JEPIN
(Jurnal Edukasi dan Penelitian Informatika)
ISSN(e): 2548-9364 / ISSN(p) : 2460-0741

Vol. 9
No. 1
April 2023

Klasifikasi Jenis Rempah Menggunakan Convolutional Neural Network dan Transfer Learning

Alvin Eka Putra[#1], Mohammad Farid Naufal[#2], Vincentius Riandaru Prasetyo[#3]

[#]*Teknik Informatika, Universitas Surabaya*
Jl. Raya Kalirungkut, Kali Rungkut, Kec. Rungkut, Kota SBY, Jawa Timur 60293
[1]alvineka0201@gmail.com
[*2]faridnaufal@staff.ubaya.ac.id
[3]vincent@staff.ubaya.ac.id

Abstrak— **Rempah merupakan salah satu kekayaan yang dimiliki oleh Indonesia. Berdasarkan data yang dimiliki Negari Rempah Foundation, terdapat sekitar 400 hingga 500 spesies rempah di dunia dan 275 jenis rempah terdapat di Asia Tenggara terutama di Indonesia. Jenis rempah beragam dan memiliki kemiripan satu dengan yang lain sehingga sulit untuk dibedakan. Maka dari itu untuk mempertahankan pengetahuan mengenai rempah-rempah yang dimiliki Indonesia, diperlukan aplikasi klasifikasi jenis rempah yang akurat sehingga pengetahuan masyarakat tentang rempah tetap terjaga. Selain itu di bidang industri dapat meningkatkan efisiensi dalam industri rempah. Penggunaan teknologi dalam klasifikasi jenis rempah dapat meningkatkan efisiensi dalam industri rempah. Dengan teknologi yang tepat, waktu yang dibutuhkan untuk mengidentifikasi jenis rempah dapat dipercepat, dan juga meminimalkan risiko kesalahan manusia. Keterbatasan citra rempah juga menjadi permasalahan pada klasifikasi jenis rempah. Convolutional Neural Network (CNN) dengan arsitektur transfer learning adalah metode klasifikasi citra yang memiliki performa yang baik pada dataset dengan jumlah yang terbatas. Eksperimen yang dilakukan menggunakan 6 arsitketur CNN, yaitu Xception, MobileNetV2, DenseNet201, VGG16, VGG19, dan ResNet50. Terdapat 10 jenis rempah yang diklasifikasikan yaitu jahe, kunyit, kunci, adas, merica, laos, jintan, kencur, temulawak, dan ketumbar. Berdasarkan hasil eksperimen yang dilakukan Xception adalah arsitektur terbaik dengan F1 Score sebesar 96.99%.**

Kata kunci— **CNN, Rempah, Deep Learning, Transfer Learning**

I. PENDAHULUAN

Rempah merupakan salah satu kekayaan yang dimiliki oleh bangsa Indonesia. Rempah sendiri didapatkan dari mengambil bagian tanaman seperti batang, daun kulit kayu, umbi, rimpang, akar, biji, bunga atau bagian tumbuhan lainnnya [1]. Berdasarkan data yang dimiliki Negari Rempah Foundation [2], terdapat sekitar 400 – 500 spesies rempah di dunia, lebih dari setengah jumlah tersebut atau 275 terdapat di Asia Tenggara dan jumlah tersebut didominasi oleh Indonesia. Hal tersebut membuat Indonesia mendapat julukan *Mother of Spices*.

Seiring dengan berjalannya waktu, masyarakat Indonesia mulai menikmati kemajuan teknologi dan pengetahuan yang belum tentu dapat beriringan dengan pengenalan budaya Indonesia. Hal tersebut mengakibatkan banyak budaya Indonesia yang mulai dilupakan, salah satunya budaya pengenalan rempah yang dimiliki oleh Indonesia [3]. Masyarakat cenderung menggunakan rempah instan yang dinilai lebih cepat dan praktis dibandingkan dengan racikan menggunakan rempah asli. Pengunaan rempah instan mulai menimbulkan ketidaktahuan mengenai bentuk dan nama rempah [3]. Berdasarkan dari hasil survey yang telah dilakukan penelitian ini dengan meminta sebanyak 100 responden untuk menebak 5 jenis rempah asli Indonesia, hanya 31% responden yang berhasil menebak lebih dari 3 jenis rempah dengan benar. Oleh karena itu dapat disimpulkan bahwa pengetahuan mengenai jenis rempah masih rendah.

