Pengenalan Bentuk Tangan dengan Convolutional Neural Network (CNN)

Muhammad Zein Ersyad[1], Kurniawan Nur Ramadhani[2], Anditya Arifianto[3]

[1,2,3] Fakultas Informatika, Universitas Telkom, Bandung
[4]Divisi Digital Service PT Telekomunikasi Indonesia
[1]zersyad@students.telkomuniversity.ac.id, [2]kurniawannr@telkomuniversity.ac.id
[3]anditya@telkomuniversity.ac.id

Abstrak — Pada saat ini terdapat beberapa sistem pengenalan gestur tangan yang memiliki tingkat komputasi dan upaya untuk persiapan perangkat yang cukup merepotkan. Tujuan utama penulis adalah membangun sistem yang mendekati pengenalan gestur / bentuk tangan yang memiliki tingkat komputasi dan upaya yang lebih rendah, serta mendapatkan hasil performa sistem pada saat proses pengujian. Sistem yang dibangun menggunakan Convolutional Neural Network (CNN) dengan input berupa citra dan output berupa label dari input tersebut. Pengembangan sistem ini sudah dilakukan hingga tahap pengujian, model yang dihasilkan mendapatkan akurasi klasifikasi sebesar 88% yang diuji dengan 2142 citra dan digambarkan dengan confusion matrix sebagai alat ukur performansi.

Kata kunci : bentuk tangan, cnn, confusion matrix.

1 Pendahuluan

1.1 Latar Belakang

Pengenalan gestur merupakan topik pada bidang ilmu computer science and language technology dengan tujuan mengenali arti dari bahasa tubuh manusia dengan menggunakan algoritma matematika. Gestur berasal dari motion atau state tubuh, gestur yang paling sering digunakan berasal dari bagian tangan atau wajah. Pada pengenalan gestur tangan, terdapat beberapa cara yang dapat dilakukan, yang pertama dapat menggunakan sinyal radar yang memancarkan aliran elektromagnetik [1], yang kedua dapat menggunakan kamera dan algoritma computer vision untuk menerjemahkan motion atau state tangan. Pengenalan gestur dapat dilakukan dengan tiga implementasi, yaitu dengan menggunakan perangkat sarung tangan, 3-dimensional location hand points dan dengan raw visual data. Implementasi pertama yang mengharuskan menggunakan sarung tangan dengan kabel yang tersambung pada sarung tangan dan komputer meskipun memiliki kecepatan proses dan tingkat akurasi yang lebih baik. Implementasi kedua menggunakan hand point extraction yang mana membutuhkan biaya komputasi yang tinggi. Implementasi terakhir menggunakan sensor gambar, seperti kamera, sensor inframerah atau sensor kedalaman yang mana tergantung kepada pengguna [2].

Convolutional Neural Network (CNN) atau biasa disebut ConvNet, CNN mengekstrak fitur dari input yang berupa gambar lalu mengubah dimensi gambar tersebut menjadi lebih kecil tanpa merubah karakteristik gambar tersebut. CNN terdiri dari neurons yang memiliki bobot dan bias. Setiap neuron menerima inputan dan diteruskan dengan melakukan perkalian titik pada setiap neuron tersebut. Pada layer terakhir CNN masih memiliki loss function seperti SVM/Softmax. Dengan menggunakan sensor gambar seperti kamera, upaya yang digunakan untuk mempersiapkan perangkat jauh lebih mudah, misalnya seperti hampir setiap orang saat ini sudah memiliki perangkat kamera pada smartphone. Penelitian ini memberikan suatu inovasi berdasarkan adanya masalah pada pengenalan gestur dalam bidang computing yaitu pengenalan gestur tangan memiliki beberapa opsi, beberapa opsi tersebut memiliki cost dan resource yang tinggi, dan juga upaya mempersiapkan perangkat yang cukup merepotkan.

Dilihat dari permasalahan tersebut, penelitian ini mengembangkan suatu solusi yaitu dengan menggembangkan sistem pengenalan bentuk tangan dengan menggunakan model yang sudah dilatih sebelumnya, model dapat digunakan dalam perangkat pintar seperti smartphone. Penelitian ini menggunakan suatu metode yang dapat merealisasikan tujuan dari penelitian ini yaitu terdapat beberapa implementasi seperti implementasi dengan sarung tangan untuk mendapatkan bentuk tangan, implementasi kedua menggunakan hand point extraction dengan biaya komputasi yang tinggi, implementasi ketiga menggunakan sensor kamera. Penelitian ini menggunakan pengenalan bentuk tangan yang merupakan salah satu proses dari pengenalan gestur tangan. Pengembangan sistem untuk pengenalan bentuk tangan dapat dibantu dengan menggunakan salah satu metode yaitu Convolutional Neural Network. CNN adalah salah satu dari beberapa metode Deep Learning yang memiliki hasil paling signifikan dalam pengenalan citra [3]. CNN mempunyai kelebihan dibandingkan dengan metode lainnya yaitu dapat memproses

