Vol 3 (2022) No 4

Jitsi Jurnal Ilmiah Teknologi Sistem Informasi

http://jurnal-itsi.org ISSN 2722-4619 (Print) ISSN 2722-4600 (Online)

Human Activity Recognition Berdasarkan Tangkapan Webcam Menggunakan Metode Convolutional Neural Network (CNN) Dengan Arsitektur MobileNet

Fauzan Akmal Hariz[#], Intan Nurma Yulita[#], Ino Suryana[#]

[#] *Computer Science Department, Padjadjaran University Jatinangor, Kab.Sumedang, Indonesia*
E-mail: fauzan18001@mail.unpad.ac.id, intan.nurma@unpad.ac.id, ino.suryana@unpad.ac.id

A B S T R A C T S

Humans cannot be separated from the daily activities that are part of human life. Human Activity Recognition is currently one of the topics that is being studied a lot along with the rapid advances in technology that are developing at this time. Almost all fields are affected by the COVID-19 pandemic, which affects human activities so that it becomes more limited. One of the areas most affected is education, where campuses implement an online learning system, which makes it more difficult for lecturers to supervise online learning and exams because they cannot monitor student activities directly. This research aims to create a model that can identify a person's activity during an online exam based on webcam capture by utilizing a deep learning model with the Convolution Neural Network method using the MobileNetV2 architecture. Hyperparameter tuning was carried out to produce the optimal model, which was carried out on batch sizes of 16, 32, and 64 and dense layers of 1, 3, 5, and 7. The test resulted in an optimal model with hyperparameters in the form of max epochs of 20, early stopping of 10, learning rate of 0.0001, batch size of 16, and dense layers of 5. The model was evaluated using cross validation and confusion matrix, which managed to provide a final F1-score performance of 84.52%.

K A T A K U N C I

Human Activity Recognition, Deep Learning, Convolution Neural Network, MobileNetV2

A B S T R A K

Manusia tidak bisa terlepas dari aktivitas sehari- hari yang mana merupakan bagian dari kehidupan manusia. Human activity recognition atau pengenalan aktivitas manusia saat ini merupakan salah satu topik yang sedang banyak diteliti seiring dengan pesatnya kemajuan di bidang teknologi yang berkembang saat ini. Hampir semua bidang terdampak dari pandemi COVID-19 yang memengaruhi aktivitas manusia sehingga menjadi lebih terbatas. Salah satu bidang yang paling terdampak yaitu pendidikan, di mana kampus menerapkan sistem pembelajaran daring, yang membuat dosen lebih sulit untuk mengawasi pembelajaran maupun ujian yang dilakukan secara daring karena tidak dapat mengawasi aktivitas yang dilakukan mahasiswa secara langsung. Penelitian ini bertujuan untuk membuat model yang dapat mengenali aktivitas seseorang saat ujian daring berdasarkan tangkapan webcam dengan memanfaatkan model deep learning dengan metode Convolution Neural Network (CNN) menggunakan arsitektur MobileNetV2. Pengujian hyperparameter dilakukan untuk menghasilkan model optimal yang dilakukan pada batch size sebesar 16, 32, dan 64 serta dense layer sebanyak 1, 3, 5, dan 7. Pengujian tersebut menghasilkan model optimal dengan hyperparameter berupa max epoch sebanyak 20, early stopping dengan patience sebesar 10, learning rate sebesar 0,0001, batch size sebesar 16, dan dense layer sebanyak 5. Model tersebut dievaluasi menggunakan cross validation dan confusion matrix yang berhasil memberikan performa F1-score akhir sebesar 84,52%.

1. PENDAHULUAN

Pada saat ini banyak aktivitas yang mengalami perubahan dikarenakan kondisi pandemi COVID-19 yang sedang terjadi. Berbagai negara telah berupaya dalam menanggulangi hal tersebut tidak terkecuali Indonesia yang memberlakukan Pembatasan Sosial Berskala Besar (PSBB) yang memengaruhi banyak aktivitas di berbagai bidang salah satunya yaitu pada bidang pendidikan. Pada bidang pendidikan, metode pengajaran yang sebelumnya menggunakan metode tatap muka bergeser menjadi metode pengajaran daring [1]. Dalam proses belajar mengajar tentu dibutuhkan metode untuk menilai capaian seseorang yang berbentuk ujian yang erat kaitannya dengan kecurangan akademik yang sudah menjadi masalah di dunia pendidikan sejak lama, ditambah lagi dengan kondisi pandemi yang semakin membuat aktivitas kecurangan tersebut meningkat [2]. Hal tersebut merupakan salah satu contoh dari aktivitas yang dilakukan manusia, di mana manusia tidak bisa terlepas dari aktivitas sehari-hari yang merupakan bagian dari kehidupan manusia. Human activity recognition atau pengenalan aktivitas manusia merupakan salah satu topik yang sedang banyak diteliti seiring dengan pesatnya kemajuan di bidang teknologi yang berkembang saat ini [3]. Human activity recognition digunakan dalam berbagai aspek kehidupan mulai dari pemantauan orang yang sedang sakit, pemantauan orang dalam rumah yang menerapkan konsep smart home, hingga pemantauan mahasiswa dalam melaksanakan ujian daring.

Berdasarkan hal tersebut, penelitian ini akan membuat model yang dapat mengenali aktivitas manusia berdasarkan tangkapan webcam yang telah direkam mengenai aktivitas yang dilakukan seseorang. Untuk melakukan penelitian tersebut digunakan metode yang cocok yaitu deep learning yang merupakan salah satu teknik pada machine learning untuk melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi. Metode deep learning yang digunakan yaitu Convolution Neural Network (CNN), karena CNN dapat digunakan untuk memecahkan permasalahan dalam melakukan klasifikasi data dengan tingkat akurasi yang tinggi dan diklaim sebagai model terbaik untuk memecahkan permasalahan object detection dan object recognition [4]. Arsitektur CNN yang digunakan pada penelitian ini adalah MobileNetV2, perbedaan antara arsitektur MobileNetV2 dan arsitektur CNN pada umumnya adalah penggunaan convolution layer dengan ketebalan filter yang sesuai dengan ketebalan dari input image. MobileNetV2 membagi konvolusi menjadi depthwise convolution dan pointwise convolution yang memungkinkan pelatihan menjadi lebih cepat dan akurasi yang lebih baik [5].

2. METODOLOGI PENELITIAN

Dalam melakukan proses penelitian ini, digunakan beberapa metode yang digunakan, metode tersebut di antaranya, yaitu:

A. Human Activity Recognition

Human activity recognition atau dalam bahasa Indonesia disebut dengan pengenalan aktivitas manusia merupakan salah satu topik yang sedang banyak diteliti oleh peneliti karena implementasinya yang cukup luas [6]. Tujuan utama dari human activity recognition adalah mengenali aktivitas- aktivitas yang dilakukan manusia sehinggadapat menghasilkan peringatan secara dini atau membantu dalam membuat keputusan tertentu [7].

Saat ini, metode human activity recognition dapat dibedakan menjadi tiga kategori berdasarkan tipe alat akuisisi data yang digunakan yakni computer vision, standalone sensor, dan smart device [8]. Contoh alat yang termasuk smart device adalah smartwatch, smartphone, dan smart glasses. Human activity recognition dengan computer vision mengandalkan sistem eksternal seperti sensor optik yang menangkap gambar atau video untuk diolah lebih lanjut dalam mengenali aktivitas manusia [9].

