAN TEORI

Pada bab ini akan dibahas tentang teori-teori dan konsep yang berhubungan dengan penelitian yang dilakukan dan mendukung dalam pemecahan masalahnya. Selain itu, bab ini juga memuat teori-teori dalam pelaksanaan pengumpulan dan pengolahan data serta saat melakukan penganalisaan.

BAB IV METODOLOGI PENELITIAN

Bab ini memaparkan populasi dan sampel, variabel penelitian, jenis dan sumber data, metode analisis data, dan tahapan penelitian.

BAB V ANALISIS DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai analisa yang dilakukan terhadap hasil pengumpulan, pengolahan dan analisa data yang diperoleh dari hasil penelitian.

BAB VI PENUTUP

Pada bab ini akan dibahas mengenai kesimpulan yang diperoleh dari hasil penelitian dan analisa data yang telah dilakukan serta saran-saran yang dapat diterapkan dari hasil pengolahan data yang dapat menjadi masukan yang berguna kedepannya.

BAB II
KAJIAN PUSTAKA

Berdasarkan penelitian yang akan dilakukan, acuan dari beberapa penelitian terdahulu menjadi sangat penting dalam melakukan sebuah penelitian dengan tujuan untuk mengetahui hubungan antara penelitian yang akan dilakukan dengan penelitian terdahulu, sehingga dengan menambahkan acuan tersebut dapat menghindari adanya suatu duplikasi dalam penelitian yang akan dilakukan.

Banyak pengembangan sistem yang meneladani *Computer Vision* seperti *face detection, image recognition* maupun pengenalan pola tetentu. Pengembangan sistem ini menjadi sebuah fungsionalitas yang dapat mempermudah perkerjaan diberbagai bidang. Pengembangan dari deep learning ini sangat tepat dan efektif untuk digunakan untuk menyelesaikan permasalahan tersebut. Hal ini tidak lepas dengan adanya riset atau penelitian di bidang tersebut. Penelitian terdahulu mengenai *deep learning* menggunakan *convolutional neural network* sudah banyak dilakukan oleh para *reaseacher* pada berbagai macam *object*. Adapun penelitan yang dilakukan oleh Ardian Yusuf Wicaksono, (Wicaksono, 2017) mengenai Modifika Arsitektur *Convolutional Neural Network* untuk klasifikasi motif gambar batik. Penelitian yang dilakukan oleh Ardian Yusuf Wicaksono, dkk menggunakan metode CNN dengan mengembangkan pada arsitektur dari modelnya dengan mengkombinasi GoogleNet dan *Residual Networks* yang dinamai IncRes. Penelitian ini menggunakan 11 *class* dari tipe motif batik dengan jumlah data gambar 7112 yang dibagi kedalam 6401 digunakan untuk data latih (*train*) dan 711 digunakan untuk data uji (*test).* Dari hasil penelitian ini memperoleh *accuracy* sebesar 70.84 % dengan waktu 733 ms (*milisecond*).

Penerapan Convolutional Neural Network juga dapat dikembangkan dari sisi arsitektur dan banyaknya lapiran yang digunakan pada jaringan. Pembuatan arsitektur yang baik sangat berpengaruh pada klasifikasi citra untuk semua kategori. Tahun 2012 penerapan *Deep Learning* dengan metode CNN dipopulerkan dengan arsitektur AlexNet yang diuji dengan *dataset* ImageNet (Krizhevsky,2012).

Penelitian ini menggunakan *dataset* ImageNet LSVRC-2010 kedalam 1000 *classes.* Arsitektur yang dibuat oleh Alex Krizhevsky menunjukan hasil yang sangat signifikan pada *testing test* dengan *test error* sebesar 17 %. Hasil ini sudah dapat dinilai sangat baik karena citra yang digunakan pada dataset sangatlah banyak.

Pada tahun 2016 penerapan *Deep Learning* dengna menggunakan *Convolutional Neural Network* yang dilakukan oleh Muhammad Zufar dan Budi Setiyono yang diimplementasikan untuk pengenalan wajah secara *real-time* (Zufar, 2017). Metode ini diimplementasikan dengan bantuan *Library* OpenCVuntuk deteksi wajah dan perangkat Web Cam M-Tech 5 MP. *Dataset* yang digunakan yaitu himpunan gambar wajah yang di bagi menjadi dua jenis himpunan yaitu himpunan wajah *indoor* (kondisi pencahayaan gelap) dan himpunan wajah *outdoor* (kondisi pencahayaan terang). Hasil uji coba dengan menggunakan konstruksi model CNN sampai kedalaman 7 lapisan dengan input dari hasil ekstraksi *Extended Local Binary Pattern* dengan radius 1 dan *neighbor* 15 menunjukan kinerja pengenalan wajah meraih rata-rata akurasi lebih dari 89% dalam 2 frame perdetik. Penelitian ini menunjukan bahwa implementasi dari model CNN dapat di terapkan pada proses pengenalan wajah secara *real-time* dengan akurasi yang cukup tinggi.

Penelitian mengenai perbandingan antara Model CNN dengan model lain pernah dilakukan oleh Yiyu Hong dan Jongweon Kim (Hong, 2017). Penelitian ini diimplementasikan pada identifikasi karya lukisan. *Dataset* yang digunkana pada penelitian ini adalah data lukisan yang didownload dari google sebanyak 30000 gambar. Pembagian dari data tersebut 25000 untuk data *trainning* dan 5000 digunkana untuk data *testing.* Perbandingan *test errore* antara metode *Convolutional Neural Network* (CNN) dengan *Scale-Invariant Feature Transform* (SIFT) menghasilkan nilai error yang sangat signifikan yaitu *error* pada CNN sebesar 2 % dan *error* pada metode SIFT sebesar 15,6 %, selisih yang didapatkan diantara keduanya sebesar 13,6 % ini artinya penggunaan metode CNN lebih unggul dibanding dengan metode SIFT.

Adapun perbandingan metode dalam deteksi objek yang dilakukan oleh Tibor Trnovszky, dkk mengenai implementasi *Convolutional Neural Network* (CNN) pada pengenalan hewan dengan membandingkan beberapa metode

klasifikasi (Trnovsky, 2017). Penelitian ini mencoba untuk membandingkan metode CNN dengan beberapa metode klasifikasi lainnya yaitu *Principal Component Analisys* (PCA), *Linear Discriminant Analisys* (LDA), *Local Binary Patterns Histograms* (LBPH), dan *Support Vector Mechine* (SVM). *Dataset* yang digunakan pada penelitian ini yaitu *animal dataset* sebanya 500 subject yang dibagi menjadi 5 kelas dengan jumlah perkelas sebanyak 100 data. Hasil penelitian menunjukan bahwa dari ke lima metode yang dibandingkan dalam melakukan klasifikasi, penggunaan metode CNN memberikan hasil yang paling baik diantara metode lainnya yakni dengan memberikan tingkat akurasi sebesar 98 %. Hal ini menunjukan metode CNN sangat baik untuk diimplementasikan pada klasifikasi sebuah citra *animal*.

Penerapan metode CNN ini dapat diimplementasikan pada pengenalan rambu-rambu lalu lintas dijalan. Seperti penelitian yang dilakukan oleh S. Visalini mengenai pengenalan rambu-rambu lalu lintas dengan menggunaka *Convolutional Neural Network* (Visalini, 2017). *Dataset* yang di ambil secara langsung dengan *geolocation* menggunakan aplikasi android. Penelitian ini tidak menyebutkan jumlah *dataset* yang digunakan namun hasil dari tingkat akurasi yang diberikan menggunakan CNN untuk mendeteksi atau pengenalan rambu-rambu lalu lintas sebesar 85 % - 90% , dengan jumlah layer konvolusi 3 layer.

Adapun beberapa penelitian mengenai *image classification* dengan menggunakan metode yang berbeda, seperti *Support Vector Mechine* (SVM), *Naive bayes*, dan *Fuzzy Logic*. Seperti penelitian yang dilakukan oleh Rosli et al. (2012), penelitian ini bertujuan untuk mengklasifikasikan kualitas dari kematangan buah mangga menggunakan metode *fuzy inference engine.* Fitru yang digunakan dalam penelitian ini adalah rata-rata warna seluruh dan tepian kulit serta ukuran buah mangga. Tingkat akurasi yang dihasilkan menggunakan metode ini adalah 80 % (Rosli, 2012).

Penelitian image classification dengan menggunakan metode *Naive Bayes* pernah dilakukan oleh Dong-Chul Park. Penelitian ini menjelaskan bagaimana metode *Naive Bayes* bekerja dalam mengklasifikasikan dataset yang memiliki banyak kategori. Data yang digunakan dalam penelitian ini menggunakan data

Caltech yaitu data yang memiliki banyak kategori misalnya gambar pesawat, mobil, motor,dan sepeda. Setiap kategori memiliki 200 gambar, sehingga total gambar yang digunakan sebanyak 800 gambar. Tingkat akurasi yang dihasilkan dengan menggunakan metode ini sebesar 77 % dalam proses pengklasifikasian kategori gambar (Park, 2016).

Kemudian, adapun penelitian tentang *image classification* dengan menggunakan *Support vector Mechine* (SVM) dilakukan oleh Lida Hosseini et al. (2017). Penelitian ini menjelaskan penggunaan metode SVM untuk mengklasifikasikan gambar hyperspektral dimensi ruang. Tingkat akurasi yang didapatkan sebesar 73 % - 80 % (Hosseini, 2017).

Berdasarkan referensi diatas dijadikan sebagai acuan dalam penelitian ini. Fokus penelitian yaitu bagaimana sebuah algoritma dapat mengenali dan mengklasifikasikan sebuah citra tokoh-tokoh pewayangan khususnya wayang golek. Berikut merupakan perbandingan diantara penelitian-penelitian terdahulu, :

Tabel 2.1 Tabel Perbandingan Pustaka Metode CNN

No	Penulis	Dataset	Jumlah Layer Konvolusi	Keterangan
1	Krizhevsky et al, (2012)	1,2 Juta Citra ImageNet ILSVRC 2012	5	Menghasilkan nilai *error rate* sebesar 17 %
3	Muhammad Zufar dan Budi Setiyono (2016)	Data diambil secara langsung melalui WEB CAM	2	Menghasilkan Tingkat akurasi lebih dari 89% dalam 2 frame perdetik
2	Andrian Yusuf Wicaksono, et al, (2017)	7112 citra Batik	2	Menghasilkan tingkat akurasi 70.84 %

4	Yiyu Hong dan Jongweon Kim (2017)	30000 citra lukisan	5	Menghasilkan nilai *error* yang sangat signifikan yaitu *error* pada CNN sebesar 2 % dan error pada metode SIFT sebesar 15,6 %, selisih yang didapatkan diantara keduanya sebesar 13,6 %
5	Tibor Trnovszky, et al, (2017)	500 citra hewan	2	Menghasilkan tingkat akurasi sebsar 98 %
6	S. Visalini (2017)	500 citra *Traffic signs*	2	Menghasilkan tingkat akurasi sebsar 85-95 %

Tabel 2.2 Tabel Perbandingan Pustaka Metode Lain

No	Penulis	Dataset	Metode	Keterangan
1	Rosli, et, al (2012)	Data Buah Mangga	*Fuzzy Logic*	Menghasilkan tingkat akurasi sebesar 80 %
2	Dong-Chul Park, (2016)	Data *Caltech*	*Naive Bayes*	Menghasilkan tingkat akurasi pada proses *training* sebesar 77 %
3	Lida Hosseini, (2017)	Data *Hyperspectral* gambar dimensi ruang	*Support Vector Mechine*	Menghasilkan tingkat akurasi pada proses *training* 73-80 %

BAB III
LANDASAN TEORI

3.1. Wayang

Wayang marupakan salah satu kesenian tradisional Indonesia yang sudah di akui oleh UNESCO pada tahun 2003 sebagai warisan dunia. Kesenian ini berkembang di pulau Jawa. Tedapat dua versi wayang yaitu wayang orang dan wayang berwujud boneka. Wayang orang adalah wayang yang dimainkan secara langsung oleh orang dengan menggunakan kostum sebagai ciri khasnya, sedangkan wayang yang berwujud boneka merupakan wayang yang dimainkan oleh dalang. Beberapa wayang yang berwujud boneka ini diantaranya wayang kulit, wayang golek, dan wayang rumput. Kisah pada pertunjukan wayang biasanya berasal dari Mahabarata dan Ramayana yang sudah diubah oleh para pujangga dan Empu di Nusantara (Pasha, 2011).

3.2. Wayang Golek

Wayang golek merupakan salah satu kebudayaan Indonesia yang hidup dan berkembang di daerah Sunda (Jawa Barat). Kesenian ini dipandang sebagai salah satu bentuk tontonan rakyat yang kental dengan nilai-nilai kerakyatannya. Wayang golek terbuat dari kayu yang menyerupai bentuk dari sebuah boneka. Sumber cerita diambil dari sejarah, misalnya cerita Untung Suropati, Batavia, Sultan Agung, Banten, Trunajaya, dan lain-lain.

Pada pertunjukan wayang golek, terdiri dari dalang yang memainkan wayang golek tersebut berdasarkan cerita. Pertunjukan wayang golek tidak jauh berbeda dengan pertunjukan wayang-wayang lainnya, pertunjukan wayang ini disertai dengan nayaga. Nayaga adalah grup atau orang yang memainkan gamelan. Pertunjukan ini biasa dilakukan pada saat-saat perayaan tertentu, misalnya diacara pernikahan, khitanan, ataupun perayaan kemerdekaan.

3.3. Karakter Tokoh Wayang

Dalam budaya jawa, wayang merupakan salah satu dimensi budaya yang sangat penting. Wayang menjadi sumber inspirasi kehidupan masyarakat Jawa dari semua golongan masyarakat. Pertunjukan wayang menampilkan tokoh-tokoh wayang dan menunjukan bagaimana setiap peran itu dimainkan. Setiap tokoh wayang memiliki karakter yang jelas dapat diketahui dari sikap dan tindakan mana yang dapat diharapkan dari tokoh-tokoh tersebut (Suseno F. , 1991). Beberapa tokoh wayang yang memiliki macam-macam karakter diantaranya wujud Semar memiliki karater yang sabar dan bijaksana, semar juga sering disebut sebagai wayang penasihat. Kemudian ada wujud Buto yang memiliki paras yang buruk menggambarkan sebagai orang memiliki ambisi, sombong serta tidak mempunyai sifat sabar, dan wujud Bima yang dikenal orang jawa sebagai tokoh satria pinandhita, profesional *religious*, pekerja sufistik, dan panglima perang sekaligus guru besar. Seperi yang dikatakan oleh Puwradi dalam penelitiannya (Purwadi, 2013). Sifat dan watak yang dimiliki oleh tokoh-tokoh dalam pewayangan sangat bermacam-macam seperti tokoh dan watak yang dimiliki oleh manusia. Berikut adalah salah satu tokoh-tokoh pewayangan.

Gambar 3.1. Tokoh-Tokoh Wayang Golek

3.4. Citra Digital

Citra Digital adalah gambar dua dimensi yang dihasilkan dari gambar analog dua dimensi yang kontinus menjadi gambar diskrit melalui proses sampling. Proses perubahan citra menjadi citra digital dinamakan dengan digitasi. Digitasi merupakan proses mengubah sebuah gambar, teks, atau suara dari benda yang

dapat diliat ke dalam data elektronik dan dapat disimpan serta diproses untuk keperluan lainnya. Dalam konteks yang lebih luas, pengolahan citra digital lebih mengacu pada pemrosesan setiap dua data dimensi. Pengolahan citra digital adalah sebuah disiplin ilmu yang mempelajari tentang bagaimana teknik pengolahan sebuah citra. Citra yang dimaksud disini adalah sebuah gambar diam (foto) maupun gambar bergerak (Video). Sedangkan digital disini mempuyai maksud penting bahwa pengolahan citra/gambar dilakukan secara digital menggunakan komputer (Sutoyo, 2009). Dalam citra digital terdapat sebuah larik (*array*) yang berisi nilai-nilai real maupun kompleks yang di representasikan dengan derean bit tertentu.

Dalam sebuah komputer, citra digital dipetakan menjadi bentuk grid dan elemen piksel berbentuk matriks 2 dimensi. Setiap piksel-piksel tersebut memiliki angka yang mempresentasikan *channel* warna. Angka pada setiap piksel disimpan secara berurutan oleh sebuah komputer dan sering dikurangi untuk keperluan kompresi maupun pengolahan tertentu. Sebuah citra digital dapat mewakili oleh sebuah matriks yang terdiri dari **M** kolom **N** baris, dimana perpotongan antara kolom dan baris disebut piksel. (*pixel = picture element*), yaitu elemen terkecil dari sebuah citra. Piksel mempunyai dua parameter, yaitu koordinat dan intensitas atau warna. Nilai yang terdapat pada koordinat (x,y) adalah *f(x,y),* yaitu besar intensitas atau warna dari piksel di titik itu. Oleh karena itu, citra dapat dituliskan kedalam sebuah matriks :

$$f\ (x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ f(1,0) & f(1,1) & \cdots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{bmatrix} \qquad (3.1)$$

Berdasarkan rumus diatas, suatu citra *f(x,y)* dapat dituliskan kedalam fungsi matematis seperti berikut ini :

$$0 \leq x \leq M\text{-}1$$

$$0 \leq x \leq N\text{-}1$$

$$0 \leq f(x,y) \leq G\text{-}1$$

Dimana :

M = jumlah piksel baris pada *array* citra

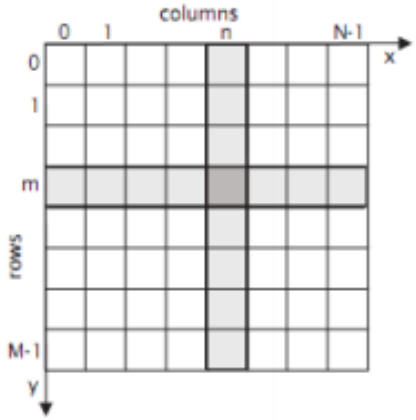
N = jumlah piksel kolom pada *array* citra

G = nilai skala keabuan (*grayscale*)

Besarnya nilai M, N, dan G biasanya merupakan perpengkatan dari dua seperti yang terlihat pada persamaan berikut :

$$M = 2m ; N = 2n; G = 2k \quad (3.2)$$

Dimana nilai ***m***, ***n***, dan ***k*** merupakan bilaingan positif. Interval (0,G) disebut dengan (*grayscale)*. Besarnya nilai G tergantung pada proses digitalisasinya. Biasanya keabuan 0 (nol) menyatakan intensitas hitam dan 1 (satu) menyatakan intensitas putih. Untuk citra 8 bit, nilai G sama dengan 28 = 256 warna (derajat keabuan).



Gambar 3.2. Representasi Citra Digital dalam 2 Dimensi
(Bernd, 2000)

3.4.1. Pengolahan Citra

Pengolahan citra adalah suatu proses pengolahan citra dengan menggunakan komputer menjadi sebuah citra yang memiliki kualitas yang lebih baik. Tujuan dari pengolahan citra ini adalah memperbaiki kualitas suatu citra sehingga dapat diinterpretasi dengan mudah oleh manusia atau atau sebuah mesin (komputer).

3.5. *Web Crawler*

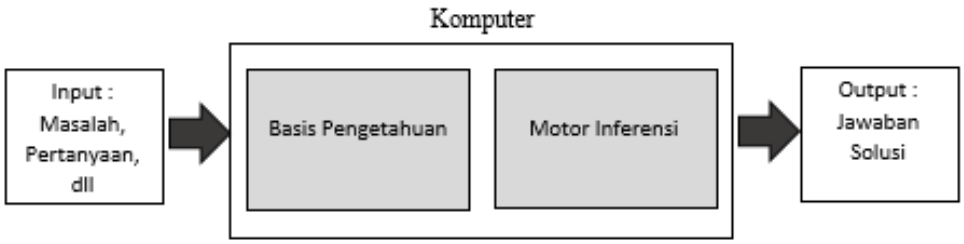
Web Crawler adalah meng-*crawl* (merayapi) seluruh informasi suatu website yang biasanya digunakan untuk meng-index suatu *website*, pemeliharaan website, atau digunakan untuk memperoleh data khusus contohnya email. Dan hal ini juga

dapat digunakan untuk memvalidasi *hyperlink* dan kode HTML. *Web Crawler* dimulai dengan me-list daftar URL yang akan dikunjungi, yang disebut dengan seed. *Web crawler* akan mengunjungi URL yang ada di daftar dan mengidentifikasi semua hyperlink di halaman tersebut serta menambahkannya kedalam daftar URL yang akan dikunjungi yang disebut *crawl frontier*. URL yang telah ada dikunjungi dan diambil informasi yang ada sesuai yang dibutuhkan. Dengan banyaknya jumlah URL yang mungkin di-*crawl* oleh *crawler server* yang membuatnya sulit untuk menghindari pengambilan konten yang sama. Misalkan protokol HTTP GET membuat kombinasi URL yang sangat banyakdan sedikit dari URL tersebut menghasilkan konten yang berbeda dan selebihnya menghasilkan konten yang sama untuk URL yang berbeda, inilah yang menimbulkan masalah bagi *crawler* agar bisa mengambil konten yang berbeda dari URL-URL tersebut (Wikipedia.org, 2018).

3.6. *Artificial Intellegence* (AI)

Artificial Intelligence (AI) merupakan bagian dari ilmu komputer yang mempelajari bagaimana menjadikan mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan manusia, bahkan bisa lebih baik. *Artificial Intelligence* (AI) menurut John McCarthy (1956) yang dikutip dari jurnal penelitian (Pannu, 2015) mengatakan bahwa AI bertujuan untuk mengetahui atau memodelkan proses berpikir manusia dan mendesain mesin sehingga bisa menirukan perilaku manusia. Dalam pembuatan aplikasi kecerdasan buatan terdapat dua hal yang menjadi bagian utama yang dibutuhkan yaitu :

1. Knowladge Base (Basis Pengetahuan). Bagian ini berisi tentang fakta-fakta, teori, pemikiran dan hubungan antara satu dengan yang lainnya.
2. *Inference Engine* (Motor Inferensi) yaitu kemampuan menarik kesimpulan berdasarkan pengalaman.



Gambar 3.3. Bagian Utama dalam *Artificial Intelligence (AI)*

Artificial Intelligence (AI) merupakan salah satu disiplin ilmu yang luas, beberapa lingkup utama AI antara lain adalah Sistem Pakar (*Expert System*), Pengolahan Bahasa Alami (*Natural Languange Processing/NLP),* Pengenalan Ucapan *(Speech Recognition), Computer Vision, Intelligent Computer-Aided Instruction,* dan lainnya. Sistem pakar adalah usaha untuk menirukan seorang pakar. Tujuan dari sistem pakar yaitu untuk mentransfer kepakaran dari seorang pakar ke komputer, kemudian ke orang lain (orang yang bukan pakar). Pengolahan Bahasa Alami yaitu dimana pengguna bisa melakukan komunikasi dengan komputer menggunakan bahasa sehari-hari. Pengenalan ucapan yaitu dimana manusia dapat melakukan komukasi dengan komputer menggunakan suara. *Computer vision* yaitu dalam hal menginterpretasikan objek atau gambar yang tampak melalui komputer. *Intelligent Computer-Aided Instruction* yaitu bagaimana komputer dapat berperan sebagai tutor yang dapat mengajar atau melatih.

Artificial Intelligence (AI) dibuat berdasarkan sistem yang memiliki keahlian seperti manusia pada domain tertentu yaitu disebut dengan *soft computing*. *Soft computing* merupakan inovasi baru dalam membangun sistem cerdas yang mampu beradaptasi dan bekerja lebih baik jika terjadi perubahan lingkungan. *Soft computing* juga mengeksploitasi adanya toleransi terhadap ketidakpastian, ketidaktepatan, dan kebenaran parsial sehingga dapat diselesaikan dan dikendalikan dengan mudah agar sesuai dengan realita. Metodologi yang sering digunakan dalam *soft computing* salah satunya adalah Jaringan Syaraf (menggunakan pembelajaran), yaitu Jaringan Syaraf Tiruan *(Artificial Neural Network/ANN)*. Metologi lain yang juga digunakan adalah Sistem *Fuzzy* (mengakomodasi ketepatan), *Probabilistic Reasoning* (Mengakomodasi Ketidakpastian), *Evolutionary Computing* (Optimasi).

Menurut Rich dan Knight (1991) ialah sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia. Pada awal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan zaman, maka peran komputer semakin mendominasi kehidupan manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, lebih dari itu, komputer diharapkan untuk dapat diberdayakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia

3.6.1. Mechine Learning

Istilah *machine learning* pertama kali didefinisikan oleh Arthur Samuel ditahun 1959. Menurut Arthur Samuel, *machine learning* adalah salah satu bidang ilmu komputer yang memberikan kemampuan pembelajaran kepada komputer untuk mengetahui sesuatu tanpa pemrogram yang jelas. Menurut Mohri dkk (2012) *machine learning* dapat didefinisikan sebagai metode komputasi berdasarkan pengalaman untuk meningkatkan performa atau membuat prediksi yang akurat. Definisi pengalaman disini ialah informasi sebelumnya yang telah tersedia dan bisa dijadikan data pembelajar.