Penelitian klasifikasi citra rempah sudah pernah dilakukan sebelumnya. Isna et al [4] melakukan klasifikasi menggunakan CNN untuk 3 jenis rempah yaitu ginseng, jahe, dan lengkuas. Akurasi yang dihasilkan pada penelitian ini adalah 88.89%. Penelitian ini belum menggunakan arsitektur transfer learning. Tanuwijaya et al [5] menggunakan CNN dengan arsitektur VGG16 yang dimodifikasi dan AlexNet. Penelitian ini dapat mengenali 5 jenis rempah, yaitu jahe, kencur, kunyit, lengkuas dan temulawak. Dataset yang digunakan diambil dari Kaggle yang terdiri dari 100 dataset *training* dan 25 dataset *testing*. Dari 100 epoch, VGG16 mendapatkan rata-rata akurasi sebesar 81% dengan nilai *loss* 0.446, nilai *recall* 76.7%, dan nilai *precision* sebesar 81.8%. AlexNet mendapatkan nilai rata-rata akurasi sebesar 79%, *loss* sebesar 3.586, *recall* sebesar 79.1% dan *precision* sebesar 80.8%. Nadya et al [6] melakukan klasifikasi jenis rempah menggunakan algoritma naïve bayes. Citra dilakukan segmentasi terlebih dahulu menggunakan metode otsu. Fitur yang digunakan

Submitted 15-09-2022; Revised 05-03-2023; Accepted 07-04-2023

adalah Grey Level Co-occurrence Matrix (GLCM) dan Red Green Blue (RGB). Jenis rempah yang diklasifikasikan adalah kunyit, temulawak, jahe dan lengkuas. Akurasi yang dihasilkan penelitian ini masih rendah yaitu 52%. Tan et al [7] melakukan klasifikasi terhadap 8 jenis rempah yaitu gingko, ginseng, barley, ficus, jujube, longan, lotus seed, dan lyceum menggunakan CNN. Terdapat 2 layer convolution dan pooling. Akurasi yang dihasilkan adalah 93%.

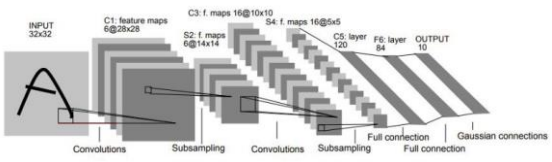
Pada penelitian sebelumnya terdapat maksimal 8 jenis rempah yang diklasifikasikan. Selain itu akurasi yang didapat maksimal 93%. Permasalahan lain adalah pada dataset rempah memiliki keterbatasan jumlah, sehingga membutuhkan metode klasifikasi yang dapat belajar dari data yang terbatas. Penelitian sebelumnya belum ada yang mencoba membandingkan arsitektur transfer learning untuk klasifikasi citra rempah. Transfer learning adalah metode untuk mempercepat proses klasifikasi dan memiliki performa baik dikarenakan memiliki pre-trained model [8]. Transfer learning dapat belajar dengan baik pada dataset yang memiliki jumlah terbatas. Penelitian ini melakukan klasifikasi terhadap 10 jenis rempah yaitu jahe, kunyit, kunci, adas, merica, laos, jintan, kencur, temulawak, dan ketumbar menggunakan CNN dengan 6 arsitektur *transfer learning* yaitu Xception, MobileNetV2, DenseNet201, VGG16, VGG19, dan ResNet50.

II. TINJAUAN PUSTAKA

Pada sub bab ini akan menjelaskan mengenai beberapa dasar teori yang digunakan dalam dalam dasar melakukan klasifikasi CNN.

A. *Convolutional Neural Network* (CNN)

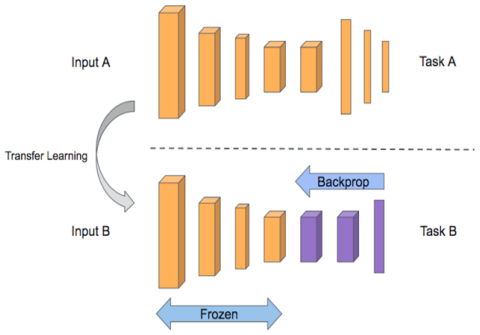
CNN merupakan sebuah jaringan syaraf tiruan yang terinsipirasi oleh sistem saraf biologis yang terdiri dari 3 *layers*, yaitu *convolutional layers*, *pooling layers*, dan *fully-connected layers* [9]. Perbedaan CNN dengan *Artificial Neural Network* (ANN) adalah CNN diutamakan untuk digunakan untuk pengenalan pola dalam sebuah gambar. Struktur dari CNN terdiri dari 3 lapis yaitu *convolutional layers*, *pooling layers*, dan *fully-connected layers*. Jika beberapa lapis tersebut digabungkan menjadi satu maka jadilah arsitektur dari CNN. Contoh dari arsitektur CNN ditunjukan pada Gambar 1.



