# Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltech 101

I Wayan Suartika E. P, Arya Yudhi Wijaya, dan Rully Soelaiman
Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
*e-mail*: arya@cs.its.ac.id

***Abstrak*— Deep Learning adalah sebuah bidang keilmuan baru dalam bidang Machine Learning yang akhir-akhir ini berkembang karena perkembangan teknologi GPU accelaration. Deep Learning memiliki kemampuan yang sangat baik dalam visi komputer. Salah satunya adalah pada kasus klasifikasi objek pada citra. Dengan mengimplementasikan salah satu metode machine learning yang dapat digunakan untuk klasifikasi citra objek yaitu CNN. Metode CNN terdiri dari dua tahap. Tahap pertama adalah klasifikasi citra menggunakan feedforward. Tahap kedua merupakan tahap pembelajaran dengan metode backpropagation. Sebelum dilakukan klasifikasi, terlebih dahulu dilakukan praproses dengan metode wrapping dan cropping untuk memfokuskan objek yang akan diklasifikasi. Selanjutnya dilakukan training menggunakan metode feedforward dan backpropagation. Terakhir adalah tahap klasifikasi menggunakan metode feedforward dengan bobot dan bias yang diperbarui. Hasil uji coba dari klasifikasi citra objek dengan tingkat confusion yang berbeda pada basis data Caltech 101 menghasilkan rata-rata nilai akurasi mencapai. Sehingga dapat disimpulkan bahwa metode CNN yang digunakan pada Tugas Akhir ini mampu melakukan klasifikasi dengan baik.**

***Kata Kunci*— Deep learning, Convolution Neural Network, Caltech 101.**

## I. PENDAHULUAN

SALAH satu problem dalam visi komputer yang telah lama dicari solusinya adalah klasifikasi objek pada citra secara umum. Bagaimana menduplikasi kemampuan manusia dalam memahami informasi citra, agar komputer dapat mengenali objek pada citra selayaknya manusia. Proses feature engineering yang digunakan pada ummumnya sangat terbatas dimana hanya dapat berlaku pada dataset tertentu saja tanpa kemampuan generalisasi pada jenis citra apapun. Hal tersebut dikarenakan berbagai perbedaan antar citra antara lain perbedaan sudut pandang, perbedaan skala, perbedaan kondisi pencahayaan, deformasi objek, dan sebagainya.

Kalangan akademisi yang telah lama bergelut pada problem ini. Salah satunya pendekatan yang berhasil digunakan adalah menggunakan Jaringan Syaraf Tiruan (JST) yang terinspirasi dari jaringan syaraf pada manusia. Konsep tersebut kemudian dikembangkan lebih lanjut dalam Deep Learning.

Pada tahun 1989, Yann LeCun dan teman-temannya berhasil melakukan klasifikasi citra kode zip menggunakan kasus khusus dari Feed Forward Neural Network dengan nama Convolution Neural Network (CNN)[1]. Karena keterbatasan perangkat keras, Deep Learning tidak dikembangkan lebih lanjut hingga pada tahun 2009 dimana Jurgen mengembangkan sebuah Recurrent Neural Network (RNN) yang mendapatkan hasil signifikan pada pengenalan tulisan tangan [2]. Semenjak itu, dengan berkembangnya komputasi pada perangkat keras Graphical Processing Unit (GPU), pengembangan DNN berjalan dengan pesat. Pada tahun 2012, sebuah CNN dapat melakukan pengenalan citra dengan akurasi yang menyaingi manusia pada dataset tertentu [3]. Dewasa ini, Deep Learning telah menjadi salah satu topik hangat dalam dunia Machine Learning karena kapabilitasnya yang signifikan dalam memodelkan berbagai data kompleks seperti citra dan suara.