Dalam pembelajaran *machine learning*, terdapat beberapa skenario-skenario. Seperti:

1. *Supervised Learning*

   Penggunaan skenario *supervised learning*, pembelajaran menggunakan masukan data pembelajaran yang telah diberi label. Setelah itu membuat prediksi dari data yang telah diberi label.
2. *Unsupervised Learning*

   Penggunaan skenario *Unsupervised Learning*, pembelajaran menggunakan masukan data pembelajaran yang tidak diberi label. Setelah itu mencoba untuk mengelompokan data berdasarkan karakteristik-karakteristik yang ditemui.
3. *Reinforcement Learning*

   Pada skenario *reinforcement learning* fase pembelajaran dan tes saling dicampur. Untuk mengumpulkan informasi pembelajar secara aktif dengan berinteraksi ke lingkungan sehingga untuk mendapatkan balasan untuk setiap aksi dari pembelajar.

Saat ini telah banyak pendekatan *machine learning* yang digunakan untuk deteksi *spam*, *Optical character recognition* (OCR), pengenalan wajah, deteksi penipuan *online*, NER (*Named Entity Recognition*), *Part-of-Speech Tagger*

3.6.2. Deep Learning

Deep Learning merupakan salah satu bidang dari *Machine Learning* yang memanfaatkan jaringan syaraf tiruan untuk implementasi permasalahan dengan dataset yang besar. Teknik *Deep Learning* memberikan arsitektur yang sangat kuat untuk *Supervised Learning*. Dengan menambahkan lebih banyak lapisan maka model pembelajaran tersebut bisa mewakili data citra berlabel dengan lebih baik. Pada *Machine Learning* terdapat teknik untuk menggunakan ekstraksi fitur dari data pelatihan dan algoritma pembelajaran khusus untuk mengklasifikasi citra maupun untuk mengenali suara. Namun, metode ini masih memiliki beberapa kekurangan baik dalam hal kecepatan dan akurasi.

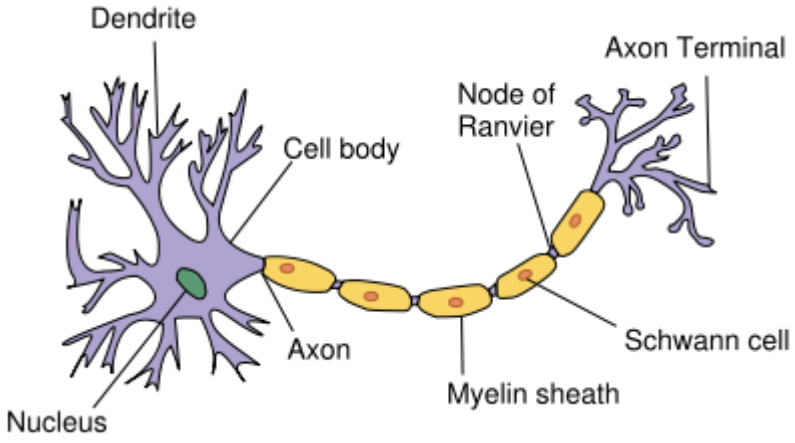
Aplikasi konsep jaringan syaraf tiruan yang dalam (banyak lapisan) dapat ditangguhkan pada algoritma Machine Learning yang sudah ada sehingga komputer sekarang bisa belajar dengan kecepatan, akurasi, dan skala yang besar. Prinsip ini terus berkembang hingga *Deep Learning* semakin sering digunakan pada komunitas riset dan industri untuk membantu memecahkan banyak masalah data besar seperti *Computer vision*, *Speech recognition*, dan *Natural Language Processing*. *Feature Engineering* adalah salah satu fitur utama dari *Deep Learning* untuk mengekstrak pola yang berguna dari data yang akan memudahkan model untuk membedakan kelas. Feature Engineering juga merupakan teknik yang paling penting untuk mencapai hasil yang baik pada tugas prediksi. Namun, sulit untuk dipelajari dan dikuasai karena kumpulan data dan jenis data yang berbeda memerlukan pendekatan teknik yang berbeda juga.

Algoritma yang digunakan pada *Feature Engineering* dapat menemukan pola umum yang penting untuk membedakan antara kelas Dalam *Deep Learning*, metode CNN atau *Convolutional Neural Network* sangatlah bagus dalam menemukan fitur yang baik pada citra ke lapisan berikutnya untuk membentuk hipotesis nonlinier yang dapat meningkatkan kekompleksitasan sebuah model.

Model yang kompleks tentunya akan membutuhkan waktu pelatihan yang lama sehingga di dunia *Deep Learning* pengunaan GPU sudah sangatlah umum (Danukusumo, 2017).

3.6.3. *Artificial Neural Network*

Artificial Neural Network (ANN) merupakan suatu model komputasi paralel yang meniru fungsi dari sistem jaringan syaraf biologi otak manusia. Dalam otak manusia terdiri dari milyaran neuron yang saling berhubungan. Hubungan ini disebut dengan *Synapses*. Komponen neuron terdiri dari satu inti sel yang akan melakukan pemrosesan informasi, satu akson (*axon*) dan minimal satu *dendrit*. Informasi yang masuk akan diterima oleh *dendrit.* Selain itu, *dendrit* juga menyertasi akson sebagai keluaran dari suatu pemrosesan informasi.



Gambar 3.4. Jaringan Syaraf Manusia

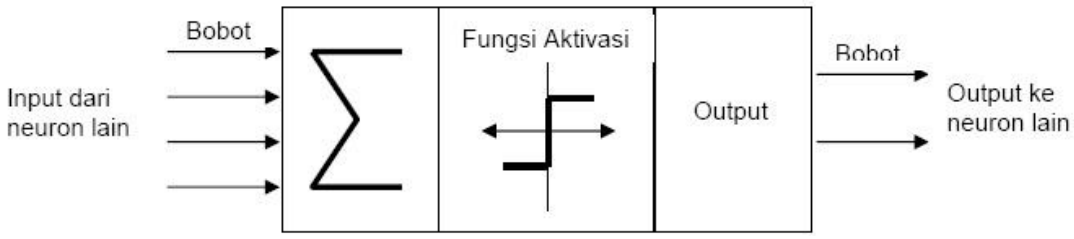
Cara kerja dari sistem syaraf diatas adalah bermula pada sinyal masuk melalui dendrit menuju *cell body.* Kemudian sinyal akan di proses didalam *cell body* berdasarkan fungsi tertentu (*Summation Proses*). Jika sinyal hasil proses melebihi nilai ambang batas (*treshold*) tertentu maka sinyal tersebut akan membangkitkan neuron untuk meneruskan sinyal tersebut. Sedang jika dibawah nilai ambang batasnya maka sinyal tersebut akan dihalangi (*inhibited*). Kemudian sinyal yang diteruskan akan menuju ke axon dan akhirnya menuju ke neuron lainnyamelewati *synapse*.

ANN merupakan sistem adatif yang dapat mengubah strukturnya untuk memecahkan suatu masalah berdasarkan informasi *internal* maupun *eksternal*. Menurut Pham dalam jurnal Hermantoro (Pham, 1994) mengatakan bahwa ANN bersifat fleksibel terhadap inputan data dan menghasilkan *output* respon konsisten. ANN telah banyak digunakan dalam area yang luas. Menurut Kumar & Haynes (Kumar, 2003) dalam jurnal Ulil Hamida (Hamida, 2014) menjelaskan, penerapan ANN dapat mengidentifikasi beberapa aplikasi yaitu:

1. Estimasi/prediksi (aproksimasi fungsi, peramalah)
2. Pengenalan Pola (klasifikasi, diagnosis, dan analisis diskriminan)
3. Klustering (pengelompokan tanpa adanya pengetahuan sebelumnya).

3.6.4. Komponen *Neural Network*

Neural Network memiliki beberapa tipe yang berbeda-beda, akan tetapi hampir semua komponen yang dimiliki sama. Seperti halnya jaringan syaraf pada otak manusia, *neural network* juga terdir dair beberapa neuron unit yang saling behubungan. Masing-masing dari neuron tersebut akan melakukan transformasi informasi yang diterima melalui sambungan keduanya menuju neuron lain. Hubungan ini biasanya disebut dengan sebutan bobot(*Weight).* Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tertentu. Berikut adalah struktur Neuron pada *neural network :*



Gambar 3.5. Struktur *Neural Network*

Berdasarkan gambar 3.5 diatas menunjukan struktur yang dimiliki oleh *Neural Network*. Komponen yang dimiliki struktur tersebut sebagai berikut :

1. Input terdiri dari variabel independet (X_1, X_2, X_3,..... X_n,) yang merupakan sebuah sinyal yang masuk ke sel syaraf.

2. Bobot (*Weigth)* terdiri dari beberapa bobot (W_1, W_2, W_3,..... W_n,) yang berhubungan dengan masing-masing node.
3. Threshod merupakan nilai ambang batas internal dari node. Besar nilai ini mepengaruhi aktivasi dari *output* node y.
4. *Activation Function* (Fungsi Aktivasi) merupakan operasi matematika yang dikenal pada sinyal *output* y.

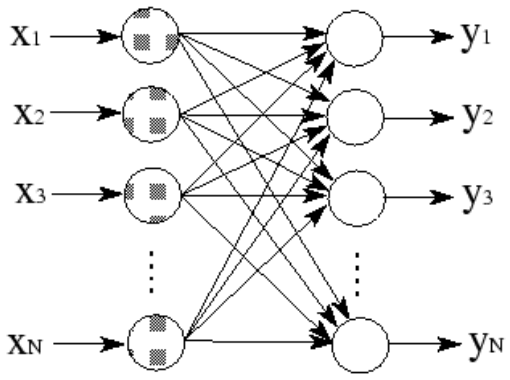
Cara kerja struktur *neural network* diatas tidak jauh berbeda dengan struktur jaringan syaraf pada manusia. Informasi (*input*) akan dikirimkan dengan bobot kedatangan tertentu. Input tersebut kemudian diproses oleh suatu fungsi perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan ini kemudian akan dibandingkan dengan suatu nilai ambang (*treshold*) tertentu melalui fungsi aktivasi setiap neuron. Jika input tersebut melewati suatu nilai ambang tertentu, maka neuron tersebut akan diaktifkan. Jika tidak, neuron tersebut tidak akan diaktifkan. Apabila neuron diaktifkan, selanjutnya *neuron* tersebut akan mengirimkan *output* melalui bobot-bobot *outputnya* ke semua *neuron* yang berhubungan dengannya, begitu seterusnya.

Pada *neuron layer*, penempatan *neuron-neuron* akan dikumpulkan dalam *neuron layer* (lapisan-lapisan). Kemudian *neuron-neuron* pada satu lapisan akan di hubungkan dengan lapsan-lapisan sebelum dan sesudahnya, kecuali lapisan *input* dan *output.* Informasi yang di bawa dari langkah *input* awal akan dirambatkan dari lapisan ke lapisan dari lapisan *input* sampai lapisan *output.* Lapisan ini sering disebut dengan istilah *hidden layer* (lapisan tersembunyi). Pada umumnya setiap *neuron* terletak pada lapisan yang sama akan memiliki keadaan yang sama. Sehingga pada setiap lapisan sama , setiap *neuron* akan memiliki fungsi aktifasi yang sama. Koneksi antar lapisan dengan *neuron* harus selalu berhubungan. Faktor terpenting dalam menentukan kelakuan suatu *neuron* adalah terletak pada pola bobot dan fungsi aktivasinya.

3.6.5. Arsitektur *Neural Network*

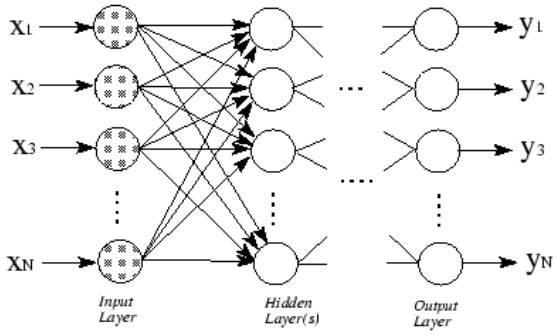
Pada *Neural Network* , *neuron-neuron* yang ada pada lapisan yang sama memiliki keadaan yang sama. Terdapat faktor penting dalam menentukan sifat suaru neuron yaitu bobot (*Weight*) dan penggunaan fungsi aktivasi dari neuron tersebut. Setiap lapisan pada neuron memiliki fungsi aktivasi yang sama. Arsitektur yag dapat dibentuk oleh ANN bermacam-macam. Dari yang paling sederhana terdiri satu *neuron* (*single neuron)* sampai yang paling rumit menjadi multi *neuron* (*multiple neuron*) dalam satu lapis (*single layer*), sampai jaringan *multiple neuron* dalam *multiple layers*. Beberapa jaringan tersebut memiliki kemampuan yang berbeda-beda. Semakin rumit suatu jaringan, maka persoalan yang dapat diselesaikan menjadi lebih luas. Namun terdapat kelemahan yaitu kerumitan tersebut dapat menimbulkan persoalan tersendiri pada kebutuhan proses *trainning* dan simulasi (*testing*) yang akan memerlukan waktu lebih lama. Menurut Hermawan (2006), Arsitektur *neural network* dapat dibagi berdasarkan jumlah lapisannya diantaranya :

1. *Single Layer Neural Network :* Jaringan dengan lapisan tunggal terdiri dari 1 lapisan *input* dan 1 lapisan *output*. Setiap neuron yang terdapat di dalam lapisan *input* selalu terhubung dengan setiap neuron yang terdapat pada lapisan *output*. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi *:*



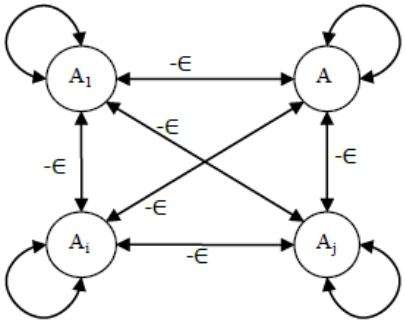
Gambar 3.6. Struktur *Single Layers Neural Networks*

2. *Multiple Layers Neural Network* : Jaringan dengan lapisan jamak memiliki ciri khas tertentu yaitu memiliki 3 jenis lapisan yakni lapisan *input*, lapisan *output*, dan lapisan tersembunyi. Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan jaringan dengan lapisan tunggal. Akan tetapi, proses pelatihan sering membutuhkan waktu yang cenderung lama.



Gambar 3.7. Struktur *Multiple Layers Neural Networks*

3. *Competitive Layers :* Pada jaringan ini sekumpulan neuron bersaing untuk mendapatkan hak menjadi aktif. Contoh algoritma yang menggunakan jaringan ini adalah LVQ.



Gambar 3.8. *Competitive Layers*

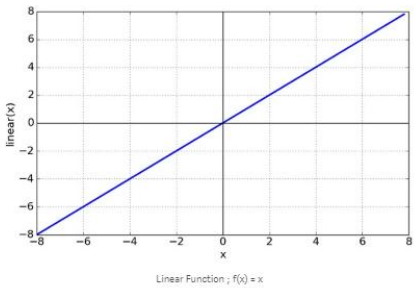
3.6.6. Fungsi Aktivasi

Fungsi aktivasi merupakan fungsi yang menggambarkan hubungan antara tingkat aktivitas internal (*summation function*) yang mungkin berbentuk linear ataupun *non-linear*. Fungsi ini bertujuan untuk menentukan apakah *neuron* diaktifkan atau tidak. Menurut Samuel Sena dalam articelnya yang diunggah dalam

website Medium (Sena, Pengenalan Deep Learning Part 1 : Neural Network, 2018) ada beberapa fungsi aktivasi yang sering digunakan dalam *Neural Nerwork*, yaitu sebagai berikut :

1. **Fungsi Aktivasi Linear**

   Fungsi Aktivasi lenear merupakan fungsi yang memiliki nilai *output* yang sama dengan nilai *inputnya*. Hal ini berkaitan dengan, jika sebuah *neuron* menggunakan *linear activation*, maka keluaran dari *neuron* tersebut adlaah *weighted sum* dari *input* + bias. Grafik fungsi *linear* ditunjukan oleh gambar 3.8.

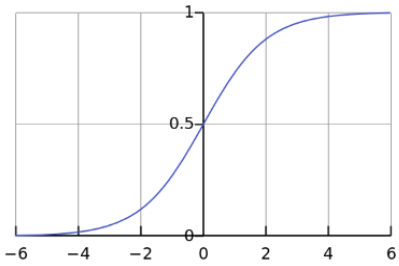


Gambar 3.9. Fungsi Aktivasi *Linear*

2. **Fungsi Aktivasi Sigmoid**

Fungsi aktivasi sigmoid merupakan fungsi nonlonear. Masukan untuk fungsi aktivasi ini berupa bilangan real dan output dari fungsi tersebut memiliki range antara 0 sampai 1. Berikut ini grafik fungsi aktivasi sigmoid:

$$A = \frac{1}{1+e^{-x}}$$



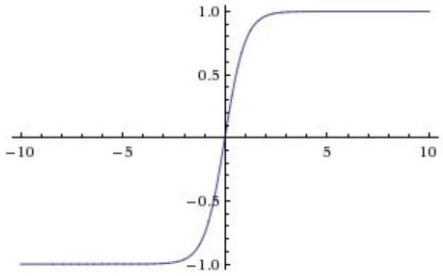
Gambar 3.10. Fungsi Aktivasi Sigmoid

Jika input dari suatu node pada *neural network* bernilai negatif maka keluaran yang didapatkan adalah 0, sedangkan jika masukannya berilai positif maka keluaran nilainya adalah satu. Fungis ini memiliki kekurangan yaitu sigmoid dapat mematikan gradient, ketika aktivasi dari *neuron* mengeluarkan nilai yang berada pada range 0 atau satu, dimana gradient di wilayah ini hampir bernilai 0. Kemudian *output* dari sigmoid tidak *zero-centered*

3. Fungsi Aktivasi Tanh

Fungsi aktivasi Tanh merupakan fungsi nonlonear. Masukan untuk fungsi aktivasi ini berupa bilangan real dan output dari fungsi tersebut memiliki range antara -1 sampai 1. Berikut ini grafik fungsi aktivasi tanh :

$$tanh(x) = 2\sigma(2x)-1$$

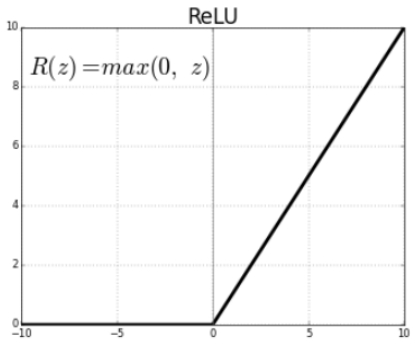


Gambar 3.11. Fungsi Aktivasi Tanh

Sama seperti fungsi sigmoid, fungsi ini memiliki kekurangan yaitu dapat mematikan gradient, akan tetapi fungsi ini juga memiliki kelebihan yaitu *output* yang dimiliki fungsi Tanh merupakan *zero-centered*. Dalam pengaplikasiannya fungsi Tanh lebih menjadi pilihan jika dibandingkan dengan fungsi sigmoid. Fungsi Perlu diketahui fungsi tanh merupakan pengembangan dari fungsi Sigmoid.

4. Fungsi Aktivasi ReLU

Pada dasarnya fungsi ReLU (*Rectified Linear Unit*) melakukan "*treshold*" dari 0 hingga *infinity*. Fungsi ini menjadi salah satu fungsi yang populer saat ini. Berikut ini grafik fungsi aktivasi tanh



Gambar 3.12. Fungsi aktivasi ReLU

Pada fungsi ini masukan dari neuron-neuron berupa bilangan negatif, maka fungsi ini akan menerjemahkan nilai tersebut kedalam nilai 0, dan jika masukan bernilai positif maka *output* dari neuron adalah nilai aktivasi itu sendiri. Fungsi aktivasi ini memiliki kelebihan yaitu dapat mempercepat proses konvigurasi yang dilakukan dengan *Stochastic Gradient Descent* (SGD) jika dibandingkan dengan fungsi sigmoid dan tanh. Namun aktivasi ini juga memiliki kelemahan yaitu aktivasi ini bisa menjadi rapuh pada proses *training* dan bisa membuat unit tersbut mati.

3.6.7. Algoritma *Backpropagation*

Neural network merupakan suatu model komputasi yang sistemnya mengikuti syaraf manusia. Jaringan ini membutuhkan proses pembelajaran. Pembelajaran ini bertujuan untuk melakukan suatu proses dalam enentuka nilai bobot (*weight*) yang tepat untuk masing-masing *input*. Salah satu algoritma pembelajaran yang dimiliki oleh *neural network* adalah *backpropagation. Backpropagation* merupakan algoritma pembelajaran yang digunakan oleh perceptron dengan banyak lapisan untuk mengubah setiap bobot yang terhubung dengan *neuron* pada *hidden layers.* Penggunaan *error* di dalam *backpropagation* bertujuan untuk mengubah nilai setiap bobot dalam arah mundur (*backward*). Sebelum mendapatkan *error* ini, terdapat tahap awal yang harus dilakukan yaitu tahap perambatan maju (*forward*).

Pelatihan *backpropagation meliputi* tiga tahap. Tahap pertama adalah tahap maju (*forward*). tahap ini menghitung maju tahap layer *input* sampai tahap layer

output dengan menggunakan fungsi aktivasi yang telah ditentukan. Tahap kedua adalah tahap mundur (*backward*), pada tahap ini selisih antara *output* jaringan dengan target yang diinginkan merupakan kesalahan yang terjadi. Kesalahan tersebut dipropagasikan mundur, mulai dari garis yang terhubung langsung dengan setiap unit pada layer *output.* Kemudian tahap yang ketiga adalah tahap yang akan memodifikasi bobot untuk menurunkan tingkat kesalahan yang terjadi (Jumarwanto, 2009). Berikut adalah langkah dari altgoritma *Backpropagation*:

Tahap Pertama : Propagasi maju (*Forward*)

a. Langkah 0 : Inisialisasi semua bobot dengan bilangan kecil
b. Langkah 1 : Jika kondisi penghentian belum terpenuhi, lakukan langkah 2-9
c. Langkah 2 : Untuk setiap pasangan data pelatihan lakukan langkah 3-8
d. Langkah 3 : Tiap unit masukan menerima sinyal dan meneruskannya ke unit tersembunyi di atasnya
e. Langkah 4 : Hitunglah semua *output* di unit tersembunyi tersebut z_j (*j*=1,2,...,*p)*. Perhatikan rumus dibawah ini :

   $z\_net_j = V_{jo} + \sum_{i=1}^{n} x_j v_{ji}$ (3.3)

   $z_j = f(z\_net_j) = \frac{1}{1+e^{-z\_net_j}}$ (3.4)

f. Langkah 5 : Hitunglah semua keluaran jaringan di unit y_k (k = 1,2, ... , m) perhatikan rumus dibawah ini :

   $y\_net_k = w_{ko} + \sum_{j=1}^{p} z_j w_{kj}$ (3.5)

   $y_k = f(y\_net_k) = \frac{1}{1+e^{-y\_net_k}}$ (3.6)

Tahap Kedua : Propagasi Mundur

g. Langah 6 : Hitung faktor δ unit keluaran berdasarkan *error* pada setiap y_k (k =1,2, ..., m) Perhatikan rumus berikut ini :

   $\delta_k = (t_k - y_k)\, f'(y\_net_k) = (t_k - y_k)\, y_k (1 - y_k)$ (3.7)

   δ_k merupakan unit kesalahan yang akan dipakai dalam perubahan bobot *layer* di bawahnya (langkah 7). Hitung suku perubahan bobot w_{kj} dengan laju percepatan α. Peratikan rumus berikut ini :

   $\Delta w_{kj}\ \alpha\delta_k\, z_j$; k = 1,2, ... , m ; j = 0,1,2, ... , p (3.8)

h. Langkah 7 : Hitung faktor δ unit tersembunyi berdasarkan *error* pada setiap unit tersembunyi z_j (j = 1,2, ... , p). Perhatikan rumus berikut ini :

$\delta\_net_j = \sum_{k=1}^{n} \delta_j w_{kj}$ (3.9)

faktor δ unit tersembunyi :

$\delta_j = \delta\_net_j \, f'(z\_net_j) = \delta\_net_j z_j(1-z_j)$ (3.10)

Hitung suku perubahan bobot v_{ji} (yang di pakai nanti untuk merubah bobot v_{ji} . Perhatikan rumus berikut ini :

$\Delta v_{kj} = \alpha \delta_j x_i$; j = 1,2, ... , p ; i= 0,1,2, ... , n (3.11)

Tahap Ketiga : Perubahan bobot

i. Langkah 8 : Hitunglah semua perubahan pada bobot. Rumus perubahan bobot garis yang menuju ke unit keluaran sebagai berikut :

w_{kj}(baru) = w_{kj} (lama) + Δw_{kj} (k =1,2, ... m ; j = 0,1,2, ... p) (3.12)

Perubahan bobot garis yang menuju ke unit tersebunyi :

v_{ji}(baru) = v_{ji} (lama) + Δv_{ji} (j =1,2, ... ,p ; i = 0,1,2, ... n) (3.13)

j. Langkah 9 : Menguji apakah kondisi berhenti sudah terpenuhi.

