**METODE *TRANSFER LEARNING* UNTUK KLASIFIKASI CITRA HURUF TULIS TANGAN AKSARA JAWA**

**SKRIPSI**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

Program Studi Informatika

A picture containing sketch, symbol, drawing, clipart

Description automatically generated

Diajukan oleh :

Paulus Caesario Dito Putra Hartono

205314159

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS SANATA DHARMA**

**YOGYAKARTA**

**2024**

**TRANSFER LEARNING METHOD TO CLASSIFY HANDWRITTEN JAVANESE CHARACTER**

**THESIS**

Present as Partial Fulfillment of The Requirement

To Obtain The *Sarjana Komputer* Degree

In Informatics Study Program

A picture containing sketch, symbol, drawing, clipart

Description automatically generated

Created By:

Paulus Caesario Dito Putra Hartono

205314159

**FACULTY OF SCIENCE AND TECHNOLOGY**

**SANATA DHARMA UNIVERSITY**

**YOGYAKARTA**

**2024**

# HALAMAN PERSETUJUAN PEMBIMBING

**SKRIPSI**

**METODE *TRANSFER LEARNING* UNTUK KLASIFIKASI CITRA HURUF TULIS TANGAN AKSARA JAWA**

Disusun oleh :

Paulus Caesario Dito Putra Hartono

NIM : 205314065

Dosen Pembimbing,

(Ir. Kartono Pinaryanto, S.T., M.Cs.) 25 Januari 2024

# HALAMAN PENGESAHAN

**SKRIPSI**

**METODE *TRANSFER LEARNING* UNTUK KLASIFIKASI CITRA HURUF TULIS TANGAN AKSARA JAWA**

Dipersiapkan dan ditulis oleh

**PAULUS CAESARIO DITO PUTRA HARTONO**

**NIM : 205314065**

Telah dipertahankan di depan panitia penguji

Pada tanggal

Dan dinyatakan memenuhi syarat

**SUSUNAN PANITIAN PENGUJI**

|  |  |  |  |
| --- | --- | --- | --- |
| **JABATAN** |  | **NAMA LENGKAP** | **TANDA TANGAN** |
| Ketua | **:** |  | **………………………** |
| Sekertaris | **:** |  | **………………………** |
| Anggota | **:** |  | **………………………** |

Yogyakarta, ……………….

Fakultas Sains dan Teknologi

Universitas Sanata Dharma

Dekan,

Ir. Drs. Haris Sriwindono, M.Kom, Ph.D.

# PERNYATAAN KEASLIAN KARYA

Saya menyatakan dengan sesungguhnya bahwa skripsi yang saya tulis ini tidak memuat karya atau bagian karya orang lain, kecuali yang telah disebutkan dalam kutipan dan daftar pustaka dengan mengikuti ketentuan sebagaimana layaknya karya ilmiah.

Apabila di kemudian hari ditemukan indikasi plagiarisme dalam naskah ini, saya bersedia menanggung segala sanksi sesuai peraturan perundang-undangan yang berlaku.

Yogyakarta, 25 Janauari 2024

Penulis,

Paulus Caesario Dito Putra Hartono

# LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini, saya mahasiswi Universitas Sanata Dharma:

Nama : Paulus Caesario Dito Putra Hartono

Nomor Mahasiswa : 205314159

Demi pengembangan ilmu pengetahuan, saya memberikan kepada Perpustakaan Universitas Sanata Dharma karya ilmiah saya yang berjudul:

**METODE *TRANSFER LEARNING* UNTUK KLASIFIKASI CITRA HURUF TULIS TANGAN AKSARA JAWA**

beserta perangkat yang diperlukan (bila ada). Dengan demikian, saya memberikan kepada Perpustakaan Universitas Sanata Dharma hak untuk menyimpan, mengalihkan dalam bentuk media lain, mengelolanya dalam bentuk pangkalan data, mendistribusikan secara terbatas, dan mempublikasikannya di Internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya maupun memerikan royalti kepada saya selama tetap mencantumkan nama saya sebagai penulis.

Atas kemajuan teknologi informasi, saya tidak berkeberatan jika nama, tanda tangan, gambar atau image yang ada di dalam karya ilmiah saya terindeks oleh mesin pencari (search engine), misalnya google.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Dibuat di Yogyakarta

Pada tanggal : 25 Januari 2024

Yang menyatakan,

Paulus Caesario Dito Putra Hartono

# KATA PENGANTAR

Dengan memanjatkan puji syukur kepada Tuhan Yang Maha Esa, penulis dapat menyelesaikan tugas akhir dengan judul “Metode *Transfer Learning* untuk Klasifikasi Citra Huruf Tulis Tangan Aksara Jawa”.

Dalam menyelesaikan tugas akhir ini, tentunya penulis mendapatkan bantuan dari berbagai pihak baik secara langsung maupun tidak langsung. Oleh karena itu, dengan tulus penulis mengucapkan terima kasih kepada:

1. Bapak Kartono Pinaryanto, S.T., M.Cs. selaku dosen pembimbing yang telah meluangkan waktu untuk memberikan arahan, motivasi, masukan, dan bimbingan kepada penulis dalam menyelesaikan tugas akhir ini.
2. Ibu Agnes Maria Polina S.Kom., M.Sc. selaku dosen pembimbing akademik dan seluruh dosen Informatika Universitas Sanata Dharma yang telah membantu perjalanan akademis penulis selama masa pendidikan hingga selesainya tugas akhir ini.
3. Keluarga tercinta, Ibu Debora Pitaloka, Kakak Angelina Arini, dan Kak Angelica Arimbi yang selalu mendukung dan mendoakan penulis sehingga dapat menyelesaikan tugas akhir ini.
4. Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu atau terlibat dalam penulisan tugas akhir ini.

Penulis menyadari bahwa masih banyak kekurangan dalam tugas akhir ini. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun untuk menyempurnakannya. Semoga tugas akhir ini dapat bermanfaat bagi para pembaca, khususnya teman-teman di Program Studi Informatika.

Yogyakarta, 25 Januari 2024

Paulus Caesario Dito Putra Hartono

# DAFTAR ISI

[HALAMAN PERSETUJUAN PEMBIMBING i](#_Toc157499816)

[HALAMAN PENGESAHAN ii](#_Toc157499817)

[PERNYATAAN KEASLIAN KARYA iii](#_Toc157499818)

[LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS iv](#_Toc157499819)

[KATA PENGANTAR v](#_Toc157499820)

[DAFTAR ISI vi](#_Toc157499821)

[DAFTAR TABEL xi](#_Toc157499822)

[DAFTAR GAMBAR xii](#_Toc157499823)

[ABSTRAK xvi](#_Toc157499824)

[ABSTRACT xvii](#_Toc157499825)

[BAB I PENDAHULUAN 1](#_Toc157499826)

[1.1. Latar Belakang 1](#_Toc157499827)

[1.2. Rumusan Masalah 3](#_Toc157499828)

[1.3. Batasan Masalah 4](#_Toc157499829)

[1.4. Tujuan Penelitian 4](#_Toc157499830)

[1.5. Manfaat Penelitian 4](#_Toc157499831)

[BAB II TINJAUAN PUSTAKA 5](#_Toc157499832)

[2.1. Aksara Jawa 5](#_Toc157499833)

[2.2. Citra Digital 6](#_Toc157499834)

[2.3. Pengolahan Citra Digital 7](#_Toc157499835)

[2.4. Rescale/Resize 7](#_Toc157499836)

[2.5. Augmentasi Data 8](#_Toc157499837)

[2.6. Normalisasi dan Standardisasi 9](#_Toc157499838)

[2.7. Data Splitting 9](#_Toc157499839)

[2.8. Deep Learning 10](#_Toc157499840)

[2.9. Convolutional Neural Network 10](#_Toc157499841)

[2.10. Activation Functions 11](#_Toc157499842)

[2.11. Dropout 11](#_Toc157499843)

[2.12. Pooling 12](#_Toc157499844)

[2.13. Transfer Learning 13](#_Toc157499845)

[2.14. VGG Pre-trained Model 15](#_Toc157499846)

[2.15. Inception Pre-trained Model 17](#_Toc157499847)

[2.16. Xception Pre-trained Model 19](#_Toc157499848)

[2.17. Confusion Matrix 20](#_Toc157499849)

[2.18. Albumentations 20](#_Toc157499850)

[2.19. Tensorflow 20](#_Toc157499851)

[2.20. Review Literatur 21](#_Toc157499852)

[3. BAB III METODE PENELITIAN 25](#_Toc157499853)

[3.1. Alur Penelitian 25](#_Toc157499854)

[3.2. Data 26](#_Toc157499855)

[3.3. Alat dan Bahan 32](#_Toc157499856)

[3.3.1. Spesifikasi Perangkat Keras 32](#_Toc157499857)

[3.3.2. Libraries 32](#_Toc157499858)

[3.3.3. Preprocessing 33](#_Toc157499859)

[3.3.4. Modeling 36](#_Toc157499860)

[3.3.4.4. Train 39](#_Toc157499861)

[3.3.5. Evaluasi 40](#_Toc157499862)

[3.4. Skenario Pengujian 41](#_Toc157499863)

[4. BAB IV PENGUJIAN DAN ANALISIS 42](#_Toc157499864)

[4.1. Detail Skenario Pengujian 42](#_Toc157499865)

[4.2. Parameter Pengujian 43](#_Toc157499866)

[4.3. Matriks Kinerja 43](#_Toc157499867)

[4.4. Preprocessing Data 43](#_Toc157499868)

[4.4.1. Augmentasi 43](#_Toc157499869)

[4.4.2. Preprocessing Pipeline 46](#_Toc157499870)

[4.5. Modelling 47](#_Toc157499871)

[4.5.1. Pembuatan Model 47](#_Toc157499872)

[4.5.2. Pelatihan Model 49](#_Toc157499873)

[4.6. Analisis Hasil Pengujian 50](#_Toc157499874)

[4.6.1. Evaluasi Matriks Kinerja 51](#_Toc157499875)

[4.6.2. Evaluasi Parameter terhadap Matriks Kinerja 58](#_Toc157499876)

[4.6.3. Evaluasi Top 10 Models 66](#_Toc157499877)

[4.7. Percobaan Model dengan Data Baru 82](#_Toc157499878)

[4.7.1. Parameter Pengambilan Model 83](#_Toc157499879)

[4.7.2. Data 84](#_Toc157499880)

[4.7.3. Hasil 88](#_Toc157499881)

[5. BAB V PENUTUP 97](#_Toc157499882)

[5.1. Kesimpulan 97](#_Toc157499883)

[5.2. Saran 98](#_Toc157499884)

[DAFTAR PUSTAKA 100](#_Toc157499885)

# DAFTAR TABEL

[**Tabel 2.1** Review Literatur 21](#_Toc151729322)

[**Tabel 3.1** Skenario Pengujian 41](#_Toc151729323)

[**Tabel 4.1** Hasil Pelatihan Model berdasarkan Skenario Pengujian 50](#_Toc151729324)

[**Tabel 4.2** Hasil uji skenario tiap model diurutkan berdasarkan kolom Time 57](#_Toc151729325)

[**Tabel 4.3** Hasil uji skenario 10 model terbaik 66](#_Toc151729326)

[**Tabel 4.4** Rincian hasil percobaan baru menggunakan *top 10 models* 88](#_Toc151729327)

[**Tabel 4.5** Rincian hasil percobaan baru menggunakan *top 3 models* 89](#_Toc151729328)

# DAFTAR GAMBAR

[**Gambar 2.1** Aksara Carakan 6](#_Toc151548293)

[**Gambar 2.2** *Rescale Bilinear Interpolation* 7](#_Toc151548294)

[**Gambar 2.3** Augmentasi Data 8](#_Toc151548295)

[**Gambar 2.4** Ilustrasi *max pooling* 12](#_Toc151548296)

[**Gambar 2.5** *Most common deep transfer learning approaches* 13](#_Toc151548297)

[**Gambar 2.6**Arsitektur VGG 16](#_Toc151548298)

[**Gambar 2.7**Arsitektur Inception 18](#_Toc151548299)

[**Gambar 2.8**Arsitektur Xception 19](#_Toc151548300)

[**Gambar 3.1** Alur Penelitian 25](#_Toc151548301)

[**Gambar 3.2**Alur Preprocessing 25](#_Toc151548302)

[**Gambar 3.3**Alur Modeling 26](#_Toc151548303)

[**Gambar 3.4** Data sumber pertama 28](#_Toc151548304)

[**Gambar 3.5** Data sumber kedua 29](#_Toc151548305)

[**Gambar 3.6** Data gabungan 30](#_Toc151548306)

[**Gambar 3.7** Data pengujian hasil pelatihan model 31](#_Toc151548307)

[**Gambar 3.8** Contoh rescale 33](#_Toc151548308)

[**Gambar 3.9** Contoh standardisasi 34](#_Toc151548309)

[**Gambar 3.10** Contoh augmentasi 35](#_Toc151548310)

[**Gambar 3.11** Contoh transfer learning 37](#_Toc151548311)

[**Gambar 3.12** Contoh pooling 38](#_Toc151548312)

[**Gambar 3.13** Contoh dropout 38](#_Toc151548313)

[**Gambar 3.14** Contoh confusion matrix 40](#_Toc151548314)

[**Gambar 4.1** Arsitektur Skenario 42](#_Toc151548315)

[**Gambar 4.2** *Source code config* augmentasi 44](#_Toc151548316)

[**Gambar 4.3** *Source code* fungsi augmentasi 45](#_Toc151548317)

[**Gambar 4.4** *Source code* fungsi *preprocessing pipeline* 46](#_Toc151548318)

[**Gambar 4.5** *Source code download pre-trained* *model* Xception 47](#_Toc151548319)

[**Gambar 4.6** *Summary Model* Xception *Full Freeze* 48](#_Toc151548320)

[**Gambar 4.7** *Source code* beserta *output* pelatihan tiap *epoch* 49](#_Toc151548321)

[**Gambar 4.8** Hasil keseluruhan skenario pengujian berdasarkan nilai akurasi test 51](#_Toc151548322)

[**Gambar 4.9** Grafik Pelatihan Model VGG dengan stagnansi (*loss*) 52](#_Toc151548323)

[**Gambar 4.10** Grafik Pelatihan Model VGG dengan stagnansi (*accuracy*) 53](#_Toc151548324)

[**Gambar 4.11** *Bar-chart* rerata akurasi tiap jenis model 54](#_Toc151548325)