Metode Deep Learning yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah Convolutional Neural Network (CNN) [4]. Hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual cortex manusia [5] sehingga memiliki kemampuan mengolah informasi citra. Namun CNN, seperti metode Deep Learning lainnya, memiliki kelemahan yaitu proses pelatihan model yang lama. Dengan perkembangan perangkat keras, hal tersebut dapat diatasi menggunakan teknologi General Purpose Graphical Processing Unit (GPGPU).

## II. DASAR TEORI

### A. *Convolutional Neural Network*

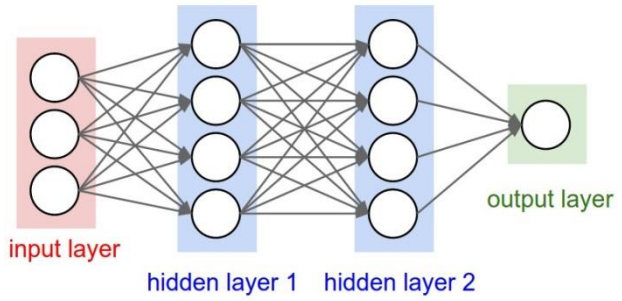
Convolutional Neural Network (CNN) adalah pengembangan dari Multilayer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Pada kasus klasifikasi citra, MLP kurang sesuai untuk digunakan karena tidak menyimpan informasi spasial dari data citra dan menganggap setiap piksel adalah fitur yang independen sehingga menghasilkan hasil yang kurang baik.

CNN pertama kali dikembangkan dengan nama NeoCognitron oleh Kunihiko Fukushima, seorang peneliti dari NHK Broadcasting Science Research Laboratories, Kinuta, Setagaya, Tokyo, Jepang [4]. Konsep tersebut kemudian dimatangkan oleh Yann LeChun, seorang peneliti dari AT&T Bell Laboratories di Holmdel, New Jersey, USA. Model CNN dengan nama LeNet berhasil diterapkan oleh LeChun pada penelitiannya mengenai pengenalan angka dan tulisan tangan [1]. Pada tahun 2012, Alex Krizhevsky dengan penerapan CNN miliknya berhasil menjuarai kompetisi ImageNet Large Scale Visual Recognition Challenge 2012. Prestasi tersebut menjadi momen pembuktian bahwa metode Deep Learning, khususnya CNN. Metode CNN terbukti berhasil

mengungguli metode Machine Learning lainnya seperti SVM pada kasus klasifikasi objek pada citra.

### B. *Konsep CNN*

Cara kerja CNN memiliki kesamaan pada MLP, namun dalam CNN setiap neuron dipresentasikan dalam bentuk dua dimensi, tidak seperti MLP yang setiap neuron hanya berukuran satu dimensi.



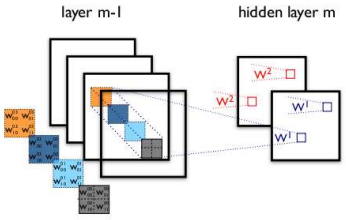

Gambar. 1. Arsitektur MLP Sederhana

Sebuah MLP seperti pada Gambar. **1**. memiliki *i layer* (kotak merah dan biru) dengan masing-masing *layer* berisi $j_i$ neuron (lingkaran putih). MLP menerima *input* data satu dimensi dan mempropagasikan data tersebut pada jaringan hingga menghasilkan *output*. Setiap hubungan antar neuron pada dua *layer* yang bersebelahan memiliki parameter bobot satu dimensi yang menentukan kualitas mode. Disetiap data *input* pada *layer* dilakukan operasi *linear* dengan nilai bobot yang ada, kemudian hasil komputasi akan ditransformasi menggunakan operasi *non linear* yang disebut sebagai fungsi aktivasi.

Pada CNN, data yang dipropagasikan pada jaringan adalah data dua dimensi, sehingga operasi linear dan parameter bobot pada CNN berbeda. Pada CNN operasi linear menggunakan operasi konvolusi, sedangkan bobot tidak lagi satu dimensi saja, namun berbentuk empat dimensi yang merupakan kumpulan kernel konvolusi seperti pada Gambar.2. Dimensi bobot pada CNN adalah:

*neuron input* x *neuron output* x *tinggi* x *lebar*

Karena sifat proses konvolusi, maka CNN hanya dapat digunakan pada data yang memiliki struktur dua dimensi seperti citra dan suara.




