JURNAL SISFOTENIKA
Vol. 13 No. 2, Juli 2023
p-ISSN: 2087-7897 ; e-ISSN : 2460-5344
DOI : 10.30700/jst.v13i2.1447

Analisis Pengaruh Optimizer pada Model CNN untuk Identifikasi Cacat pada Perekat Kemasan

Optimizer Analysis on the CNN Model for Identification Packaging Defects

Richo[1], Ryan Yudha Adhitya[2], Muhammad Khoirul Hasin[3], Mat Syai'in[4], Edy Setiawan[5]
[1,2,3,4,5] Program Studi Teknik Otomasi, Politeknik Perkapalan Negeri Surabaya, Surabaya,
e-mail: *[1]**richo@student.ppns.ac.id,** [2]ryanyudhaadhitya@ppns.ac.id,
[3]khoirul.hasin@ppns.ac.id [4]matt.syaiin@ppns.ac.id [5]edy_setiawan@ppns.ac.id

Abstrak

Industri makanan dan minuman yang mengalami peningkatan pesat salah satunya yakni industri produksi tepung terigu. Namun, dalam proses produksi masih mengalami kendala salah satunya klasifikasi kelayakan kemasan produk tepung terigu yang sesuai standard. Penelitian ini bertujuan untuk mengetahui pengaruh optimizer pada model Convolutional Neural Network dalam mendeteksi hasil kelayakan produk kemasan tepung terigu berdasarkan 2 kondisi yakni cacat atau normal. Area deteksi kemasan dilakukan pada area vertikal kemasan yang ditentukan berdasarkan kerusakan pada area perekat kemasan yang menimbulkan keluarnya bercak tepung pada area tersebut. Proses deteksi menggunakan webcam untuk capture image yang kemudian akan dilakukan ekstraksi fitur, reduksi citra, dan classification berdasarkan nilai probabilitas tertinggi. Pada penelitian ini kami membandingkan optimizer pada model CNN untuk meminimalisir terjadinya overfitting serta menghasilkan akurasi deteksi yang optimal. Optimizer yang dibandingkan yakni optimizer Adadelta, Adagrad, Adam, Adamax, RMSprop, dan SGD. Dataset berjumlah 250 gambar, 125 gambar kelas cacat dan 125 gambar lainnya merupakan kelas normal. Sementara itu, split data pada proses training terbagi atas 90% data training dan 10% data validation. Setelah dilakukan pengujian diperoleh hasil training model CNN terbaik yakni menggunakan optimizer Adam dengan validation accuracy sebesar 92.77% dan akurasi testing mencapai 90%. Peneliti berharap adanya pembaruan optimizer dan arsitektur yang lebih variatif agar akurasi deteksi lebih optimal.

Kata kunci*— CNN, Optimizer, Kemasan, Tepung Terigu, Klasifikasi.*

Abstract

One of the food and beverage industries that experienced rapid growth was the wheat flour production industry. However, in the production process there are still problems, one of which is the classification of the feasibility of standardized wheat flour product packaging. This study aims to determine the effect of the optimizer on the Convolutional Neural Network model in detecting the feasibility results of wheat flour packaged products based on 2 conditions, namely defects or normal. The packaging detection area is carried out in the vertical area of the packaging which is determined based on damage to the packaging adhesive area which causes powdery spots to come out in that area. The detection process uses a webcam to capture an image which will then be performed with feature extraction, image reduction, and classification based on the highest probability value. In this study we compare the optimizer to the CNN

model to minimize overfitting and produce optimal detection accuracy. The optimizers being compared are the Adadelta, Adagrad, Adam, Adamax, RMSprop, and SGD optimizers. The dataset consists of 250 images, 125 images of the disabled class and 125 other images are normal class. Meanwhile, split data in the training process is divided into 90% training data and 10% validation data. After testing, the best CNN model training results were obtained using the Adam optimizer with validation accuracy of 92.77% and testing accuracy reaching 90%. Researchers hope that there will be updates to optimizers and more varied architectures so that detection accuracy is more optimal.

Keywords— *CNN, Optimizer, Packaging, Wheat Flour, Classification.*

1. PENDAHULUAN

Industri makanan dan minuman yang mengalami peningkatan pesat salah satunya yakni industri produksi tepung terigu, hal tersebut tertuang pada peraturan Kementerian Perindustrian RI Nomor 11 tahun 2021 [1]. Namun, berbagai permasalahan pada sektor industri produksi tepung terigu terus bermunculan. Beberapa industri produksi tepung terigu di Indonesia didapati masih melakukan cara manual dalam proses klasifikasi kelayakan kemasan berupa pengamatan dengan mata operator secara langsung. Hal inilah yang memicu terjadinya kelengahan dan kesalahan pengamatan oleh operator dalam hal pengawasan dan pengenalan kondisi kemasan tepung terigu yang cacat. Indikasi cacat kemasan dihasilkan akibat kegagalan proses pengemasan karena ketidaksesuaian *setting* suhu pada alat perekat dan proses pelipatan yang kurang rapat sehingga mengakibatkan area perekat tidak merekat dengan semestinya [2].

