

Implementasi Arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss* (Studi Kasus Klasifikasi Pose Yoga)

MUHAMMAD ICHWAN¹, ANNISA OLGA ZERLINDA²

^{1,2}Program Studi Informatika, Institut Teknologi Nasional Bandung
Email: ichwan@itenas.ac.id

Received 3 Agustus 2022 | *Revised* 9 September 2022 | *Accepted* 18 Desember 2022

ABSTRAK

Aplikasi computer vision dapat digunakan untuk mengurangi kecelakaan yang terjadi ketika melakukan yoga akibat postur tubuh saat melakukan yoga yang tidak tepat. Pada penelitian, dilakukan sebagai langkah awal dalam penentuan model klasifikasi yang digunakan pada aplikasi yang dapat melakukan koreksi postur tubuh saat yoga. Penelitian dilakukan dengan mengklasifikasikan 11 studi kasus pose yoga dengan mengimplementasikan arsitektur InceptionResnet-v2 dan Squared Hinge Loss. Pada hasil yang didapatkan, model dengan kinerja terbaik diperoleh pada learning rate 0.0001, epoch 200, scaling residual InceptionResnet blok A 0.15, blok B 0.1, dan blok C 0.2, serta blok InceptionResnet 5 iterasi, B 10 iterasi, dan C 5 iterasi dengan kinerja arsitektur berdasarkan hasil evaluasi performa model didapatkan 89.98% accuracy, 90.38% precision, 89.79% recall, 89.83% F1 score, and 99% specificity pada pengujian 888 data uji dengan 11 pose yoga berbeda. Rata-rata pengujian waktu klasifikasi 1.301s dan loss 0.9494.

Kata kunci: CNN, InceptionResnet-v2, Klasifikasi Citra, Squared Hinge Loss, Yoga

ABSTRACT

Computer vision applications can be used to reduce accidents caused by improper posture while doing yoga. In this study, it was carried out as a first step in determining the classification model used in applications that can make corrections to a body posture while doing yoga. The research was conducted to classify 11 case studies of yoga poses by implementing the InceptionResnet-v2 architecture and Squared Hinge Loss. In the results, the model with the best performance was obtained at a learning rate of 0.0001, epoch 200, scaling residual InceptionResnet block A 0.15, block B 0.1, and block C 0.2, and 5 iteration of InceptionResnet block A, 10 iterations of block B, and 5 iterations of block C. The results of the model performance evaluation obtained 89.98% accuracy, 90.38% precision, 89.79% recall, 89.83% F1 score, and 99% specificity in the test of 888 test data with 11 different yoga poses and 1.301s average testing time of the classification model and loss 0.9494.

Keywords: CNN, Image Classification, InceptionResnet-v2, Squared Hinge Loss, Yoga

1. PENDAHULUAN

Deep learning telah banyak digunakan untuk kasus klasifikasi citra, dimana model akan mengambil *input* dalam bentuk citra dan *output* yang dihasilkan berupa hasil prediksi atau klasifikasi dari citra *input*. Pada dasarnya algoritma *deep learning* menggunakan *neural network* untuk menentukan koneksi antara *input* dan juga *output* (Kumar & Sinha, 2020). *Convolutional Neural Network* (CNN) mencapai kinerja yang baik dalam melakukan klasifikasi citra diantara banyaknya algoritma *deep learning* (Dong et al., 2020).

Pada tahun 2017, para peneliti Google (Szegedy et al., 2017) mengusulkan Arsitektur *InceptionResnet-v2* dengan mengkombinasikan dua buah arsitektur yaitu *Inception-v3* dan *Resnet*. Pada modul *Resnet* memperkenalkan *residual connection* yang memungkinkan untuk melatih *neural network* yang lebih dalam dan memperoleh akurasi yang lebih tinggi, sedangkan pada blok *Inception* dapat memperoleh informasi lebih banyak dari berbagai skala *input* citra. Menurut (Peng et al., 2020) kombinasi antara blok *Inception* dan struktur *Resnet* dapat meningkatkan kinerja arsitektur.

Squared Hinge Loss merupakan salah satu jenis *loss function* yang digunakan untuk menghitung nilai kuadrat dari *Hinge Loss* pada pelatihan model klasifikasi. Pada *Hinge Loss* sendiri merupakan tipe lain dari *loss function* yang biasa digunakan sebagai alternatif dari *Cross Entropy* untuk permasalahan klasifikasi citra. Namun, performa *Hinge Loss* terkadang lebih baik dibandingkan *Cross Entropy* dan terkadang lebih buruk dibandingkan *Cross Entropy*, salah satu perbandingannya dari penelitian yang dilakukan oleh (Agarap, 2017) dan (Tang, 2013) percobaan dilakukan pada data MNIST dan Fashion MNIST yang sama namun didapatkan hasil lebih baik pada model dengan *Cross Entropy* pada Agarap (2017) dan *Squared Hinge* pada Tang (2013).

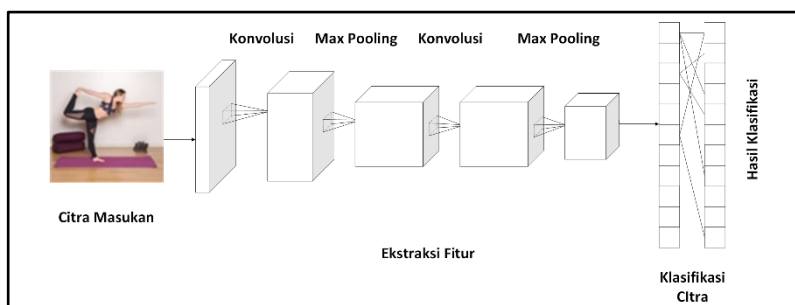
Postur tubuh saat melakukan yoga yang tidak tepat dapat menyebabkan kerusakan otot dan ligamen serta mengakibatkan cedera karena aktivitas seperti yoga terdiri dari pose yang beragam dan kompleks (Verma et al., 2020). Aplikasi *computer vision* dapat digunakan untuk mengklasifikasi pose yoga yang benar sehingga dapat mengurangi kecelakaan yang terjadi ketika melakukan yoga (Rishan et al., 2020). Namun, pada proses anotasi *keypoints* apabila dilakukan secara manual pada pembuatan *dataset* pose manusia, rentan terhadap kesalahan dan dapat dipengaruhi oleh berbagai faktor eksternal lain seperti resolusi, oklusi, iluminasi, sudut pandang pengambilan gambar, dan keragaman pose sehingga tingkat kesalahan pada saat melakukan klasifikasi akan semakin tinggi dan tingkat akurasi yang dihasilkan semakin rendah (Verma et al., 2020).