B. Citra Digital

Citra adalah representasi, kemiripan atau imitasi dari suatu objek atau benda. Citra dinyatakan sebagai suatu fungsi kontinu dari intensitas cahaya pada bidang dua dimensi. Citra dibedakan menjadi dua yaitu citra kontinu yang diperoleh dari sinyal analog (mata manusia dan kamera analog) dan citra diskrit (digital) yang dihasilkan melalui proses digitalisasi terhadap citra kontinu.

Terdapat pengolahan citra digital yang merupakan sebuah disiplin ilmu yang mempelajari hal yang berkaitan dengan perbaikan kualitas citra, augmentasi citra, melakukan pemilihan ciri sebuah citra, melakukan proses penarikan informasi atau deskripsi objek atau pengenalan objek yang terkandung pada citra, dan melakukan reduksi data untuk tujuan penyimpanan data [10].

C. Deep Learning

Deep learning atau pembelajaran mendalam adalah salah satu bidang machine learning yang memanfaatkan banyak layer pengolahan informasi nonlinier untuk melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi [11]. Deep learning memanfaatkan jaringan saraf tiruan atau biasa disebut artificial neural network, disebut sebagai deep karena banyaknya hidden layer maupun neuron didalamnya bisa banyak sekali. Algoritma deep

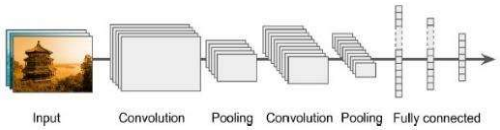
learning dilatih untuk mengidentifikasi pola dan mengklasifikasikan berbagai jenis informasi untuk memberikan output yang diinginkan ketika menerima input baru.

Perbedaan utama dari machine learning dan deep learning yaitu di machine learning, feature extraction perlu dirancang secara manual, yang mana memakan banyak sekali waktu dan tenaga. Sedangkan deep learning akan secara otomatis mengekstrak features untuk klasifikasi. Namun di sisi lain, deep learning menuntut sejumlah data yang besar untuk melatih algoritmanya.

D. Convolutional Neural Network

Convolutional Neural Network (CNN) merupakan salah satu algoritma deep learning yang dirancang untuk mengolah data dalam bentuk data dua dimensi, misalnya gambar atau suara [12]. CNN digunakan untuk mengklasifikasikan data yang berlabel dengan menggunakan metode supervised learning, di mana terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini adalah mengelompokan suatu data ke data yang sudah ada. CNN sering digunakan untuk mengenali benda atau pemandangan, melakukan deteksi, segmentasi objek dan klasifikasi citra [13].

CNN terinspirasi dari bagaimana cara manusia bagaimana memproses dan menghasilkan suatu persepsi secara visual sehingga dapat digunakan untuk mengenali atau mendeteksi suatu objek yang terdapat pada citra digital [14]. CNN adalah model yang dapat digunakan dalam human activity recognition yang digambarkan melalui teknik jaringan saraf yang sangat kuat untuk memodelkan fitur secara efektif [15]. Terdapat tiga layer atau lapisan pada CNN, yaitu convolutional layer, pooling layer, dan fully connected layer yang memiliki variasi yang sangat banyak, saat layer-layer tersebut ditumpuk akan menghasilkan sebuah CNN [16]. Pada Gambar 1 merupakan contoh proses CNN.



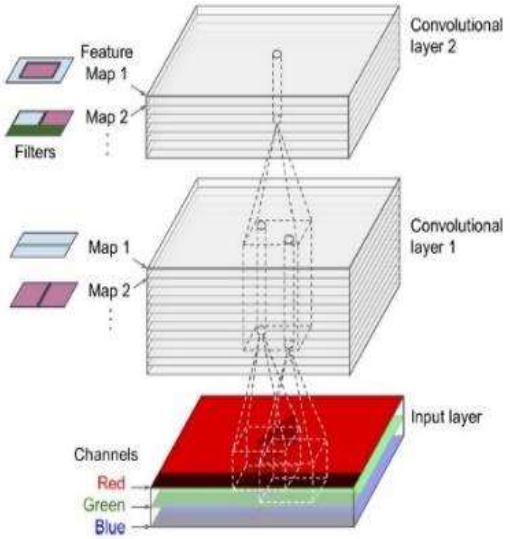
GAMBAR 1. Proses pada Convolutional Neural Network [12]

1. Convolution Layer

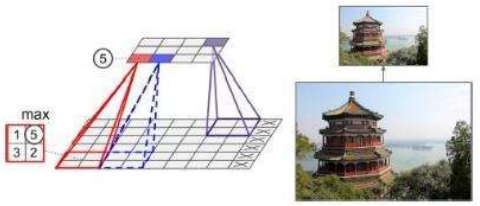
Convolution layer merupakan lapisan pertama pada CNN yang akan melakukan konvolusi citra masukan dengan filter yang telah didefinisikan tanpa merusak struktur citra awal. Fungsi convolution layer yaitu *mengambil* fitur pada citra yang akan digunakan untuk melatih model [17]. Ilustrasi convolutional layer dapat dilihat pada Gambar 2.

2. Pooling Layer

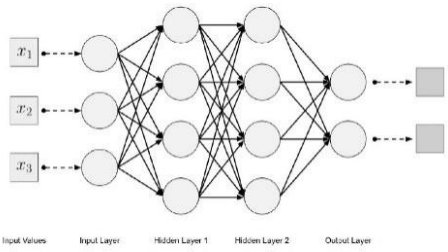
Pooling layer merupakan proses mengurangi jumlah parameter dan jumlah perhitungan pada jaringan, serta melakukan pencegahan overfitting pada citra [18]. Layer ini terbagi menjadi average pooling dan max pooling. Average *pooling* yaitu mengambil nilai rata-rata dari area yang dipilih, sedangkan max pooling yaitu mengambil nilai terbesar dari area yang dipilih. Ilustrasi pooling layer dapat dilihat pada Gambar 3



GAMBAR 2. Ilustrasi Convolutional Layer [12]



GAMBAR 3. Ilustrasi Pooling Layer [12]



GAMBAR 4. Ilustrasi Fully Connected Layer [17]

3. Fully Connected Layer

Fully connected layer merupakan lapisan di mana dimensi seluruh data akan diubah menjadi satu dimensi, proses perubahan ukuran dimensi disebut *flatten*. Lapisan ini memiliki *node* yang saling berhubungan, memiliki bobot, dan memiliki fungsi aktivasi. Lapisan ini mengeluarkan sebuah prediksi berdasarkan data masukan [19]. Ilustrasi *fully connected layer* dapat dilihat pada Gambar 4.

E. MobileNetV2

MobileNetV2 adalah salah satu arsitektur CNN untuk mendeteksi citra yang merupakan model pengembangan dari MobileNetV1, dimana MobileNetV2 menggunakan konvolusi yang mendalam dan terarah seperti MobileNetV1 [18]. MobileNetV2 menunjukan hasil akurasi yang lebih baik dibandingkan MobileNetV1 dengan jumlah parameter yang lebih sedikit. MobileNetV2 menambahkan dua fitur baru yaitu linear bottlenecks dan shortcut connections between bottlenecks [19].