[**Gambar 4.12** *Bar-chart* rerata akurasi tiap jenis model dengan filter 55](#_Toc151548326)

[**Gambar 4.13** *Bar-chart* jumlah model terfilter 55](#_Toc151548327)

[**Gambar 4.14** *Bar-chart* waktu pelatihan tiap model 56](#_Toc151548328)

[**Gambar 4.15** *Bar-chart* rerata waktu pelatihan tiap jenis model 56](#_Toc151548329)

[**Gambar 4.16** *Bar-chart* performa akurasi berdasarkan parameter augmentasi 58](#_Toc151548330)

[**Gambar 4.17** *Bar-chart* performa waktu berdasarkan parameter augmentasi 58](#_Toc151548331)

[**Gambar 4.18** *Bar-chart* performa akurasi berdasarkan parameter *freeze* 59](#_Toc151548332)

[**Gambar 4.19** *Bar-chart* performa waktu berdasarkan parameter *freeze* 60](#_Toc151548333)

[**Gambar 4.20** *Bar-chart* performa akurasi berdasarkan parameter *learning rate* 60](#_Toc151548334)

[**Gambar 4.21** *Bar-chart* performa waktu berdasarkan parameter *learning rate* 61](#_Toc151548335)

[**Gambar 4.22** *Bar-chart* performa akurasi berdasarkan parameter *layer* 62](#_Toc151548336)

[**Gambar 4.23** *Bar-chart* performa waktu berdasarkan parameter *layer* 62](#_Toc151548337)

[**Gambar 4.24** *Bar-chart* performa akurasi berdasarkan parameter *optimizer* 63](#_Toc151548338)

[**Gambar 4.25** *Bar-chart* performa waktu berdasarkan parameter *optimizer* 63](#_Toc151548339)

[**Gambar 4.26** C*orrelation matrix* dari data hasil uji skenario 65](#_Toc151548340)

[**Gambar 4.27** *Bar-chart* performa akurasi set uji 10 model terbaik 68](#_Toc151548341)

[**Gambar 4.28** *Bar-chart* jumlah model berdasarkan parameter augmentasi 69](#_Toc151548342)

[**Gambar 4.29** *Bar-chart* jumlah model berdasarkan parameter *freeze* 69](#_Toc151548343)

[**Gambar 4.30** *Bar-chart* jumlah model berdasarkan parameter *learning rate* 70](#_Toc151548344)

[**Gambar 4.31** *Bar-chart* jumlah model berdasarkan parameter *layer* 71](#_Toc151548345)

[**Gambar 4.32** *Bar-chart* jumlah model berdasarkan parameter *optimizer* 72](#_Toc151548346)

[**Gambar 4.33** Grafik akurasi pelatihan 10 model terbaik 73](#_Toc151548347)

[**Gambar 4.34** Grafik akurasi validasi pelatihan 10 model terbaik 73](#_Toc151548348)

[**Gambar 4.35** Grafik *loss* pelatihan 10 model terbaik 74](#_Toc151548349)

[**Gambar 4.36** Grafik *validation loss* pelatihan 10 model terbaik 74](#_Toc151548350)

[**Gambar 4.37** *Bar-chart epoch* masing – masing 10 model terbaik 75](#_Toc151548351)

[**Gambar 4.38** Grafikrerata akurasi 10 model terbaik tiap *epoch* 76](#_Toc151548352)

[**Gambar 4.39** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* 77](#_Toc151548353)

[**Gambar 4.40** Grafikrerata *loss* 10 model terbaik tiap *epoch* 77](#_Toc151548354)

[**Gambar 4.41** Grafikrerata *validation* *loss* 10 model terbaik tiap *epoch* 77](#_Toc151548355)

[**Gambar 4.42** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter augmentasi 78](#_Toc151548356)

[**Gambar 4.43** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *freeze* 79](#_Toc151548357)

[**Gambar 4.44** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *learning rate* 79](#_Toc151548358)

[**Gambar 4.45** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *layer* 79](#_Toc151548359)

[**Gambar 4.46** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *optimizer* 80](#_Toc151548360)

[**Gambar 4.47** *Bar-chart* 3 model terbaik berdasarkan nilai tiap parameter *freeze* 83](#_Toc151548361)

[**Gambar 4.48** Data Aksara Jawa setelah di-*scan* dari kertas 84](#_Toc151548362)

[**Gambar 4.49** Data Aksara Jawa setelah di-*crop* menggunakan perangkat lunak 85](#_Toc151548363)

[**Gambar 4.50** Data Aksara Jawa setelah dilakukan *preprocessing* 86](#_Toc151548364)

[**Gambar 4.51** Data Aksara Jawa yang dihapus (*cleaning*) 87](#_Toc151548365)

[**Gambar 4.52** Rincian hasil percobaan baru menggunakan *top 10 models* dengan *bar-chart* 89](#_Toc151548366)

[**Gambar 4.53** Rincian hasil percobaan baru menggunakan *top 3 models* dengan *bar-chart* 90](#_Toc151548367)

[**Gambar 4.54** *Bar-chart* performa maksimum masing – masing model berdasarkan parameter *freeze* 91](#_Toc151548368)

[**Gambar 4.55** *Confusion matrix* dengan model id 66 92](#_Toc151548369)

[**Gambar 4.56** Visualisasi prediksi 20 data secara acak 93](#_Toc151548370)

[**Gambar 4.57** Visualisasi prediksi 20 data yang salah secara acak 94](#_Toc151548371)

# ABSTRAK

Indonesia kaya akan warisan budaya, salah satunya adalah aksara Jawa yang menjadi bagian dari kekayaan budaya nasional. Namun, saat ini, tidak semua individu Jawa, khususnya generasi muda, memiliki kemampuan membaca aksara Jawa. Oleh karena itu, integrasi teknologi muncul sebagai solusi potensial untuk melestarikan warisan budaya ini. Deep learning, sebagai cabang dari machine learning, membuktikan diri sebagai alat penting dalam mengenali pola-pola rumit dalam data kompleks. Convolutional Neural Network (CNN) menonjol sebagai jaringan saraf yang efektif dalam bidang visi komputer, menunjukkan akurasi yang mendekati kemampuan manusia. Namun, efektivitas CNN sangat tergantung pada dataset yang besar. Untuk mengatasi keterbatasan data, metode transfer learning dapat digunakan untuk menanggulangi permasalahan ini. Penelitian ini berusaha mengimplementasikan dan mengidentifikasi model pra-latih optimal dari VGG, Inception, dan Xception untuk mengklasifikasikan citra huruf tulis tangan aksara Jawa. Hasil penelitian menunjukkan bahwa di antara ketiga model yang diuji, Inception V3 menunjukkan kinerja terbaik secara keseluruhan dalam hal akurasi, efisiensi komputasi, dan stabilitas hasil. Akurasi tertinggi yang dicapai adalah 100% pada dataset pengujian, meskipun terjadi penurunan ke sekitar 90% saat diuji dengan data baru yang sebelumnya belum pernah dilihat. Potensi untuk meningkatkan model masih terbuka lebar melalui penambahan variasi data, penyempurnaan *preprocessing pipeline*, dan penyesuaian *hyperparameter*.

Kata Kunci: *Transfer Learning*, Aksara Jawa, *Data Augmentation*, VGG, Inception, Xception

# ABSTRACT

Indonesia is rich in cultural heritage, one of which is the Javanese script that forms part of the national cultural wealth. However, not all Javanese individuals, especially the younger generation, have the ability to read Javanese script. Therefore, the integration of technology emerges as a potential solution to preserve this cultural heritage. Deep learning, as a branch of machine learning, has proven itself as an important tool in recognizing complex patterns in complex data. The Convolutional Neural Network (CNN) stands out as an effective neural network in the field of computer vision, showing accuracy close to human capabilities. However, the effectiveness of CNNs is highly dependent on large datasets. To overcome data limitations, transfer learning methods can be used to address this issue. This research attempts to implement and identify the optimal pre-trained models from VGG, Inception, and Xception to classify images of handwritten Javanese script letters. The research results show that among the three models tested, Inception V3 shows the best overall performance in terms of accuracy, computational efficiency, and result stability. The highest accuracy achieved is 100% on the test dataset, although there is a decrease to around 90% when tested with new data that has never been seen before. The potential to improve the model is still wide open through the addition of data variations, refinement of the preprocessing pipeline, and adjustment of hyperparameters.

Kata Kunci: *Transfer Learning*, Javanese Script, Data Augmentation, VGG, Inception, Xception

# BAB I PENDAHULUAN

## Latar Belakang

Indonesia, sebuah negara dengan berbagai macam budaya, religi, dan bahasa, menghadapi tantangan dalam melestarikan salah satu warisan budayanya, yaitu aksara Jawa. Aksara ini berkembang di pulau Jawa, pulau dengan populasi tinggi di Indonesia, dan masih dapat ditemukan di papan jalanan, tembok, dan peninggalan barang historikal. Meski dianggap sebagai warisan budaya nasional, aksara Jawa menghadapi ancaman kepunahan.

Hal ini disebabkan oleh semakin sedikitnya masyarakat, terutama generasi muda, yang mampu membaca dan menulis aksara ini (Abdul Robby et al., 2019). Faktor utamanya adalah kurangnya pengetahuan dan pemahaman tentang pentingnya melestarikan aksara Jawa sebagai bagian dari identitas budaya nasional. Selain itu, penggunaan aksara Jawa dalam kehidupan sehari-hari juga semakin berkurang. Aksara ini lebih banyak ditemukan dalam konteks sejarah dan budaya, seperti di papan nama jalan dan artefak bersejarah, dan kurang digunakan dalam konteks modern, seperti media digital dan pendidikan formal.

Jika kondisi ini terus berlanjut, ada kemungkinan aksara Jawa akan terlupakan dan punah. Ini akan menjadi kerugian besar bagi warisan budaya Indonesia, karena aksara Jawa merupakan simbol penting dari kekayaan budaya dan sejarah bangsa. Oleh karena itu, penting untuk mencari solusi agar aksara Jawa dapat terus dilestarikan dan dipelajari oleh generasi mendatang. Salah satu solusinya adalah dengan bantuan teknologi, proses pembelajaran dapat menjadi lebih menarik dan interaktif, sehingga dapat meningkatkan minat masyarakat untuk mempelajari aksara Jawa. Dengan demikian, warisan budaya ini dapat dilestarikan dan tetap relevan di era modern.

Salah satu teknologi yang dapat digunakan untuk pengenalan aksara jawa adalah *deep learning*. *Deep learning* merupakan salah satu cabang dari *machine learning* yang dapat digunakan untuk mengenali pola dari data yang kompleks (Lecun et al., 2015). Dalam hal ini, *deep learning* dapat digunakan untuk mengenali pola dari aksara jawa. Berbagai penelitian terkait klasifikasi citra terkhususnya citra tulis tangan telah dilakukan dan menghasilkan akurasi 83% (Ahmed et al., 2022). *Deep learning* telah secara dramatis meningkatkan standar terbaik dalam *speech recognition*, *visual object recognition*, *object detection* dan banyak domain lainnya seperti *drug discovery* dan *genomics* (Lecun et al., 2015).

*Convolutional Neural Network* (CNN) yang merupakan salah satu jenis *network* atau jaringan pada *deep learning*,telah menjadi *representative neural networks* pada bidang *computer vision* karena performa yang dihasilkan dari *network* tersebut hampir menyentuh *human level accurate*  (Chandrarathne et al., 2020; Li et al., 2022). Namun hal tersebut dapat berhasil karena terdapat dataset yang besar (He et al., 2015).

Untuk mengatasi masalah limitasi data, metode *transfer learning* telah berhasil mengatasi masalah keterbatasan data dengan mentransfer pengetahuan yang dipelajari dari satu domain aplikasi ke domain lain yang relevan (Yosinski et al., 2014). Dalam praktiknya, cara umum dari *deep transfer learning* adalah menggunakan CNN yang telah dilatih sebelumnya sebagai model sumber, yang dilatih dengan data dalam jumlah besar seperti ImageNet (Deng et al., 2010). Beberapa model sumber tersebut antara lain adalah Xception, Inception, dan VGG.

Penelitian yang serupa pada bidang citra tulis tangan juga sudah dilakukan oleh berbagai peneliti lainnya. Pada penelitian terdahulu metode *transfer learning* telahmenghasilkan akurasi 98% pada citra aksara jawa dan 91% pada citra aksara sunda (Kesaulya et al., 2022; Khalifa et al., 2022). Dari penemuan tersebut, peneliti akan mencoba mengimplementasikan dan mencari *pre-trained* model mana yang terbaik dari Xception, Inception, dan VGG untuk klasifikasi citra huruf tulis tangan aksara jawa.

## Rumusan Masalah

1. Bagaimana cara melakukan implementasi *transfer learning* pada kasus citra huruf tulis tangan aksara jawa?
2. Berapa akurasi yang diperoleh dari metode *transfer learning*?
3. Dari ketiga *pre-trained* model yang penulis ambil, manakah yang terbaik?

## Batasan Masalah

1. Penelitian ini difokuskan untuk klasifikasi pada citra huruf tulis tangan aksara jawa
2. Metode yang digunakan adalah *transfer learning* menggunakan 3 *pre-trained* model yaitu Xception, Inception, dan VGG
3. Data yang digunakan adalah citra huruf tulis tangan aksara jawa tanpa pasangan atau aksara carakan (nglegena) yang sesuai dengan tata penulisan aksara Jawa.
4. Data yang digunakan pada penelitian inimerupakan data citra huruf aksara Jawa yang berwarna hitam dengan background warna putih.
5. Penelitian ini berfokus untuk mencari *pre-trained* model terbaik dari ketiga model yang diambil penulis

## Tujuan Penelitian

1. Mengklasifikasikan citra huruf tulis tangan aksara jawa (carakan) dengan jumlah total 20 jenis huruf menggunakan metode *transfer learning*
2. Mengevaluasi ketiga *pre-trained* model yang digunakan oleh penulis

## Manfaat Penelitian

1. Mengetahui performa *pre-trained* model terbaik yang diambil oleh penulis, untuk kasus citra huruf aksara Jawa
2. Membantu orang awam untuk mengenali aksara Jawa
3. Dapat digunakan penulis lain untuk pengembangan penelitian lebih lanjut

# BAB II TINJAUAN PUSTAKA

Bab ini menjelaskan tentang definisi dan teori-teori yang digunakan sebagai landasan penelitian yang berasal dari hasil publikasi dan penelitian dan/atau buku yang relevan.

