p-ISSN: 2620-3383
e-ISSN: 2528-6544

Deteksi Ras Kucing Menggunakan Compound Model Scaling Convolutional Neural Network

Nadia Azahro Choirunisa[1]
Tita Karlita[2]
Rengga Asmara[3]

Program Studi D3 Teknik Informatika[1], Departemen Teknik Informatika dan Komputer[2,3]
Politeknik Elektronika Negeri Surabaya
Kampus PENS, Jalan Raya ITS Sukolilo, Surabaya, Indonesia 60111
Email: nadiachoirunisa23@gmail.com[1], tita@pens.ac.id[2], rengga@pens.ac.id[3]

Notifikasi Penulis
12 Agustus 2021
Akhir Revisi
23 September 2021
Terbit
01 Februari 2022

Azahro Choirunisa, N., Karlita, T., & Asmara, R. Deteksi Ras Kucing Menggunakan Compound Model Scaling Convolutional Neural Network. Technomedia Journal, 6(2), 236–251.

https://doi.org/10.33050/tmj.v6i2.1704

ABSTRAK

Kucing merupakan hewan yang sangat popular di dunia. Jumlah dari ras kucing di dunia hanya sekitar 1% saja, sehingga didominasi oleh ras campuran maupun kucing domestik. Namun demikian, ada begitu banyak jenis ras kucing di dunia, sehingga terkadang sulit untuk mengidentifikasinya. Oleh karena itu, dibutuhkan sistem yang dapat mengenali jenis-jenis ras kucing. Dalam penelitian ini, penulis menggunakan salah satu metode deep learning yang dapat mengenali dan mengklasifikasikan suatu objek, yaitu Neural Convolutional Network (CNN). Penulis menggunakan 9 jenis ras kucing yang berbeda berisi 2700 gambar. Dalam pengujiannya, penulis menggunakan arsitektur EfficientNet-B0. Model paling optimal dari pengujian yang dilakukan terhadap 180 gambar kucing memperoleh tingkat akurasi sebesar 95%.

Kata Kunci : Deep Learning, Convolutional Neural Network (CNN) , Ras kucing, EfficientNetB0.

ABSTRACT

Cat is one of a popular animals in the world. Number of cat breeds in the world only about 1%, so most are dominated by cats mixed or domestic cat. Nevertheless, there are so many different types of cat breeds in the world, that it is sometimes difficult to identify them. Therefore, we need a system that can recognize the types of cat breeds. One technique of deep learning that may apprehend and hit upon gadgets in a photograph is Convolutional Neural Network (CNN). CNN functionality is alleged because the nice technique in phrases of

item detection and item recognition. The author used 9 different types of cat breeds containing 2700 images. The EfficientNet-B0 architecture is used on the system. The most optimal model has earned the accuracy of 95%.

Keywords : Deep Learning, Convolutional Neural Network (CNN), Cat breeds, EfficientNetB0.

PENDAHULUAN

Kucing (*Felis catus*) adalah mamalia karnivora dari keluarga *Felidae*. Sejak 6000 tahun SM, kucing diketahui telah berbaur dengan manusia dan menyebar di berbagai penjuru dunia. Kucing yang dimaksud biasanya adalah kucing yang sudah dijinakkan. Jumlah kucing ras di dunia hanya sekitar 1% sehingga kebanyakan didominasi oleh kucing campuran atau kucing kampung. Jumlah kucing ras yang sangat kecil ini membuat harga kucing ras jauh lebih mahal. Setiap ras kucing memiliki ciri khusus, namun karena banyaknya terjadi perkawinan silang, penentuan ras kucing menjadi lebih sulit [1].

Penampilannya yang menarik dengan berbagai jenis yang berbeda membuat kucing menjadi salah satu hewan peliharaan yang popular di dunia. Banyak orang memelihara kucing karena berbagai kelebihannya seperti bisa mengusir tikus, praktis karena tidak berisik, pakan sedikit, dan tidak membutuhkan ruangan yang begitu luas untuk pemeliharaannya. Selain itu memelihara kucing juga dapat memberikan dampak positif bagi pemiliknya, salah satunya yaitu bisa mengurangi rasa stres.

Perkembangan era kecerdasan buatan untuk mengenali citra berkembang sangat pesat. Seringkali gambar yang digunakan pada beberapa tahap dalam sistem kategori berada di kelas yang tidak sempurna. Seperti banyaknya gangguan dalam bentuk bayangan, gambar buram, dan perangkat mencurigakan yang digunakan. Jadi, kami menginginkan metode yang dapat mengatur dan menjalankan sistem kategori dalam kondisi gambar yang kurang sempurna. Salah satu pendekatan yang dapat digunakan adalah teknik *deep mastering* yang dapat menangkap dan mengenai suatu objek dalam sebuah foto virtual [2]. Salah satu teknik *deep learning* yang dapat menangkap dan menangkap gadget dalam sebuah foto adalah *Convolutional Neural Networ*k (CNN). Fungsionalitas CNN diduga karena teknik yang bagus dalam hal deteksi item dan pengenalan item . Teknik yang dimiliki CNN mirip dengan *neuron*, biasanya fungsi bobot, bias, dan aktivasi. Sederhananya, CNN dicapai dengan *filter* konvolusi. Teknik CNN juga memiliki kelemahan yang dimiliki berbagai teknik *deep mastering*, yaitu sistem pembelajaran membutuhkan waktu yang cukup lama.

Berdasarkan latar belakang masalah tersebut, penulis tertarik untuk mengangkat isu jual beli kucing sebagai tugas akhir dengan membuat sebuah aplikasi untuk mendeteksi jenis ras kucing. Perlu

p-ISSN: 2620-3383
e-ISSN: 2528-6544

pengetahuan bagi pemilik kucing untuk lebih memperhatikan cara merawat kucing sesuai dengan jenis rasnya sebelum memutuskan untuk memelihara kucing .

PERMASALAHAN

Berikut dipaparkan beberapa masalah yang mendasari penelitian tersebut:

1. Kebanyakan orang membeli atau memelihara kucing yang menurut mereka menarik, tanpa mencari tahu terlebih dahulu jenis ras kucing dan cara perawatan jenis ras kucing yang akan dibeli/dipelihara.

