Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network

Sarirotul Ilahiyah[1)], Agung Nilogiri[2)]

[1,2)]Jurusan Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Jember
Email : [1)] ila.ilahiyah@gmail.com, [2)] agungnilogiri@unmuhjember.ac.id

ABSTRAK

Convolutional Neural Network adalah salah satu algoritma *Deep Learning* yang merupakan pengembangan dari *Multilayer Peceptron* (MLP) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN dibuat dengan prinsip *translation invariance* yaitu dapat mengenali objek dalam citra pada berbagai macam posisi yang mungkin. Terdapat 2000 citra daun yang diklasifikasi menggunakan Alexnet. Alexnet merupakan arsitektur CNN milik Krizhevsky yang memiliki delapan layer ekstraksi fitur. Layer tersebut terdiri dari lima layer konvolusi dan tiga *pooling layer*. Dalam layer klasifikasinya, Alexnet mempunyai dua layer *Fully Connected* yang masing-masing mempunyai 4096 neuron. Pada akhir layer terdapat pengklasifikasian kedalam 20 kategori menggunakan aktifasi *softmax*. Rata-rata akurasi dari hasil klasifikasi mencapai 85%. Sedangkan akurasi dari identifikasi berhasil mencapai 90% yang didapatkan dari pengujian 40 citra.

Kata Kunci : *Deep Learning, Convolutional Neural Network, Alexnet.*

1. PENDAHULUAN

Data yang masif atau biasa disebut *Big Data* memiliki ciri berukuran besar, variatif, pertumbuhannya yang cepat dan mungkin tidak terstruktur akan susah diolah dengan pendekatan konvensional. Kelemahan lain pendekatan konvensional adalah keterbatasan dalam mengolah target kelas yang banyak dan hanya dapat mengenali objek yang berada di tengah citra atau tidak dapat mengenali objek dalam citra pada berbagai macam posisi yang mungkin (*translation invariance*). Selain variasi posisi objek, ada juga kendala lain seperti rotasi objek dan perbedaan ukuran (*scalling*). Machine Learning bisa juga mempelajari prinsip *translation invariance*, tetapi memerlukan jauh lebih banyak parameter dibanding *deep learning* yang memang dibuat dengan prinsip *translation invariance* (*built-in*) (Putra, 2018).

Deep Learning adalah cabang ilmu *machine learning* berbasis Jaringan Saraf Tiruan (JST) atau bisa dikatakan sebagai perkembangan dari JST. Dalam *deep learning*, sebuah komputer belajar mengklasifikasi secara langsung dari gambar atau suara. *Convolutional Neural Network* (CNN/ConvNet) adalah salah satu algoritma *deep learning* yang merupakan pengembangan dari *Multilayer Perceptron* (MPL) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN dapat belajar langsung dari citra sehingga mengurangi beban dari pemrograman.

Selain itu, metode *machine learning* konvensional hanya mengandalkan CPU dan RAM dalam proses komputasi, sehingga spesifikasi CPU dan RAM menentukan kecepatan komputasi. Sedangkan metode *deep learning*, selain

menggunakan CPU dan RAM dalam proses komputasi, metode ini juga memanfaatkan kemampuan GPU sehingga proses komputasi data yang besar dapat berlangsung lebih cepat.

Dengan menggunakan CNN, dataset citra daun dari Neeraj Kumar yang mempunyai 20 jenis genus tumbuhan dan masing-masing genus memiliki seratus citra daun beresolusi tinggi dapat diidentifikasi lebih mudah. Kemudahan ini karena CNN dapat mengenali objek dalam citra pada berbagai macam posisi yang mungkin (*translation invariance*).

2. TINJAUAN PUSTAKA

2.1 Convolutional Neural Network

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

Lapisan-lapisan CNN memiliki susunan *neuron* 3 dimensi (lebar, tinggi, kedalaman). Lebar dan tinggi merupakan ukuran lapisan, sedangkan kedalaman mengacu pada jumlah lapisan. Sebuah CNN dapat memiliki puluhan hingga ratusan lapisan yang masing-masing belajar mendeteksi berbagai gambar. Pengolahan citra diterapkan pada setiap citra latih pada resolusi yang berbeda, dan output dari masing-masing gambar yang diolah dan digunakan sebagai input ke lapisan berikutnya. Pengolahan citra dapat dimulai sebagai fitur yang sangat sederhana seperti kecerahan dan tepi atau meningkatkan kompleksitas pada fitur yang secara unik menentukan objek sesuai ketebalan lapisan (Mathworks, 2017).