## Aksara Jawa

Aksara Jawa, yang juga dikenal sebagai Hanacaraka dan Dentawyanjana, adalah salah satu aksara tradisional di Indonesia yang berkembang di Pulau Jawa. Aksara ini terutama digunakan untuk menulis bahasa Jawa, tetapi juga digunakan untuk menulis beberapa bahasa daerah lainnya seperti bahasa Sunda, Madura, Sasak dan Melayu serta bahasa historis seperti Sanskerta dan Kawi. Aksara Jawa berasal dari aksara Brahmi India melalui aksara Kawi dan berkerabat dekat dengan aksara Bali. Aksara ini aktif digunakan dalam sastra dan tulisan sehari-hari oleh masyarakat Jawa dari pertengahan abad ke-15 hingga pertengahan abad ke-20 sebelum perlahan-lahan digantikan oleh huruf Latin. Aksara ini masih diajarkan di DI Yogyakarta, Jawa Tengah, Jawa Timur, Cirebon dan Indramayu sebagai bagian dari muatan lokal, tetapi penggunaannya terbatas dalam kehidupan sehari-hari. Aksara Jawa merupakan sistem tulisan *abugida* yang terdiri dari sekitar 20 aksara dasar (Poerwadarminta, 1939; Everson, 2008).

A picture containing text, font, screenshot, number

Description automatically generated

**Gambar 2.1** Aksara Carakan

sumber: <https://www.rukita.co/stories/aksara-jawa-lengkap/>

## Citra Digital

Citra digital terdiri dari sejumlah elemen terbatas, masing-masing memiliki lokasi dan nilai tertentu. Elemen-elemen ini disebut elemen citra, pel, dan piksel. Sebuah citra digital dapat didefinisikan sebagai fungsi dua dimensi, f(x,y), di mana x dan y adalah koordinat spasial (bidang), dan amplitudo f pada setiap pasangan koordinat (x,y) disebut intensitas atau tingkat abu-abu citra di titik tersebut. Bidang pengolahan citra digital mengacu pada pengolahan citra digital dengan menggunakan komputer digital (Gonzalez & Woods, 2018)

## Pengolahan Citra Digital

Pemrosesan citra adalah kumpulan teknik komputasi untuk menganalisis, meningkatkan, mengompres, dan merekonstruksi citra. Komponen utamanya meliputi impor, di mana sebuah citra ditangkap melalui pemindaian atau fotografi digital; analisis dan manipulasi citra yang dilakukan menggunakan berbagai aplikasi perangkat lunak khusus; serta output (misalnya, ke printer atau monitor). Pemrosesan citra memiliki berbagai aplikasi yang luas di berbagai bidang, termasuk astronomi, kedokteran, robotik industri, dan pemantauan jarak jauh oleh satelit (Gonzalez & Woods, 2018).

## Rescale/Resize

Proses *rescale/resize* pada citra merupakan tahap pra-pemrosesan yang kritis dalam visi komputer. Secara prinsip, model *deep learning* dapat dilatih lebih cepat pada citra berukuran kecil. Citra input yang lebih besar memerlukan jaringan saraf untuk belajar dari empat kali lipat jumlah piksel, yang pada akhirnya meningkatkan waktu pelatihan untuk arsitektur tersebut (Saponara & Elhanashi, 2022).

A picture containing diagram, line

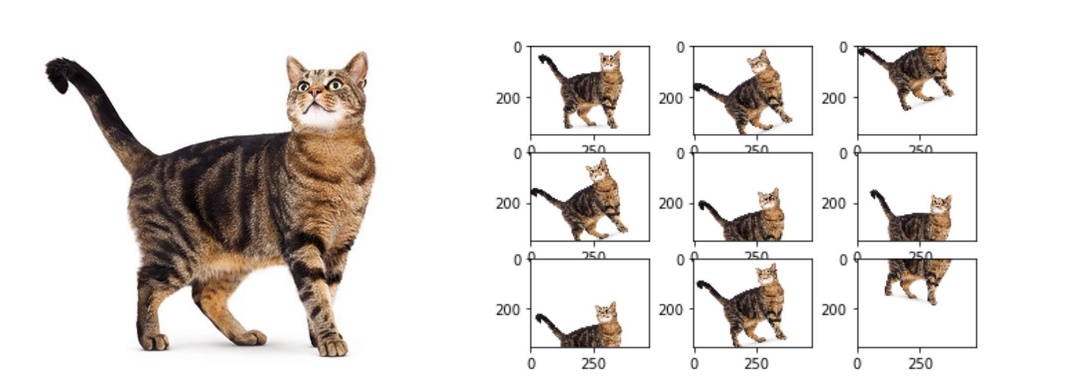
Description automatically generated

**Gambar 2.2** *Rescale Bilinear Interpolation*

sumber: <https://theailearner.com/2018/12/29/image-processing-bilinear-interpolation/>

## Augmentasi Data

Tujuan dari augmentasi data adalah menambahkan titik data baru ke ruang input dengan memodifikasi citra pelatihan sambil mempertahankan informasi semantik dan label target. Oleh karena itu, augmentasi data digunakan untuk mengurangi *overfitting*. Beberapa penelitian mengkonfirmasi pentingnya augmentasi data baik dalam pelatihan maupun pengujian, dan menunjukkan bahwa hal tersebut dapat menghasilkan peningkatan kinerja yang lebih besar daripada mengumpulkan citra sebagai dataset baru (Perez et al., 2018).



**Gambar 2.3** Augmentasi Data

sumber: <https://analyticsindiamag.com/image-data-augmentation-impacts-performance-of-image-classification-with-codes/>

## Normalisasi dan Standardisasi

Normalisasi dan standarisasi citra adalah proses yang digunakan dalam pemrosesan citra untuk memastikan perbandingan yang optimal antara metode akuisisi data dan instansi tekstur. Tujuan utama dari normalisasi dan standarisasi citra adalah mengubah piksel citra sehingga mereka dapat dibandingkan secara konsisten dalam berbagai situasi (Gonzalez & Woods, 2018).

(2.1)

(2.2)

## Data Splitting

*Data splitting* atau pemisahan data melibatkan pembagian dataset menjadi tiga bagian: set pelatihan, set validasi, dan set pengujian. Set pelatihan digunakan untuk membangun model, sementara set validasi dan pengujian berfungsi sebagai set penahan. Tidak ada proporsi yang pasti untuk pembagian dataset ini, namun aturan umum sebelumnya adalah 70% untuk pelatihan dan 15% untuk validasi dan pengujian. Akan tetapi, dalam era *big data*, proporsi seperti 95% untuk pelatihan dan 2,5%/2,5% untuk validasi/pengujian dapat dipertimbangkan. Dengan menggunakan ketiga subset ini, diharapkan model yang dibangun dapat memprediksi dengan baik contoh-contoh yang tidak dilihat oleh algoritma pembelajaran. Set validasi digunakan untuk memilih algoritma pembelajaran dan menentukan nilai *hyperparameter* terbaik, sementara set pengujian digunakan untuk mengevaluasi model sebelum digunakan di lapangan (Burkov, 2019).

## Deep Learning

*Deep learning* adalah cabang dari *machine learning* yang menggunakan jaringan saraf tiruan dengan banyak lapisan tersembunyi untuk memodelkan data yang kompleks (Lecun et al., 2015). Teknik ini memungkinkan komputer untuk mempelajari representasi data yang abstrak dan hierarkis secara otomatis dari data mentah, tanpa perlu pengetahuan domain yang spesifik atau rekayasa fitur yang rumit. *Deep learning* telah mencapai hasil yang sangat baik dalam berbagai tugas pengenalan pola, termasuk pengenalan suara, pengenalan gambar, dan pemrosesan bahasa alami.

## Convolutional Neural Network

*Convolutional Neural Network* (CNN) adalah jenis *network* khusus yang secara signifikan mengurangi jumlah parameter dalam *deep neural* *network* dengan banyak unit tanpa kehilangan terlalu banyak kualitas model. CNN telah menemukan aplikasi dalam pengolahan gambar dan teks di mana mereka mengalahkan banyak patokan yang sebelumnya ditetapkan (Burkov, 2019). Dalam implementasi *transfer learning*, CNN bertugas sebagai mengekstrak fitur dari citra data pada model tersebut.

## Activation Functions

Untuk memungkinkan jaringan saraf mempelajari batas keputusan yang kompleks, diterapkan fungsi aktivasi non-linier pada beberapa lapisannya. Fungsi yang umum digunakan meliputi *tanh, ReLU, softmax*, dan varian dari fungsi-fungsi tersebut. Secara teknis, setiap neuron menerima sinyal masukan yang merupakan jumlah terbobot dari bobot sinaptik dan nilai aktivasi dari neuron yang terhubung (Zaccone & Karim, 2018).

(2.3)

(2.4)

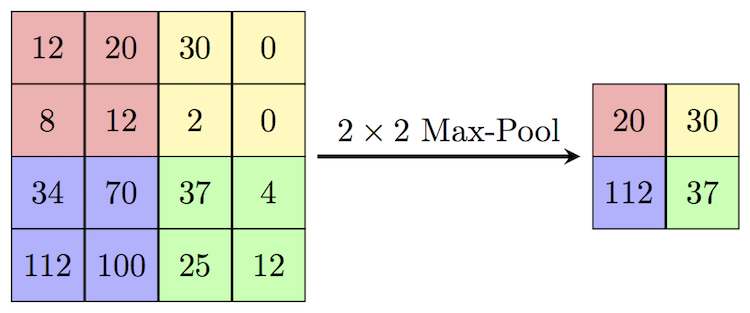
## Dropout

*Dropout* adalah teknik yang digunakan untuk mengurangi *overfitting* dalam jaringan dengan banyak lapisan dan/atau neuron. Secara umum, lapisan *dropout* ditempatkan setelah lapisan yang memiliki sejumlah besar neuron yang dapat dilatih (Zaccone & Karim, 2018).

Teknik ini mengubah pendekatan pembelajaran bobot. Alih-alih mempelajari semua bobot jaringan bersama-sama, *dropout* melatih sebagian dari mereka dalam iterasi pelatihan *batch* (Ranjan, 2019).

## Pooling

Fungsi *pooling* dalam *deep learning* adalah untuk mengurangi jumlah koefisien peta fitur yang diproses. Ini dicapai melalui *downsampling*, yang mengurangi dimensi spasial dari input. *Pooling* juga membantu menginduksi hirarki filter spasial dengan membuat lapisan konvolusi berturut-turut melihat jendela yang semakin besar dari segi fraksi input asli yang mereka tutupi. *Max pooling* (salah satu jenis *pooling*) cenderung bekerja lebih baik daripada metode *downsampling* lainnya, seperti *average pooling* atau menggunakan *strides* di lapisan konvolusi sebelumnya, karena lebih informatif untuk melihat kehadiran maksimal fitur yang berbeda daripada kehadiran rata-rata tersebut (Chollet, 2021).



**Gambar 2.4** Ilustrasi *max pooling*

sumber: <https://paperswithcode.com/method/max-pooling>

## Transfer Learning

*Transfer learning* adalah peningkatan pembelajaran dalam tugas baru melalui transfer pengetahuan dari tugas terkait yang sudah dipelajari (Torrey & Shavlik, 2009). *Transfer learning* terdiri dari mengambil jaringan yang sudah dibangun dan membuat perubahan yang sesuai pada parameter dari berbagai lapisan sehingga dapat menyesuaikan dengan dataset lain (Zaccone & Karim, 2018).

**A picture containing text, screenshot, font, diagram

Description automatically generated**

**Gambar 2.5** *Most common deep transfer learning approaches*

(Iman et al., 2023)

Pendekatan populer pertama adalah menyesuaikan model yang telah dilatih pada data target, hal ini merupakan metode *Deep Transfer Learning* (DTL) yang paling umum digunakan karena kemudahannya. Metode ini dapat meningkatkan pelatihan pada data target dengan mengurangi biaya pelatihan dan mengatasi kebutuhan dataset target yang luas, tetapi masih rentan terhadap pelupakan yang mematikan. Metode ini telah diterapkan pada dataset gambar dan tabular di berbagai bidang. Pendekatan populer kedua adalah membekukan lapisan CNN dalam model yang telah dilatih dan menyesuaikan hanya lapisan terhubung sepenuhnya lateral. Dalam metode ini, lapisan CNN mengekstrak fitur dari dataset yang diberikan, dan *fully connected layer* bertanggung jawab untuk klasifikasi dan akan disesuaikan dengan tugas baru untuk data target (Iman et al., 2022).

## VGG Pre-trained Model

Visual Geometric Group (VGG) adalah model arsitektur jaringan saraf tiruan yang sangat dalam dan dikembangkan oleh Visual Geometry Group di Universitas Oxford (Simonyan & Zisserman, 2014). Model ini terdiri dari beberapa lapisan konvolusi yang diikuti oleh lapisan *max-pooling* dan beberapa lapisan sepenuhnya terhubung di bagian akhir. Salah satu keunggulan utama dari model VGG adalah penggunaan filter konvolusi berukuran kecil (3x3) di semua lapisannya, yang memungkinkan jaringan untuk memiliki kedalaman yang lebih besar dengan jumlah parameter yang sama dibandingkan dengan jaringan yang menggunakan filter konvolusi berukuran lebih besar. Model VGG mencapai hasil yang sangat baik pada *benchmark* klasifikasi ILSVRC 2012 dan telah digunakan secara luas sebagai titik awal untuk berbagai tugas pengenalan gambar.

**A picture containing text, screenshot, parallel, number

Description automatically generated**

**Gambar 2.6**Arsitektur VGG

(Simonyan & Zisserman, 2014)

## Inception Pre-trained Model

Inception adalah model arsitektur jaringan saraf tiruan yang digunakan untuk visi komputer. Model ini dirancang untuk meningkatkan kualitas jaringan dengan cara yang efisien secara komputasi dengan menggunakan konvolusi yang difaktorkan dan regularisasi agresif (Szegedy et al., 2015). Inception-v3 adalah salah satu versi dari model Inception yang mencapai kinerja tinggi pada *benchmark* klasifikasi ILSVRC 2012 dengan biaya komputasi yang relatif rendah dibandingkan dengan arsitektur yang lebih sederhana dan monolitik (Szegedy et al., 2016). Model ini juga menggunakan jumlah parameter yang lebih sedikit dan regularisasi tambahan dengan *classifier auxilary* yang dinormalisasi *batch* dan *label-smoothing* untuk melatih jaringan berkualitas tinggi pada set pelatihan berukuran relatif sedang (Szegedy et al., 2016).

