ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika

ISSN(p): 2338-8323 | ISSN(e): 2459-9638 | Vol. 11 | No. 4 | Halaman 919 - 932
DOI: http://dx.doi.org/10.26760/elkomika.v11i4.919 Oktober 2023

Sistem Verifikasi Kekerabatan berbasis 3D ResNet-18 menggunakan Jetson Nano

FAUZAN AWWAL MUKHRODI, IKE FIBRIANI, KHAIRUL ANAM, ALI RIZAL CHAIDIR

Jurusan Teknik Elektro, Universitas Jember, Indonesia
Email: khairul@unej.ac.id

Received 9 Juni 2023 **|** *Revised* 18 Juli 202x **|** *Accepted* 14 Agustus 2023

ABSTRAK

Verifikasi kekerabatan berbasis citra wajah merupakan salah satu penerapan sistem Artificial Intelligence yang berguna dalam kehidupan, misalnya untuk penyelidikan kriminal, analisis silsilah, dan lainnya. Perancangan sistem pengenalan wajah pada verifikasi kekerabatan dapat dilakukan menggunakan salah satu algoritma deep learning yaitu metode Convolutional Neural Network. Penelitian ini dilakukan dengan tujuan untuk mengetahui kinerja dari 3D ResNet-18 pada verifikasi kekerabatan berdasarkan sistem pengenalan wajah dan mengetahui kinerja 3D ResNet-18 saat menggunakan embedded system secara real time. Hasil penelitian kinerja ResNet-18 tanpa embedded system memperoleh nilai akurasi training sebesar 0,9771 menggunakan optimizer RMSprop dengan epoch 30 dan batch size 25. Pada pengujian kinerja real time ResNet-18, optimizer SGD berhasil pada ukuran batch size 10, 15, dan 25. Namun untuk pengujian pada perangkat Jetson Nano, optimizer RMSprop gagal akibat ukuran model yang terlalu besar.

***Kata kunci**: embedded sistem, CNN, 3D Resnet18, RMSprop, kekerabatan*

ABSTRACT

Face-based kinship verification is one of the applications of artificial intelligence systems that are useful in various aspects of life, such as criminal investigations, pedigree analysis, and more. The design of a face recognition system for kinship verification can be done using one of the deep learning algorithms, namely the convolutional neural network method. This research was conducted with the aim of determining the performance of 3D ResNet-18 in kinship verification based on face recognition systems and assessing the performance of 3D ResNet-18 when using an embedded system in real time. The results of the ResNet-18 performance research without an embedded system obtained a training accuracy of 0.9771 using the RMSprop optimizer with 30 epochs and a batch size of 25. In real-time performance testing of ResNet-18, the SGD optimizer succeeded with batch sizes of 10, 15, and 25. However, during testing on the Jetson Nano device, the RMSprop optimizer failed due to the size of the model being too large.

***Keywords**: Embedded System, CNN, 3D Resnet-18, RMSprop, Kinship*

1. PENDAHULUAN

Pengenalan wajah telah berkembang pesat dengan menerapkan berbagai metode dan algoritma. Pengenalan wajah (*face recognition technology*) adalah salah satu teknologi berbasis kecerdasan buatan atau *Artificial Intelligence* (AI) yang dapat mengkarakterisasi wajah melalui gambar, video, atau elemen audiovisual. Dalam praktiknya, pengenalan wajah menggunakan informasi dari wajah berupa jaringan geometrik dengan titik-titik *landmark* wajah. Teknologi pengenalan wajah dapat digunakan secara luas di berbagai bidang seperti kedokteran, kesehatan, robotika, dan mengemudi otonom. Pengenalan wajah menghadirkan beberapa tantangan, yaitu sangat bergantung pada kumpulan data (*database*) dan penerapan fitur dinamis. Sebagian besar area aplikasi pengenalan wajah memerlukan respons wajah waktu nyata **(Kim, 2020).**

Permasalahan yang menarik adalah verifikasi kekerabatan berbasis citra wajah. Verifikasi kekerabatan berbasis citra wajah sangat berguna untuk aplikasi, salah satunya digunakan untuk penyelidikan kriminal, analisis silsilah, analisis pustaka, bahkan juga untuk interaksi komputer dan manusia. Verifikasi kekerabatan berbasis citra wajah akan memprediksi kekerabatan dari kedua citra wajah tersebut **(Rachmadi & Purnama, 2018).** Klasifikasi kekerabatan dari citra wajah merupakan masalah yang muncul dalam visi komputer. Dari aspek pengenalan wajah, kekerabatan memberi kita kesempatan yang berharga dan operasional untuk membangun hubungan yang bermanfaat antara orang-orang berdasarkan sinyal. Analisis dan pemodelan citra wajah merupakan salah satu riset aktif di bidang visi komputer dan biometrika, terutama diera internet dan *big data*. Pembelajaran kekerabatan juga dimotivasi oleh tujuan jangka panjang dari visi komputer untuk melampaui pemahaman entitas visual (misalnya, "wajah siapa ini?") supaya bisa menyelidiki hubungan dua atau tiga di antara beberapa entitas visual seperti menjawab pertanyaan apakah seorang anak dalam foto milik orang tua dan sebenarnya. Penelitian terbaru menunjukkan bahwa algoritma visi komputer mampu memahami citra wajah individu **(Qin, dkk, 2015).**