Penelitian yang dilakukan menjadi langkah awal dalam penentuan model klasifikasi yang digunakan pada aplikasi yang dapat melakukan koreksi pada postur tubuh saat melakukan yoga. Pada penelitian menggunakan Yoga-82 *Dataset* (Verma et al., 2020) sebagai *dataset* yang digunakan untuk melatih arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss* dalam melakukan klasifikasi pose yoga. Dengan menerapkan arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss* pada studi kasus klasifikasi pose yoga diharapkan didapatkan metode terbaik untuk melakukan klasifikasi pose yoga yang memiliki keberagaman pose dan gerakan yang kompleks berdasarkan tingkat akurasi, presisi, *recall*, *specificity* dan *F-Measure* yang diperoleh pada saat melakukan evaluasi model serta nilai *loss* dan rata-rata waktu eksekusi pada saat melakukan pengujian.

2. METODE PENELITIAN

2.1. Deep Learning

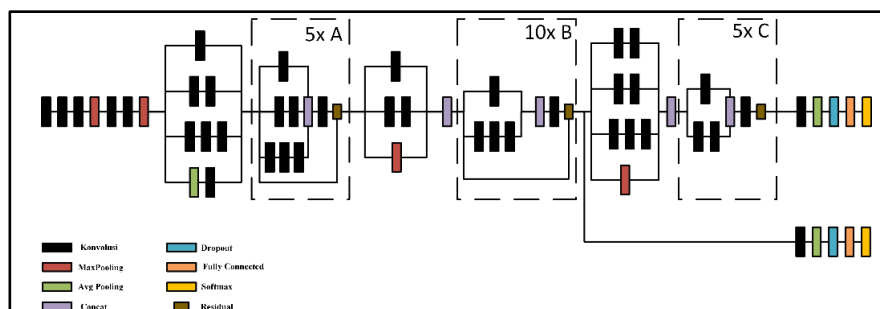
Deep learning memanfaatkan *input layer*, *hidden layer*, dan *output layer* untuk pengolahan informasi dalam melakukan analisis atau klasifikasi terhadap suatu pola. CNN yang merupakan bagian dari *deep learning* dapat meningkatkan kinerja pemrosesan gambar seperti deteksi objek dan pengklasifikasian objek. Arsitektur CNN terdiri dari lapisan *input*, lapisan konvolusi, *sublayer*, dan lapisan non linear. Lapisan selanjutnya diikuti oleh *fully connected layers*, dan pada lapisan akhir banyak digunakan *softmax classification* (Nguyen Quoc & Truong Hoang, 2020). Ilustrasi arsitektur CNN ditunjukkan pada Gambar 1.



Gambar 1. Ilustrasi Arsitektur CNN

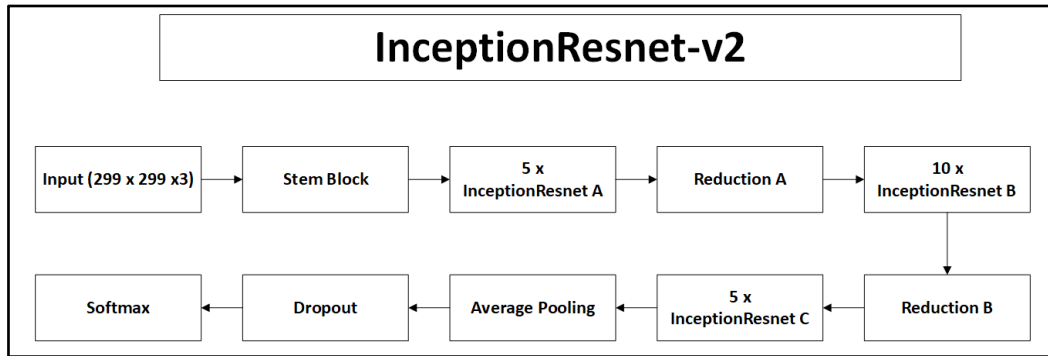
2.2. InceptionResNet-v2

Pada tahun 2016, para peneliti dari Google (Szegedy et al., 2017) mengusulkan kombinasi dari dua buah arsitektur baru yaitu *Inception-v3* dan *Residual Connection*. Dengan melakukan *training* pada *layer* yang dalam menggunakan *Residual Connection* dapat meningkatkan kecepatan yang dibutuhkan untuk melatih model awal (Al-masni et al., 2020). Ilustrasi diagram dari arsitektur *InceptionResnet-v2* ditunjukkan pada Gambar 2.



Gambar 2. Diagram Arsitektur *InceptionResnet-V2*

Pada blok *InceptionResnet-v2*, beberapa *filter* konvolusi digabungkan dengan *residual connection*. Pada Gambar 3 menunjukkan masing-masing blok *InceptionResnet-v2* secara umum yang terdiri dari Stem Blok, *InceptionResnet A*, *Reduction A*, *InceptionResnet B*, *Reduction B*, dan *InceptionResnet C*. Pada setiap blok terdapat proses konvolusi, *Batch Normalization*, *ReLU Activation*, *Residual Connection*, *Scaling* dari *Residual Connection* dan *Pooling*. Pada tahap klasifikasi terdapat *global average pooling* kemudian diikuti dengan *softmax activation*.



Gambar 3. Blok Pada Arsitektur *InceptionResnet-v2*

Arsitektur *InceptionResnet-v2* merupakan kombinasi dari blok *Inception* dan struktur *ResNet*. Pada modul *ResNet* memperkenalkan *residual connection* yang memungkinkan untuk melatih *neural network* di *layer* yang lebih dalam. Blok *Inception* dapat memperoleh informasi lebih banyak dari berbagai skala *input* gambar dan *ResNet* dapat memperoleh akurasi yang tinggi pada kedalaman yang lebih dalam. Sehingga kombinasi antara blok *Inception* dan struktur *ResNet* dapat meningkatkan kinerja arsitektur (Peng et al., 2020). Pada Tabel 1 menunjukkan arsitektur *InceptionResnet-V2* dengan masing-masing *output* pada setiap bloknya.