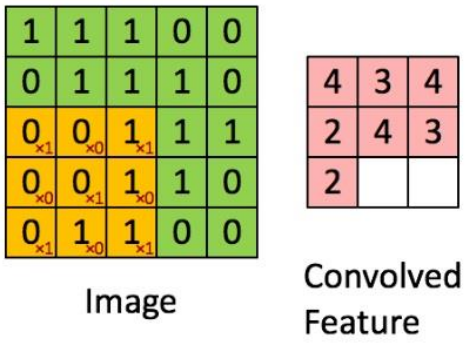
Secara umum tipe lapisan pada CNN dibagi menjadi dua. Lapisan pertama adalah lapisan ekstraksi fitur (*feature extraction layer*), letaknya berada pada awal arsitektur tersusun atas beberapa lapisan dan setiap lapisan tersusun atas *neuron* yang terkoneksi pada daerah lokal (*local region*) dari lapisan sebelumnya. Lapisan jenis pertama adalah *convolutional layer* dan lapisan kedua adalah *pooling layer*. Setiap lapisan diberlakukan fungsi aktivasi dengan posisinya yang berselang-seling antara jenis pertama dengan jenis kedua. Lapisan ini menerima *input* gambar secara langsung dan memprosesnya hingga menghasilkan *output* berupa vektor untuk diolah pada lapisan berikutnya. Lapisan kedua adalah lapisan klasifikasi (*classfication layer*), tersusun atas beberapa lapisan dan setiap lapisan tersusun atas *neuron* yang terkoneksi secara penuh (*fully connected*) dengan lapisan lainnya. Layer ini menerima input dari hasil keluaran layer ekstrasi fitur gambar berupa vektor, kemudian ditransformasikan seperti *Multi Neural Networks* dengan tambahan beberapa *hidden layer*. Hasil keluaran berupa akurasi kelas untuk klasifikasi.

2.2 Operasi Konvolusi

Konvolusi didefinisikan sebagai proses untuk memperoleh suatu piksel didasarkan pada nilai piksel itu sendiri dan tetangganya dengan melibatkan suatu matriks yang disebut kernel yang merepresentasikan pembobotan (Kusumanto et al. 2011). Operasi ini menerapkan fungsi *output* sebagai *Feature Map* dari masukan citra. Masukan

p-ISSN : 2502-5724; e-ISSN : 2541-5735

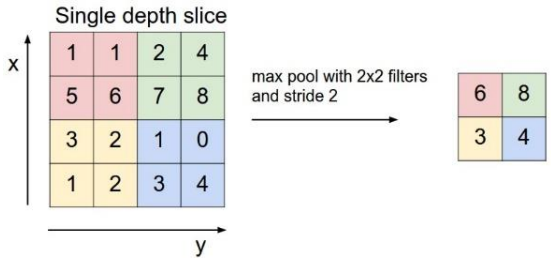
dan keluaran ini dapat dilihat sebagai dua argumen bernilai riil (Goodfellow et al 2016). Operasi konvolusi ini ditunjukkan dalam Gambar 1.



Gambar 1. Operasi Konvolusi

2.3 Pooling Layer

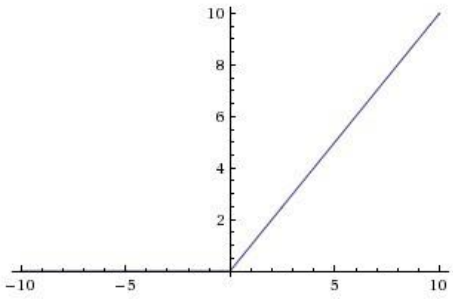
Pooling Layer adalah lapisan yang menggunakan fungsi dengan *Feature Map* sebagai masukan dan mengolahnya dengan berbagai macam operasi statistik berdasarkan nilai piksel terdekat. Pada model CNN, lapisan *Pooling* biasanya disisipkan secara teratur setelah beberapa lapisan konvolusi. Lapisan *Pooling* 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, dan untuk mengendalikan *Overfitting*. Lapisan *Pooling* bekerja di setiap tumpukan *Feature Map* dan mengurangi ukurannya. Bentuk lapisan *Pooling* yang paling umum adalah dengan menggunakan filter berukuran 2x2 yang diaplikasikan dengan langkah sebanyak 2 dan kemudian beroperasi pada setiap irisan dari input. Bentuk seperti ini akan mengurangi *Feature Map* hingga 75% dari ukuran aslinya. (Prijono 2018). Contoh operasi *Max Pooling* ditunjukkan dalam Gambar 2.