Hal tersebut tentunya menyebabkan produk kemasan tepung terigu kurang terjaga kualitasnya dan akan memengaruhi kesehatan seseorang ketika dikonsumsi karena kontaminasi mikroba dan bakteri yang masuk pada area perekat kemasan cacat. Identifikasi cacat produk dideteksi berdasarkan kerusakan atau kebocoran kemasan yang mengakibatkan keluarnya tepung pada area perekat vertikal kemasan. Keluarnya tepung pada area perekat vertikal kemasan diidentifikasi sebagai bercak yang dapat dideteksi dengan *web camera* untuk mengolah dan memroses standarisasi kelayakan kemasan produk tepung terigu berdasarkan 2 kondisi klasifikasi yakni cacat atau normal. Dalam proses deteksi penulis menerapkan metode *Convolutional Neural Network* (CNN) sebagai pemrosesan data *input* dengan mengekstraksi citra, mereduksi, dan mengklasifikasikan citra berdasarkan karakteristik *high probability* [3].

Kemampuan klasifikasi *image* merupakan hal paling penting ketika menyusun model CNN untuk mendapatkan akurasi terbaik [4]. Nilai akurasi yang tinggi umumnya dapat dilakukan dengan menyusun arsitektur yang kompleks, konvergensi parameter, hingga total *epochs* dan data *augmentation* yang bervariasi [5]. Namun, pembelajaran model CNN dalam proses *training* tak akan luput dengan adanya *overfitting* yang dapat melemahkan kemampuan *probability identification model* [6].

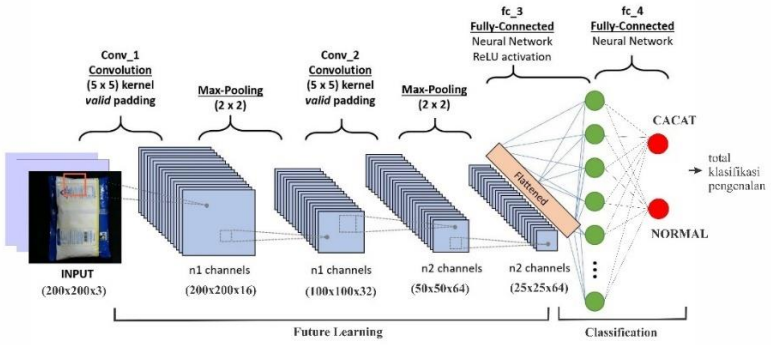
Implementasi metode CNN telah marak diterapkan untuk mendeteksi dan pengenalan objek pada aspek kehidupan. Zacky dkk [7] melakukan penelitian dengan metode CNN untuk klasifikasi kendaraan tanpa awak skala kecil, diperoleh hasil pelatihan model tanpa optimizer yang masih mengalami *overfitting* pada *batch* 50 dan 100. Peneliti lain Billy dkk [8] melakukan penambahan optimizer SGD pada model CNN untuk klasifikasi hasil deteksi beras putih guna mengatasi terjadinya *overfitting* meskipun masih relatif tinggi. Penerapan optimizer SGD pada model CNN dipilih karena memiliki kemampuan untuk melakukan penurunan dengan parameter *Exponentially Weighted Averages* guna memperbarui perubahan gradien yang akan diproses.

Peneliti ini mengimplementasikan optimizer pada *training* model CNN untuk menghasilkan nilai akurasi yang maksimal dalam hal ketepatan deteksi kemasan tepung terigu dengan kategori cacat atau normal. Perbedaan dengan penelitian yang ada yakni peneliti membangun arsitektur model CNN dengan 3 lapis *extraction fitur* serta parameter data generator yang lebih kompleks. Optimizer yang digunakan pada penelitian ini menggunakan perbandingan 6 optimizer diantaranya yakni optimizer Adadelta, Adagrad, Adam, Adamax, RMSprop, dan SGD. Total *data set* yang digunakan berjumlah 250 *image* yang terdiri dari 125 kelas cacat dan 125 gambar merupakan kelas normal dengan ukuran citra pada masing-masing kelas yakni berukuran 200x200 *pixel*. Data *split* yang digunakan terbagi atas 90% data *training* dan 10% merupakan data *validation*. Peneliti menerapkan optimizer pada model CNN untuk melihat perubahan apa yang terjadi dalam klasifikasi pengenalan produk kemasan cacat dan normal berdasarkan parameter pembelajaran model.