Tabel 1. Arsitektur *InceptionResnet-v2* (Szegedy et al., 2017)

Type	Output Size
Input (299 x 299 x 3)	-
Stem Block	35x35x256
5 x Inception-resnet-A	35x35x256
Reduction-A	17x17x896
10 x Inception-resnet-B	17x17x896
Reduction-B	8x8x1792
5 x Inception-resnet-C	8x8x1792
Average Pooling	1792
Dropout	1792
Softmax	1000

2.2.1 Residual Connection

Residual Connections dapat meneruskan informasi *input* secara langsung ke lapisan berikutnya. Pada *Residual Connections* terdapat fitur *Skip Connection* seperti yang digunakan pada *Residual Network (ResNet)* (He et al., 2016). Pada operasi yang dilakukan menggunakan *shortcut connection* dan *element wise addition*. Pada Persamaan (1) menunjukkan persamaan *residual connection*.

$$y = F(x, \{W_i\}) + x \quad (1)$$

Keterangan

- x : matriks *input*
- y : matriks *output*
- F(x, {W_i}) : representasi pemetaan residual

2.2.2 *Scaling Residual*

Scaling Residual digunakan untuk menstabilkan pelatihan saat ditempatkan sebelum aktivasi *layer* karena pada penelitian (**Szegedy et al., 2017**) ketika jumlah *filter* melebihi 1000, varian residual mulai menunjukkan ketidakstabilan dan jaringan baru saja "mati" di awal *training*. Hal tersebut tidak dapat dicegah baik dengan menurunkan *learning rate* maupun menambahkan *batch normalization* pada *layer*. *Scaling residual* yang digunakan diantara 0.1 hingga 0.3. Pada Persamaan (2) menunjukkan persamaan *Scaling of Residual*.

$$y = (F(x, \{W_i\}) \times scale) + x \quad (2)$$

Keterangan:

scale : ukuran skala residual

2.3 *Squared Hinge Loss*

Fungsi yang menggambarkan kerugian terkait dengan semua kemungkinan yang dihasilkan oleh model *deep learning* biasa disebut dengan *Loss Function* atau *Cost Function*. *Hinge Loss* merupakan tipe lain dari *loss function* yang biasa digunakan sebagai alternatif dari *Cross Entropy* untuk permasalahan klasifikasi citra. Performa *Hinge Loss* terkadang lebih baik dibandingkan *Cross Entropy* dan terkadang lebih buruk dibandingkan *Cross Entropy*.

Squared Hinge Loss menghitung kuadrat dari *score hinge loss* sehingga memberikan efek penghalusan pada permukaan fungsi *loss* serta memudahkan untuk dikerjakan secara numerik. *Squared Hinge Loss* memastikan saat dilakukan proses *training*, *classifier* akan menemukan batasan terjauh antara satu kelas dengan kelas lainnya. Dengan kata lain, akan ditemukan margin maksimum antara poin data dari kelas yang berbeda. *Squared Hinge Loss* memiliki efek *smoothing* pada permukaan *error function* yang dihasilkan. Menurut (**Agarap, 2017**) dan (**Tang, 2013**) dengan menggunakan *Squared Hinge Loss* pada model *deep learning* dikatakan bahwa model tersebut memiliki *layer SVM*. Pada Persamaan (3) menunjukkan persamaan *Squared Hinge Loss*.

$$L(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^n (\max(0, 1 - y_i \hat{y}_i))^2 \quad (3)$$

Keterangan:

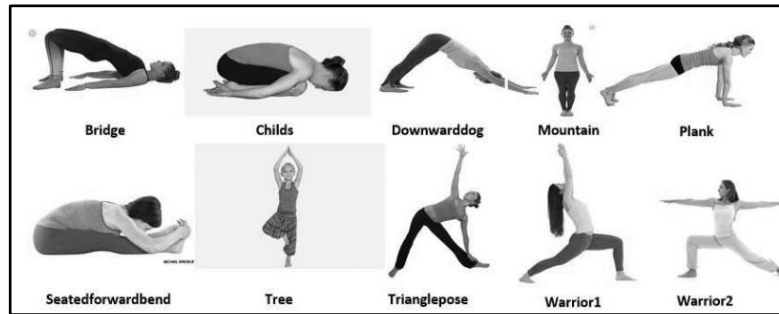
L : nilai loss

\hat{y}_i : *output* klasifikasi berupa nilai aktual numerik

y_i : *output* untuk y ke i

2.4 Yoga

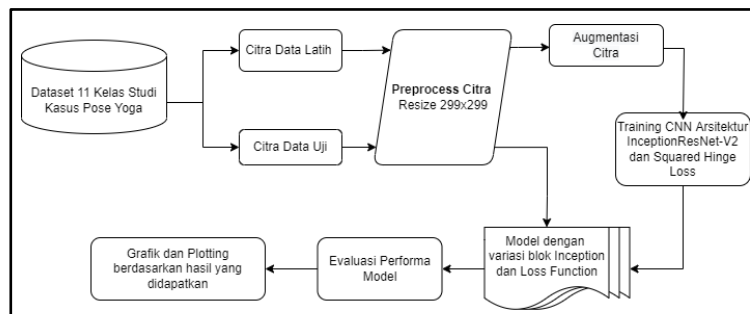
Yoga merupakan latihan fisik dan spiritual yang memiliki kemampuan untuk mengobati penyakit, meningkatkan kesehatan baik fisik maupun mental (**Kumar & Sinha, 2020**). Tanpa didampingi oleh instruktur yoga dapat terjadi kesalahan dalam melakukan pose yoga yang dapat mengakibatkan cedera. Untuk mencegah terjadinya cedera dibutuhkan sebuah teknologi untuk membantu pelatihan pose yoga (**Jose & Shailesh, 2021**). Pada Gambar 4 menampilkan citra pose manusia pada saat melakukan aktivitas yoga yang terdiri dari beberapa pose beragam.



Gambar 4. Pose Yoga Asana (Jose & Shailesh, 2021)

2.5 Arsitektur Sistem

Pada proses perancangan sistem klasifikasi pose yoga menggunakan *InceptionResNet-v2* dan *Squared Hinge Loss*, terdapat beberapa tahapan yang dilakukan untuk membuat sebuah model pada penelitian ini Pada Gambar 5 merupakan *framework* sistem penelitian.



