Klasifikasi Ras pada Kucing menggunakan Algoritma Convolutional Neural Network(CNN)

Muhammad Afif Amanullah Fawwaz[1], Kurniawan Nur Ramadhani, S.T., M.T.[2], Febryanti Sthevanie, S.T., M.T.[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]muhammadfawwaz@students.telkomuniversity.ac.id, [2]kurniawannr@telkomuniversity.ac.id, [3]sthevanie@telkomuniversity.ac.id

Abstrak

Kucing adalah hewan peliharaan yang bisa dimiliki oleh semua orang. Namun terkadang tidak semua pemilik kucing mengetahui ras kucingnya. Perawatan kucing seharusnya dilakukan dengan perawatan khusus untuk menjamin kesehatan kucing, disesuaikan dengan ras yang dimiliki kucing tersebut karna setiap ras kucing mempunyai karateristik yang berbeda sehingga diperlukan perawatan yang berbeda pula. Maka dari itu diperlukan sebuah program yang dapat mengenali ras dari seekor kucing. Pada penelitian tugas akhir ini yang berjudul "Klasifikasi Ras pada Kucing menggunakan Algoritma Convolutional Neural Network (CNN)" telah dilakukan training dan testing dengan menggunakan dataset yang berasal dari Oxford-IIIT yang berjumlah 2393 citra dengan jumlah kelas sebanyak 12 kelas. Model yang digunakan yaitu VGG16, InceptionV3, ResNet50 dan Xception. Hasil testing didapatkan berupa akurasi untuk tiap modelnya yaitu 60.85%, 84.94%, 71.39%, dan 93.75%.

Kata Kunci: CNN, klasifikasi, visi komputer

Abstract

Cats are pets that everyone can have. But sometimes not all cat owners know their cat breeds. Caring for a cat can not be arbitrary, but it requires special care. The treatment of cat must be specific based on each breed in cats for ensure the cat's healt adjusted to the cat breeds has different characteristics so that different care is needed. So we need a program that can recognize the breed of a cat. In this research, entitled "Breed Classification in Cats using Convolutional Neural Network (CNN) Algorithm" training and testing has been carried out using a dataset originating from Oxford-IIIT totaling 2393 images with 12 classes. The models used are VGG16, InceptionV3, ResNet50 and Xception. The test accuracy results are 60.85%, 84.94%, 71.39%, dan 93.75%.

Keywords: CNN, classification, computer vision

1. Pendahuluan

Latar Belakang

Pada zaman sekarang sebagian manusia memiliki hewan peliharaan. Selain berfungsi untuk menghilangkan rasa bosan, hewan peliharaan juga dapat digunakan untuk menjadi teman manusia di berbagai aktivitas. Salah satu hewan peliharaan yang paling populer adalah kucing. Kucing dipilih menjadi hewan peliharaan manusia karena tingkahnya yang lucu. Kucing memiliki variasi ras yang cukup banyak. Setidaknya terdapat 315 ras kucing di seluruh dunia.

Kucing adalah hewan peliharaan yang bisa dimiliki semua orang. Namun terkadang tidak semua pemilik kucing mengetahui ras kucingnya. Perawatan kucing seharusnya dilakukan dengan perawatan khusus untuk menjamin kesehatan kucing, disesuaikan dengan ras yang dimiliki kucing tersebut karna setiap ras kucing mempunyai karateristik yang berbeda sehingga diperlukan perawatan yang berbeda pula. Maka dari itu diperlukan sebuah program yang dapat mengenali ras dari seekor kucing. Contohnya pada gambar 1 produsen makanan kucing "Royal Canin", mereka menjual makanan kucing yang berbeda untuk tiap rasnya. Seperti pada ras bengal dan british shorthair, "Royal Canin" menjual makanan yang berbeda untuk dua ras tersebut.

Gambar 1. Produk Makanan "Royal Canin"

Maka diperlukan sebuah program yang dapat mengenali ras dari sebuah kucing. Pada penelitian tugas akhir ini yang berjudul "Klasifikasi Ras pada Kucing menggunakan Algoritma Convolutional Neural Network(CNN)". Algoritma CNN dipilih karena akurasinya yang cukup baik, seperti pada model Xception dan InceptionV3.

Topik dan Batasannya

Metode klasifikasi ras yang kami terapkan menggunakan algoritma Convolutional Neural Network(CNN). Model yang dipakai adalah VGG16, InceptionV3, ResNet50, dan Xception. Penelitian dilakukan dengan membandingkan ketiga parameter yang telah ditentukan yaitu tanpa menggunakan transfer learning, menggunakan transfer learning tanpa fine tuning. dan menggunakan transfer learning dengan fine tuning.

Tujuan

Penelitian kali ini bertujuan untuk mencari model yang terbaik antara VGG16, InceptionV3, ResNet50, dan Xception. Evaluasi dilakukan dengan menggunakan metrik akurasi.

Organisasi Tulisan

Studi yang menjadi acuan dari penelitian ini dipaparkan pada bagian studi terkait. Lalu cara kerja dan metode evaluasi sistem diterangkan pada bagian sistem yang dibangun. Bagian evaluasi melaporkan hasil pengujian yang dilakukan. Dan yang terakhir, insight yang didapat dari pengujian disampaikan pada bagian kesimpulan.