Contoh dari model *neural network,* CNN dengan bentuk sederhananya berupa *fully connected layer* yang artinya masing – masing neuron di setiap *layer* yang terhubung terhadap masing-masing setiap neuron pada *layer* sebelumnya dan setelahnya dengan sebutan *Multi-Layer Perceptron* **(IIKKA TEIVAS, 2017)**. Perancangan sistem pengenalan wajah pada verifikasi kekerabatan dapat dilakukan dengan menerapkan metode *Convolutional Neural Network* (CNN). CNN merupakan algoritma *deep learning* yang dapat mengambil gambar *input*, menetapkan kepentingan berbagai objek dalam gambar, dan dapat membedakan satu dari yang lain. Banyak kasus tentang CNN yang masih dalam tahap pengembangan hingga saat ini. CNN memiliki sebuah *preprocessing* dengan ukuran yang relatif sedikit dan juga terdapat model yang telah diekstraksi dari CNN {Formatting Citation}. Pada riset ini dilakukan proses pengenalan wajah dengan menerapkan metode CNN untuk verifikasi kekerabatan. Bahasa pemrograman yang digunakan pada metode CNN salah satunya adalah bahasa python yang banyak digunakan oleh kalangan umum dan bagus untuk *machine learning*. Python memiliki banyak fitur yang digunakan dan kompleks. Bahasa python mudah dimengerti dan dinamis sehingga bahasa ini dapat dikatakan dengan *multiplatform* sehingga dapat digunakan untuk *web development*, *mobile apps*, *data science*, *robotics, machine learning*, data analis dan *Artificial Intelligence*.

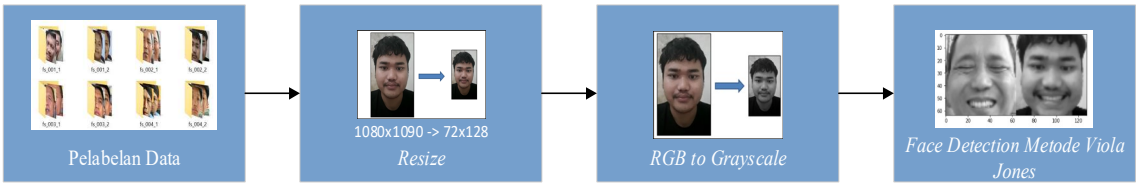
Sistem identifikasi kekerabatan telah diteliti oleh Hamdi Dibeklioglu, yaitu menganalisis keefektifan verifikasi kekerabatan atau garis keturunan dalam video dengan memanfaatkan transformasi visual atau wajah. Proses identifikasi menggunakan *visual tracing* dan ekstraksi fitur *landmark*, di mana ekstraksi menggunakan titik yang dilacak untuk mewakili deformasi

permukaan pada daerah mata, alis dan mulut. *Dataset* yang digunakan adalah Uva-NEMO *smile database* **(Dibeklioglu, 2017)**. Penelitian selanjutnya dilakukan oleh **(Chergui, dkk, 2020)** yaitu penerapan klasifikasi CNN dengan model VGG-16 dan VGG-Face, di mana *database* yang digunakan berupa 2D seperti Cornell KinFace, UB KinFace, Family 101, KinFace W-I, dan KinFace W-II. Pada penelitian Chergui et al. disebutkan bahwa sistem verifikasi *kinship* melalui proses klasifikasi SVM dan dilanjutkan dengan *cross-validation*. Verifikasi kekerabatan dengan gambar biasanya masih rumit untuk dikonfirmasi karena masih dalam tahap pengembangan hingga sekarang dan literatur serta sumber yang relatif langka **(Wang, dkk, 2018)**. Berdasarkan permasalahan uraian tersebut, penulis bermaksud untuk melakukan penelitian berupa perancangan *embedded* sistem pengenalan wajah untuk mendeteksi kekerabatan berdasarkan wajah manusia pada video *dataset* menggunakan CNN 3D ResNet-18.

2. METODE PENELITIAN

2.1 Pra – Proses

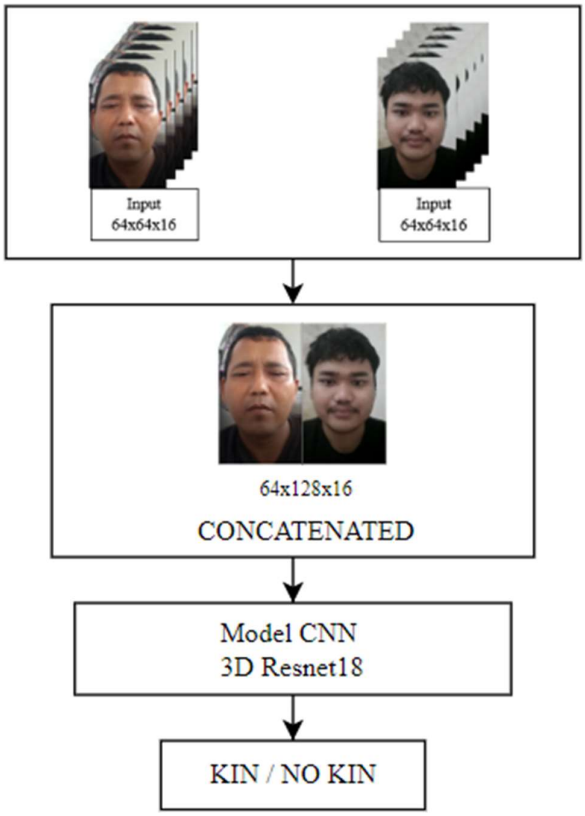
Blok pra-proses dapat dilihat pada Gambar 1. Data yang digunakan sebelum dimasukkan ke dalam model *deep learning*, dipraproses dengan memberi label atau mengklasifikasikan setiap sampel dalam *dataset*.