Gambar.2. Proses Konvolusi pada CNN

### C. *Arsiterktur Jaringan CNN*

JST terdiri dari berbagai layer dan beberapa neuron pada masing-masing layer. Kedua hal tersebut tidak dapat ditentukan menggunakan aturan yang pasti dan berlaku berbeda-beda pada data yang berbeda [7].

Pada kasus MLP, sebuah jaringan tanpa hidden layer dapat memetakan persamaan linear apapun, sedangkan jaringan dengan satu atau dua hidden layer dapat memetakan sebagian besar persamaan pada data sederhana. Namun pada data yang lebih kompleks, MLP memiliki keterbatasan. Pada permasalahan jumlah *hidden layer* dibawah tiga *layer*, terdapat pendekatan untuk menentukan jumlah neuron pada masing-masing *layer* untuk mendekati hasil optimal. Penggunaan layer diatas dua pada umumnya tidak direkomendasikan dikarenakan akan menyebabkan *overfitting* serta kekuatan *backpropagation* berkurang secara signifikan.

Dengan berkembangnya *deep learning*, ditemukan bahwa untuk mengatasi kekurangan MLP dalam menangani data kompleks, diperlukan fungsi untuk mentransformasi data input menjadi bentuk yang lebih mudah dimengerti oleh MLP. Hal tersebut memicu berkembangnya *deep learning* dimana dalam satu model diberi beberapa *layer* untuk melakukan transformasi data sebelum data diolah menggunakan metode klasifikasi. Hal tersebut memicu berkembangnya model neural network dengan jumlah *layer* diatas tiga. Namun dikarenakan fungsi *layer* awal sebagai metode ekstraksi fitur, maka jumlah *layer* dalam sebuah DNN tidak memiliki aturan universal dan berlaku berbeda-beda tergantung dataset yang digunakan.

Karena hal tersebut, jumlah *layer* pada jaringan serta jumlah neuron pada masing-masing *layer* dianggap sebagai *hyperparameter* dan dioptimasi menggunakan pendekatan *searching*.

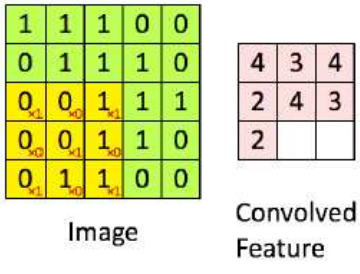
Sebuah CNN terdiri dari beberapa *layer*. Berdasarkan aristektur LeNet5 [8], terdapat empat macam *layer* utama pada sebuah CNN namun yang diterapkan pada TA ini hanya tiga macam lapisan lantara lain:

#### 1) *Convolution Layer*

*Convolution Layer* melakukan operasi konvolusi pada *output* dari *layer* sebelumnya. *Layer* tersebut adalah proses utama yang mendasari sebuah CNN.

Konvolusi adalah suatu istilah matematis yang berati mengaplikasikan sebuah fungsi pada *output* fungsi lain secara berulang. Dalam pengolahan citra, konvolusi berati mengaplikasikan sebuah *kernel*(kotak kuning) pada citra disemua *offset* yang memungkinkan seperti yang ditunjukkan pada Gambar.3. Kotak hijau secara keseluruhan adalah citra yang akan dikonvolusi. *Kernel* bergerak dari sudut kiri atas ke kanan bawah. Sehingga hasil konvolusi dari citra tersebut dapat dilihat pada gambar disebelah kanannya.