2. Kesalahan perawatan pada kucing akan menyebabkan kucing mudah terkena penyakit dan dapat mengubah penampilannya yang cantik menjadi tidak menarik lagi.

METODOLOGI PENELITIAN

Dalam penelitian ini dilakukan metode penelitian eksperimen dengan berbagai tahapan:

A. Dataset

Pengkondisian data dilakukan untuk mempersiapkan data untuk diproses ke tahap selanjutnya. Penulis menggunakan dataset dari *Kaggle* dan *Google Images*, dan *capture* manual dari kamera handphone (khusus jenis Kucing Lokal saja) yang berisi 2700 gambar kucing. Dataset ini berisi 9 jenis ras kucing diantaranya *Abyssinian*, *Bombay*, *British Shorthair*, Kucing Lokal, *Persian*, *Ragdoll*, *Savannah*, *Sphynx*, dan *Turkish Angora*. Berikut adalah contoh beberapa gambar dari dataset yang digunakan.

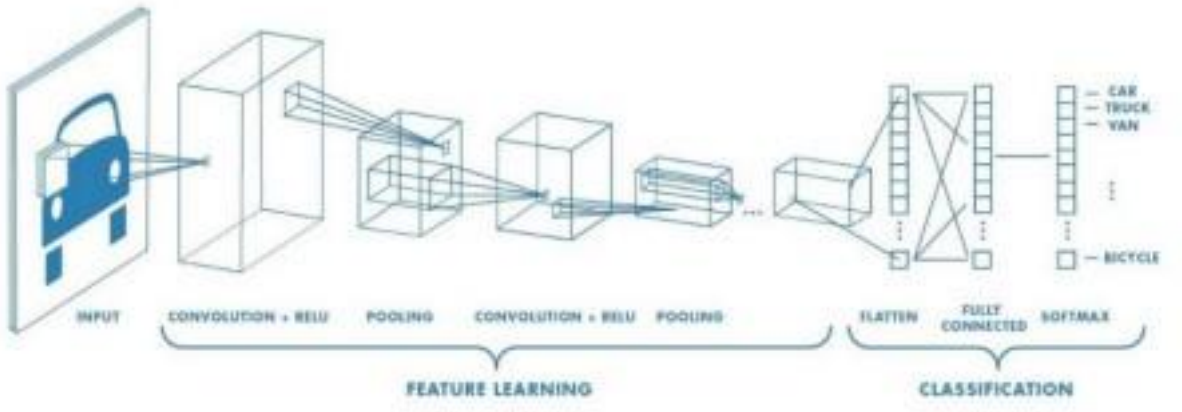
Gambar 1. Contoh gambar pada dataset

B. Metode

Metode *Convolutional Neural Network (CNN)* merupakan pengembangan lebih lanjut dari *Multilayer Perceptron* (MLP) yang dirancang untuk memproses data dua dimensi. CNN

p-ISSN: 2620-3383
e-ISSN: 2528-6544

diklasifikasikan sebagai jaringan saraf dalam karena kedalaman jaringannya yang besar dan banyak digunakan dalam data gambar. Dalam hal klasifikasi citra, MLP kurang cocok karena tidak menyimpan informasi spasial dari data citra dan mengasumsikan bahwa setiap piksel merupakan fitur independen, sehingga hasilnya buruk [3]. CNN memiliki dua metode. Metode pertama adalah penggunaan klasifikasi citra *feedforward* dan penggunaan metode *backpropagation* pada tahap pembelajaran. CNN terdiri dari beberapa *hidden layer*, yaitu *convolutional layer*, *activation function* (ReLU), *pooling layer* dan *fully connected layer*.



Gambar 2. Arsitektur *Convolutional Neural Network*

Berdasarkan gambar diatas, tahap pertama dari CNN adalah *Feature Learning* yang mengubah suatu *input* menjadi fitur berdasarkan ciri dari *input* tersebut (berbentuk angka dalam vektor). Lapisan ini terdiri dari *convolution layer*, fungsi aktivasi, dan *pooling layer*. Pada *convolution layer* digunakan *filter* berisi bobot yang digunakan untuk mendeteksi karakter dari objek yang dideteksi. Selanjutnya menuju fungsi aktivasi. Pada gambar diatas fungsi aktivasi yang digunakan adalah fungsi aktivasi ReLU (*Reactified Linear Unit*). Kemudian dilanjutkan menuju *pooling layer* [4]. Lapisan selanjutnya adalah klasifikasi yang berguna untuk mengklasifikasi tiap *neuron* yang telah diekstraksi pada lapisan sebelumnya. Lapisan ini terdiri dari *flatten* yang akan me-*reshape* fitur menjadi vektor, *fully connected* yang melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear, dan fungsi *Softmax*, digunakan untuk menghitung kemungkinan setiap kategori target di semua kategori target yang mungkin dan membantu menentukan kategori target untuk *input* yang diberikan.

- EfficientNet

EfficientNet merupakan salah satu dari arsitektur CNN yang dikembangkan oleh tim *Google Brain* bernama Mingxing Tan and Quoc V. Le. Mereka menggunakan *model scaling*, yaitu melakukan *scaling* pada model yang telah ada dalam hal *model depth*, *model width*, dan resolusi gambar untuk memperbaiki performa model. Melakukan *scaling* pada ketiganya secara bersamaan ini disebut dengan metode *compound scaling* [5]. *EfficientNet-B0* merupakan versi terkecil dari *EfficientNet*. Berikut merupakan gambar arsitektur B0:

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224 × 224	32	1
2	MBConv1, k3x3	112 × 112	16	1
3	MBConv6, k3x3	112 × 112	24	2
4	MBConv6, k5x5	56 × 56	40	2
5	MBConv6, k3x3	28 × 28	80	3
6	MBConv6, k5x5	28 × 28	112	3
7	MBConv6, k5x5	14 × 14	192	4
8	MBConv6, k3x3	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7	1280	1

Gambar 3. Baseline jaringan EfficientNet-B0

Blok MBConv merupakan blok *inverted residuals structure* yang terdapat pada arsitektur *MobileNetV2*, dengan pengaturan yang berbeda. Dari *baseline* ini, mereka mengaplikasikan metode *compound scaling* dan mendapatkan arsitektur *EfficientNet-B0* sampai B7.