2. Studi Terkait

Pada karya ini[1] dilakukan penelitian mengenai klasifikasi ras pada anjing. Dataset yang digunakan sebanyak 8351 citra dengan 133 ras. Metode yang digunakan yaitu Local Binary Patterns(LBP), Histograms of Oriented Gradients(HOG), dan CNN dengan model InceptionV3 menggunakan transfer learning. Pada eksperimen 20 ras, hasilnya menunjukkan bahwa untuk tiap-tiap metode akurasinya 62.25%, 79.25%, dan 96.75%. Pada eksperimen 133 ras yang menggunakan CNN transfer learning dengan model InceptionV3, MobileNetV2, dan NASNet didapatkan hasil akurasinya adalah 89.50%, 89,60%, dan 91.00%.

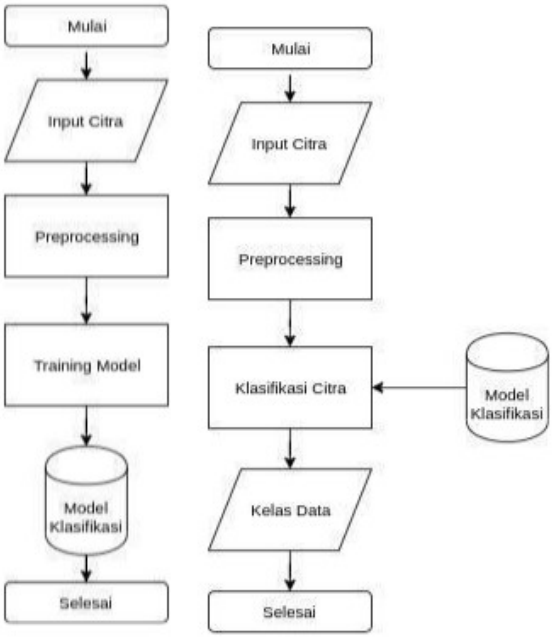
Berbeda dengan karya ini[2] yang menggunakan dataset dari stanford dengan jumlah 20.580 citra dengan 120 ras. Lalu dipilih 2.247 citra dengan 20 ras. 80% dataset digunakan untuk pelatihan dan 20% digunakan untuk pengujian. Metode yang digunakan yaitu CNN dengan model VGG16, InceptionV3, dan Xception. Hasil yang didapat yaitu akurasi sebesar 84.7%, 93.98%, dan 93.89% untuk tiap modelnya.

Ada juga karya [3] yang menggunakan dataset yang berasal dari stanford. Jumlah dataset tersebut dibagi menjadi 70% untuk pelatihan dan 30% untuk pengujian. Berbeda dengan dua karya sebelumnya, karya ini menggunakan teknik fine tuning dan data augmentasi. Model yang dipakai yaitu ResNet-50, DenseNet-121, DenseNet-129, dan GoogleNet yang menghasilkan akurasi sebesar 89.66%, 85.37%, 84.01%, dan 82.08%.

Berbeda juga dengan karya [4] yang melakukan pengenalan ras pada kuda. Dataset berjumlah 1693 citra dengan 6 ras kuda. Untuk mencegah overfitting digunakan teknik data augmentasi dan dropout. Model yang digunakan yaitu VGG16, VGG19, InceptionV3, ResNet50, dan Xception yang menghasilkan akurasi sebesar 90.69%, 90.05%, 88.79%, 95.90%, dan 93.00%.

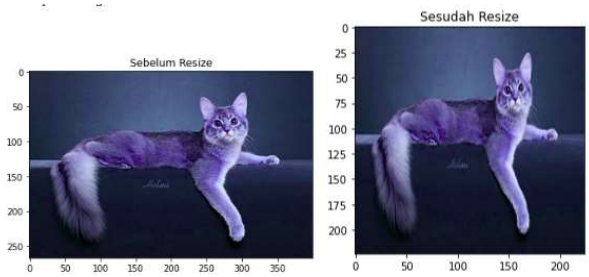
3. Sistem yang Dibangun

Sistem klasifikasi yang dibangun seperti pada gambar 2 terdiri dari dua tahap yaitu *training* dan *testing*. *Training* digunakan untuk melatih model agar dapat melakukan klasifikasi dengan baik, sedangkan *testing* digunakan untuk mendapatkan hasil prediksi klasifikasi dengan menggunakan model yang telah dilatih pada tahap *training*.



Gambar 2. Diagram Alir *Training*(Kiri) dan *Testing*(Kanan)

3.1 *Preprocessing*



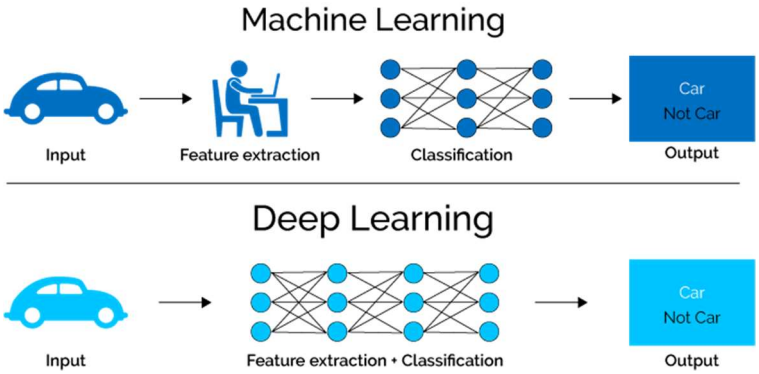
Gambar 5. Citra Sebelum(Kiri) dan Sesudah(Kanan) *Resize*