Gambar 5. *Framework* Sistem

Input berupa citra yang berasal dari Yoga-82 *Dataset* (Verma et al., 2020). Dilakukan *pre-processing* pada citra masukan dengan dilakukannya *resize* menjadi 299 x 299 dengan 3 buah *channel* warna RGB (*Red*, *Green*, dan *Blue*). Pada proses *pre-processing* ini menyesuaikan dengan format citra model *InceptionResNet-v2*. Kemudian dilakukan augmentasi citra untuk menambahkan variasi pada *dataset*. Proses selanjutnya menggunakan *Convolutional Neural Network* arsitektur *InceptionResNet-v2* dengan operasi umum seperti konvolusi, reduksi, *pooling*, dan modul blok *InceptionResnet* serta penggunaan *loss function Squared Hinge Loss*. *Output* yang dihasilkan berupa hasil pengklasifikasian pose yoga. Setelah pengujian dilakukan selanjutnya evaluasi model menggunakan *confusion matrix* lalu dilakukan *plotting* pada hasil yang didapatkan untuk mengetahui performa model yang dihasilkan.

2.6 Metrik Evaluasi

Evaluasi kinerja model klasifikasi yang telah dirancang dilakukan dengan menghitung tingkat akurasi, *specificity* dan *F-Measure* dari evaluasi *confusion matrix* (Agarwal et al., 2020) sebagai pengujian performa model dari proses pengujian *dataset*. Pada *confusion matrix* terdapat 4 istilah sebagai berikut :

- TP (*True Positive*) : perkiraan banyaknya data positif dan terbukti kebenarannya.
- FP (*False Positive*) : perkiraan banyaknya data negatif dan terbukti kebenarannya.
- TN (*True Negative*) : perkiraan banyaknya data positif dan tidak terbukti kebenarannya.
- FN (*False Negative*) : perkiraan banyaknya data negatif dan tidak terbukti kebenarannya.

Accuracy : Rasio prediksi benar dengan total keseluruhan data, akurasi merupakan kriteria utama dalam evaluasi model. Perhitungan akurasi menggunakan Persamaan (4).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (4)$$

Precision : Bagian dari suatu objek diprediksi benar. Perhitungan *precision* menggunakan Persamaan (5)

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

Recall : Mengetahui seberapa akurat kinerja suatu model untuk mengklasifikasi benar atau *menghitung* banyaknya kekeliruan klasifikasi *false negative* pada model. Perhitungan *recall* menggunakan Persamaan (6)

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

F-Score : Perbandingan rata – rata harmonik antara nilai *precision* dan nilai *recall*. Perhitungan F1 Score menggunakan Persamaan (7)

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (7)$$

Specificity : Mengidentifikasi kemampuan untuk mengklasifikasikan sampel sebagai benar-benar negatif didasarkan pada tingkat negatif sebenarnya. Perhitungan *specificity* menggunakan Persamaan (8)

$$Specificity = \frac{TN}{TN + FP} \times 100 \quad (8)$$

3. HASIL DAN PEMBAHASAN

3.1 Dataset

Pada *dataset* yang digunakan merupakan Yoga-82 *Dataset* yang dibuat (Verma et al., 2020). Citra diambil dari beberapa sudut pandang yang berbeda. Total citra pada *dataset* ini lebih dari 28400 citra. Terdapat 82 kelas pada *dataset*, jumlah citra pada setiap kelas bervariasi mulai dari 64 hingga 1133 citra dengan rata-rata 347 citra perkelasnya. Pada *dataset* yang digunakan diambil 11 kelas dengan jumlah data minimal 359 hingga maksimal 482 untuk setiap kelasnya. Pada Tabel 2 menunjukkan penggunaan *dataset* pada penelitian.

Tabel 2. Penggunaan Dataset

No	Kelas	Data		
		Latih	Uji	Jumlah
1	<i>Boat Pose</i> atau <i>Paripurna Navasana</i>	350	88	438
2	<i>Cat Cow Pose</i> atau <i>Marjaryasana</i>	312	78	390
3	<i>Child Pose</i> atau <i>Balasana</i>	327	82	409
4	<i>Corpse Pose</i> atau <i>Savasana</i>	318	79	397
5	<i>Cow Face Pose</i> atau <i>Gomukhasana</i>	308	77	385
6	<i>Extended Revolved Triangle Pose</i> atau <i>Utthita Trikonasana</i>	386	96	482
7	<i>Lord of the Dance Pose</i> atau <i>Natarajasana</i>	365	91	456
8	<i>Seated Forward Bend pose</i> atau <i>Paschimottanasana</i>	293	73	366
9	<i>Side Plank Pose</i> atau <i>Vasisthasana</i>	310	76	386
10	<i>Supta Virasana Vajrasana</i>	287	72	359
11	<i>Upward Bow (Wheel) Pose</i> atau <i>Urdhva Dhanurasana</i>	304	76	380
Total		3560	888	4448

3.2 Pemrosesan Awal *Dataset*

Citra pada *dataset* yang digunakan dilakukan pemrosesan awal dengan melakukan *resize* menjadi 299 x 299 piksel. Kemudian untuk menambahkan variasi pada *dataset* dilakukan augmentasi citra dengan *horizontal flip* dan *vertical flip*. Gambar 6 menunjukkan sampel citra yang diambil dari *dataset* yang digunakan.



Gambar 6. Sampel Citra *Dataset*

3.3 Pelatihan Model

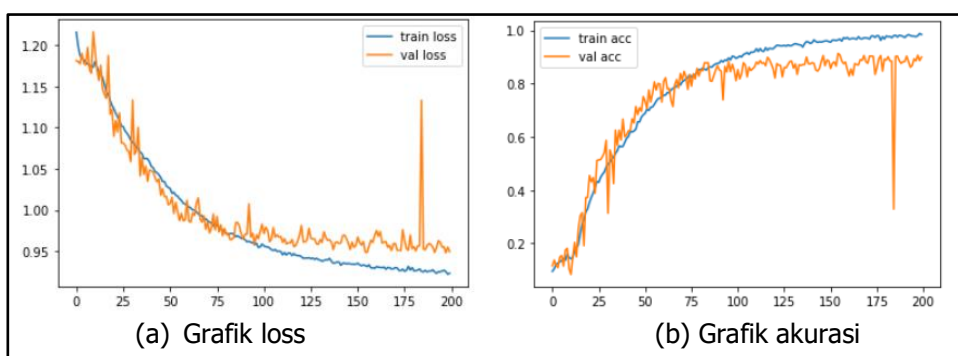
Pelatihan model dilakukan menggunakan *Google Colaboratory* dengan menggunakan NVIDIA-SMI 460.32.03 dengan GPU NVIDIA Tesla P100 PCIe dan ukuran RAM 16GB. Pada penelitian dilakukan pelatihan pada 9 variasi model (Tabel 3) berdasarkan jumlah *epoch* dan *learning rate* serta 11 model (Tabel 4) variasi *scaling residual* dengan menggunakan *epoch* dan *learning rate* pada model dengan akurasi tertinggi berdasarkan variasi model sebelumnya. Pada proses pelatihan dihasilkan model terbaik pada *learning rate* 0.0001, *epoch* 200, dengan ukuran *batch size* 10, *scaling residual InceptionResnet* blok A sebesar 0.15, *scaling residual InceptionResnet* blok B sebesar 0.1, dan *scaling residual InceptionResnet* blok C sebesar 0.2 pada saat pelatihan dilakukan selama 8 jam. Pada Gambar 7 (a) menunjukkan hasil *loss* dan (b) hasil akurasi saat melakukan pelatihan pada model terbaik diantara 9 variasi *learning rate* dan *epoch* dan 11 model variasi *scaling residual* menggunakan arsitektur *InceptionResnet-V2*.