**A picture containing text, screenshot, number, font

Description automatically generated**

**Gambar 2.7**Arsitektur Inception

(Szegedy et al., 2016)

## Xception Pre-trained Model

Xception adalah arsitektur jaringan saraf konvolusional yang sepenuhnya didasarkan pada lapisan konvolusi terpisah secara mendalam *(depthwise separable convolution layers).* Arsitektur ini diusulkan sebagai perpanjangan dari arsitektur Inception, di mana modul Inception telah digantikan dengan konvolusi terpisah secara mendalam. Arsitektur Xception adalah tumpukan linier lapisan konvolusi terpisah secara mendalam dengan koneksi residual (Chollet, 2016).

**A screenshot of a computer program

Description automatically generated with low confidence**

**Gambar 2.8**Arsitektur Xception

(Chollet, 2016)

## Confusion Matrix

Confusion Matrix adalah tabel yang merangkum seberapa sukses model klasifikasi dalam memprediksi contoh yang termasuk dalam berbagai kelas. Salah satu sumbu dari *confusion matrix* adalah label yang diprediksi oleh model, dan sumbu lainnya adalah label sebenarnya. Dalam masalah klasifikasi biner, terdapat dua kelas (Burkov, 2019).

## Albumentations

Albumentations adalah *library* Python untuk augmentasi citra yang cepat dan fleksibel. *Library* ini secara efisien mengimplementasikan berbagai operasi transformasi gambar yang kaya dan dioptimalkan untuk kinerja, sambil memberikan antarmuka augmentasi gambar yang ringkas namun kuat untuk berbagai tugas visi komputer, termasuk klasifikasi objek, segmentasi, dan deteksi (Buslaev et al., 2020).

## Tensorflow

TensorFlow adalah antarmuka untuk mengekspresikan algoritma *machine learning* dan implementasi untuk menjalankan algoritma tersebut. Sistem ini fleksibel dan dapat digunakan untuk mengekspresikan berbagai macam algoritma. TensorFlow telah digunakan untuk penelitian dan penerapan sistem pembelajaran mesin di berbagai bidang. API TensorFlow dan implementasi referensi dirilis sebagai paket sumber terbuka pada November 2015 dan tersedia di [www.tensorflow.org](http://www.tensorflow.org) (Abadi et al., 2016).

## Review Literatur

**Tabel 2.1** Review Literatur

| **No** | **Judul** | **Penulis (Tahun)** | **Metode** | **Hasil** |
| --- | --- | --- | --- | --- |
| 1 | *Transfer Learning Implementation on Sundanese Script Recognition Using Convolutional Neural Network* | MUHAMMAD KHALIFA U, et al. (2022) | *Transfer learning* dengan *pre-trained* model dari dataset *alphabet*, *devanagari*, *arabic*, dan aksara jawa | Model *transfer learning* terbaik didapat dari *pre-trained* dataset *arabic,* mencapai akurasi 91,86% dan *loss* 0.2814 |
| 2 | Javanese Script Text Image Recognition Using Convolutional Neural Networks | Goldy Najma Adli Kesaulya, et al. (2022) | *Transfer learning* dengan *pre-trained* model ResNeXt dengan melakukan *freezing* di 4 layer pertama dari 10 layer model *pre-trained* | Akurasi testing mencapai 98.19% |
| 3 | Deep Learning for Image Classification on Very Small Datasets Using Transfer Learning | Menying Shu (2019) | *Fine-tuned pre-trained* model VGG16, VGG19, InceptionV3, InceptionResNetV2 | Akurasi testing mencapai 96% dengan menggunakan *fine-tuned* model dari InceptionResNetV2 |
| 4 | A Close Look at Deep Learning with Small Data | L Brigato, et al. (2020) | Review metode yang dapat digunakan untuk menyelesaikan permasalahan *small data*, menggunakan berbagai variasi dari model CNN dan ResNet | *Regularization technique* seperti *data augmentation* dan dropout dapat meningkatkan performa dari *deep learning* |
| 5 | A Comprehensive Study on Deep Image Classification with Small Datasets | Gayani Chandrarathne, et al. (2019) | Review metode *transfer learning* dengan *fine-tuned pre-trained* model VGG-16 pada berbagai macam dataset. | *Fine-tuning* (*re-init few layers*)dan *(whole network)* menunjukkan performa yang signifikan dibandingkan dengan *scratch training.* Hasil akurasi yang diperoleh dari *fine-tuning* adalah 91.4%, 95.52% dan 79.6% secara terurut |
| 6 | Understanding the Mechanisms of Deep Transfer Learning for Medical Images | Hariharan Ravishankar, et al. (2017) | CNN *transfer learning* CaffeNet | *Transferred* dan *fine-tuned* model dapat menungguli performa *state-of-the-art* *feature engineered pipeline* (Haar) dan menghasilkan akurasi 85% |
| 7 | Transfer Learning using CNN for Handwritten Devanagari Character Recognition | Nagender Aneja, et al. (2019) | *Transfer learning* AlexNet, DenseNet, VGG, dan Inception | InceptionV3 menghasilkan akurasi sebesar 99% |
| 8 | A Deep Learning-Based Framework for Automatic Brain Tumors Classification Using Transfer Learning | Arshia Rehman, et al. (2019) | *Transfer learning* menggunakan *pre-trained* model AlexNet, GoogLeNet, dan VGG16 dari dataset ImageNet | *Fine-tuned* VGG16 menghasilkan akurasi tertinggi sebesar 98.69% |
| 9 | A Review of Deep Transfer Learning and Recent Advancements | Mohammadreza Iman, et al. (2023) | Review berbagai metode *deep transfer learning* yang sudah diteliti dalam waktu dekat | (i) *Finetuning*, (ii) *Freezing* CNN *Layers*, dan (iii) *Progressive Learning* adalah teknik yang telah terbukti kemampuan dan efektivitasnya untuk berbagai masalah *machine learning*. |
| 10 | Text recognition on images using pre-trained CNN | Afgani Fajar Rizky, et al. (2023) | *Transfer leraning* dengan *augmentation, freeze-layers,* dan *fine-tuning* telah dilakukan dengan menggunakan model AlexNet, VGG, ResNet, dan DenseNet | Akurasi terbaik dihasilkan dari model VGG dengan 0 *freeze layer* dan *augmentation* |
| 11 | Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images | Srikanth Tammina (2019) | *Transfer learning* menggunakan *augmentation, fine-tuning,* dan *freeze layers* dengan *pre-trained* model VGG | Hasil terbaik diperoleh dari proses *fine-tuning* dan *augmentation* dengan akurasi 95.40% pada validation |

# BAB III METODE PENELITIAN

## Alur Penelitian

Penelitian akan dilakukan dengan bahasa pemrograman Python dan berbagai *library* yang disebutkan pada bab 3.3.2. Platform yang digunakan untuk mengimplementasikan penelitian adalah Kaggle Notebooks. Untuk mencapai tujuan yang telah dipaparkan pada bab I. Berikut flowchart penelitian yang dirancang untuk menjelaskan langkah - langkah yang akan diambil di penelitian ini, yang akan didetailkan pada sub bab 3.2 dan seterusnya.

A picture containing text, font, screenshot, graphics

Description automatically generated

**Gambar 3.1** Alur Penelitian

Secara garis besar, terdapat 4 tahap utama dari penelitian ini. Tahap pre-processing dan modeling akan diperjelas pada diagram flowchart dibawah ini.

A picture containing screenshot, line, text, font

Description automatically generated

**Gambar 3.2**Alur Preprocessing

A picture containing text, screenshot, line, rectangle

Description automatically generated

**Gambar 3.3**Alur Modeling

Detail dari kedua diagram diatas akan dijelaskan lebih lanjut pada bab 3.3.3 dan seterusnya.

## Data

Data yang penulis gunakan diambil dari kaggle. Kaggle adalah sebuah platform komunitas *data science* yang berisi berbagai sumber dataset, notebook, kompetisi, dan lain sebagainya. Terdapat tiga sumber dataset yang penulis ambil yaitu,

1. <https://www.kaggle.com/datasets/phiard/aksara-jawa>  
   dengan total data 2634 sebagai sumber pertama
2. <https://www.kaggle.com/datasets/vzrenggamani/hanacaraka> dengan total data 1579 sebagai sumber kedua
3. Pengumpulan dari 30 peserta berbeda dengan total data kurang lebih 800 data

Sumber data pertama dan kedua akan digabung menjadi satu sebagai dataset yang akan diuji oleh penulis. Total data yang akan digunakan pada tahap pelatihan model di dalam penelitian ini adalah 4213 data. Kemudian, sumber data ketiga akan digunakan sepenuhnya sebagai pengujian model terbaik dari hasil pelatihan pembuatan model.

Data ketiga merupakan sumber data baru yang diperoleh melalui tulisan tangan di atas kertas hvs menggunakan bolpoin, kemudian dipindai menggunakan printer, dan dipotong menggunakan perangkat lunak Adobe Photoshop untuk mengambil setiap huruf dari tulisan tangan. Data ini diperoleh dari 30 individu yang memiliki berbagai tingkat pengalaman dalam menulis aksara Jawa. Proses pembersihan data juga dilakukan oleh penulis untuk menghapus data yang tidak sesuai dengan tulisan aksara Jawa.

Detail lebih lengkap pada data dapat ditemukan pada Bab 4.7.2 yang membahas tentang tahapan proses pengolahan data.

A picture containing text, diagram, sketch, design

Description automatically generated

**Gambar 3.4** Data sumber pertama

A picture containing text, font, black and white, typography

Description automatically generated

**Gambar 3.5** Data sumber kedua

A picture containing text, diagram, font, design

Description automatically generated

**Gambar 3.6** Data gabungan

A sheet of black and white writing

Description automatically generated

**Gambar 3.7** Data pengujian hasil pelatihan model

## Alat dan Bahan

### Spesifikasi Perangkat Keras

Perangkat yang digunakan pada penelitian ini diambil dari platform notebook kaggle. Kaggle menyediakan jupyter notebook yang bisa digunakan untuk menjalankan kode dengan menggunakan GPU, CPU, dan RAM yang cukup untuk mengimplementasikan *deep learning*. Berikut detail spesifikasinya:

* + 1. *20 GB of auto-saved disk space*
    2. *1 Nvidia Telsa P100 GPU*
    3. *2 CPU cores*
    4. *13 Gigabytes of RAM*

Lebih lengkapnya dapat dilihat di dokumentasi kaggle: <https://www.kaggle.com/docs/notebooks>.

### Libraries

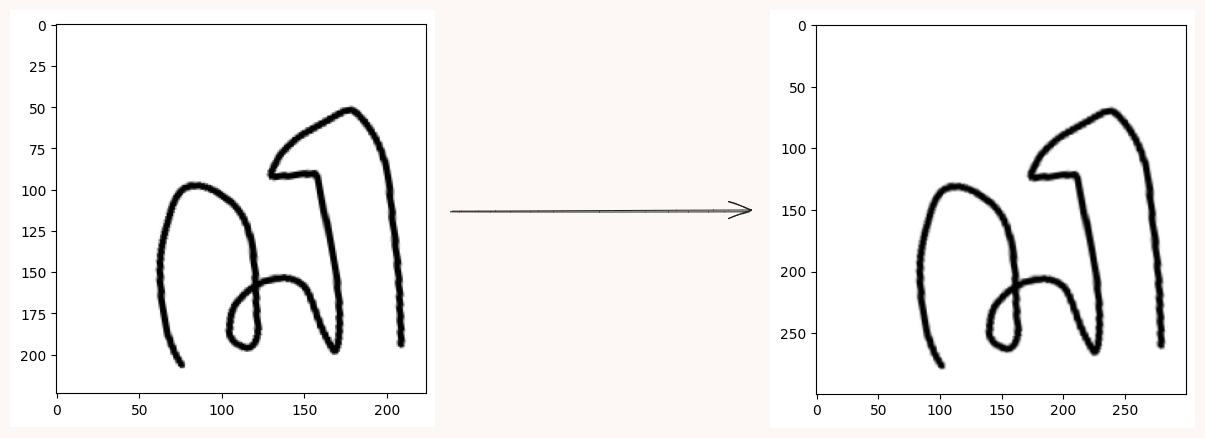
*Library* yang akan digunakan pada penelitian ini mencakup visualisasi, *preprocessing*, *modeling*, dan evaluasi. Tensorflow, Albumentations, Numpy, Pandas, Matplotlib, Plotly, Sklearn akan digunakan di dalam penelitian ini.

### Preprocessing

Dari data tersebut akan dilakukan *preprocessing* (lihat **Gambar 3.2**Alur Preprocessing) untuk mengoptimalkan pelatihan model yang digunakan nantinya. Pada penelitian ini, dilakukan berbagai skenario penelitian untuk melihat perbandingan hasil dari berbagai proses yang dilakukan. Pada bagian ini, augmentasi data akan dilakukan sesuai dengan skenario pengujian penulis.

#### Rescale/Resize

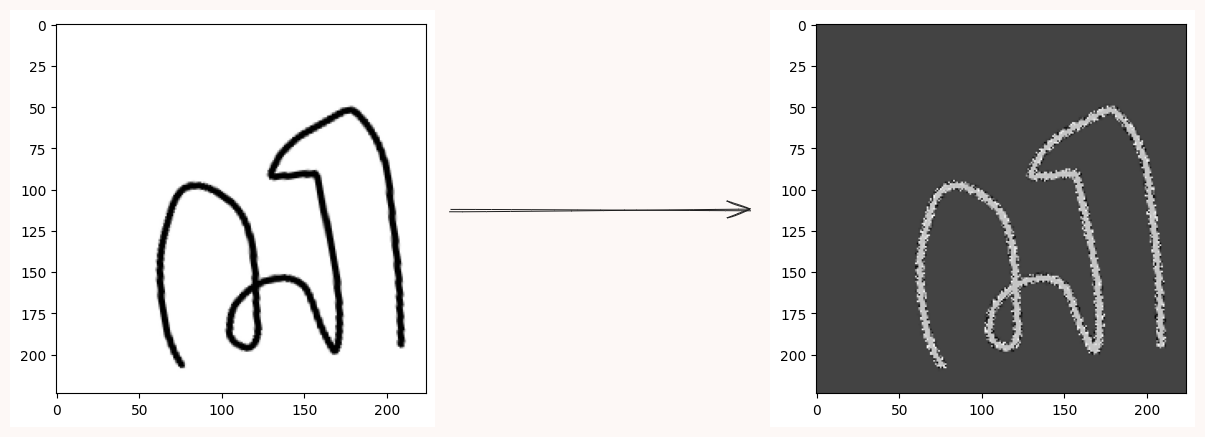
Dalam tahap ini citra akan melalui tahap *rescale* ke dalam ukuran 300 x 300. Hal ini dilakukan agar data yang akan digunakan untuk proses *training* seragam. Citra yang masuk ke dalam proses *training* juga diharuskan seragam, dalam data sebelumnya, beberapa citra mempunyai dimensi yang tidak sama dengan citra lainnya. Berikut adalah contoh rescale dari citra dimensi 224 x 224 menjadi 300 x 300.