Gambar. 1 Contoh arsitektur CNN

B. *Transfer Learning*

Transfer learning adalah sebuah teknik *machine learning* yang menggunakan model yang telah dilatih sebelumnya untuk menyelesaikan sebuah tugas dan kemudian digunakan untuk menyelesaikan tugas lain yang berkaitan dengan tugas awal. Teknik ini biasanya digunakan jika memiliki dataset yang sedikit. Proses *transfer learning* adalah dengan membekukan beberapa *convolutional layers* di awal dan hanya melatih beberapa lapisan terakhir yang digunakan untuk melakukan klasifikasi. Lapisan yang dibekukan digunakan untuk

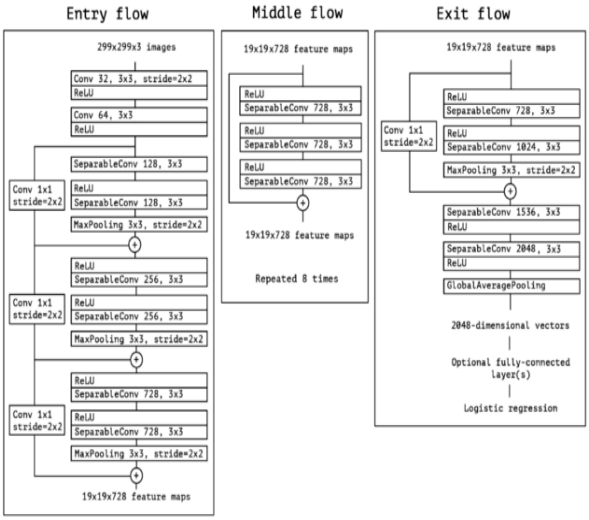


Gambar. 2 Proses *tranfer learning*

mengekrtraksi fitur yang dapat diaplikasikan pada semua gambar seperti sudut, gradien, dan pola. Untuk lapisan yang tidak dibekukan digunakan untuk mengekstrasi fitur lainnya yang terdapat pada *dataset* baru. Dengan begitu model tersebut dapat digunakan meskipun pada awalnya model yang dilatih menggunakan kategori *dataset* yang berbeda dengan *dataset* yang baru. Proses dari *transfer learning* ditunjukan pada Gambar 2. *Image database* yang digunakan di transfer learning adalah ImageNet yang merupakan *dataset* yang berisi lebih dari 14 juta gambar. 6 *pre-trained model* yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Xception

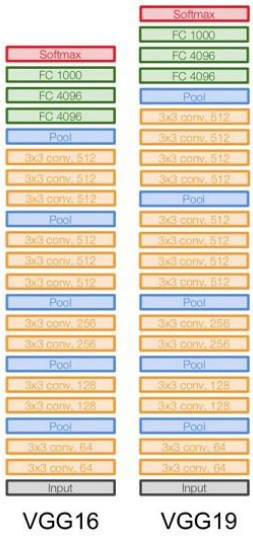
Xception merupakan arsitektur CNN yang diadaptasi dari arsitektur Inception. Xception menggunakan hipotesis dari Inception namun ditingkatkan lagi ke level yang ekstrim, hal ini merupakan asal dari penamaan Xception atau Extreme Inception [10]. Arsitektur ini menggunakan *depth-wise separable convolutions* sebagai pengganti dari modul Inception. Xception terdiri dari 3 flow, yaitu *entry flow*, *middle flow*, dan *exit flow*. Data yang digunakan sebagai input nantinya akan masuk melalui *entry flow* terlebih dahulu kemudian melalui *middle flow* sebanyak 8 kali kemudian menuju ke *exit flow*. Gambar 3 menunjukkan arsitektur Xception.



Gambar. 3 Arsitektur Xception

2. VGG16 dan VGG19

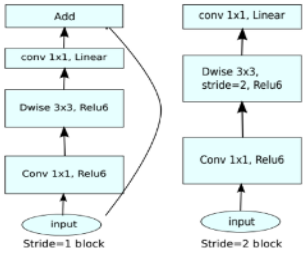
VGG16 & VGG19 merupakan sebuah model CNN yang memiliki jaringan yang lebih dalam dan menggunakan *filter* yang kecil [11]. *Filter* yang digunakan pada *convolutional layer* ada 3x3 semua. VGG menggunakan jumlah *filter* yang berukuran kecil karena memiliki jumlah parameter yang sedikit dan memiliki jumlah *convolutional layer* yang banyak [12]. VGG16 dan VGG19 memiliki arsitektur yang mirip, perbedaannya hanya pada jumlah *convolutional layers* saja. VGG16 memiliki 16 *convolutional layers* sedangkan VGG19 memiliki 19 *convolutional layers* [13] [14]. Gambar 4 menunjukkan arsitektur VGG16 dan VGG19.