Kondisi berhenti ini terpenuhi jika nilai kesalahan yang dihasilkan lebih kecil dari nilai kesalahan referensi.

3.6.8. *Stochastic Gradient Descent*

Gradient Descent adalah salah satu algoritma paling populer dalam melakukan optimasi pada *artificial neural network.* Algoritma ini digunakan untuk mengupdate sebuah parameter dalam hal ini adalah bobot (*weight*) dan bias. Algoritma ini cukup sederhana untuk dipahami. Pada dasarnya algoritma ini berfungsi untuk mengurangi inisial *weight* dengan "sebagian" dari nilai *gradient* yang sudah didapatkan. *Gradient Descent* bekerja dengan cara meminimalkan fungsi J(θ) yang memiliki parameter θ dengan memperbarui parameter ke suatu arah menurun. Tujuan pengoptimalan dari algoritma ini untuk menemukan parameter yang dapat meminimalkan *loss function* (Ruder, 2018) . *Gradient Descent* memiliki *Learning Rate* (η) yang digunakan untuk menentukan langkah-langkah yang kita ambil untuk mencapai titik minimum. Hal ini bisa digambarkan dimana suatu objek akan menuruni

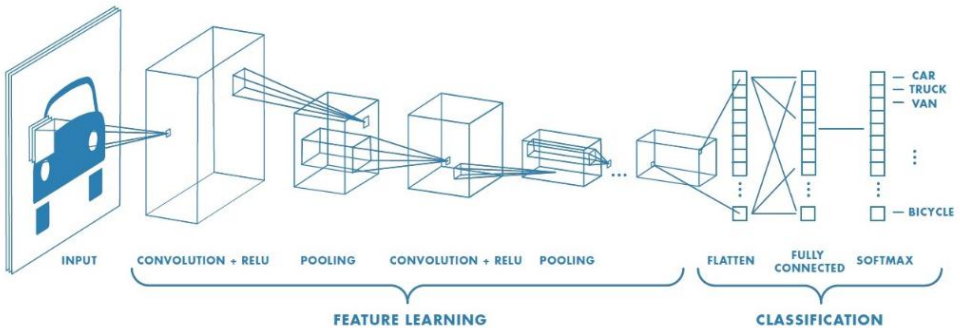
sebuah bukit dengan langkah tersebut hingga mencapai pada lembah (titik minimum). Stochastic Gradient Descent (SGD) adalah metode gradient descent yang melakukan update parameter untuk setiap data pelatihan x(i) serta label y(i). Persamaan dari algoritma ini sebagai berikut :

$$\theta = \theta - \eta \,.\nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \quad (3.14)$$

SGD ini sering melakukan *update* dengan varians tinggi yang menyebabkan fungsi objektif menigkat secara tidak beraturan. Disisi lain, hal ini dapat membuat *loss function* akan melompat ke titik minimal yang baru dan berpotensi melompat ke minimum yang tidak pasti. Namun, hal ini dapat dicegah dengan cara mengurangi *learning rate,* dan SGD akan menurunkan nilai *loss function* ke titik minimum secara optimal.

3.7. *Convolutional Neural Network*

Convolutional Neural Network (CNN) merupakan pengembangan dari *multilayer perceptron* (MLP) yang didesain untuk mengolah data dua dimensi dalam bentuk citra. CNN ini termasuk kedalam jenis *Deep Neural Network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Pada dasarnya klasifikasi citra dapat digunakan dengan MLP, akan tetapi dengan metode MLP kurang sesuai untuk digunakan karena tidak menyimpan informasi spasial dari data cita dan menganggap setiap piksel adalah fitur yang independen sehingga menghasilkan hasil yang kurang baik. Penelitian awal yang mendasari oenemuan CNN ini pertama kali dilakukan oleh Hubel dan Wiesel (Hubel & Wiesel, T, 1968) mengenai *viual cortex* pada indera penglihatan kucing. Secara teknis, CNN adalah sebuah arsitektur yang dapat dilatih dan terdiri dari beberapa tahap. Masukan (*input*) dan keluaran (*output*) dari setiap tahap adalah terdiri dari beberapa *array* yang biasa disebut *feature map.* Setiap tahap terdiri dari tiga *layer* yaitu konvolusi, fungsi aktivasi *layer* dan *pooling layer*. Berikut adalah jaringan arsitektur *Convolutional Neural Network* :



Gambar 3.13. Arsitektur *Convolutional Neural Network*

Berdasarkan gambar diatas, Tahap pertama pada arsitektur CNN adalah tahap konvolusi. Tahap ini dilakukan dengan menggunakan sebuah kernel dengan ukuran tertentu. Perhitungan jumlah kernel yang dipakai tergantung dari jumlah fitur yang dihasilkan. Kemudian dilanjutkan menuju fungsi aktivasi, biasanya menggunakan fungsi aktivasi ReLU (*Rectifier Linear Unit*), Selanjutnya setelah keluar dari proses fungsi aktivasi kemudian melalui proses *pooling*. Proses ini diulang beberapa kali sampai didapatkan peta fitur yang cukup untuk dilanjutkan ke *fully connected neural network,* dan dari *fully connected network* adalah *output class.*

3.7.1. *Convolution Layer*

Convolution layer merupakan bagian dari tahap pada arsitektur CNN. Tahap ini melakukan operasi konvolusi pada *output* dari layer sebelumnya. Layer tersebut adalah proses utama yang mendasari jaringan arsitektur CNN. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *Feature Map* dari input citra. *Input* dan *output* ini dapat dilihat sebagai dua argumen bernilai riil. Operasi konvolusi dapat dituliskan sebagai berikut :

$$s(t) = (x*t)\,(t) = \sum_{\alpha=-\infty}^{\infty} x(\alpha) * w(t-\alpha) \qquad (3.15)$$

Keterangan :

S(t) = Fungsi hasil operasi konvolusi

X = Input

W = bobot (kernel)

Fungsi *s(t)* memberikan *output* tunggal berupa *feature Map*. Argumen pertama adalah *input* yang merupakan *x* dan argumen kedua *w* sebagai kernel atau filter. Apabila dilihat *input* sebagai citra dua dimensi, maka bisa dikatakan *t* sebagai piksel dan menggantinya dengan *i* dan *j*. Maka dari itu, operasi untuk konvolusi ke *input* dengan lebih dari satu dimensi dapat menulis sebagai berikut :

$$s(i,j) = (K*I)\,(i,j) = \sum_{\infty}\sum_{\infty} I(i-m, j-n)K(m,n) \quad (3.16)$$

$$s(i,j) = (K*I)\,(i,j) = \sum_{\infty}\sum_{\infty} I(i+m, j+n)K(m,n) \quad (3.17)$$

Berdasarkan kedua persamaan diatas merupakan perhiutngan dasdar dalam opersai konvolusi, dengan *i* dan *j* adalah sebuah piksel dari citra. Perhitungan tersebut bersifat komulatif dan muncu saat K sebagai kernel, kemudian I sebagai *input* dan kernel yang dapat dibalik relatif terhadap *input.* Sebagai alternatif operasi konvolusi dapat dilihat sebagai perkalian perkalian matriks antara citra masukan dan kernel dimana keluarannya dihitung dengan *dot product.* Selain itu, penentuan volume *output* juga dapat ditentukan dari masing-masing lapisan dengan *hyperparameters*. *Hyperparameter* yang digunakan pada persamaan di bawah ini digunakan untuk menghitung banyaknya neuron aktivasi dalam sekali *output.* Perhatikan persamaan berikut

$$(W - F + 2P)/(S + 1) \quad (3.18)$$

Keterangan :

W = Ukuran volume gambar

F = Ukuran Filter

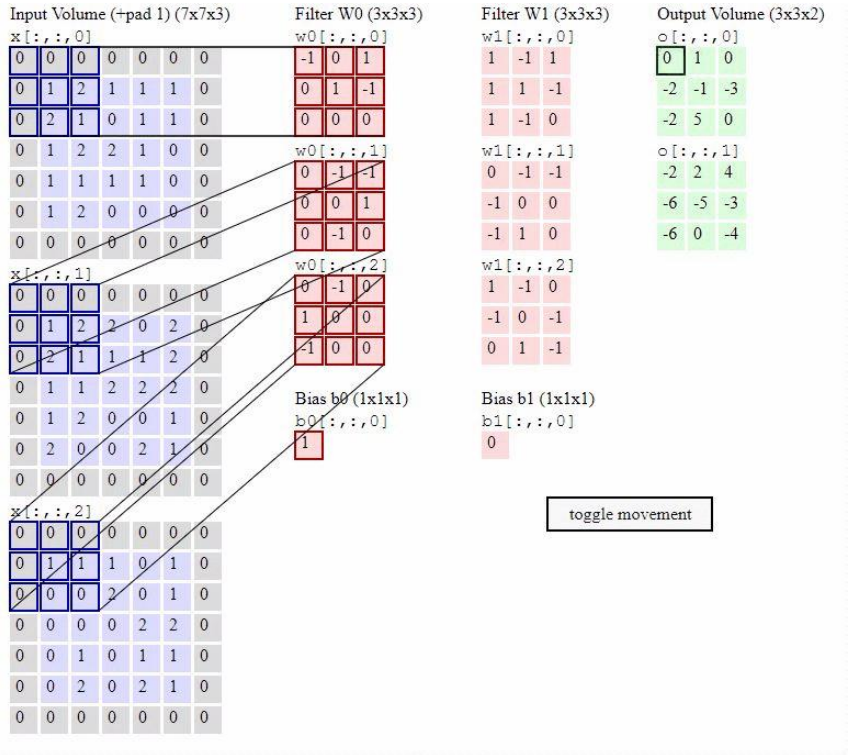
P = Nilai *Padding* yang digunakan

S = Ukuran Pergeseran (*Stride*)

Berdasarkan persamaan di atas, dapat dihitung ukuran spasial dari volume *output* dimana *hyperparameter* yang dipakai adalah ukuran volume (W), *filter* (F), *Stride* yang diterapkan (S) dan jumlah *padding* nol yang digunakan (P). *Stride* merupakan nilai yang digunakan untuk menggeser *filter* melalui *input* citra dan

Zero Padding adalah nilai untuk mendapatkan angka nol di sekitar *border* citra. Berikut adalah operasi

Convolutional Layer terdiri dari *neuron* yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (*pixels*). Sebagai contoh, layer pertama pada *feature extraction layer* biasanya adalah conv. Layers dengan ukuran 5x5x3. Panjang 5 *pixels*, tinggi 5 *pixels* dan tebal/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.* Perhatikan ilustrasi berikut :

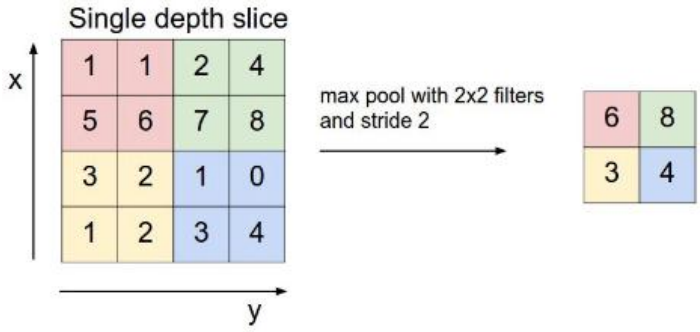


Gambar 3.14. *Convolution Layer*
(Medium Samuel Sena, 2017)

3.7.2. Operasi *Pooling*

Pooling merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling. Pooling Layer* biasanya berada setelah conv. Pada dasarnya pooling layer terdiri dari sebuah filter dengan ukuran dan stride tertentu yang akan

secara bergantian bergeser pada seluruh area *feature map.* Dalam pooling layer terdapat dua macam *pooling* yang biasa digunakan yaitu *average pooling* dan *max-pooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata, sedangkan pada *max-pooling* adalah nilai maksimal. Lapisan *Pooling* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran volume output pada *Feature Map*, sehingga mengurangi jumlah parameter dan perhitungan di jaringan, untuk mengendalikan *Overfitting*. Lapisan pooling bekerja di setiap tumpukan *feature map* dan melakukan pengurangan pada ukurannya. Bentuk lapisan *pooling* umumnya dengan menggunakan filter dengan ukuran 2x2 yang diaplikasikan dengan langkah sebanyak dua dan beroperasi pada setiap irisan dari inputnya. Berikut ini adalah contoh gambar operasi *max-pooling* :



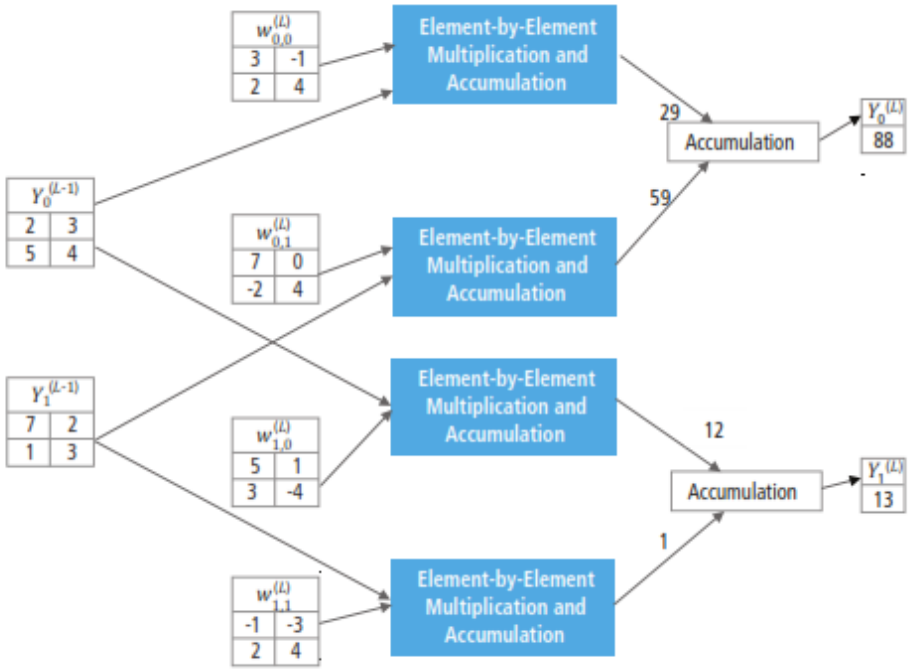
Gambar 3.15. Operasi *Max-Pooling*
(Medium Samuel Sena, 2017)

Berdasarkan gambar diatas menunjukan proses dari *max-pooling*. *Output* dari proses *pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal. Lapisan pooling diatas akan beroperasi pada setiap irisan kedalaman volume input secara bergantian. Jika dilihat dari gambar diatas operasi *max-pooling* dengan menggunakan ukuran filter 2x2. Masukan pada proses tersebut berukuran 4x4, dari masing-masing 4 angka pada input operasi tersebut diambil nilai maksimalnya kemudian dilanjutkan membuat ukuran *output* baru menjadi ukuran 2x2.

3.7.3. *Fully-Conected Layer*

Fully-Conected Layer adalah sebuah lapisan dimana seua *neuron* aktivasi dari lapisan sebelumnya terhubung semua dengan *neuron* di lapisan selanjutnya sama seperti halnya dengan *neural network biasa.* Pada dasarnya lapisan ini biasanya digunakan pada MLP (*Multi Layer Perceptron*) yang mempunyai tujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear.

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. Berikut ini adalah proses *fully-connected* :

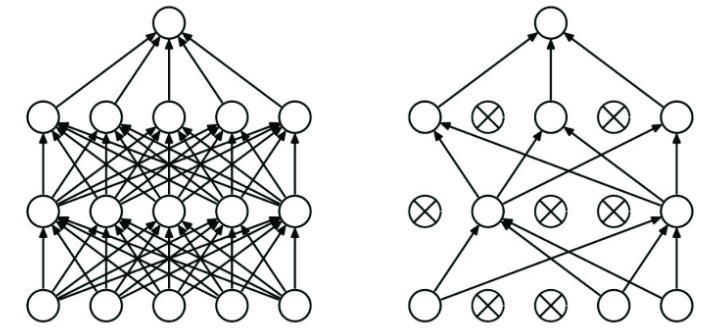


Gambar 3.16. *Processing of a Fully-Connected Layer*

3.7.4. *Droput Regulation*

Dropout merupakan sebuah teknik regulasi jaringan syaraf dengan tujuan memilih beberapa neuron secara acak dan tidak akan dipakai selama proses

pelatihan, dengan kata lain neuron-neuron tersebut dibuang secara acak. Hal ini berarti bahwa kontribusi neuron yang dibuang akan diberhentikan sementara jaringan dan bobot baru juga tidak diterapkan pada neuron pada saat melakukan *backpropagation.* Berikut adalah gambar proses *dropout :*



Gambar 3.17. *Dropout Regulation*

Berdasarkan gambar diatas pada bagian a merupakan jaringan syaraf biasa yang memiliki dua *hidden layer.* Sedangkan pada bagian (b) merupakan jaringan syaraf dengan menggunakan *dropout.* Dari gambar tersebut terlihat terdapat beberapa neuron aktivasi yang tidak dipakai lagi. Penggunaan teknik ini sangat mudah diimplementasikan pada model CNN dan akan berdampak pada performa model dalam melatih serta mengurangi *overfitting.* (Srivastava, Hinton, G, & Kriszhevsky, A, 2014). Pada jaringan syaraf tiruan biasa, dimisalkan y^l adalah nilai keluaran dari suatu lapisan l dan z^l adalah nilai masukan pada *layer* l dengan W^l dan bl adalah bobot dan bias dari lapisan l, dengan unit ke i maka perhitungan proses *feedforward* menggunakan fungsi aktivasi f dapat dilakukan dengan persamaan (3.19).

$$Z_i^{l+1} = W_i^{(l+1)} y^l + b_i^{(l+1)}$$

$$y_i^{l+1} = f\left(z_i^{(l+1)}\right) \tag{3.19}$$

Sementara pada jaringan yang mengimplementasikan teknik *Dropout*, variable r^l melambangkan vector sepanjang j yang menyimpan nilai yang diperoleh dari distribusi Bernoulli. Proses *feedforward* dilakukan dengan persamaan (3.20).

$$y^{\sim l} = r_j^l * y^l$$
$$Z_i^{l+1} = W_i^{(l+1)} y^l + b_i^{(l+1)}$$
$$y_i^{l+1} = f\left(z_i^{(l+1)}\right) \quad (3.20)$$

3.7.5. *Softmax Classiefer*

Softmax Classiefer adalah generalisasi dari fungsi logistik. *Output* dari *softmax* ini dapat digunakan untuk mewakili distribusi sebuah katrgori. *Softmax function* digunakan dalam berbagai macam metode klasifikasi contohnya *multinomial logistic regression, multiclass linear discriminant analisys, naive Bayes classiefer,* dan *neural network.* Secara sepsifiknya fungsi ini biasa digunakan pada metode klasifikasi *multinomial logistic regression* dan *multiclass linear discriminant analisys.* Berikut adalah fungsi yang diberikan :

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (3.21)$$

Notasi fj menunjukkan hasil fungsi untuk setiap elemen ke-j pada vektor keluaran kelas. Argumen z adalah hipotesis yang diberikan oleh model pelatihan agar dapat diklasifikasi oleh fungsi *Softmax. Softmax* juga memberikan hasil yang lebih intuitif dan juga memiliki interpretasi probabilistik yang lebih baik dibanding algoritma klasifikasi lainya. *Softmax* memungkinkan kita untuk menghitung probabilitas untuk semua label. Dari label yang ada akan diambil sebuah vektor nilai bernilai riil dan merubahnya menjadi vektor dengan nilai antara nol dan satu yang bila semua dijumlah akan bernilai satu.

3.7.6. *Cross Entropy Loss Function*

Loss Function atau *Cost Function* merupakan fungsi yang menggambarkan kerugian yang terkait dengan semua kemungkinan yang dihasilkan oleh model. *Loss Function* bekerja ketika model pembelajaran memberikan kesalahan yang

harus diperhatikan. *Loss Function* yang baik adalah fungsi yang menghasilkan error yang diharapkan paling rendah.

Ketika suatu model memiliki kelas yang cukup banyak, perlu adanya cara untuk mengukur perbedaan antara probabilitas hasil hipotesis dan probabilitas kebenaran yang asli, dan selama pelatihan banyak algoritma yang dapat menyesuaikan parameter sehingga perbedaan ini diminimalkan. *Crossentropy* adalah pilihan yang masuk akal. Gambaran umum algoritma ini adalah meminimalkan kemungkinan log negatif dari dataset, yang merupakan ukuran langsung dari performa prediksi model. Berikut adalah fungsi yang diberikan :

$$\frac{1}{N}\sum_x(target(x) - activation(x))^2 = \frac{1}{N}\sum_x(target(x) - \max(0, \sum_i^{|x|} w_i x_i + b))^2 \quad (3.22)$$

3.7.7. Proses *forward propagation* pada CNN

Proses *forward propagation* pada jaringan CNN dilakukan untuk meneruskan nilai pada lapisan masukan hingga pada lapisan keluaran. Nilai ini diteruskan melalui lapisan konvolusi, subsampling dan lapisan *fully connected* sesuai dengan urutan lapisan tersebut ditempatkan pada jaringan yang digunakan. Maka dari itu perlu dilakukan perancangan bentuk struktur CNN yang akan digunakan terlebih dahulu. Urutan proses runut maju pada CNN dapat diringkas sebagai berikut :

1. Inisialisasi nilai awal pada *filter* pada lapisan konvolusi dan bobot pada lapisan *fully connected* dengan nilai acak, dan bias dengan nilai awal 0.
2. Melakukan proses konvolusi gambar masukan sesuai dengan *filter* pada lapisan konvolusi. Proses konvolusi dilakukan sesuai dengan persamaan (3.19) (Zhang, 2016) untuk menghasilkan *feature maps* ke *p* (C_p^1) dari *filter* ($k_{1,p}^1$) dan bias (b_p^1) yang dioperasikan pada gambar masukan (I). Tanda $*$ menotasikan proses konvolusi, dan $\sigma(x)$ menotasikan fungsi aktivasi.
   $$C_p^1 = \sigma\left(I * k_{1,p}^1 + b_p^1\right) \quad (3.23)$$
3. *Feature maps* yang didapatkan akan dikurangi ukurannya untuk mengurangi kompleksitas perhitungan pada lapisan selanjutnya. Proses ini dilakukan pada

lapisan subsampling. Proses subsampling dengan menggunakan *max pooling*, atau meloloskan nilai tertinggi dari *feature maps* yang ada dalam sebuah jendela subsampling.