2. METODE PENELITIAN

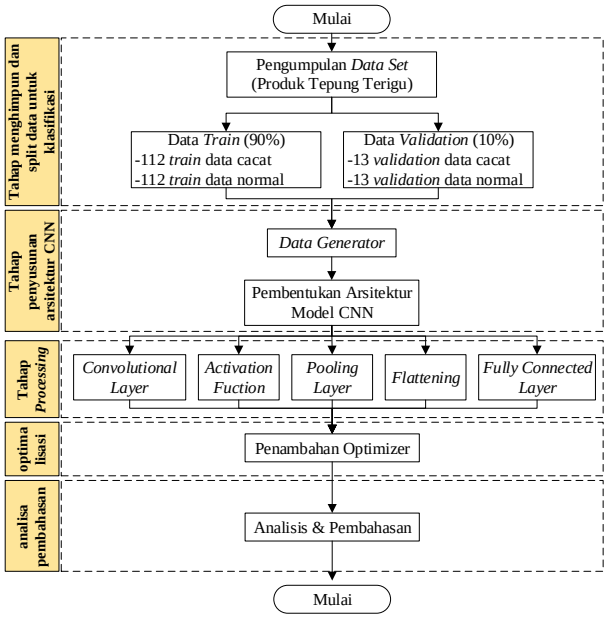
2.1 Metode Convolutional Neural Network

Convolutional Neural Network atau CNN merupakan susunan khusus dari jaringan syaraf tiruan (*neural network*) yang bertujuan untuk melakukan pemrosesan data yang telah terbentuk berdasarkan fisiologi topologi jala atau *grid-like topology* [9]. Secara umum arsitektur CNN merupakan deklarasi antara *layer feature learning* untuk melakukan translasi *input* citra menjadi suatu fitur baru berdasarkan ekstraksi dari ciri citra yang dideteksi, sedangkan *classification* bertujuan untuk klasifikasi *neuron* dalam merespon ketepatan pengenalan [10].



Gambar 1. Struktural *Convolutional Neural Network*

2.2 Alur Penelitian



Gambar 2. Alur Penelitian

Alur penelitian didasarkan atas tahapan awal berupa pengumpulan *data set* dan *split* data untuk menghimpun data dalam proses klasifikasi, selanjutnya dilakukan tahap penyusunan arsitektur CNN, Tahap *processing image* merupakan tahapan dimana data *input* akan diproses berdasarkan konstruksi *layer* dari model yang dibuat, selanjutnya dilakukan perbandingan Optimasi sistem guna menghilangkan *overfitting*, serta dilakukan tahap terakhir yakni analisa dan pembahasan.

2.2.1 Tahap Pengumpulan Data Set

Teknik pengumpulan data dilakukan dengan meng-*capture* produk oleh *Webcam Logitech C310* menggunakan *background* hitam dengan jarak pengambilan yang konstan yakni 25 – 35 cm. Data yang terkumpul kemudian dilakukan proses *labelling* untuk mengidentifikasi data berdasarkan kelas cacat atau normal. Data yang digunakan merupakan *capture* produk khususnya pada area perekat vertikal kemasan tepung terigu dengan total pengambilan *sample* sebanyak 250 data *image* pada tingkat pencahayaan setiap *sample* yakni antara 142 – 158 lux. Produk yang diteliti yakni produk tepung terigu bogasari segitiga biru kemasan 500 gram.

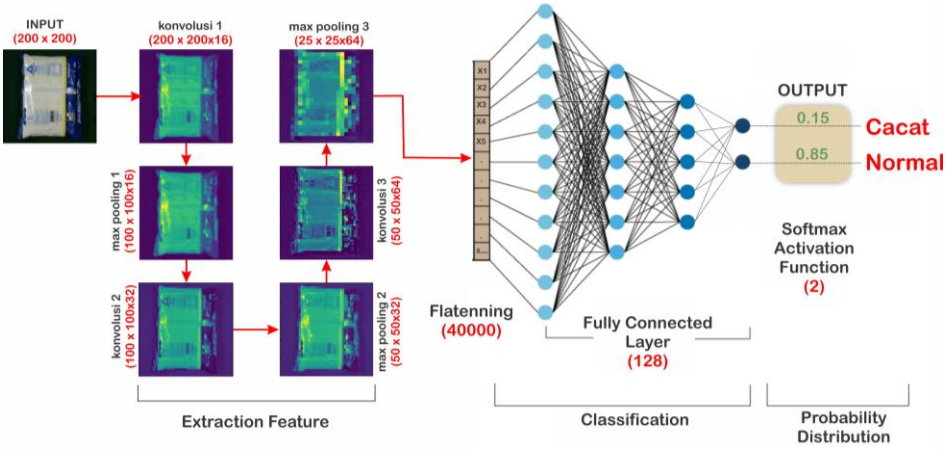
Gambar 3. Hasil *Capture* dan Area Perekat Kemasan yang Dideteksi

2.2.2 Tahap Penyusunan Arsitektur CNN

Penyusunan arsitektur diawali dengan membuat parameter data generator untuk memperkaya pola gambar yang meliputi *rescale*, *rotation image*, *shear range*, *fill mode*, dan *zoom range* [11]. Selanjutnya menyusun model *sequential* berfungsi untuk membentuk pola arsitektur CNN yang diimplementasikan pada penelitian ini.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 200, 200, 16)      448

 max_pooling2d (MaxPooling2D  (None, 100, 100, 16)     0
 )

 conv2d_1 (Conv2D)           (None, 100, 100, 32)      4640

 max_pooling2d_1 (MaxPooling  (None, 50, 50, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 50, 50, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 25, 25, 64)       0
 2D)