Gambar. 4 Arsitektur VGG16 dan VGG19

3. MobileNetV2

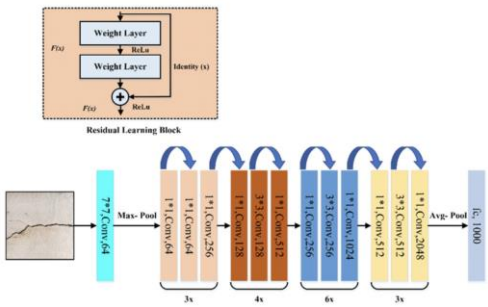
MobileNetV2 adalah arsitektur yang dibuat berdasarkan ide dari MobileNetV1 yang menggunakan *depthwise separable convolutions* [15]. Namun MobileNetV2 mengenalkan 2 fitur baru, yaitu *linear bottlenecks* dan *shortcut connections*. MobileNetV2 membutuhkan *computation time* yang lebih dibandingkan dengan MobileNetV1 namun tetap dapat mendapatkan akurasi yang lebih baik juga [16]. Gambar 5 menunjukkan arsitektur MobileNetV2.



Gambar. 5 Arsitektur MobileNetV2

4. ResNet50

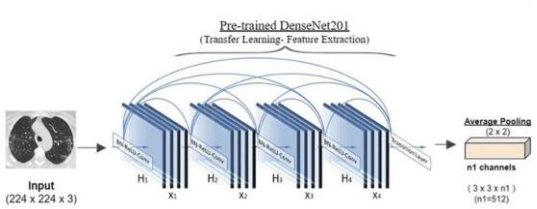
ResNet atau disebut *residual network* merupakan sebuah neural network yang dibuat untuk mengatasi masalah degradasi yang disebabkan oleh *neural network* yang menggunakan banyak *layers*. ResNet menerapkan konsep *skip connections* untuk meningkatkan akurasi yang menurun yang disebabkan oleh degradasi. *Skip connections* bekerja dengan 2 cara, yaitu pertama dengan mengurangi permasalahan menghilangnya *gradient* dengan membuat sebuah jalan pintas alternatif untuk gradient tersebut dan yang kedua membuat model dapat mempelajari *identity function*. Dengan begitu *layers* pada *neural network* dapat mempelajari *identity function* dengan lebih mudah sehingga meningkatkan efisiensi dari *deep neural network* yang memiliki *layers* yang banyak, selain itu juga dapat mengurangi tingkat *error* yang dihasilkan. Hal ini memungkinkan untuk membuat sebuah *deep neural network* dengan lapisan yang jauh lebih banyak tanpa menghadapi masalah degradasi. ResNet50 merupakan salah satu dari variasi ResNet [17]. Gambar 6 menunjukkan arsitektur ResNet50.



Gambar. 6 Arsitektur ResNet50

5. DenseNet201

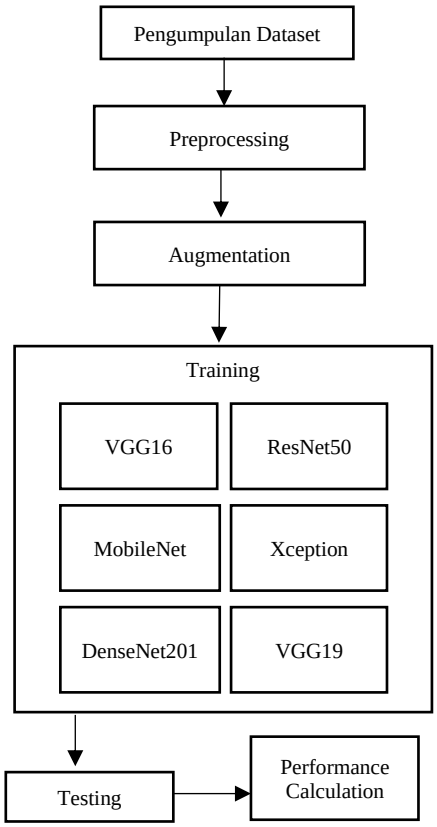
DenseNet atau Densely Connected Convolutional Network ditemukan oleh Gao Huang sebagai penemu utama arsitektur DenseNet [18]. DenseNet merupakan salah satu arsitektur yang ditemukan untuk menyelesaikan permasalahan kerika sebuah neural network semakin dalam atau memiliki layers yang semakin banyak maka akan muncul masalah hilangnya gradient. Gradient ini akan semakin hilang dengan bertambahnya layers yang digunakan. Dengan hilangnya gradient ini menyebabkan layers di awal kehilangan kemampuan untuk mempelajari basic low-features. Detail arsitektur DenseNet dapat dilihat pada Gambar 7.