Preprocessing adalah tahap yang dilakukan sebelum *training* atau *testing* model. Pada tahap ini dilakukan proses *resize,* augmentasi, dan normalisasi. *Resize* adalah mengubah ukuran citra. Teknik ini digunakan untuk menyesuaikan citra supaya dapat dilakukan *training* atau *testing*. *Resize* dilakukan dengan membuat titik-titik baru yang disebut dengan *interpolation*. Teknik *Interpolation* umumnya dibagi menjadi dua yaitu *Non-adaptive* dan *Adaptive*. *Non-adaptive* yaitu manipulasi piksel secara langsung tanpa memperhatikan fitur. *Adaptive* yaitu mempertimbangkan fitur gambar seperti nilai intensitas, informasi tepi, tekstur, dll. Pada penelitian kali ini digunakan teknik *bilinear interpolation* yang termasuk dalam *Non-Adaptive*. *Bilinear Interpolation* adalah teknik *interpolation* yang mengambil rata-rata dari 4 piksel lingkungan untuk menghitung nilai interpolasi akhirnya[7]. Adapun kernel *interpolation*-nya adalah sebagai berikut[8]:

$$u(x) = 0x > 11 - x < 1$$

Normalisasi adalah mengubah nilai piksel citra menjadi lebih sederhana. Nilai piksel pada citra bernilai 0 hingga 255. Nilai piksel tersebut masih bisa disederhanakan untuk mempermudah proses klasifikasi. Nilai piksel akan dinormalisasi ke dalam nilai bilangan riil antara 0 hingga 1.

3.2 *Deep Learning*



Gambar 6. Perbedaan *Machine Learning* dan *Deep Learning*

Machine Learning[19] adalah sebuah sub bab pada *Artificial Intelligence* yang mencoba untuk meniru proses perilaku belajar pada manusia. Serupa dengan cara bayi yang belajar dari suatu hal yang ditemui, *Machine Learning* juga akan memberikan respon terhadap sebuah insiden di masa depan dengan mengobservasi data sebelumnya sebagai input. Metode pada *machine learning* ini yaitu *supervised learning*, *semi supervised learning*, *unsupervised learning* dan lain sebagainya. Dalam sebuah sistem *machine learning*, tahap *feature extraction* dan *classification* adalah dua tahap yang berbeda.

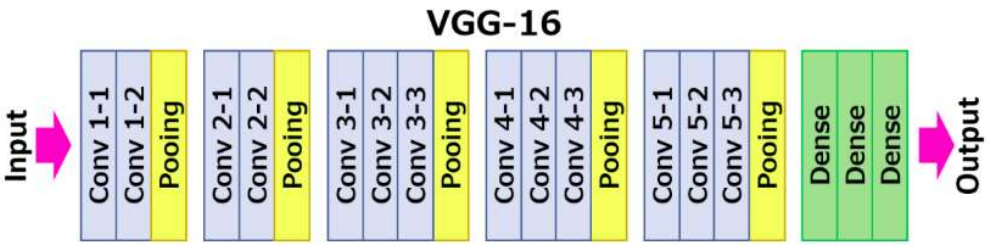
Seperti pada gambar 6, *Deep Learning*[19] sama saja dengan *machine learning*, namun pada *deep learning* tahap *feature extraction* dan *classification* berada pada satu tahap. Ada beberapa tipe *neural networks* yang ada pada *deep learning* yaitu *Artificial Neural Networks*(ANN), *Convolutional Neural Networks*(CNN), dan *Recurrent Neural Networks*(RNN).

3.3 *Convolutional Neural* Networks(CNN)

Convolutional Neural Networks(CNN) adalah salah satu jenis *neural networks* yang ada pada *deep learning*. Berbeda dengan ANN dan RNN, CNN adalah jenis *neural networks* yang biasanya digunakan untuk mengolah data dalam bentuk citra. CNN bekerja menggunakan *kernel*. *Kernel* tersebut akan mengekstrak fitur dari input menggunakan operasi konvolusi.

Kelebihan daripada CNN adalah *parameter sharing* yang dapat membantu mengurangi jumlah parameter pada keseluruhan sistem dan membuat beban komputasinya berkurang. Selain itu juga CNN memliki keunggulan *spatial features*. *Spatial features* mengacu pada susunan piksel dan hubungan antar piksel dalam sebuah citra. Hal ini memudahkan untuk melakukan identifikasi suatu objek, lokasi suatu objek, dan hubungannya dengan citra yang lain.

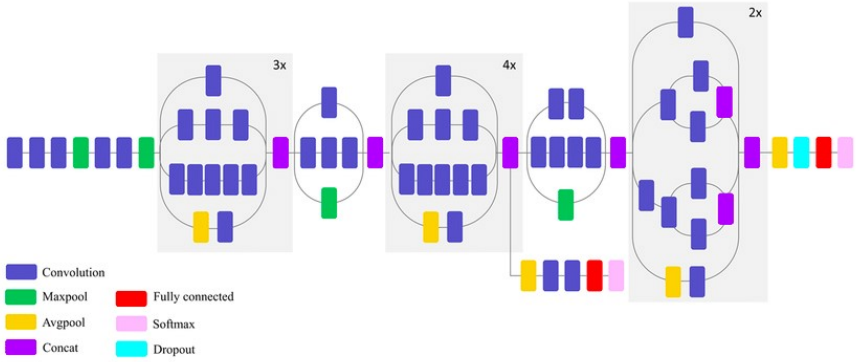
Terdapat banyak model CNN yang bisa digunakan untuk melakukan klasifikasi. Pada penelitian ini digunakan model *VGG16*[11], *InceptionV3*[12], *ResNet50*[13], dan *Xception*[14.]. Model-model ini dipilih karena pada beberapa kasus sebelumnya[2][4] terbukti mampu untuk melakukan klasifikasi.