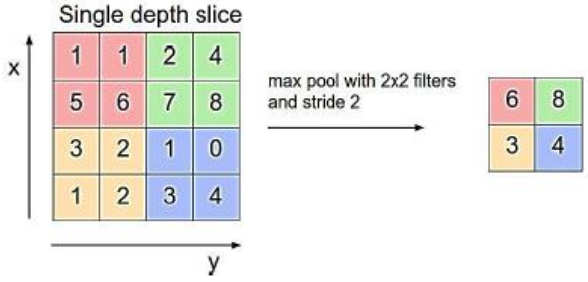
Tujuan dilakukannya konvolusi pada data citra adalah untuk mengekstraksi fitur dari citra *input*. Konvolusi akan menghasilkan transformasi *linear* dari data *input* sesuai informasi spasial pada data. Bobot pada *layer* tersebut menspesifikasikan *kernel* konvolusi yang digunakan, sehingga kernel konvolusi dapat dilatih berdasarkan *input* pada CNN.




Gambar.3. Operasi Konvolusi

*2) Subsampling Layer*

*Subsampling* adalah proses mereduksi ukuran sebuah data citra. Dalam pengolahan citra, *subsampling* juga bertujuan untuk meningkatkan invariansi posisi dari fitur. Dalam sebagian besar CNN, metode *subsampling* yang digunakan adalah *max pooling*. *Max pooling* membagi output dari *convolution layer* menjadi beberapa *grid* kecil lalu mengambil nilai maksimal dari setiap *grid* untuk menyusun matriks citra yang telah direduksi seperti yang ditunjukkan pada Gambar.4. Grid yang berwarna merah, hijau, kuning dan biru merupakan kelompok grid yang akan dipilih nilai maksimumnya. Sehingga hasil dari proses tersebut dapat dilihat pada kumpulan grid disebelah kanannya. Proses tersebut memastikan fitur yang didapatkan akan sama meskipun objek citra mengalami translasi (pergeseran).




Gambar.4. Operasi Max Pooling

Menurut Springenberg et al. [10], penggunaan *pooling layer* pada CNN hanya bertujuan untuk mereduksi ukuran citra sehingga dapat dengan mudah digantikan dengan sebuah *convolution layer* dengan *stride* yang sama dengan *pooling layer* yang bersangkutan.

*3) Fully Connected Layer*

Layer tersebut adalah *layer* yang biasanya digunakan dalam penerapan MLP dan bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear.

Setiap *neuron* pada *convolution layer* perlu ditransformasi menjadi data satu dimensi terlebih dahulu sebelum dapat dimasukkan ke dalam sebuah *fully connected layer*. Karena hal tersebut menyebabkan data kehilangan informasi spasialnya dan tidak reversibel, *fully connected layer* hanya dapat diimplementasikan di akhir jaringan.

Dalam sebuah jurnal oleh Lin et al., dijelaskan bahwa *convolution layer* dengan ukuran kernel 1 x 1 melakukan fungsi yang sama dengan sebuah *fully connected layer* namun dengan tetap mempertahankan karakter spasial dari data. Hal tersebut membuat penggunaan *fully connectedlayer* pada CNN sekarang tidak banyak dipakai.

### *D. Fungsi Aktivasi*

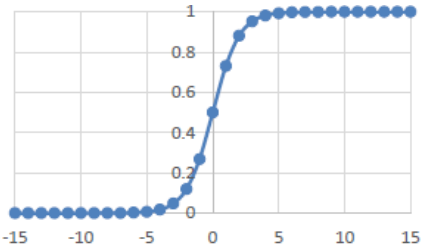
Fungsi aktivasi adalah fungsi *non linear* yang memungkinkan sebuah JST untuk dapat mentransformasi data *input* menjadi dimensi yang lebih tinggi sehingga dapat dilakukan pemotongan *hyperlane* sederhana yang memungkinkan dilakukan klasifikasi. Dalam CNN terdapat fungsi aktivasi digunakan yaitu fungsi *sigmoid*.