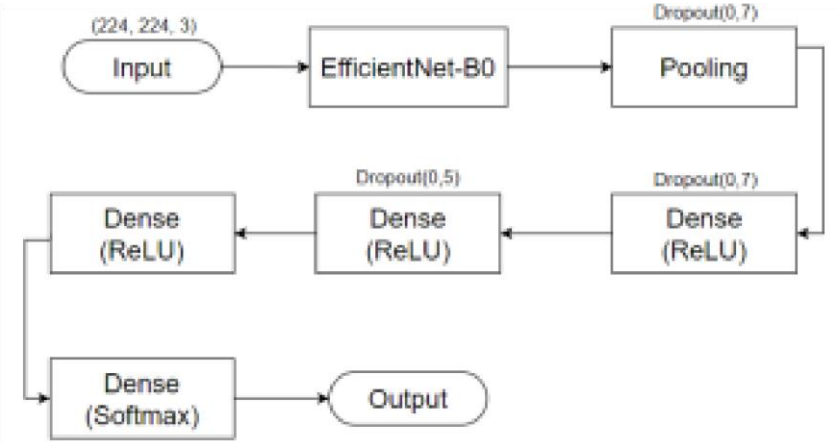
Penelitian Klasifikasi Penyakit Daun Pada Tumbuhan Menggunakan Model *Deep Learning EfficientNet* [6] menggunakan *Compound Model Scaling EfficientNet* dilakukan terhadap 38 kelas sebanyak 54305 gambar, dimana 12 kelas merupakan tanaman sehat, dan 26 kelas merupakan tanaman berpenyakit. Selain itu, didalam dataset terdapat satu kelas yang berisi 1143 gambar latar belakang yang berbeda. Jadi, jumlah seluruh dataset sebanyak 55448 gambar. Dataset yang digunakan pada penelitian ini dibagi secara acak menjadi 90% untuk *training*, 7% untuk *validation*, dan 3% untuk *testing* [7]. Penulis menggunakan metode *fine-tuning* untuk mempercepat proses pembelajaran dengan menggunakan *pre-trained* model berdasarkan *ImageNet*. *Layer fully-connected* terakhir dengan jumlah *output* sebanyak 1000 diubah menjadi 38, sesuai dengan permasalahan yang diteliti. Semua *layer* dari *pre-trained* model di-set sebagai *trainable*. Untuk fungsi aktivasi dan *loss function*, penulis menggunakan fungsi *Softmax* dan *categorical crossentropy* [8]. Proses *testing* dilakukan terhadap dataset original dan dataset yang telah di augmentasi. Nilai tertinggi didapatkan oleh model *EfficientNet-B4* dengan nilai akurasi sebanyak 99,97% terhadap

p-ISSN: 2620-3383
e-ISSN: 2528-6544

dataset augmentasi, dan model *EfficientNet-B5* dengan nilai akurasi sebanyak 99,91% terhadap dataset original.

● Arsitektur Usulan

Dalam pembuatan model, penulis menggunakan metode *transfer learning*. *Transfer learning* merupakan sebuah metode *training* dengan menggunakan metode yang telah di *training* dahulu, lalu menggunakan model tersebut untuk *dataset* yang baru. Untuk model CNN penulis menggunakan arsitektur *EfficientNet-B0*.



Gambar 4. Arsitektur usulan

Data yang masukannya berukuran 224x224x3, dimana angka 3 merupakan *channel* RGB (karena data masukannya adalah gambar berwarna) [9]. Selanjutnya digunakan arsitektur *EfficientNetB0* sebagai *base model* dari *layer* CNN yang akan digunakan [10]. Selanjutnya ditambahkan *layer Global Average Pooling* untuk mengurangi jumlah parameter sehingga proses komputasi dapat berjalan lebih cepat. Lalu ditambahkan dengan *dense* dan *dropout*, gunanya yaitu untuk mengurangi kemungkinan *overfitting. Overfitting* terjadi saat model terlalu menggeneralisasi data sehingga tidak dapat mengenali data yang baru dengan baik. Hal ini ditandai dengan nilai akurasi yang jauh lebih tinggi dibandingkan dengan nilai validasi. *Layer dense* yang terakhir menggunakan fungsi aktivasi *softmax* karena hasilnya akan berupa kategorik. Kemudian *output* didapatkan berupa hasil klasifikasi 9 jenis ras kucing.

HASIL DAN PEMBAHASAN

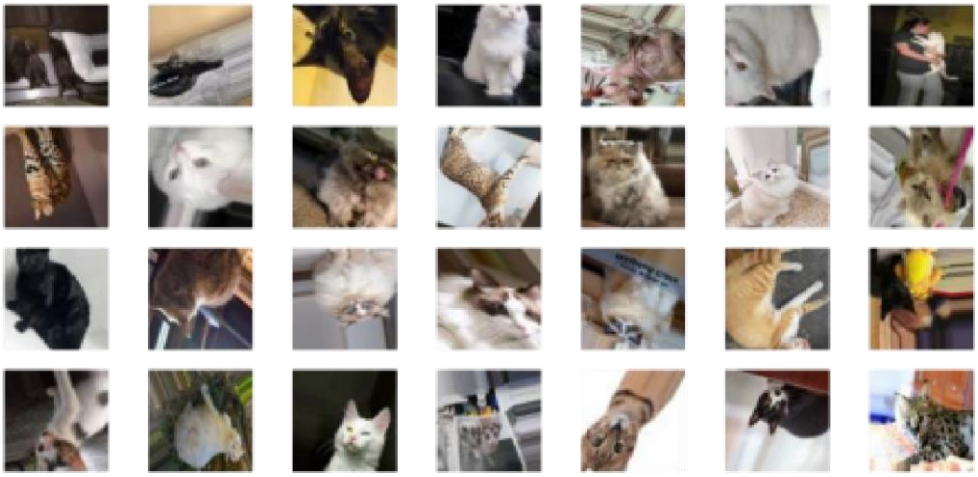
A. Data *pre-processing*

Dataset yang digunakan pada eksperimen dibagi menjadi data *train* dan data *validation*. Sedangkan untuk melakukan *testing*, ditambahkan data tambahan yaitu data *test* . Dataset ini berisi