4. Hasil dari lap1san subsampling merupakan *feature maps* yang telah direduksi ukurannya, jika pada struktur lapisan CNN yang digunakan terdapat lapisan konvolusi setelah lapisan subsampling, maka tahapan selanjutnya adalah sama dengan tahap 1-3, jika tidak maka lanjutkan ke tahap 5.
5. *Feature maps* yang didapat dari lapisan subsampling terakhir merupakan *feature maps* yang akan digunakan pada lapisan *fully connected* sebagai fitur untuk melakukan klasifikasi. *Feature maps* yang berupa matriks akan diuraikan menjadi vector yang panjang seperti pada Gambar 3.2. Proses ini disebut *vectorization and concatenation* (Zhang, 2016) yang dinotasikan pada persamaan (3.24). Fitur yang masuk ke dalam lapisan *fully connected* (f) merupakan hasil proses vektorisasi ($F(x)$) dari hasil subsampling pada lapisan sebelumnya (S_p^1), proses ini menggabungkan seluruh n buah *feature maps.*

   $$f = F\left(\{S_p^1\} p = 1,2,3, \dots n\right) \quad (3.24)$$

6. Selanjutnya adalah proses perhitungan prediksi target dari fitur yang masuk ke dalam lapisan *fully connected.* Nilai prediksi kelas ($y(i)$) ini dilakukan dengan melakukan perhitungan menggunakan persamaan (3.25). Perhitungan pada persamaan ini menggunakan fitur dari lapisan subsampling sebelumnya ($f(j)$) yang dikalikan dengan bobot yang terkoneksi ($W(i,j)$) dan ditambahkan dengan bias ($b(i)$).

   $$\hat{y}(i) = \sigma\ \left(\sum_{j=1}^{n} W(i,j) f(j) + b(i)\right) \quad (3.25)$$

7. Untuk mengetahui seberapa baik proses pembelajaran telah dilakukan, maka nilai *Loss* dihitung dengan persamaan (3.16)

3.7.8. Proses Propagasi Balik Pada CNN

Proses untuk Proses untuk memperbaharui nilai *filter* dan bobot pada jaringan adalah proes propagasi balik. Perhitungan perubahan nilai bobot dihitung dimulai dari lapisan *fully connected.* Pada lapisan ini perubahan bobot dicari dengan mencari derivatif *loss function* terhadap bobot (Zhang, 2016). Perhitungan

perubahan ($\Delta W(i,j)$) yang terhubung dengan node penghasil nilai fitur $f(j)$ berdasarkan selisih prediksi kelas dari data ke *i* ($\hat{y}(i)$) dengan target aktual dari data ke *i* ($y(i)$) pada lapisan *fully connected* dapat dilihat pada persamaan .

$$\Delta W(i,j) = \begin{cases} \big(\hat{y}(\mathrm{i}) - \mathrm{y}(1)\big).f(j), & if\ \hat{y}(\mathrm{i}) < 0 \\ 0, & otherwise \end{cases} \quad (3.26)$$

Perubahan bias ($\Delta b(i,j)$) juga dapat dilakukan dengan mencari derivatif *loss function* terhadap bias. Perubahan bias dapat dihitung menggunakan persamaan (3.27)

$$\Delta b(i) = \begin{cases} \big(\hat{y}(\mathrm{i}) - \mathrm{y}(1)\big), & if\ \hat{y}(\mathrm{i}) < 0 \\ 0, & otherwise \end{cases} \quad (3.27)$$

Selanjutnya adalah menghitung perubahan nilai *filter* pada lapisan konvolusi, perubahan ini diasarkan atas galat pada lapisan subsampling. Sehingga, sebelum menghitung perubahan bobot pada lapisan konvolusi, perlu dilakukan *upsampling* dari galat, karena setelah melakukan konvolusi *feature maps* melewati lapisan subsampling dan proses vektorisasi. Perhitungan perubahan *feature maps* (Δf) dilakukan dengan persamaan (3.28).

$$\Delta f = \begin{cases} \big(\hat{y}(\mathrm{i}) - \mathrm{y}(1)\big).W(i,j), & if\ \hat{y}(\mathrm{i}) < 0 \\ 0, & otherwise \end{cases} \quad (3.28)$$

Setelah didapat perubahan dari *feature maps* yang masih berbentuk vector panjang, maka dilakukan proses untuk membalikkan vector ini ke bentuk matriks 2 dimensi. Perubahan ini dapat dinotasikan pada persamaan (3.29)

$$\{S_p^1\}_{p=1,2,3,\dots n} = f^{-1}(\Delta f) \quad (3.29)$$

Proses *upsampling* adalah merubah matriks $\{\Delta S_p^1\}$ yang merupakan matriks hasil subsampling kembali ke ukuran awal sebelum dilakukan proses subsampling. Hal ini dilakukan dengan meneruskan nilai matriks $\{\Delta S_p^1\}$ kepada koordinat dari *feature maps* yang diloloskan nilainya pada proses subsampling (berkontribusi). Sedangkan untuk koordinat yang tidak diloloskan nilainya pada proses subsampling dapat diberi nilai 0. Penerusan nilai ini dinotasikan pada persamaan (3.30)

$$\Delta b(i) = \begin{cases} \Delta S_p^1\left(\left[\frac{i}{2}\right], \left[\frac{j}{2}\right]\right), & if\ C\,{}_p^1(i,j)\ contributed \\ 0, & otherwise \end{cases} \quad (3.30)$$

Setelah proses *upsampling*, maka $\Delta C_p^1(i,j)$ dapat digunakan untuk menghitung perubahan nilai pada *filter* konvolusi di lapisan sebelumnya. Pencarian perubahan nilai *filter* ($\Delta k_{1,p}^1$) dilakukan dengan melakukan konvolusi gambar masukan (I) dengan menggunakan ($\Delta C_{1,\sigma}^1$). Proses pencarian nilai perubahan nilai *filter* konvolusi dapat dinotasikan pada persamaan (3.31).

$$\Delta C_{1,\sigma}^1(i,j) = \begin{cases} \Delta C_p^1(i,j), & if\ C\,{}_p^1(i,j) > 0 \\ 0, & otherwise \end{cases}$$

$$\Delta k_{1,\mathrm{p}}^1 = I_{rot180} * \Delta C_{p,\sigma}^1 \quad (3.31)$$

Pada lapisan konvolusi juga terdapat bias, nilai bias juga diperbaharui untuk mendukung proses pembelajaran. Perhitungan perubahan nilai bias ($\Delta b_{1,p}^1$) dilakukan hampir sama dengan perhitungan perubahan nilai *filter* konvolusi, namun tidak melibatkan nilai masukan. Sehingga perubahan nilai bias sama dengan jumlah seluruh ($\Delta C_{1,\sigma}^1$) setelah *upsampling* seperti yang telah dinotasikan pada persamaan (3.32)

$$\Delta b_{1,\mathrm{p}}^1 = \sum_{i=1}^{row\ of\ C^1} \sum_{j=1}^{column\ of\ C^1} \Delta C_p^1(i,j) \quad (3.32)$$

Setelah menghitung perubahan pada tiap-tiap lapisan, maka proses memperbaharui nilai *filter*, bias pada lapisan konvolusi, bobot pada lapisan *fully connected*, serta bias yang lama dapat dilakukan sebagaimana dijabarkan pada persamaan (3.33), (3.34), (3.35) dan (3.36)

$$k_{1,\mathrm{p}}^1 = k_{1,\mathrm{p}}^1 - \alpha \cdot \Delta k_{1,\mathrm{p}}^1 \quad (3.33)$$

$$b_{1,\mathrm{p}}^1 = b_{1,\mathrm{p}}^1 - \alpha \cdot \Delta b_{1,\mathrm{p}}^1 \quad (3.34)$$

$$W = W - \alpha \cdot \Delta W \quad (3.35)$$

$$b = b - \alpha \cdot \Delta b \quad (3.36)$$

Proses ini dilakukan hingga kondisi terhenti ditemukan, kondisi terhenti ini bisa saja berupa epoch maksimum yang tercapai atau nilai *loss* yang berada

dibawah batasan yang ditetapkan. Proses perubahan nilai bobot, bias dan *filter* dilakukan setiap satu data masuk ke dalam jaringan

3.7.9. *Consufion Matriks*

Penentuan baik atau tidaknya performa suatu model klasifikasi dapat dilihat dari parameter pengukuran performanya, yaitu tingkat akurasi, *recall,* dan presisi. Untuk menghitung faktor-faktor tersebut diperlukan sebuah matrik yang biasa disebut *confusion matriks.* Salah satu *Confusion-matrix* yang kerap digunakan dalam pengukuran dapat dilihat pada Gambar 3.17 (Fawcett, 2006).

		Kejadian Sebenarnya	
		P	N
Hipotesis Kejadian	P	True Positive	False Positive
	N	False Negative	True Negative

Gambar 3.18. *Confosuion Matriks*

Berdasarkan gambar di atas terdapat beberapa nilai didalam matriks yaitu "*True Positive*" (*TP*), "*True Negative*" (*TN*), "*False Positive*" (*FP*), dan "*False Negative*" (*FN*), seluruh kemungkinan kejadian sebenarnya positif (*P*) dan seluruh kemungkinan kejadian sebenarnya negatif (*N*). Nilai tersebut dapat digunakan untuk menghitung akurasi dengan persamaan (3.37)

$$Akurasi = \frac{TP+TN}{P+N} \tag{3.37}$$

Akurasi digunakan sebagai parameter sebagaimana akurat suatu model melakukan klasifikasi. Sementara untuk menghitung tingkat presisi prediksi kejadian dapat digunakan persamaan (3.38).

$$Presisi\ prediksi = \frac{TP}{TP+FP} \tag{3.38}$$

Presisi menggambarkan seberapa tepat suatu model memprediksi kejadian positif dalam serangkaian kegiatan prediksi. Perhitungan presisi biasanya

bermanfaat pada pengembangan model prediksi hujan di suatu daerah. Selain presisi dan akurasi, untuk dapat melihat lebih detail lagi kinerja suatau sistem, *recall* atau sensitifitas sistem terhadap suatu kelas juga dapat dilihat. *Recall* dapat dihitung dengan menggunakan persamaan (3.39).

$$Sensitifitas\ prediksi = \frac{TP}{TP+FN} \tag{3.39}$$

BAB IV
METODOLOGI PENELITIAN

4.1. Populasi dan Sampel

Populasi dalam penelitian ini adalah citra tokoh-tokoh wayang penggaris yang diambil dari situs pencarian *google.* Sedangkan sampel yang digunakan dalam penelitian ini adalah tiga karakter wayang yaitu Cepot, Gatotkaca, dan Semar. Total citra yang dikumpulkan untuk sampel sebanyak 300, dengan masing-masing kategori jenis wayang sebanyak 100 citra wayang golek.

4.2. Variabel dan Definisi Operasional Variabel

Variabel yang digunakan dalam penelitian ini ditampilkan dalam Tabel 4.1 tentang penjelasan dan definisi operasional penelitian:

Tabel 4.1. Definisi Operasional Variabel

Variabel	Definisi Operasional Variabel
Cepot	Citra berupa wayang golek Cepot
Gatotkaca	Citra berupa wayang golek Gatotkaca
Semar	Citra berupa wayang golek Semar

4.3. Jenis dan Sumber Data

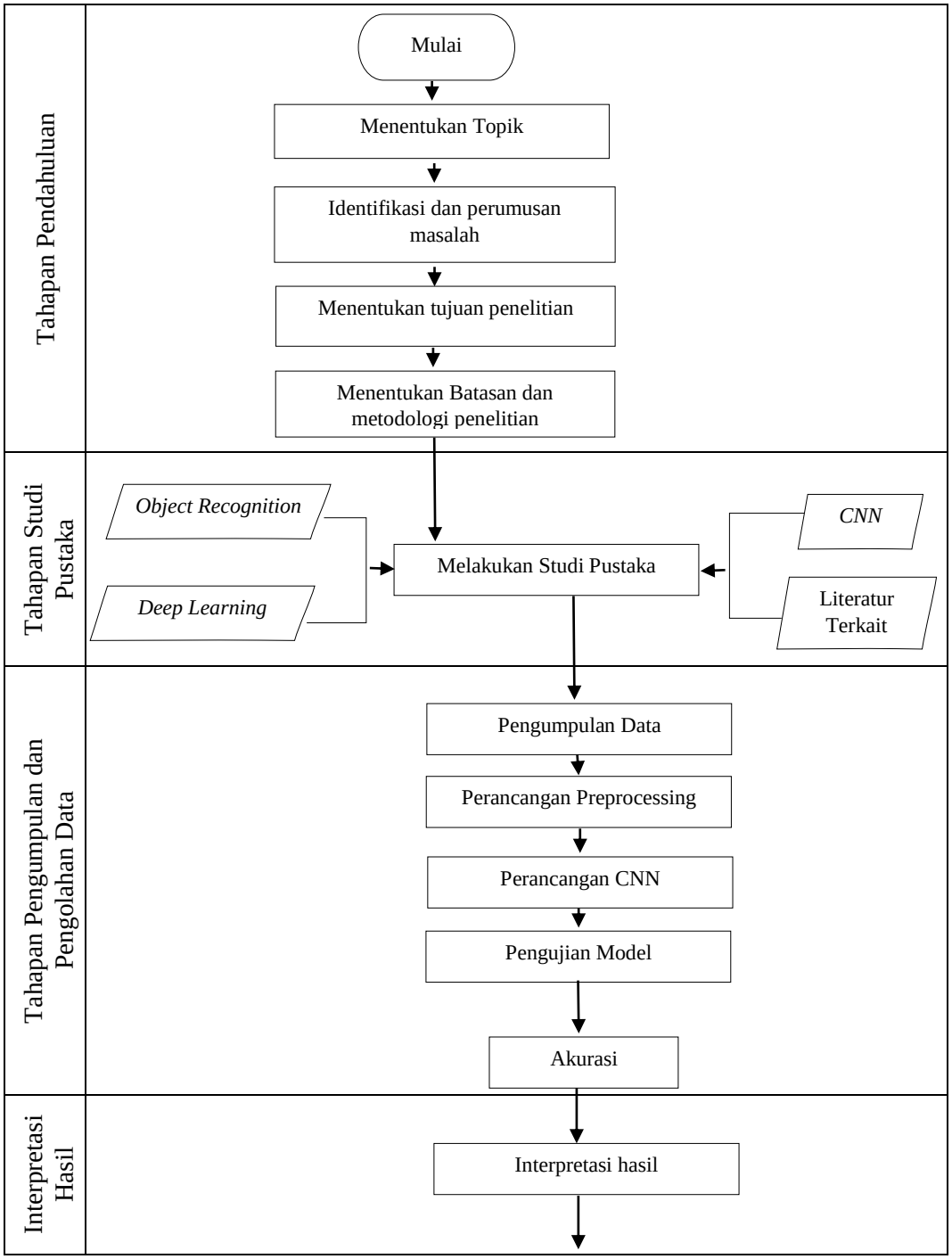
Jenis data yang digunakan dalam penelitian ini adalah data primer. Data tersebut diperoleh dengan cara *crawling* citra tokoh-tokoh wayang pada *search engine google.*

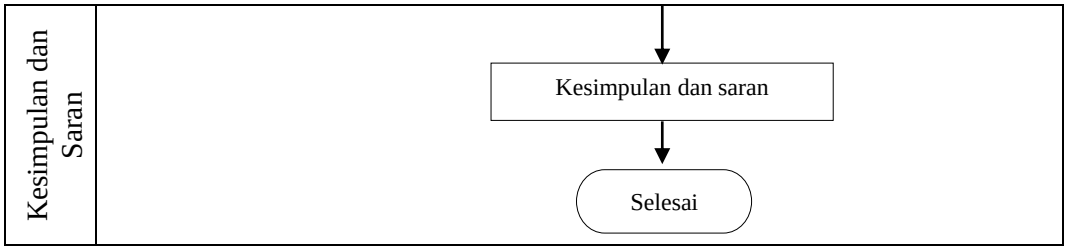
4.4. Metode Analisis Data

Software yang digunakan pada penelitian ini adalah *software* Python 3.6.2. Metode analisis data yang digunakan dalam penelitian ini adalah metode *Convolutional Neural Network* yang bertujuan untuk mengklasifikasikan citra wayang golek yaitu Cepot, Gatotkaca, dan Semar.

4.5. Tahapan Penelitian

Langkah atau tahapan yang dilakukan pada penelitian ini digambarkan melalui Gambar 4.1 berikut ini :





Gambar 4.1. Tahapan Penelitian

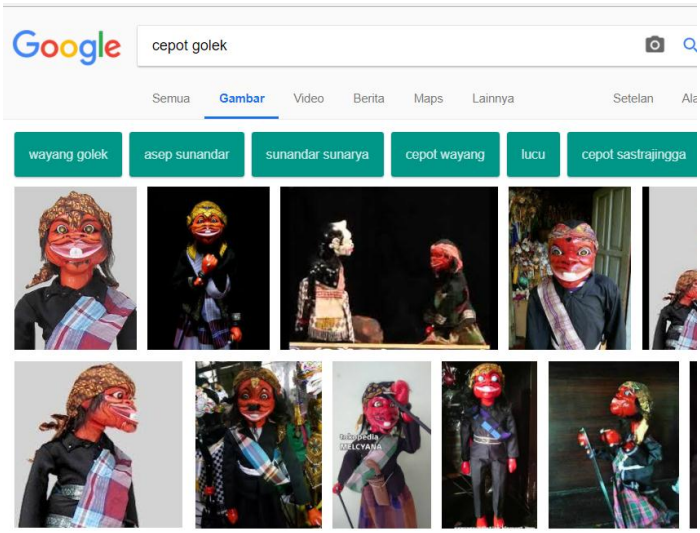
4.6. Rancangan *Dataset*

Penggunaan dataset pada metode CNN yaitu berupa data gambar. Model CNN akan berjalan dengan baik ketika menggunakan data train gambar yang banyak. Sehingga sebuah model dapat belajar mengenali gambar tersebut. Dataset yang digunakan dalam penelitian ini berupa gambar yang dikumpulkan melalui *search engine google*. Data gambar yang digunakan kali ini adalah gambar tokoh-tokoh wayang seperti Cepot, Gatotkaca, dan Semar. Pengumpulan dataset jika dilakukan secara manual akan memakan waktu yang cukup lama. Sehingga peneliti menggunakan metode *crawling* gambar dengan menggunakan program *javascript* dan python. Program *javascript* bertujuan untuk pengambilan *URL* gambar yang terdapat pada *google* dan program python yang melakukan eksekusi untuk men*download* gambar tersebut.

4.7. Program *Javascript*

Program ini digunakan untuk mengambil *URL* pada *google image* dengan menggunakan *library javascriptscript* yaitu *jQuery*. *jQuery* adalah perpustakaan *Javascript* yang cepat, kecil, dan kaya fitur. *jQuery* membuat hal-hal seperti *traversal* dan manipulasi dokumen HTML, penanganan *event*, animasi, dan *Ajax* lebih sederhana dengan API (*application programming interface)* yang mudah digunakan yang bekerja di banyak *browser*. Adapun langkah yang dijalankan sebagai berikut :

1. Langkah pertama adalah masuk ke *google* dan ketik nama gambar yang diinginkan. Penlelitian ini menggunakan contoh gambar Cepot.



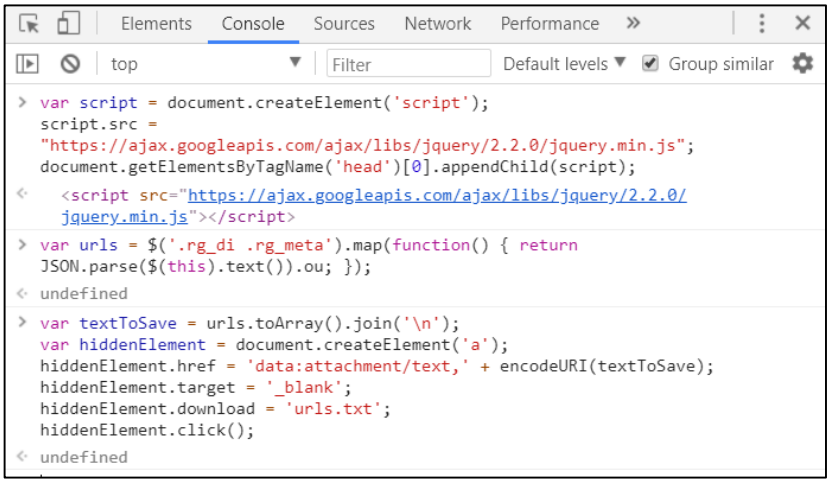
Gambar 4.2. *Google Image Cepot*

2. Langkah kedua gunakan *javascripts* untuk mengumpulkan *URL* gambar, namum sebelumnya peneliti akan menentukan batas *URL* terlebih dahulu yang akan di *download.*porgram *javascripts* diinputkan pada *console* yang terdapat pada developer *tools google chrome*, dengan cara klik kanan inspect.
3. Masukan URL berikut pada developer *tools google chrome* satu persatu

```
var textToSave = urls.toArray().join('\n');
var hiddenElement = document.createElement('a');
hiddenElement.href = 'data:attachment/text,' +
enkodeURI(textToSave);
hiddenElement.target = '_blank';
hiddenElement.download = 'urls.txt';
```

Gambar 4.3. *Google Image Cepot*

Berdaarkan kode diatas, file akan diolah pada program *python* maka semua URL yang didapatkan akan disimpan kedalam satu file yaitu *.txt.* Seperti yang sudah dijelaskan sebelumnya masukan kode tersebut kedalam developer *tools google chrome.*



Gambar 4.4. *Output Hidden Element*

Berdasarkan gambar 4.3 menunjukan bahwa semua perintah untuk menyimpan hasil download dengan javascripts dengan *extention .txt*. Terlihat bahwa pada baris terakhir pada gambar perintah untuk mendownload file dalam bentuk *.txt*.

4.8. Program *Python*

Setelah melakukan proses pembuatan data dalam bentuk *.txt*, kemudian untuk melakukan proses download gambar kali ini menggunakan program python. Berikut adalah kode yang dimasukan kedalam program *python* :

Tabel 4.2. *Import Packages*

```
1 # import the necessary packages
2 from imutils import paths
3 import argparse
4 import requests
5 import cv2
6 import os
```

Tabel 4.1 menunjukan beberapa packages yang harus di gunakan di dalam program python . Berikut adalah fungsi dari packages diatas :

1. *Packages Argparse* : membuat argument menulis inputan kode.
2. *Packages Requests* : *download* gambar pada URL yang sudah dibuat.
3. *Packages cv2* : membaca lokasi *output* gambar .
4. *Packages os* : menyimpan gambar pada *folder output* yang sudah di tentukan
5. *Packages Imutils* : Melakukan perulangan pada saat list file gambar pada folder.

Tabel 4.3 Membuat *Argument*

```
8  # arguments
9  ap = argparse.ArgumentParser()
10 ap.add_argument("-u", "--urls", required=True,
11       help="path to file containing image URLs")
12 ap.add_argument("-o", "--output", required=True,
13       help="path to output directory of images")
14 args = vars(ap.parse_args())
15
16 # list URL dari input data kemudian di inisialisasi untuk
17 menentukan
18 # total gambar
19 rows = open(args["urls"]).read().strip().split("\n")
   total = 0
```

Berdasarkan table 4.2 berfungsi untuk mengurai argumen baris perintah dan memuat *urls* dari *disk* ke memori. Urutan baris perintah *parsing* dilakukan pada baris 9-14 disini menggunakan dua *parsing*:

1. --urls : *Path* dari *file* yang berisi *urls* gambar yang dihasilkan oleh *Javascript* di atas.
2. --output : *Path* dari *output* untuk menyimpan gambar yang di*download* dari *Google Images.*

Pada baris 18 digunakan untuk memuat setiap *urls* dari *file* ke dalam daftar, kemudian juga menginisialisasi sebuah *counter*, total, untuk menghitung *file* yang telah kami *download.* Pada baris 18 terlihat proses yaitu membuka *parsing urls* kemudian membaca *file* tersebut (.*read*()). Selanjutnya mengembalikan salinan *string* dengan karakter terdepan dan *trailing* yang dihapus (berdasarkan argumen *string* yang dilewati) (.strip()). Kemudian menghapus "\n" pada data *urls*

Tabel 4.4 *Perulangan Download URL*

```
21 # Perulangan URLs
22 for url in rows:
23       try:
24             # download image
25             r = requests.get(url, timeout=60)
26
27             # save image
28             p = os.path.sep.join([args["output"],
29 "{}.jpg".format(
30                   str(total).zfill(8))])
31             f = open(p, "wb")
32             f.write(r.content)
33             f.close()
34
35             # update the counter
36             print("[INFO] downloaded: {}".format(p))
```

```
37              total += 1
38
39        # jika terdapat kesalahan dalam proses download
40        except:
                print("[INFO] error downloading
{}...skipping".format(p))
```

Berdasarkan tabel diatas dengan menggunakan *requests*, disini hanya perlu menentukan *urls* dan *timeout* untuk *download*. Peneliti mencoba men*download file* gambar ke dalam variabel, r, yang menampung *file* biner (bersama dengan *header HTTP*, dll.) dalam memori sementara atau biasa disebut RAM (*Random Access Memory*) (Kode baris 25). Selanjutnya menyimpan gambar ke *disk*, hal pertama yang diperlukan adalah *path* dan nama *file* yang *valid*. Kode baris 28-29 menghasilkan *path* + *filename*, p, yang akan menghitung secara bertahap dari 00000000.jpg. Peneliti kemudian membuat sebuah *file pointer*, f, menentukan *path output*, p, dan menunjukkan peneliti ingin menulis *mode* dalam format biner ("wb") pada kode baris 30. Selanjutnya, menulis isi *file* dari f *(r.content)* dan kemudian menututup *file* (Kode baris 31 dan 32). Akhirnya memperbarui jumlah total gambar yang diunduh pada kode baris 35 dan 36). Jika ada kesalahan yang ditemukan di sepanjang proses pengunduhan, maka sebuah pesan dicetak ke terminal (Kode baris 39 dan 40. Peneliti akan mengulang semua file yang baru saja di*download* dan mencoba membukanya dengan *OpenCV*. Jika *file* tidak bisa dibuka dengan *OpenCV*, maka akan dihapus dan dilanjutkan. Ini tercakup dalam *blok kode* berikut :

Tabel 4.5 *Perulangan Load Image*

```
42 # perulangan image path
43 for imagePath in paths.list_images(args["output"]):
44        # inisialisasi jika gambar di hapus atau tidak
45        delete = False
46
47        # load image
48        try:
49               image = cv2.imread(imagePath)
50
51               # jika gambar tidak ada dari disk
52               # maka akan di hapus
53               if image is None:
54                      delete = True
55
56        # jika opencv tidak bisa meload gambar
57        # seperti gambar tersebut corrupt maka akan dihapus
58        except:
59               print("Except")
```