Perbedaan mendasar antara arsitektur MobileNet dan arsitektur CNN pada umumnya adalah penggunaan convolution layer dengan ketebalan filter yang sesuai dengan ketebalan dari input image. MobileNet membagi konvolusi menjadi depthwise convolution dan pointwise convolution [5]. Pada bagian bottleneck, terdapat input dan output antara model sedangkan lapisan bagian dalam mengenkapsulasi kemampuan model untuk mengubah input dari konsep tingkat yang lebih rendah contohnya seperti piksel ke deskriptor tingkat yang lebih tinggi. Sehingga, shortcut antar bottlenecks memungkinkan pelatihan yang lebih cepat dan akurasi yang lebih baik [20]. Tabel 1 menunjukan arsitektur MobileNetV2

TABEL 1. Arsitektur MobileNetV2 [20]

Input	Operator	*t*	*c*	*n*	*s*
224^2 x 3	conv2d	-	32	1	2
112^2 x 32	bottleneck	1	16	1	1
112^2 x 16	bottleneck	6	24	2	2
56^2 x 24	bottleneck	6	32	3	2
28^2 x 32	bottleneck	6	64	4	2
14^2 x 64	bottleneck	6	96	3	1
14^2 x 96	bottleneck	6	160	3	2
7^2 x 160	bottleneck	6	320	1	1
7^2 x 320	conv2d 1x1	-	1280	1	1
7^2 x 1280	avgpool 7x7	-	-	1	-
1 x 1 x 1280	conv2d 1x1	-	k	-	

F. Hyperparameter

Hyperparameter adalah properti dari algoritma pembelajaran yang biasanya memiliki nilai numerik yang mempengaruhi cara kerja algoritma. Nilai-nilai itu tidak dipelajari oleh algoritma itu sendiri dari data, tetapi harus ditetapkan sebelum menjalankan algoritma [21]. Penyetelan hyperparameter (hyperparameter tuning) adalah proses yang dilakukan untuk memilih hyperparameter sehingga model dapat memberikan hasil dan waktu yang optimal [22].

Penyetelan hyperparameter berfokus untuk memastikan bahwa model tidak mengalami underfitting maupun overfitting terhadap dataset pelatihan, sambil mempelajari struktur data secepat mungkin [17]. Terdapat beberapa parameter yang dapat dilakukan hyperparameter tuning, berikut adalah beberapa hyperparameter yang digunakan pada penelitian ini.

1) Batch Size

Batch size adalah banyaknya data sampel yang digunakan antara pembaruan bobot model untuk satu kali batch. Banyaknya dataset yang digunakan pada proses pelatihan dapat mempengaruhi pemilihan dari batch size. Batch size atau ukuran batch dapat berdampak secara signifikan pada performa model dan waktu pelatihan model [23]. Ukuran batch yang lebih besar dapat menghasilkan konvergensi model yang lebih cepat, tetapi lebih sedikit set bobot akhir yang optimal. Apabila batch size terlalu kecil, algoritma dapat mengabaikan jumlah variasi sebenarnya dalam distribusi yang dijadikan sampel, sehingga dapat menghasilkan proses pelatihan yang sangat noise [24]. Batch size pada umumnya menggunakan nilai pangkat 2 dari 32 hingga 256 atau lebih. Hal tersebut dikarenakan akses memori dan desain perangkat keras lebih dioptimalkan untuk beroperasi pada array dengan dimensi yang berpangkat dua, dibandingkan dengan ukuran lainnya [17].

2) Dense Layer

Dense layer merupakan model tradisional neural network yang berfungsi untuk melakukan klasifikasi sesuai dengan class pada output. Banyaknya dense layer menentukan seberapa kompleks model yang dibangun, semakin banyak dense layer yang digunakan, berarti model akan semakin kompleks serta memperbesar learning capacity yang artinya model dapat mempelajari data yang lebih kompleks [25]. Terlalu sedikit dense layer dapat membuat model underfitting karena tidak sanggup untuk mempelajari data latih. Sedangkan apabila dense layer terlalu banyak dapat membuat model overfitting karena model mempelajari data latih terlalu baik dan mendetail sehingga cenderung menghafal data termasuk noise yang seharusnya tidak dipelajari. Selain itu, menambah banyaknya dense layer berarti akan meningkatkan waktu pelatihan serta memory cost menjadi lebih besar karena model mempelajari lebih banyak parameter pada jaringan.

G. Imbalance Class

Imbalance class merupakan kondisi yang terjadi ketika satu atau lebih kelas memiliki proporsi yang sangat rendah dalam data pelatihan dibandingkan dengan kelas lainnya [26]. Hal ini dapat menyebabkan model belajar lebih banyak data mayoritas dibandingkan dengan yang minoritas sehingga dapat menghasilkan performa model yang tidak sama bagus untuk setiap kelasnya

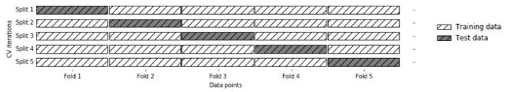
Matriks *F1-score* digunakan daripada akurasi dalam kasus *imbalance class*. Hal ini dilakukan karena nilai *F1-score* menggabungkan matriks *recall* dan *precision* sehingga perhitungannya lebih adil. Selain itu, terdapat beberapa cara untuk mengatasi *class* imbalance yang secara garis besar terbagi menjadi dua, yaitu *undersampling* dan *oversampling*. *Oversampling* dilakukan dengan mereplikasi data pada kelas minoritas, sedangkan *downsampling* berarti menghapus data pada kelas mayoritas secara acak.

H. Mekanisme Evaluasi Model

Mekanisme evaluasi model pada penelitian ini terdiri dari dua proses yaitu *cross validation* dan *confusion matrix*. Masing-masing dari proses tersebut akan dijelaskan sebagai berikut.

1) Cross Validation

Cross validation adalah sebuah proses untuk melakukan validasi data yang mana pasangan data latih dan data validasi dibagi secara terdistribusi untuk suatu dataset [27]. Salah satu teknik *cross validation* yang paling umum digunakan adalah k-fold *cross validation*, di mana k adalah banyaknya iterasi yang dilakukan atau disebut juga dengan *split* yang biasanya bernilai 5 atau 10 [25]. Pada Gambar 5 menunjukan ilustrasi dari *k-fold cross validation* dengan nilai k=5



GAMBAR 5. Data Split pada 5-Fold Cross Validation [25]

2) Confusion Matrix

Confusion matrix atau sering juga disebut *error matrix* adalah tabel yang digunakan untuk melakukan evaluasi model yang digunakan pada *machine learning* [21]. Model *confusion matrix* untuk kasus klasifikasi dua kelas (*binary classification*) dapat dilihat pada Gambar 6.

	P' (Predicted)	N' (Predicted)
P (Actual)	True Positive	False Negative
N (Actual)	False Positive	True Negative

GAMBAR 6. Model Confusion Matrix [17]

Berdasarkan Gambar 6 hasil evaluasi dari model confusion matrix dapat dijabarkan sebagai berikut:

1. *True Positive* (TP), banyaknya data label positif dan diprediksi dengan benar sebagai data positif.
2. *True Negative* (TN), banyaknya data label negatif dan diprediksi dengan benar sebagai data negatif.
3. *False Positive* (FP) , banyaknya data label negatif, namun diprediksi sebagai data positif.
4. *False Negative* (FN), banyaknya data label positif, namun diprediksi sebagai data negatif.