Gambar. 7 Arsitektur DenseNet201

METODOLOGI PENELITIAN

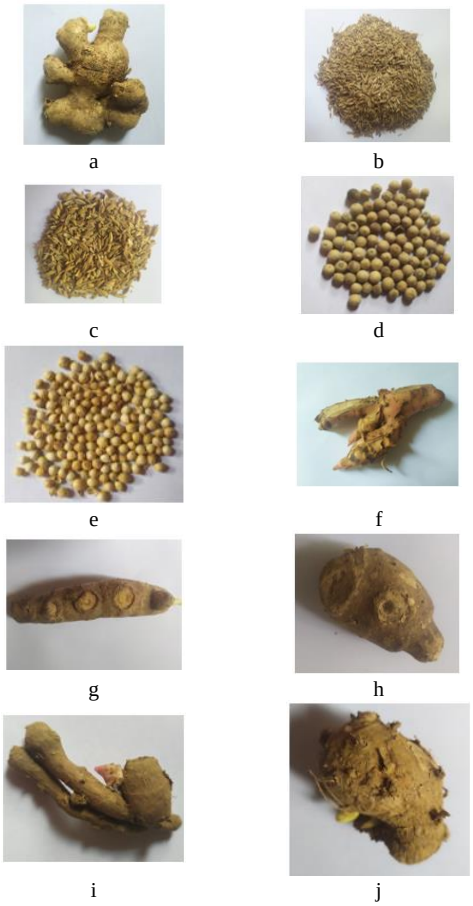
Penelitian ini memiliki 5 tahapan, yaitu pengumpulan *dataset*, *pre-processing*, *augmentation*, *training*, *testing*, perhitungan performa. Alur dari metodologi penelitian dapat dilihat pada Gambar 8.



Gambar. 8 Metodologi Penelitian

C. Pengumpulan *Dataset*

Dataset yang digunakan dikumpulkan menggunakan kamera *smartphone*. Spesifikasi kamera Jumlah *class* yang digunakan adalah sebanyak 10, yaitu jahe, adas, jintan, merica, ketumbar, laos, kunyit, kencur, kunci, dan temulawak. Tiap *class* memiliki total 100 gambar yang berarti total *dataset* yang digunakan adalah 1000 gambar. Contoh *dataset* telah ditunjukan pada Gambar 9.



Gambar 9. Contoh dataset jahe (a), adas (b), jintan (c), merica (d), ketumbar (e), laos (f), kunyit (g), kencur (h), kunci (i), temulawak (j).

B. Pre-processing

Preprocessing di citra digital adalah proses persiapan atau pengolahan awal citra digital sebelum masuk ke tahap pengolahan selanjutnya. Tujuan dari preprocessing adalah untuk memperbaiki kualitas citra, menghilangkan noise, dan menyesuaikan karakteristik citra agar sesuai dengan kebutuhan analisis atau pengolahan citra selanjutnya. Preprocessing yang digunakan dalam penelitian ini adalah *resizing*. Proses ini digunakan untuk menyesuaikan gambar yang digunakan untuk *dataset training*, *testing* atau saat melakukan klasifikasi gambar. Semua gambar diubah menjadi ukuran 299x299 pixel. Citra kemudian diubah

menjadi RGB dan dibagi dengan 255 sehingga tiap pixel akan bernilai 0 – 1 untuk mempercepat waktu komputasi.

C. Augmentation

Proses ini bertujuan untuk memperbanyak dataset yang digunakan dalam proses *training*. Parameter yang digunakan dalam penelitian ini adalah *zoom_range* dengan nilai 0.2, *fill_mode* dengan nilai *nearest*, *rotation_range* dengan nilai 0.4, *width_shift_range* dengan nilai 0.2, *height_shift_range* dengan nilai 0.2, *shear_range* dengan nilai 0,2, dan *horizontal_flip* dengan nilai *true*.

D. Training

Proses *training* dilakukan dengan menggunakan *k fold cross validation* dengan nilai k = 10. Jumlah dataset yang digunakan adalah sebanyak 1000 gambar dengan jumlah *class* 10 sehingga 900 *dataset* digunakan untuk proses *training* dan 100 *dataset* digunakan untuk proses *validation*. *Dataset* dilakukan proses *image pre-processing* dan *image augmentation* terlebih dahulu sebelum digunakan. Penelitian ini menggunakan 6 arsitektur dan dibandingkan untuk mendapatkan model dengan performa yang terbaik. Arsitektur yang digunakan adalah Xception, MobileNetV2, VGG16, VGG19, DenseNet201, dan ResNet50. Parameter dari model yang dilatih dapat dilihat pada Tabel I. *Training* tiap arsitektur memiliki 4 macam variasi dengan perbedaan pada *top layers* yang digunakan untuk klasifikasi. Spesifikasi komputer yang digunakan untuk proses training dapat dilihat pada Tabel II. *Top layers* yang digunakan pada tiap variasi dapat dilihat pada Tabel III.

TABEL I
PARAMETER TRAINING MODEL

Parameter	Tipe
Jumlah epoch	20
Optimizer	Adam
Loss Function	Categorical, Cross Entropy
Metric	Accuracy

TABEL II
SPESIFIKASI KOMPUTER

Spesifikasi	Nilai
GPU	Nvidia Tesla P100 16GB
RAM	25 GB
Storage	166 GB

E. Testing

Proses *testing* menggunakan 100 gambar dari dataset yang tidak pernah digunakan baik dalam proses *training* maupun *validation*. Gambar yang digunakan dalam *testing* hanya akan melalui proses *testing* untuk mengubah ukuran menjadi 299x299 dan membagi tiap pixel dengan 255. Model yang sebelumnya telah selesai dilatih dan disimpan dalam file dengan ekstensi .h5 digunakan kembali untuk melakukan proses *testing*.