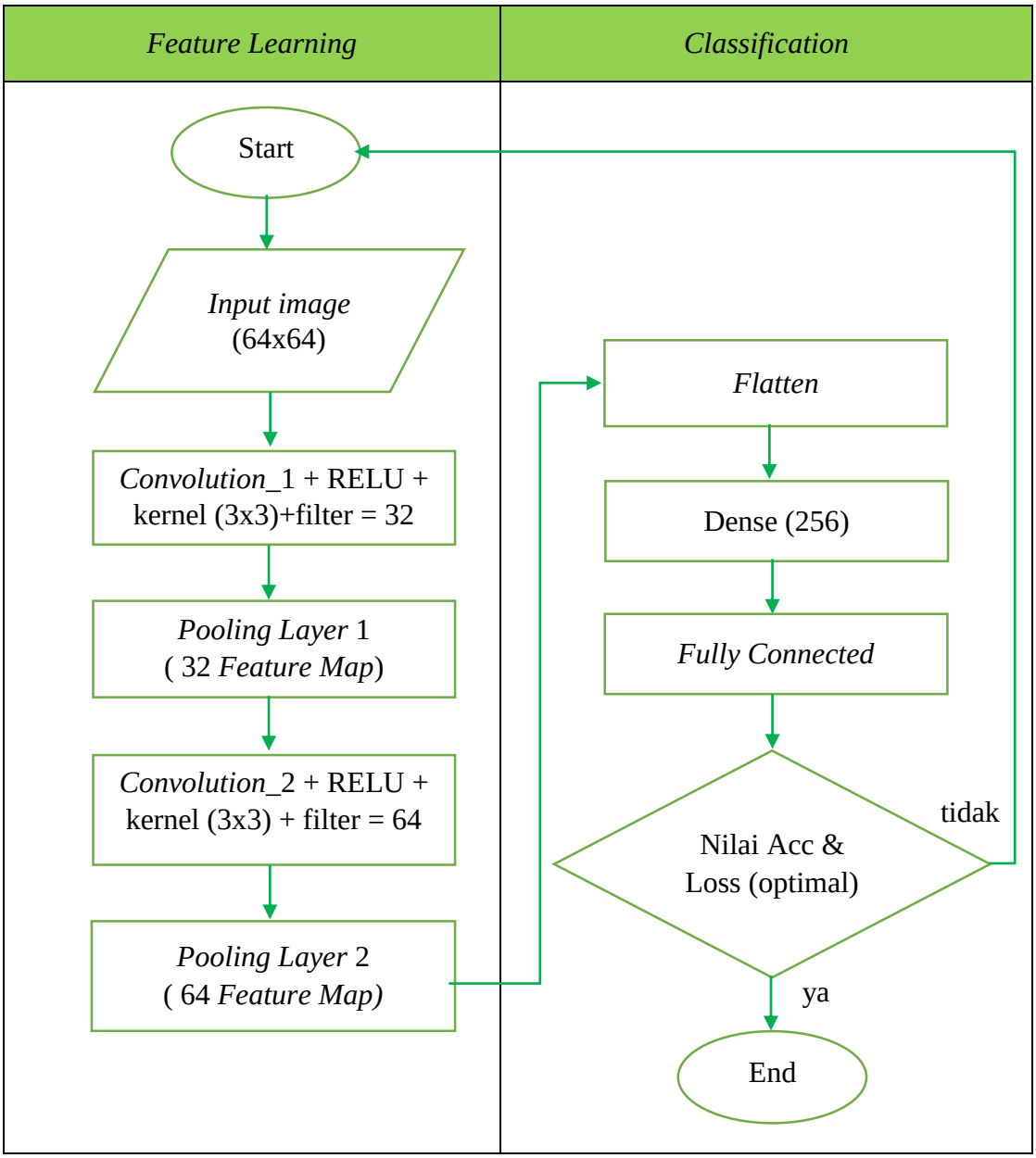
```
60              delete = True
61
62      # check image yang di delete
63      if delete:
64              print("[INFO] deleting {}".format(imagePath))
65              os.remove(imagePath)
```

Saat melakukan perrulangan setiap *file*, akan diinisialisasi *delete* ke *False* (Kode baris 45). Kemudian peneliti akan mencoba memuat *file* gambar di Kode baris 49. Jika gambar dimuat sebagai *None*, atau jika ada pengecualian, maka akan menyetel *delete = True* (Kode baris 53-54 dan kode baris 58-60). Alasan umum gambar tidak dapat dimuat yaitu kesalahan selama unduhan (seperti *file* yang tidak diunduh sepenuhnya), gambar rusak, atau format *file* gambar yang tidak dapat dibaca *OpenCV*. Terakhir jika *flag delete* diset, kita panggil *os.remove* untuk menghapus gambar pada kode baris 63-65.

4.9. Rancangan *Convolutional Neural Network* (CNN)

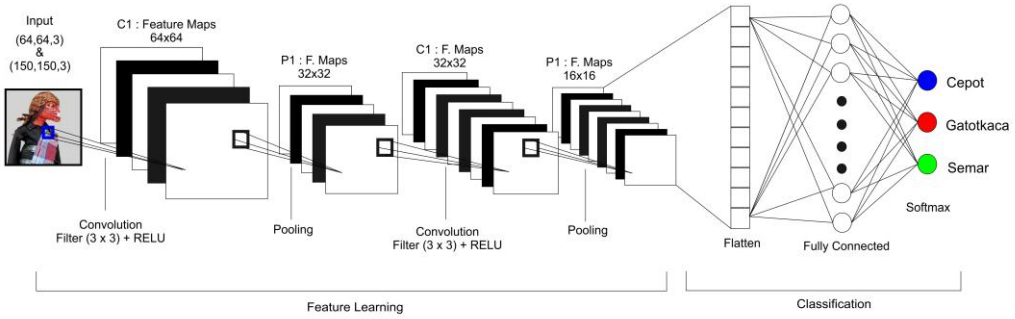
Setelah dilakukan pembuatan data, langkah selanjutnya adalah melakukan pelatihan model CNN. Umumnya dalam CNN memiliki 2 tahapan, yaitu tahap *feature learning* dan *classification.* Input gambar pada model CNN menggunakan citra yang berukuran 64x64x3. Angka tiga yang dimaksud adalah sebuah citra yang memiliki 3 *channel* yaitu *Red*, *Green*, dan *Blue* (RGB) Citra masukan kemudian akan diproses terlebih dahulu melalui proses konvolusi dan proses pooling pada tahapan *feature learning*. Jumlah proses konvolusi pada rancangan ini memiliki dua lapisan konvolusi. Setiap konvolusi memiliki jumlah filter dan ukuran kernel yang berbeda. Kemudian dilakukan proses *flatten* atau proses mengubah *feature map* hasil *pooling layer* kedalam bentuk vector. Proses ini biasa disebut dengan tahap *fully Connected layer*. Berikut adalah rancangan dari arsitektur CNN pada penelitian ini :

Tabel 4.6 *Flow Chart Model*



Berdasakan tabel diatas dijelaskan terdapat dua tahap dalam arsitektur CNN, yaitu *Feature Learning* dan *classification. Feature learning* adalah teknik yang memungkinkan sebuah *system* berjalan secara otomatis untuk menentukan representasi dari sebuah *image* menjadi features yang berupa angka-angka yang merepresentasikan image tersebut. Tahap *Classification* adalah sebuah tahap dimana hasil dari *feature learning* akan digunakan untuk proses klasifikasi

berdasarkan *subclass* yang sudah ditentukan. Jika flow chart diatas diubah kedalam bentuk gambar, maka dapat dilihat seperti gambar 4.5 berikut :



Gambar 4.5. Rancangan Arsitektur CNN

Pada konvolusi pertama menggunakan jumlah filter sebanyak 32 dan kernel dengan matriks 3x3. Kemudian dilakukan proses *pooling* menggunakan ukuran *pooling* 2x2 dengan pergeseran mask sebanyak dua langkah. Kemudian pada tahapan konvolusi kedua dengan menggunakan jumlah filter sebanyak 64 dan kernel dengan matriks 2x2. Kemudian di lanjutkan dengan *flatten* yaitu merubah *output* dari proses konvolusi yang berupa matriks menjadi sebuah vector yang selanjutnya akan diteruskan pada proses klasifikasi dengan menggunakan MLP (*Multi Layer Perceptron*) dengan jumlah neuron pada lapisan tersembunyi yang telah ditentukan. Kelas dari citra kemudian diklasifikasikan berdasarkan nilai dari neuron pada lapisan tersembunyi dengan menggunakan fungsi aktivasi *softmax.*

4.10. Rancangan Pengujian

Pengujian ini dilakukan ntuk melakukan evaluasi terhadap model yang dihasilkan oleh CNN. Pengujian ini dilakukan dua tahap yaitu tahapan training dan *testing*. Tahap training adalah tahap dimana model CNN diuji dengan data latih yang sudah disediakan. Jumlah data latih yang disediakan sebanyak 300 data gambar, dengan jumlah gambar perkelas sebanyak 100 gambar. Data *training* di bagi kembali menjadi dua yaitu training dan validasi, yaitu sebanyak 240 training dan 60 validasi. Tahap *Testing* adalah tahap pengujian model yang sudah dilakukan tahap pelatihan. Jumlah data latih dalam penelitian ini sebanyak 60 data gambar,

dengan jumlah gambar perkelas sebanyak 20 gambar. Pada tahap ini model di uji dengan gambar yang berbeda dengan tujuan menguji apakah model sudah menghasilkan performa yang baik dalam mengklasifikasikan sebuah gambar.

4.11. Perangkat Pengujian

Pengujian dilakukan pada laptop dengan spesifikasi sebagai berikut :

1. Inter core i7-6700HQ
2. 16 GB RAM
3. GPU : NVIDIA GeForce GTX 960
4. Sistem operasi Windows
5. Bahasa Pemrograman python 3.6

4.12. Pelatihan Model

Pelatihan model dijalankan dengan membuat directory dengan nama folder "data" terlebih dahulu yang bertujuan untuk penyimpanan data. Dalam directory tersebut dibuat dua folder yaitu data train dan validation.

```
train_data_path = './data/train'
validation_data_path = './data/validation'
```

Gambar 4.6. Rancangan Arsitektur CNN

Berdasarkan kedua *directory* tersebut adalah penyimpanan dataset gambar wayang. Setiap *directory* tersebut dibuatkan directory untuk untuk setiap kelas wayang. *Directory* yang dibuat untuk penyimpanan wayang Cepot, Gatotkaca, dan Semar. Setelah pembuatan *directory* selanjutnya adalah penentuan parameter dari model CNN.

```
"""
Parameters
"""
img_width, img_height = 64 , 64
batch_size = 30
samples_per_epoch = 240
validation_steps = 60
nb_filters1 = 32
nb_filters2 = 64
conv1_size = 3
conv2_size = 3
pool_size = 2
classes_num = 3
lr = 0.001
```

Gambar 4.7. Penentuan Parameter

Jika dilihat dari gambar 4.7 merupakan penentuan awal dari beberapa parameter yang dibutuhkan didalam model CNN. Input gambar pada model ini adalah 64x64 dan 150x150 . Untuk *batch sizenya* berukuran 32. *Batch size* adalah jumlah sampel yang disebarkan ke dalam arsitektur *neural network*. *Sample per epoch* adalah jumlah sampel yang digunakan dalam tahap pelatihan. Jumlah sampel yang digunakan sebanyak 240 data. *Validation step* adalah jumlah data validasi dibagi dengan nilai *batch size*. Jumlah data validasi yaitu 60 gambar. Kemudian terdapat *number of filter* yang dimasukkan kedalam proses konvolusi nya. Pada tahap konvolusi pertama digunakan jumlah filter sebanyak 32 dan pada konvolusi kedua digunakan jumlah filter sebanyak 64. Kemudian untuk ukuran kernel nya menggunakan dua ukuran kernel yaitu 3x3 dan 5x5. Kernel adalah sebuah matriks untuk menghitung dan mendeteksi suatu pola yang digunakan pada saat proses *convolution. Pooling sizenya* diberikan nilai 2. *Pooling* adalah Proses Mengurangi dimensi dari *feature map* (*downsampling*). Penelitian ini menggunakan 3 kelas gambar wayang. Kemudian *learning rate* nya dengan membandingkan nilai learning rate 0.001 dan 0.0001.

```
44  model = Sequential()
45  model.add(Conv2D(nb_filters1,conv1_size, conv1_size),
46            padding ="same", input_shape=(img_width, img_height, 3)))
47  model.add(Activation("relu"))
48  model.add(MaxPooling2D(pool_size=(pool_size, pool_size)))
49
50  model.add(Conv2D(nb_filters2, (conv2_size, conv2_size), padding ="same"))
51  model.add(Activation("relu"))
52  model.add(MaxPooling2D(pool_size=(pool_size, pool_size), dim_ordering='th'))
53
54  model.add(Flatten())
55  model.add(Dense(256))
56  model.add(Activation("relu"))
57  model.add(Dropout(0.5))
58  model.add(Dense(classes_num, activation='softmax'))
59
60  model.compile(loss='categorical_crossentropy',
61                optimizer=optimizers.RMSprop(lr=lr),
62                metrics=['accuracy'])
```

Gambar 4.8. Arsitekture CNN

Berdasarkan gambar 4.8 menunjukan arsitektur dari model CNN. Penelitian ini menggunakan model CNN dengan 2 proses konvolusi dan 2 proses *pooling layer* . pada masing masing proses konvolusi digunakan aktivasi fungsi RELU. Aktivasi fungsi ini bertujuan mengubah nilai minus pada sebuah matriks dari hasil proses konvolusi. Aktivasi RELU melakukan "*treshold*" dari 0 hingga *infinity*.

Kemudian pada proses konvolusi digunakan *zero padding*. *Zero padding* adalah Parameter jumlah piksel yang berisi nilai nol yang ditambahkan disetiap sisi input. Setelah melalui proses konvolusi hasil akhir dari max pooling akan diubah kedalam bentuk *vector* dua dimensi. Pada baris 55 terdapat layer *dense* dengan jumlah *vector* 256 dengan menggunakan aktifasi fungsi RELU. Nilai *Dropout* yang digunakan yaitu 0.5 dan menggunakan fungsi aktivasi *sofmax*. Fungsi aktivasi ini bertujuan untuk mengklasifikasi kedalam banyak kelas. Kemudian untuk *loss function* nya menggunakan optimasi Adam.

```
63
64  train_datagen = ImageDataGenerator(
65      rescale=1./255,
66      shear_range=0.2,
67      zoom_range=0.2,
68      horizontal_flip=True)
69
70   test_datagen = ImageDataGenerator(rescale=1./255)
71
72  train_generator = train_datagen.flow_from_directory(
73      train_data_path,
74      target_size=(img_height, img_width),
75      batch_size=batch_size,
76      class_mode='categorical')
77
78  validation_generator = test_datagen.flow_from_directory(
79      validation_data_path,
80      target_size=(img_height, img_width),
81      batch_size=batch_size,
82      class_mode='categorical')
83
```

Gambar 4.9. Augumentasi Data

Berdasarkan gambar 4.9 merupakan proses augumentasi data/gambar. Proses ini biasa disebut preprocessing dan pembangkitan data. Script pada baris ke 64-68 ini berguna untuk *merescale* data gambar sebelum melakukan pelatihan. Rescale 1./255 adalah untuk mengubah setiap nilai piksel dari jangkauan [0,255] -> [0,1]. Kemudian untuk nilai *Share* dan *zoom* ini digunakan untuk merotasi kea rah berlawanan dengan arah jarum jam dan juga memperbesar gambar ketika proses membangkitkan data. *Train generator* dan *validation generator* digunakan untuk proses membangkitkan data berdasarkan data *train* dan validasi . JIka penentuan Batch size sebanyak 30 maka ketika proses *training data*, akan diambil sebanyak 30 data secara random dari semua *sample dataset* untuk setiap *epoch* hingga semua *epoch* memenuhi batas samplenya. Kemudian untuk melihat grafik hasil proses training data maka dapat digunakan script sebagai berikut :

```
"""
Tensorboard log
"""
log_dir = './tf-log/tf-log(epoch=30,lr=0.001,Op=adam)/'
tb_cb = callbacks.TensorBoard(log_dir=log_dir, histogram_freq=0)
cbks = [tb_cb]

model.fit_generator(
    train_generator,
    samples_per_epoch=samples_per_epoch,
    epochs=epochs,
    validation_data=validation_generator,
    callbacks=cbks,
    validation_steps=validation_steps)

target_dir = './models/model(epoch=30,lr=0.001,Op=adam)/'
if not os.path.exists(target_dir):
    os.mkdir(target_dir)
model.save('./models/model(epoch=30,lr=0.001,Op=adam)/model.h5')
model.save_weights('./models/model(epoch=30,lr=0.001,Op=adam)/weights.h5')
```

Gambar 4.10. Grafik dan *Save Model*

Berdasarkan gambar 4.10 digunakan untuk memanggil grafik hasil dari proses *training* dapat di gunakan *Tensorboard. Tensorboard* adalah sebuah *visualize tools* yang disediakan oleh *tensorflow*, dengan bantuan *tools* ini dapat mempermudah dalam melihat tingkat *accuracy* dan loss model dari data *train* dan *validation.* Model fit generator digunakan untuk membangkitkan data untuk setiap *epoch* sampai semua *epoch* memenuhi jumlah sampelnya. *Save* model dalam sebuah directory dan gunakan model ini untuk mengklasifikasi dengan databaru.

4.13. Pengujian Model

Algoritma *Convolutional Neural Network* membutuhkan proses training dan *testing*. Proses *training* ini bertujuan untuk melatih algoritma CNN dalam mengenali datasetnya dan membentuk sebuah model berdasarkan pelatihan tersebut. Proses *testing* betujuan menguji sebuah model yang dibentuk pada saat proses *training*. Berikut ini adalah proses testing pada penelitian ini.

```
img_width, img_height = 64, 64
model_path = './models/model(epoch=50,lr=0.001,Op=adam)/model.h5'
model_weights_path = './models/model(epoch=50,lr=0.001,Op=adam)/weights.h5'
model = load_model(model_path)
model.load_weights(model_weights_path)
```

Gambar 4.11. *Callback* Model

Step pertama yang dilakukan adalah memanggil model yang sudah dibentuk sebelumnya pada saat *training data.* Input image harus disamakan dengan input pada proses *train*. Jika tidak sama maka akan terjadi kesalahan pada program dan model tidak akan membaca gambar tersebut.

```
11
12  def predict(file):
13    x = load_img(file, target_size=(img_width,img_height))
14    x = img_to_array(x)
15    x = np.expand_dims(x, axis=0)
16    array = model.predict(x)
17    result = array[0]
18    answer = np.argmax(result)
19    if answer == 0:
20      print("Label: cepot")
21    elif answer == 1:
22      print("Label: gatotkaca")
23    elif answer == 2:
24      print("Label: semar")
25
26    return answer
```

Gambar 4.12. *Predict Image*

Gambar 4.12 merupakan proses *predict image* . Terdapat tiga label pada pada baris ke 20-24. Label pertama adalah hasil prediksi wayang cepot dengan nilai 0, label kedua adalah hasil prediksi wayang gatotkaca dengan nilai prediksi 1, kemudian yang terakhir adalah hasil predksi wayang dnegan nilai prediksi 2.

```
35  for i, ret in enumerate(os.walk('./data/test/cepot')):
36    for i, filename in enumerate(ret[2]):
37      if filename.startswith("."):
38        continue
39      print("Label: Cepot")
40      result = predict(ret[0] + '/' + filename)
41      if result == 0:
42        cepot_t += 1
43      else:
44        cepot_f += 1
45
46  for i, ret in enumerate(os.walk('./data/test/gatotkaca')):
47    for i, filename in enumerate(ret[2]):
48      if filename.startswith("."):
49        continue
50      print("Label: Gatotkaca")
51      result = predict(ret[0] + '/' + filename)
52      if result == 1:
53        gatotkaca_t += 1
54      else:
55        gatotkaca_f += 1
56
57  for i, ret in enumerate(os.walk('./data/test/semar')):
58    for i, filename in enumerate(ret[2]):
59      if filename.startswith("."):
60        continue
61      print("Label: Semar")
62      result = predict(ret[0] + '/' + filename)
63      if result == 2:
64        semar_t += 1
65      else:
66        semar_f += 1
67
```

Gambar 4.13. *Looping Image*

Gambr 4.13 menunjukan proses *looping image*. Proses ini bertujuan untuk membaca semua image dari ketiga *directory* yang ada. Baris 35-44 ini adalah proses

looping untuk satu kelas/jenis wayang, wayang tersebut adalah Cepot. Jika hasil prediksi menunjukan nilai 0 maka hasil klasifikasi menunjukan gambar wayang cepot dan lainnya bukan Cepot. Begitupun seterusnya untuk wayang Gatotkaca dan wayang Semar. Hasil akhir dari prediksi ini berupa table matriks atau kontigensi.

```
"""
Check metrics
"""
print("True Cepot: ", cepot_t)
print("False Cepot: ", cepot_f)
print("True Gatotkaca: ", gatotkaca_t)
print("False Gatotkaca: ", gatotkaca_f)
print("True Semar: ", semar_t)
print("False Semar: ", semar_f)
```

Gambar 4.14. *Check Matrics*

Hasil dari prediksi ini termuat dalam sebuah matriks atau table kontigensi. Jika dijasikan kedalam table maka akan seperti berikut :

Tabel 4.7. *Matrics Predict*

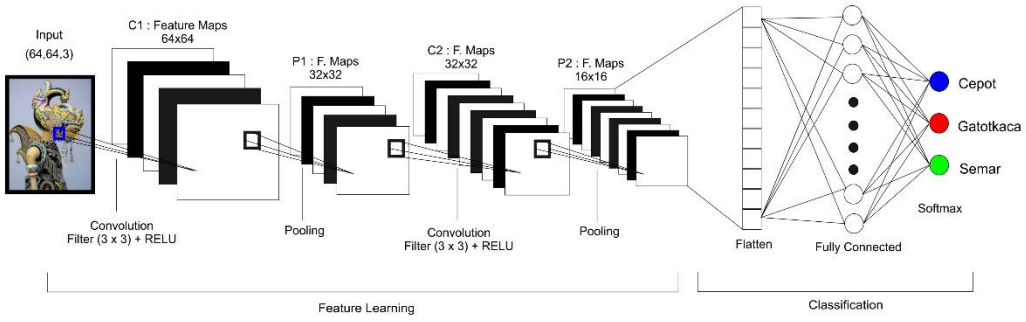
Matrics		Predict Class		
		Cepot	Gatotkaca	Semar
Actual Class	Cepot	True Positive		
	Gatotkaca		True Positive	
	Semar			True Positive

BAB V
HASIL DAN PEMBAHASAN

Pada penelitian ini, peneliti melakukan klasifikasi tiga kelas gambar wayang golek, yaitu Cepot, Gatotkaca dan Semar dengan menggunakan algoritma *Convolutional Neural Network* (CNN). Proses utama dalam pembuatan model ini diawali dengan proses *training data*. Proses ini bertujuan untuk pembentukan model yang akan digunakan untuk pengujian data *testing.* Parameter untuk mengukur tingkat keberhasilan model adalah nilai akurasi. Nilai akurasi model dapat ditentukan dengan melakukan pengujian menggunakan data *testing*. Proses *training* menggunakan *packages Keras* pada *python* dengan *back-end tensorflow*. Keras merupakan salah satu modul yang dibuat oleh *Google* untuk mempermudah dalam *research* mengenai *neural network* dan mampu berjalan diatas *tensorflow*, *theano*, *MXNet.*

5.1. Arsitektur Jaringan

Dalam algoritma *Convolutional Neural Network* (CNN) pembentukan arsitektur jaringan dapat mempengaruhi hasil dari akurasi model.