Tabel 3. Variasi Model Pelatihan Berdasarkan *Epoch* dan *Learning Rate*

No	Model	Epoch	Learning Rate
1	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	50	0.01
2	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	50	0.001
3	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	50	0.0001
4	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	100	0.01
5	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	100	0.001
6	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	100	0.0001
7	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	200	0.01
8	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	200	0.001
9	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	200	0.0001

Tabel 4. Variasi Model Pelatihan Berdasarkan *Scaling Residual*

No	Epoch	Learning Rate	<i>Scaling Residual</i>		
			<i>Inception Resnet A</i>	<i>Inception Resnet B</i>	<i>Inception Resnet C</i>
1	200	0.0001	0.2	0.25	0.3
2	200	0.0001	0.15	0.2	0.25
3	200	0.0001	0.1	0.15	0.2
4	200	0.0001	0.1	0.1	0.1
5	200	0.0001	0.15	0.15	0.15
6	200	0.0001	0.2	0.2	0.2
7	200	0.0001	0.1	0.15	0.1
8	200	0.0001	0.2	0.15	0.2
9	200	0.0001	0.3	0.2	0.1
10	200	0.0001	0.15	0.1	0.2
11	200	0.0001	1	1	1



Gambar 7. Grafik *Loss* dan *Accuracy*

3.4 Skenario Pengujian

Pengujian sistem dalam melakukan klasifikasi pose yoga yang terdapat pada citra dengan menggunakan metode *convolutional neural network* arsitektur *InceptionResnet-V2*. Pengujian dilakukan dengan membandingkan beberapa model pelatihan, yang selanjutnya dilakukan pengujian akurasi menggunakan data uji. Pengujian model yang didapat tersebut memiliki parameter jumlah blok *InceptionResnet* yang bervariasi digunakan pada pelatihan model terhadap 888 citra uji. Dengan konfigurasi model pelatihan awal ditunjukkan pada Tabel 5. Berikut merupakan variasi blok *InceptionResnet* dari model 1 sampai model 5 dengan skenario pengujian yang ditunjukkan pada Tabel 6.

Tabel 5. Konfigurasi Model Pengujian

No	Konfigurasi	Pengaturan
1	GPU	NVIDIA Tesla P100 PCIe 16 GB
2	<i>Batch Size</i>	10
3	<i>Loss Function</i>	<i>Squared Hinge Loss</i>
4	<i>Optimizer</i>	<i>Adaptive Moment Estimation (ADAM)</i>
5	<i>Epoch</i>	200
6	<i>Learning Rate</i>	0.0001
7	<i>Scaling Residual Block A</i>	0.15
8	<i>Scaling Residual Block B</i>	0.1
9	<i>Scaling Residual Block C</i>	0.2

Tabel 6. Skenario Pengujian Berdasarkan Nilai Blok *InceptionResnet*

No	Model	<i>InceptionResnet</i>		
		Blok A	Blok B	Blok C
1	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	3	5	2
2	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	4	8	4
3	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	6	10	4
4	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	5	10	5
5	<i>InceptionResnet-v2 + Squared Hinge Loss</i>	10	20	10

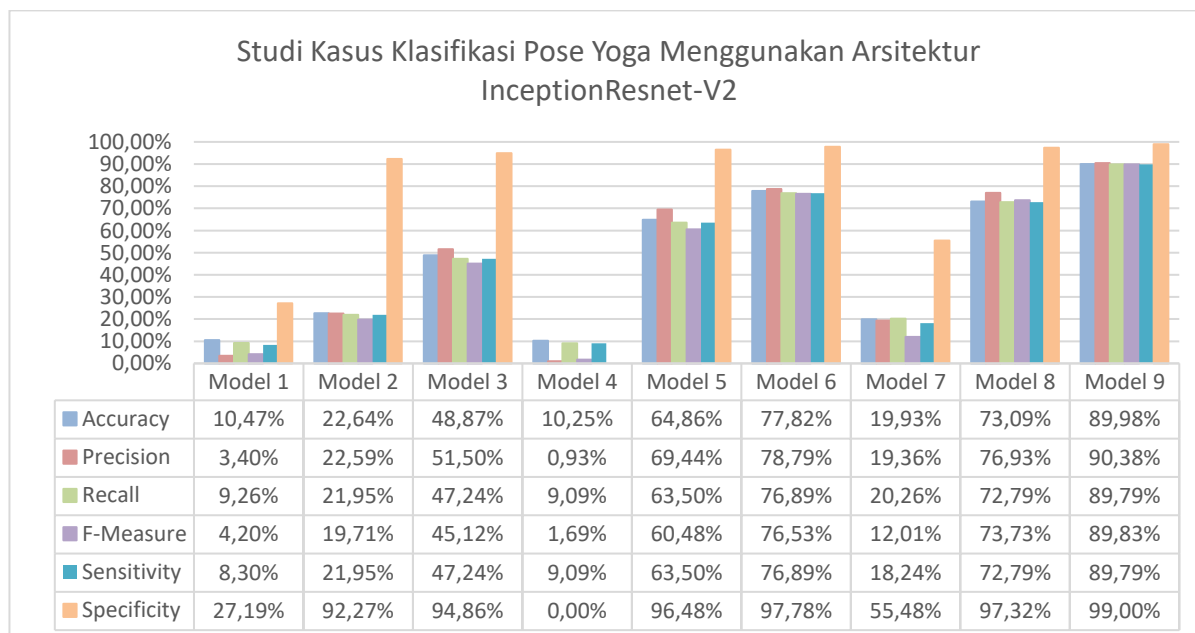
Dilakukan proses pengujian dengan membandingkan penggunaan *scaling residual* pada masing masing blok *InceptionResnet* berdasarkan 2 variasi *loss function* yang digunakan. Pada Tabel 7 menunjukkan variasi pengujian *loss function* yang digunakan pada penelitian. Pengujian dilakukan terhadap 888 citra uji pada model dengan nilai akurasi tertinggi saat pengujian blok *InceptionResnet* yang dilakukan sebelumnya.