TABEL III
VARIASI TOP LAYER

Variasi	*Layers*
1	1. *Flatten* 2. *Dense*(256) 3. *Dense*(10)
2	1. *GlobalAveragePooling2D* 2. *Dense(256)* 3. *Dense(10)*
3	1. *MaxPooling2D* (2x2) 2. *Flatten* 3. *Dense*(256) 4. *Dense*(10)
4	1. *MaxPooling2D*(2x2) 2. *MaxPooling2D*(2x2) 3. *Flatten* 4. *Dense*(128) 5. *Dense*(10)

F. Perhitungan Performa

Pada tahapan ini performa dari tiap arsitektur transfer learning dihitung. Nilai performa yang dihitung adalah akurasi, precision, recall, dan F1 Score dan dapat dilihat secara berturut turut pada persamaan (1) (2) (3) (4). Variabel TP menunjukkan jumlah True Positive, TN menunjukkan jumlah True Negative, FP menunjukkan jumlah False Positive, dan FN menunjukkan jumlah False Negative.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1\ Score = \frac{Precision\ x\ Recall}{Precision + Recall} \quad (4)$$

III. HASIL DAN PEMBAHASAN

Pada bagian ini dijelaskan mengenai hasil testing dari masing-masing variasi top layer tiap arsitektur. Tabel 4 menunjukan hasil perbandingan performa untuk semua arsitektur dengan kombinasi variasi top layer. Gambar 10 menunjukkan perbandingan performa dari tiap arsitektur dengan mengambil variasi top layer terbaik. Sedangkan gambar 11 menunjukkan perbandingan waktu komputasi per epoch dari tiap arsitektur dengan variasi top layer terbaik. Pada variasi 1 Xception mendapatkan nilai terbaik pada semua *metric* kecuali pada waktu komputasi. Waktu komputasi adalah waktu yang dibutuhkan oleh sebuah arsitektur untuk menjalani proses training per epoch. Waktu komputasi terbaik didapatkan oleh MobileNetV2 yaitu 31.5256 detik. Nilai terburuk didapatkan oleh arsitektur ResNet50 yang memiliki perbedaan nilai yang lebih dari setengah jika dibandingkan dengan Xception,

namun nilai komputasi terburuk didapatkan oleh VGG19 yaitu sebesar 78.1086 detik.

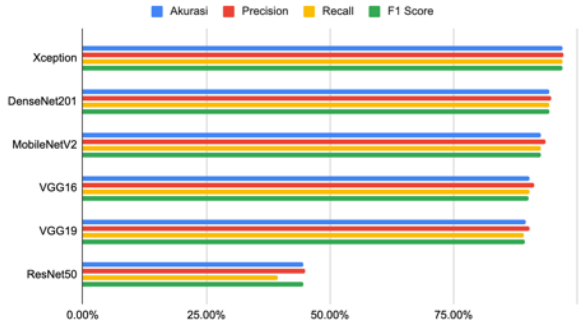
Selanjutnya pada variasi kedua, Xception masih mendapatkan nilai akurasi, precision, recall, dan F1 Score terbaik. Namun untuk waktu komputasi terbaik masih didapatkan oleh MobileNetV2 pada variasi 2. Nilai performa akurasi, precision, recall, dan F1 Score terburuk didapatkan oleh ResNet50 dan waktu komputasi terburuk didapatkan oleh VGG19. Jika dibandingkan dengan variasi 1, terdapat peningkatan nilai pada semua *metric* pada variasi 2.