Gambar 5.1 Arsitektur Jaringan

Gambar 5.1 merupakan arsitektur jaringan pada proses *training* untuk menghasilkan model yang optimal. Penelitian ini menggunakan input gambar dengan ukuran 64x64x3, tujuannya adalah untuk membandingkan nilai akurasi

berdasarkan ukuran gambarnya. Arsitektur diatas dapat dijelaskan seperti penjelasan dibawah ini :

1. Proses Konvolusi pertama digunakan kernel berukuran 3x3 dan jumlah filter sebanyak 32 filter , proses konvolusi ini adalah proses kombinasi antara dua buah matriks yang berbeda untuk menghasilkan suatu nilai matriks yang baru. Setelah proses konvolusi, maka ditambahkan sebuah aktivasi fungsi yaitu RELU (*Retrified Linear Unit*). Fungsi aktivasi ini bertujuan untuk mengubah nilai negative menjadi nol(menghilangkan nilai *negative* dalam sebuah matriks hasil konvolusi). Hasil konvolusi ini memiliki ukuran yang sama yakni 64x64 karena pada saat proses konvolusi digunakan nilai *padding* 0.
2. Proses *pooling*. *Pooling* merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling*. Proses *pooling.* Pada dasarnya pooling layer terdiri dari sebuah filter dengan ukuran dan tertentu yang akan secara bergantian bergeser pada seluruh area *feature map.* Penelitian ini menggunakan *max-pooling* untuk mendapatkan nilai matriks yang baru hasil dari proses *pooling*. Berdasarkan hasil pooling menghasilkan matriks baru berukuran 32x32 dengan menggunakan kernel *pooling* 2x2. Cara kerja *max-pooling* adalah mengambil nilai paling maksimum berdasarkan pergeseran kernelnya sebnayak nilai *stride*nya yaitu 2.
3. Proses Kovolusi kedua yaitu meneruskan hasil dari proses *pooling* pertama yakni dengan input matriks gambar sebesar 32x32 dengan jumlah filter sebanyak 64 filter dan dengan ukuran kernel 3x3. Proses konvolsi kedua ini sama-sama menggunakan fungsi aktivasi RELU .
4. Proses selanjutnya masuk ke proses *pooling* yang kedua, proses ini hampir sama dengan proses *pooling* hang pertama, namun ada perbedaan pada nilai *output* akhir dari matriksnya . *Output* yang dihasilkan memiliki ukuran gambar 16x16.
5. Selanjutnya *Flatten* atau *fully connected*. Pada tahap ini digunakan hanya satu *hidden layer* pada jaringan MLP (*Multi Layer Perceptron*). *Flatten* disini mengubah *output pooling* layer menjadi sebuah vector. Sebelum melakukan proses klasifikasi atau memprediksi gambar, pada proses ini digunakan nilai

Dropout. *Dropout* adalah sebuah teknik regulasi jaringan syaraf dengan tujuan memilih beberapa neuron secara acak dan tidak akan dipakai selama proses pelatihan, dengan kata lain neuron-neuron tersebut dibuang secara acak. Tujuan dari proses ini yaitu mengurangi *overfitting* pada saat proses *training.*

6. Proses terakhir adalah menggunakan aktivasi fungsi *Softmax*. Fungsi ini secara sepsifiknya fungsi ini biasa digunakan pada metode klasifikasi *multinomial logistic regression* dan *multiclass linear discriminant analisys.*

Berdasarkan uraian penjelasan dari arsitektur jaringan diatas, arsitektur tersebut digunakan untuk proses *training.* Sehingga dari proses *training* didapatkan model dari arsitektur tersebut. Berikut model yang terbentuk :

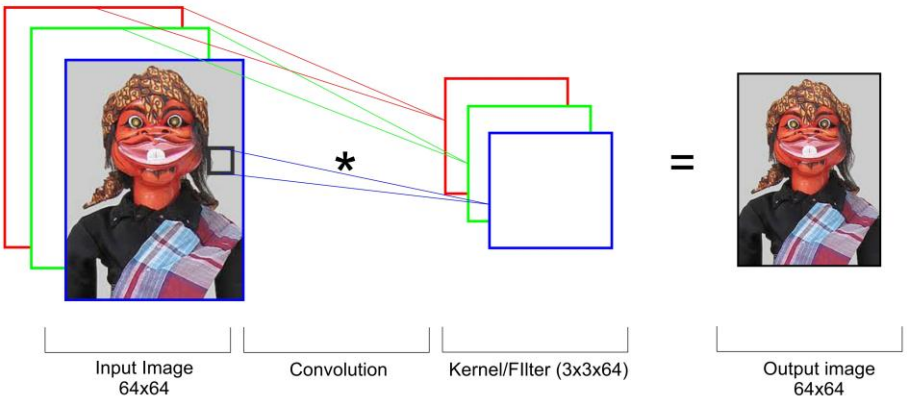
No	Nama	Size			Parameter		
0	Input			64*64*3			0
1	Conv2d_1	(64+(2*1)-(3-1))	=	64*64*32	((3*3*3)+1)*32	=	896
2	MaxPool_1			32*32*32			0
3	Conv2d_2	(32+(2*1)-(3-1))	=	32*32*64	((3*3*32)+1)*64	=	18496
4	MaxPool_2			16*16*32			0
5	Flatten			16384			0
6	Dense			256	(16384*256)+256	=	4194560
7	Output			3	(256+1)*3	=	771
Total							4.214.723

Gambar 5.2 Model CNN

Gambar diatas merupakan model yang terbentuk dari hasil *training*. Untuk menghitung input kedalam konvo digunakan rumus "*input_size + 2*padding - (filter_size -1)".* Total parameter yang terbentuk dari model sebanya 4.214.723 neuron.

5.1.1. Proses *Convolution Layer*

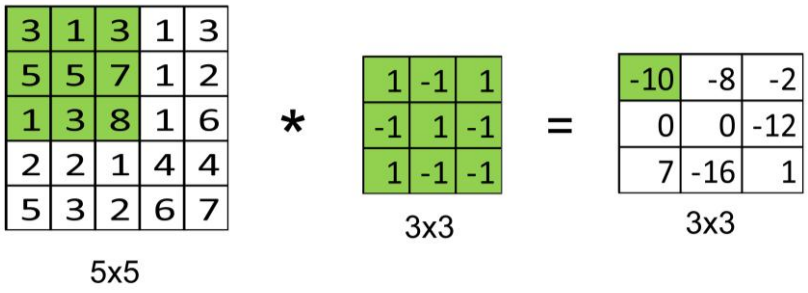
Berdasarkan Penguraian dari arsitektur jaringan, berikut ini adalah pembahasan mengenai proses konvolusi.



Gambar 5.3 Proses Konvolusi

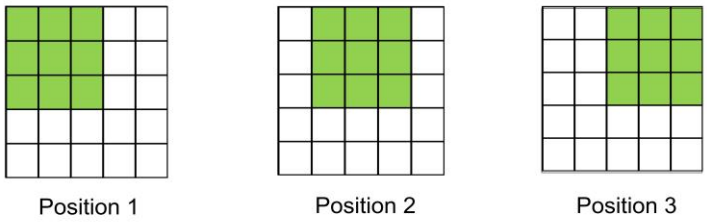
Konvolusi merupakan proses mengkombinasi dua buah deret angka yang menghasilkan deret angka yang ketiga. Jika di implementasikan angka pada konvolusi ini adalah berbentuk matriks *array*. Pada input , gambar memiliki ukuran piksel 64x64x3, ini menunjukan bahwa tinggi dan lebar piksel dari gambar sebesar 64 dan gambar tersebut memiliki 3 channel yaitu *red*, *green*, dan *blue* atau yang biasa disebut dengan RGB. Setiap *channel* piksel memiliki nilai matriks yang berbeda-beda. Input akan di konvo dengan nilai filter yang sudah ditentukan. Filter merupakan blok lain atau kubus dengan tinggi dan lebar yang lebih kecil namun kedalaman yang sama yang tersapu di atas gambar dasar atau gambar asli. Filter digunakan untuk menentukan pola apa yang akan dideteksi yang selanjutnya dikonvolusi atau dikalikan dengan nilai pada matriks input, nilai pada masing-masing kolom dan baris pada matriks sangat bergantung pada jenis pola yang akan dideteksi. Jumlah filter pada konvo ini sebanyak 64 piksel dengan ukuran kernel (3x3), ini artinya gambar yang dihasilkan dari hasil konvolusi akan sebanyak 64 fitur map.

Supaya dapat lebih memahami cara kerja dari proses konvolusi, peneliti akan menggunakan sampel matriks pada input image. Karena *input image* memiliki ukuran pikses 64x64, maka peneliti hanya mengambil sebagian nilai matriks saja yang akan di jadikan sampel dala proses konvolusi.



Gambar 5.4 Perhitungan Proses Konvolusi

Gambar 5.4 menunjukan proses konvolusi dengan menggunakan ukuran kernel 3x3, dengan menggunakan *stride* 1. *Stride* disini artinya jumlah pergeseran kernel terhadap matriks input berjumlah satu. Jika divisualisasikan sebagai berikut :



Gambar 5.5 Posisi Kernel pada Konvolusi

Gambar 5.5 menunjukan perhitungan dot *product* pada proses konvolusi dimana sebuah kernel ukuran 3x3 yang dimulai pada sisi bagian kiri. Proses ini disebut dengan *sliding window*. Namun pada penelitian ini diberikan nilai *padding* 1, yaitu adanya penambahan nilai 0 disekeliling nilai matriks input supaya input dan output memiliki nilai matriks yang sama, sehingga tidak mengurangi informasi-informasi pada gambar. Proses ini dilakukan dari ujung kiri atas sampai ujung kiri bawah. Perhitungan *dot product* dapat dilihat sebagai berikut :

a. $Position\ 1 = (3x1) + (5x(-1) + (1x1) + \big(1x(-1)\big) + (5x1) + \big(3x(-1)\big) + (3x1) + \big(7x(-1)\big) + (8x1) = -10$
b. $Position\ 2 = (1x1) + (5x(-1) + (3x1) + \big(3x(-1)\big) + (7x1) + \big(8x(-1)\big) + (1x1) + \big(1x(-1)\big) + (1x1) = -8$
c. $Position\ 3 = (3x1) + (7x(-1) + (8x1) + \big(1x(-1)\big) + (1x1) + \big(1x(-1)\big) + (3x1) + \big(2x(-1)\big) + (6x1) = -2$

d. $Position\ 4 = (5x1) + (1x(-1) + (2x1) + \big(5x(-1)\big) + (3x1) + \big(2x(-1)\big) + (7\text{x}1) + \big(8\text{x}(-1)\big) + (1\text{x}1) = 0$

e. $Position\ 5 = (5x1) + (3x(-1) + (2x1) + \big(7x(-1)\big) + (8x1) + \big(1x(-1)\big) + (1\text{x}1) + \big(1\text{x}(-1)\big) + (4\text{x}1) = 0$

f. $Position\ 6 = (7x1) + (8x(-1) + (1x1) + \big(1x(-1)\big) + (1x1) + \big(4x(-1)\big) + (2\text{x}1) + \big(6\text{x}(-1)\big) + (4\text{x}1) = -12$

g. $Position\ 7 = (1x1) + (2x(-1) + (5x1) + \big(3x(-1)\big) + (2x1) + \big(3x(-1)\big) + (8\text{x}1) + \big(1\text{x}(-1)\big) + (2\text{x}1) = 7$

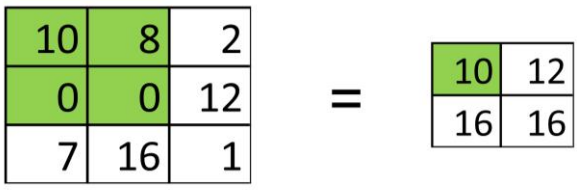
h. $Position\ 8 = (3x1) + (2x(-1) + (3x1) + \big(8x(-1)\big) + (1x1) + \big(2x(-1)\big) + (1\text{x}1) + \big(4\text{x}(-1)\big) + (6\text{x}1) = -16$

i. $Position\ 9 = (8x1) + (1x(-1) + (2x1) + \big(1x(-1)\big) + (4x1) + \big(6x(-1)\big) + (6\text{x}1) + \big(4\text{x}(-1)\big) + (7\text{x}1) = 1$

Kemudian sebelum di lanjutkan ke proses *pooling layer*, untuk menghilangkan nilai *negative* pada hasil, pada arsitektur jaringan digunakan aktivasi ReLU (*Rectified Linear Unit*) setelah proses konvolusi. Fungsi dari aktivasi ini adalah melakukan "*treshold*" dari 0 hingga *infinity.* Nilai yang ada pada hasil konvolusi yang bernilai *negative* akan diubah dengan aktivasi ini menjadi nol dan yang lainnya sampai *infinity*.

5.1.2. Proses *Pooling*

Pooling merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling* (penggabungan). Metode yang digunakan dalam proses *pooling* ini menggunakan *max-pooling. Max-pooling* merupakan salah satu metode umum yang biasa digunakan oleh peneliti yang berkaitan dengan penelitian *deep learning.* Dalam penelitian yang dilakukan oleh Dominik Scherer dkk (Scherer, 2010) menunjukan bahwa penggunaan metode *max pooling* lebih unggul dibanding dengan metode *sub sampling*. penggunaan metode ini menjadi salah satu metode terbaik dalam proses *pooling.* Berikut ini gambaran dari proses *pooling* :

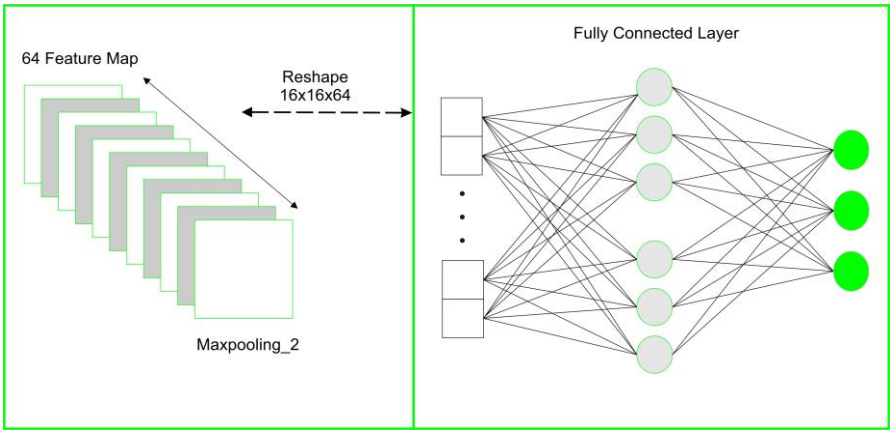


Gambar 5.6 Proses *Pooling*

Proses *pooling* ini menggunakan ukuran 2x2 dengan stride 1 dimana jumlah pergeseran kernel terhadap matriks input berjumlah satu. Dalam proses *pooling* ini digunakan metode *max-pooling*, dimana *window* akan bergeser sesuai dengan ukuran dan *stride*nnya untuk mendapatkan nilai paling maksimum. Terlihat pada gambar 5.5 *output* dari proses ini memiliki nilai yang paling maksimum yang di ambil dari matriks fitur map hasil konvolusi. Hasil max-pooling tersebut berukuran 2x2.

5.1.3. Proses *Fully Connected*

Selanjutnya adalah *Fully connected Layer*. Proses ini bertujuan untuk untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear.



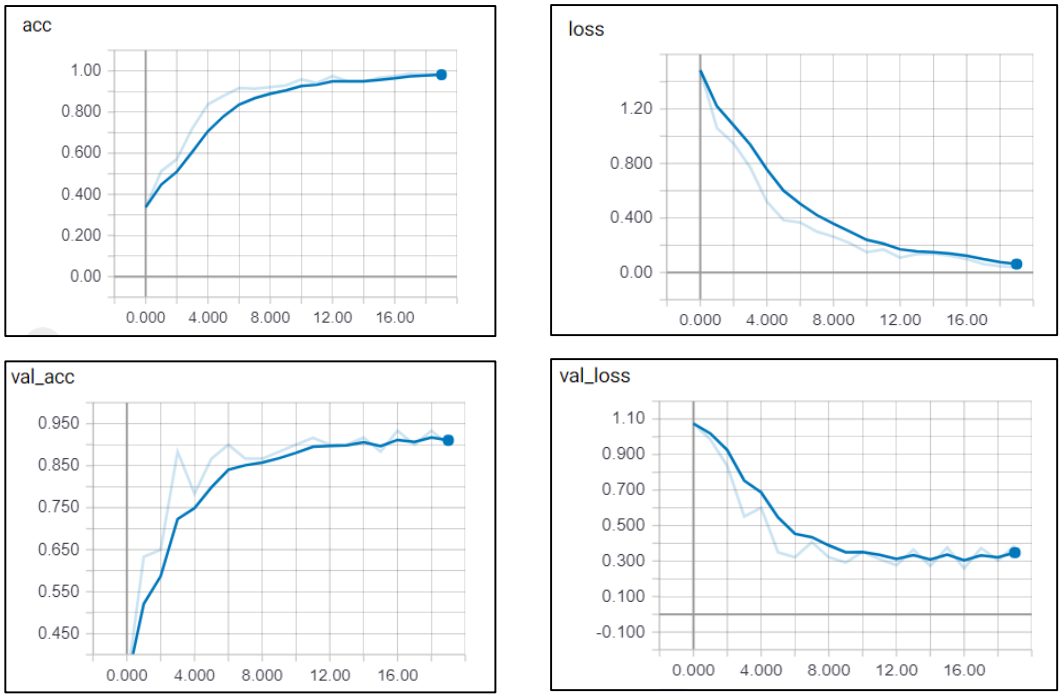
Gambar 5.7 Proses *Fully Connected Layer*

Gambar 5.7 merupakan proses *converting* hasil dari fitur map max-pooling menjadi *flatten* atau vector. Dalam proses ini nilai input matriks dari layer sebelumnya akan diubah menjadi vector. Proses ini sama dengan Proses MLP (*Multilayer Perceptron*). Jaringan ini umumnya menggunakan lapisan yang terhubung sepenuhnya di mana setiap piksel dianggap sebagai neuron terpisah. Dalam proses

ini biasanya diterapkan metode "*dropout*". Metode ini bertujuan untuk menonaktifkan beberapa edge yang terhubung ke setiap neuron untuk menghindari *overfitting*. Setelah itu proses terakhir adalah klasifikasi. Dalam proses ini digunakan aktivasi fungsi *softmax*. Aktivasi ini akan membantu MLP untuk mengklasifikasikan input terhadap targetnya, yaitu kedalam 3 kelas wayang (Cepot, Gatotkaca, Semar).

5.1.4. Model Hasil *Training*

Setelah melalu beberapa proses dalam algoritma *Convolutional Neural Network* (CNN) didapatkan hasil *training dan validation*. Proses ini menggunakan jumlah 20 *epoch*, nilai *learning rate* 0.001. berikut grafik hasil *training* menggunakan *tensorboard* :



Gambar 5.8 *Training Graph*

Berdasarkan gambar 5.8 *accuracy* dari *training* model mencapai 95 % dengan nilai *loss* sebesar 0.03864. Proses training disini menggunakan *learning rate* 0.001 dengan input gambar sebesar 64 x 64 piksel. Waktu pelatihan yang dibutuhkan untuk 20 *epoch* dalam menjalankan *training* model ini yaitu 2 menit. Semakin

Banyak *epoch* maka semakin lama juga waktu yang dibutuhkan untuk *training* model. Kemudian *accuracy* dari data *validation* mencapai 90 % dengan nilai *loss* sebesar 0.3872.

5.1.5. Hasil *Testing* Data Baru

Proses *testing* menggunakan data uji sebanyak 60, untuk setiap kelas jenis wayang sebanyak 20 gambar. Hasil *confusion matriks* adalah sebagai berikut :

Tabel 5.1. *Confusion Matriks*

Matriks		Predict Class		
		Cepot	Gatotkaca	Semar
Actual Class	Cepot	20	0	0
	Gatotkaca	0	19	1
	Semar	0	3	17

Berdasarkan tabel 5.1 diatas hasil hasil prediksi dari model terhadap data *testing* data baru menunjukan hasil yang baik. Prediksi terhadap wayang golek Cepot di klasifikasikan ke dalam Cepot, ini artinya klasifikasi terhadap gambar tersebut adalah benar. Prediksi pada wayang golek kedua Gatotkaca diklasifikasikan benar sebagai Gatotkaca sebanyak 19 dan *missing data* dari input Gatotkaca diklasifikasikan sebagai Semar sebanyak 1 data. Kemudian yang terakhir adalah prediksi pada wayang golek kedua Semar diklasifikasikan benar sebagai Semar sebanyak 17 dan *missing data* dari input Semar diklasifikasikan sebagai Gatotkaca sebanyak 3 data. Perhitungan akurasi dari keseluruhan matriks diatas adalah sebagai berikut :

$$Overall\ Accuracy = \frac{TTP_{all}}{Total\ Number\ of\ Testing\ Entries}$$

$$Overall\ Accuracy = \frac{56}{60} = 93\%$$

Jadi akurasi yang dihasilkan oleh model dengan input gambar 64x64 piksel, nilai *learning rate* sebesar 0.001 dan jumlah sampel *testing* 60 data didapatkan nilai akurasi sebsesar 93%.

5.2. Penentuan Parameter Model

Penentuan model terbaik, harus dicari nilai terbaik parameter parameter dalam model CNN. Parameter yang dimaksud adalah pengaruh jumlah *epoch*,pengaruh ukuran input gambar, pengaruh jumlah data *train*, peengaruh scenario data, ukuran kernel, dan *learning rate.* Tujuan dari penentuan parameter model ini ingin membandingkan model mana yang paling terbaik dengan memperhatikan nilai parameternya.

5.2.1. Pengaruh Jumlah *Epoch*

Epoch adalah ketika seluruh *dataset* sudah melalui proses *training* pada *Neural Network* sampai dikembalikan ke awal dalam satu putaran. Dalam *Neural Network* satu *epoch* itu terlalu besar dalam proses pelatihan karena seluruh data diikutkan kedalam proses training sehingga akan membutuhkan waktu cukup lama. Untuk mempermudah dan mempercepta proses *training* biasanya, data sate dibagi per *batch* (*Batch Size*). Penentuan nilai dari *batch size* biasanya tergantung peneliti dengan melihat banyak sampel. Berikut adalah hasil perbandingan epoch dari hasil *training*.

Tabel 5.2. *Accuracy Based on Epoch*

Epoch	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
20	91%	0.2003	147
30	90%	0.2018	185
50	93%	0.1924	313
100	97%	0.1818	636

Berdasarkan table 5.2 diatas dengan menggunakan nilai *learning rate* 0.001 didapatkan akurasi yang cukup tinggi yakni mencapai 97 %. Jika dilihat dari tabel dapat disimpulkan bahwa semakin menuju nilai 100 *epoch* yang digunkan maka akurasi dari hasil *testing* semakin tinggi. Tetapi ketika ditambahkan *epoch* lebih dari seratus nilai akurasi akan mengalami penurunan. Ini dapat disebabkkan oleh jumlah *epoch* yang terlalu banyak bisa juga dipengaruhi oleh banyaknya *dataset.*

5.2.2. Pengaruh Jumlah Layer Konvolusi

Layer Konvolusi merupakan bagian hal terepenting dalam *convolutional neural network. Tujuan digunakannya layer konvolusi untuk proses ekstraksi fitur pada gambar*. Penggunaan dari banyaknya layer konvolusi yang digunakan dapat mempengaruhi tingkat akurasi dari model.

Tabel 5.3. *Accuracy Based on Convolution Layer*

Jumlah Konvolusi	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
2	90%	0.3872	719
3	96%	0.1274	837
4	95%	0.2672	977

Berdasarkan tabel diatas menunjuakan bahwa penggunaan dari banyaknya layer konvolusi pada penelitian ini dapat meningkatkan tingkat akurasi yang lebih tinggi dibanding dengan menggunakan 2 layer konvolusi. Namun ketika semakin banyak penggunaan layer konvolusi akan memperlambat proses pelatihan model, hal ini disebabkan oleh banyaknya tahap ekstraksi dari fitur/gambar yang dilakukan oleh komputer sehingga memakan waktu yang cukup lama. Sehingga dari tabel dapat dilihat semakin banyak jumlah layer konvolusi, waktu yang dibutuhkan dalam proses pelatihan model akan semakin banyak. Sehingga penelitian ini hanya menggunakan 2 layer konvolusi untuk meminimalkan waktu pada proses pelatiahan model.

5.2.3. Pengaruh *Pooling Layer*

Pooling layer merupakan proses pengurangan ukuran matriks dari hasil proses konvolusi. Proses ini bertujuan untuk mengurangi nilai parameter sehingga mengendalikan *overfitting* pada proses *training* model. Terdapat dua metode dalam proses ini, yaitu *max-pooling* dan *Average-pooling.* Oleh karena itu penelitian ini mencoba melakukan perbandingan diantara keduanya pada saat *training* model.

Tabel 5.4. *Accuracy Based on Pooling Methods*

Pooling Layer	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
Max Pooling	95%	0.1403	932
Average Pooling	91%	0.3725	889

Berdasarkan tabel diatas, penelitian ini melakukan percobaan pada metode *pooling layer*. Metode yang digunakan adalah *max pooling* dan *average pooling*. Tabel menunjukan akurasi dengan menggunakan *max pooling* lebih tinggi daripada menggunakan *average pooling*. Namun tingkat akurasi dengan menggunakan metode *max pooling* ini tidak selalu menjadi yang terbaik, karena hal ini tergantung pada permasalahan yang dihadapi.