Gambar 1. Blok Pra-Proses

2.2 Blok Diagram Sistem Perangkat Lunak

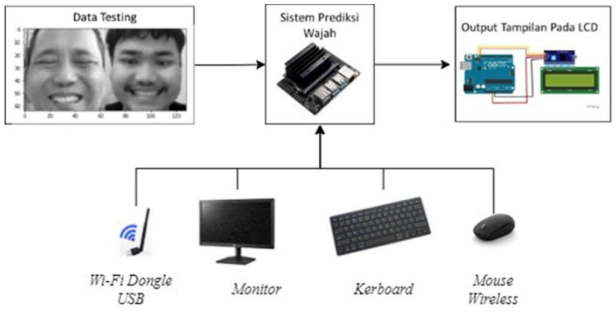
Berdasarkan Gambar 2 yaitu blok diagram sederhana proses verifikasi kekerabatan menggunakan 3D ResNet-18, terdapat dua *input* berupa video ayah dan anak kandung laki-laki yang memiliki ukuran $64 \times 64 \times 16$. Kemudian data *input* digabung dengan proses *concatenated* dan dilanjutkan dengan proses *training* menggunakan algoritma CNN arsitektur 3D ResNet-18. Selanjutnya, data diproses pada *fully connected layer* agar *output* yang dihasilkann sesuai harapan yaitu hubungan kekerabatan (kin/no-kin).



Gambar 2. Blok Diagram Sederhana Verifikasi Kekerabatan dengan CNN

2.3 Rancangan Perangkat Keras

Rancangan perangkat keras yang ditunjukkan pada Gambar 3 pada penelitian ini menggunakan jetson nano. Jetson nano merupakan mini PC yang digunakan pada pemrosesan *data testing* **(Sasono, 2022)**, di mana data akan ditampilkan di monitor yang telah dihubungkan dengan jetson nano, kemudian hasil *data testing* tersebut akan diproses ulang oleh Arduino UNO. *Output* yang ditampilkan berupa *text* terdeteksi *kinship* dan terdeteksi *no kinship* pada LCD 16×2.

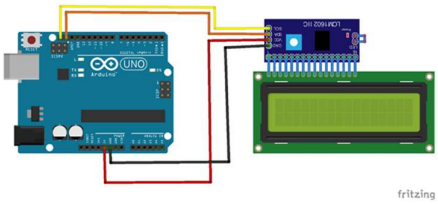


Gambar 3. Rancangan Alat Prediksi Wajah

2.4 Rangkaian Elektrik

Pada Gambar 4, dapat diketahui bahwa semua pin pada LCD dihubungkan pada pin I2C yang masing-masing memiliki pin berjumlah 16. Pada bagian pin I2C yang lain yang mana terdapat pin SDA dan SCL, masing-masing dihubungkan ke pin A4 dan pin A5 yang ada pada pin arduino

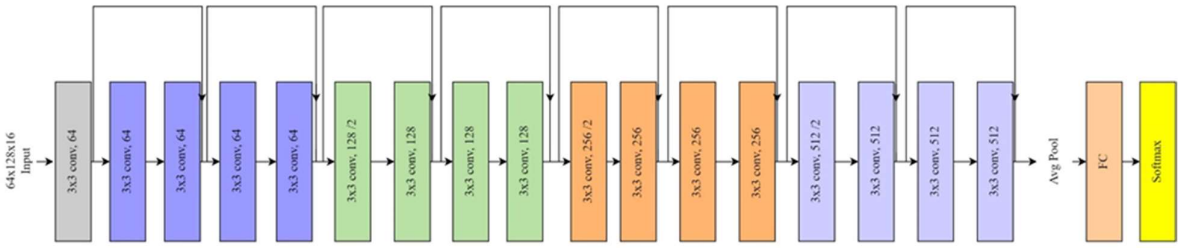
UNO. Sedangkan pin Vcc dan *ground* yang ada pada pin I2C dihubungkan ke pin 5V dan *ground* yang ada pada pin arduino UNO **(Sokop, dkk, 2016)**.