Gambar 2. Contoh Operasi Max Pooling

2.4 Aktivasi ReLu

Aktivasi ReLU (*Rectified Linear Unit*) merupakan lapisan aktivasi pada model CNN yang mengaplikasikan fungsi f(x)=max(0,x) yang berarti fungsi ini melakukan *thresholding* dengan nilai nol terhadap nilai piksel pada input citra. Aktivasi ini membuat seluruh nilai piksel yang bernilai kurang dari nol pada suatu citra akan dijadikan 0.



Gambar 3. Aktivasi ReLu

2.5 Fully-Connected Layer

Layer tersebut adalah lapisan 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.

2.6 Aktivasi Softmax

Aktivasi Softmax atau *Softmax Classifier* merupakan bentuk lain dari algoritma *Logistic Regression* yang dapat digunakan untuk mengklasifikasi lebih dari dua kelas. Standar klasifikasi yang biasa dilakukan oleh algoritma *Logistic Regression* adalah tugas untuk klasifikasi kelas biner. Pada Softmax bentuk persamaan yang muncul adalah sebagai berikut ini.

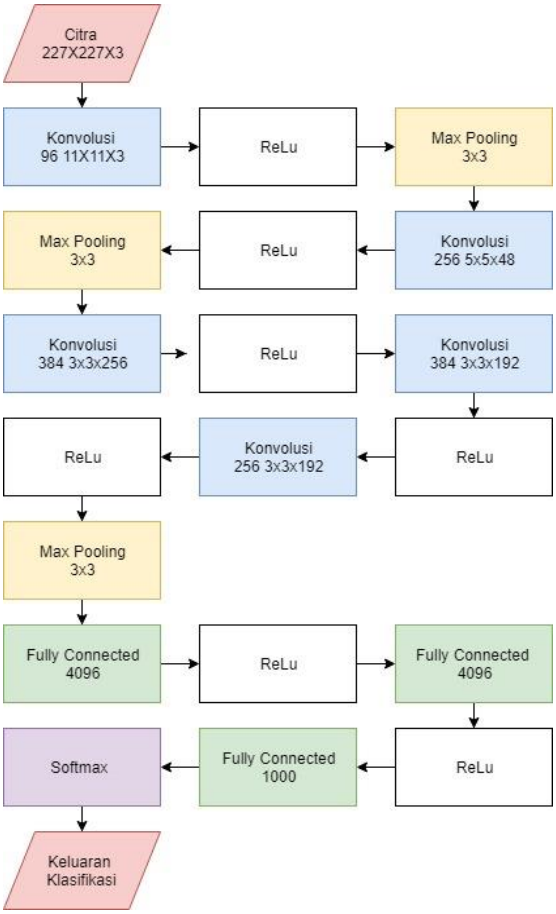
$$f_j(Z) = \frac{e^{zJ}}{\sum_k e^{zk}} \quad (1)$$

Notasi f_j 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 lainnya. Softmax memungkinkan 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. METODE PENELITIAN

Pada penelitian implementasi *deep learning* pada identifikasi jenis tumbuhan berdasarkan citra daun menggunakan *Convolutional Neural Network* (CNN) ini digunakan arsitektur CNN dari Krizhevsky et al.(2012) yang disebut dengan AlexNet.

Secara garis besar, cara kerja sistem arsitektur Alexnet dibagi menjadi dua kelompok *layer*. Pertama adalah *layer* ekstrasi fitur yang tersusun dari *layer* konvolusi dan *pooling layer*, dan kedua adalah layer klasifikasi.



Gambar 4. Arsitektur Alexnet

Jika di susun, arsitektur AlexNet terdiri dari 20 lapisan yang rinciannya sebagai berikut ini.