 flatten (Flatten)           (None, 40000)             0

 dense (Dense)               (None, 128)               5120128

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 2)                 258

=================================================================
Total params: 5,143,970
Trainable params: 5,143,970
Non-trainable params: 0
_________________________________________________________________
```

Gambar 4. Hasil Parameter Model *Sequential*



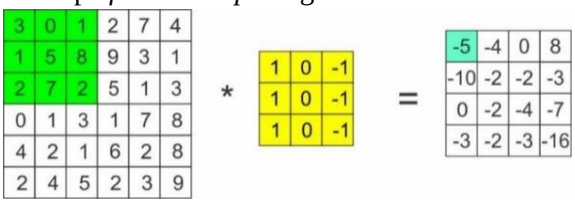
Gambar 5. Hasil Pola Arsitektur CNN

2.2.3 Tahap Processing Data

Tahap *processing* data bertujuan untuk memproses *input* citra dengan beberapa fungsi konvolusi yang meliputi *convolutional layer*, *activation*, *subsampling* atau *pooling layer*, *fully connected layer*, serta *softmax function*.

1. *Convolutional Layer*

   Layer ini digunakan untuk menerima *input* citra yang kemudian akan dilakukan operasi konvolusi terhadap *filters* untuk mengekstraksi fitur yang akan diproses [12]. Operasi pada *layer* ini didapatkan dari total perkalian pada posisi titik yang sama antara *input* dengan *filters* menggunakan operasi *dot* pada setiap pergeseran (*stride*) sehingga menghasilkan *output* berupa *feature map* dengan ukuran dimensional yang lebih kecil.



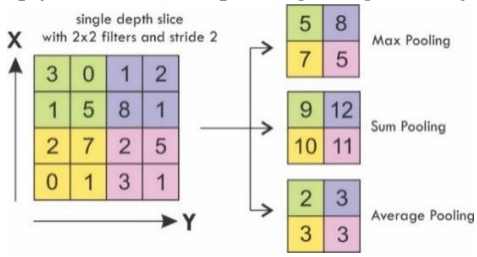
Gambar 6. Operasi *Convolutional*

Perhitungan dari operasi konvolusi yang sapat dituliskan dengan persamaan:

$$TM[s]a,b = \Sigma c\ \Sigma d\ N[a-c, b-d]\ F[m,n]) + Rt \quad (1)$$

Keterangan
TM[s] : *Feature map* dari matriks ke-i
a,b : Koordinat *pixel* pada *input*
c,d : Koordinat *pixel convolutional filters*
N : *Input* matriks
T : Matriks *filters*

2. *Activation Function*
   Fungsi aktivasi digunakan untuk mengaktifkan jaringan *neuron* berdasarkan perambatan balik menggunakan karakteristik yang *continue*, *deferential*, dan *non underload* [13]. *Activation* yang digunakan pada penelitian ini yakni aktivasi ReLu berdasarkan nilai *input* yang terbesar terhadap (0, *max*) yang dapat mempercepat proses pelatihan data.
3. *Pooling Layer*
   Pooling layer umumnya digunakan untuk mereduksi ukuran citra guna mempercepat proses komputasi [14]. *Pooling layer* yang biasanya digunakan yakni *maxpooling* dengan mengambil *range* nilai terbesar di setiap *filters*, *average pooling* merupakan nilai rata-rata dalam setiap *filters*, dan *sumpooling* dengan menjumlah area per *filters*.



Gambar 7. Struktural *Pooling Layer*

4. *Fully Connected Layer*
   Layer ini digunakan untuk menentukan *output* dari *neuron* yang terhubung secara keseluruhan dengan *input* data citra. *Output* citra yang dikeluarkan akan menghasilkan klasifikasi pengenalan terhadap objek yang dideteksi.
5. *Softmax Function*
   Softmax function merupakan *layer* tambahan yang bertujuan untuk meminimalisir terjadinya kerugian *predict* data *input* yang ditambahkan pada lapisan paling akhir dari jaringan *neuron* [15]. *Softmax* berperan untuk memprediksi banyaknya kelas berdasarkan probabilitas yang tertinggi.

2.2.4 Optimalisasi Model dengan Optimizer

Proses Optimasi model merupakan suatu hal penting. Selain untuk meningkatkan nilai akurasi pelatihan model, optimizer umumnya digunakan untuk meminimalisir terjadinya *overfitting* selama proses pelatihan model.

2.2.5 Analisis dan Kesimpulan

Tahap ini merupakan langkah untuk mengevaluasi berdasarkan arsitektur CNN yang dibuat dengan melibatkan optimizer pada konstruksi model CNN. Dalam penelitian ini dilakukan evaluasi untuk melakukan perbandingan akurasi data menggunakan 6 optimizer yakni

Adadelta, Adagrad, Adam, Adamax, RMSprop, dan SGD terhadap pengenalan klasifikasi cacat atau normal pada perekat kemasan tepung terigu.

3. HASIL DAN PEMBAHASAN

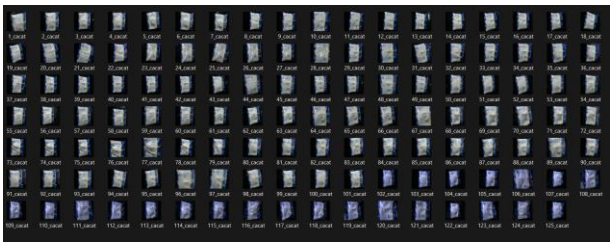
3.1 Pengumpulan Data Set dan Build Model

Pengumpulan *dataset* dilakukan untuk mengetahui kelompok himpunan data *image* yang telah dikelompokkan berdasarkan 2 kelas yakni cacat dan normal. Sedangkan *build* model ditujukan untuk mengetahui arsitektur CNN yang dibuat.

3.1.1 Data Set Penelitian

Dataset diambil dari *capture* produk kemasan tepung terigu menggunakan *webcam* berjumlah 125 data cacat dan 125 data normal dengan ukuran *pixel* sebesar 200x200 *pixel*.

Gambar 8. Himpunan *Dataset* normal



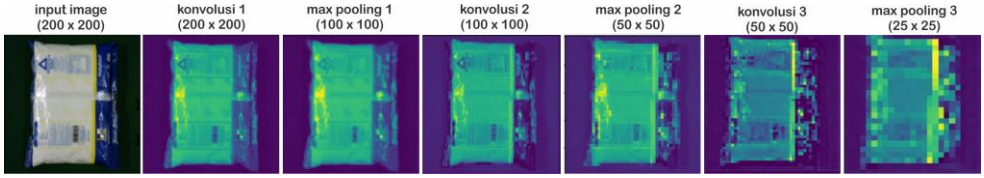
Gambar 9. Himpunan *Dataset* Cacat

3.1.2 Build Model

Pembuatan model arsitektur ditujukan untuk menyusun kesesuaian *layer* yang mempengaruhi ketepatan deteksi. Semakin kompleks susunan *layer* model maka semakin besar kemungkinan model dapat mengenali hasil deteksi dengan tepat. Berikut merupakan arsitektur yang dibangun ketika berlangsungnya proses pembelajaran pada penelitian ini.

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, kernel_size=(3, 3), strides=1, activation ='relu', padding='same', input_shape = (200, 200, 3)),
    tf.keras.layers.MaxPooling2D(pool_size =(2,2)),

    tf.keras.layers.Conv2D(32, (3, 3),strides=1, activation ='relu', padding='same'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3, 3), strides=1, activation ='relu', padding='same'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation = 'relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(2, activation = 'sigmoid')
])
```