komputasi untuk melatih model sebelum melakukan pengujian, dengan itu saat pengujian dilakukan tidak memerlukan pelatihan yang berulang, model bisa digunakan dimana saja. Namun metode CNN memiliki kelemahan yaitu semakin banyak dataset yang digunakan pada proses pelatihan, semakin lama juga proses pelatihan [3] dan model yang dilatih berdasarkan dataset yang diberikan, oleh karena itu hasil dari pengenalan bentuk bergantung pada input (dataset) yang diberikan pada saat pelatihan. Penelitian ini mengharapkan rencana pembangunan sistem bentuk tangan dengan CNN dapat berhasil sesuai tujuan yang telah ditentukan.

1.2 Topik dan Batasannya

Topik yang diutamakan ialah klasifikasi gambar dengan menggunakan Convolutional Neural Network.Pada klasifikasi memiliki batasan untuk pengenalan tujuh bentuk tangan dengan masing-masing bentuk tangan diberikan label Palm Oke, Fist, C, L, Index point dan Up, pemilihan tujuh kelas diambil karena keterbatasan resource yang digunakan pada saat proses pelatihan model CNN.

1.3 Tujuan

Tujuan yang dicapai dalam pengerjaan tugas akhir ini adalah membangun sistem untuk mengenali bentuk tangan dan mendapatkan hasil dari kinerja performa dari sistem setelah dilakukan pengujian oleh data uji.

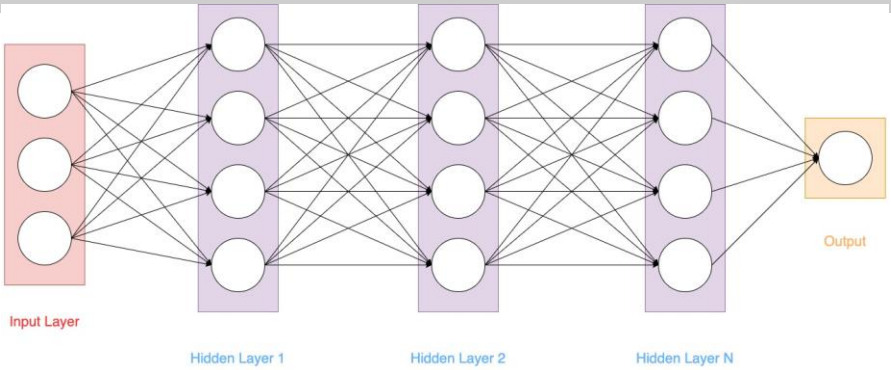
2 Kajian Pustaka

2.1 Studi Terkait

Pada [3] dilakukan penelitian berupa klasifikasi citra menggunakan CNN dengan menggunakan dataset Caltech101. Pada penelitian tersebut dilakukan praproses dan pengolahan pada data input yang berupa citra, praproses yang dilakukan yaitu wrapping dan cropping, selanjutnya citra juga diolah kembali dengan melakukan konversi menjadi grayscale dan merubah ukuran menjadi 140 x 140. Peneliti melakukan proses pelatihan dengan menggunakan tiga tahap, tahap pertama dilakukan proses feedforward, tahap kedua dilakukan proses backpropagation dan tahap ketiga melakukan perhitunga gradient dari jaringan konvolusi. Pengujian dilakukan dengan cara menggunakan bobot dan bias dari proses pelatihan dan data yang diuji adalah lima kategori unggas dan tiga kategori lainnya, hasil klasifikasi dari pengujian menunjukan bahwa persentase keberhasilan pada kategori unggas sebesar 20% dan pada kategori lainnya menunjukan persentase keberhasilan sebesar 50%, bila digabungkan klasifikasi citra menghasilkan persentase kebenaran 20% hingga 50%.

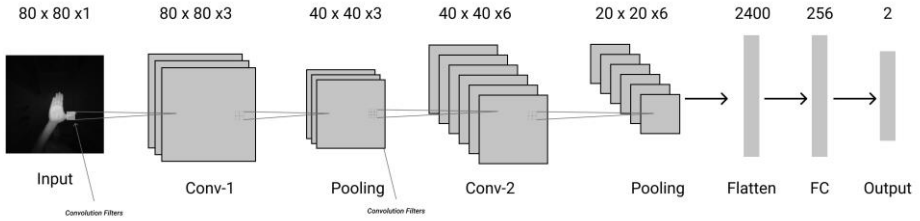
2.2 Dasar Teori

2.2.1 Convolution Neural Network (CNN) CNN merupakan versi peningkatan dari Multi Layer Perceptron (MLP), yang mana MLP memiliki satu dimensi pada setiap neuronnya sedangkan CNN melakukan peningkatan ukuran dimensi menjadi dua dimensi. CNN memiliki tiga layer utama, yaitu Convolution Layer, Pooling Layer dan Fully Connected Layer.