Gambar 7. Arsitektur *VGG16*

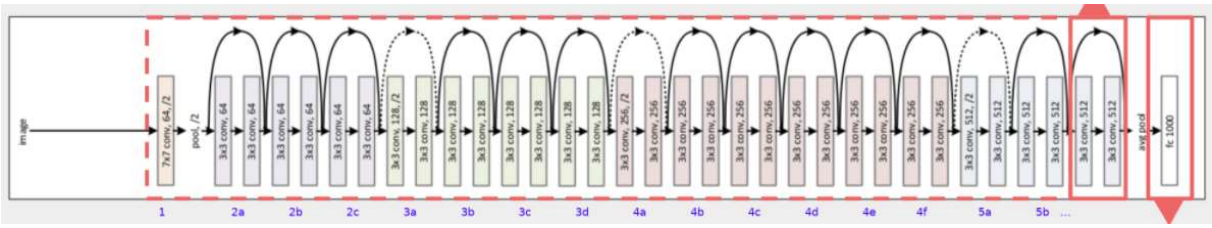
VGG16[11] merupakan model CNN yang memanfaatkan *convolutional layer* dengan spesifikasi *convolutional* filter yang kecil (3×3). Dengan ukuran *convolutional filter* tersebut, kedalaman *neural network* dapat ditambah dengan lebih banyak lagi *convolutional layer*. Hasilnya, model CNN menjadi lebih akurat

daripada model-model CNN sebelumnya. Model *VGG16*[11] mempunyai 16 layer yang terdiri dari 13 *convolutional layer* dan 3 *fully-connected layer*. Seperti pada gambar 7 terdapat 13 *convolutinal layer* dan 3 *fully connected layer*. Terdapat 5 *pooling layer* yang bertipe *max pooling*. Bentuk inputnya yaitu 224x224x3.



Gambar 8. Arsitektur *InceptionV3*

InceptionV3[12] adalah seri ketiga dari *Google Deep Learning Architecture*. Model ini sebelumnya telah dilakukan training menggunakan dataset imagenet dan mampu menjuarai ILSVRC2015. Seperti pada gambar 8*InceptionV3* memiliki arsitektur yang cukup dalam daripada VGG16. *InceptionV3* menggunakan 103 *convolutional layer*, 4 *max pooling layer*, 5 *average pooling layer*, dengan menggunakan *dropout* sebagai regularisasi.

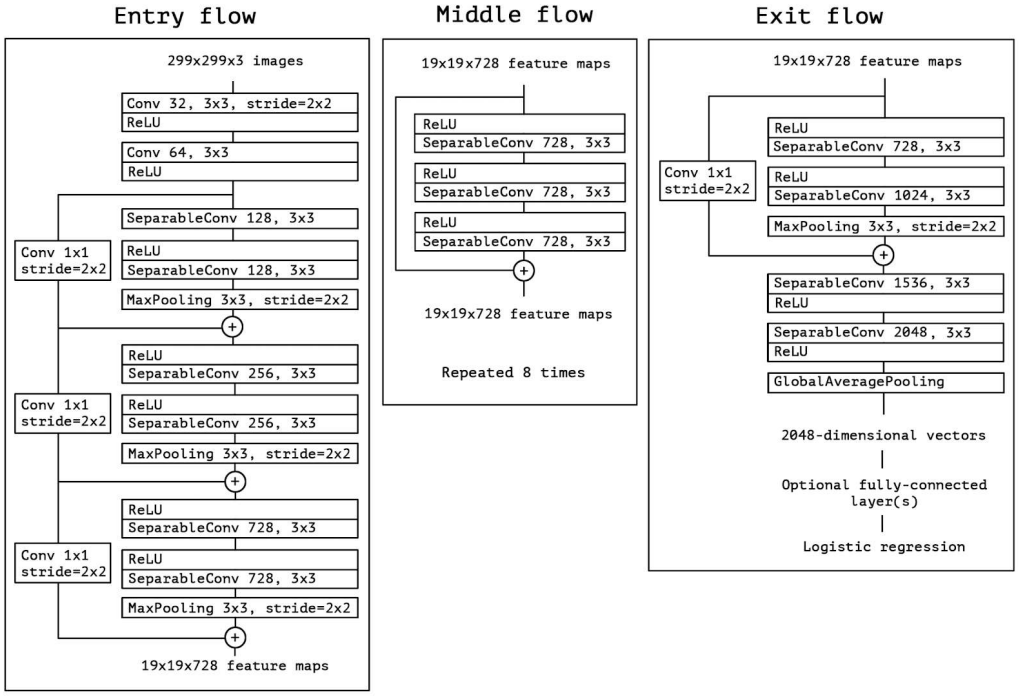


Gambar 9. Arsitektur *ResNet50*

Resnet50[13] diperkenalkan oleh Kaiming He et al[20]. Arsitektur ini cukup revolusioner pada saat itu karena arsitektur ini menjadi state of the art pada saat itu tidak hanya dalam klasifikasi, namun dalam semua kategori termasuk *object detection*, dan *semantic segmentation*. Kaiming He et al[20] mengusulkan untuk menggunakan *residual block*. *Block* ini adalah *block* yang ada pada tiap lapis arsitektur CNN *ResNet* dan menjadi fundamental dari arsitektur tersebut. *Block* ini menambahkan suatu jalan pintas yang berfungsi sebagai fungsi identitas, yang secara tidak langsung akan melewati proses training untuk satu layer atau lebih, sehingga membuat sesuatu yang bernama *residual block*[21].

Residual block[21] dapat mencegah terjadinya *vanishing gradient* yaitu suatu keadaan dimana hasil gradien yang dipelajari oleh model, tidak dapat mencapai layer pertama karena mengalami perkalian berkali-kali sehingga layer pertama tidak menerima gradien apa-apa, atau secara singkatnya, hal ini menyebabkan suatu CNN tidak dapat belajar dari error yang telah dikalkulasi. *Residual Block* digambarkan dengan panah yang melengkung seperti pada gambar 9.