Berdasarkan kombinasi dari nilai yang ada pada tabel *confusion matrix*, dapat dihitung beberapa nilai evaluasi model (*performance metrics*) yang berbeda seperti *precision, recall,* dan *F1-score* [17].

a) Precision

Precision atau *positive predictive value* adalah rasio sampel yang benar diprediksi sebagai positif terhadap *seluruh* sampel yang diprediksi positif. *Precision* dapat dihitung menggunakan persamaan 1

$$Precision = \frac{TP}{TP+TP} \quad (1)$$

b) Recall

Recall yang juga dikenal dengan *sensitivity* atau *true positive rate* adalah rasio sampel yang benar diprediksi sebagai positif terhadap seluruh sampel positif. *Recall* dapat dihitung menggunakan persamaan 2

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

c) F1-score

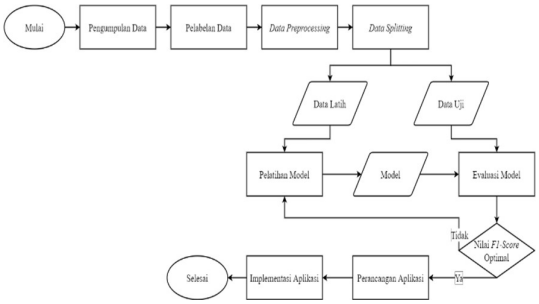
F1-score adalah harmonic mean atau rasio perbandingan rata-rata dari nilai precision dan recall yang dibobotkan. F1- score memperhitungkan nilai precision dan recall sekaligus, sehingga dapat mengukur performa model lebih akurat untuk kasus imbalance class dibandingkan dengan accuracy [25]. Nilai F1-score dapat dihitung menggunakan persamaan 3.

$$F1\ Score = \frac{2TP}{2TP+FN+FN} \quad (3)$$

3. METODOLOGI

Pada penelitian ini, metodologi penelitian yang digunakan terdiri dari tahapan penelitian seperti pengumpulan data, pelabelan data, data preprocessing, data splitting, pembuatan model, evaluasi model, dan analisis kebutuhan.

A. Tahapan Penelitian



GAMBAR 7. Diagram Alur Tahapan Penelitian

Tahapan penelitian bertujuan menggambarkan kegiatan yang akan dilaksanakan selama penelitian. Kegiatan yang akan dilaksanakan selama penelitian ini dilakukan dengan tahapan-tahapan penelitian yang meliputi pengumpulan data aktivitas ujian daring berupa rekaman video yang diekstrak menjadi frame sampai pengaplikasian model. Gambar 7 merupakan diagram alur tahapan penelitian yang dilakukan

1) Pengumpulan Data

Data yang digunakan dalam penelitian ini berupa data video yang merupakan rekaman dari webcam sejumlah orang yang melaksanakan ujian daring. Dataset dikumpulkan dari web Computer Vision Lab Michigan State University yang dapat diakses pada tautan berikut: http://cvlab.cse.msu.edu/ project-OEP.html [28]. Di dalam dataset terdapat 24 subject yang merupakan orang yang mengikuti ujian daring, data terbagi menjadi 15 subject yang merupakan aktor yang berperan sebagai mahasiswa yang mengikuti ujian daring dan 9 subject yang merupakan mahasiswa asli yang mengikuti ujian daring. Dalam masing- masing subject terdapat video bertipe file .avi yang merupakan rekaman aktivitas ujian daring dan terdapat ground truth bertipe file .txt yang merupakan data yang menjelaskan aktivitas yang terjadi di dalam video. Gambar 8 menunjukan contoh salah satu data subject berisi video dan ground truth.

Aktivitas ujian daring yang ada di dalam video yang digunakan pada penelitian ini yaitu aktivitas tidak mencontek atau tidak curang (*No Cheat*), membaca teks (*Read Text*), bertanya ke teman (*Ask Friend*), dan menelpon teman (*Call Friend*). Pada Gambar 9 menunjukan contoh frame aktivitas ujian daring yang digunakan pada penelitian.

GAMBAR 8. Diagram Alur Tahapan Penelitian

GAMBAR 9. Diagram Alur Tahapan Penelitian

2) Pelabelan Data

Setelah data terkumpul, data berupa video pada setiap *subject* diekstrak menjadi *frame* dan dikelompokan sesuai dengan kondisi (*class*) yang sesuai dengan *ground truth* pada masing-masing video. Gambar 10 merupakan contoh hasil *frame* bertipe *file* .jpg yang didapat dari hasil ekstrak video bertipe *file* .avi.

Data *frame* bertipe *file* .jpg tersebut lalu akan dilakukan seleksi data untuk memastikan masing-masing *frame* telah sesuai dengan *class*-nya masing-masing lalu dilakukan data *preprosessing* berupa *data augmentation* agar menghasilkan data yang lebih banyak dan lebih bervariasi.

3) Data Preprocessing

Data preprocessing adalah proses melakukan penyetaraan ukuran data (*resizing*) dan *data augmentation* untuk seluruh data. Fungsi dari *data preprocessing* untuk mengatur data agar sesuai dengan format yang diinginkan, menghasilkan data yang lebih banyak dan lebih bervariasi yang dapat memperkuat hasil pada model. Teknik *data augmentation* juga dilakukan sebagai metode *oversampling* untuk mengurangi *imbalance class* pada. Berikut ini beberapa teknik augmentasi yang dilakukan pada penelitian ini.

a. Random Rotation

Random rotation adalah salah satu teknik augmentasi yang memungkinkan model menjadi invarian terhadap orientasi objek. *Data augmentation* berupa *rotation range* memungkinkan untuk memutar *frame* secara acak melalui derajat apa pun antara 0 sampai 360 derajat. Gambar 11 merupakan contoh *frame* yang telah dilakukan

data augmentation berupa *random rotation*.

GAMBAR 10. Frame yang Didapat Dari Hasil Ekstrak Video

GAMBAR 11. Hasil Data Augmentation Menggunakan Random Rotation

b. Random Shift

Random shift adalah salah satu teknik augmentasi yang digunakan untuk mengatasi masalah yang mungkin saja objek tidak selalu *berada* di tengah *frame*. Gambar 12 merupakan contoh *frame* yang telah dilakukan *data augmentation* berupa *random shift*.

c. Random Flip

Random flip adalah salah satu teknik augmentasi yang digunakan untuk membalik frame. Gambar 13 merupakan contoh frame yang telah dilakukan data augmentation berupa random flip.

d. Random Zoom

Random zoom adalah salah satu teknik augmentasi untuk memperbesar frame atau memperkecil frame secara acak. Gambar 14 merupakan contoh frame yang telah dilakukan data augmentation berupa random zoom