Tabel 7. Skenario Pengujian Berdasarkan *Loss Function*

Model	Arsitektur	<i>Loss Function</i>
1	<i>InceptionResnet-v2</i>	<i>Squared Hinge Loss</i>
2	<i>InceptionResnet-v2</i>	<i>Cross Entropy Loss</i>

3.5 Hasil Pengujian Sistem

Skema pelatihan dengan mengukur kinerja dari sistem yang dibuat menggunakan *confusion matrix* untuk menghitung tingkat *accuracy*, *precision*, *recall*, *specificity* dan *F-Measure*. Pada Gambar 8 menunjukkan grafik perbandingan kinerja antara 9 variasi model yang telah dirancang dan diuji pada 888 citra uji pose yoga. Didapatkan akurasi tertinggi pada model 9 (*epoch* = 200, dan *learning rate* = 0.0001) dengan 89.98% *accuracy*, 90.38% *precision*, 89.79% *recall*, 89.83% *F1 score*, dan 99% *specificity*.



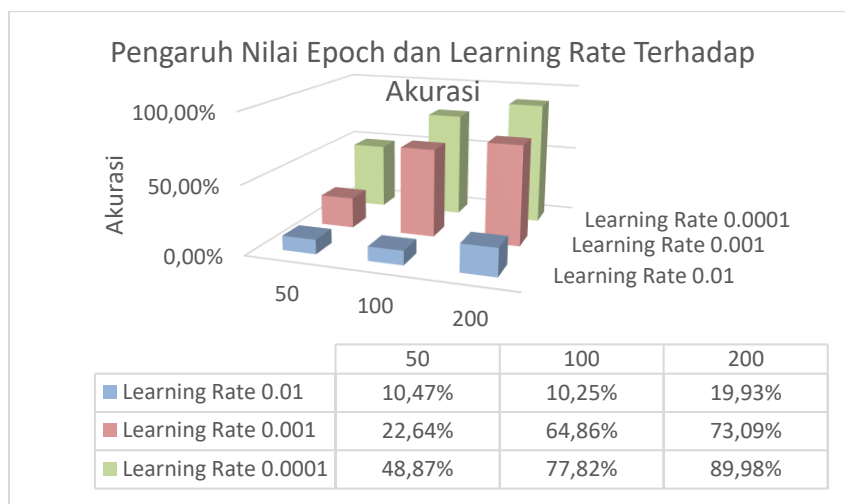
Gambar 8. Grafik Perbandingan Kinerja Model

Epoch yang dipilih pada saat melakukan pelatihan model akan berpengaruh pada akurasi yang dihasilkan, Gambar 9 menunjukkan pengaruh jumlah *epoch* yang digunakan terhadap akurasi

Implementasi Arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss*
(Studi Kasus Klasifikasi Pose Yoga)

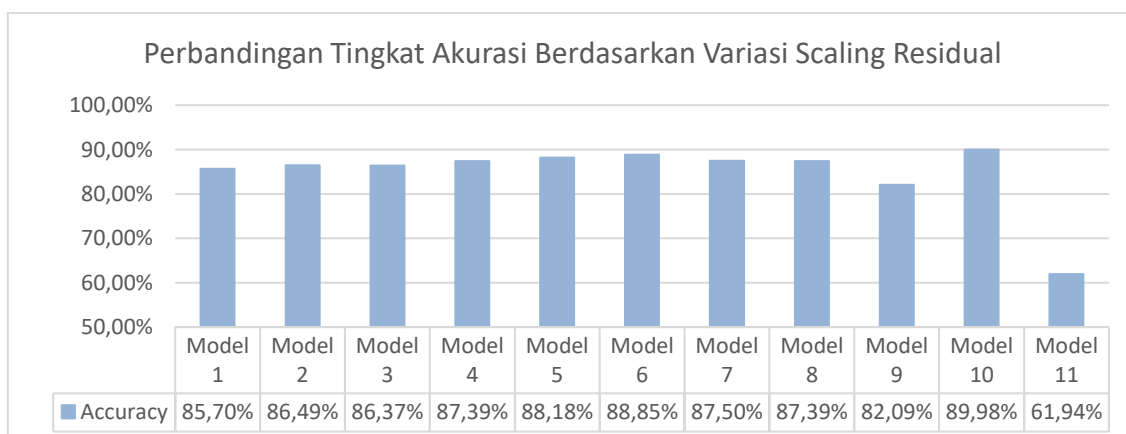
yang dihasilkan. Peningkatan akurasi terus terjadi pada pada setiap *epoch* dengan *learning rate* berbeda tiap modelnya. Pada *learning rate* 0.001 dan 0.0001 mulai dari *epoch* ke 50, 100, lalu 200 mengalami peningkatan yang signifikan.

Learning rate berpengaruh terhadap akurasi yang dihasilkan dapat dilihat pada Gambar 9 menampilkan kenaikan tingkat akurasi yang signifikan pada saat *learning rate* diperkecil baik pada *epoch* 50, 100, maupun 200. Namun, pada saat melakukan pelatihan model, semakin kecil *learning rate* yang digunakan akan semakin lama waktu yang dibutuhkan untuk melakukan proses pelatihan model tersebut.