p-ISSN: 2620-3383
e-ISSN: 2528-6544

9 jenis ras kucing yang berbeda dengan jumlah data *train* sebanyak 2160 gambar, data *validation* sebanyak 540 gambar, dan data *test* sebanyak 180 gambar. Jadi total keseluruhan dari dataset berjumlah 2700 gambar dan data *test* sebanyak 180 gambar. Dalam pembagian subset, data *training* harus memiliki nilai yang lebih besar dibandingkan dengan data *validation* dan data *testing.* Hal ini karena jika nilai data *training* lebih kecil, maka model akan mengalami kesulitan dalam pembelajaran, sehingga hasil yang didapatkan menjadi tidak optimal. Sebelum digunakan untuk proses *training*, data *train* akan melalui *pre-processing* menggunakan *filter gaussian blur*. Setelah melalui *pre-processing*, data akan melalui proses augmentasi [11]. Tujuannya untuk memperkaya variasi dari data *train* dan menambah jumlahnya *zoom_range*, *rotation_range*, *width_shift_range*, *height_shift_range*, *horizontal_flip*, dan *vertical_flip*.

Berikut beberapa contoh gambar dari data *train* yang telah melalui proses augmentasi.



Gambar 5. Contoh dataset yang telah melalui proses augmentasi

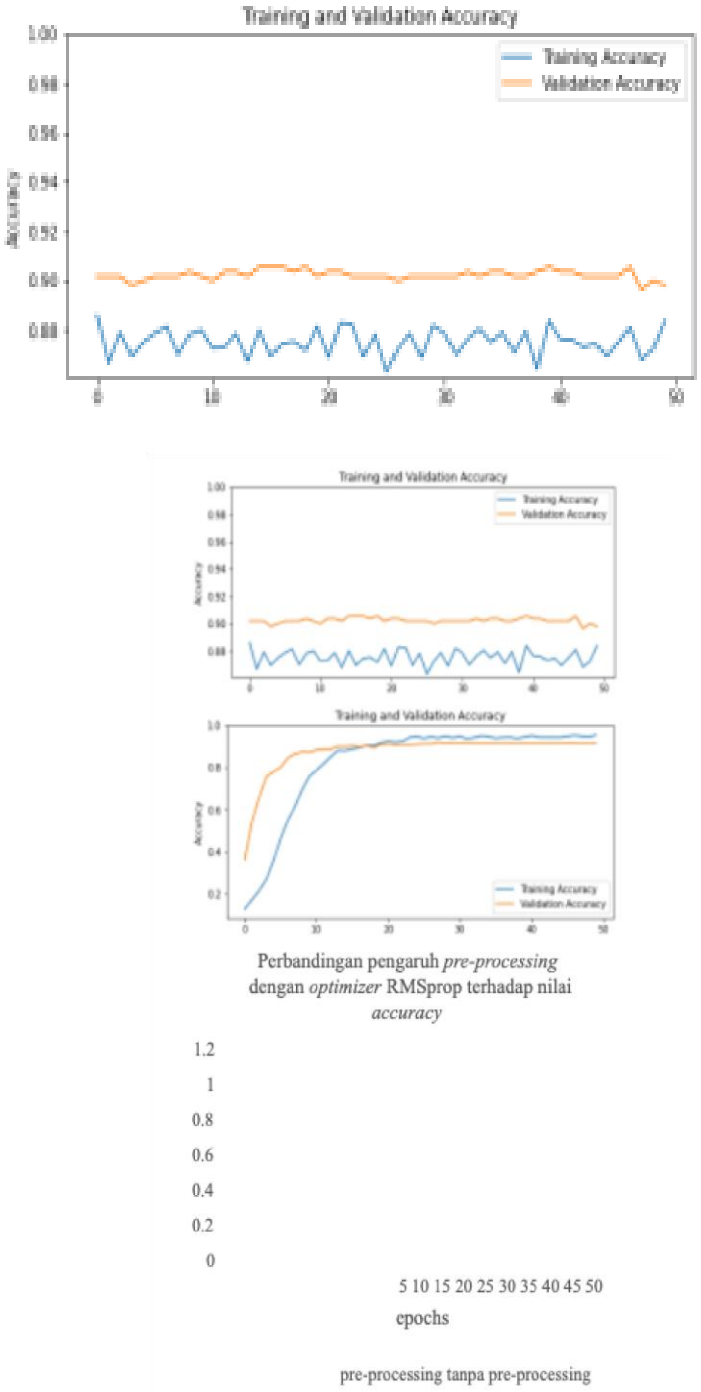
B. Hasil Ujicoba

Pada penelitian ini akan dilakukan beberapa skenario uji coba dengan menggunakan parameter yang berbeda-beda. Ujicoba yang pertama yaitu dengan membandingkan model dengan data *pre processing* dan tanpa data *pre-processing* [12]. Selanjutnya yaitu akan membandingkan model dengan *optimizer* Adam dan RMSprop dengan *learning rate* sebesar 0.001. Yang ketiga yaitu sama dengan sebelumnya, dengan menggunakan *optimizer* Adam dan RMSprop, namun dengan *learning rate* yang berbeda, yaitu sebesar 0.0001 [13].