Gambar 1. Model MLP Sederhana

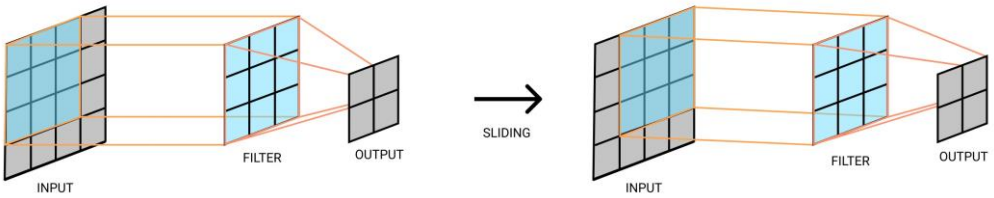
Pada jaringan MLP seperti gambar 1 memiliki beberapa hidden layer dengan neurons pada setiap layernya. Setiap hubungan antar layer memiliki parameter bobot yang dihasilkan. MLP menerima input data satu dimensi dengan input tersebut dilakukan penyebaran pada hidden layer hingga menghasilkan output [3].



Gambar 2. Model CNN Sederhana

Terlihat pada gambar 2 input dari CNN berupa dua dimensi yang setiap nilai pada input dapat disebut de- ngan neurons, input tersebut dilakukan proses convolution yang mana proses tersebut dapat menghasilkan output Convolution Layer lalu untuk mengurangi biaya computational dengan cara mengurangi dimensi spasialnya dapat dilakukan proses Pooling, begitu seterusnya hingga proses Flatten yang mana proses tersebut dilakukan untuk me- nyiapkan input yang awalnya dua dimensi menjadi satu dimensi agar dapat digunakan pada Fully Connected (FC) Layer untuk didapatkan prediksi dari input tersebut.

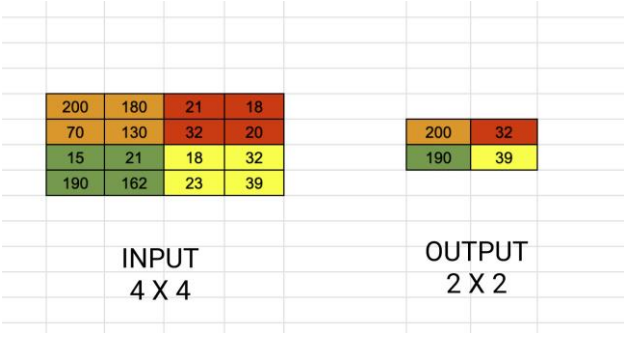
2.2.2 Convolution Layer Convolution Layer adalah hasil dari proses convolution yang mana proses tersebut melakukan pengenalan pola, pola yang dimaksud diantaranya ialah edges, shapes, colors, textures pada suatu gambar. Pola yang dihasilkan bergantung pada filter, filter tersebut berupa matriks dengan nilai awal didalamnya berupa nilai random, lalu ukuran filter definisikan nilai baris dan kolom nya.



Gambar 3. Proses Convolution

Seperti yang terlihat pada 3, sebuah filter akan melakukan proses dot product pada neuron yang berada didalam lingkup filter, lalu melakukan proses sliding untuk menghitung nilai selanjutnya, hasil dari proses tersebut akan menjadi output pada proses convolution, proses ini akan terus dilakukan selama ada input yang diterima.

2.2.3 Pooling Layer Pooling merupakan sebuah matriks yang nilai baris dan kolomnya sudah didefinisikan. Perhitungan pooling memiliki beberapa cara, dimulai dari MaxPooling dan AveragePooling, prosesnya mirip de- ngan convolution yaitu melakukan proses sliding, perbedaannya ialah pada proses pooling yang didapatkan adalah nilai terbesar atau rata-rata pada setiap neuron. Hasil dari proses pooling adalah berupa Pooling Layer yang sudah ter-reduce nilai spasialnya berdasarkan ukuran baris dan kolomnya seperti yang terlihat pada gambar 4. Pada [4] juga disebutkan bila performansi model tanpa pooling layer dan dengan pooling layer itu sama.



Gambar 4. Proses MaxPooling

2.2.4 Dropout Untuk mencegah permasalahan Overfit, proses Dropout dilakukan proses tersebut dilakukaan sebelum melakukan prediksi input, yaitu saat proses pelatihan model, yang Dropout lakukan adalah mengabaikan secara acak neurons atau mengeluarkan neurons dari jaringannya. Karena mengabaikan neurons secara acak, efek dari menggunakan Dropout ialah jaringan yang dibangun menjadi semakin lebih ringan [5].