GAMBAR 13. Hasil Data Augmentation Menggunakan Random Flip

GAMBAR 12. Hasil Data Augmentation Menggunakan Random Shift

GAMBAR 14. Hasil Data Augmentation Menggunakan Random Zoom

4) Data Splitting

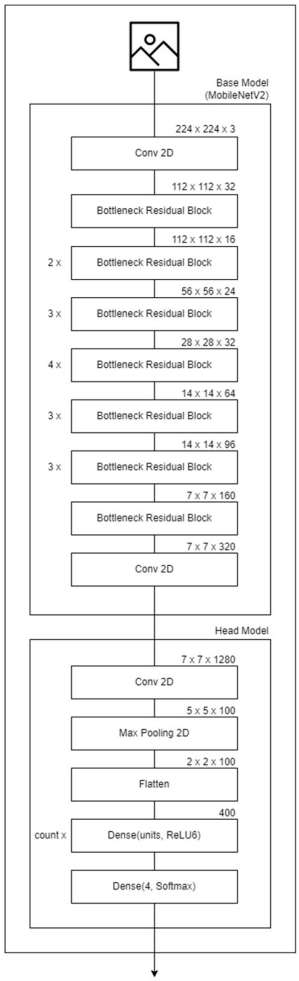
Data splitting merupakan proses membagi data ke dalam beberapa kelompok. Pada tahap ini data frame yang telah dikumpulkan dibagi menjadi data latih dan data uji. Data latih diambil dari frame subject yang merupakan 14 subject aktor yang berperan sebagai mahasiswa yang mengikuti ujian daring yang berjumlah 9000 frame dan untuk data uji diambil dari frame subject yang merupakan 9 subject mahasiswa asli dan 1 subject aktor yang berperan sebagai mahasiswa yang mengikuti ujian daring yang berjumlah 7000 frame. Data latih selanjutnya dibagi menjadi data latih itu sendiri dan data validasi yang digunakan untuk validasi dengan mekanisme 5 stratified k-fold cross validation.

5) Pembuatan Model

Pada penelitian ini pembuatan model dari algoritma CNN menggunakan arsitektur MobileNetV2. MobileNetV2 adalah sebuah model pengembangan arsitektur model CNN dengan jumlah parameter yang lebih sedikit dibanding dengan arsitektur CNN lainnya, namun hanya mengurangi akurasi dengan angka yang kecil. Pada pembuatan model CNN terdapat langkah-langkah yang harus dilewati untuk membangun model yang dapat berfungsi dengan baik. Setiap langkah di CNN terbagi menjadi beberapa layer yang memiliki fungsi untuk memperbesar nilai performa model. Pembuatan model CNN menggunakan arsitektur MobileNetV2 dilakukan dengan cara melakukan pelatihan pada data yang sudah disiapkan. Pada proses ini mesin akan mempelajari data frame yang sudah melawati proses sebelumnya sehinga model dapat memprediksi data frame sesuai dengan class-nya

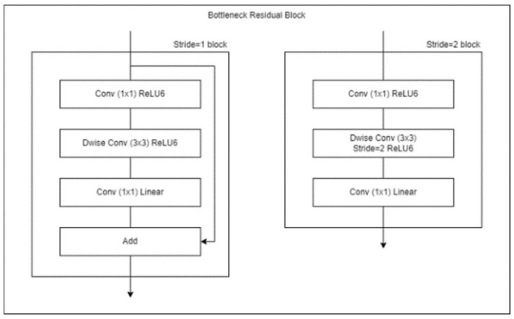
Pada Gambar 15 dapat dilihat model CNN menggunakan arsitektur MobileNetV2 yang dimodifikasi pada bagian head model yang digunakan. Pada arsitektur MobileNetV2 di atas terdapat komponen utama yaitu bottleneck residual block yang dapat dilihat pada Gambar 16.

Pada dasarnya semakin banyak layer yang digunakan dalam model, semakin baik juga nilai akurasi yang akan dihasilkan. Namun dalam penentuan layer ada beberapa kondisi yang perlu diperhatikan saat menentukan banyaknya layer. Jika menggunakan layer yang sedikit maka dapat mengurangi nilai performa bahkan dapat masuk ke dalam kondisi underfitting, tetapi jika menggunakan layer yang berlebih maka dapat masuk ke dalam kondisi yang disebut dengan overfitting. Overfitting adalah kondisi di mana data yang dilatih masuk ke dalam kondisi terbaik, yang membuat saat dilakukan pengujian akan mengurangi nilai performa.



GAMBAR 15. Arsitektur MobileNetV2 yang Digunakan

Oleh karena itu perlu dilakukan percobaan untuk menentukan banyaknya layer yang digunakan disertai memperhatikan dataset beserta hyperparameter. Setelah mendefinisikan arsitektur model, selanjutnya adalah melakukan kompilasi model dengan konfigurasi yang ditentukan sebelum melakukan pelatihan. Konfigurasi yang digunakan pada penelitian di antaranya adalah fungsi loss berupa categorical cross-entropy, optimizer Adam, serta matriks yang digunakan adalah F1-score. Selanjutnya adalah melakukan fitting model, yaitu melatih trainable parameter dari model yang telah didefinisikan dan dikompilasi pada tahap sebelumnya. Pada penelitian ini proses pelatihan model dilakukan pada data latih dengan melakukan uji coba pada hyperparameter yang ditentukan dengan max epoch sebanyak 20serta menggunakan early stopping untuk mencegah terjadinya overfitting pada model. Model kemudian disimpan dengan memanfaatkan pickle pada Python dalam format .H5 sehingga dapat digunakan kembali untuk melakukan prediksi aktivitas manusia.



GAMBAR 16. Bottleneck Residual Block pada Arsitektur MobileNetV2

Pada penelitian ini hyperparameter yang akan dicari dan diuji nilai optimalnya adalah banyaknya dense layer dan batch size. Pemilihan hyperparameter yang akan diuji dilakukan dengan menggunakan studi literatur dari penelitian-penelitian terdahulu. Nilai hyperparameter yang akan diuji pada penelitian ini dapat dilihat pada Tabel.2

TABEL 2. Nilai Pengujian Hyperparameter]

Hyperparameter	**Nilai**
Optimizer	Adam
Activation	Softmax
Max Epoch	20
Learning Rate	0,0001
Batch Size	[16; 32; 64]
Dense Layer	[1; 3; 5; 7]

Berdasarkan Tabel 3.1 hyperparameter yang dibuat konstan adalah optimizer, activation, max epoch, early stopping, dan learning rate. Optimizer yang digunakan adalah Adam yang merupakan algoritma optimasi untuk pelatihan model dalam deep learning yang dapat menangani gradien yang menyebar dan memiliki noise [29]. Keuntungan utama menggunakan Adam adalah efisien secara komputasi, persyaratan memori yang kecil, dan sesuai untuk gradien yang dengan noise yang tinggi. Activation yang digunakan adalah Softmax yang akan menghasilkan output dengan probabilitas pada rentang 0 dan 1 dengan jumlah seluruh probabilitas tersebut adalah 1. Softmax menjadi fungsi aktivasi yang cocok digunakan untuk task berupa klasifikasi multi class dengan hasil prediksi yang dikeluarkan adalah hasil dengan probabilitas tertinggi. Epoch yang digunakan diatur ke nilai yang tinggi yaitu sebesar 20, tanpa perlu dicari nilai optimal secara manual karena model akan berhenti secara otomatis ketika indikasi overfitting terlihat karena menggunakan early stopping. Early stopping yang digunakan pada penelitian ini yaitu patience sebesar 10.

Learning rate yang digunakan sebesar 0,0001 dapat memungkinkan model lebih optimal dibandingkan dengan learning rate yang lebih besar maupun yang lebih kecil.

Teknik yang digunakan dalam mencari *hyperparameter* yang optimal adalah dengan mencoba setiap kombinasi dari *hyperparameter* yang telah ditentukan lalu mengambil kombinasi yang menghasilkan nilai *F1-score* yang paling baik. Dengan mencoba setiap kombinasi, semua *hyperparameter* diberi kesempatan yang sama untuk diuji sehingga terhindar dari masalah maksimum lokal.