**Gambar 3.8** Contoh rescale 224x224 menjadi 300x300

#### Normalisasi

Di tahap ini, penulis akan menggunakan teknik standardisasi pada citra. Hal ini dilakukan untuk mengoptimalkan dan menjaga kualitas data yang akan digunakan pada proses *training* nantinya. Berikut adalah contoh standardisasi data yang dilakukan dengan *library* tensorflow.

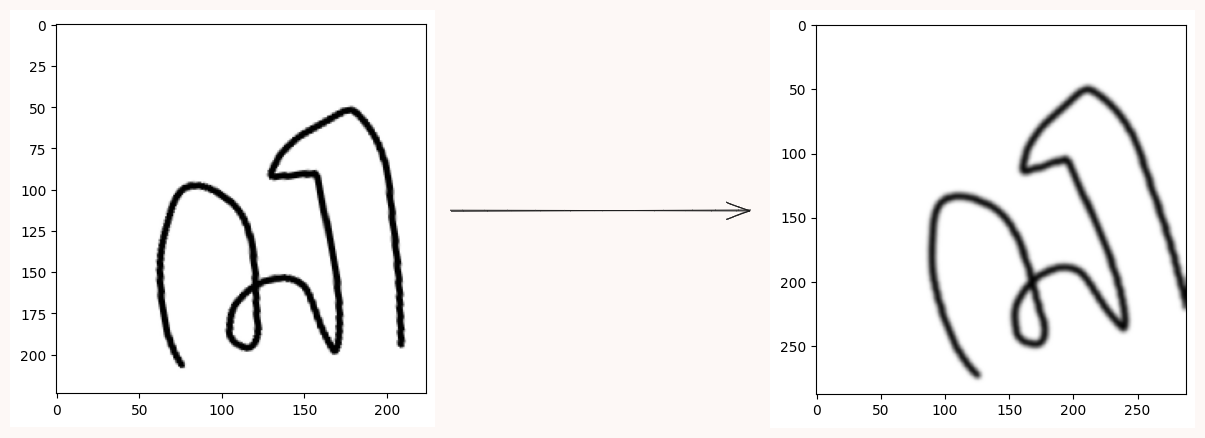


**Gambar 3.9** Contoh standardisasi

#### Augmentasi

Dalam tahap ini, dilakukan augmentasi data dengan *config* seperti pada penelitian sebelumnya terkait data citra teks (Rizky et al., 2023). Berikut confignya:

* *Rotation: 15°*
* *Image Scale: 0.9*
* *Blur Effect*



**Gambar 3.10** Contoh augmentasi

#### Data Splitting

Penulis akan melakukan tiga split menjadi *train, validation*, dan *test set* dari total data yang ada dengan perbandingan 0.9 : 0.05 : 0.05. Total data awal untuk pelatihan model adalah 4213, hal ini akan menghasilkan *data spli*t sebesar 3793, 210, 210 secara terurut dari *train, validation,* dan *test set*.

Data dari sumber ketiga akan sepenuhnya digunakan untuk uji coba model. Total data ketiga adalah 940.

### Modeling

Tahap *modeling* (lihat **Gambar 3.3**Alur Modeling, **Gambar 3.1** Alur Penelitian) akan dilakukan sesuai skenario pengujian yang tertera. Berikut detail dari tiap prosesnya.

#### Download Pre-trained Model

Semua pre-trained model ini akan diperoleh menggunakan *library* keras tensorflow. *Keras Applications* menyediakan berbagai *pre-trained* model yang dapat digunakan untuk penggunaan lebih lanjut seperti *prediction, feature extraction,* dan *fine-tuning*. Dokumentasi lebih lanjut dapat dilihat di <https://keras.io/api/applications/>. Model yang akan diunduh dari *keras application library* adalah VGG, Inception, dan Xception.

#### Apply Transfer Learning

Untuk melakukan transfer learning diperlukan kustomisasi pada layer yang diunduh dari *pre-trained* model keras *library*. Dalam beberapa penelitian sebelumnya, *transfer learning* dengan *pre-trained* model VGG dilakukan dengan menambahkan *fully connected layer* dan *output layer* dengan *activation function* softmax (Tammina, 2019).

...

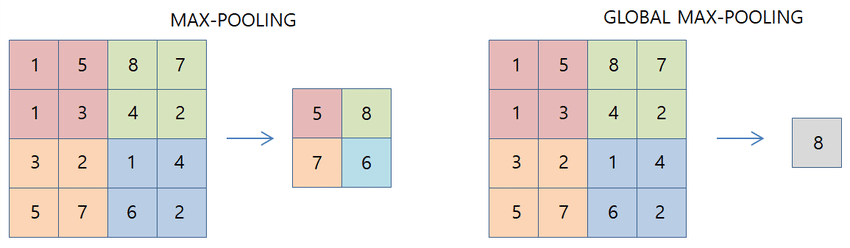
A picture containing text, screenshot, font, line

Description automatically generated

**Gambar 3.11** Contoh transfer learning

(Tammina, 2019)

Untuk menjaga kualitas model dan mencegah *overfitting,* digunakan juga *pooling layer* dan *dropout layer* pada lapisan *fully connected layer*. Ilustrasi dari kedua *layer* tersebut adalah sebagai berikut.



**Gambar 3.12** Contoh pooling

sumber: <https://www.researchgate.net/figure/The-difference-of-max-pooling-and-global-max-pooling_fig4_338079465>

A picture containing diagram, line, circle, origami

Description automatically generated

**Gambar 3.13** Contoh dropout

(Zaccone & Karim, 2018)

#### Fine Tuning

Definisi dari *fine-tune* menurut *Oxford Languages* adalah membuat penyesuaian kecil (sesuatu) untuk mencapai yang terbaik atau kinerja yang diinginkan. Dalam konteks *deep transfer learning*, hal ini dapat dicapai dengan melakukan *freezing layer, train fully connected layer*, maupun *train all the network* (Iman et al., 2022). Pelaksanaan *fine-tuning* lebih detailnya dapat dilihat pada tabel skenario pengujian.

#### **Train**

Pada proses ini, dilakukan pelatihan model *transfer learning* dengan data yang sudah dilakukan *preprocessing*. Di dalam *training*, digunakan data *train* dan *validation* untuk mengukur nilai *overfitting* yang terjadi.

### Evaluasi

Evaluasi akan dilakukan dengan metriks akurasi, sebagai tambahan digunakan juga *confusion matrix* untuk melihat kesalahan prediksi yang dilakukan oleh model yang sudah di-*train*. Berikut adalah contoh *confusion matrix* dengan 6 kelas prediksi. Dalam kasus penelitian ini, total kelas adalah 20, sehingga total baris dan kolom masing – masing adalah 20.

A picture containing screenshot, square, rectangle

Description automatically generated

**Gambar 3.14** Contoh confusion matrix

(Zaccone & Karim, 2018)

Percobaan model dengan data baru juga akan dilakukan untuk melihat kestabilan performa model (*generalization*). Rincian data dapat dilihat pada **Gambar 3.7** Data pengujian hasil pelatihan model atau bab IV, Percobaan Model dengan Data Baru.

## Skenario Pengujian

Dalam tahap penelitian akan dilakukan evaluasi terhadap skenario pengujian dengan mengimplementasikan variabel augmentasi dan *freezing* *layer* dengan tiga model berbeda yaitu VGG, Inception, dan Xception. *Config* dari parameter augmentasi akan disesuaikan dengan penelitian terdahulu yang sudah disinggung pada bab 3.3.6. Skenario secara garis besar adalah seperti tabel berikut. Total pengujiannya adalah 18.

**Tabel 3.1** Skenario Pengujian

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** |
| VGG | Yes, No | Full, ½, None | 0.001, 0.0001 | 1, 2 | Adam, SGD |
| Inception | Yes, No | Full, ½, None | 0.001, 0.0001 | 1, 2 | Adam, SGD |
| Xception | Yes, No | Full, ½, None | 0.001, 0.0001 | 1, 2 | Adam, SGD |

# BAB IV PENGUJIAN DAN ANALISIS

Bab ini membicarakan tentang tahap-tahap dalam proses pengenalan aksara Jawa dan hasil dari pengujian pengenalan aksara Jawa dengan menggunakan metode *Transfer Learning*.

## Detail Skenario Pengujian

Skenario pengujian ini dilaksanakan dengan mengimplementasikan model VGG, Inception, dan Xception, masing-masing dengan parameter yang berbeda. Untuk memperjelas arsitektur model transfer learning yang digunakan, berikut adalah gambaran umum mengenai masing-masing model.

A group of colorful rectangular objects

Description automatically generated with medium confidence

**Gambar 4.1** Arsitektur Skenario

Ketiga model, yaitu VGG, Inception, dan Xception, memiliki kesamaan dalam tiga lapisan terakhirnya, yang mencakup proses pooling, pengaturan ulang klasifikasi (tuned classifier), dan lapisan softmax. Perbedaan utama antara model-model tersebut terletak pada bagian Feature Extractor (Ekstraktor Fitur), yang sesuai dengan arsitektur yang telah dijabarkan pada bab sebelumnya.

## Parameter Pengujian

Penulis telah merinci sejumlah parameter yang akan diuji dalam penelitian ini, seperti yang tertera di **Tabel 3.1** Skenario Pengujian. Berdasarkan tabel tersebut, seluruh kombinasi yang ada telah diuji, menghasilkan total 144 skenario yang berbeda. Dalam proses pelatihan (*training*) masing-masing model, penulis juga telah mengimplementasikan *early stopping callback* yang berguna untuk mengehentikan pelatihan model (*training*) jika variabel yang dimonitor tidak mengalami peningkatan performa. Dalam kasus ini variabel tersebut adalah *validation loss*. Hal ini berguna untuk meminimalkan pemborosan sumber daya komputasi yang tidak perlu.

## Matriks Kinerja

Hasil pengujian setiap skenario mencakup informasi tentang waktu pelatihan dan akurasi yang dihasilkan. Untuk menentukan model terbaik dari keseluruhan rangkaian pengujian, penentuan dilakukan berdasarkan akurasi pada set data uji (*test set*) tertinggi, serta waktu pelatihan terpendek yang dicapai.

## Preprocessing Data

Tahap *preprocessing* pada data akan melalui berbagai proses yang telah disebutkan pada bab 3. Berikut adalah detail rinciannya.

### Augmentasi

Pada keseluruhan data penulis akan melakukan augmentasi untuk menambah jumlah data yang dapat digunakan di penelitian ini. *Config* atau pengaturan yang digunakan untuk melakukan augmentasi adalah rotasi, *scaling*, dan *blur*.

A screen shot of a computer code

Description automatically generated

**Gambar 4.2** *Source code config* augmentasi

Rinciannya terdapat pada gambar x, rotasi memiliki limit 15 yang berarti perubahan pada gambar baru akan mengalami rotasi sebanyak -15 – 15 derajat. *Random scale* mempunyai limit sebanyak 0.7, hal ini berarti gambar *source* akan mengalami penurunan kualitas dalam range 0.7 – 1. *Gaussian blur* mempunyai blur limit default sebanyak (3, 7) detail dari hal tersebut dapat dilihat pada sumber referensi *library* albumentations (Buslaev dkk., 2020). Variabel p mengindikasikan probabilitas *config* tersebut dapat tereksekusi. Jika nilai p=1 hal ini berarti *config* tersebut akan selalu dijalankan.

A screen shot of a computer program

Description automatically generated

**Gambar 4.3** *Source code* fungsi augmentasi

**Gambar 4.3** *Source code* fungsi augmentasi merupakan fungsi yang digunakan untuk melakukan augmentasi. Fungsi tersebut mempunyai 3 parameter yang mengandung informasi terkait config augmentasi, input folder, dan output folder. Di dalam fungsi ini, setiap satu gambar source akan diaugmentasi sebanyak 5 gambar berbeda. Hasil dari tiap gambar akan berbeda dikarenakan augmentasi dilakukan secara random berdasarkan limit yang telah ditentukan masing – masing config. Total sumber data yang telah dibersihkan adalah 3876 gambar dan setelah diaugmentasi, total data menjadi 19128 gambar.

### Preprocessing Pipeline

Proses preprocessing data selanjutnya akan dilakukan didalam sebuah fungsi pipeline. Berikut adalah cuplikan dari source code-nya.

A screen shot of a computer code

Description automatically generated

**Gambar 4.4** *Source code* fungsi *preprocessing pipeline*

Hal yang dilakukan berdasarkan **Gambar 4.4** *Source code* fungsi *preprocessing pipeline* adalah *resizing* dan standarisasi gambar. Dalam implementasi ini, variabel IMAGE\_SIZE adalah 300, yang berarti setiap gambar akan di-*resize* ke dalam dimensi 300 x 300 pixel.

## Modelling

### Pembuatan Model

Proses pembuatan model yang pertama kali adalah melakukan download *pre-trained* model yang dibutuhkan. Dengan *library* TensorFlow, hal ini dapat dilakukan dengan menulis kode saja, berikut adalah rinciannya.

A screenshot of a computer program

Description automatically generated

**Gambar 4.5** *Source code download pre-trained* *model* Xception

**Gambar 4.5** *Source code download pre-trained* *model* Xception menunjukkan source code yang diperlukan untuk mengunduh 3 model yang sama. Masing – masing model ini nantinya akan dirubah untuk dapat mengimplementasikan freeze layer yang sudah disebutkan pada bab 3. Pada model pertama, dilakukan inisialisasi trainable = False, ini berarti model tidak akan mengalami pelatihan ulang pada layer feature extractor. Sebaliknya untuk nilai True, model tersebut akan melalui proses pelatihan ulang bagian layer feature extractor.