p-ISSN: 2620-3383
e-ISSN: 2528-6544

1. Skenario 1

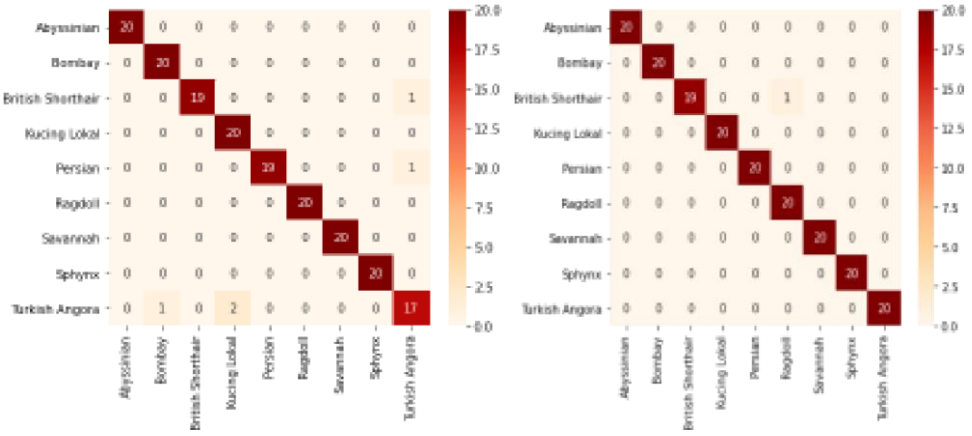
Ujicoba ini dilakukan dengan membandingkan model dengan data pre-processing dengan model tanpa data pre-processing [14]. *Pre-processing* yang digunakan pada ujicoba ini yaitu *filter gaussian blur* dari modul *scikit-image*. Hasil dari ujicoba dapat dilihat pada grafik *training* dibawah.



Gambar 6. Grafik perbandingan menggunakan *pre-processing* dan tanpa *pre-processing*

p-ISSN: 2620-3383
e-ISSN: 2528-6544

Dari gambar 6 dapat dilihat bahwa model yang menggunakan data *pre-processing* memiliki nilai akurasi yang lebih baik daripada model tanpa data *pre-processing* [15]. Nilai akurasi dari model tanpa data *pre-processing* mengalami naik turun di sekitar angka 88% saja. Sedangkan model dengan data *pre processing* menunjukkan nilai yang lebih tinggi yaitu 95%. Hal ini menunjukkan bahwa performa model yang menggunakan data *pre-processing* lebih baik dibandingkan dengan model tanpa data *pre processing* .



(a) (b)

Gambar 7. Confusion matrix skenario 1, (a) tanpa *pre-processing*, (b) dengan *pre-processing*

Pada percobaan tanpa menggunakan data *pre-processing*, model berhasil mengklasifikan dengan benar sebanyak 175 gambar. Sedangkan pada percobaan menggunakan data *preprocessing*, model berhasil mengklasifikan dengan benar sebanyak 179 gambar. Pada percobaan tanpa data *pre processing*, kesalahan terbanyak terdapat pada kelas *Turkish Angora*.

Tabel 1. Tabel *classification metrics* skenario 1

Skenario	*precision*	*recall*
Tanpa *pre-processing*	0.98	0.97
Dengan *pre-processing*	0.99	0.99

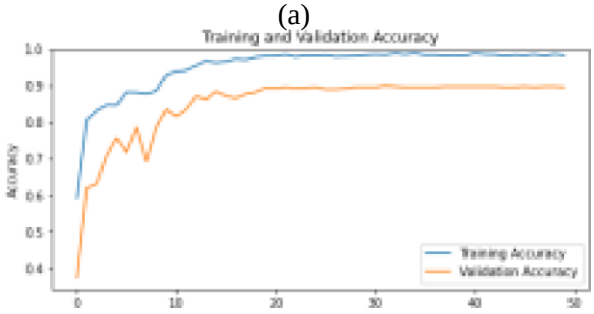
Tabel 1 menunjukkan bahwa kedua percobaan memiliki nilai *precision* dan *recall* diatas 0.9, dimana ini adalah hasil yang baik. Namun kedua nilai dari percobaan menggunakan *pre-processing* tetap lebih tinggi dibandingkan dengan percobaan tanpa menggunakan *pre-processing* [16].

p-ISSN: 2620-3383
e-ISSN: 2528-6544

2. Skenario 2

Ujicoba ini dilakukan dengan membandingkan *optimizer* Adam dan RMSprop dengan *learning rate* sebesar 0.001, dan *epoch* sebanyak 50 *epoch*. Hasil dari ujicoba dapat dilihat pada grafik *training* dibawah.

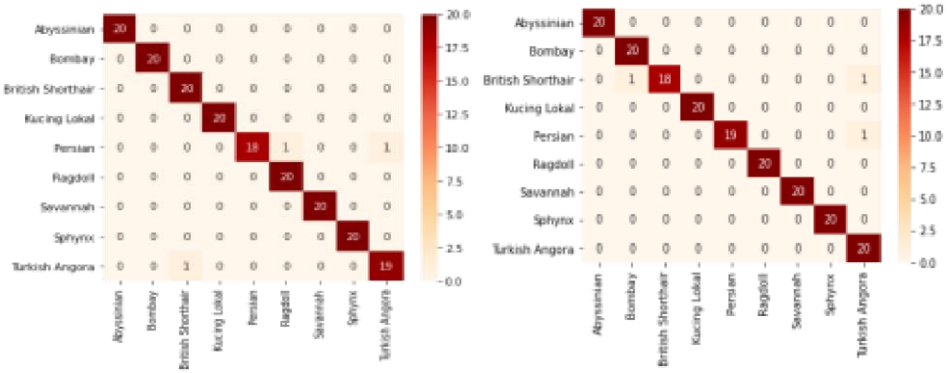
(a)



(b)

Gambar 8. Grafik training skenario 2, (a) Adam, (b) RMSprop

Pada grafik diatas dapat dilihat bahwa nilai akurasi pada kedua percobaan naik sampai mendekati angka 1.0 pada sekitar *epoch* ke-15. Sedangkan nilai validasi mengalami naik turun hingga sekitar *epoch* ke-15. Pada *epoch* selanjutnya [17], kedua nilai baik akurasi maupun validasi mengalami nilai konstan. Pada percobaan menggunakan *optimizer* Adam, nilai akurasi yang didapat sebesar 98% dan nilai validasi sebesar 88%. Sedangkan pada percobaan menggunakan *optimizer* RMSprop, nilai akurasi yang didapatkan sebesar 98% dan nilai validasi sebesar 89%.



p-ISSN: 2620-3383
e-ISSN: 2528-6544

(a) (b)

Gambar 9. *Confusion matrix* skenario 2, (a) Adam, (b) RMSprop

Pada percobaan 2(a), yaitu menggunakan *optimizer* Adam, jumlah gambar yang berhasil diprediksi sebanyak 177 dan jumlah gambar yang salah prediksi sebanyak 3 yang terletak pada kelas *British Shorthair* dan *Persian*. Sedangkan pada percobaan 2(b) jumlah gambar yang berhasil diprediksi sebanyak 177, hasil ini sama dengan percobaan 2(a), namun terdapat perbedaan pada kelas yang salah prediksi, yaitu *Persian* dan *Turkish Angora*.