Gambar 10. Arsitektur Model CNN



Gambar 11. Hasil Proses *Convolutional* dan *Max Pooling*

3.2 Implementasi Optimizer Pada Arsitektur Model CNN

Implementasi optimizer diletakkan pada model *compile* struktur model CNN dengan fungsi kerugian menggunakan *categorical crossentropy*, laju pembelajaran masing-masing optimizer dengan *learning rate* sebesar 0,01 dan *metrics* menggunakan parameter *accuracy*.

```
#from keras.optimizers import Adam
model.compile(loss = 'categorical_crossentropy',
              #optimizer = 'Adam',
              optimizer =tf.keras.optimizers.Adam(learning_rate=0.01),
              metrics = ['accuracy'])
```

Gambar 12. *Syntax* Implementasi Optimizer Pada Model CNN

Adanya optimizer pada arsitektur CNN memiliki pengaruh besar dalam hal perbaikan nilai *accuracy*, dan meminimalisir *overfitting* selama proses *training* berlangsung. Berikut merupakan perbandingan model dengan optimizer dan tanpa menggunakan optimizer.

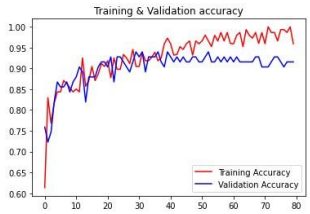
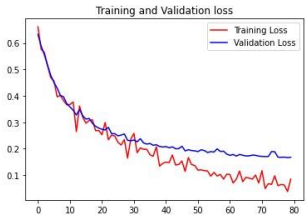
Tabel 1. Perbandingan Model dengan Menggunakan Optimizer

	Training Accuracy	***Training Loss***	***Validation Accuracy***	***Validation Loss***
Tanpa Optimizer	67.53%	21.34%	65.31%	19.58%
Menggunakan Optimizer	95.57%	7.49%	92.84%	12.74%

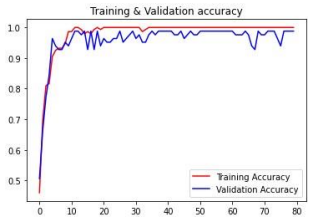
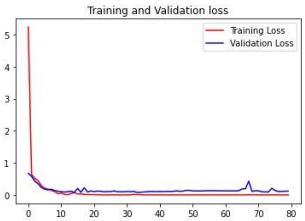
3.3 Hasil Training Model Berdasarkan Optimizer

Pelatihan pembelajaran model dilakukan dengan melakukan perbandingan beberapa optimizer untuk menghindari *overfitting* dan meningkatkan akurasi. Pada pengujiannya total *epochs* yang digunakan berjumlah 80 *epochs* dengan *step per epochs* sebanyak 15 *step*.

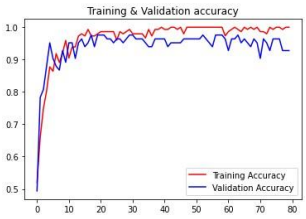
Tabel 2. *Accuracy* dan *Loss Training Validation*

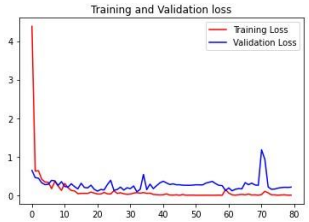
Accuracy	***Loss***
	

Training Model Dengan Optimizer **Adadelta**

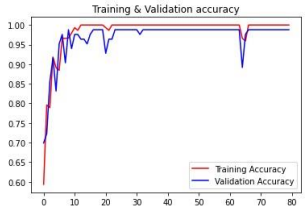
	

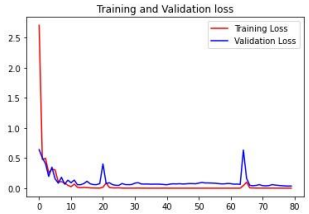
Training Model Dengan Optimizer **Adagrad**