1) Masukan citra : 227x227x3
   Masukan citra berukuran 227x227x3. Jika citra masukan mempunyai ukuran yang berbeda, maka pada tahap pertama ini citra akan di *reshape* menjadi 227x227x3.
2) *Convolution* : 96 11x11x3 dengan *stride* [4 4] dan *padding* [0 0]
   Pada layer kedua terdapat proses konvolusi pertama dengan ukuran kernel 11x11x3 sebanyak 96 kali, dengan *stride* 4 dan *padding* 0. Lapisan konvolusi yang berukuran 11x11x3 sebanyak 96 ini pada awal jaringan menangkap fitur citra dasar, seperti tepian dan gumpalan Keluaran

p-ISSN : 2502-5724; e-ISSN : 2541-5735

dari layer ini berukuran 55x55x96 yang dapat dihitung menggunakan rumus Dimensi Keluaran seperti berikut :

$$Nout = \left[\frac{Nin + 2p - k}{s}\right] + 1$$
$$Nout = \left[\frac{227 + 2.0 - 11}{4}\right] + 1 = 55$$

Untuk menghitung parameter dari layer ini dapat dihitung dengan cara :

55x55x96= 290400 *neuron*

Masing-masing memiliki 11x11x3 = 363 bobot + 1 bias

Total parameter layer ini = 290400 x 364 = 105,705,600 parameter

3) ReLU

ReLU merupakan lapisan aktivasi pada model CNN yang mengaplikasikan fungsi f(x)=max(0,x) yang berarti fungsi ini melakukan *thresholding* dengan nilai nol terhadap nilai piksel pada input citra.

4) *Max pooling* : 3x3 *stride* [2 2] dan *padding* [0 0]

Pooling layer digunakan untuk mengurangi ukuran dan mempercepat perhitungan, serta membuat beberapa fitur yang diperkirakan sedikit lebih akurat (Prijono 2018). Dimensi keluaran dari layer ini berukuran 27x27x96 yang didapatkan dari :

$$Nout = \left[\frac{55 + 2.0 - 3}{2}\right] + 1 = 27$$

5) *Convolution* : 256 5x5x48 dengan *stride* [1 1] dan *padding* [2 2]

Proses konvolusi dengan ukuran kernel 5x5x48 sebanyak 256, dengan *stride* 1 dan *padding* 2. Layer konvolusi kedua yang berukuran 5x5x48 sebanyak 256 *mask* merespon sudut dan konjungsi tepi warna Keluaran dari layer ini berukuran 27x27x256 yang dapat dihitung menggunakan rumus Dimensi Keluaran seperti berikut :

$$Nout = \left[\frac{27 + 2.2 - 5}{1}\right] + 1 = 27$$

6) ReLU

7) *Max Pooling* : 3x3 dengan *stride* [2 2] dan *padding* [0 0]

Dimensi keluaran dari layer ini berukuran 13x13x256 yang didapatkan dari :

$$Nout = \left[\frac{27 + 2.0 - 3}{2}\right] + 1 = 13$$

8) *Convolution* : 384 3x3x256 dengan *stride* [1 1] dan *padding* [1 1]

Proses konvolusi dengan ukuran kernel 3x3x256 sebanyak 384, dengan stride 1 dan padding 1. Layer konvolusi ketiga yang memiliki ukuran kernel 3x3x256 menangkap tekstur. Keluaran dari layer ini berukuran 13x13x384 yang dapat dihitung menggunakan rumus Dimensi Keluaran seperti berikut :

$$Nout = \left[\frac{13 + 2.1 - 3}{1}\right] + 1 = 13$$

9) ReLU

10) *Convolution* : 384 3x3x192 dengan *stride* [1 1] dan *padding* [1 1]

Proses konvolusi dengan ukuran kernel 3x3x192 sebanyak 384, dengan stride 1 dan padding 1. Layer konvolusi keempat dengan ukuran kernel 3x3x192 menunjukkan menunjukkan bagian-bagian objek Keluaran dari layer ini berukuran 13x13x384 yang dapat dihitung menggunakan rumus Dimensi Keluaran seperti berikut :

$$Nout = \left[\frac{13 + 2.1 - 3}{1}\right] + 1 = 13$$

11) ReLU

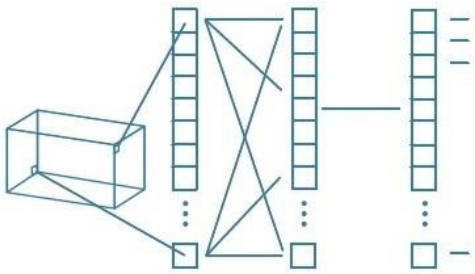
12) *Convolution* : 256 3x3x192 dengan *stride* [1 1] dan *padding* [1 1]
Proses konvolusi dengan ukuran kernel 3x3x192 sebanyak 256, dengan *stride* 1 dan *padding* 1. Layer konvolusi kelima dengan ukuran kernel 3x3x192 menunjukkan seluruh objek. Keluaran dari layer ini berukuran 13x13x384 yang dapat dihitung menggunakan rumus Dimensi Keluaran seperti berikut :