2.2.5 Fully Connected Layer Dapat dilihat pada gambar 2 output yang dihasilkan pada proses convolution dan pooling berbentuk matriks, fungsi aktivasi yang digunakan adalah softmax, pada softmax input harus berupa vektor maka dari itu perlu dilakukan proses perubahan dari bentuk matriks kedalam vektor yang bisa disebut dengan proses Flatten, lalu setelah dilakukan perubahan output vektor yang dihasilkan akan digunakan untuk mendapatkan prediksi dari sebuah input.

2.2.6 Activation Function Activation function pada Neural Network memiliki fungsi untuk menentukan output neurons menjadi sebuah input, dapat dilihat pada gambar 1 masing-masing input menjadi penentu bagi neuron yang ada pada layer selanjutnya, perlu diketahui variabel x pada fungsi dibawah merupakan nilai masukan pada sebuah fungsi. Berikut adalah Activation function yang digunakan :

A Rectified Linear Units (ReLU) ReLU adalah fungsi aktivasi yang memiliki kelebihan dalam fondasi matematis. Pada tahun 2011, dibuktikan untuk meningkatkan training dalam deep neural networks [6] dan menurut Alex Krizhevsky [7] fungsi ReLU memberikan peningkatan kecepatan dalam melakukan proses training, hal tersebut dikarenakan fungsi ReLU merupakan fungsi non-saturating yang berarti fungsi tersebut tidak menekan nilai aslinya menjadi range tertentu.

$$f(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases} \quad (1)$$

B Softmax Softmax adalah fungsi aktivasi yang seringkali digunakan pada setiap output layer pada model deep-learning [8]. Softmax digunakan untuk mendapatkan probabilitas dari input yang diberikan, input yang berupa vektor didapatkan dari proses Flatten lalu output dari fungsi softmax berupa nilai probabilitas dari 0 hingga 1. Fungsi softmax menggunakan model multi-class yang artinya setiap class akan mendapatkan nilai probabilitasnya lalu nilai yang tertinggi dari setiap class akan digunakan sebagai prediksi dari sebuah input.

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=0}^{k} e^{x_j}} \quad (2)$$

2.2.7 Confusion Matrix Confusion Matrix adalah sebuah metode perhitungan performansi untuk permasalahan klasifikasi, metode ini sangatlah berguna untuk mengukur nilai Recall, Precision, Accuracy, F-Measure dan yang terpenting Confusion Matrix dapat memberikan informasi perihal jumlah class prediksi yang diberikan terlepas dari benar atau salah prediksi tersebut. Pada persamaan 3, 4 dan 5 beberapa definisi yang memiliki arti sebagai berikut.True Positive memiliki arti model melakukan prediksi positive dan sesuai dengan hasil sebenarnya, True Negative memiliki arti model melakukan prediksi negative dan sesuai dengan hasil sebenarnya, False Positive memiliki arti model melakukan prediksi positive dan tidak sesuai dengan hasil sebenarnya, False Negative memiliki arti model melakukan prediksi negative dan tidak sesuai dengan hasil sebenarnya.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad (3)$$

Precision digunakan untuk mengetahui seberapa akurat model melakukan prediksi True terhadap suatu class, yang artinya dari total prediksi Positive seberapa banyak yang mengandung True atau sesuai dengan hasil sebenarnya.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (4)$$

Recall digunakan untuk mengetahui seberapa banyak model melakukan prediksi bernilai Positive dan sesuai dengan hasil sebenarnya, semakin tinggi nilainya maka semakin baik juga model tersebut dalam melakukan prediksi untuk class tersebut karena model hanya melakukan prediksi pada class tersebut.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{False Negative} + \text{True Negative}} \quad (5)$$

$$\text{Accuracy} = \frac{\text{Prediksi sesuai dengan sebenarnya}}{\text{Seluruh Data}} \quad (6)$$

Perhitungan akurasi dapat dilakukan dengan mengambil seluruh True dan membaginya dengan seluruh data yang ada, bila fungsinya disederhanakan, maka akan menjadi seperti persamaan 6. F-Measure merupakan persamaan yang didapat dari precision dan recall, yang artinya F-Measure dapat digunakan untuk menghitung precision dan recall secara bersamaan. F-Measure dapat mengukur performansi untuk machine learning classification yang mana output dapat beberapa class. Berikut persamaan F-Measure:

$$\text{F} - \text{Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

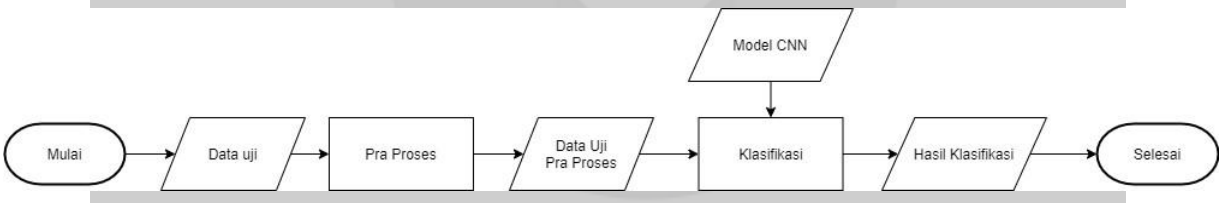
3 Sistem yang Dibangun

3.1 Diagram Alir

Proses pelatihan dan pengujian dilakukan dengan alur yang sama pada dua model arsitektur yang berbeda. Pada proses pelatihan terdapat input yang merupakan dataset dari 7 class dengan total jumlah data sebanyak 3677 dan output yang merupakan model CNN dari hasil proses pelatihan. Bisa dilihat pada gambar 5 terdapat 2 proses yang dilakukan setelah input dan sebelum output yaitu pra proses dan proses pelatihan dengan CNN. Seperti yang diterangkan pada subbab 3.3, pra proses dilakukan sebelum pelatihan salah satunya karena adanya keterbatasan komputasi lalu pada proses pelatihan juga model yang dibangun disusun seperti yang terlihat pada tabel 2.

Gambar 5. Diagram alir proses pelatihan model

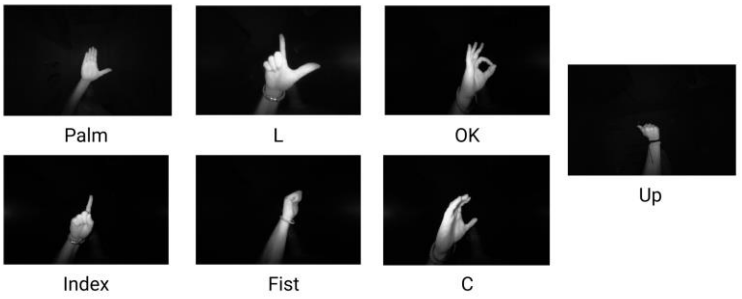
Pada proses pengujian, input yang diberikan merupakan data pengujian yang memiliki spesifikasi serupa dengan data latih lalu output dari pengujian berupa hasil klasifikasi. Pada proses ini juga terdapat pra proses, pra proses yang dilakukan adalah center crop, hal tersebut dilakukan agar ukuran gambar sama seperti saat pelatihan. Seperti yang terlihat pada gambar 6, model yang telah dilatih sebelumnya disimpan dan digunakan untuk melakukan proses klasifikasi, proses ini akan menghasilkan prediksi atau klasifikasi dari input yang diberikan, hasil prediksi berupa label gambar yang sesuai atau tidak sesuai dengan label asli.



Gambar 6. Diagram alir proses pengujian model

3.2 Dataset

Spesifikasi dataset yang digunakan terlihat seperti gambar 7, yaitu berupa susunan citra visual grayscale dari sensor inframerah yang mana hasil dari sensor inframerah tersebut memiliki hasil kontras yang cukup baik antara objek klasifikasi dan latar belakangnya, pada tangkapan sensor inframerah juga memiliki nilai kedalaman yang dapat digunakan untuk mendapatkan .



Gambar 7. Contoh Dataset

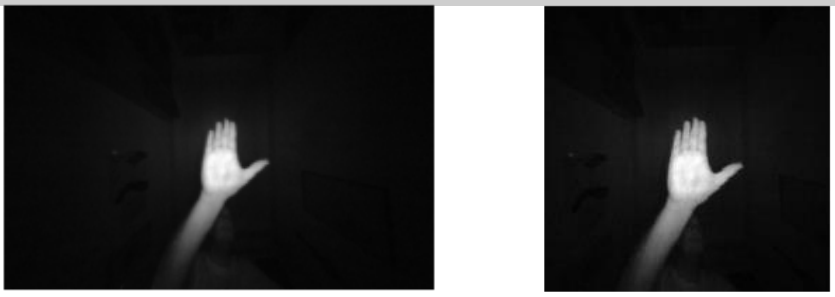
Tabel 1. Jumlah persebaran data training, validation, test

Class	Training	Validation	Test
PALM	541	224	306
L	556	198	306
FIST	560	210	306
INDEX	500	207	306
OK	507	206	306
C	528	219	306
UP	485	223	306
Total	3677	1487	2142

Dari total data latih sebanyak 5164 citra, data tersebut dipisah menjadi sekitar 70% untuk data training yaitu sebanyak 3677 citra dan sekitar 30% untuk data validation yaitu sebanyak 1487 citra, untuk persebaran dapat dilihat pada tabel 1, persebaran kelas pada data latih dan validasi kurang lebih sebesar 13% hingga 15% untuk setiap kelasnya. Pada pengujian digunakan data uji sebanyak 2142 citra dengan total 7 kelas dengan persebaran kelas sebanyak 306 untuk setiap kelasnya dan ukuran re- solusi citra kurang lebih 420 x 273. Subjek berupa lima orang pria dan lima orang wani- ta.