Gambar 4. Rangkaian Elektronika LCD dengan Arduino UNO

2.5 Merancang Model CNN

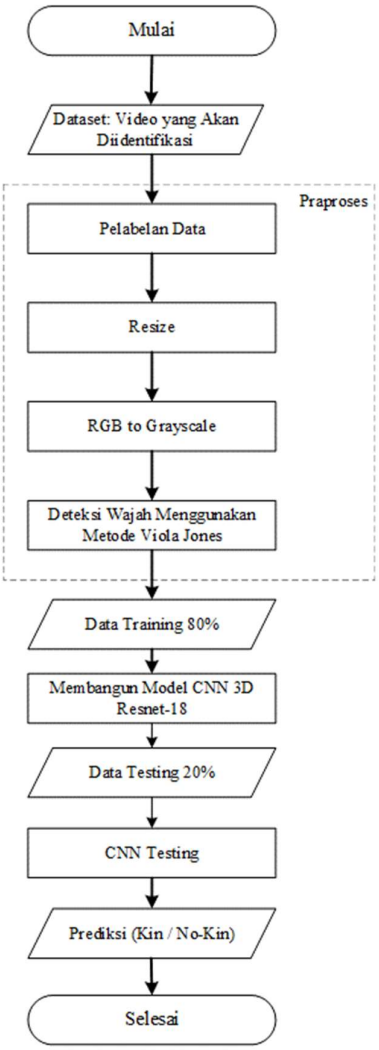
Pada tahap ini dilakukan proses perancangan arsitektur CNN 3D. Arsitektur 3D-CNN diambil dari pengembangan pengujian terkini yang dilakukan oleh **(Vu, dkk, 2021)**, diperkenalkan oleh **(Tran, dkk, 2015)** dengan sebutan arsitektur C3D. CNN 3D pada dasarnya adalah dimensi ekstra dari CNN 2D. Tidak seperti 2D-CNN, ukuran *kernel layer* di CNN 3D berukuran 3 dimensi (2x2x2x1) **(Arunnehru, dkk, 2018)**. Perancangan model dilakukan untuk mengekstraksi fitur dan klasifikasi yang dirancang untuk 3D-CNN menggunakan model ResNet-18 berdasarkan karya **(He, dkk, 2015)**. ResNet-18 merupakan salah satu model *deep learning* yang memiliki 18 lapisan. ResNet-18 memperkenalkan konsep *residual connections* di mana *input* diteruskan langsung ke beberapa lapisan jaringan yang selanjutnya menambahkan hasilnya ke hasil dari lapisan lainnya. Untuk model arsitektur dari ResNet-18 ditunjukkan pada Gambar 5 berikut.



Gambar 5. *Original ResNet-18 Architecture*

2.6 *Flowchart* Pengambilan Data Hasil

Berdasarkan *flowchart* pengambilan data hasil, pada tahap pra-proses menyiapkan *dataset* yang berupa video yang kemudian dilanjutkan untuk pembagian *dataset* untuk *testing* dan *training*. Untuk data *training* akan dilakukan pelatihan oleh model 3d ResNet-18 tertentu, jika proses *training* selesai dan model telah terlatih maka akan dilakukan proses *testing* untuk dilakukan klasifikasi kekerabatan, *flowchart* dapat dilihat pada Gambar 6. Adapun kode program dapat dilihat pada https://github.com/khanamsk/verkekerabatan.git.



Gambar 6. *Flowchart* Pengambilan Data Hasil

2.7 Pembagian Data *Testing* dan Data *Training*

Dataset yang ada dibagi menjadi dua yaitu data *testing* dan data *training*. Pada data *training* ini digunakan untuk mengoptimalkan model CNN 3D ResNet-18, sedangkan data *testing* ini digunakan untuk menguji model CNN 3D ResNet-18. *Dataset* yang akan digunakan tidak tersedia untuk pembagian data tersebut. Oleh sebab itu, *dataset* akan dibagi menjadi dua bagian pertama data *training* digunakan untuk melatih model cara membuat prediksi yang benar, data *training* merupakan data yang sangat penting dari proses pembuatan model karena data *training* akan sangat berpengaruh terhadap kualitas model yang dihasilkan. Bagian selanjutnya data *testing*, pada proses ini digunakan untuk membandingkan hasil prediksi dari model dengan data yang real. Data ini berisi *input* dan *output* yang digunakan untuk menguji kinerja dari model. Maka, *dataset* dibagi secara acak dengan persentase data *training* 80% dan data *testing* 20%.

3. HASIL DAN PEMBAHASAN

3.1 Proses Pelabelan Data

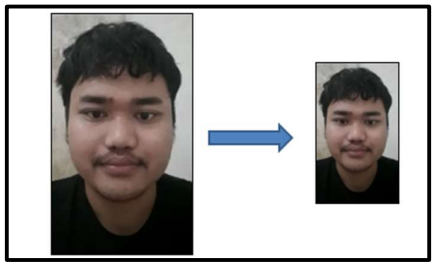
Proses pelabelan data dilakukan pada *dataset* memberikan label atau kategori di setiap *item* dalam *dataset*. Label ini digunakan untuk membedakan *item-item* yang ada pada *dataset*. Pelabelan penting dilakukan karena akan memengaruhi performa model yang digunakan. Pelabelan harus dilakukan dengan akurat supaya model yang dihasilkan baik dalam melakukan proses prediksi. Apabila pelabelan salah, model akan kurang efektif dalam membuat prediksi. Pelabelan *dataset* ditunjukkan pada Gambar 7.

Gambar 7. Pelabelan *Dataset*

Algoritma CNN melabeli data video dalam bentuk *numpy* yang berfungsi untuk mempermudah CNN membaca data label. Data label ini diubah menjadi bentuk *array* agar dikenali oleh CNN saat dilakukan proses *training*, sehingga saat pemanggilan data tidak perlu dilakukan secara manual untuk proses pelabelan data video.

3.2 *Resize*

Mengubah ukuran atau *resize* dilakukan pada sebuah *input* atau masukan berupa kumpulan *frame*, yang bertujuan untuk memudahkan proses pengolahan citra. Jika ukuran *frame* tetap atau tidak berubah maka akan membutuhkan waktu cukup lama pada saat proses *training* dilakukan. Dalam ukuran penelitian yang dilakukan dilakukan pengubahan ukuran dari 1080x1920 *pixel* diubah menjadi 72x128. Perubahan ukuran dalam berukuran *frame* dapat ditinjau pada Gambar 8. Pada proses ini menggunakan fungsi "cv2.resize" dalam modul *Open*CV digunakan untuk mengubah ukuran citra.