$$Nout = \left[\frac{13 + 2.1 - 3}{1}\right] + 1 = 13$$

13) ReLU
14) *Max Pooling* : 3x3 dengan *stride* [2 2] dan *padding* [0 0]
Dimensi keluaran dari layer ini berukuran 6x6x256 yang didapatkan dari :

$$Nout = \left[\frac{13 + 2.0 - 3}{2}\right] + 1 = 6$$

15) *Fully Connected* : 4096 layer
Untuk dapat diproses dalam *Fully Connected Layer*, masukan yang berukuran 6x6x256 harus di *reshape* menjadi ukuran yang kedalamannya hanya sebanyak satu layer. Maka, ukuran yang akan diproses pada layer ini adalah 9216x1 yang didapatkan dari perkalian 6x6x256.



Gambar 5. Fully Connected Layer

16) ReLU
17) *Fully Connected* : 4096 layer
18) ReLU
19) *Fully Connected* : 1000 layer
20) Softmax
Aktifasi *Softmax* atau *Softmax Classifier* merupakan aktifasi yang sering dipakai untuk menangani kasus *multiclass classification*, karena *output layer* biasanya memiliki lebih dari satu *neuron*.
21) Keluaran klasifikasi

4. HASIL DAN PEMBAHASAN

Dalam tahap ini dilakukan pengujian algoritma dengan menggunakan *cross validation*. *Cross validation* adalah metode statistik untuk mengevaluasi dan membandingkan belajar algoritma dengan membagi data menjadi dua segmen, satu segmen digunakan untuk belajar atau melatih data, dan yang lain digunakan untuk memvalidasi model. Dalam *cross validation* set pelatihan dan validasi harus *crossover* berturut-turut sehingga setiap data memiliki kesempatan tervalidasi.

Pengujian ini membagi dataset menjadi sepuluh folder yang didalamnya terdapat 2000 citra yang dibagi menjadi citra uji dan citra latih secara acak. Citra latih pada masing-masing folder berjumlah 1800 citra, sedangkan citra uji berjumlah 200 citra. Setiap folder dilakukan tiga kali percobaan dan dihitung akurasinya. Hasil pengujian dengan menggunakan *K-Fold Cross Validation* ini dapat dilihat pada Tabel 1.

Diantara tiga kali percobaan, *Fold* 10 mendapatkan rata-rata tertinggi dengan nilai akurasi 88,6 persen. Sedangkan nilai rata-rata akurasi terkecil didapatkan dari *Fold* 4 dengan nilai akurasi sebesar 83 persen. Presisi tertinggi didapatkan pada *Fold* 7 yang mendapatkan nilai akurasi sebesar 90 persen pada percobaan pertama dari tiga percobaan. Akurasi sistem yang didapat dari metode *K-Fold*

p-ISSN : 2502-5724; e-ISSN : 2541-5735

Cross Validation sebesar 85 persen yang didapat dari hasil rata-rata seluruh akurasi *Fold* pada setiap percobaan. *Fold* 10, sebagai *fold* yang mempunyai nilai akurasi rata-rata tertinggi digunakan sebagai data latih pada percobaan sekenario berikutnya.

Tabel 1. Akurasi Hasil Percobaan dengan Metode *K-Fold Cross Validation*

Fold	Percobaan-1 (%)	Percobaan-2 (%)	Percobaan-3 (%)	Rata-rata (%)
1	84,5	84,5	83,5	84,1
2	85,5	84	86	85,1
3	86	85,5	84,5	85,3
4	84	83	82	83
5	83	86	84,5	84,5
6	85,5	86,5	89	87
7	90	82,5	85	85,8
8	84,5	86,5	82	84,3
9	83	84,5	85	84,1
10	89,5	89	87,5	88,6
Rata-rata	85,55	85,2	84,9	85,2

Pada percobaan kedua, citra uji menggunakan citra baru yang tidak digunakan pada sistem. Jumlah citra yang digunakan pada percobaan kedua ini sebanyak 40 citra yang mewakili 20 kategori genus. Jadi, tiap-tiap genus mempunyai dua citra yang akan diuji. Hasil percobaan kedua ini dapat dilihat pada Tabel 2.