3.3 Pra Proses

Terlihat seperti pada gambar 8, pra Proses yang dilakukan ialah dengan melakukan crop. Crop dilakukan dari ukuran resolusi citra yang kurang lebih dengan panjang 420 dan lebar 273, karena keterbatasan komputasi, ukuran tersebut cukup besar untuk dilakukan proses training, maka dilakukan crop menjadi ukuran maksimal 250 pixel pada panjang dan lebar. Crop juga dilakukan untuk membuat ukuran kedua sisi menjadi sama, hal ini perlu dilakukan agar mendapatkan konsistensi ukuran citra pada input.



(a) Sebelum Pra Proses (b) Sesudah Pra Proses

Gambar 8. Gambar sebelum dan sesudah Pra Proses

3.4 Model

Model yang dibangun berdasarkan referensi dari arsitektur VGG16 [9] karena VGG16 memiliki jumlah layer yang lebih sedikit diantara arsitektur lainnya dan juga VGG16 berhasil memperoleh akurasi 92.7% top-5 tes akurasi pada ImageNet.

Tabel 2. Arsitektur yang digunakan pada penelitian

Name	Kernel / Pool Size	Bias
Input		
Conv2D	3 x 3 x 1	8
MaxPooling	3 x 3	
Conv2D	3 x 3 x 8	32
Conv2D	3 x 3 x 32	128
MaxPooling	3 x 3	
Conv2D	3 x 3 x 128	128
Conv2D	3 x 3 x 128	128
MaxPooling	3 x 3	
Conv2D	3 x 3 x 128	256
Conv2D	3 x 3 x 256	256
MaxPooling	3 x 3	
Flatten		
Dense		4096
Dropout		
Dense		7

Berdasarkan referensi tersebut model yang dibangun memiliki nilai input 250 x 250 x 1, ukuran input tersebut digunakan untuk meningkatkan nilai akurasi, lalu citra yang hanya memiliki nilai keabuan didalamnya, hal tersebut dapat meringankan pekerjaan komputasi didalamnya. Seperti yang telah disebutkan pada sub-subbab 2.2.6, ReLU telah terbukti memberikan peningkatan kecepatan dalam melakukan proses training. Dengan melakukan proses percobaan fitting beberapa kali, yang mana setiap percobaan merubah beberapa parameter seperti kernel size, filter size, pool size dan dropout rate. Hasil arsitektur yang dibangun setelah melakukan proses percobaan fitting secara berulang dapat dilihat pada tabel 2, model dibangun dengan 7 convolution layer, digunakan juga dropout setelah Fully Connected ke 1 dengan persentase untuk mengabaikan neurons dari jaringannya sebesar 50%, hal tersebut sudah terlampau cukup untuk mencegah terjadinya overfit setelah dilakukan beberapa percobaan fitting, MaxPooling juga dilakukan karena dapat meringankan komputasi dan juga dapat mencegah overfit. Pada perbandingan MaxPooling dan AveragePooling dengan dataset MNIST, MaxPooling dapat meningkatkan akurasi sekitar 1% hingga 1.5% dibandingkan dengan AveragePooling [10].

3.5 Pengujian

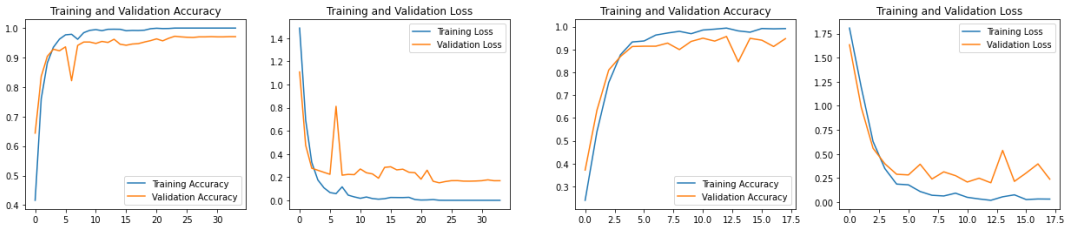
Pengujian dilakukan kepada tujuh class dengan menggunakan citra yang telah diubah ukurannya menjadi sesuai dengan ukuran input dari model. Perhitungan akurasi dilakukan dengan menggunakan Confusion Matrix dari jumlah data pengujian sebanyak 2142 data. Didapatkan Precision, Recall dari setiap class. Akurasi pengujian keseluruhan juga didapatkan untuk mengetahui seberapa baik performa akurasi model untuk melakukan prediksi pada tujuh class tersebut. Pengujian juga dilakukan dengan jumlah data sebanyak 2142 citra, hal tersebut dilakukan untuk memvalidasi konsistensi akurasi pada model yang dibangun.