Model – model tersebut akan ditambahkan layer yang lain (Pooling, Tuned Classifier, dan Softmax) untuk membuat arsitektur seperti yang telah disebutkan pada skenario simulasi.

A screenshot of a computer

Description automatically generated

**Gambar 4.6** *Summary Model* Xception *Full Freeze*

**Gambar 4.6** adalah salah satu contoh model dengan 2 tuned classifier. Model ini mengimplementasikan full freeze dengan feature extractor model Xception.

### Pelatihan Model

Model – model yang telah dibuat sebelumnya, akan dilatih dengan maksimum epoch sebanyak 50. Berikut merupakan cuplikan kode untuk melatih model.

A screenshot of a computer program

Description automatically generated

A number of steps and numbers

Description automatically generated with medium confidence

**Gambar 4.7** *Source code* beserta *output* pelatihan tiap *epoch*

Berdasarkan pada **Gambar 4.7,** pada setiap epoch, nilai accuracy, loss, validation accuracy, dan validation loss akan disimpan ke dalam sebuah csv yang nantinya akan digunakan untuk menganalisis performa model berdasarkan epoch yang telah dilalui. Proses pelatihan juga menggunakan ModelCheckpoint untuk menyimpan model dalam setiap epoch-nya, untuk menghindari hilangnya model jika terjadi crash. Waktu pelatihan model juga akan dihitung dengan fungsi timer yang telah dibuat penulis sebagai evaluasi performa model.

## Analisis Hasil Pengujian

Pelatihan setiap model telah terekam dan akan dianalisis dalam bentuk tabel seperti yang terlampir di bawah ini:

**Tabel 4.1** Hasil Pelatihan Model berdasarkan Skenario Pengujian

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** | **Train** | **Validation** | **Test** | **Test Loss** | **Time (min)** | **Epoch** |
| 1 | VGG | Yes | Full | 0.001 | 1 | SGD | 0.21 | 0.55 | 0.51 | 2.63 | 67 | 50 |
| 2 | VGG | Yes | ½ | 0.001 | 1 | SGD | 1.00 | 0.99 | 0.99 | 0.04 | 62 | 30 |
| 3 | VGG | Yes | None | 0.001 | 1 | SGD | 0.99 | 0.99 | 0.99 | 0.03 | 50 | 13 |
| 4 | VGG | Yes | Full | 0.0001 | 1 | SGD | 0.07 | 0.12 | 0.11 | 2.95 | 66 | 50 |
| 5 | VGG | Yes | ½ | 0.0001 | 1 | SGD | 0.97 | 0.98 | 0.98 | 0.07 | 107 | 50 |
| 6 | VGG | Yes | None | 0.0001 | 1 | SGD | 0.96 | 0.98 | 0.98 | 0.06 | 118 | 33 |
| 7 | VGG | Yes | Full | 0.001 | 2 | SGD | 0.23 | 0.53 | 0.49 | 2.38 | 65 | 50 |
| 8 | VGG | Yes | ½ | 0.001 | 2 | SGD | 0.99 | 0.99 | 0.98 | 0.07 | 59 | 28 |
| 9 | VGG | Yes | None | 0.001 | 2 | SGD | 0.99 | 0.99 | 0.98 | 0.09 | 72 | 19 |
| 10 | VGG | Yes | Full | 0.0001 | 2 | SGD | 0.07 | 0.22 | 0.23 | 2.92 | 67 | 50 |
| … | … | … | … | … | … | … | … | … | … | … | … | … |
| 141 | Xception | No | None | 0.001 | 2 | Adam | 0.99 | 1.00 | 0.98 | 0.13 | 15 | 14 |
| 142 | Xception | No | Full | 0.0001 | 2 | Adam | 0.80 | 0.82 | 0.75 | 0.87 | 16 | 50 |
| 143 | Xception | No | ½ | 0.0001 | 2 | Adam | 1.00 | 0.97 | 0.93 | 0.21 | 8 | 14 |
| 144 | Xception | No | None | 0.0001 | 2 | Adam | 1.00 | 1.00 | 0.98 | 0.04 | 14 | 15 |

Untuk memfasilitasi analisis, penulis menggunakan berbagai grafik seperti line dan bar chart. Analisis dilakukan secara komprehensif, meliputi ruang lingkup model serta dampak dari masing-masing parameter yang digunakan.

### Evaluasi Matriks Kinerja

A graph of different colored lines

Description automatically generated

**Gambar 4.8** Hasil keseluruhan skenario pengujian berdasarkan nilai akurasi test

**Gambar 4.8** menggambarkan visualisasi dari seluruh skenario yang telah diuji. Plotting dilakukan dengan nilai indeks pada tabel sebagai sumbu x (x-axis) dan akurasi uji (test accuracy) sebagai sumbu y (y-axis). Hasil plot menunjukkan bahwa beberapa model VGG menghasilkan akurasi uji yang stagnan, hanya mencapai sekitar 5% pada beberapa indeks. Penelitian mendalam menunjukkan bahwa model tersebut tidak mengalami peningkatan performa seiring berjalannya setiap epoch. Berdasarkan grafik x terlihat hasil dari pelatihan (training) salah satu model VGG tersebut.

A graph of a graph

Description automatically generated with medium confidence

**Gambar 4.9** Grafik Pelatihan Model VGG dengan stagnansi (*loss*)

A graph of a graph

Description automatically generated with medium confidence

**Gambar 4.10** Grafik Pelatihan Model VGG dengan stagnansi (*accuracy*)

Pada **Gambar 4.9** dan **Gambar 4.9**, terjadi stagnansi pada validation accuracy dna validation loss. Stagnansi pada pelatihan ini dapat disebabkan oleh ketidakcocokan parameter, dataset, atau arsitektur model yang digunakan dengan konteks penelitian ini (klasifikasi citra huruf tulis tangan aksara Jawa). Hasil skenario juga menunjukkan bahwa penggunaan optimizer Adam pada model VGG berpotensi menghasilkan stagnansi dalam pelatihan.

A graph of different colored bars

Description automatically generated

**Gambar 4.11** *Bar-chart* rerata akurasi tiap jenis model

Dari **Gambar 4.11**, dapat disimpulkan bahwa model VGG memiliki rata-rata akurasi terendah dibandingkan dengan model Inception dan Xception. Namun, perlu diingat bahwa hasil tersebut bisa saja terpengaruh oleh beberapa model yang mengalami stagnansi dalam pelatihan sehingga grafik yang tertera menjadi bias. Grafik di bawah menunjukkan jumlah model dari masing-masing tipe model yang berhasil mencapai akurasi uji di atas 50%.

A graph of different colored bars

Description automatically generated

**Gambar 4.12** *Bar-chart* rerata akurasi tiap jenis model dengan filter

A graph with blue rectangular bars

Description automatically generated with medium confidence

**Gambar 4.13** *Bar-chart* jumlah model terfilter

Berdasarkan grafik tersebut, terlihat bahwa performa model VGG tidak kalah dengan model Inception dan Xception. Namun, jumlah model yang berhasil mencapai kategori akurasi uji di atas 50% lebih banyak pada model Inception, yakni sebanyak 40 model.

A graph of different colored lines

Description automatically generated

**Gambar 4.14** *Bar-chart* waktu pelatihan tiap model

A graph of a graph with numbers and a number

Description automatically generated with medium confidence

**Gambar 4.15** *Bar-chart* rerata waktu pelatihan tiap jenis model

**Gambar 4.14** menunjukkan waktu yang diperlukan oleh seluruh model untuk menyelesaikan pelatihan mereka. Sementara itu, **Gambar 4.15** menggambarkan rata-rata waktu pelatihan dari masing-masing model. Berdasarkan informasi tersebut, dapat disimpulkan bahwa secara umum, model Inception memiliki waktu pelatihan tercepat dibandingkan dengan model VGG dan Xception. Berikut adalah beberapa contoh hasil uji skenario yang diurutkan berdasarkan kolom Time secara menaik (*ascending*):

**Tabel 4.2** Hasil uji skenario tiap model diurutkan berdasarkan kolom Time

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** | **Result Train** | **Validation** | **Test** | **Test Loss** | **Time (min)** | **Epoch** |
| 95 | Inception | No | ½ | 0.0001 | 2 | Adam | 1.00 | 0.98 | 0.95 | 0.17 | 3 | 11 |
| 86 | Inception | No | ½ | 0.001 | 1 | Adam | 0.99 | 0.99 | 0.99 | 0.05 | 3 | 11 |
| 44 | VGG | No | ½ | 0.001 | 2 | Adam | 0.05 | 0.05 | 0.05 | 3.00 | 4 | 8 |
| 139 | Xception | No | Full | 0.001 | 2 | Adam | 0.75 | 0.79 | 0.74 | 0.91 | 5 | 15 |
| 134 | Xception | No | ½ | 0.001 | 1 | Adam | 0.98 | 0.99 | 0.93 | 0.16 | 5 | 9 |
| … | … | … | … | … | … | … | … | … | … | … | … | … |

Hasil **Tabel 4.2** menunjukkan bahwa Inception adalah model yang paling cepat menyelesaikan proses pelatihan hanya dalam 11 *epoch* dengan total waktu pelatihan selama 3 menit.

### Evaluasi Parameter terhadap Matriks Kinerja

Dalam bagian ini, penulis akan menguraikan parameter-parameter yang digunakan serta dampak masing-masing parameter tersebut pada performa model, terutama dari segi waktu pelatihan dan akurasi set uji (*test set*).

A graph of different colored bars

Description automatically generated

**Gambar 4.16** *Bar-chart* performa akurasi berdasarkan parameter augmentasi

A graph with numbers and a number

Description automatically generated

**Gambar 4.17** *Bar-chart* performa waktu berdasarkan parameter augmentasi

Berdasar pada **Gambar 4.16** dan **Gambar 4.17**, parameter augmentasi ini memiliki dampak yang signifikan pada akurasi uji dan waktu pelatihan. Ketika augmentasi diterapkan, dampak terhadap waktu pelatihan menjadi lebih besar. Hal ini dapat dibuktikan dari grafik … terjadi perbedaan signifikan antara nilai parameter “Yes” dan “No”. Dalam kasus ini, Xception mengalami perbedaan waktu pelatihan paling banyak.

A graph of different colored bars

Description automatically generated

**Gambar 4.18** *Bar-chart* performa akurasi berdasarkan parameter *freeze*

A graph of numbers and a number

Description automatically generated

**Gambar 4.19** *Bar-chart* performa waktu berdasarkan parameter *freeze*

Berdasar pada **Gambar 4.18** dan **Gambar 4.19**, parameter *freeze* dengan nilai ‘None’ menunjukkan dampak yang paling signifikan terhadap akurasi uji pada model Inception dan Xception. Sebaliknya, model VGG mencapai akurasi tertinggi ketika parameter *freeze* diatur ke ‘Full’. Selain itu, parameter ini juga memiliki pengaruh signifikan terhadap durasi pelatihan model. Nilai ‘None’ menghasilkan durasi pelatihan yang paling lama, sementara nilai ‘½’ menghasilkan durasi pelatihan terpendek. Nilai ‘Full’ berada di antara keduanya. Durasi pelatihan yang lebih lama untuk nilai ‘None’ dapat dijelaskan oleh fakta bahwa model perlu melatih ulang semua *weight* dan *bias* dari setiap *layer* model.

A graph of different colored bars

Description automatically generated

**Gambar 4.20** *Bar-chart* performa akurasi berdasarkan parameter *learning rate*

A graph of numbers and a number of people

Description automatically generated with medium confidence

**Gambar 4.21** *Bar-chart* performa waktu berdasarkan parameter *learning rate*

Berdasar pada **Gambar 4.20** dan **Gambar 4.21**, parameter *learning rate* dengan nilai 0.001 menghasilkan akurasi yang lebih tinggi dibandingkan dengan nilai 0.0001. Namun, efek ini hanya berlaku untuk model Inception dan Xception, sementara VGG tidak mengalami perubahan yang signifikan dalam perbedaan nilai parameter ini. Hasil grafik menunjukkan bahwa perbedaan performa akurasi set uji hanya berbanding 1%. Terkait waktu pelatihan model, nilai learning rate 0.0001 menghasilkan waktu pelatihan yang lebih lama. Hal ini dikarenakan semakin kecil nilai *learning rate*, semakin lama pula proses konvergensi terjadi dalam tahap pelatihan model.

A graph of different colored bars

Description automatically generated

**Gambar 4.22** *Bar-chart* performa akurasi berdasarkan parameter *layer*

A graph of numbers and a number of people

Description automatically generated with medium confidence

**Gambar 4.23** *Bar-chart* performa waktu berdasarkan parameter *layer*

Berdasar pada **Gambar 4.22** dan **Gambar 4.23**, parameter *layer* tidak memberikan dampak yang signifikan pada akurasi uji maupun waktu pelatihan model untuk ketiga model tersebut.

A graph of different colored bars

Description automatically generated

**Gambar 4.24** *Bar-chart* performa akurasi berdasarkan parameter *optimizer*

A graph of numbers and a number

Description automatically generated with medium confidence

**Gambar 4.25** *Bar-chart* performa waktu berdasarkan parameter *optimizer*

Berdasar pada **Gambar 4.24** dan **Gambar 4.25**, parameter *optimizer* ini memiliki dampak signifikan pada akurasi uji dan waktu pelatihan model. Terdapat perbedaan antara model VGG dan dua model lainnya (Inception dan Xception). Model VGG mencapai hasil akurasi uji yang lebih baik dengan nilai parameter SGD dibandingkan dengan Adam. Sedangkan pada kedua model lainnya, Adam menghasilkan akurasi uji yang lebih baik. Selain itu, waktu pelatihan untuk nilai parameter SGD pada umumnya membutuhkan lebih banyak waktu dibandingkan dengan parameter Adam. Model Xception mengalami perbandingan tertinggi untuk waktu pelatihan model. Lama waktu pelatihan dengan nilai SGD hampir mencapai 3x waktu pelatihan dengan nilai parameter Adam.