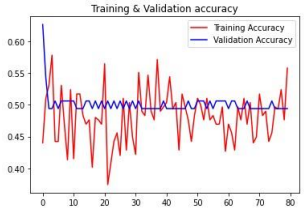


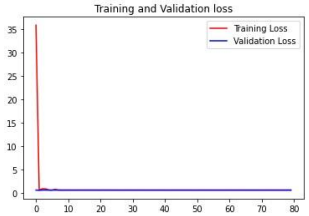
Training Model Dengan Optimizer **Adam**



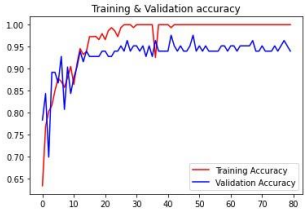


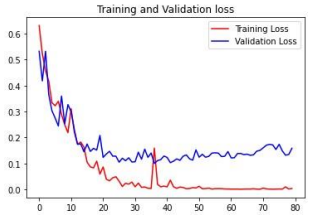
Training Model Dengan Optimizer **Adamax**





Training Model Dengan Optimizer **RMSprop**





Training Model Dengan Optimizer **SGD**

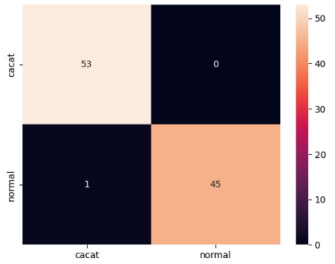
Pada dasarnya nilai *accuracy* yang baik yakni pola grafik yang cenderung naik dan konstan, sedangkan nilai *loss* yang baik yakni dengan pola grafik yang cenderung menurun.

Tabel 3. Hasil Akumulasi *Training*

	Training Accuracy	***Validation Accuracy***	***Training Loss***	***Validation Loss***
Adadelta	95.92 %	91.57 %	8.50 %	16.76 %
Adagrad	100 %	98.80 %	0.0087 %	11.90 %
Adam	100 %	92.77 %	0.55 %	21.13 %
Adamax	100 %	98.80 %	0.0014 %	3.60 %
RMSprop	55.78 %	49.40 %	69.16 %	69.54 %
SGD	100 %	93.98 %	0.33 %	15.83 %

3.4 Hasil Confusion Matriks

Confusion matriks bertujuan untuk menganalisis ketepatan pengelompokan data ketika proses *training* berdasarkan *random state* data yang berjumlah 50 data. Parameter yang disertakan yakni *True Positif* (TP) atau ketepatan deteksi cacat, *True Negatif* (TN) merupakan ketepatan deteksi normal, *False Positif* (FP) atau *error* deteksi cacat, dan *False Negatif* (FN) merupakan *error* deteksi normal. Berikut merupakan hasil parameter *confusion matrix* terbaik.



Gambar 13. Konstruksi Parameter *Confusion Matrix*

Tabel 4. Hasil *Confusion Matrix Masing-masing Optimizer*

Nilai *Confusion*	Optimizer					
	Adadelta	**Adagrad**	**Adam**	**Adamax**	**RMSprop**	**SGD**
TP	47	49	53	50	51	47
TN	42	43	45	43	2	41
FP	6	4	0	3	0	6
FN	3	2	1	2	45	4

Selanjutnya untuk mengetahui parameter lain dari *confusion matrix* dilakukan evaluasi *output* yang meliputi *Accuracy*, *Presisi*, *Recall*, dan F1 *Score*.

1. *Accuracy* merupakan *probability* benar (positif dan negatif) terhadap keseluruhan data yang diujikan.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

2. *Precision* merupakan rasio *probability* positif berbanding dengan hasil prediksi data positif.

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

3. *Recall* merupakan *ratio probability* prediksi benar berbanding pada keseluruhan data yang bernilai positif.

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

4. F1 *score* merupakan nilai akumulasi dari parameter *precision* dan *recall*.

$$\text{F1 } Score = \frac{2\ x\ accuracy\ x\ precision}{accuracy+precision} \quad (5)$$

Berikut merupakan akumulasi perhitungan parameter *confusion* dalam proses *training*, nilai yang diambil didasarkan dengan penjumlahan data cacat dan normal kemudian berbanding dengan total kelas yang digunakan.

$$\text{X } confusion = \frac{\text{Data Cacat+Data Normal}}{\text{Jumlah Kelas (n) yang digunakan}} \quad (6)$$

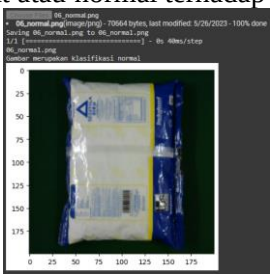
Tabel 5. Parameter *Confusion*

Parameter	Optimizer

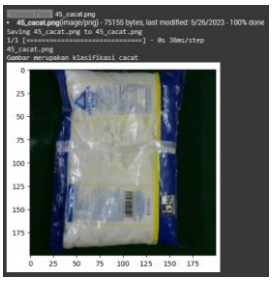
	Adadelta	Adagrad	Adam	Adamax	RMSprop	SGD
Accuracy	0.92	0.99	0.99	0.93	0.49	0.94
Precision	0.92	0.99	0.99	0.93	0.25	0.94
Recall	0.91	0.99	0.99	0.92	0.99	0.94
F1-score	0.91	0.99	0.99	0.92	0.33	0.94

3.5 Testing

Pengujian pada model yang dibuat, dilakukan dengan meng-*upload* gambar uji pada IDE (*Integrasi Development Environment*) selama proses *training* untuk mengklasifikasikan pengenalan deteksi kemasan berupa cacat atau normal terhadap model yang telah dibuat.