Tabel 2. Hasil Percobaan dengan Citra Uji Baru

Percobaan	Prediksi Benar	Prediksi Salah	Akurasi
1	36	4	90 %
2	38	2	95 %
3	35	5	87.5%
Rerata			90,8 %

Hasil pengujian 40 citra baru yang dilakukan sebanyak tiga kali percobaan mendapatkan nilai akurasi yang cukup tinggi. Percobaan pertama mendapatkan prediksi benar sebanyak 36 citra dari 40 citra uji sehingga mendapatkan nilai akurasi sebesar 90 persen. Pada percobaan kedua hanya didapati dua prediksi salah dari 40 citra uji dan menghasilkan nilai akurasi 95 persen. Sedangkan pada percobaan ketiga menghasilkan nilai akurasi sebesar 87 persen. Dari ketiga percobaan tersebut, didapatkan nilai akurasi sistem sebesar 90,8 persen.

5. KESIMPULAN DAN SARAN

Dari hasil pengamatan selama perancangan, implementasi, dan proses uji coba sistem, dapat diambil kesimpulan yaitu : (1) Percobaan menggunakan *fold cross validation* dengan nilai *k=10* didapatkan presisi tertinggi pada percobaan pertama dari fold ke 7 yakni dengan nilai akurasi sebesar 90%, (2) Tingkat rata-rata akurasi klasifikasi sistem yang didapatkan dari percobaan *fold cross validation* dengan nilai *k=10* yakni sebesar 85,21%, (3) Sistem yang telah dibuat dapat mengidentifikasi jenis genus tumbuhan dengan nilai akurasi sistem sebesar 90,8%.

DAFTAR PUSTAKA

Ghifary, Muhammad. 2015. Deep Convolutional Neural Network. Diakses Maret 2, 2018. https://ghifar.wordpress.com/2015/07/21/deep-convolutional-neural-networks-part-1/.

Gonzalez, Rafael C, and Richard E Woods. 2008. Digital Image Processing Third Edition. 3rd ed. New Jersey: Pearson Prentice Hall.

Goodfellow et al. 2016. Deep Learning. Diakses Maret 1, 2018. http://www.deeplearningbook.org/.

Kadir, Abdul, and Adhi Susanto. 2013. Teori Dan Aplikasi Pengolahan Citra Digital. Yogyakarta: Andi.

Krizhevsky, Alex, Ilya Sutskever, and Hinton Geoffrey E. 2012. "ImageNet Classification with Deep Convolutional Neural Networks." Advances in Neural Information Processing Systems 25 (NIPS2012), 1–9. https://doi.org/10.1109/5.726791.

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I., & Soares, V. B. (2012). Leafsnap : A Computer Vision System for Automatic Plant Species Identification, 1–14.

Kusumanto, R D, Alan Novi Tompunu, Setyo Pambudi, Jurusan Teknik Komputer, and Politeknik Negeri Sriwijaya. 2011. "Klasifikasi Warna Menggunakan Pengolahan Model Warna HSV" 2 (2): 83–87.

Lecun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning" 521. https://doi.org/10.1038/nature14539.

Prasetyo, Eko. 2011. Pengolahan Citra Digital dan Aplikasinya Menggunakan Matlab. Yogyakarta: Andi.

Prijono, Benny. 2018. Convolutional Neural Networks (CNN) Introduction. 7 Maret. Diakses Maret 29, 2018. https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/.

Putra, Jan Wira Gotama. 2018. Pengenalan Konsep Pembelajaran Mesin Dan Deep Learning. 1.0. Tokyo: Tokyo Institute of Technology.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." Journal of Machine Learning Research 15: 1929–58. https://doi.org/10.1214/12-AOS1000.

Wei, Donglai, Bolei Zhou, Antonio Torrabla, and William Freeman. n.d. "Understanding Intra-Class Knowledge Inside CNN" 6 (2): 6–12. http://vision03.csail.mit.edu/cnn_art/.

Zeiler, Matthew D, and Rob Fergus. 2014. "Visualizing and Understanding Convolutional Networks" 1: 818–33. http://vision03.csail.mit.edu/cnn_art/.