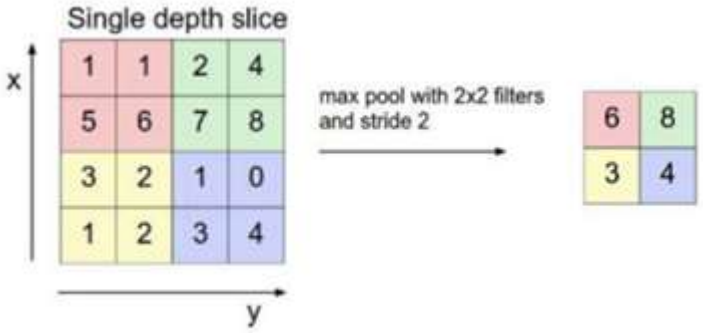
Xception[14] adalah singkatan dari *Xtreme of Inception*. Seperti namanya arsitektur ini terinspirasi dari *Inceptionv3*. *Xception*[14] sedikit lebih unggul daripada *inceptionv3* untuk dataset imagenet, dan secara signifikan mengungguli *InceptionV3* pada kasus klasifikasi citra dengan dataset sebanyak 350 juta citra dengan 17.000 class. *Xception*[14] memiliki parameter yang sama dengan *InceptionV3* namun lebih unggul dalam hal efisiensi[22]



Gambar 10. Arsitektur *Xception*

Pada gambar 10 terdapat *convolutional layer* yang tidak biasa yaitu *separable convolutional*. *Separable conv*olutinoal ini berbeda dengan *convolutional* pada umunya. Separable conv melakukan operasi *dot product* yang lebih sedikit daripada *convolutional* biasanya sehingga parameter yang digunakan akan lebih sedikit.

3.4 *Pooling Layer*



Gambar 11. Ilustrasi *Max Pooling*

Proses pada layer ini adalah untuk mereduksi input secara spasial untuk mengurangi jumlah parameter. Terdapat 3 jenis pooling yaitu *max pooling*[15], *min pooling*[15], dan *average pooling*[15]. Pada gambar 11 dilakukan *max pooling* dengan menggunakan *filter* ukuran 2x2 dan *stride* sebesar 2. Ukuran *filter* memengaruhi cakupan *pooling*-nya dan *stride* akan memengaruhi pergeseran *filter*-nya. *Max pooling* akan memilih nilai terbesar sesuai dengan bentuk *filter*-nya, kemudian bergeser sesuai dengan nilai *stride*-nya. Begitu juga dengan *min pooling* dan *average pooling*, masing-masing akan memilih nilai yang terkecil dan rata-ratanya.

3.5 *Learning Rate*

Learning rate merupakan salah satu parameter training untuk menghitung nilai koreksi *weights* pada waktu proses *training*. Nilai *learning rate* ini berada pada range nol 0 sampai 1. Semakin besar nilai *learning rate*, maka proses *training* akan berjalan semakin cepat. Namun apabila nilai *learning rate* relatif terlalu besar, pada umumnya proses *training* dapat melampaui keadaan optimal yaitu pada saat dicapai nilai *error* yang paling minimal

3.6 *Optimizer*

Optimizer adalah algoritma yang digunakan untuk mengganti parameter seperti *weight* dan *bias* pada *neural network* untuk mengurangi *error*/*loss*. Ada banyak sekali algoritma *optimizer* seperti *Gradient Descent*, *Stochastic Gradient Descent*, *Adaptive Moment Estimation, RMSprop*, dan lain sebagainya.

Ide utama dari RMSprop adalah mempertahankan rata-rata gradien kuadrat untuk setiap *weights*. Dan kemudian kami membagi gradien dengan akar kuadrat rata-rata. Itulah mengapa disebut RMSprop (*root mean square*) yang memiliki persamaan sebagai berikut:

$$E[g^2]_t = \beta E\,[g^2]_{t-1} + (1-\beta)(\frac{\delta C}{\delta W}^2)$$
$$W_t = W_{t-1} - \frac{\eta}{\sqrt{E[g^2]_t}}\frac{\delta C}{\delta W}$$

3.7 *Activation Function*

Activation Function adalah persamaan matematika yang akan menentukan output dari sebuah *neural network*. Fungsi tersebut ada di setiap *neuron* dan akan menentukan apakah *neuron* tersebut aktif atau tidak berdasarkan input *neuron*-nya. *Activation function* dibagi menjadi dua yaitu *linear* dan *non-linear*. *Linear* memiliki persamaan sebagai berikut:

$$f(x) = x$$

Kemudian ada juga *non-linear*. Ada banyak contoh *non-linear activation function* seperti *Reactified Linear Unit*(ReLU) dan *softmax*. *ReLU* memiliki persamaan sebagai berikut:

$$R(z) = \max(0, z)$$

Karena persamaan berikut yang memiliki nilai terendah 0, maka relu akan mengubah semua nilai yang kurang dari 0 menjadi 0. Adapun *softmax* memiliki persamaan sebagai berikut:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Secara matematika, fungsi *softmax* (z) adalah vektor dari input ke output *layer* dan j adalah indeks dari output unit yang bernilai 1,2,3…K. *Softmax* biasanya digunakan di akhir *layer* dan digunakan pada kasus *multi-class classification*.