A colorful squares with text

Description automatically generated with medium confidence

**Gambar 4.26** C*orrelation matrix* dari data hasil uji skenario

Didukung oleh matriks korelasi (correlation matrix) pada **Gambar 4.26**, temuan-temuan yang telah disebutkan sebelumnya menguatkan kesimpulan tersebut, membuktikan hubungan antara parameter-parameter tersebut dengan performa model. Sebagai salah satu contoh, variabel ‘Augmentasi\_Yes’ dan ‘Time (min)’ mempunyai nilai korelasi positif yang paling tinggi, yaitu sebesar 0,61. Artinya, penggunaan nilai “Yes” pada parameter Augmentasi akan memberikan dampak positif terhadap variabel Time (min). Dengan kata lain, waktu yang dibutuhkan untuk melatih model akan menjadi lebih panjang. Hal ini dapat diulas kembali pada gambar **Gambar 4.17** yang menunjukkan bahwa nilai perbandingan terbesar waktu pelatihan terletak pada parameter Augmentasi.

### Evaluasi Top 10 Models

Pada bagian ini, penulis akan mengulas 10 model terbaik untuk setiap jenis model berdasarkan akurasi uji (*test set*) mereka. Dengan adanya 3 jenis model yang berbeda, total model yang akan dievaluasi adalah 30 model yang berbeda. Tujuan dari analisis ini adalah untuk mendalami parameter-parameter yang paling berpengaruh dalam menciptakan model terbaik. Penulis akan menganalisis parameter-parameter yang dominan digunakan dalam total 30 model terbaik ini. Di bawah ini adalah daftar model yang masuk dalam kategori 10 model terbaik pada masing – masing jenis model.

**Tabel 4.3** Hasil uji skenario 10 model terbaik

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** | **Result Train** | **Validation** | **Test** | **Test Loss** | **Time (min)** | **Epoch** |
| 65 | Inception | Yes | None | 0.0001 | 1 | Adam | 0.9982 | 0.9961 | 1 | 0.002035 | 38 | 14 |
| 89 | Inception | No | None | 0.0001 | 1 | Adam | 0.9991 | 0.9952 | 1 | 0.004126 | 11 | 19 |
| 62 | Inception | Yes | None | 0.001 | 1 | Adam | 0.9956 | 1 | 0.996 | 0.00795 | 74 | 27 |
| 68 | Inception | Yes | None | 0.001 | 2 | Adam | 0.9949 | 0.9942 | 0.996 | 0.009453 | 52 | 19 |
| 50 | Inception | Yes | None | 0.001 | 1 | SGD | 0.999 | 0.9903 | 0.9921 | 0.02624 | 65 | 23 |
| 56 | Inception | Yes | None | 0.001 | 2 | SGD | 0.9996 | 0.9961 | 0.9901 | 0.03117 | 79 | 31 |
| 74 | Inception | No | None | 0.001 | 1 | SGD | 0.9988 | 0.9952 | 0.9901 | 0.05131 | 28 | 50 |
| 86 | Inception | No | None | 0.001 | 1 | Adam | 0.9882 | 0.9952 | 0.9901 | 0.04759 | 9 | 16 |
| 91 | Inception | No | ½ | 0.001 | 2 | Adam | 0.9962 | 0.9903 | 0.9901 | 0.03443 | 5 | 17 |
| 95 | Inception | No | None | 0.0001 | 2 | Adam | 0.9941 | 1 | 0.9901 | 0.03589 | 8 | 14 |
| 25 | VGG | No | ½ | 0.001 | 1 | SGD | 0.9838 | 0.9903 | 0.995 | 0.1023 | 22 | 44 |
| 40 | VGG | No | ½ | 0.0001 | 1 | Adam | 0.9971 | 1 | 0.995 | 0.01521 | 18 | 35 |
| 1 | VGG | Yes | ½ | 0.001 | 1 | SGD | 0.9952 | 0.9942 | 0.9921 | 0.04196 | 62 | 30 |
| 2 | VGG | Yes | None | 0.001 | 1 | SGD | 0.9871 | 0.9913 | 0.9921 | 0.02506 | 50 | 13 |
| 31 | VGG | No | ½ | 0.001 | 2 | SGD | 0.9471 | 0.9855 | 0.9901 | 0.0664 | 19 | 39 |
| 46 | VGG | No | ½ | 0.0001 | 2 | Adam | 0.9927 | 0.9952 | 0.9901 | 0.02785 | 24 | 50 |
| 11 | VGG | Yes | None | 0.0001 | 2 | SGD | 0.9454 | 0.9826 | 0.9881 | 0.04981 | 185 | 50 |
| 23 | VGG | Yes | None | 0.0001 | 2 | Adam | 0.9931 | 0.9961 | 0.9851 | 0.07386 | 66 | 19 |
| 26 | VGG | No | None | 0.001 | 1 | SGD | 0.9888 | 0.9807 | 0.9851 | 0.1034 | 26 | 28 |
| 7 | VGG | Yes | ½ | 0.001 | 2 | SGD | 0.9918 | 0.9932 | 0.9842 | 0.06673 | 59 | 28 |
| 134 | Xception | No | None | 0.001 | 1 | Adam | 0.9815 | 0.9855 | 1 | 0.01162 | 10 | 9 |
| 113 | Xception | Yes | None | 0.0001 | 1 | Adam | 0.9992 | 0.9981 | 0.997 | 0.01017 | 45 | 10 |
| 104 | Xception | Yes | None | 0.001 | 2 | SGD | 0.9991 | 0.9865 | 0.996 | 0.0247 | 221 | 50 |
| 116 | Xception | Yes | None | 0.001 | 2 | Adam | 0.9912 | 0.9971 | 0.9931 | 0.02588 | 44 | 10 |
| 119 | Xception | Yes | None | 0.0001 | 2 | Adam | 0.999 | 1 | 0.9921 | 0.0427 | 83 | 19 |
| 98 | Xception | Yes | None | 0.001 | 1 | SGD | 0.9994 | 0.9894 | 0.9911 | 0.03065 | 220 | 50 |
| 115 | Xception | Yes | ½ | 0.001 | 2 | Adam | 0.9949 | 0.9981 | 0.9911 | 0.02447 | 37 | 13 |
| 137 | Xception | No | None | 0.0001 | 1 | Adam | 0.9982 | 0.9903 | 0.9901 | 0.02281 | 11 | 11 |
| 110 | Xception | Yes | None | 0.001 | 1 | Adam | 0.9949 | 0.9971 | 0.9881 | 0.0348 | 52 | 12 |
| 109 | Xception | Yes | ½ | 0.001 | 1 | Adam | 0.992 | 0.9971 | 0.9871 | 0.041 | 23 | 8 |

Digambarkan secara *bar-chart*, berikut adalah performa masing – masing 10 model terbaik dalam nilai akurasi set uji (*test set*).

A graph of different colored bars

Description automatically generated with medium confidence

**Gambar 4.27** *Bar-chart* performa akurasi set uji 10 model terbaik

Berikut adalah analisis dampak masing – masing parameter terhadap waktu pelatihan model dan hasil akurasi set uji (*test set*) untuk top 10 models.

A graph with red and blue bars

Description automatically generated

**Gambar 4.28** *Bar-chart* jumlah model berdasarkan parameter augmentasi

Gambar 4.28 menggambarkan hasil 10 model terbaik untuk masing-masing jenis model beserta parameter augmentasi (Yes/No). Dapat diperhatikan bahwa 8 dari 10 model terbaik Xception menggunakan augmentasi. Untuk kedua model lainnya (VGG dan Inception), nilai parameter (Yes/No) diambil dengan rasio yang sama.

A graph with red and blue bars

Description automatically generated

**Gambar 4.29** *Bar-chart* jumlah model berdasarkan parameter *freeze*

Dari Gambar 4.29, tidak ada model yang menggunakan nilai "Full" untuk parameter ini. Mayoritas model terbaik menggunakan nilai "½" atau "None". Model terbaik Inception dan Xception rata-rata menggunakan nilai parameter "None", sementara model VGG lebih banyak menggunakan nilai "½." Hal ini menunjukkan bahwa *transfer learning* dengan model yang telah dilatih dari data ImageNet tidak selalu sesuai dengan tugas klasifikasi citra huruf tulisan tangan aksara Jawa. Salah satu alasannya adalah fitur-fitur yang diperoleh dari pelatihan dengan dataset citra ImageNet berbeda dengan citra tulisan tangan aksara Jawa.

A graph with red and blue bars

Description automatically generated

**Gambar 4.30** *Bar-chart* jumlah model berdasarkan parameter *learning rate*

Berdasarkan informasi **Gambar 4.30**, parameter *learning rate* dengan nilai 0.001 dominan digunakan dalam masing – masing 10 model terbaik.

A graph of red and blue bars

Description automatically generated

**Gambar 4.31** *Bar-chart* jumlah model berdasarkan parameter *layer*

Berdasarkan **Gambar 4.31**, rata – rata dari 10 model terbaik ini, dihasilkan dengan nilai *layer* 1, terutama untuk model Inception dan Xception. Untuk model VGG, masing – masing nilai *layer* (1 atau 2) memiliki jumlah count yang sama. Dengan kata lain, parameter ini tidak terlalu berdampak terhadap performa untuk model VGG.

A graph of different colored bars

Description automatically generated with medium confidence

**Gambar 4.32** *Bar-chart* jumlah model berdasarkan parameter *optimizer*

Sama seperti evulasi parameter keseluruhan model, *top 10 models* Inception dan Xception cenderung lebih cocok dengan nilai parameter *optimizer* Adam. Sedangkan rata – rata model VGG lebih cocok menggunakan *optimizer* SGD. Hal ini dapat diamati dari jumlah hitungan (*count*) masing-masing nilai parameter *optimizer* yang menunjukkan dominasi dibandingkan dengan nilai-nilai lainnya. Ini menunjukkan bahwa pemilihan *optimizer* memiliki peran penting dalam peningkatan performa model dan dapat berbeda-beda tergantung pada jenis model yang digunakan.

A graph of different colored lines

Description automatically generated

**Gambar 4.33** Grafik akurasi pelatihan 10 model terbaik

A graph with different colored lines

Description automatically generated

**Gambar 4.34** Grafik akurasi validasi pelatihan 10 model terbaik

Grafik dari **Gambar 4.33** dan **Gambar 4.34** merupakan hasil akurasi *training* dan *validation* untuk 10 model terbaik berdasarkan tiap epochnya. Dari grafik tersebut, tidak semua model menjalani proses *training* hingga *50 epoch*.

A graph with colorful lines and text

Description automatically generated

**Gambar 4.35** Grafik *loss* pelatihan 10 model terbaik

A graph with numbers and lines

Description automatically generated

**Gambar 4.36** Grafik *validation loss* pelatihan 10 model terbaik

Grafik yang digambarkan di atas (**Gambar 4.33Gambar 4.34Gambar 4.35**,**Gambar 4.36**) mewakili nilai akurasi dan *loss* dari proses pelatihan dan validasi dari masing-masing *top 10 models*. Terlihat jelas bahwa semua model terbaik telah mampu “belajar” secara efektif dari dataset yang digunakan untuk pelatihan. Hal ini dapat disimpulkan dari keseluruhan grafik tersebut, yang secara umum menunjukkan bahwa setiap model telah mengalami konvergensi, dengan peningkatan akurasi yang bertahap dan penurunan *loss* yang landai walaupun terjadi fluktuasi dalam akurasi validasi dan *loss* dalam epoch 1 – 25.

A graph of different colored bars

Description automatically generated

**Gambar 4.37** *Bar-chart epoch* masing – masing 10 model terbaik

Grafik di atas mewakili jumlah *epoch* yang dilakukan dalam proses pelatihan masing-masing *top 10 models*. Berdasarkan gambar tersebut, hanya lima model yang mencapai 50 *epoch*. Setiap model memiliki rata-rata *epoch* yang berbeda, menunjukkan kapabilitas masing-masing model dalam proses pelatihan. Sebagai contoh, model Xception sebagian besar hanya membutuhkan kurang dari 20 epoch untuk mencapai kinerja maksimal yang dapat diukur dari pengujian dengan dataset validasi. Namun, kapabilitas masing-masing model dalam pelatihan tidak selalu menentukan kinerja akhir model. Hal ini akan dibahas lebih lanjut pada bab 4.7, yang akan meninjau kinerja model dengan membuat prediksi dari data di luar sumber pelatihan model (pelatihan, validasi, uji).

A graph with green and orange lines

Description automatically generated

**Gambar 4.38** Grafikrerata akurasi 10 model terbaik tiap *epoch*

A graph with a line

Description automatically generated

**Gambar 4.39** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch*

A graph with different colored lines

Description automatically generated

**Gambar 4.40** Grafikrerata *loss* 10 model terbaik tiap *epoch*

A graph with a line graph

Description automatically generated

**Gambar 4.41** Grafikrerata *validation* *loss* 10 model terbaik tiap *epoch*

Grafik di atas menggambarkan rata-rata akurasi dan loss pada pelatihan dan validasi per *epoch*, yang dikelompokkan berdasarkan model. Pengamatan signifikan dari gambar ini terkait dengan model VGG. Model ini menunjukkan konvergensi yang lebih lambat dibandingkan dengan dua model lainnya (Inception dan Xception). Namun, pada akhirnya, model tersebut masih berhasil mencapai kinerja yang setara dengan model Inception dan Xception.

A graph with a line going up

Description automatically generated

**Gambar 4.42** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter augmentasi

A graph with blue and orange lines

Description automatically generated

**Gambar 4.43** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *freeze*

A graph with a line going up

Description automatically generated

**Gambar 4.44** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *learning rate*

A graph with a line

Description automatically generated

**Gambar 4.45** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *layer*

A graph with a line

Description automatically generated

**Gambar 4.46** Grafikrerata akurasi validasi 10 model terbaik tiap *epoch* berdasar nilai parameter *optimizer*

Grafik di atas (**Gambar 4.42Gambar 4.43Gambar 4.44Gambar 4.45Gambar 4.46**) mewakili dampak masing-masing parameter terhadap akurasi validasi selama *epoch* pelatihannya. Dampak yang dihasilkan dengan nilai yang berbeda dari masing-masing parameter hanya mempengaruhi proses pelatihan model di epoch 1 – 25. Setelah itu, kinerja yang dihasilkan oleh masing-masing model di *epoch* akhir adalah sama.