4 Evaluasi

Dataset yang digunakan untuk pengujian mencapai 2142 data berformat gambar, serta class yang diprediksi berjumlah tujuh class. Sistem ini tidak dapat melakukan prediksi pada gambar selain gambar yang mendekati spesifikasi data latih, bila tetap dilakukan prediksi dengan spesifikasi gambar yang tidak mendekati data latih maka hasil prediksi klasifikasi tidak akan tepat. Pada proses pengujian, model yang telah dilatih disimpan lalu digunakan kembali untuk dilakukan proses pengujian. Data uji yang digunakan memiliki spesifikasi serupa dengan data latih yaitu susunan citra visual grayscale dari sensor inframerah hal ini dilakukan karena model memiliki batasan data latih dengan spesifikasi tersebut. Model yang akan diuji adalah Model yang dibangun pada subbab 3.4 dan karena memiliki keterbatasan komputasi, Model VGG 11 layer yang ada pada [9] dipilih dan dengan sedikit tunning dengan menambahkan dropout pada fully connected layer agar tidak terjadi overfit sebagai perbandingan.

4.1 Hasil Pengujian

Berdasarkan model yang dibangun dengan spesifikasi yang ada pada subbab 3.4, hasil dari proses pelatihan memiliki line plot seperti pada gambar 9a yang mana pada plot tersebut terlihat bahwa training loss berada dibawah validation loss dan tidak ada perkembangan dengan berkurang nya nilai validation loss. Sedangkan pada gambar 9b terlihat memiliki loss yang kurang begitu stabil pada saat proses pelatihan.



(a) Model arsitektur yang dibangun (b) Model VGG 11 layer

Gambar 9. Line plot Akurasi dan Loss pada proses pelatihan pada model dengan arsitektur pada subbab 3.4 dan model VGG 11 layer

Didapatkan juga hasil pengujian berupa Confusion Matrix. Tabel Confusion Matrix dihasilkan dari pengujian pada kedua model yang telah dilatih pada proses pelatihan.

Tabel 3. Confusion Matrix dari arsitektur yang dibangun

		Actual Values						
		PALM	L	FIST	INDEX	OK	C	UP
Predicted Values	PALM	265	1	0	1	26	13	0
	L	0	273	0	17	13	3	0
	FIST	0	0	289	4	0	10	3
	INDEX	0	7	0	296	3	0	0
	OK	4	3	1	5	282	11	0
	C	0	2	7	1	45	251	0
	UP	34	0	14	17	7	12	222
	Akurasi				0.88			

Berikut adalah tabel Confusion Matrix pada pengujian yang memiliki jumlah data uji terbanyak yaitu 2142 data uji. Terlihat pada tabel 3 bahwa class index mendapatkan jumlah prediksi benar yang paling tinggi. Jika kita lihat kembali untuk nilai prediksi pada class up, model melakukan prediksi pada gambar up adalah palm sebanyak 34 prediksi gambar. Bukan hanya class up saja yang banyak memberikan kontribusi prediksi salah, tetapi class c dan class palm juga memberikan kontribusi yang cukup banyak pada hampir semua class.

4.2 Analisis Hasil Pengujian

Tabel 4. Precision, Recall, F-score setiap class dari arsitektur yang dibangun

Class	Precision	Recall	F-Score	Total Data Uji
PALM	0.87	0.87	0.87	306
L	0.95	0.89	0.92	306
FIST	0.93	0.94	0.94	306
INDEX	0.87	0.97	0.91	306
OK	0.75	0.92	0.83	306
C	0.84	0.82	0.83	306
UP	0.99	0.73	0.84	306
Akurasi			0.88	2142

4.2.1 Performansi Pada tabel 4 akurasi keseluruhan class yang didapatkan sebesar 0.88, sementara class index mendapatkan recall sebesar 0.97 yang artinya class tersebut memiliki jumlah prediksi benar yang paling tinggi,

lalu class ok memiliki precision 0.75 yang artinya, dengan precision sekecil itu banyak gambar selain class ok yang diprediksikan sebagai class ok, hal tersebut membuat performa model dalam memprediksi class lainnya menjadi menurun.