6) Evaluasi Model

Pada tahap ini dilakukan evaluasi dari pengujian model dari data pengujian yang sudah dilakukan sebelumnya. Hasil pelatihan dapat dibuktikan apakah sudah dapat dipelajari oleh mesin dengan baik atau belum, cara membuktikannya dengan menggunakan data uji. Jika dilakukan pengujian dengan hasil yang sesuai maka proses pelatihan bisa disimpulkan sudah baik, namun jika pada proses pengujian belum dapat memberikan hasil yang sesuai maka pada proses pelatihan masih perlu diperbaiki hingga mesin dapat mempelajari dengan baik. Hasil pengujian dievaluasi dengan tujuan memastikan model berjalan dengan baik. Evaluasi model dilakukan menggunakan nilai *confusion matrix*. Pada tahap ini dapat dihitung nilai *F1-score* dari model yang telah dibuat berdasarkan nilai *confusion matrix* yang diperoleh menggunakan persamaan 3. Matriks yang akan digunakan pada penelitian ini adalah *F1-score* karena data yang telah dikumpulkan mengalami permasalahan *imbalance class* yang mana terdapat perbedaan banyaknya data pada setiap kelas.

B. Analisis Kebutuhan

Analisis kebutuhan pada penelitian ini terdiri dari kebutuhan perangkat lunak (*software*) dan kebutuhan perangkat keras (*hardware*).

1) Perangkat Keras

Dalam melakukan penelitian ini dibutuhkan perangkat keras yaitu sebuah laptop dengan spesifikasi tertentu. Spesifikasi di bawah ini merupakan spesifikasi yang digunakan pada penelitian ini: Manufaktur: ASUS X555Q, *Processor*: AMD A12-9720P RADEON R7, RAM: 8.00 GB, *Storage*: HDD 1.00 TB

2) *Perangkat Lunak*

Berikut ini adalah perangkat lunak yang digunakan dalam penelitian ini, yaitu: Sistem Operasi Windows 10 Home Single Language, Jupyter Notebook, Visual Studio Code, *Browser* Google Chrome, Microsoft Excel, Bahasa Pemrograman Python

3) *Library*

Berikut ini adalah *library* yang digunakan dalam penelitian ini, yaitu: Tensorflow/Keras, OpenCV, NumPy, Pandas, Matplotlib, Scikit-Learn, Streamlit

4. HASIL DAN ANALISIS

Setelah dilakukan perancangan model seperti yang telah dibahas pada bab sebelumnya, proses dilanjutkan dengan analisis pengujian hyperparameter dan evaluasi model akhir.

A. Analisis Pengujian Hyperparameter

Pada model deep learning, hyperparameter menjadi salah satu faktor yang memengaruhi performa model. Hyperparameter yang tepat akan menghasilkan model dengan performa optimal. Oleh karena itu, diperlukan angka hyperparameter yang tepat untuk mendapatkan performa model yang baik.

Nilai hyperparamater yang akan diuji pada model ini yaitu batch size dan banyaknya dense layer yang digunakan. Hyperparameter tersebut akan diuji dan dianalisis mengenai pengaruhnya terhadap performa model. Performa model akan diamati berdasarkan nilai F1-score rata-rata yang diperoleh dari hasil 5-fold cross validation. Hyperparameter selain dari yang diuji akan dibuat konstan dengan nilai yang terdapat pada Tabel 3 Hasil dari seluruh kombinasi hyperparameter yang telah diuji yaitu sebanyak 12 kali percobaan

TABEL 3. Nilai Pengujian Hyperparameter

Hyperparameter	Nilai
Optimizer	Adam
Activation	Softmax
Max Epoch	20
Early Stopping	10
Learning Rate	0,0001

1) Analisis Batch Size

Pengujian *hyperparameter batch size* pada penelitian ini dilakukan dengan menggunakan nilai *batch size* sebesar 16, 32, dan 64. Hasil dari pengujian tersebut menunjukkan bahwa *batch size* sebesar 16 memberikan nilai *F1-score* yang paling tinggi dibandingkan dengan yang lainnya. Hasil pengujian *batch size* dapat dilihat pada Tabel 4 sampai Tabel 5

TABEL 4. Hasil Pengujian *Batch Size* dengan *Dense Layer* Sebanyak 1

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	1	66,53
20	0,0001	32	1	73,15
20	0,0001	16	1	73,65

TABEL 5. Hasil Pengujian *Batch Size* dengan *Dense Layer* Sebanyak 3

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	3	69,02
20	0,0001	32	3	75,43
20	0,0001	16	3	76,73

TABEL 6. Hasil Pengujian *Batch Size* dengan *Dense Layer* Sebanyak 5

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	5	70,80
20	0,0001	32	5	71,58
20	0,0001	16	5	80,81

TABEL 7. Hasil Pengujian Batch Size dengan Dense Layer Sebanyak 7

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	7	71,87
20	0,0001	32	7	71,03
20	0,0001	16	7	77,55

Berdasarkan hasil pengujian *batch size* pada Tabel 4 sampai Tabel 7 menunjukan bahwa pengurangan *batch size menghasilkan* performa yang semakin baik. Hal tersebut ditandai dengan lebih rendahnya nilai *F1-score* pada saat menggunakan *batch size* sebesar 64 dan 32 dibandingkan menggunakan nilai *batch size* sebesar 16 yang menghasilkan nilai *F1-score* yang paling tinggi. Hal tersebut menunjukkan penggunaan nilai *batch size* yang tidak terlalu besar, cocok untuk diterapkan pada penelitian yang dilakukan karena ukuran *batch size* yang terlalu besar menyebabkan konvergensi model yang lebih cepat, namun menyebabkan model sulit mempelajari pola dari data yang diberikan karena jumlahnya terlalu banyak. Sehingga pada pengujian *batch size* yang dilakukan pada penelitian ini memiliki nilai *F1-score* yang optimal pada saat menggunakan *batch size* sebesar 16.

2) *Analisis Banyaknya* Dense Layer

Analisis hyperparameter selanjutnya yang dilakukan untuk model adalah banyaknya *dense layer*. *Dense layer* yang akan diuji pada penelitian ini yaitu sebanyak 1, 3, 5 dan 7 lapisan yang ditumpuk satu sama lain. Hasil pengujian banyaknya *dense layer* dapat dilihat pada Tabel 8 sampai Tabel 10 berikut

TABEL 8. Hasil Pengujian *Batch Size* sebesar 16

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	16	1	73,65
20	*0,0001*	16	3	76,73
20	0,0001	16	5	80,81
20	0,0001	16	7	77.55

TABEL 9. Hasil Pengujian *Batch Size* sebesar 32

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	32	1	73,15
20	*0,0001*	32	3	75,43
20	0,0001	32	5	71,58
20	0,0001	32	7	71.03

TABEL 10. Hasil Pengujian *Batch Size* sebesar 64

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	1	66,53
20	*0,0001*	64	3	69,02
20	0,0001	64	5	70,80
20	0,0001	64	7	71.87

Berdasarkan hasil pengujian *dense layer* pada Tabel 8 sampai Tabel 10, menunjukan bahwa pada awalnya dengan menambahkan banyaknya *dense layer* membuat performa model meningkat ditandai dengan kenaikan nilai *F1-score* dari *dense layer* sebanyak 1 ke 3. Namun, pada Tabel 9 ketika menggunakan *batch size* sebesar 32 dan *dense layer* ditambahkan menjadi sebanyak 5 performa model menjadi menurun

Pada Tabel 8 ketika menggunakan *batch size* sebesar 16 dan *dense layer* ditambahkan menjadi sebanyak 5 performa model menjadi optimal, karena penambahan *dense layer* menjadi sebanyak 7 membuat nilai *F1-score* menurun. Sedangkan pada Tabel 4.8 ketika menggunakan *batch size* sebesar 64 dan *dense layer* ditambahkan menjadi sebanyak 5 membuat nilai *F1-score* meningkat dan ketika *dense layer* ditambahkan lagi menjadi sebanyak 7 membuat nilai *F1-score* optimal.