TABEL IV
PERFORMA TIAP ARSITEKTUR TRANSFER LEARNING

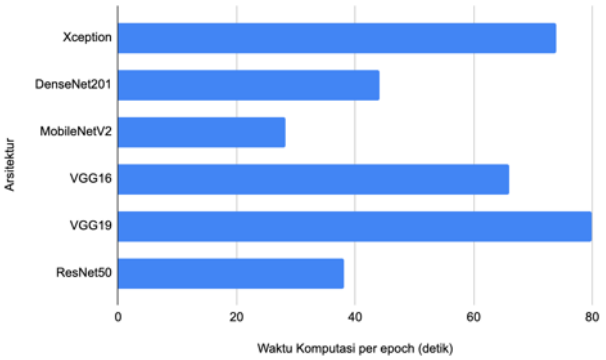
Arsitektur	Variasi	Akurasi	Precision	Recall	F1 Score	Waktu Komputasi
Xception	1	93,80%	95,01%	93,80%	93,76%	63,0239
	2	96,60%	96,87%	96,60%	96,53%	74,0947
	3	**97,00%**	**97,28%**	**97,00%**	**96,99%**	**73,9812**
	4	94,40%	95,03%	94,40%	94,28%	77,6292
DenseNet201	1	93,70%	94,49%	91,90%	91,88%	44,5251
	2	**94,40%**	**94,77%**	**94,40%**	**94,38%**	**44,1909**
	3	94,00%	94,55%	94,00%	93,98%	43,8777
	4	90,00%	91,75%	95,55%	90,00%	45,898
MobileNetV2	1	91,98%	93,12%	91,90%	91,88%	31,5256
	2	**92,60%**	**93,71%**	**92,60%**	**92,66%**	**28,211**
	3	90,90%	92,42%	90,90%	90,73%	33,1057
	4	88,30%	90,27%	88,30%	87,90%	29,8522
VGG16	1	**90,40%**	**91,25%**	**90,40%**	**90,17%**	**65,9681**
	2	81,90%	84,19%	74,22%	81,90%	67,221
	3	90,00%	91,42%	90,00%	89,80%	73,9867
	4	66,00%	68,74%	66,00%	63,86%	72,2262
VGG19	1	87,60%	89,26%	78,11%	87,60%	87,212
	2	80,00%	82,79%	79,20%	80,00%	79,215
	3	**89,60%**	**90,29%**	**89,33%**	**89,50%**	**79,9379**
	4	69,10%	72,15%	69,10%	67,10%	72,2262
RESNET50	1	40,00%	34,45%	40,00%	31,56%	39,8586
	2	40,10%	31,27%	40,10%	31,05%	40,656
	3	**44,70%**	**45,04%**	**39,45%**	**44,70%**	**38,1259**
	4	18,22%	9,33%	18,90%	9,94%	42,548

Selanjutnya adalah variasi ketiga, seperti pada variasi 1 dan 2 Xception masih memiliki nilai terbaik dan MobileNetV2 juga masih mendapatkan waktu komputasi terbaik. Nilai terburuk masih juga diraih oleh ResNet50 dan waktu komputasi terburuk masih didapatkan oleh VGG19. Pada variasi 3 juga terdapat kenaikan nilai pada semua *metric* dibandingkan dengan variasi 1 dan 2.

Pada variasi 4, terdapat penurunan nilai namun Xception tetap mendapatkan nilai terbaik dibandingkan dengan arsitektur lainnya pada variasi 4. Hasil dari variasi 4 lebih baik dari variasi 1 namun lebih buruk jika dibandingkan dengan variasi 2 dan 3.



Gambar 10. Perbandingan performa tiap arsitektur dengan variasi top layer terbaik



Gambar 11. Perbandingan waktu komputasi per epoch tiap arsitektur dengan variasi top layer terbaik.

Nilai terbaik dari 4 variasi didapatkan oleh Xception pada variasi 3 dengan nilai akurasi sebesar 97%, nilai presisi sebesar 97.27%, nilai *recall* sebesar 97% dan F1 Score sebesar 96.99%, namun untuk nilai komputasi terbaik didapatkan oleh MobileNetV2 pada variasi 2 yaitu sebesar 28.211 detik. DenseNet201 mendapatkan nilai kedua terbaik setelah Xception dengan perbedaan rata-rata performa 3% untuk tiap variasi. Nilai terburuk didapatkan oleh ResNet50 pada variasi 4 dengan nilai akurasi sebesar 1,822, nilai presisi sebesar 9,32%, nilai *recall* 18,9%, dan nilai f1 sebesar 9.93%, namun untuk nilai komputasi terburuk didapatkan oleh VGG19 sebesar 79,9379 detik.

IV. KESIMPULAN

Berdasarkan hasil yang telah dijabarkan pada bagian hasil dan pembahasan disimpulkan bahwa arsitektur Xception dengan variasi 3 paling cocok digunakan untuk melakukan klasifikasi rempah. Jika dibutuhkan model yang dapat melakukan klasifikasi dalam waktu singkat maka dapat dipertimbangkan untuk menggunakan MobileNetV2 pada variasi 2 yang memiliki waktu komputasi terendah dibandingkan dengan semua arsitektur pada semua variasi. Namun perbedaan nilai akurasi terbaik yang didapatkan oleh MobileNetV2 dan Xception memiliki *gap* yang besar yaitu sebesar 5%. Jika memang dibutuhkan alternatif yang memiliki perbedaan akurasi yang lebih tinggi dan waktu komputasi yang lebih rendah dibandingkan dengan Xception maka dapat dipilih arsitektur DenseNet201 yang

hanya memiliki *gap* sebesar 3% namun memiliki waktu komputasi yang lebih baik dibandingkan dengan Xception.