Fungsi *sigmoid* mentransformasi *range* nilai dari *input* x menjadi antara 0 dan 1 dengan bentuk distribusi fungsi seperti pada Gambar.5. Sehingga fungsi *sigmoid* memiliki bentuk sebagai berikut:

$$\sigma(x) = \frac{1}{(1 + e^{-x})} \quad (1)$$

Fungsi *sigmoid* sekarang sudah tidak banyak digunakan dalam praktek karena memiliki kelemahan utama yaitu *range* nilai *output* dari fungsi *sigmoid* tidak terpusat pada angka nol.

Hal tersebut menyebabkan terjadinya proses *backpropagation* yang tidak ideal, selain itu bobot pada JST tidak terdistribusi rata antara nilai positif dan negatif serta nilai bobot akan banyak mendekati ekstrim 0 atau 1. Dikarenakan komputasi nilai propagasi menggunakan perkalian, maka nilai ekstrim tersebut akan menyebabkan efek *saturating gradients* dimana jika nilai bobot cukup kecil, maka lama kelamaan nilai bobot akan mendekati salah satu ekstrim sehingga memiliki gradien yang mendekati nol. Jika hal tersebut terjadi, maka neuron tersebut tidak akan dapat mengalami *update* yang signifikan dan akan nonaktif.

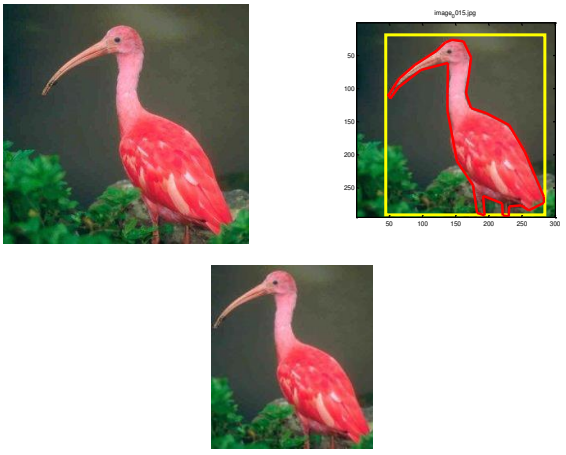



Gambar.5. Distribusi Fungsi Sigmoid

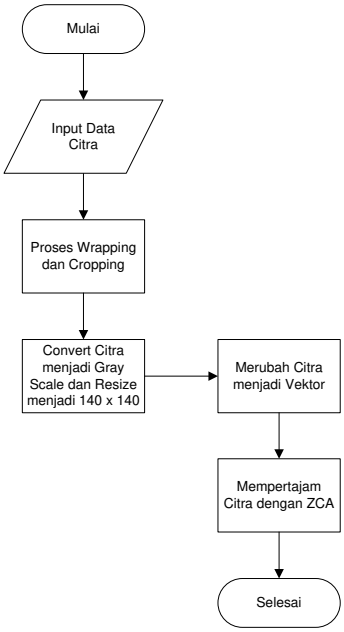
## III. DESAIN PERANGKAT LUNAK

### *A. Praproses dan Pengolahan Data Input*

Citra masukan akan diolah ke dalam pra proses yaitu proses *wrapping* dan *cropping*. Pada *wrapping*, citra masukan dilakukan pengecekan terhadap *edge* dari objek utama pada citra tersebut. Dari *edge* pada citra tersebut ditentukan *edge* maksimalnya sehingga saat hasil *cropping* objek pada citra tersebut tetap utuh seperti pada Gambar.6. Tahap *training* dimulai dengan merubah citra menjadi bentuk vektor. Sehingga alur proses pertama berbentuk seperti Gambar.7.