5.2.4. Pengaruh *Input Image*

Penelitian ini melakukan percobaan terhadap input *image* yang digunakan. Peneliti menggunakan input image sebesar 64x64 dan 150x150. Setelah dilakukan training didapatkan hasil seperti pada table 5.3

Tabel 5.5. *Accuracy Based on Input Image*

Input Shape	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
64x64	97%	0.1613	636
150x150	97%	0.1818	718

Berdasarkan percobaan diatas dapat dilihat bahwa ketika *input image* berbeda ternyata tidak memberikan hasil yang signifikan. Tingkat akurasi dari model sama-sama memiliki akurasi yang tinggi yaitu 97 %. Namun pada penelitian ini belum dilakukan percobaan kembali dengan input *image* yang memiliki ukuran piksel yang besar.

5.2.5. Pengaruh Jumlah Data *Train*

Penelitian ini mencoba untuk menggunakan jumlah data *train* yang berbeda. Data train yang digunakan dibagi menjadi tiga bagian, yaitu 150, 210, dan 300. Dari masing-masing data *train* sebanyak 20 data gambar digunakan untuk proses validasi. Hasil dari sekenario dari jumlah data train sebagai berikut :

Tabel 5.6. *Accuracy Based on Epoch*

Data train	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
150	90%	0.4754	709
210	93%	0.4462	687
300	95%	0.3189	716

Berdasarkan tabel 5.4 hasil akurasi yang didapatkan memiliki *range* 90 – 95 %. Model CNN yang telah dibuat cukup baik dalam mengklasifikasikan gambar wayang golek. Semakin tinggi jumlah data *train* yang digunakan maka akurasi yang didapatkan semakin besar. Hal ini menunjukan bahwa sebuah mesin/komputer lebih banyak memahami pola gambar sehingga, ketepatan dalam proses klasifikasi akan semakin baik.

5.2.6. Pengaruh Skenario Data

Penelitian ini mencoba menggunakan sekenario jumlah data *training* dan *testing* pada proses *training* model. Jumlah data yang digunakan sebanyak 360 data yang dibagi menjadi tiga sekenari. Hasil sekenario dapat dilihat pada tabel 5.5 berikut :

Tabel 5.7. Sekenario Data

Sekenario (Training :Testing)	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
(210 : 90)	90 %	0.3915	719
(240 : 60)	93 %	0.2646	629
(270 : 30)	100 %	0.01259	343

Berdasarkan percobaan diatas menunjukan bahwa dengan menggunakan ketiga skenario , dapat diasumsikan bahwa semakin banyaknya jumlah data *train*, maka akurassi yang didapatkan oleh model semakin tinggi. Hal ini dikarenakan semakin model tersebut dilatih dengaan banyak gambar, maka model akan semakin mengenali pola gambar dengan akurat.

5.2.7. Pengaruh Ukuran Kernel

Peneliti juga mencoba untuk menggunakan ukuran kernel yang berbeda. Pada umumnya algoritma CNN kebanyakan menggunakan ukuran filter 3x3, 5x5, dan 7x7. Tabel 5.4 menunjukan hasil percobaan ukuran kernel terhadap model.

Tabel 5.8. *Accuracy Based on Epoch*

Ukuran Kernel	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
3x3	97%	0.1818	636
5x5	93%	0.2122	638
7x7	93%	0.305	647

Tabel 5.6 menunjukan ukuran kernel menghasilkan tingkat akurasi yang paling tinggi dengan nilai 97 % model yang menggunakan ukuran filter 3x3, Sedang dengan penggunaan ukuran kernel 5x5 dan 7x7 tingkat akurasi lebih kecil dari penggunaan kernel 3x3. Hal ini dapat diasumsikan bahwa semakin kecil ukuran kernel maka pengamatan terhadap gambar akan semakin detail sehingga nilai akurasi model kemungkinan akan menjadi lebih tinggi.

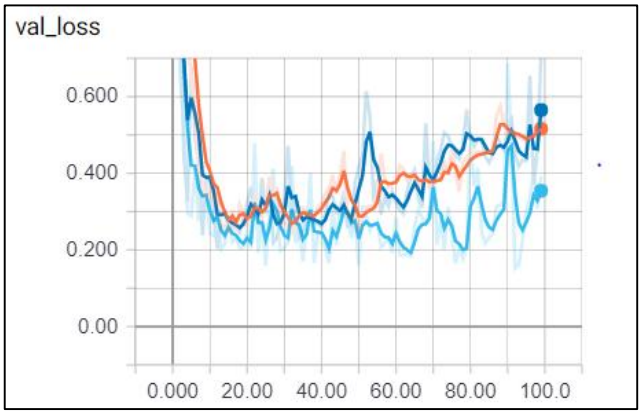
5.2.8. Pengaruh Nilai *Learning Rate*

Penelitian ini juga melakukan uji coba dengan menggunakan nilai *learning rate* yang berbeda. Dalam klasifikasi gambar pada umumnya banyak menggunakan nilai *learning rate* sebesar 0.1 sampai 0.0001. Penentuan nilai *learning rate* biasanya ditentukan oleh peneliti. Peneliti menggunakan tiga nilai yaitu 0.01, 0.001, dan 0.0001. Penentuan niali dari *learning rate* ini sangat berpengaruh pada performa akurasi. Hasil *learning rate* adalah sebagai berikut :

Tabel 5.9. *Learning Rate*

Learning rate	*Accuracy Validation*	*Loss Validation*	*Time (Seconds)*
0.01	33%	1.099	634
0.001	97%	0.1818	650
0.0001	90%	0.2935	636

Berdasarkan tabel 5.7 penggunaan nilai *learning rate* 0.01 menghasilkan tingkat akurasi yang tidak optimal yaitu sebesar 33 %, karena ketika menggunakan nilai *learning rate* dengan nilai cukup besar, maka nilai *loss* akan semakin meningkat ketika menjalankan beberapa itersi pada saat *training.* Penggunaan nilai *learning rate* 0.001 menghasilkan tingkat akurasi yang sangat besar yaitu 97 %. Hal ini disebabkan pada beberapa nilai titik funsi *loss* mulai mulai menurun dalam beberapa iterasi pertama. Berbeda dengan penggunaan *learning rate* 0.0001, tabel menunjukan memiliki tingkat akurasi 90 % lebih kecil disbanding *learning rate* sebelumnya. Hal ini tentunya disebabkan oleh lambatnya proses konvergensi nilai loss pada saat proses training. Sehingga hasil *loss validation* yang dihasilkan sebagai berikut :



Gambar 5.9. *Graph Learning Rate*

Berdasarkan gambar diatas menunjukan grafik *loss function* pada proses validasi. Grafik berwarna biru tua menunjukan penggunaan nilai *learning rate* sebesar 0.01, menghasilkan nliai loss yang cukup tinggi yaitu 1.099 , nilai ini dapat disebabkan adanya *overfitting* pada saat pelatihan model. Grafik berwarna biru muda menunjukan bahwa penggunaan nilai *learning rate* 0.001 . Jika dilihat dari dari grafik nilai *loss* yang didapatkan lebih baik dibanding dengan penggunaan nilai *learning rate* 0.01, Karena penggunaan nilai learning rate yang digunakan pada penelitian ini adalah *trial and error*, sehingga tidak dapat secara langsung menentukan nilai *learning rate* yang paling optimum. Kemudian yang terakhir , grafik yang ditunjukkan dengan warna orange merupakan penggunaan nilai *learning rate* sebesar 0.0001. Penggunaan nilai yang terakhir menghasilkan tingkat akurasi yang cukup baik, namun penggunaan nilai *learning rate* sebesar ini cukup lambat dalam memperkecil nilai *loss*, sehingga grafik nilai *loss* akan mengalami konvergensi menjadi lambat.

BAB VI
KESIMPULAN

5.1. Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan, diperoleh beberapa kesimpulan yaitu :

1. Model CNN pada penelitian ini menggunakan input shape berukuran 64x64, nilai learning rate 0.001, ukuran filter 3x3, Jumlah Epoch 20, Data training 240, dan data testing 60. Menghasilkan tingkat akurasi *training* dan *testing* dalam melakukan klasifikasi gambar wayang golek sebesar 95 % *training* dan 90 % *testing.*
2. Penelitian ini menggunakan data *testing* baru sebanyak 60 untuk diujikan kedalam model yang telah dibuat. Hasil *testing* menghasilkan tingkat akurasi baru dalam melakukan klasifikasi gambar wayang golek sebesar 93 %.
3. Dari beberapa *trial and error* pada beberapa parameter, yaitu penejlasannya sebagai berikut :
    a. Skenario penggunaan nilai epoch didapatkan tingkat akurasi terbaik menggunakan nilai epoch sebesar 100, dengan akurasi 97 %.
    b. Skenario penggunaan layer konvolusi didapatkan tingkat akurasi terbaik menggunakan 3 layer konvolusi, dengan akurasi 96 %.
    c. Skenario penggunaan *pooling layer* didapatkan tingkat akurasi terbaik menggunakan metode *max-pooling*, dengan akurasi 95 %.
    d. Skenario penggunaan input *shape image* 64x64 dan 150x150, menghasilkan tingkat akurasi yang sama yaitu 97 %.
    e. Skenario penggunaan jumlah data *training* didapatkan akurasi terbaik menggunakan jumlah data 300, dengan akurasi 95 %.
    f. Skenario penggunaan perbandingan jumlah data *training* dan *testing* didapatkan tingkat akurasi terbaik menggunakan 90 % :10 % atau 270:10 data *training* dan *testing*, dengan akurasi 100 %.
    g. Skenario penggunaan ukuran kernel didapatkan didapatkan tingkat akurasi terbaik menggunakan ukuran kernel 3x3, dengan akurasi 97 %.

h. Skenario penggunaan nilai *learning rate* didapatkan didapatkan tingkat akurasi terbaik menggunakan nilai learning rate 0.001, dengan akurasi 97 %.

5.2. Saran

Adapun saran yang diberikan pada penelitian ini sebagai berikut :

1. Penelitian selanjutnya diharapkan dapat menambah jumlah kelas klasifikasi dari seluruh tokoh wayang golek.
2. Menambahkan parameter seperti perbandingan input gambar yang lebih besar ukuran pikselnya, nilai *dropout*, fungsi aktivasi, penggunaan *optimizer*. Sehingga dengan menghasilkan model dengan penggunaan *hyperparameter* terbaik.
3. Penelitian ini dapat di kembangkan kedalam sebuah aplikasi yang digabungkan dengan *smartphone.*
4. Dapat membantu para pecinta wayang dalam mengenali tokoh-tokoh wayang golek melalui smartphone yang sudah di tambahkan aplikasi pengenal tokoh wayang, sehingga dapat melestarikan kebudayaan wayang golek ini kembali.
5. Kelemahan dari penelitian ini salah satunya tidak dapat menentukan pemilihan parameter secara optimum. Penentuan parameter harus dilakukan metode *trial and error* untuk mendapatkan tingkat akurasi yang tinggi.

DAFTAR PUSTAKA

A.Coates, H.Lee, and A.Y. Ng. (2011). *An Analisys of Singe-Layer Network in Unsupervised Feature learning.*

Alpaydin, E. (2009). *Introduction to Mechine Learning, Second Edition.* London: MIT Press.

Bernd, J., & H. Horst. (2000). *Computer Vision and Aplication.* San Diego: Academic Press.

Budianita, E., Jasril. (2015). Implementasi Pengolahan Citra dan Klasifikasi K-Nearest Neighbour Untuk Membangun Aplikasi Pembeda Daging Sapi dan Babi. *Jurnal Sains, Teknologi dan Industri* , 242-247.

Danukusumo, K. (2017). Implementasi Deep Learning Menggunakan Convolutional Neutal Network untuk Klasifikasi Citra Candi Berbasis GPU. *Tugas Akhir*.

Effendi, A. (2009). *Prof. Dr. Ir. Sedyatmo: Intuisi Mencetus daya cipta.* Jakarta: Mizan.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters 27*, pp. 861-874.

Goodfellow, I., Bengio, Y, and Courville, A. (2016). *Deep Learning (Adaptive Computation and Mechine Learning Series).* The IMT Press.

Hamida, U. (2014). PENGGUNAAN ARTIFICIAL NEURAL NETWORK (ANN) UNTUK MEMODELKAN KEBUTUHAN ENERGI UNTUK TRANSPORTASI. *Jurnal Teknologi Manajemen*, Vol. 12 , No.2.

Hermawan, A. (2006). *Jaringan Syaraf Tiruan dan Aplikasinya.* Yogyakarta: Andi.

Hong, Y., Jong Weon. (2017). Art Painting Identification using Convolutonal Neural Network. *International Journalof Applied Engineering Research*, 532-539.

Hosseini, L., & Ramin Shaghaghi Kandovan. (2017). Hyperspectral Image Classification Based on Hierarchical SVM Algorithm for Improving Overall Accuracy. *Scientific Research Publishing*, 66-75.

Hubel, D., and Wiesel, T. (1968). Receptive Fields and Functional architecture of monkey striate kortex. *Journal of Physiology (London)*, 195, 215-243.

Jumarwanto, A. (2009). Apllikasi Jaringan Syaraf Tiruan Backpropagation Untuk Memprediksi Penyakit THT di Rumah Sakit Mardi Rahayu Kudus. *Jurnal Teknik Elektro*, Vol. 1, No. 1.

Krizhevsky, A., Ilya Sutskever, and Geoferry E. Hinton. (2012). Image Net Classification with Deep Covolutional Neural Network. *Communications of the ACM*, 1097-1105.

Kumar, K., Haynes, J.D. (2003). Forecasting Credit ratings Using an ANN and Statistitical Techniques. *International journal of Business Studies*, 91-108.

Lukman, A. (2012). Implementasi pengolahan citra dan Algoritma LVQ Untuk Pengenalan Buku. *Seminar Nasional Informatika*, (hal. 145-155).

Mohri, et al. (2012). *Foundations of Mechine Learning.* Cambridge: MIT Press.

Pannu, A., & M. Tech Student. (2015). Artificial Intelligence and its Application in Different Areas. *International Journal of Engineering and Innovative Technology (IJEIT)*, Volume 4, ISSN: 2277-3754.

Park, D.-C. (2016). Image Classification Using Naïve Bayes Classifier. *International Journal of Computer Science and Electronics Engineering (IJCSEE)*, Vol. 4 , ISSN 2320–4028.

Pasha, L. (2011). *Buku Pintar Wayang.* Yogyakarta: Bentang Pustaka.

Pham, D. (1994). *Neural Network for Chemical Engineers.* Amsterdam: Elsevier Press.

Purwadi. (2013). *Jurnal Kebudayaan jawa : Pendidikan Budi Pekerti dalam Seni Pewayangan.* Yogyakarta: Narasi.

Rich, Elaine, and Kevin Knight. (1991). *Artificial Intellegence.* New York: McGraw-Hill inc.

Rosli, R., et al. (2012). Mango Grading By Using Fuzzy Image Analysis. *International Conference on Agricultural, Environment and Biological*, (hal. pp.18–22).

Ruder, S. (2018, May 30). *An overview of gradient descent optimization algorithms*. Diambil kembali dari Ruder.io: http://ruder.io/optimizing-gradient-descent/

Scherer, D., Andreas Muller, and Sven Behnke. (2010). Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. *20th International Conference on Artificial Neural Networks (ICANN).* Thessaloniki, Greece.

Sena, S. (2018, Mei 27). *Pengenalan Deep Learning Part 1 : Neural Network*. Diambil kembali dari Medium: https://medium.com/@samuelsena/pengenalan-deep-learning-8fbb7d8028ac

Sena, S. (2018, Mei 27). *Pengenalan Deep Learning Part 7 : Convolutional Neural Network (CNN)*. Diambil kembali dari Medium: https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-cnn-b003b477dc94

Setiawan, B., dan Rudiyanto. (2004). Aplikasi Neural Networks Untuk Prediksi Aliran Sungai. *Prosiding Semiloka Teknologi Simulasi dan Komputasi serta Aplikasi 2004.* Jakarta: BPPT.

Srivastava, N., Hinton, G, and Kriszhevsky, A. (2014). Dropout : A Simple Way to Prevent Neural Network . *Journal Conference Learning Research*, 1929-1958.

Sudjarwo, Heru S, Sumari, dan Undung Wijaya. (2010). *Rupa & Karakter Wayang.* Jakarta: Kakilangit Kencana.

Sulaeman, M. (1998). *Ilmu Budaya Dasar Suatu Pengantar*. Bandung: Rafika Aditama.

Suseno, F. (1991). *Wayang dan Pnggilan Manusia.* Jakarta: Gramedia Pustaka Utama.

Sutoyo, T., Mulyanto, E., Suhartono, Dwi Nurhayati Oky, & Wijanarto. (2009). *Teori Pengolahan Citra Digital.* Yogyakarta: Andi Yogyakarta dan UDINUS Semarang.

Trnovsky, T., Dkk. (2017). Animal Recognition System Base On Convolutional Neural Network . *Digital Image Processing And Computer Graphics*, Vol.15, No.3.

Visalini, S. (2017). Traffic Sign Recognition Using Convolutional Neural Network. *International Jurnal of Innovative Research in Computer and Communication Engineering*, Vol.5.

Wicaksono, A., Dkk. (2017). Midified Concolutional Neural Network Architecture for Batik Motif Image Classification. *IPTEK, Journal of Science*, Vol.2, No.1.

Wikipedia.org. (2018, Mei 26). *Web Crawler*. Diambil kembali dari Wikipedia: https://en.wikipedia.org/wiki/Web_crawler

Zhang, Z. (2016). *Derivation of Backpropagation in Convolutional Neural Network (CNN).* Tennessee : University of Tennessee .

Zufar, M., dan Budi Setiyono. (2017). Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time. *Jurnal Sains dan Seni ITS*, 2337-3520.

RINGKASAN TUGAS AKHIR

IMPLEMENTASI *DEEP LEARNING* UNTUK *IMAGE CLASSIFICATION* MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK (CNN)* PADA CITRA WAYANG GOLEK

[1]Triano Nurhikmat, [2]Tuti Purwaningsih
Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Islam Indonesia
14611209@students.uii.ac.id

Indonesia merupakan bangsa yang terdiri dari berbagai etnik dan memiliki latar belakang budaya yang beraneka ragam. Salah satu hasil kebudayaan masyarakat Indonesia adalah Wayang golek. Wayang merupakan seni tradisional yang bekembang di indonesia terutama di pulau Jawa Barat. Kebudayaan ini telah diakui oleh UNESCO sebagai budaya adiluhung. Melihat penghargaan tersebut sudah seharusnya masyarakat Indonesia menjaga dan melestarikannya. Akan tetapi, diera sekarang ini dunia teknologi sudah semakin berkembang, sehingga banyak masyarakat yang melupakan akan kebudayaan tradisional ini, tertuama dikalangan remaja. Hasil survey berdasarkan citra digital toko-tokoh pewayangan menunjukan sebanyak 71 % dari 60 orang tidak mengenalinya. Ini bertujuan untuk membatu mengklasifikasi objek tokoh-tokoh pewayangan berdasarkan citra digital. Sehingga, dibutuhkan suatu pendekatan dalam penyelesaian permasalan ini. Salah satu pendekatan dalam pengenalan suatu gambar adalah menggunakan metode Convolutional Neural Network. Metode ini salah satu metode Deep learning yang dapat digunakan untuk mengenali dan mengklasifikasi sebuah objek pada sebuah citra digital. Berdasarkan hasil pembahasan didapatkan tingkat akurasi sebesar 95% pada proses training dan 90 % pada proses testing. Kemudian penelitian ini menggunakan data baru untuk menguji model yang telah dibuat. Tingkat akurasi yang dihasilkan menggunakan data baru sebesar 93 % dalam mengklasifikasikan gambar wayang golek. Sehingga, performa dari model yang dibuat pada penelitian ini dapat dikatakan optimal dalam mengklasifikasikan gambar wayang golek.

Kata Kunci : *Deep Learning, Image Classification, Wayang Golek*

PENDAHULUAN

Indonesia merupakan salah satu negara yang memiliki budaya yang beraneka ragam. Salah satu hasil kebudayaan masyarakat Indonesia adalah Wayang. Wayang merupakan seni tradisional yang bekembang di indonesia terutama di pulau Jawa dan Bali. Terdapat 2 versi jenis wayang

yaitu wayang orang yang di mainkan langsung oleh beberapa orang dan wayang yang berwujud boneka yang dimainkan oleh dalang. Salah satu wayang berwujud boneka adalah Wayang Golek. Wayang golek merupakan suatu seni pertunjukan wayang yang terbuat dari boneka kayu yang berasal dari Jawa Barat. Wayang golek pada umumnya kebanyakan ceritanya diambil dari cerita *Ramayana* dan *Mahabarata* dengan menggunakan bahasa Sunda. Di dunia internasional wayang kini telah tercatat sebagai karya seni budaya *adiluhung*, yaitu oleh UNESCO, sebuah lembaga di bawah PBB yang menangani masalah pendidikan, ilmu pengetahuan, dan kebudayaan. Pada tanggal 7 November 2003 wayang Indonesia diumumkan oleh UNESCO sebagai karya agung dunia di Paris.

Melihat Penghargaan tersebut sudah seharusnya budaya ini dijaga dan dilestarika oleh masyarakat Indonesia. Namun seiring perkembangan zaman masyarakat sudah banyak beralih dan meninggalkan budaya ini, Hal ini dapat mengakibatkan pengetahuan mengenai kebudayaan ini akan semakin menurun pada generasi muda zaman sekarang. Hasil survey berdasarkan citra digital toko-tokoh wayang berdasarkan karakternya menunjukan sebanyak 71 % dari 60 orang remaja tidak mengenalinya. Survey ini bertujuan untuk mengetahui apakah remaja saat ini masih mengenal tokoh-tokoh wayang berdasarkan karakter gambar wayang golek .

Seiring dengan kemajuan zaman, klasifikasi citra digital sangat dibutuhkan diberbagai macam bidang, seperti : informatika, kedokteran, kelautan, pertanian, dan bisnis. Beberapa penelitian yang telah dilakukan misalnya klasifikasi buku (Lukman, 2012) dan klasifikasi pada daging sapi (Budianita, 2015). Tujuan dari klasifikasi citra adalah mengklasifikasikan masukkan citra kedalam beberapa kategori tertentu. Klasifikasi citra saat ini menjadi salah satu problem yang telah lama dicari solusinya dalam *computer vision.* Bagaimana menduplikasikan kemampuan manusia dalam memahami informasi citra digital, supaya komputer dapat mengenali

objek pada citra selayaknya manusia. Kalangan akademisi telah banyak bergelut dalam problem ini. Salah satu pendekatan yang berhasil digunakan dengan menggunakan Jaringan Syaraf Tiruan (*Artificial Neural Network*, ANN). ANN adalah salah satu bentuk kecerdasan buatan yang mempunyai kemampuan untuk belajar dari data dan tidak membutuhkan waktu lama dalam pembuatan model (Setiawan, 2004). ANN merupakan bagian dari *Mechine Learning* (ML). *Mechine Learning* adalah kecerdasan buatan yang bertujuan untuk mengoptimalkan performa dari suatu sistem dengan mempelajari data sampel atau data histori (Alpaydin, 2009). Jenis model ANN yang terdiri dari banyak lapisan disebut sebagai *Multi-Layer Perceptron* (MLP) yang berfungsi menghubungkan penuh diantara neuronnya. Kemampuan dari MLP ini dapat mmengklasifikasikan secara *powerfull*. Namun teknik klasifikasi menggunakan MLP ini memiliki kelemahan ketika input yang dimasukan berupa gambar. Gambar yang harus dilakukan pre-processing, segmentasi, dan di ekstrak untuk medapatkan kinerja yang optimal. Pengembangan lain dari MLP yang dapat mengatasi permasalahan ini adalah *Convolutional Neural Network* (CNN).

Convolutional Neural Network (CNN) merupakan salah satu metode *Deep learning* (DL) yang dapat digunakan untuk mendeteksi dan mengenali sebuah objek pada sebuah citra digital. *Deep Learning* merupakan salah satu sub bidang dari *Mechine Learning.* Pada dasarnya *Deep Learning* adalah implementasi konsep dasar dari *Mechine Learning* yang menerapkan algoritma ANN dengan lapisan yang lebih banyak. Banyaknya lapisan tersembunyi yang digunakan antara lapisan masukan dan lapisan keluaran, maka jaringan ini dapat dikatakan *deep neural net.* Beberapa tahun terakhir *Deep Learning* telah menunjukan performa yang luar biasa. Hal ini sebagain besar dipengaruhi faktor komputasi yang lebih kuat, data set yang besar dan teknik untuk melatih jaringan yang lebih dalam (Goodfellow, Bengio, Y, dan Courville, A., 2016). Kemampuan CNN di klaim sebagai model terbaik untuk memecahkan

permasalahan *object detection* dan *object recognition*. Pada tahun 2012, Penelitian tentang CNN dapat melakukan pengenalan citra digital dengan akurasi yang menyaingi manusia pada dataset tertentu. (A. Coates, H.Lee, A.Y. Ng, 2011). Namun dalam CNN, seperti model *deep learning* lainnya, memiliki kelemahan yaitu proses pelatihan model yang cukup lama. Tetapi dengan perkembangan *hardware* yang semakin pesat, hal tersebut dapat diatasi menggunakan teknologi *Graphical Procesing Unit* (GPU) dan PC yang memiliki spesifikasi tinggi. Berdasarkan latar belakang di atas, penelitian ini menerapkan implementasi dari metode *deep learning* menggunakan CNN untuk membantu mengenali tokoh-tokoh pewayangan. Berdasarkan permasalahan yang ada sehingga penelitian ini bertujuan untuk Mengetahui implementasi metode *Deep Learning* menggunakan CNN untuk mengklasifikasikan citra wayang berdasarkan tokoh-tokoh wayang golek dan mengetahui bagaimana tingkat akurasi yang didapatkan dari hasil klasifikasi menggunakan CNN.