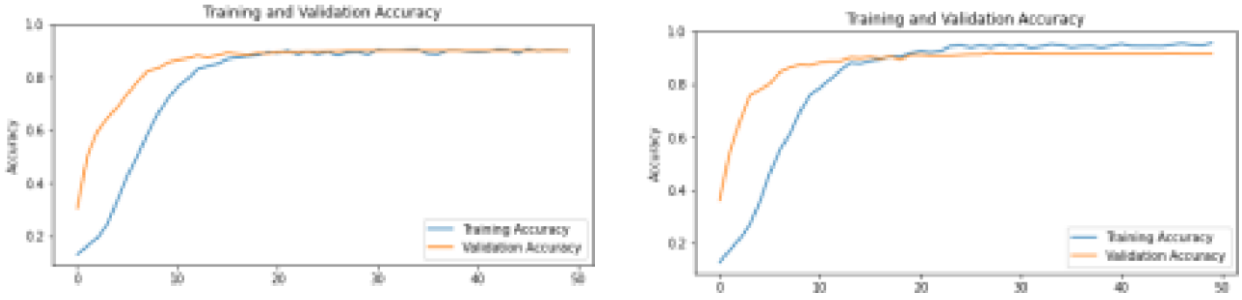
Tabel 2. Tabel *classification metrics* skenario 2

Skenario	*precision*	*recall*
Adam (0.001)	0.99	0.98
RMSprop (0.001)	0.98	0.98

Nilai *precision* dari kedua ujicoba menunjukkan angka diatas 0.9, dimana hal ini berarti model sudah sangat baik dalam mengenali setiap kelasnya [18]. Sedangkan nilai dari *recall* kedua ujicoba sebesar 0.98. Nilai *recall* yang tinggi menunjukkan bahwa model tersebut efektif dalam mengidentifikasi hasil *true positive* dan mengurangi jumlah hasil *false negative* [19].

3. Skenario 3

Ujicoba ini dilakukan dengan membandingkan *optimizer* Adam dan RMSprop, sama dengan 47cenario sebelumnya tetapi dengan *learning rate* sebesar 0.0001, dan *epoch* sebanyak 50 *epoch*. Hasil dari ujicoba dapat dilihat pada grafik *training* dibawah.

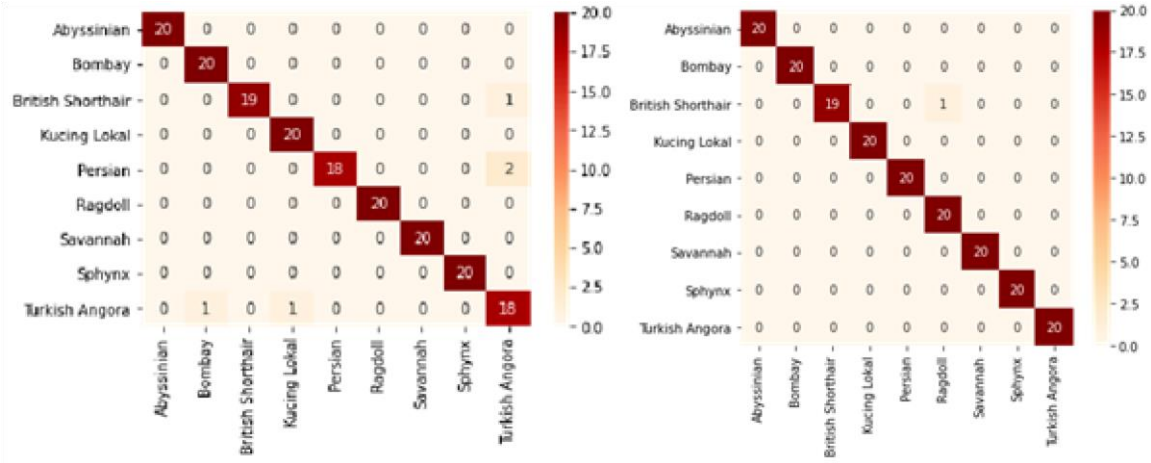


(a) (b)

Gambar 10. Grafik *training* 47cenario 3, (a) Adam, (b) RMSprop

p-ISSN: 2620-3383
e-ISSN: 2528-6544

Pada grafik diatas dapat dilihat bahwa nilai akurasi pada kedua percobaan naik secara signifikan sampai sekitar angka 0.9 pada sekitar epoch ke-15 . Sedangkan nilai validasi juga mengalami kenaikan pada sekitar *epoch* ke-15. Pada *epoch* selanjutnya, kedua nilai baik akurasi maupun validasi mengalami nilai konstan. Pada percobaan menggunakan *optimizer* Adam, nilai akurasi yang didapat sebesar 90% dan nilai validasi sebesar 90%. Sedangkan pada percobaan menggunakan *optimizer* RMSprop, nilai akurasi yang didapatkan sebesar 95% dan nilai validasi sebesar 91%. Jika dibandingkan dengan 48cenario 1, nilai akurasi pada 48cenario ini sedikit lebih rendah, namun tidak terjadi *overfitting* [20].



(a) (b)

Gambar 11. *Confusion matrix* 48cenario 3, (a) Adam, (b) RMSprop

Pada percobaan 3(a), yaitu menggunakan *optimizer* Adam, jumlah gambar yang berhasil diprediksi sebanyak 175 dan jumlah gambar yang salah prediksi sebanyak 1 yang terletak pada kelas *British Shorthair*, 2 pada *Persian*, dan 2 pada *Turkish Angora*. Sedangkan pada percobaan 3(b) yaitu menggunakan *optimizer* RMSprop, jumlah gambar yang berhasil diprediksi sebanyak 179, dengan kesalahan hanya terletak pada kelas *British Shorthair*.