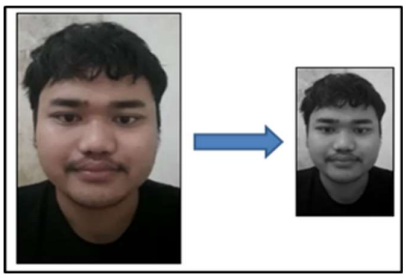


Gambar 8. Perubahan Ukuran *Frame*

3.3 RGB *to Grayscale*

Proses pengubahan warna citra dari warna RGB menjadi abu-abu dimaksudkan untuk mempermudah proses pencitraan. Jika kondisi citra adalah warna RGB maka terdapat 3 *layer* matrik yaitu, R-*layer*, G-*layer*, dan B-*layer*, ketiga *layer* tersebut tetap diperhatikan di setiap pengolahan citra yang terjadi sedemikian rupa sehingga membutuhkan waktu yang cukup

lama dalam pengolahan citra. Dapat dilihat pada Gambar 9 yang menunjukkan gambar citra RGB diubah menjadi citra abu-abu.

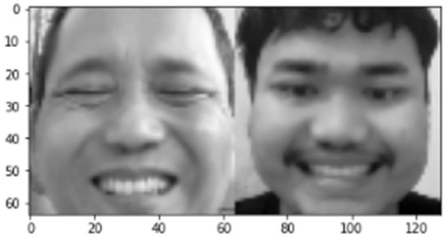


Gambar 9. Citra RGB Diubah Menjadi *Grayscale*

Proses selanjutnya yaitu proses konversi *dataset* yang dilakukan dengan cara mempertahankan kualitas citra, mengubah ukuran *file* agar lebih ringan dan tujuannya untuk mempermudah dalam proses *training* CNN. Pada penelitian ini, konversi dilakukan dari video yang diubah menjadi gambar dengan menggunakan salah satu fungsi yang berada pada OpenCV. Video diambil menjadi beberapa *frame* yaitu sebanyak 16 *frame* dan diproses pada CNN. Setelah proses *capture* dilakukan, langkah selanjutnya proses *resize* dapat dilihat pada Gambar 8, kemudian diteruskan menuju proses citra RGB yang diubah menjadi *grayscale* proses ini dapat dilihat pada Gambar 9 tentang perubahan RGB menjadi *grayscale*.

3.4 *Face Detection*

Proses *face detection* menggunakan metode *viola jones* (**Viola & Jones, 2018**). Metode ini melakukan proses klasifikasi dengan waktu yang singkat yaitu dengan berpusat pada daerah wajah atau objek berada. Hasilnya berupa sebuah *Region of Interest* (ROI) yang berbentuk kotak yang berada pada area wajah (**Vikam & Padmavathi, 2017**). Pada tahap terakhir menggabungkan dua *frame* dengan ukuran *pixel* dari 72×128 menjadi 64×128, di mana ukuran 64×128 ini yang akan diolah pada pengolahan citra CNN. Penggabungan dua *frame* ditunjukkan pada Gambar 10.



Gambar 10. Gambar Penggabungan Dua *Frame* dengan Metode *Viola Jones*

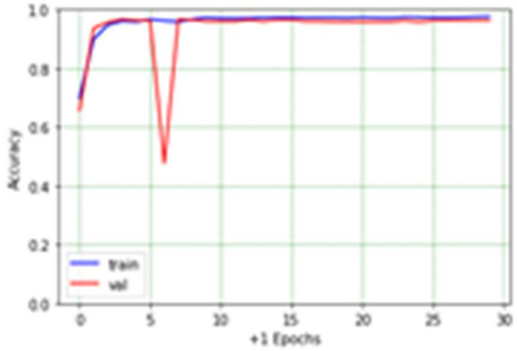
3.5 Sistem CNN 3D ResNet-18

Arsitektur ResNet-18 sendiri terdiri dari lapisan-lapisan konvolusi, lapisan normalisasi *batch*, lapisan aktivasi ReLU, dan lapisan *pooling*. CNN 3D ResNet-18 menambahkan dimensi waktu ke dalam lapisan-lapisan ini, sehingga memungkinkan model untuk memperoleh pemahaman yang lebih baik tentang perubahan spasial dan temporal dalam video. *Input*_1 merupakan ukuran dari *input*-an yang digunakan memiliki ukuran 64x128, 16 merupakan jumlah *frame* yang diambil pada awal proses konvolusi. Conv3D_1 merupakan lapisan konvolusi 3D dengan 64 filter, ukuran kernel 3x3 dan fungsi altivasi RelU yang di mana ukuran *ouput*-nya 8x32x64x64. Lalu ada proses *max pooling* 3D. Lapisan *max pooling* 3D dengan ukuran *output* 4x16x32x64. Selanjutnya, *Residual Blocks* : Arsitektur ResNet terdiri dari beberapa blok residual. Dalam ResNet-18, terdapat total empat blok residual. Setiap blok residual terdiri dari beberapa lapisan konvolusi 3D dan lapisan normalisasi. *AveragePooling*3D : Lapisan *average*

pooling 3D dengan menghasilkan *output* dengan ukuran 512 x 1 x 1 x 1. *Flatten*: Mengubah *output* dari lapisan sebelumnya menjadi vektor satu dimensi dengan ukuran 512. *Dense*: *Lapisan fully connected* dengan dua *neuron*, yang menghasilkan *output* dengan ukuran 2.