Gambar 14. Tampilan Keberhasilan Uji Produk Normal



Gambar 15. Tampilan Keberhasilan Uji Produk Cacat

Pengujian dilakukan sebanyak 30 data yang meliputi 15 *sample* data cacat dan 15 *sample* lainnya merupakan data normal. Sehingga didapatkan hasil pengujian sebagai berikut:

Tabel 6. Hasil *Testing*

	Benar		Salah		Akurasi
	Normal	Cacat	Normal	Cacat	
Adadelta	11	12	4	3	76.67%
Adagrad	13	12	2	3	83.34%
Adam	14	13	1	2	90.00%
Adamax	13	13	2	2	86.67%
RMSprop	9	5	6	10	46.67%
SGD	12	11	3	4	76.67%

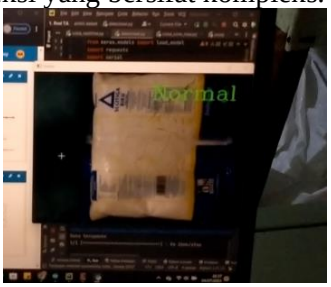
Nilai akurasi dalam proses pengujian atau *testing* diambil berdasarkan penjumlahan antara total denar deteksi normal dan cacat berbanding dengan banyaknya data yang diujikan, sehingga dituliskan dengan persamaan berikut:

$$X\ confusion = \frac{\text{Total Data benar deteksi cacat+Total benar deteksi normal}}{\text{Total keseluruhan data yang diujikan}}\ x\ 100\% \quad (7)$$

3.6 Analisis Keputusan Optimizer Terbaik dan Uji Sistem Deteksi

Adanya implementasi optimizer pada model CNN berpengaruh pada sistem deteksi. Berdasarkan nilai *accuracy* dan *loss training validation*, optimizer dengan konstruksi grafik

terbaik yakni optimizer Adagrad, Adam, dan Adamax. Namun konstruksi grafik yang baik belum menjamin kebaikan dalam ketepatan deteksi. Oleh karenanya, penulis menganalisis lebih lanjut menggunakan *confusion matrix*. Jika ditinjau menurut hasil *confusion matrix*, optimizer yang memiliki ketepatan deteksi paling tinggi pada penelitian ini yakni optimizer Adam dengan total terdeteksi benar sebesar 98 data. Penelitian ini membuktikan proses pembentukan arsitektur dan optimizer sangat mempengaruhi akurasi klasifikasi. Penelitian ini menghasilkan beberapa penemuan penting bahwa optimizer Adam mampu menghasilkan akurasi deteksi yang optimal karena pada strukturnya terdapat skema generik data yang bersifat *adaptif*, laju pembaruan konsisten, dan konvergensi model yang stabil sehingga mampu mempelajari setiap pola *sample* yang terjal dan objek deteksi yang bersifat kompleks.



Gambar 16. Visualisasi Sistem Deteksi dengan Optimizer Adam

4. KESIMPULAN

Berdasarkan tahapan analisis dan pembahasan, secara garis besar sistem deteksi berlangsung pada 2 kelas yakni cacat atau normal. Hasil pada penelitian ini disimpulkan bahwa:

1. Adanya penambahan optimizer sangat berpengaruh terhadap hasil *training*. Berdasarkan tahapan *training* yang telah dilakukan, diperoleh bahwa model dengan penambahan optimizer menghasilkan *training accuracy* yang lebih optimal dibandingkan model tanpa optimizer dengan perbandingan sebesar 95.57% berbanding 67.53%.
2. Berdasarkan hasil keseluruhan parameter yang diujikan pada proses *training*, bahwa optimizer Adam memiliki perolehan parameter yang lebih unggul dibandingkan dengan optimizer lain dengan nilai *validation accuracy* 92.77%, total prediksi pengelompokan himpunan data benar sebanyak 98 data latih, dan parameter *confusion* sebesar 99%.
3. Selanjutnya pada tahapan *testing* atau pengujian, diperoleh bahwa nilai ketepatan deteksi dengan Optimizer Adam memiliki nilai paling tinggi dengan total deteksi cacat dan normal sebanyak 28 data dari 30 data yang diujikan, dan total keberhasilan *testing* dengan optimizer Adam sebesar 90.00%.
4. Dengan demikian, optimizer yang terbaik untuk mengklasifikasikan hasil ketepatan deteksi kemasan tepung terigu pada model CNN yakni optimizer Adam.