Hasil pengujian tersebut menunjukan penambahan banyaknya *dense layer* dapat meningkatkan kemampuan model untuk mempelajari data, namun peningkatan banyaknya *dense layer* di atas jumlah tertentu justru membuat model perlahan-lahan mengalami *overfitting* dan memberikan performa yang kurang optimal.

Bertambah banyaknya *dense layer* dapat menambahkan *learning capacity* dari model tersebut, dengan bertambahnya *learning capacity* membuat model dapat mengekstrak pola- pola yang lebih kompleks dari data yang diberikan. Secara umum, hal tersebut menguntungkan karena model belajar lebih banyak untuk dapat memprediksi data lebih akurat. Namun di sisi lain hal tersebut juga memiliki risiko terjadinya *overfitting* pada model yang dibangun. Kondisi ini berarti model dapat memprediksi data yang telah dipelajari dengan baik, namun memiliki performa yang buruk saat memprediksi data yang belum pernah dilihat.

Hal tersebut terjadi karena lapisan yang lebih banyak membuat model lebih kompleks dan mempelajari pola-pola yang tidak seharusnya dipelajari. Sehingga pada pengujian *dense layer* yang dilakukan pada penelitian ini memiliki nilai *F1-score* yang optimal pada saat menggunakan *dense layer* sebanyak 5.

B. Analisis Evaluasi Model Akhir

Setelah melakukan beberapa percobaan dengan beberapa jenis *hyperparameter*, penelitian ini mendapatkan hasil mengenai model yang memberikan rata-rata performa terbaik dengan pengaturan *hyperparameter* yang telah didapatkan yang dapat dilihat pada Tabel 11

TABEL 11. Hyperparameter Optimal

Max Epoch	*Learning Rate*	*Batch Size*	Banyaknya *Dense Layer*	Rata-rata *F1- score* (%)
20	*0,0001*	64	7	71,87
20	0,0001	32	7	71,03
20	0,0001	16	7	77,55

TABEL 12. Hasil 5-*Fold Cross Validation*

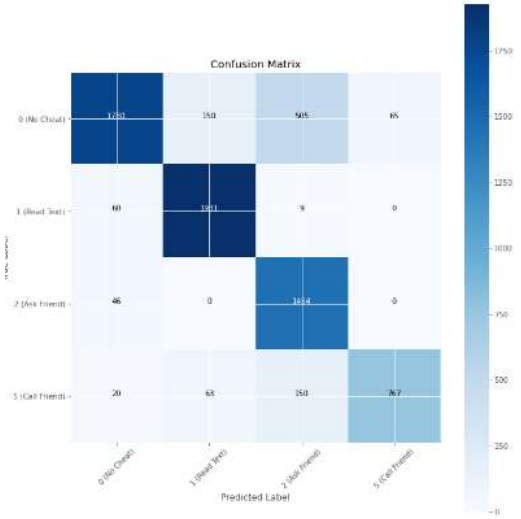
Fold	Nilai
1	Adam
2	Softmax
3	20
4	10
5	0,0001
6	5
Macro Average	16

Model tersebut kemudian dievaluasi lebih jauh menggunakan data uji yang telah dipisahkan sebelumnya. Bukan hanya hasil evaluasi secara keseluruhan, melainkan juga setiap model akan dievaluasi berdasarkan hasil dari setiap fold pada cross validation dan setiap kelasnya.

Berdasarkan seluruh skema pengujian yang dilakukan sebelumnya, didapatkan sebuah model dengan rata-rata performa terbaik dari 5-fold stratified cross validation. Dari hasil 5-fold stratified cross validation tersebut kemudian dapat dilihat hasil dari confusion matrix serta hasil evaluasi dari masing-masing labelnya.

Tabel 12 menunjukkan hasil 5-fold stratified cross validation yang didapat dari model dengan rata-rata hasil prediksi terbaik. Dari cross validation tersebut dijabarkan performa dari masing-masing fold dan rata-rata dari keseluruhan fold.

Dari ke-lima *fold* tersebut, diambil satu model dengan performa *F1-score* terbaik. Model pada *fold* ke-5 yang memberikan performa terbaik, kemudian model tersebut dilakukan evaluasi menggunakan *confusion matrix*. Pada Tabel 4.11 menampilkan hasil pengujian model terakhir terhadap data uji yang disajikan dalam *confusion matrix*.



GAMBAR17 . Hasil Evaluasi per Kelas

Hasil nilai *precision, recall,* dan *F1-score* dari masing- masing kelas di atas dapat disajikan seperti pada Tabel 13

TABEL 13. Hasil Evaluasi per Kelas

Class	*Precision* (%)	*Recall* (%)	*F1- score* (%)
0 *(No Cheat)*	93,38	71,20	80,79
1 (Read Text)	90,06	96,55	93,19
2 *(Ask Friend)*	68,64	96,93	80,37
5 *(Call friend)*	92,18	76,70	83,73
Macro Average	86,07	85,34	84,52

Berdasarkan evaluasi per-kelas tersebut, didapatkan bahwa performa berdasarkan *F1-score* paling rendah didapat oleh kelas 2 *(Ask Friend)* sebesar 80,37% dan paling tinggi didapatkan oleh kelas 1 *(Read Text)* sebesar 93,19%. Hal tersebut terjadi karena adanya *imbalance class*, yang mana kelas dengan jumlah data yang lebih banyak mendominasi ruang sampel. Kelas dengan data lebih banyak cenderung memberikan hasil yang lebih baik, sebab mesin dapat mengenali polanya lebih baik dibanding dengan kelas dengan jumlah data lebih sedikit. Akhirnya, performa dari model yang telah dibangun mendapatkan nilai *F1-score* sebesar 84,52%.

5. KESIMPULAN

Pada penelitian ini telah disajikan proses dan hasil evaluasi dari pembuatan model, berikut ini simpulan dan saran dari penelitian yang sudah dilakukan. Berdasarkan penelitian yang telah dilakukan dalam melakukan prediksi aktivitas manusia dengan metode CNN menggunakan arsitektur MobileNetV2, dapat ditarik kesimpulan sebagai berikut:

1. Human activity recognition atau pengenalan aktivitas manusia dapat dilakukan menggunakan model deep learning dengan metode Convolutional Neural Network (CNN). menggunakan arsitektur MobileNetV2, data yang digunakan berasal dari rekaman video seseorang saat melaksanakan ujian daring berdasarkan tangkapan webcam diantara empat kelas, yaitu kelas 0 (No Cheat), 1 (Read Text), 2 (Ask Friend), dan 5 (Call Friend).
2. Model CNN menggunakan arsitektur MobileNetV2 yang optimal dalam mengenali aktivitas manusia dapat dicari dengan dilakukan beberapa pengujian hyperparameter. Pada penelitian ini, model menghasilkan performa yang optimal ketika menggunakan hyperparameter berupa max epoch sebanyak 20, early stopping dengan patience sebanyak 10, learning rate sebesar 0,0001, batch size sebesar 16, dan dense layer sebanyak 5.
3. Hasil evaluasi model dengan arsitektur MobileNetV2 yang optimal secara keseluruhan fold cross validation memiliki nilai F1-score sebesar 80,81% dan berdasarkan hasil fold yang mempunyai performa terbaik memiliki nilai F1-score sebesar 84,52%.