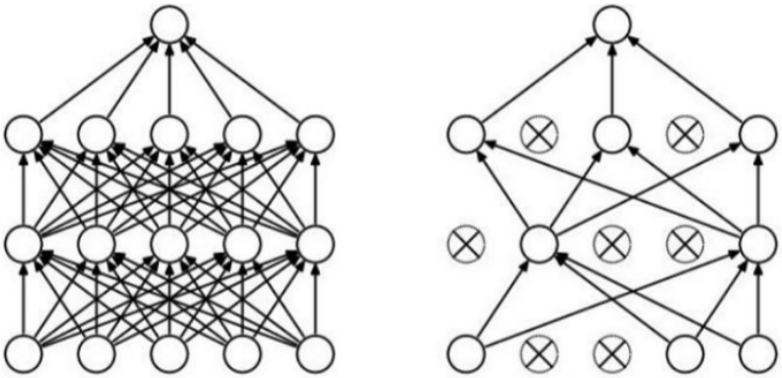
3.8 *Loss Function*

Loss function adalah persamaan matematika yang digunakan untuk menghitung nilai *loss*. Nilai *loss* inilah yang akan digunakan pada proses *back propagation* untuk mengevaluasi parameter seperti *weight* dan *bias* supaya *neural network* menjadi lebih baik. *Loss function* dibagi menjadi dua yaitu *classification* dan *regression*. Bedanya adalah *classification* akan memprediksi nilai kontinyu sedangkan *regression* akan memprediksi nilai diskrit.

Salah satu dari *loss function* dari *classification* adalah *categorical cross entropy* atau bisa juga disebut sebagai *softmax loss*. Ini digunakan untuk *multiclass classification*. *Categorical cross entropy* memiliki persamaan matematika sebagai berikut:

$$CE = -\log(\frac{e^{s_p}}{\sum_j^C e^{s_j}})$$

3.9 *Dropout*



Gambar 12. Sebelum memakai *Dropout*(kiri) dan setelah memakai *Dropout*(kanan)

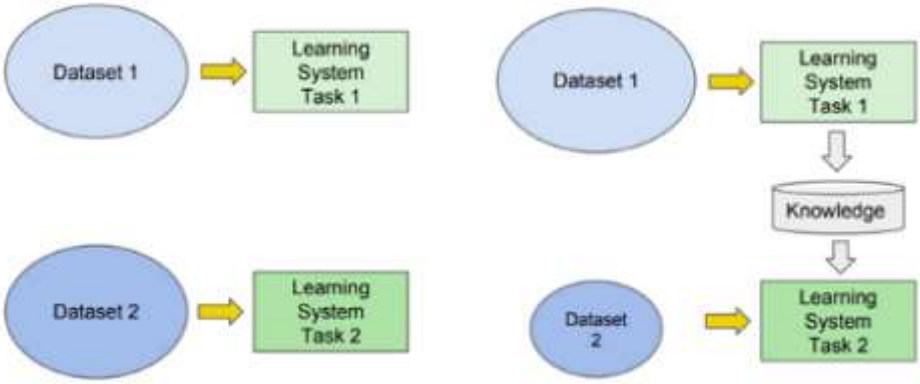
Dropout[17] adalah sebuah teknik regularisasi yang digunakan pada *neural networks*. Seperti pada gambar 12 tersebut cara kerja *dropout*[17] yaitu dengan memilih *neuron* secara acak kemudian *neuron* tersebut tidak akan dipakai selama *training*. Hal ini akan mempercepat proses *training* itu sendiri karena *neuron* yang terlewati juga lebih sedikit. Selain itu *Dropout*[17] juga digunakan untuk mengurangi *overfitting*. *Dropout*[17] dapat digunakan di *extraction layer* maupun di *fully connected layer*.

3.10 *Batch Normalization*

Batch Normalization adalah teknik yang digunakan untuk mengkonversi output dari layer menjadi sebuah standar *neural network* atau biasa disebut dengan normalisasi. Teknik ini akan mempercepat proses *learning* karena tidak ada nilai *activation* yang terlalu tinggi atau terlalu rendah.

Batch Normalization dapat digunakan untuk menggunakan learning rate yang tinggi dan tidak terlalu membutuhkan inisiasi. *Batch Normalization* juga bertindak sebagai *regularizer* seperti *dropout*. *Batch Normalization* dapat mencapai akurasi yang sama dengan 14 *step training* yang lebih sedikit[18]

3.11 *Transfer Learning*



Gambar 13. Alur *Learning* Konvensional(kiri) dan alur *Transfer Learning*(kanan)

Transfer Learning adalah pengembangan dari proses learning pada masalah yang baru dengan cara *transfer knowledge* dari masalah yang berhubungan sebelumnya yang sudah dilakukan *learning*. Seperti pada gambar 13 bahwa pada tiap masalah maka diperlukan proses *learning* yang berbeda. Namun jika masalah yang dihadapi berhubungan atau cenderung sama, maka knowledge dari masalah tertentu dapat digunakan pada masalah lainnya daripada harus dilakukan *learning* dari awal pada tiap masalah tersebut.

3.12 *Fine Tuning*

Jika pada *transfer learning* digunakan sebuah *knowledge* untuk menyelesaikan masalah lainnya, maka *fine tuning* akan mengoptimasi *knowledge* tersebut untuk mendapatkan hasil yang lebih baik lagi. Optimasi *knowledge* tersebut dengan berbagai cara yaitu mengganti jumlah *layer* yang dipakai, mengganti nilai *learning rate*-nya, mengganti ukuran filternya, dan lain sebagainya.