Gambar.6. Praproses Citra *Input*



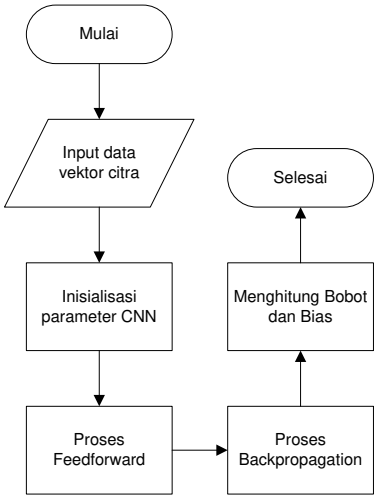

Gambar.7. Alur Praproses dan *Input*

Proses pengolahan data citra dimulai dengan citra ukuran sembarang yang kemudian dirubah ukurannya menjadi 140 x 140. Citra tersebut dijadikan *grey scale* agar bisa diproses dengan mudah pada tahap *Training*.

### *B. Proses Training*

Proses *training* merupakan tahapan dimana CNN dilatih untuk memperoleh akurasi yang tinggi dari klasifikasi yang dilakukan. Tahapan ini terdiri dari proses *feed forward* dan proses *backpropagation*. Untuk memulai proses *feedforward* diperlukan jumlah dan ukuran *layer* yang akan dibentuk, ukuran *subsampling*, citra vektor yang diperoleh pada subbab III.A. Proses *feedforward* bekerja seperti pada subbab II.C dimana citra vektor akan melalui proses konvolusi dan *Max pooling* untuk mereduksi ukuran citranya dan memperbanyak neuronnya. Sehingga terbentuk banyak jaringan yang mana menambah variant data untuk dipelajari.

Hasil dari proses *feedforward* berupa bobot yang akan digunakan untuk mengevaluasi proses neural network tadi. Alur prosesnya seperti pada Gambar.8.




Gambar.8. Alur Proses Training

*1) Proses Feedforward*

Proses *feed forward* merupakan tahap pertama dalam proses *training*. Proses ini akan menghasilkan beberapa lapisan untuk mengklasifikasi data citra yang mana menggunakan bobot dan bias yang telah diperbarui dari proses *backpropagation*. Tahap ini juga akan digunakan kembali saat proses *testing*.

*2) Proses Backpropagation*

Proses *backpropagation* merupakan tahap kedua dari proses *training*. Pada tahap ini seperti yang telah dijelaskan pada sub bab 2.3.6 hasil proses dari *feed forward* di-*trace* kesalahannya dari lapisan *output* sampai lapisan pertama. Untuk menandai bahwa data tersebut telah di-*trace* diperoleh bobot dan bias yang baru.

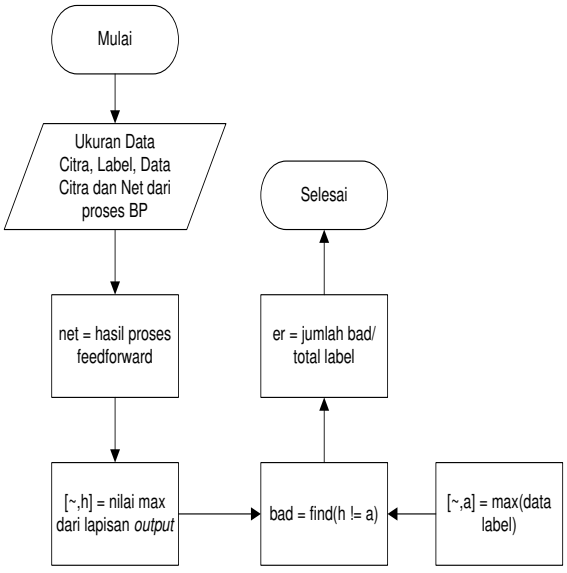
*3) Perhitungan Gradient*

Pada proses *gradient* untuk jaringan konvolusi merupakan proses untuk memperoleh nilai bobot dan bias yang baru yang akan diperlukan saat *training* dan

### *C. Proses Testing*

Proses *testing* merupakan proses klasifikasi menggunakan bobot dan bias dari hasil proses *training*. Proses ini tidak jauh berbeda dengan proses *training* yang membedakannya tidak terdapat proses *backpropagation* setelah proses *feedforward*. Sehingga hasil akhir dari proses ini menghasilkan akurasi dari klasifikasi yang dilakukan, data yang gagal diklasifikasi, nomor citra yang gagal diklasifikasi, dan bentuk network yang terbentuk dari proses *feedforward*.