5. SARAN

Implementasi optimizer dalam arsitektur model CNN berhasil meminimalisir terjadinya *overfitting* dan meningkatkan nilai *accuracy*. Penelitian ini bertujuan untuk menganalisis optimizer yang cocok pada klasifikasi kemasan tepung terigu berdasarkan kategori cacat atau normal. Namun, pada proses *training* masih terdapat beberapa titik *epochs* yang belum stabil dan cenderung menurun pada ranah *accuracy training validation* karena *step per epochs* yang digunakan masih cenderung kecil sehingga proses pembelajaran model masih belum optimal. Hasil penelitian yang dilakukan masih belum diimplementasikan secara *real-time* menggunakan integrasi sistem mikrokontroler. Oleh karena itu, Peneliti berharap nantinya hasil dari penelitian ini dapat diimplementasikan dengan koneksi integrasi sistem deteksi *real-time*. Di masa mendatang, peneliti juga berharap adanya perbandingan optimizer yang lebih bervariasi dan

diimplementasi pada objek penelitian yang beragam sehingga *output* yang diperoleh dapat bermanfaat dengan aspek pengembangan dan penyempurnaan sistem yang lebih optimal.

DAFTAR PUSTAKA

[1] Kementerian Perindustrian Republik Indonesia, "Lembaga Penilaian Kesesuaian Dalam Rangka Pemberlakuan Dan Pengawasan Standar Nasional Indonesia Tepung Terigu Sebagai Bahan Makanan Secara Wajib," 2021, [Online]. Available: http://pustan.kemenperin.go.id/public/default/file_penunjukan/Permenperin_No._11_Tahun_2021_1 (1).pdf.

[2] Y. Zhao, Y. Zhao, L. Wang, Y. Yang, and Y. Wang, "Femtosecond Laser Processing Assisted SiC High-Temperature Pressure Sensor Fabrication and Performance Test," *Micromachines*, vol. 14, no. 3, p. 587, 2023, doi: 10.3390/mi14030587.

[3] A. A. Santosa, R. Y. N. Fu'adah, and S. Rizal, "Deteksi Penyakit pada Tanaman Padi Menggunakan Pengolahan Citra Digital dengan Metode Convolutional Neural Network," *J. Electr. Syst. Control Eng.*, vol. 6, no. 2, pp. 98–108, 2023, doi: 10.31289/jesce.v6i2.7930.

[4] F. Chan Uswatun, Angkin, "Implementasi Data Augmentation Random Erasing dan GridMask pada CNN untuk Klasifikasi Batik," vol. 13, no. 1, pp. 16–28, 2023.

[5] M. S. Rifki Dita, Ryan Yudha, "MIdentification System of Personal Protective Equipment Using Convolutional Neural Network (CNN) Method," *Int. Symp. Electron. Smart Devices*, pp. 1–6, 2019.

[6] P. N. Dacipta and R. E. Putra, "Sistem Klasifikasi Limbah Menggunakan Metode Convolutional Neural Network (CNN) Pada Webservice Berbasis Framework Flask," *J. Informatics Comput. Sci.*, vol. 3, no. 04, pp. 394–402, 2022, doi: 10.26740/jinacs.v3n04.p394-402.

[7] A. Muhammad Zacky, "Implementasi Algoritma Convolutional Neural Network Pada Kendaraan Tanpa Awak Skala Kecil," vol. 5, pp. 19–26, 2023.

[8] M. E. A.-R. Billy Gunawan, "Klasifikasi Jenis Beras Putih Menggunakan Cnn Residual Network Optimizer SGD," pp. 128–132, 2023.

[9] S. R. Dewi, "Deep Learning Object Detection Pada Video Menggunakan Tensorflow Dan Convolutional Neural Network," pp. 1–60, 2018, [Online]. Available: https://dspace.uii.ac.id/bitstream/handle/123456789/7762/14611242_Syarifah Rosita Dewi_Statistika.pdf?sequence=1.

[10] S. Septhyan, R. Magdalena, and N. K. C. Pratiwi, "Deep Learning Untuk Deteksi Covid-19 , Pneumonia , Dan Tuberculosis Pada Citra Rontgen Dada Menggunakan CNN Dengan Arsitektur Alexnet," vol. 8, no. 6, pp. 2869–2878, 2022.

[11] A. F. Anavyanto, M. Resa, and A. Yudianto, "EfficientNetV2M for Image Classification of Tomato Leaf Deseases," vol. 11, no. 225, pp. 55–76, 2023.

[12] D. M. Agung Wira, Ikhwan, "Klasifikasi Kerusakan Jalan Pada Citra Jalan Raya Pontianak Dan Sekitarnya Dengan Menggunakan Convolutional Neural Network," vol. 11, no. 01, pp. 11–20, 2023.

[13] D. N. Herdianto, "Implementasi Metode CNN Untuk Klasifikasi Objek," *J. Manaj. Inform. Komputerisasi Akunt.*, vol. 7, no. 1, pp. 54–60, 2023.

[14] R. T. Andhika Bagas, Hendry, "Implementasi Model Deep Learning Convolutional Neural Network (CNN) Pada Citra Penyakit Daun Jagung Untuk Klasifikasi Penyakit Tanaman," *J. Pendidik. Teknol. Inf.*, vol. 6, no. 1, pp. 107–116, 2023.

[15] M. H. Sukriyandi and A. Solichin, "Identifikasi Garis Telapak Tangan Dengan Metode MobileNet Convolutional Neural Network Untuk Sistem Presensi Siswa," *Fakt. Exacta*, vol. 16, no. 1, pp. 31–41, 2023, doi: 10.30998/faktorexacta.v16i1.15138.