4. **Evaluasi**

4.1 Dataset dan Parameter

Dataset yang digunakan berasal dari Oxford-IIIT[5]. Dataset dibagi menjadi data training sebanyak 1068 citra, data validation sebanyak 528 citra, dan data testing sebanyak 797. Pada dataset tersebut terdapat 12 ras kucing diantaranya:

1. Abyssinian
2. Bengal
3. Birman
4. Bombay
5. British Shorthair
6. Egyptian Mau
7. Maine Coon
8. Persian
9. Ragdoll
10. Russian Blue
11. Siamese
12. Sphynx

Parameter yang digunakan yaitu *RMSprop* sebagai *optimizer* dengan *learning rate* sebesar 0.0001. Ukuran citra pada input di tiap masing-masing model *VGG16*, *InceptionV3*, *ResNet50*, dan *Xception* yaitu 224 x 224 x3, 224 x 224 x 3, 299 x 299 x 3, dan 299 x 299 x 3.

model *VGG16* menggunakan 5 *max pooling, InceptionV3* menggunakan 4 *max pooling* dan 9 *average pooling*, *ResNet50* hanya menggunakan 1 *max pooling layer* saja, dan *Xception* menggunakan 4 *max pooling layer.*

Pada penelitian ini juga digunakan *dropout* pada model *VGG16 + Transfer Learning*, *InceptionV3 + Transfer Learning*, dan *InceptionV3 + Transfer Learning + Fine Tuning* pada bagian *fully connected layer* yang berjumlah 2 *dropout* dengan nilai masing-masing 0.6, dan 0.6. Sedangkan pada model *ResNet50 + Transfer Learning + Fine Tuning* pada bagian *fully connected layer* digunakan 3 *dropout* dengan masing-masing nilainya 0.7, 0.6, dan 0.6. Sedangkan pada model *Xception + Transfer Learning* dan *Xception + Transfer Learning + Fine Tuning* pada bagian *fully connected layer* digunakan 1 *dropout* saja dengan nilai 0.6.

4.2 Hasil Pengujian

Tabel 1. Hasil Pengujian Tiap Model

Model	**Training**	**Testing**
InceptionV3	88.01%	28.98%
InceptionV3 + Transfer Learning	94.38%	82.43%
InceptionV3 + Transfer Learning + Fine Tuning	**94.19 %**	**84.94%**
VGG16	95.41%	24.34%
VGG16 + Transfer Learning	**87.73 %**	**60.85 %**
VGG16 + Transfer Learning + Fine Tuning	100%	59.97%
ResNet50	91.48%	28.98%
ResNet50 + Transfer Learning	50.09%	20.95%
ResNet50 + Transfer Learning + Fine Tuning	**97.94%**	**71.39%**
Xception	99.91%	31.99%
Xception + Transfer Learning	96.54%	87.45%
Xception + Transfer Learning + Fine Tuning	**99.63%**	**93.75%**

Setelah dilakukan *testing* maka didapat hasil akurasi yang terbaik di tiap modelnya pada tabel 1 ketika menggunakan kombinasi antara *transfer learning* dan *fine tuning*. Dari model-model tersebut didapat hasil terbaik jatuh kepada model *Xception* dengan akurasi *training* sebesar 99.63% dan akurasi *testing* sebesar 93.75%. Sedangkan akurasi terburuk yaitu pada model ResNet50 dengan akurasi *training* sebesar 50.09% dan akurasi *testing* sebesar 20.95%.

4.2 Analisis Hasil Pengujian

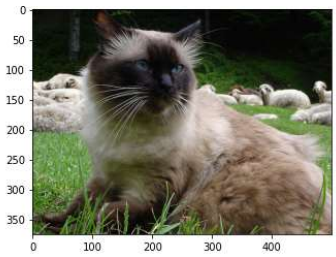
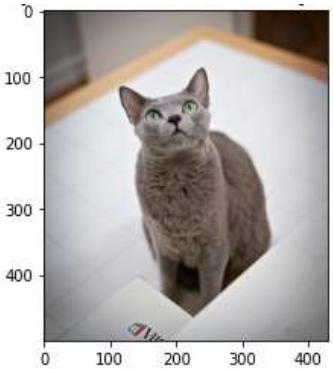
Setelah dilakukan penelitian pada model-model yang digunakan didapatkan hasil seperti pada tabel 2 bahwa model *Xception* + *Transfer Learning* + *Fine Tuning* menghasilkan akurasi tertinggi.

Tabel 2. Hasil Pengujian Terbaik Tiap Model

Model	**Training**	**Testing**
VGG16 + Transfer Learning	87.73%	60.85%
ResNet50 + Transfer Learning + Fine Tuning	97.94%	71.39%
InceptionV3 + Transfer Learning + Fine Tuning	94.19%	84.94%
Xception + Transfer Learning + Fine Tuning	**99.63%**	**93.75%**

Namun jika dilihat secara kasat mata, model tersebut masih dapat melakukan kesalahan klasifikasi pada citra tertentu. Tabel 3 menunjukkan citra dan prediksi kelas yang salah diklasifikasi. Beberapa citra tersebut memang memiliki ciri yang sangat mirip.