Gambar 9. Grafik Pengaruh Nilai *Epoch* dan *Learning Rate* Terhadap Akurasi

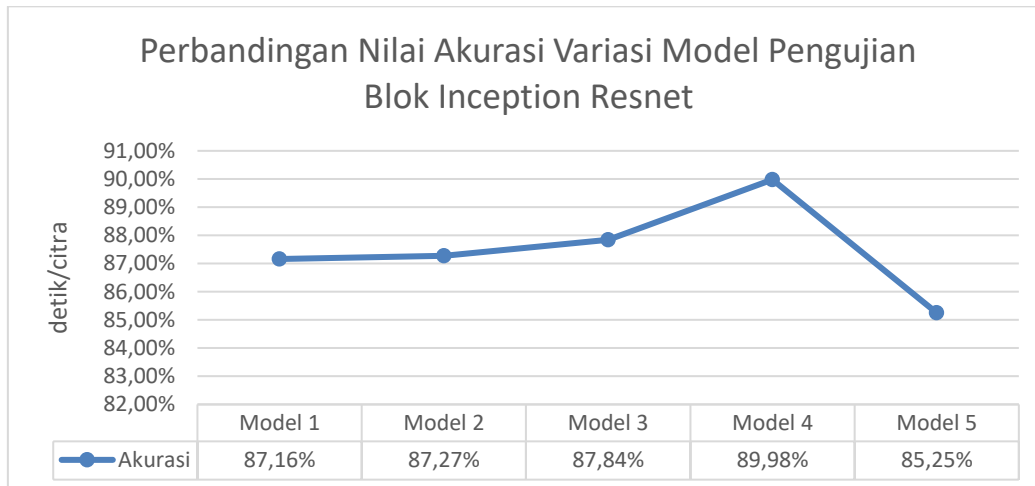
Dilanjutkan dengan skema pengujian model berdasarkan 11 variasi *scaling residual* dan diuji pada 888 citra uji pose yoga. Pada hasil pengujian didapatkan akurasi tertinggi pada model 10 *scaling residual InceptionResnet* blok A sebesar 0.15, *scaling residual InceptionResnet* blok B sebesar 0.1, dan *scaling residual InceptionResnet* blok C sebesar 0.2 dengan nilai akurasi 89.98%. Perbandingan tingkat akurasi berdasarkan variasi model pelatihan *scaling residual* dapat dilihat pada Gambar 10.



Gambar 10. Perbandingan Tingkat Akurasi Berdasarkan Variasi *Scaling Residual*

Pada hasil perbandingan model pelatihan berdasarkan variasi *scaling residual* (Gambar 10) diketahui bahwa tidak ada pengaruh signifikan mengenai besar atau kecilnya *scaling residual* terhadap tingkat akurasi, hanya dibutuhkan penentuan *scaling residual* yang tepat untuk

menghasilkan tingkat akurasi yang semakin tinggi. Namun, jika *neural network* tidak menggunakan *scaling residual* pada lapisannya maka akan menghasilkan nilai akurasi yang rendah dibandingkan dengan *neural network* yang menggunakan *scaling residual*. Maka dari itu dapat dikatakan bahwa penggunaan *scaling residual* pada arsitektur *InceptionResnet-v2* digunakan untuk menstabilkan jaringan agar akurasi yang dihasilkan tidak menurun.



Gambar 11. Grafik Perbandingan Nilai Akurasi Variasi Model Pengujian Blok *InceptionResnet*

Dilakukan proses pengujian membandingkan 5 model berdasarkan variasi dari Blok *InceptionResnet*. Pada skenario pengujian yang telah dilakukan, model dengan jumlah parameter paling kecil berada pada Model 1 dengan 18.945.467, sedangkan nilai akurasi, presisi, *recall*, *F1 Score*, dan *specificity* paling tinggi didapatkan oleh Model 4 dengan 89.98%, presisi 90.38%, *recall* 89.79%, *F1 Score* 89.83%, dan *specificity* 99.00%. Lalu dari pengujian kecepatan didapatkan Model 1 dengan 0.888 citra/detik. Gambar 11 dalam bentuk grafik untuk memudahkan pengamatan pengaruh blok *InceptionResnet* terhadap nilai akurasi model pengklasifikasian.

Pada akurasi yang dihasilkan dari setiap model yang dirancang pada pengujian menghasilkan akurasi tertinggi pada Model 4 dengan jumlah 5 Blok *InceptionResnet A*, 10 Blok *InceptionResnet B*, dan 5 Blok *InceptionResnet C*. Pada Model 1, 2, 3, dan 4 mengalami peningkatan nilai akurasi yang signifikan artinya semakin banyak lapisan yang digunakan, jumlah parameter yang dihasilkan semakin banyak dan tingkat akurasi semakin meningkat. Namun, pada model 5 mengalami penurunan tingkat akurasi, dibutuhkan pengaturan ulang nilai *scaling residual* sehingga jaringan akan kembali stabil. Pada Tabel 8 menunjukkan hasil pengujian berdasarkan variasi blok *InceptionResnet*.

Tabel 8. Hasil Pengujian 1 Berdasarkan Blok *InceptionResnet*

Model	Acc	Prs	Rcll	F1 Scr	Spf	JP	RK
1	87.16%	87.94%	86.68%	86.98%	98.71%	18.945.467	0.888168
2	87.27%	87.69%	87.18%	87.18%	98.73%	28.209.003	1.147238
3	87.84%	90.27%	87.35%	88.04%	98.78%	30.835.915	1.271491
4	89.98%	90.38%	89.79%	89.83%	99.00%	33.497.499	1.300950
5	85.25%	87.55%	85.00%	85.31%	98.53%	59.939.979	2.203468

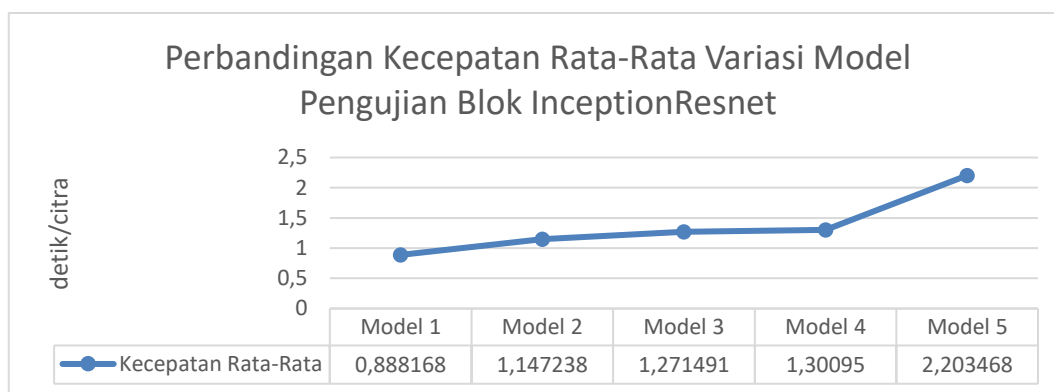
Keterangan :

Acc : *accuracy*

Implementasi Arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss*
(Studi Kasus Klasifikasi Pose Yoga)

- Prccs : *precision*
- Rcll : *recall*
- F1Scr : *F1 Score*
- Spcf : *specificity*
- JP : jumlah parameter
- RK : rata-rata kecepatan (detik/citra)

Berdasarkan pengamatan dari pengujian yang dilakukan, semakin sedikit jumlah blok *InceptionResnet* yang digunakan maka akan semakin sedikit jumlah parameter yang dihasilkan begitu pula sebaliknya. Selain itu, semakin sedikit jumlah parameter yang digunakan maka akan semakin cepat proses klasifikasi citra yang dihasilkan. Dapat dibuktikan pada Model 1 memiliki jumlah parameter paling kecil 18.945.467 dan kecepatan klasifikasi rata rata tercepat dengan 0.888168 detik/citra. Gambar 12 dalam bentuk grafik untuk memudahkan pengamatan pengaruh blok *InceptionResnet* terhadap kecepatan pengklasifikasian.