Dengan bobot dan bias yang baru proses *feedforward* diterapkan yang kemudian menghasilkan lapisan *output*. Lapisan *output* sudah *fully connected* dengan label yang disediakan. Hasil *fully connected* tersebut diperoleh data yang gagal dan berhasil diklasifikasi. Dari penjelasan di atas bentuk alur proses *Testing* berbentuk seperti pada Gambar.9.




Gambar.9. Alur Proses Testing

## IV. UJI COBA DAN EVALUASI

Uji coba dalam artikel ini dilakukan pada 390 citra pada data Caltech-101 yang didapat dari 150 citra adalah kategori Unggas dan sisanya adalah Crocodile, Cougar, dan Face masing-masing dari ketiga kategori tersebut 80 citra.

### *A. Hasil Klasifikasi*

Salah satu contoh hasil klasifikasi dari aplikasi ini ditunjukkan pada Gambar. Pada klasifikasi tersebut, terdapat

5 Kategori unggas, yaitu: Emu, Flamingo, Ibis, Pigeon, dan Rooster yang terdiri dari 150 citra. Selain Kategori tersebut terdapat 3 Kategori lainnya, yaitu: Cougar (Cougar Body dan Cougar Face), Crocodile (Crocodile dan Crocodile Head) dan Face (Face dan Face Easy).

Hasil klasifikasi 5 Kategori Unggas ditunjukkan pada Tabel.1. Menunjukkan bahwa persentase keberhasilan 20%.Tabel.2 menunjukkan persentase keberhasilan 50% untuk kategori Cougar, Crocodile, dan Face.

Tabel.1 Hasil Klasifikasi Kategori Unggas

| Kategori | Berhasil | | Gagal | |
|---|---|---|---|---|
| | Jumlah Data | Prosentase (%) | Jumlah Data | Prosentase (%) |
| Emu | 0 | 0 | 10 | 100 |
| Flamingo | 0 | 0 | 10 | 100 |
| Ibis | 0 | 0 | 10 | 100 |
| Pigeon | 0 | 0 | 10 | 100 |
| Rooster | 10 | 100 | 0 | 0 |
| Hasil Klasifikasi | 10 | 20 | 40 | 80 |
| Nilai Error Training | | | | |
| Rata-Rata Min | 0.1224 | | | |
| Rata-Rata Max | 0,9043 | | | |
| Rata- Rata Waktu (detik) | | | | |
| Training | 9.754,17 | | | |
| Testing | 16,28 | | | |

Tabel.2 Hasil Klasifikasi Kategori Cougar, Crocodile dan Face

| Kategori | Hasil Klasifikasi dengan Training 60 Citra | | Hasil Klasifikasi dengan Training 20 Citra | |
|---|---|---|---|---|
| | Jumlah Data (Berhasil) | Prosentase (%) | Jumlah Data (Berhasil) | Prosentase (%) |
| Cougar Body | 30 | 100 | 10 | 100 |
| Cougar Face | 0 | 0 | 0 | 0 |
| Crocodile | 30 | 100 | 0 | 0 |
| Crocodile Head | 0 | 0 | 10 | 100 |
| Face | 30 | 100 | 0 | 0 |
| Face Easy | 0 | 0 | 10 | 100 |
| Hasil Klasifikasi | 90 | 50 | 30 | 50 |
| Nilai Error Training | | | | |
| Rata-Rata Min | 0.0171 | | | |
| Rata-Rata Max | 0,3458 | | | |
| Rata- Rata Waktu (detik) | | | | |
| Training | 6.115,075 | | | |
| Testing | 10,19 | | | |

Peresentase terendah (20%) dihasilkan dari kategori Unggas yang disebabkan Kategori Ebis, Flamingo, Pigeon dan Emu tidak dapat diklasifikasi dengan baik.