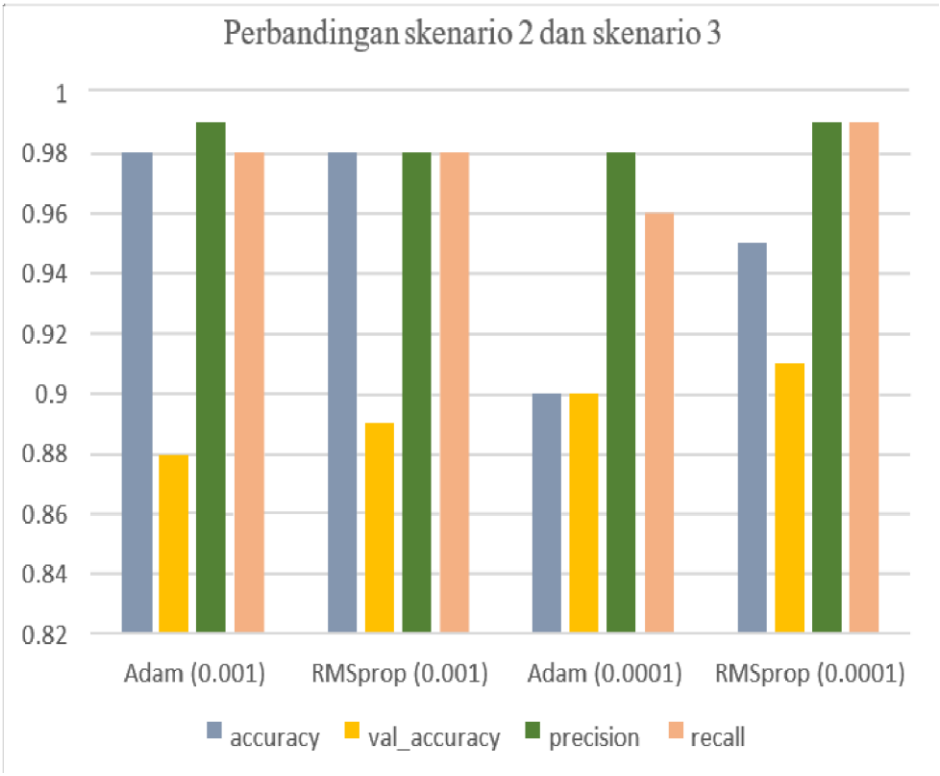
Tabel 3. Tabel *classification metrics* skenario 3

Skenario	*precision*	*recall*
Adam (0.0001)	0.98	0.96

p-ISSN: 2620-3383
e-ISSN: 2528-6544

RMSprop (0.0001)	0.99	0.99

Nilai *precision* dari kedua ujicoba menunjukkan angka diatas 0.9, dimana hal ini berarti model sudah sangat baik dalam mengenali setiap kelasnya. Sedangkan nilai dari *recall* dari ujicoba menggunakan Adam dengan *learning rate* 0.0001 menunjukkan nilai yang lebih rendah daripada ujicoba menggunakan Adam dengan *learning rate* 0.001. sedangkan nilai *recall* dari ujicoba menggunakan RMSprop dengan *learning rate* 0.0001 menunjukkan nilai yang lebih tinggi dibandingkan dengan ujicoba menggunakan RMSprop dengan *learning rate* 0.001.



Gambar 12. Perbandingan skenario 2 dan 3

Dari gambar 3.6 disajikan perbandingan dari skenario uji coba kedua dan ketiga. Dari gambar dapat dilihat bahwa nilai *accuracy* tertinggi terdapat pada ujicoba menggunakan *optimizer* Adam dengan *learning rate* 0.001. Namun jaraknya dengan nilai *val_accuracy* terlalu banyak jadi dapat disimpulkan bahwa model masih kurang bagus dalam mengklasifikasikan. Sedangkan jarak antara *accuracy* dan *val_accuracy* yang paling sedikit terdapat pada ujicoba menggunakan *optimizer* RMSprop dengan *learning rate* 0.0001. Nilai *precision* dan *recall* dari uji coba ini juga sangat tinggi

p-ISSN: 2620-3383
e-ISSN: 2528-6544

sehingga dapat disimpulkan bahwa ujicoba ini adalah ujicoba dengan hasil terbaik dibandingkan dengan uji coba yang lainnya.

Tabel 4. Tabel hasil percobaan 1, 2, dan 3

Optimizer	Learning rate	Pre-processing (Gaussian blur)	Accuracy	Validation accuracy
RMSprop	0.0001	Tidak	88%	89%
Adam	0.001	Ya	98%	88%
RMSprop	0.001	Ya	98%	89%
Adam	0.0001	Ya	90%	90%
RMSprop	0.0001	Ya	95%	91%

Berdasarkan tabel 4, percobaan dengan nilai akurasi paling tinggi yaitu percobaan menggunakan *optimizer* Adam dan RMSprop dengan *learning rate* 0.001. Namun, kedua percobaan ini mengalami *overfitting* karena memiliki nilai akurasi yang jauh lebih tinggi dibandingkan dengan nilai validasi, sehingga masih kurang dapat mengklasifikasikan gambar dengan baik. Percobaan menggunakn *optimizer* RMSprop dengan *learning rate* 0.0001 mendapatkan akurasi sebesar 95%, lebih rendah dibandingkan dengan percobaan sebelumnya. Namun hasil percobaan ini tidak mengalami *overfitting* sehingga model dapat mengklasifikasikan gambar lebih baik. Sedangkan percobaan dengan nilai akurasi paling rendah terdapat pada kedua percobaan yang tanpa menggunakan data *pre-processing*. Sehingga dapat disimpulkan bahwa penggunaan data *pre-processing* sangat penting untuk meningkatkan performa dari model.