Gambar 12. Grafik Perbandingan Kecepatan Rata-Rata Berdasarkan Variasi Pengujian Blok *InceptionResnet*

Dilakukan pula pengujian dengan membandingkan beberapa model berdasarkan variasi dari *Loss Function* yang digunakan. Pada skenario pengujian, model dengan nilai *loss* pada pengujian 888 data uji paling kecil berada pada Model 1 dengan nilai *loss* 0.9494 dan akurasi 89.98%. Pada *loss function* digunakan oleh model untuk belajar sedangkan pada akurasi mengukur seberapa baik model memprediksi dengan membandingkan prediksi model dengan nilai sebenarnya dalam hal persentase. Berdasarkan pengamatan yang dilakukan, nilai *loss* yang dihasilkan menggunakan *Squared Hinge Loss* pada pengujian didapatkan lebih kecil dibandingkan dengan menggunakan *Cross Entropy*. Pada Tabel 9 menunjukkan hasil pengujian berdasarkan *loss function*.

Tabel 9. Hasil Pengujian 2 Berdasarkan *Loss Function*

No. Model	<i>Precision</i>	<i>Recall</i>	<i>F1 Score</i>	<i>Specificity</i>	<i>Accuracy</i>	<i>Loss</i>
1	90.38%	89.79%	89.83%	99.00%	89.98%	0.9494
2	85.56%	84.80%	84.83%	98.54%	85.02%	2.1988

4. KESIMPULAN

Kinerja model yang memiliki akurasi tertinggi merupakan model dengan arsitektur *InceptionResnet-v2* dan *Squared Hinge Loss* dengan *epoch* 200, *learning rate* 0.0001, *scaling residual* blok A 0.15, blok B 0.1, blok C 0.2. Dengan jumlah parameter 33.497.499,

menghasilkan akurasi sebesar 89.98%, presisi 90.38%, *recall* 89.79%, *F1 Score* 89.83%, *specificity* 99.00%, dan nilai *loss* pada pengujian 888 data uji sebesar 0.9494. Kecepatan rata-rata pengklasifikasian citra yang dilakukan pada total 30 citra adalah 1.300950 detik/citra. Berdasarkan eksperimen yang telah dilakukan, penentuan parameter seperti *epoch* dan *learning rate* yang digunakan pada model pelatihan perlu dilakukan dengan percobaan untuk mendapatkan model dengan akurasi yang tinggi dan *learning curve* yang baik. Pada model yang tidak menggunakan *scaling residual* pada proses pelatihannya akan mendapatkan akurasi yang rendah jika dibandingkan dengan model yang menggunakan *scaling residual*. Sedangkan pada penggunaan blok *InceptionResnet*, semakin banyak blok yang digunakan akan semakin baik tingkat akurasi dari model pelatihan yang dihasilkan. Pada penggunaan *squared hinge loss*, nilai *loss* pada model didapatkan lebih baik jika dibandingkan dengan *cross entropy* pada pengujian menggunakan data uji klasifikasi citra studi kasus pose yoga.

DAFTAR RUJUKAN

- Agarap, A. F. (2017). *An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification*. 5–8. <http://arxiv.org/abs/1712.03541>
- Agarwal, M., Gupta, S. K., & Biswas, K. K. (2020). Development of Efficient CNN model for Tomato crop disease identification. *Sustainable Computing: Informatics and Systems*, 28, 100407. <https://doi.org/10.1016/j.suscom.2020.100407>
- Al-masni, M. A., Kim, D. H., & Kim, T. S. (2020). Multiple skin lesions diagnostics via integrated deep convolutional networks for segmentation and classification. *Computer Methods and Programs in Biomedicine*, 190, 105351. <https://doi.org/10.1016/j.cmpb.2020.105351>
- Dong, N., Zhao, L., Wu, C. H., & Chang, J. F. (2020). Inception v3 based cervical cell classification combined with artificially extracted features. *Applied Soft Computing*, 93, 106311. <https://doi.org/10.1016/j.asoc.2020.106311>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- Jose, J., & Shailesh, S. (2021). Yoga Asana Identification: A Deep Learning Approach. *IOP Conference Series: Materials Science and Engineering*, 1110(1), 012002. <https://doi.org/10.1088/1757-899x/1110/1/012002>
- Kumar, D., & Sinha, A. (2020). Yoga Pose Detection and Classification Using Deep Learning. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 160–184. <https://doi.org/10.32628/cseit206623>
- Nguyen Quoc, T., & Truong Hoang, V. (2020). Medicinal Plant identification in the wild by using CNN. *International Conference on ICT Convergence, 2020-Octob*, 25–29. <https://doi.org/10.1109/ICTC49870.2020.9289480>

Implementasi Arsitektur *InceptionResNet-v2* dan *Squared Hinge Loss*
(Studi Kasus Klasifikasi Pose Yoga)

- Peng, S., Huang, H., Chen, W., Zhang, L., & Fang, W. (2020). More trainable inception-ResNet for face recognition. *Neurocomputing*, *411*, 9–19. <https://doi.org/10.1016/j.neucom.2020.05.022>
- Rishan, F., Lanka, S., Nijabdeen, S., Lanka, S., Silva, B. De, Lanka, S., Rupasinghe, L., Lanka, S., Alawathugoda, S., Lanka, S., Liyanapathirana, C., & Lanka, S. (2020). *Infinity Yoga Tutor: Yoga Posture*.
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-ResNet and the impact of residual connections on learning. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 4278–4284.
- Tang, Y. (2013). *Deep Learning using Linear Support Vector Machines*. <http://arxiv.org/abs/1306.0239>
- Verma, M., Kumawat, S., Nakashima, Y., & Raman, S. (2020). Yoga-82: A new dataset for fine-grained classification of human poses. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2020-June*, 4472–4479. <https://doi.org/10.1109/CVPRW50498.2020.00527>