Tabel 5. Precision, Recall, F-score setiap class dari arsitektur VGG 11 layer

Class	Precision	Recall	F-Score	Total Data Uji
PALM	0.92	0.77	0.84	306
L	0.90	0.83	0.86	306
FIST	0.86	0.76	0.81	306
INDEX	0.77	0.97	0.86	306
OK	0.60	0.91	0.73	306
C	0.89	0.72	0.79	306
UP	0.77	0.60	0.70	306
Akurasi			0.79	2142

Bisa dilihat juga pada table 5, VGG 11 layer mendapatkan hasil akurasi keseluruhan class yang didapatkan sebesar 0.79, precision terendah didapatkan oleh class OK yaitu sebesar 0.60, dari total 306 data uji yang melakukan prediksi class OK sebanyak 278 prediksi benar memprediksi class OK sisanya yaitu sebanyak 182 prediksi adalah salah sebagai class OK. Bila dibandingkan dengan model dengan arsitektur yang dibangun sebanyak 282 melakukan prediksi benar class OK sisanya yaitu sebanyak 94 prediksi adalah salah sebagai class OK.

4.2.2 Kemiripan Untuk mengetahui kenapa model melakukan prediksi data uji class up sebagai class palm, maka kita perlu melihat kembali pada data uji dari kedua class tersebut, dapat terlihat banyak kemiripan pada gambar palm dan up.

(a) Kumpulan Gambar Palm (b) Kumpulan Gambar Up

Gambar 10. Gambar yang memiliki kemiripan

Pada gambar 10 dapat dilihat, meski berbeda tetapi model melakukan prediksi bahwa kedua kumpulan gambar tersebut beberapa diantaranya memiliki kemiripan, hal tersebut berdampak pada proses prediksi model yang menjadi salah. Meski begitu precision pada up justru mendapatkan nilai yang hampir sempurna, yang artinya sangat sedikit sekali prediksi salah menjadi up itu terjadi.

4.2.3 Prediksi Klasifikasi Salah Model dapat melakukan prediksi salah pada gambar, karena akurasi model mencapai sekitar 80% maka kemungkinan model melakukan prediksi salah dapat terjadi sebanyak 20%.

(a) Gambar Asli class OK (b) Gambar Asli class C

Gambar 11. Prediksi Gambar Salah

Terlihat pada gambar 11, prediksi salah dapat terjadi karena gambar yang diprediksi memiliki kemiripan pada bentuk. Meski ciri yang ada pada gambar didapatkan pada kedua class tersebut saat model melakukan pelatihan.

5 Kesimpulan

Pada penelitian ini, penulis menggunakan CNN untuk membangun model yang digunakan untuk memprediksi input berupa citra bentuk tangan. Dengan melakukan pelatihan pada model, model yang dihasilkan dapat mengenali suatu input. Pra proses dilakukan untuk menangani keterbatasan komputasi pada saat proses pelatihan model. Dropout juga digunakan sebagai cara untuk menagani keterbatasan komputasi, lalu terdapat maxpooling yang sudah terbukti dapat meningkatkan akurasi sebesar 1% hingga 1.5% dibandingkan dengan averagepooling.

Pada saat pengujian dengan 2142 citra, model yang dibangun memiliki akurasi lebih dari 80% lebih tinggi dibandingkan dengan VGG 11 Layer yang memiliki akurasi sebesar 79%. Pada data pelatihan memang class palm dan up memiliki kemiripan pada dataset bila diperhatikan dengan kasat mata, hal tersebut terbukti dengan model sering melakukan prediksi palm sebagai up, tetapi tidak dengan sebaliknya. Untuk mendapatkan hasil akurasi klasifikasi yang lebih baik dapat dilakukan pengembangan terhadap dataset pada class yang memiliki f-score paling kecil.

Daftar Pustaka

[1]Jaime Lien, Nicholas Gillian, M. Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. Soli: Ubiquitous gesture sensing with millimeter wave radar. ACM Trans. Graph., 35(4), July 2016.

[2]Okan K öpüklü, Ahmet Gunduz, Neslihan Kose, and Gerhard Rigoll. Real-time hand gesture detection and classification using convolutional neural networks. arXiv 1901.10323, 2019.

[3]Rully Soelaiman I Wayan Suartika E. P, Arya Yudhi Wijaya. Klasifikasi citra menggunakan convolutional neural network (cnn) pada caltech 101. Jurnal Teknik ITS, 2016.

[4]Jost Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. arXiv 1412.6806, 12 2014.

[5]Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15:1929–1958, 06 2014.

[6]Abien Fred Agarap. Deep learning using rectified linear units (relu). arXiv 1803.08375, 2018.

[7]Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems, 25, 01 2012.

[8]Anthony Gachagan Stephen Marshall Chigozie Nwankpa, Winifred Ijomah. Activation functions: Comparison of trends in practice and research for deep learning. arXiv 1811.03378, 2018.

[9]Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv 1409.1556, 09 2014.

[10]Sun Manli, Zhanjie Song, Xiaoheng Jiang, Jing Pan, and Yanwei Pang. Learning pooling for convolutional neural network. Neurocomputing, 224, 11 2016.