KESIMPULAN

Setelah melalui berbagai tahapan, maka dapat diambil kesimpulan yaitu Penelitian menggunakan arsitektur *EfficientNet-B0* sebagai *base* model. Model mendapatkan akurasi yang paling tinggi yaitu sebesar 98% menggunakan *optimizer* Adam dan RMSprop dengan *learning rate*

0.001. Namun masih terjadi *overfitting* pada model ini. Model paling optimal mendapatkan akurasi sebesar 95% dan akurasi validasi sebesar 91%. Model ini berhasil mengklasifikan 179 dari 180 gambar kucing dengan kelas yang benar. Model ini yaitu menggunakan *optimizer* RMSprop dengan *learning rate* 0.0001 dan menggunakan data *pre-processing*. Semakin kecil learning rate semakin kecil kemungkinan terjadinya *overfitting*, namun akurasi yang didapatkan juga lebih rendah.

SARAN

Untuk penelitian lebih lanjut, diberikan beberapa saran yang dapat membantu dalam pengembangan, diantaranya:

1. Menambahkan kelas pada dataset untuk jenis ras kucing yang ciri-cirinya lumayan mirip.
2. Menambahkan variasi pada dataset seperti warna bulu, corak, atau ras campuran. Menggunakan metode *pre-processing* lainnya agar model lebih optimal.

DAFTAR PUSTAKA

[1] L. Mariani, "Aplikasi Pendeteksian RAS Kucing Dengan Mendeteksi Wajah Kucing Dengan Metode Viola Jones Berbasis Android," 2016.

[2] I. N. Purnama, "HERBAL PLANT DETECTION BASED ON LEAVES IMAGE USING CONVOLUTIONAL NEURAL NETWORK WITH MOBILE NET ARCHITECTURE," *JITK (Jurnal Ilmu Pengetah. dan Teknol. Komputer)*, vol. 6, no. 1, pp. 27–32, 2020.

[3] W. Anggraini, "Deep Learning Untuk Deteksi Wajah Yang Berhijab Menggunakan Algoritma Convolutional Neural Network (CNN) Dengan Tensorflow." UIN Ar-Raniry Banda Aceh, 2020.

[4] I. Putra, "Klasifikasi citra menggunakan convolutional neural network (CNN) pada caltech 101." Institut Teknologi Sepuluh Nopember, 2016.

[5] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, 2019, pp. 6105–6114.

[6] Ü. Atila, M. Uçar, K. Akyol, and E. Uçar, "Plant leaf disease classification using EfficientNet deep learning model," *Ecol. Inform.*, vol. 61, p. 101182, 2021.

[7] T. Nurhikmat, "Implementasi deep learning untuk image classification menggunakan algoritma Convolutional Neural Network (CNN) pada citra wayang golek," 2018.

[8] K. O. Lauw, L. W. Santoso, and R. Intan, "Identifikasi Jenis Anjing Berdasarkan Gambar Menggunakan Convolutional Neural Network Berbasis Android," *J. Infra*, vol. 8, no. 2, pp. 37–43, 2020.

[9] A. Salsabila, R. Yunita, and C. Rozikin, "Identifikasi Citra Jenis Bunga menggunakan Algoritma KNN dengan Ekstrasi Warna HSV dan Tekstur GLCM," *Technomedia J.*, vol.

6, no. 1, pp. 124–137, 2021.

[10] N. P. L. Santoso, Y. Durachman, S. Watini, and S. Millah, "Manajemen Kontrol Akses Berbasis Blockchain untuk Pendidikan Online Terdesentralisasi," *Technomedia J.*, vol. 6, no. 1, pp. 111–123, 2021.

[11] D. H. Fudholi *et al.*, "Image Captioning with Attention for Smart Local Tourism using EfficientNet," in *IOP Conference Series: Materials Science and Engineering*, 2021, vol. 1077, no. 1, p. 12038.

[12] S. Indriyani, F. Sthevanie, and K. N. Ramadhani, "Pengenalan Ras Kucing Scottish Fold Menggunakan Metode Histogram Of Oriented Gradients Dan Jaringan Saraf Tiruan," *eProceedings Eng.*, vol. 6, no. 2, 2019.

[13] S. Khatri, A. Rajput, S. Alhat, V. Gursal, and J. Deshmukh, "Image-Based Animal Detection and Breed Identification Using Neural Networks."

[14] H. S. Hopipah and R. Mayasari, "Optimasi Backward Elimination untuk Klasifikasi Kepuasan Pelanggan Menggunakan Algoritme k-nearest neighbor (k-NN) and Naive Bayes," *Technomedia J.*, vol. 6, no. 1, pp. 99–110, 2021.

[15] P. Pangestu and R. Yusuf, "Implementasi Metode QINQ Pada Jaringan Metro Ethernet Untuk Memaksimalkan Penggunaan VLAN Menggunakan Teknologi GPON Studi Kasus: PT. Telkom Indonesia," *Technomedia J.*, vol. 6, no. 1, pp. 82–98, 2021.

[16] R. A. A. Rahman and A. Adhitya, "Perancangan Sistem Informasi Pengusulan Kenaikan Pangkat Berbasis Web Pada Korps Marinir TNI AL," *Technomedia J.*, vol. 6, no. 1, pp. 30–42, 2021.

[17] B. Basri and A. Qashlim, "Relay Kontrol Menggunakan Google Firebase dan Node MCU pada Sistem Smart Home," *Technomedia J.*, vol. 6, no. 1, pp. 15–29, 2021.

[18] R. Rosyid and M. A. W. Prasetyo, "Robot Peraga 12 Gerakan Pengaturan Lalu Lintas Berbasis Arduino Mega 2560," *Technomedia J.*, vol. 5, no. 2 Februari, pp. 193–205, 2021.

[19] A. Roihan, N. Rahayu, and D. S. Aji, "Perancangan Sistem Kehadiran Face Recognition Menggunakan Mikrokomputer Berbasis Internet of Things," *Technomedia J.*, vol. 5, no. 2 Februari, pp. 155–166, 2021.

[20] A. Setiadi, I. Handayani, and F. Fadilah, "Perancangan Aplikasi Fit Your Weight Untuk Menghitung Berat Badan Ideal Berbasis Android," *Technomedia J.*, vol. 5, no. 2 Februari, pp. 144–154, 2021.