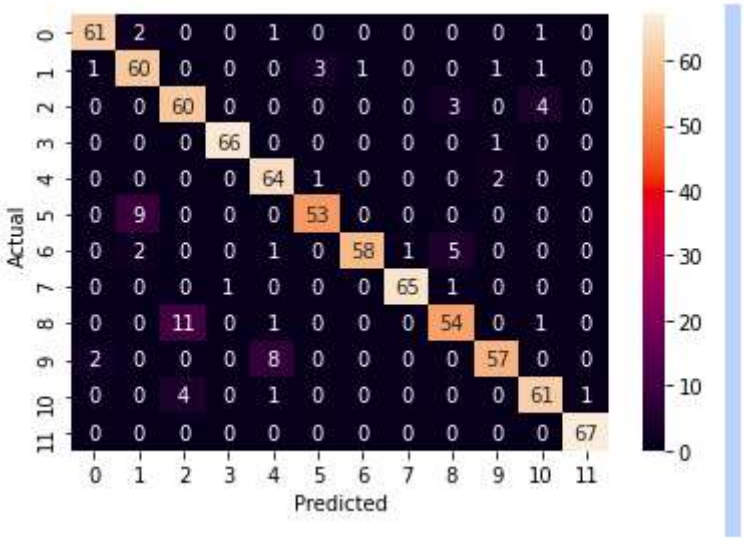
Tabel 3. Contoh Citra yang Mengalami Kesalahan Klasifikasi

Citra	Kelas	Prediksi	Gambaran Citra Prediksi
	Siamese	Birman	
	Russian Blue	British Shorthair	

Metode untuk mengukur performansi dari sebuah sistem yang dibangun adalah dengan menggunakan metriks *accuracy, precision, recall,* dan *f-1 score*. Nilai daripada metriks-metriks tersebut didapatkan dari tabel *confusion matrix* yang ada di tabel 4. Tabel ini mengandung nilai *True Positive, True Negative, False Positive,* dan *False Negative*.

Tabel 4. *Confusion Matrix*

	Positive (Predicted Class)	***Negative (Predicted Class)***
Positive (True Class)	TP *(True Positive)*	FN *(False Negative)*
Negative (True Class)	FP *(False Positive)*	TN *(True Negative)*



Gambar 14. *Confusion Matrix* masing-masing kelas. Kelas 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, dan 11 adalah representasi dari kelas Abyssinian, Bengal, Birman, Bombay, British Shorthair, Egyptian Mau, Maine Coon, Persian, Ragdoll, Russian Blue, Siamese, dan Sphynx

Setelah mendapatkan nilai *True Positive, True Negative, False Positive,* dan *False Negative* seperti pada gambar 14 maka dapat diketahui nilai pada masing-masing metriks dengan formulasi berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ Score = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall}$$

Tabel 5. Nilai dari Metriks Akurasi, Precision, Recall, dan F1-Score pada Model Xception + Transfer Learning + Fine Tuning187.3

Model	Akurasi	Precision	Recall	F1-Score
Xception + Transfer Learning + Fine Tuning	93.75%	93.74%	93.56%	93.64%

5. Kesimpulan

Pada penelitian ini disimpulkan bahwa model Xception + Transfer Learning + Fine Tuning adalah model yang terbaik. Model ini menghasilkan akurasi, precision, recall, dan f1-score dengan nilai 93.75%, 93.74%, 93.56%, dan 93.64%. Adapun saran untuk penelitian selanjutnya adalah pada saat proses *training* dilakukan sebaiknya menggunakan perangkat dengan spesifikasi yang mumpuni, karena model-model yang diuji memberikan beban komputasi yang besar.

Daftar Pustaka

[1] Borwarnginn, P., Thongkanchorn, K., Kanchanapreechakorn, S., & Kusakunniran, W. (2019, October). Breakthrough Conventional Based Approach for Dog Breed Classification Using CNN with Transfer Learning. In 2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE) (pp. 1-5). IEEE.
[2] Wang, Z., Zhao, D., & Hong, K. ECE 228 PROJECT DOG BREED CLASSIFICATION.
[3] Ayanzadeh, A., & Vahidnia, S. (2018). Modified Deep Neural Networks for Dog Breeds Identification. Preprints.
[4] Atabay, H. A. Deep Learning for Horse Breed Recognition.
[5] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, C. V. Jawahar. [Online]. https://www.robots.ox.ac.uk/~vgg/data/pets/
[6] ma7555. [Online]. https://kaggle.com/ma7555/cat-breeds-dataset
[7] Parsania, P., & Virpari, P. V. (2014). A review: Image interpolation techniques for image scaling. Int. J. Innov. Res. Comput. Commun. Eng, 2, 7409-7413.
[8] Prajapati, A., Naik, S., & Mehta, S. (2012). Evaluation of different image interpolation algorithms. International Journal of Computer Applications, 58(12), 6-12.
[9] Bloice, Marcus D., Christof Stocker, and Andreas Holzinger. (2017). "Augmentor: An Image Augmentation Library for Machine Learning." arXiv preprint arXiv:1708.04680.
[10] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. Journal of Big Data, 6(1), 60.
[11] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
[12] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).
[13] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[14] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).
[15] Ye, C., Zhao, C., Yang, Y., Fermüller, C., & Aloimonos, Y. (2016, October). Lightnet: A versatile, standalone matlab-based environment for deep learning. In Proceedings of the 24th ACM international conference on Multimedia (pp. 1156-1159).
[16] Zeiler, M. D., & Fergus, R. (2013). Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint arXiv:1301.3557.
[17] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
[18] Ioffe, Sergey, and Christian Szegedy. (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." arXiv preprint arXiv:1502.03167.
[19] Chinnamgari, S. K. (2019). R Machine Learning Projects: Implement Supervised, Unsupervised, and Reinforcement Learning Techniques Using R 3. 5. Birmingham: Packt Publishing.
[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv:1512.03385 [cs], Dec 2015. arXiv: 1512.03385.
[21] Mahmud, K. H., Adiwijaya, Al Faraby, S. (2019). Klasifikasi Citra Multi-Kelas Menggunakan Convolutional Neural Network. e-Proceeding of Engineering : Vol.6, No.1 April 2019(pp. 2127)
[22] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).