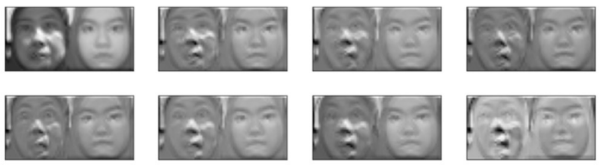
3.6 *Training* dan *Testing*

Pada proses kali ini melakukan pemberian data *training* yang sudah dilakukan pelabelan untuk model CNN dengan tujuan untuk meminimalisir kesalahan model dalam melakukan *training*. Sederhananya, model CNN dipandu untuk mendapatkan bobot terbaik di setiap lapisan neuron berdasarkan data pelatihan yang diterima. *Input* model yang masuk ke dalam proses pelatihan disebut model CNN terlatih. Terdapat istilah *batch size* yaitu jumlah data yang masuk pada model *training*, untuk parameter model CNN lansung diperbarui setiap satu *batch* atau juga disebut dengan kesuluruhan setiap data *training* yang berjumlah 4898 data *training* masuk ke dalam model dan dihitung sebagai satu *epoch*. Proses *training* ini dilakukan sebanyak 30 *epoch* dan hasil akurasi keakuratan model selama proses *training* dapat dipantau dengan validasi yang menggambarkan keakuratan selama tes akhir dengan tetap mempertahankan kinerja model CNN. Grafik pemantauan ditunjukkan pada Gambar 11.



Gambar 11. Pemantauan Akurasi *Training* (Biru) dan Akurasi Validasi (Merah)

Untuk proses *testing* di sini menggunakan jumlah data sebanyak 1225 data *testing* setiap data yang masuk dalam sistem diambil sebanyak 16 *frame* pertama. Selanjutnya hasil konvolusi yang didapat dalam bentuk *feature map*, kemudian hasil *feature map* akan dilanjutkan sebagai *input* untuk proses konvolusi selanjutnya. Proses konvolusi ini dilakukan berulang kali sesuai dengan model CNN yang digunakan pada penelitian ini model yang digunakan ResNet-18 di mana proses konvolusi sebanyak 18 kali. Dari data 16 *frame* pertama setelah memasuki proses konvolusi maka pada setiap satu *kernel* ini terdiri dari delapan *frame* dengan jumlah total *kernel* yaitu 64 *kernel*, sehingga ukuran yang awalnya 16x64x128, 16 di sini menunjukkan jumlah *frame* lalu 64x128 adalah ukuran resolusi gambar. Setelah proses konvolusi pertama menjadi 8x32x64 dan jumlah *frame* ikut berkurang menjadi 8 dapat dilihat pada Gambar 12 hasil konvolusi pertama pada 1 *kernel*.



Gambar 12. Hasil Konvolusi Pertama 3D Seluruh Dimensi dari Kernel Ke-1

Setelah memasuki proses konvolusi, ukuran *feature map* akan menjadi lebih kecil setelah melalui proses *pooling* dan ukuran *feature map* akan menjadi lebih kecil lagi. Hal tersebut dapat membuat model lebih mudah dalam menangkap pola – pola dan membuat proses

training menjadi lebih efisien. Ukuran yang lebih kecil juga membuat model lebih mudah menangkap fitur – fitur penting dan mengurangi jumlah parameter yang harus dilatih, yang membuat proses dari *training* menjadi efisien dan mencegah *overfitting*. Dari Gambar 13 diketahui dari hasil konvolusi ketiga pada dimensi 1 di mana ukurannya menjadi 4x16x32 jumlah *frame* yang diambil sejumlah empat dengan ukuran resolusi 16x32.

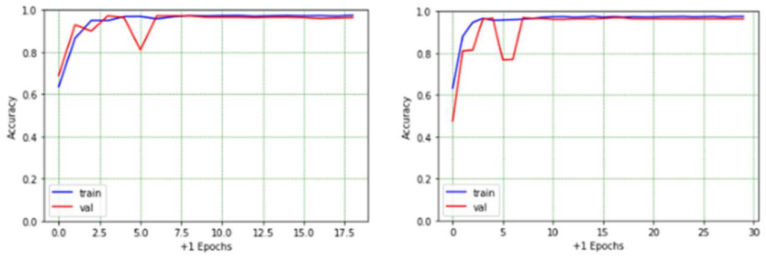
Gambar 13. Hasil Konvolusi Ketiga 3D Dimensi Ke-1

3.7 Pengujian Kinerja 3D ResNet-18

Pengujian kinerja 3D ResNet-18 terdiri dari pengujian *optimizer*, pengujian *early stopping*, pengujian *epoch*, dan pengujian *batch size*. Pengujian *optimizer* dilakukan dengan beberapa *optimizer* yang berbeda dan membandingkan hasil dari masing-masing algoritma. *Optimizer* yang digunakan dalam pengujian *optimizer* pada CNN adalah *Adaptive Moment Estimation* (Adam), *Root Mean Square Propagation* (RMSprop), Adaelta, dan Adagrad dengan masing-masing *learning rate* sebesar 0.001, serta *Stochastic Gradient Descent* (SGD) dengan *learning rate* sebesar 0.01. Hasil pengujian dari beberapa *optimizer* ditunjukkan pada Tabel 1.