LANDASAN TEORI

a. Convolutional Neural Network

Convolutional Neural Network (CNN) merupakan pengembangan dari *multilayer perceptron* (MLP) yang didesain untuk mengolah data dua dimensi dalam bentuk citra. CNN ini termasuk kedalam jenis *Deep Neural Network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Pada dasarnya klasifikasi citra dapat digunakan dengan MLP, akan tetapi dengan metode MLP kurang sesuai untuk digunakan karena tidak menyimpan informasi spasial dari data cita dan menganggap setiap piksel adalah fitur yang independen sehingga menghasilkan hasil yang kurang baik. Penelitian awal yang mendasari oenemuan CNN ini pertama kali dilakukan oleh Hubel dan Wiesel (Hubel & Wiesel, T, 1968) mengenai *viual cortex* pada indera penglihatan kucing.

b. Convolutional Layer

Convolution layer merupakan bagian dari tahap pada arsitektur CNN.

Tahap ini melakukan operasi konvolusi pada *output* dari layer sebelumnya. Layer tersebut adalah proses utama yang mendasari jaringan arsitektur CNN. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *Feature Map* dari input citra. *Input* dan *output* ini dapat dilihat sebagai dua argumen bernilai riil. Operasi konvolusi dapat dituliskan sebagai berikut :

$s(t) = (x*t)\,(t) = \sum_{\alpha=-\infty}^{\infty} x(\alpha) * w(t-\alpha)$ (1)

Keterangan :

S(t) = Fungsi hasil operasi konvolusi

X = Input

W = bobot (kernel)

Selain itu, penentuan volume *output* juga dapat ditentukan dari masing-masing lapisan dengan *hyperparameters*. *Hyperparameter* yang digunakan pada persamaan di bawah ini digunakan untuk menghitung banyaknya neuron aktivasi dalam sekali *output*. Perhatikan persamaan berikut :

$(W - F + 2P)/(S + 1)$ (2)

Keterangan :

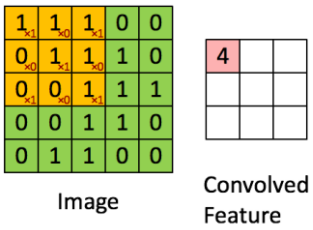
W = Ukuran volume gambar

F = Ukuran Filter

P = Nilai *Padding* yang digunakan

S = Ukuran Pergeseran (*Stride*)

Berikut ini merupakan gambar proses konvolusi.

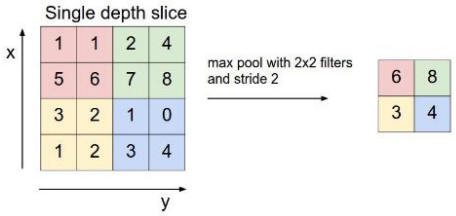


Gambar 1. *Convolution Proccess*

c. Pooling Layer

Pooling merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling*. Pooling Layer biasanya berada setelah conv. Pada dasarnya pooling layer terdiri dari sebuah filter dengan ukuran dan stride tertentu yang akan secara bergantian bergeser pada seluruh area *feature map*. Bentuk lapisan *pooling* umumnya dengan menggunakan filter dengan ukuran 2x2 yang diaplikasikan dengan langkah sebanyak dua dan beroperasi pada

setiap irisan dari inputnya. Berikut ini adalah contoh gambar operasi *max-pooling* :

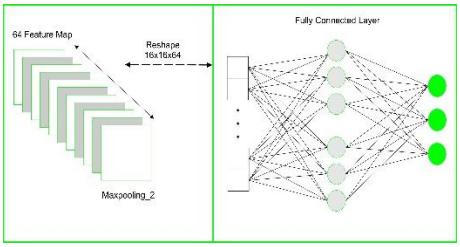


Gambar 2. Operasi *Max-Pooling*

d. Pooling Layer

Fully-Conected Layer adalah sebuah lapisan dimana seua *neuron* aktivasi dari lapisan sebelumnya terhubung semua dengan *neuron* di lapisan selanjutnya sama seperti halnya dengan *neural network biasa*. Pada dasarnya lapisan ini biasanya digunakan pada MLP (*Multi Layer Perceptron*) yang mempunyai tujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear.

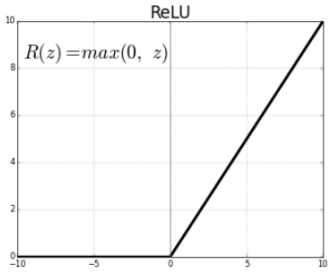
Perbedaan antara lapisan *Fully-Connected* dan lapisan konvolusi biasa adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada input, sementara lapisan *Fully-Connected* memiliki neuron yang secara keseluruhan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan dot produk, sehingga fungsinya tidak begitu berbeda.



Gambar 3. *Fully Connected*

e. Fungsi Aktivatsi

Fungsi aktivasi merupakan fungsi yang menggambarkan hubungan antara tingkat aktivitas internal (*summation function*) yang mungkin berbentuk linear ataupun *non-linear*. Fungsi ini bertujuan untuk menentukan apakah *neuron* diaktifkan atau tidak. Salah satu fungsi aktivasi yang biasa digunakan dalam CNN adalah fungsi aktivasi ReLU (*Rectified Linear Unit*). Pada dasarnya fungsi ReLU (*Rectified Linear Unit*) melakukan "*treshold*" dari 0 hingga *infinity*. Berikut adalah grafik dari fungsi aktivasi ReLU :



Gambar 4. Fungsi Aktivasi *ReLU*

Pada fungsi ini masukan dari neuron-neuron berupa bilangan negatif, maka fungsi ini akan menerjemahkan nilai tersebut kedalam nilai 0, dan jika masukan bernilai positif maka *output* dari neuron adalah nilai aktivasi itu sendiri.

METODOLOGI PENELITIAN

a. Populasi dan Sampel

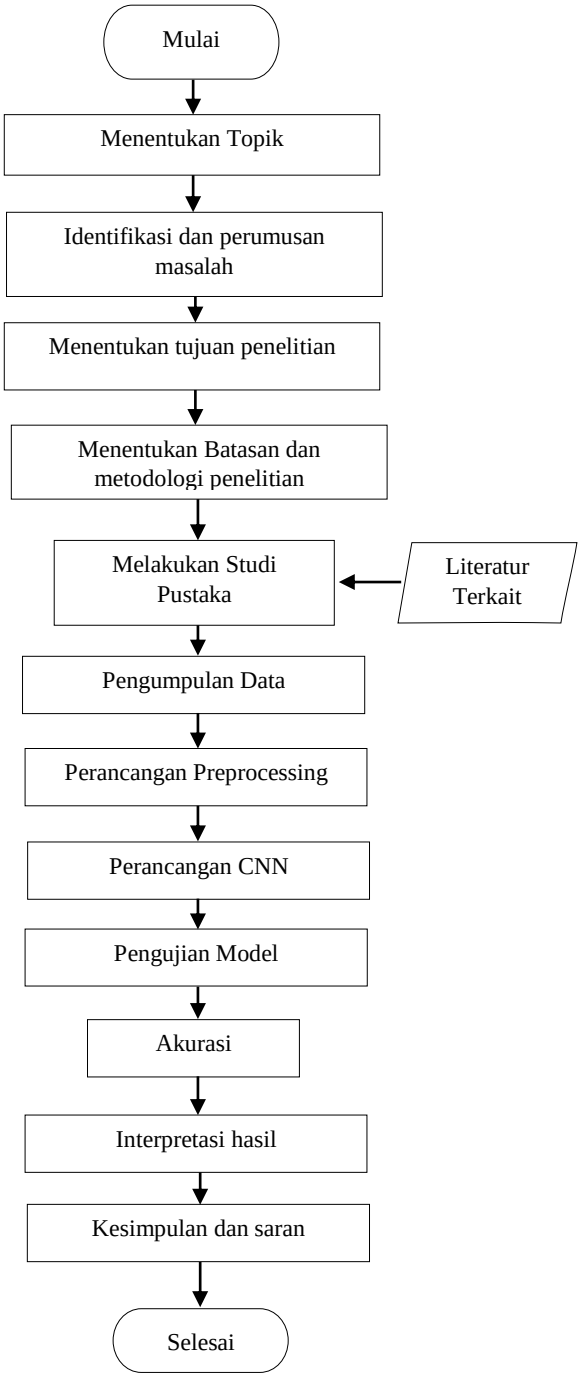
Populasi dalam penelitian ini adalah citra tokoh-tokoh wayang penggaris yang diambil dari situs pencarian *google.* Sedangkan sampel yang digunakan dalam penelitian ini adalah tiga karakter wayang yaitu Cepot, Gatotkaca, dan Semar. Total citra yang dikumpulkan untuk sampel sebanyak 300, dengan masing-masing kategori jenis wayang sebanyak 100 citra wayang golek

b. Jenis dan Sumber data

Jenis data yang digunakan dalam penelitian ini adalah data primer. Data tersebut diperoleh dengan cara *crawling* citra tokoh-tokoh wayang pada *search engine google.*

c. Tahapan Penelitian

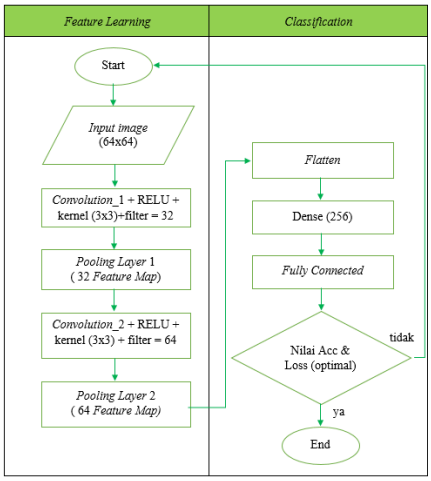
Langkah atau tahapan yang dilakukan pada penelitian ini digambarkan melalui *flow* berikut :



Gambar 5. Tahapan Penelitian

d. Rancangan Model CNN

Rancangan Model yang digunakan pada penelitian ini memiliki dua tahapan seperti pada model CNN pada umumnya, yaitu *feature extraction* dan *classification.* Pada tahap *feature extraction*, input gambar pada model berukuran 64x64x3. Angka tiga yang dimaksud adalah sebuah citra yang memiliki 3 *channel* yaitu *Red*, *Green*, dan *Blue* (RGB). Penelitian ini menggunakan 3 layer konvolusi dan 2 pooling layer. Masing-masing layer konvolusi menggunakan filter ukuran 3x3. Kemudian proses pooling menggunakan metode *max-pooling.* Kemudian pada tahap classification digunakan *neural network* yang memiliki satu *hidden layer*. Perhatikan *flow chart* model berikut :



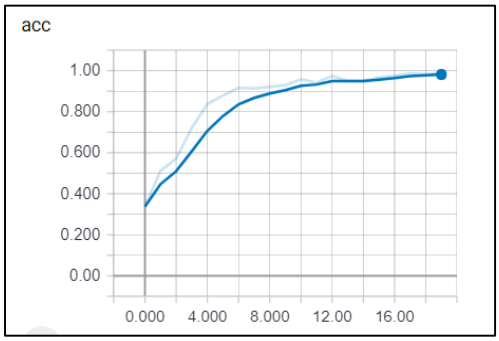
Gambar 6. Model CNN

HASIL DAN PEMBAHASAN

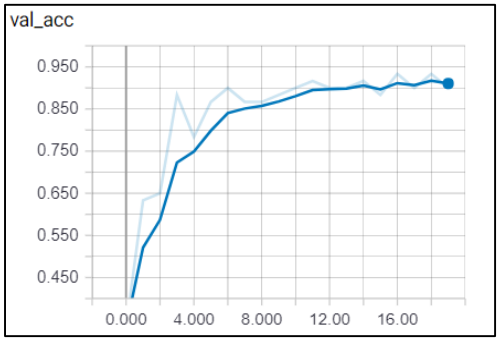
Berdasarkan dari hasil pelatihan model didapatkan hasil sebagai berikut :

a. Hasil *Training* Model

Setelah melalu beberapa proses dalam algoritma *Convolutional Neural Network* (CNN) didapatkan hasil *training dan validation*. Proses ini menggunakan jumlah 20 *epoch*, nilai *learning rate* 0.001. berikut grafik hasil *training* menggunakan *tensorboard :*



Gambar 7. *Accuracy Training*



Gambar 8. *Accuracy Validation*

Berdasarkan gambar 5.8 *accuracy* dari *training* model mencapai 95 % dengan nilai *loss* sebesar 0.03864.

Waktu pelatihan yang dibutuhkan untuk 20 *epoch* dalam menjalankan *training* model ini yaitu 2 menit. Banyak *epoch* maka semakin lama juga waktu yang dibutuhkan untuk *training* model. Kemudian *accuracy* dari data *validation* mencapai 90 % dengan nilai *loss* sebesar 0.3872.

b. Hasil *Testing* Data Baru

Proses *testing* menggunakan data uji sebanyak 60, untuk setiap kelas jenis wayang sebanyak 20 gambar. Hasil *confusion matriks* adalah sebagai berikut **:**

Tabel 1. *Confusion Matriks*

Matriks		Pred Class		
		Cepot	Gatot kaca	Semar
Act Class	Cepot	20	0	0
	Gatot kaca	0	19	1
	Semar	0	3	17

Berdasarkan tabel 5.1 diatas hasil hasil prediksi dari model terhadap data *testing* data baru menunjukan hasil yang baik. Prediksi terhadap wayang golek Cepot di klasifikasikan ke dalam Cepot, ini artinya klasifikasi terhadap gambar tersebut adalah benar. Prediksi pada wayang golek kedua Gatotkaca diklasifikasikan benar sebagai Gatotkaca sebanyak 19 dan *missing data* dari input Gatotkaca diklasifikasikan sebagai Semar sebanyak 1 data. Kemudian yang terakhir adalah prediksi pada wayang golek kedua Semar diklasifikasikan benar sebagai Semar sebanyak 17 dan *missing data* dari input Semar diklasifikasikan sebagai Gatotkaca sebanyak 3 data. Perhitungan akurasi dari keseluruhan matriks diatas adalah sebagai berikut :

$$Overall\ Accuracy = \frac{TTP_{all}}{Total\ Number\ of\ Testing\ Entries}$$

$$Overall\ Accuracy = \frac{56}{60} = 93\%$$

Jadi akurasi yang dihasilkan oleh model dengan input gambar 64x64 piksel, nilai *learning rate* sebesar 0.001 dan jumlah sampel *testing* 60 data didapatkan nilai akurasi sebsesar 93%.

KESIMPULAN DAN SARAN

a. Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan, diperoleh beberapa kesimpulan yaitu :

1. Model CNN pada penelitian ini menggunakan input shape berukuran 64x64, nilai learning rate 0.001, ukuran filter 3x3,

Jumlah Epoch 20, Data training 240, dan data testing 60. Menghasilkan tingkat akurasi *training* dan *testing* dalam melakukan klasifikasi gambar wayang golek sebesar 95 % *training* dan 90 % *testing.*

2. Penelitian ini menggunakan data *testing* baru sebanyak 60 untuk diujikan kedalam model yang telah dibuat. Hasil testing menghasilkan tingkat akurasi baru dalam melakukan klasifikasi gambar wayang golek sebesar 93 %.

b. Saran

Adapun saran yang diberikan pada penelitian ini sebagai berikut :

1. Penelitian selanjutnya diharapkan dapat menambah jumlah kelas klasifikasi dari seluruh tokoh wayang golek.
2. Menambahkan parameter seperti perbandingan input gambar yang lebih besar ukuran pikselnya, nilai *dropout,* fungsi aktivasi, penggunaan *optimizer*. Sehingga dengan menghasilkan model dengan penggunaan *hyperparameter* terbaik.
3. Penelitian ini dapat di kembangkan kedalam sebuah aplikasi yang digabungkan dengan *smartphone.*

DAFTAR PUSTAKA

A.Coates, H.Lee, and A.Y. Ng. (2011). *An Analisys of Singe-Layer Network in Unsupervised Feature learning*.

Alpaydin, E. (2009). *Introduction to Mechine Learning, Second Edition.* London: MIT Press

Budianita, E., Jasril. (2015). Implementasi Pengolahan Citra dan Klasifikasi K-Nearest Neighbour Untuk Membangun Aplikasi Pembeda Daging Sapi dan Babi. *Jurnal Sains, Teknologi dan Industri* , 242-247**.**

Goodfellow, I., Bengio, Y, and Courville, A. (2016). *Deep Learning (Adaptive Computation and Mechine Learning Series).* The IMT Press.

Hubel, D., and Wiesel, T. (1968). Receptive Fields and Functional architecture of monkey striate kortex. *Journal of Physiology (London)*, 195, 215-243

Lukman, A. (2012). Implementasi pengolahan citra dan Algoritma LVQ Untuk Pengenalan Buku. *Seminar Nasional Informatika*, (hal. 145-155)

Sena, S. (2018, Mei 27). *Pengenalan Deep Learning Part 7 : Convolutional Neural Network (CNN)*. Diambil kembali dari Medium: https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-cnn-b003b477dc94 .

Setiawan, B., dan Rudiyanto. (2004). Aplikasi Neural Networks Untuk Prediksi Aliran Sungai. *Prosiding Semiloka Teknologi Simulasi dan Komputasi serta Aplikasi 2004*. Jakarta: BPPT.

LAMPIRAN

***Lampiran 1**. Script Crawling data with Python*

```
# USAGE
# python download_images.py --urls urls.txt --output images/santa

# import the necessary packages
from imutils import paths
import argparse
import requests
import cv2
import os

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-u", "--urls", required=True,
	help="path to file containing image URLs")
ap.add_argument("-o", "--output", required=True,
	help="path to output directory of images")
args = vars(ap.parse_args())

# grab the list of URLs from the input file, then initialize the
# total number of images downloaded thus far
rows = open(args["urls"]).read().strip().split("\n")
total = 0

# loop the URLs
for url in rows:
	try:
		# try to download the image
		r = requests.get(url, timeout=60)

		# save the image to disk
		p = os.path.sep.join([args["output"], "{}.jpg".format(
			str(total).zfill(8))])
		f = open(p, "wb")
		f.write(r.content)
		f.close()

		# update the counter
		print("[INFO] downloaded: {}".format(p))
		total += 1

	# handle if any exceptions are thrown during the download
process
	except:
		print("[INFO] error downloading {}...skipping".format(p))

# loop over the image paths we just downloaded
for imagePath in paths.list_images(args["output"]):
	# initialize if the image should be deleted or not
	delete = False

	# try to load the image
	try:
		image = cv2.imread(imagePath)

```

```
56                # if the image is `None` then we could not properly load
57 it
58                # from disk, so delete it
59                if image is None:
60                      print("None")
61                      delete = True
62
63          # if OpenCV cannot load the image then the image is likely
64          # corrupt so we should delete it
65          except:
66                print("Except")
67                delete = True
68
69          # check to see if the image should be deleted
70          if delete:
71                print("[INFO] deleting {}".format(imagePath))
72                os.remove(imagePath)
```

Lampiran 2. *Script Training Model with Python*

```
 1 import sys
 2 import os
 3 from keras.preprocessing.image import ImageDataGenerator
 4 from keras import optimizers
 5 from keras.layers.convolutional import Conv2D
 6 from keras.layers.convolutional import MaxPooling2D
 7 from keras.layers.core import Activation
 8 from keras.layers.core import Flatten
 9 from keras.layers.core import Dense
10 from keras.layers.core import Dropout
11 from keras.models import Sequential
12 from keras import callbacks
13
14 DEV = False
15 argvs = sys.argv
16 argc = len(argvs)
17
18 if argc > 1 and (argvs[1] == "--development" or argvs[1] == "-d"):
19   DEV = True
20
21 if DEV:
22   epochs = 2
23 else:
24   epochs = 100
25
26 train_data_path = './data/train'
27 validation_data_path = './data/validation'
28
29 """
30 Parameters
31 """
32 img_width, img_height = 64,64
33 batch_size = 30
34 samples_per_epoch = 240
35 validation_steps = 60
36 nb_filters1 = 32
37 nb_filters2 = 64
38 conv1_size = 3
39 conv2_size = 3
```

```
pool_size = 2
classes_num = 3
lr = 0.001

model = Sequential()
model.add(Conv2D(nb_filters1,(conv1_size, conv1_size), padding ="same",
input_shape=(img_width, img_height, 3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(pool_size, pool_size)))

model.add(Conv2D(nb_filters2, (conv2_size, conv2_size), padding ="same"))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(pool_size, pool_size),
dim_ordering='th'))

model.add(Flatten())
model.add(Dense(256))
model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(classes_num, activation='softmax'))

model.compile(loss='categorical_crossentropy',
              optimizer=optimizers.Adam(lr=lr, beta_1=0.9,
                      beta_2=0.999, epsilon=None, decay=0.0,
amsgrad=False),
              metrics=['accuracy'])

train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_data_path,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical')

validation_generator = test_datagen.flow_from_directory(
    validation_data_path,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical')

"""
Tensorboard log
"""
log_dir = './tf-log/tf-log(epoch=100,lr=0.001,Op=adam)/'
tb_cb = callbacks.TensorBoard(log_dir=log_dir, histogram_freq=0)
cbks = [tb_cb]

model.fit_generator(
    train_generator,
    samples_per_epoch=samples_per_epoch,
    epochs=epochs,
    validation_data=validation_generator,
    callbacks=cbks,
    validation_steps=validation_steps)

```

```
target_dir = './models/model(epoch=100,lr=0.001,Op=adam)/'
if not os.path.exists(target_dir):
  os.mkdir(target_dir)
model.save('./models/model(epoch=100,lr=0.001,Op=adam)/model.h5')
model.save_weights('./models/model(epoch=100,lr=0.001,Op=adam)/weights.h5')
```

Lampiran 3. *Script Testing Model with Python*

```
import os
import numpy as np
from keras.preprocessing.image import ImageDataGenerator, load_img,
img_to_array
from keras.models import Sequential, load_model

img_width, img_height = 64, 64
model_path = './models/model(epoch=100,lr=0.001,Op=adam)/model.h5'
model_weights_path =
'./models/model(epoch=100,lr=0.001,Op=adam)/weights.h5'
model = load_model(model_path)
model.load_weights(model_weights_path)

def predict(file):
  x = load_img(file, target_size=(img_width,img_height))
  x = img_to_array(x)
  x = np.expand_dims(x, axis=0)
  array = model.predict(x)
  result = array[0]
  answer = np.argmax(result)
  if answer == 0:
    print("Label: cepot")
  elif answer == 1:
    print("Label: gatotkaca")
  elif answer == 2:
    print("Label: semar")

  return answer

cepot_t = 0
cepot_f = 0
gatotkaca_t = 0
gatotkaca_f = 0
semar_t = 0
semar_f = 0

for i, ret in enumerate(os.walk('./data/test/cepot')):
  for i, filename in enumerate(ret[2]):
    if filename.startswith("."):
      continue
    print("Label: Cepot")
    result = predict(ret[0] + '/' + filename)
    if result == 0:
      cepot_t += 1
    else:
      cepot_f += 1

for i, ret in enumerate(os.walk('./data/test/gatotkaca')):
  for i, filename in enumerate(ret[2]):
    if filename.startswith("."):
      continue
```

```
52      print("Label: Gatotkaca")
53      result = predict(ret[0] + '/' + filename)
54      if result == 1:
55        gatotkaca_t += 1
56      else:
57        gatotkaca_f += 1
58
59  for i, ret in enumerate(os.walk('./data/test/semar')):
60    for i, filename in enumerate(ret[2]):
61      if filename.startswith("."):
62        continue
63      print("Label: Semar")
64      result = predict(ret[0] + '/' + filename)
65      if result == 2:
66        semar_t += 1
67      else:
68        semar_f += 1
69
70  """
71  Check metrics
72  """
73  print("True Cepot: ", cepot_t)
74  print("False Cepot: ", cepot_f)
75  print("True Gatotkaca: ", gatotkaca_t)
76  print("False Gatotkaca: ", gatotkaca_f)
77  print("True Semar: ", semar_t)
78  print("False Semar: ", semar_f)
```