Penelitian ini masih memiliki beberapa keterbatasan sehingga dapat dikembangkan lebih jauh lagi agar lebih baik. Oleh karena itu, terdapat beberapa saran dari penulis yang dapat dilakukan pada penelitian selanjutnya, yaitu sebagai berikut:

1. Penanganan imbalance class dapat dilakukan lebih lanjut dengan harapan dapat membandingkan strategi penanganan imbalance class yang cocok digunakan untuk menghasilkan hasil performa yang lebih baik, karena kelas 0 (No Cheat) memiliki jumlah data yang lebih banyak dibandingkan dengan kelas lainnya.
2. Model CNN menggunakan arsitektur MobileNetV2 yang dibangun dapat dikembangkan dengan menambahkan lapisan lainnya maupun dikombinasikan dengan model lainnya untuk menghasilkan model yang lebih baik

REFERENSI

[1] Sharma, N. K. et al. (2021) 'CNN Implementation for Detect Cheating in Online Exams During COVID-19 Pandemic: A CVRU Perspective', Materials Today: Proceedings. Elsevier Ltd, (xxxx). doi:10.1016/j.matpr.2021.05.490.

[2] Rukajat, A. (2018) Teknik Evaluasi Pembelajaran. Deepublish.

[3] Ehatisham, M. et al. (2020) 'C2FHAR: Coarse-to-Fine Human Activity Recognition with Behavioral Context Modeling Using Smart Inertial Sensors', IEEE Access, 8, pp. 7731–7747. doi: 10.1109/ACCESS.2020.2964237.

[4] Nurhikmat, T. (2018) Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Convolutional Neural Network (CNN) Pada Citra Wayang Golek. doi: 10.13140/RG.2.2.10880.53768.

[5] Budiman, B. (2021) 'Pendeteksian Penggunaan Masker Wajah Dengan Metode Convolutional Neural Network', Jurnal Ilmu Komputer dan Sistem Informasi, Vol.9 No.1.

[6] Ferrari, A. et al. (2021) 'Trends in Human Activity Recognition using Smartphones', Journal of Reliable Intelligent Environments. Springer International Publishing, 7(3), pp. 189–213. doi: 10.1007/s40860-021-00147-0.

[7] Sanhudo, L. et al. (2021) 'Activity Classification using Accelerometers and Machine Learning for Complex Construction Worker Activities', Journal of Building Engineering, 35 (December). doi: 10.1016/j.jobe.2020.102001.

[8] San-Segundo, R. et al. (2018) 'Robust Human Activity Recognition Using Smartwatches and Smartphones', Engineering Applications of Artificial Intelligence, 72(October 2017), pp. 190–202. doi: 10.1016/j.engappai.2018.04.002.

[9] Zhang, S. et al. (2017) 'A Review on Human Activity Recognition Using Vision-Based Method', Journal of Healthcare Engineering, 2017. doi: 10.1155/2017/3090343.

[10] Sutojo, T. (2017) Pengolahan Citra Digital. Penerbit Andi.

[11] Deng, L. and Yu, D. (2013) 'Deep Learning: Methods and Applications', Foundations and Trends in Signal Processing, 7(3–4), pp. 197–387. doi: 10.1561/2000000039.

[12] Géron, A. (2019) Hands-on Machine Learning whith Scikit-Learing, Keras and Tensorfow, O'Reilly Media, Inc.

[13] Wei, L. (2017) 'Human Activity Recognition Using Deep Neural Network With Contextual Information', VISIGRAPP 2017 - Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, (Visigrapp), pp. 34–43. doi: 10.5220/0006099500340043.

[14] Kim, H. (2019) 'Human Activity Recognition By Using Convolutional Neural Network', International Journal of Electrical and Computer Engineering, 9(6), pp. 5270–5276. doi: 10.11591/ijece.v9i6.pp5270-5276.

[15] Ignatov, A. (2018) 'Real-Time Human Activity Recognition from Accelerometer Data using Convolutional Neural Networks', Applied Soft Computing Journal. Elsevier B.V., 62, pp. 915–922. doi:10.1016/j.asoc.2017.09.027.

[16] Yeole, C. et al. (2021) 'Deep Neural Network Approaches for Video Based Human Activity Recognition', 6(6), pp. 1586–1589.

[17] Patterson, J. and Gibson, A. (2017) Deep Learning: A Practitioner's Approach, O'Reilly.

[18] Rahmaniar, W. and Hernawan, A. (2021) 'Real-Time Human Detection Using Deep Learning On Embedded Platforms: A Review', Journal of Robotics and Control (JRC), 2(6), pp. 462-468Y. doi: 10.18196/jrc.26123.

[19] Nagrath, P. et al. (2021) 'SSDMNV2: A Real Time DNN-Based Face Mask Detection System Using Single Shot Multibox Detector and MobileNetV2', Sustainable Cities and Society. Elsevier Ltd, 66(December 2020), p. 102692. doi: 10.1016/j.scs.2020.102692.

[20] Sandler, M. et al. (2018) 'MobileNetV2: Inverted Residuals and Linear Bottlenecks', Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 4510–4520.

[21] Burkov, A. (2019) 'The Hundred-Page Machine Learning Book', Expert Systems, 5(2), pp. 132–150. doi: 10.1111/j.1468-0394.1988.tb00341.x.

[22] Zhang, F. et al. (2021) 'Accelerating Hyperparameter Tuning in Machine Learning for Alzheimer's Disease With High Performance Computing', Frontiers in Artificial Intelligence, 4(December), pp. 1–9. doi: 10.3389/frai.2021.798962.

[23] Molokwu, B. C. et al. (2020) Node Classification in complex Social Graphs via Knowledge-Graph Embeddings and Convolutional Neural Network, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer International Publishing. doi: 10.1007/978-3-030-50433-5_15.

[24] Goodfellow, I., Bengio, Y. and Courville, A. (2016) Deep Learning. Available at: http://www.deeplearningbook.org/front_matter.pdf.

[25] Müller, A. and Guido, S. (2017) Introduction to Machine Learning with Python. doi: 10.1007/978-3-030-36826-5_10.

[26] Kuhn, M. and Johnson, K. (2013) Applied Predictive Modeling, Applied Predictive Modeling. doi: 10.1007/978-1-4614-6849-3.

[27] Jukes, E. (2017) Encyclopedia of Machine Learning and Data Mining, Encyclopedia of Machine Learning and Data Mining. doi:10.1007/978-1-4899-7687-1.

[28] Atoum, Y. et al. (2017) 'Automated Online Exam Proctoring', IEEE Transactions on Multimedia, 19(7), pp. 1609–1624. doi:10.1109/TMM.2017.2656064.

[29] Khan, S. et al. (2018) 'A Guide to Convolutional Neural Networks for Computer Vision', Synthesis Lectures on Computer Vision, 8(1), pp. 1–207. doi: 10.2200/s00822ed1v01y201712cov015