Tabel 1. Pengujian Optimizer

Optimizer	**Akurasi**	
	Training	***Testing***
SGD	0.9750	0.9592
RMSprop	0.9637	0.9771
Adam	0.9517	0.9706
Adadelta	0.8342	0.8024
Adagrad	0.9705	0.9551



(a) **(b)**

Gambar 14. (a) Pengujian dengan *Early Stopping*, (b) Pengujian tanpa *Early Stopping*

Pengujian tanpa *early stopping* akan melakukan proses *training* dan akan berhenti ketika akurasi yang diperoleh sudah stabil dan dinyatakan bagus. Hal ini bertujuan untuk mencegah *overfitting*. Dalam pengujian *early stopping* memantau kinerja model selama proses pelatihan

dan memutuskan untuk menghentikan pelatihan jika tidak ada peningkatan yang signifikan dalam kinerja. Dari Gambar 17 pengujian dengan *early stopping training* berhenti saat *epoch* ke-19, dikarenakan proses *training* tidak terjadi peningkatan yang signifikan menghasilkan akurasi 97% dan untuk akurasi *testing* 95%. Pelatihan dihentikan karena akurasi tidak meningkat setelah beberapa *epoch*. Hasil pengujian ditunjukkan pada Gambar 14.

Tabel 2. Pengujian *Epoch Training*

Epoch	**Akurasi *Training***	**Akurasi *Testing***
10	0.9569	0.9143
20	0.9569	0.9551
30	0.9637	0.9771
40	0.9646	0.9135
50	0.9673	0.9282

Pengujian *epoch* pada pelatihan CNN bertujuan untuk menentukan jumlah *epoch* yang optimal pada proses *training*. Pengujian *epoch* dilakukan dengan melatih model menggunakan jumlah *epoch* yang berbeda-beda dan membandingkan nilai akurasinya. Pengujian *epoch* disini menggunakan *optimizer* RMSprop dengan menggunakan *learning rate* 0.001 dikarenakan hasil dari akurasi *training* dan *testing* yang didapat tinggi dan pengujian *epoch* ini menggunakan parameter *epoch* 10, 20, 30, 40 dan 50. Hasil pengujian *epoch* pada Tabel 2 diperoleh yaitu nilai akurasi tertinggi saat *epoch* 30.

Tabel 3. Pengujian dengan Beberapa *Batch Size*

Batch Size	**Akurasi *Training***	**Akurasi *Testing***
10	0.9576	0.8718
15	0.9594	0.9576
25	0.9637	0.9771
30	0.9614	0.9714
35	0.9571	0.9208

Pengujian *batch size* dilakukan untuk menemukan ukuran *batch* yang paling cocok untuk digunakan. Pada pengujian ini dilakukan menggunakan *optimizer* RMSprop *learning rate* 0.001 dan menggunakan *epoch* 30, dikarenakan pengujian sebelumnya menghasilkan nilai akurasi yang tinggi. Hasil pengujian pada Tabel 3 menunjukkan bahwa semakin besar ukuran *batch* yang digunakan, semakin tinggi pula kecepatan pelatihan model.

3.8 Pengujian Kinerja *Real-Time*

Pada pengujian ini di mana untuk mengetahui kinerja secara *real time*, dilakukan uji coba pada dua *optimizer* SGD dan *optimizer* RMSprop. Ukuran *batch size* yang berbeda. Hasil pengujian menunjukkan bahwa *optimizer* SGD di sini berhasil pada ukuran *batch size* 10, 15, dan 25. Namun, *optimizer* RMSprop tidak berhasil pada saat diuji pada jetson nano ukuran model yang terlalu besar dapat menjadi masalah ketika menjalankan model di jetson nano, karena jetson nano memiliki keterbatasan sumber daya seperti CPU, GPU dan memori yang terbatas. Dari pengujian *batch size* 10 pada *optimizer* RMSprop terdapat *error*, yang menunjukkan bahwa memori tersedia pada perangkat GPU sudah penuh sehingga tidak dapat lagi melakukan alokasi memori baru. Hasil Pengujian ditunjukkan pada Tabel 4.

Tabel 4. Pengujian *Batch Size* pada Kinerja *Real-Time*

Optimizer	***Batch Size***	**Keterangan**
SGD	10	Berhasil
SGD	15	Berhasil
SGD	25	Berhasil
RMSprop	10	Tidak Berhasil

Tabel 5. Pengujian *Delay* pada Kinerja *Real-Time*

Optimizer	*Batch Size*	Menit	Keterangan
SGD	10	3.75	Berhasil
SGD	15	3.4	Berhasil
SGD	25	3.47	Berhasil
RMSprop	10	-	Tidak Berhasil

Pada pengujian *real-time*, *delay* merupakan waktu yang dibutuhkan untuk memproses data secara simulasi sehingga dapat dianggap sebagai waktu tunggu yang muncul pada saat aplikasi dijalankan. Apabila dibandingkan dengan kemampuan manusia dalam melihat kesamaan wajah alat ini dapat dikatakan lebih lambat, karena waktu *delay* 3.75 menit dan juga dari faktor spefikasi dari jetson nano yang membutuh spesifikasi yang lebih bagus. Hasil pengujian ditunjukkan pada Tabel 5.

3.9 Hasil Pengujian Alat

Pengujian ini dilakukan untuk mengetahui hasil dari jetson nano dan digunakan sebagai *embedded system* yang menampilkan hasil dari performa CNN dalam mendeteksi kin atau no kin. Hasil Pengujian dari jetson nano ditunjukkan pada Tabel 6.

Tabel 6. Hasil Pengujian Jetson Nano dan LCD 16x2

No	**Citra Wajah Pada Jetson Nano**	**Hasil Prediksi (Kin / No Kin)**	**Tampilan LCD 16x2**
1.		Kin	
2.		No Kin	
3.		No Kin	
4.		Kin	
5.		Kin	

Berdasarkan Tabel 6 data hasil pengujian jetson nano, dapat diketahui bahwa hasil yang ditampilkan dari jetson nano dan LCD adalah sama. pada penelitian ini, LCD mampu untuk menampilkan hasil prediksi dari jetson nano secara terus menerus hingga 1225 data *testing* selesai melalui proses *testing*. prediksi yang dilakukan oleh algoritma CNN ResNet-18 berhasil

karena menghasilkan nilai akurasi sebesar 96% dan hasil prediksi dapat dimonitor menggunakan LCD dengan bantuan jetson nano.

4. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan dapat disimpulkan bahwa dalam pengujian kinerja ResNet-18 untuk verifikasi kekerabatan dengan menggunakan *optimizer*, jumlah *epoch*, dan *batch size* yang berbeda-beda, ditemukan bahwa model tersebut mampu mencapai akurasi *training* yang sangat tinggi, yaitu sebesar 0.9771, saat menggunakan *optimizer* RMSprop dengan *epoch* 30 dan *batch size* 25 Sehingga dapat disimpulkan bahwa ResNet-18 sangat bagus digunakan dalam pengolahan citra.. Hasil dari pengujian kinerja *real time* ResNet-18 untuk verifikasi kekerabatan dengan menggunakan berbagai *optimizer* dan ukuran *batch size* yang berbeda, ditemukan bahwa *optimizer* SGD berhasil pada ukuran *batch size* 10, 15, dan 25. Namun, pada pengujian di perangkat jetson nano, *optimizer* RMSprop gagal karena ukuran model yang terlalu besar membutuhkan lebih banyak memori daripada yang tersedia pada perangkat GPU. Dalam hal ini, ukuran model terlalu besar untuk dapat dijalankan pada perangkat dengan sumber daya terbatas seperti jetson nano.

DAFTAR RUJUKAN

Arunnehru, J., Chamundeeswari, G., & Bharathi, S. P. (2018). Human Action Recognition using 3D Convolutional Neural Networks with 3D Motion Cuboids in Surveillance Videos. *Procedia Computer Science*, *133*, 471–477. https://doi.org/10.1016/j.procs.2018.07.059

Chergui, A., Ouchtati, S., Mavromatis, S., Bekhouche, S. E., Lashab, M., & Sequeira, J. (2020). Kinship verification through facial images using CNN-based features. *Traitement Du Signal*, *37*(1), 1–8. https://doi.org/10.18280/ts.370101

Dibeklioglu, H. (2017). Visual Transformation Aided Contrastive Learning for Video-Based Kinship Verification. *Proceedings of the IEEE International Conference on Computer Vision 2017*, (pp. 2478–2487). https://doi.org/10.1109/ICCV.2017.269

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*. http://image-net.org/challenges/LSVRC/2015/

IIKKA TEIVAS. (2017). Video event classification using 3D convolutional neural networks. In *Master of science thesis in information technology, Tampere University of Technology* (Issue December). https://dspace.cc.tut.fi/dpub/handle/123456789/24703

Kim, B.-G. (2020). Real-time Facial Expression Recognition using 3D Appearance and Geometric Network for Public Security. *Journal of Defense Acquisition and Technology*, *2*(1), 33–37. https://doi.org/10.33530/jdaat.2020.2.1.33

Qin, X., Tan, X., & Chen, S. (2015). *Tri-Subject Kinship Verification: Understanding the Core of A Family*. http://arxiv.org/abs/1501.02555

Rachmadi, R. F., & Purnama, I. K. E. (2018). Paralel Spatial Pyramid Convolutional Neural Network untuk Verifikasi Kekerabatan berbasis Citra Wajah. Jurnal Teknologi Dan Sistem Komputer, *6*(4), 152–157. https://doi.org/10.14710/jtsiskom.6.4.2018.152-157

Sasono, M. A. H. (2022). Pengenalan Pola Gerak Tangan Fungsional Pada Bidang Pertanian Berbasis Electroencephalograph Untuk Kontrol Robot Tangan Dengan Kombinasi Metode Long Short-Term Memory Dan Stacked Autoencoder.

Sokop, S. J., Mamahit, D. J., Eng, M., Sompie, S. R. U. A., Mahasiswa,), & Pembimbing,). (2016). Trainer Periferal Antarmuka Berbasis Mikrokontroler Arduino Uno. *Jurnal Teknik Elektro Dan Komputer*, *5*(3),13–23. ttps://ejournal.unsrat.ac.id/index.php/elekdankom/article/view/11999

Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3D convolutional networks. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, (pp. 4489–4497). https://doi.org/10.1109/ICCV.2015.510

Vikram, K. & Padmavathi, S. (2017). Facial Parts Detection Using Viola. *International Conference on Advanced Computing and Communication Systems (ICACCS -2015)*, (pp. 1–4).

Viola, P., & Jones, M. (2018). R*apid Object Detection using a Boosted Cascade of Simple Feature*. *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing, ICECDS 2017*, (pp. 1193–1197). https://doi.org/10.1109/ICECDS.2017.8389630

Vu, D. Q., Le, N. T. H., & Wang, J.-C. (2021). *Self-Supervised Learning via multi-Transformation Classification for Action Recognition*. http://arxiv.org/abs/2102.10378

Wang, M., Feng, J., Shu, X., Jie, Z., & Tang, J. (2018). Photo to family tree: Deep kinship understanding for nuclear family photos. *ASMMC-MMAC 2018 - Proceedings of the Joint Workshop of the 4th Workshop on Affective Social Multimedia Computing and 1st Multi-Modal Affective Computing of Large-Scale Multimedia Data, Co-Located with MM 2018*, (pp. 41–46). https://doi.org/10.1145/3267935.3267936