Pada pengembangan penelitian selanjutnya dapat diimplementasikan *image segmentation* yang dapat membantu model untuk mengklasifikasikan gambar yang memiliki latar belakang yang rumit atau terdapat banyak objek. Selain itu dapat dicari alternatif artsitektur lainnya yang memiliki akurasi yang tinggi seperti Xception namun memiliki waktu komputasi yang rendah seperti pada MobileNetV2.

REFERENSI

[1] L. Hakim, *Rempah & Herba Kebun-Pekarangan Rumah Masyarakat*, no. 164. 2015.

[2] "Beranda - Negeri Rempah." https://negerirempah.org/id/ (accessed Sep. 02, 2022).

[3] D. Marihandono and B. Kanumoyoso, *Rempah, Jalur Rempah, Dan Dinamika Masyarakat Nusantara*. 2016.

[4] I. Wulandari, H. Yasin, and T. Widiharih, "Klasifikasi Citra Digital Bumbu Dan Rempah Dengan Algoritma Convolutional Neural Network (Cnn)," *J. Gaussian*, vol. 9, no. 3, pp. 273–282, 2020, doi: 10.14710/j.gauss.v9i3.27416.

[5] E. Tanuwijaya and A. Roseanne, "Modifikasi Arsitektur VGG16 untuk Klasifikasi Citra Digital Rempah- Rempah Indonesia Classification of Indonesian Spices Digital Image using Modified VGG 16 Architecture," *Matrik J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, vol. 21, no. 1, pp. 191–198, 2021, doi: 10.30812/matrik.v21i1.xxx.

[6] N. P. Batubara, D. Widiyanto, and N. Chamidah, "Klasifikasi rempah rimpang berdasarkan ciri warna rgb dan tekstur glcm menggunakan algoritma naive bayes," *Inform. J. Ilmu Komput.*, vol. 16, no. 3, p. 156, 2020, doi: 10.52958/iftk.v16i3.2196.

[7] J. W. Tan, K. M. Lim, and C. P. Lee, "Herb Classification with Convolutional Neural Network," *3rd IEEE Int. Conf. Artif. Intell. Eng. Technol. IICAIET 2021*, Sep. 2021, doi: 10.1109/IICAIET51634.2021.9573706.

[8] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11141 LNCS, pp. 270–279, 2018, doi: 10.1007/978-3-030-01424-7_27.

[9] Y. Lecun, L. Bottou, Y. Bengio, and P. Ha, "LeNet," *Proc. IEEE*, no. November, pp. 1–46, 1998.

[10] F. Chollet, "XCeption: Deep Learning with Depthwise Separable Convolutions," *Comput. Vis. Found.*, 2016, doi: 10.4271/2014-01-0975.

[11] T. Ben Abdallah, I. Elleuch, and R. Guermazi, "Student Behavior Recognition in Classroom using Deep Transfer Learning with VGG-16," *Procedia Comput. Sci.*, vol. 192, pp. 951–960, Jan. 2021, doi: 10.1016/J.PROCS.2021.08.098.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *undefined*, vol. 2016-December, pp. 770–778, Dec. 2016, doi: 10.1109/CVPR.2016.90.

[13] J. Xiao, J. Wang, S. Cao, and B. Li, "Application of a Novel and Improved VGG-19 Network in the Detection of Workers Wearing Masks," *J. Phys. Conf. Ser.*, vol. 1518, no. 1, pp. 0–6, 2020, doi: 10.1088/1742-6596/1518/1/012041.

[14] A. Victor Ikechukwu, S. Murali, R. Deepu, and R. C. Shivamurthy, "ResNet-50 vs VGG-19 vs training from scratch: A comparative analysis of the segmentation and classification of Pneumonia from chest X-ray images," *Glob. Transitions Proc.*, vol. 2, no. 2, pp. 375–381, Nov. 2021, doi: 10.1016/J.GLTP.2021.08.027.

[15] L. Zhang, J. Wang, B. Li, Y. Liu, H. Zhang, and Q. Duan, "A MobileNetV2-SENet-based method for identifying fish school feeding behavior," *Aquac. Eng.*, vol. 99, p. 102288, Nov. 2022, doi: 10.1016/J.AQUAENG.2022.102288.

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4510–4520, Jan. 2018, Accessed: May 27, 2021. [Online]. Available: http://arxiv.org/abs/1801.04381

[17] M. B. Hossain, S. M. H. S. Iqbal, M. M. Islam, M. N. Akhtar, and I. H. Sarker, "Transfer learning with fine-tuned deep CNN ResNet50 model for classifying COVID-19 from chest X-ray images," *Informatics Med. Unlocked*, vol. 30, p. 100916, Jan. 2022, doi: 10.1016/J.IMU.2022.100916.

[18] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 2261–2269, Aug. 2016, Accessed: May 27, 2021. [Online]. Available: http://arxiv.org/abs/1608.06993