Secara gabungan, klasifikasi citra yang dilakukan menghasilkan persentase kebenaran antara 20% sampai 50%. Ditunjukkan pada Tabel.1 dan Tabel.2.

## V. KESIMPULAN/RINGKASAN

Metode praproses dan metode klasifikasi dengan menggunakan Convolutional Neural Network cukup handal untuk menentukan kebenaran dari klasifikasi citra objek. Hal ini terbukti dengan hasil akurasi sebesar 20% - 50%.

Perubahan tingkat confusion tidak mempengaruhi hasil akurasi. Hal ini membuktikan bahwa klasifikasi menggunakan metode CNN relatif handal terhadap perubahan parameter yang dilakukan. Dengan menggunakan data *training* yang baik dan optimal, maka subset dari data *training* tersebut juga akan menghasilkan klasifikasi yang baik.

## UCAPAN TERIMA KASIH

Penulis S.E.P mengucapkan terima kasih kepada Tuhan Y.M.E yang telah melimpahkan rahmat-Nya sehingga penulis dapat menyelesaikan artikel ini. Penulis juga mengucapkan terima kasih kepada Bapak Rully Soelaiman dan Bapak Arya Yudhi Wijaya yang telah membimbing penulis dalam mengerjakan artikel ini.

## DAFTAR PUSTAKA

[1] [1] Y. LeCun, "Handwritten Digit Recognition with a Back-Propagation Network," 1990.
[2] [2] Wikipedia, "Feature Learning," [Online]. Available: http://en.wikipedia.org/wiki/Feature_learning.
[3] [3] A.Coates, H.Lee and A.Y. Ng, "An Analysis of Single-Layer Networks in Unsupervised Feature Learning," 2011.
[4] [4] K. Fukushima, "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position," Biological Cybernetics,1980.
[5] [5] A.Karpathy, "CS231n Convolutional Neural Network for Visual Recognition, "Stanford University, [Online].Available: http://cs231m.github.io/.
[6] [6] LISA Lab, "Deep Learning Tutorial," [Online]. Available: http://deeplearning.net/tutorial/contents.html.
[7] [7] D. Stathakis, "How Many Hidden Layers And Nodes?," International Journal of Remote Sensing, 2008.
[8] [8] Stanford University, "An Introduction to Convolutional Neural Network," Vision Imaging Science and Technology Lab, Stanford University, [Online]. Available: http://white.stanford.edu/teach/index.php/An_Introduction_to_Convolutional_Neural_Networks.
[9] [9] A. Ng, J. Ngiam, C. Yu Foo, Y. Mai and C. Suen, "Unsupervised Feature Learning and Deep Learning Tutorial," Stanford University, [Online]. Available: http://ufdl.stanford.edu/wiki/index.php/UFLDL_Tutorial.
[10] [10] J. T. Springenberg, A. Dosovitskiy, T. Brox and M. Riedmiller, "Striving For Simplicity: The All Convolutional Net," ICLR 2015, 2015.
[11] [11] C.Olah, "Neural Networks, Manifolds, and Topology," [Online]. Available: http://cola.github.io/posts/2014-03-NN-Manifolds-Topology/.
[12] [12] Kristen Grauman, Trevor Darrell, "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features," Cambridge, MA, USA, 2005.
[13] [13] Hao Zhang, Alexander C.Berg, Michael Maire, Jitendra Malik, "SVM-KNN: Discriminative Nearest Neighbor Classfication for Visual Category Recognition," University of California, Berkeley, CA 94720.
[14] [14] Alexander C. Berg, Tamara L.Berg, Jitendra Malik, "Shape Matching and Object Recognition using Low Distorion Correspondences," U.C.Berkeley.