Penggunaan augmentasi mempercepat konvergensi di awal proses pelatihan model. Hal ini karena dataset yang digunakan untuk pelatihan di setiap *epoch* lebih besar dibandingkan dengan dataset yang tidak diaugmentasi. Nilai “½” pada *freeze* menunjukkan proses konvergensi yang lebih lambat dibandingkan dengan nilai “None”. Hal ini karena proses pembaruan bobot dalam model dilakukan secara menyeluruh untuk nilai “None” dan hanya setengah untuk nilai “½”. *Learning rate* 0.001 menunjukkan konvergensi yang lebih cepat. Hal ini umum, karena semakin kecil *learning rate*, semakin lama konvergensi terjadi dalam proses pelatihan model. *Layer* dengan nilai 1 menunjukkan konvergensi yang lebih cepat. Hal ini bergantung pada arsitektur model. Karena ada lebih banyak *layer* *dropout* pada parameter *layer* dengan nilai 2, hal ini mungkin terjadi. *Optimizer* SGD menunjukkan konvergensi yang lebih lambat pada tahap awal pelatihan model. Namun, di sisi lain, Adam mengalami fluktuasi dan penurunan kinerja pada *epoch* 10 – 20.

Analisis ini memberikan wawasan berharga tentang perilaku berbagai model di bawah berbagai parameter selama proses pelatihan. Penting untuk dicatat bahwa meskipun parameter tertentu dapat mempercepat proses konvergensi, hal tersebut mungkin tidak selalu mengarah ke kinerja yang lebih baik di epoch akhir. Oleh karena itu, pertimbangan dan penyetelan yang hati-hati terhadap tiap pameter sangat penting dalam mencapai kinerja model yang optimal.

## Percobaan Model dengan Data Baru

Pada bagian ini, beberapa model yang telah dibuat akan diuji performanya berdasarkan data baru di luar dataset yang digunakan dalam proses pelatihan model. Dengan uji coba ini, penulis berharap untuk memahami sejauh mana model yang telah dilatih dapat menangani data baru yang memiliki karakteristik yang berbeda. Hasilnya diharapkan dapat memberikan wawasan yang berharga dalam mengevaluasi performa model pada aplikasi dunia nyata.

### Parameter Pengambilan Model

Dalam uji coba ini, *top 10 models* yang telah dianalisis sebelumnya akan digunakan, ditambah dengan *top 3 models* lainnya berdasarkan parameter *freeze*. Pengambilan *top 3 models* ini bertujuan untuk memahami sejauh mana *transfer learning* dapat diimplementasikan dalam kasus ini. Berikut adalah grafik yang menunjukkan hasil akurasi uji pada *top 3 models* tersebut.

A graph of different colored bars

Description automatically generated with medium confidence

**Gambar 4.47** *Bar-chart* 3 model terbaik berdasarkan nilai tiap parameter *freeze*

Semua model dengan nilai “Full” pada parameter *freeze* tidak terlihat pada *top 10 models*. Namun, kinerja model dengan nilai parameter tersebut tidak terlalu buruk. Untuk model VGG, bisa mencapai akurasi 90%. Hal ini layak ditelusuri lebih lanjut terkait pengujian kinerja model dengan data baru.

### Data

Berikut adalah contoh data sampel baru yang akan digunakan untuk memprediksi citra huruf aksara Jawa. Data ini berbeda dari data yang digunakan selama pelatihan model karena merupakan hasil tulisan tangan di atas kertas. Sedangkan data pelatihan adalah citra tulisan tangan digital. Dengan uji coba ini, penulis bertujuan untuk mengevaluasi kemampuan kinerja model yang telah dilatih. Detail dari data adalah sebagai berikut.

* Data yang telah di-*scan*

A sheet of black and white writing

Description automatically generated

**Gambar 4.48** Data Aksara Jawa setelah di-*scan* dari kertas

* Data yang telah di-*crop*

A collage of different letters

Description automatically generated

**Gambar 4.49** Data Aksara Jawa setelah di-*crop* menggunakan perangkat lunak

* Data yang sudah melalui tahap *preprocessing*

A collage of multiple letters

Description automatically generated

**Gambar 4.50** Data Aksara Jawa setelah dilakukan *preprocessing*

* Beberapa contoh data yang dibuang (*cleaning*)

A screenshot of a phone

Description automatically generatedA screenshot of a computer screen

Description automatically generatedA screenshot of a signature

Description automatically generated

**Gambar 4.51** Data Aksara Jawa yang dihapus (*cleaning*)

### Hasil

Pengujian ini dilakukan menggunakan dua pendekatan, yaitu menggunakan *top 10 models* yang telah diidentifikasi sebelumnya dan dengan menggunakan *top 3 models* berdasarkan nilai parameter *freeze*. Berikut adalah rincian hasilnya dalam format tabel.

#### Menggunakan top 10 models

**Tabel 4.4** Rincian hasil percobaan baru menggunakan *top 10 models*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Loss** | **Accuracy** | **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** | **Test** | **Time (min)** | **Epoch** |
| 66 | 0.29 | 0.91 | Inception | Yes | None | 0.0001 | 1 | Adam | 1 | 38 | 14 |
| 63 | 0.56 | 0.88 | Inception | Yes | None | 0.001 | 1 | Adam | 0.996 | 74 | 27 |
| 111 | 0.54 | 0.85 | Xception | Yes | None | 0.001 | 1 | Adam | 0.9881 | 52 | 12 |
| … | … | … | … | … | … | … | … | … | … | … | … |
| 138 | 1.57 | 0.56 | Xception | No | None | 0.0001 | 1 | Adam | 0.9901 | 11 | 11 |
| 110 | 2.19 | 0.54 | Xception | Yes | ½ | 0.001 | 1 | Adam | 0.9871 | 23 | 8 |
| 75 | 1.99 | 0.43 | Inception | No | None | 0.001 | 1 | SGD | 0.9901 | 28 | 50 |

#### Menggunakan top 3 models berdasarkan nilai parameter freeze

**Tabel 4.5** Rincian hasil percobaan baru menggunakan *top 3 models*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Loss** | **Accuracy** | **Model** | **Augmentasi** | **Freeze** | **Learning Rate** | **Layer** | **Optimizer** | **Test** | **Time (min)** | **Epoch** |
| 66 | 0.29 | 0.91 | Inception | Yes | None | 0.0001 | 1 | Adam | 1 | 38 | 14 |
| 63 | 0.56 | 0.88 | Inception | Yes | None | 0.001 | 1 | Adam | 0.996 | 74 | 27 |
| 3 | 0.52 | 0.84 | VGG | Yes | None | 0.001 | 1 | SGD | 0.9921 | 50 | 13 |
| … | … | … | … | … | … | … | … | … | … | … | … |
| 70 | 4.11 | 0.10 | Inception | Yes | Full | 0.0001 | 2 | Adam | 0.7119 | 60 | 50 |
| 91 | 3.76 | 0.10 | Inception | No | Full | 0.001 | 2 | Adam | 0.7079 | 9 | 31 |
| 67 | 3.91 | 0.09 | Inception | Yes | Full | 0.001 | 2 | Adam | 0.7139 | 46 | 39 |

#### Analisis

A graph of blue and white bars

Description automatically generated

**Gambar 4.52** Rincian hasil percobaan baru menggunakan *top 10 models* dengan *bar-chart*

A graph of blue and white bars

Description automatically generated

**Gambar 4.53** Rincian hasil percobaan baru menggunakan *top 3 models* dengan *bar-chart*

*Bar-chart* yang digambarkan di atas (**Gambar 4.52** dan **Gambar 4.53**) mewakili kinerja akurasi dari setiap model saat diuji kembali dengan data baru. Dari grafik tersebut, tampak bahwa dua model teratas dari pengujian *top 10 models* dan *top 3 models* (*freeze*) berasal dari model yang sama yaitu dari indeks 66 dan 63. Perbedaan hasil pengujian model terletak pada posisi ketiga dan seterusnya. Hasil pengujian dengan *top 3 models* (*freeze*) tidak menunjukkan perbandingan akurasi yang signifikan terhadap *top 10 models*. Bahkan, rata-rata akurasi yang diperoleh dari *top 3 models* (*freeze*) lebih rendah dibandingkan dengan *top 10 models*. Ini menunjukkan bahwa model-model yang dibuat tanpa nilai “None” (“½”, “Full”) untuk parameter *freeze* tidak menghasilkan kinerja yang sebanding dengan model yang dibuat dengan nilai “None”. Akurasi tertinggi yang dapat dihasilkan oleh model tanpa nilai “None” untuk parameter *freeze* adalah 83%, yang berasal dari model VGG dengan true id 8 dan nilai parameter “½”. Berikut adalah analisis yang dilakukan pada *top 3 models* (*freeze*) dalam bentuk grafik untuk memperdalam performa masing-masing nilai parameter *freeze*.

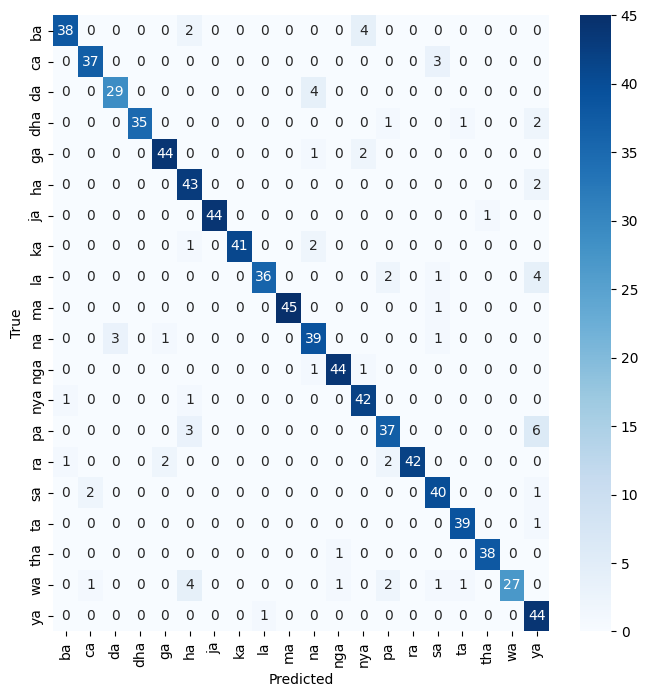
A graph of different colored bars

Description automatically generated

**Gambar 4.54** *Bar-chart* performa maksimum masing – masing model berdasarkan parameter *freeze*

Hasil yang diperoleh dari **Gambar 4.54** mencerminkan informasi tentang akurasi maksimum yang dapat dicapai dari prediksi menggunakan data baru, berdasarkan masing-masing nilai parameter *freeze* dan dikelompokkan berdasarkan modelnya. Nilai maksimum yang dicapai dari parameter *full* *freeze* hanya mencapai 31% pada model Xception. Hal ini memvalidasi informasi yang diperoleh dari evaluasi hasil pengujian sebelumnya, yaitu bahwa fitur yang diperoleh dari *pre-trained model* yang dilatih dengan data ImageNet kurang sesuai dengan fitur yang dibutuhkan dalam kasus penelitian ini (citra tulis tangan aksara Jawa). Dalam konteks parameter ½ *freeze*, model masih mampu mengenali 70-80% data baru dengan akurat. Potensi untuk meningkatkan akurasi model dengan konteks tersebut masih ada, mengingat model yang dilatih dalam penelitian ini dibatasi hanya sampai 50 *epoch*. Untuk parameter *none freeze*, model Inception menunjukkan akurasi maksimum 91%, sedangkan model lainnya hanya mencapai akurasi 84%.

Penulis akan melakukan visualisasi prediksi dan evaluasi untuk mengetahui data seperti apa yang membuat model salah prediksi. Berikut adalah beberapa gambar visualisasi dilakukan dengan model id 66 yang merupakan model terbaik dari prediksi dengan data baru.



**Gambar 4.55** *Confusion matrix* dengan model id 66

A collage of images of letters

Description automatically generated

**Gambar 4.56** Visualisasi prediksi 20 data secara acak

A collage of multiple images of symbols

Description automatically generated

**Gambar 4.57** Visualisasi prediksi 20 data yang salah secara acak

Setelah melakukan berbagai analisis berdasarkan **Gambar 4.55**, **Gambar 4.56**, dan **Gambar 4.57**. Penulis mengamati bahwa data yang diprediksi secara tidak tepat memiliki karakteristik sebagai berikut:

* **Tulisan aksara Jawa yang miring**, yang menyulitkan model untuk mengenali beberapa bentuk dan posisi huruf.
* **Tulisan yang tidak ditengah dan terlalu kecil**, yang menyebabkan model untuk mengabaikan sebagian informasi pada gambar.
* **Tulisan yang tidak sesuai dengan tata penulisan aksara Jawa**, yang menimbulkan ambiguitas dan kesalahan dalam pengenalan huruf.

Secara garis besar, kesalahan prediksi pada suatu data terjadi dikarenakan ketidaksesuaian yang signifikan data tersebut terhadap data latih yang digunakan pada tahap pelatihan model. Untuk mengatasi masalah ini, penulis menyarankan untuk meningkatkan *preprocessing pipeline*, misalnya dengan menambahkan algoritma *cropping* yang dapat menyesuaikan ukuran dan posisi huruf secara otomatis. Selain itu, penulis juga menyarankan untuk mengevaluasi dan menyesuaikan *config* augmentasi, agar data *training* lebih mencerminkan variasi dan kondisi data *testing* di dunia nyata. Secara umum, penulis berpendapat bahwa peningkatan performa model dapat dicapai dengan menambahkan data baru yang lebih beragam dan representatif sesuai dengan data di dunia nyata.

# BAB V PENUTUP

## Kesimpulan

Berdasarkan serangkaian proses pengujian dan analisis yang telah dilakukan pada bab sebelumnya, dapat ditarik beberapa kesimpulan sebagai berikut:

* + 1. Dari ketiga model yang diuji, yaitu VGG16, Inception V3, dan Xception, model Inception V3 menunjukkan performa terbaik secara keseluruhan ditinjau dari sisi akurasi *testing*, waktu komputasi (38 menit), dan kestabilan hasil.
    2. Nilai akurasi tertinggi yang dicapai mencapai 100% pada data *testing*. Namun, ketika diuji dengan data baru di luar data latih, akurasi turun hingga sekitar 90%. Hal ini tidak buruk, mengingat pelatihan model dilakukan dengan data citra huruf tulis tangan aksara Jawa digital, sedangkan untuk pengujian model dengan data baru, berasal dari data citra tulis tangan aksara Jawa non-digital.
    3. Nilai parameter *freeze* yang paling cocok untuk kasus ini adalah “None” yaitu melatih kembali keseluruhan arsitektur model di bagian *feature extractor*. Hal ini bisa terjadi dikarenakan fitur yang didapat pada *pre-trained model* kurang sesuai.
    4. Parameter augmentasi data, pembekuan lapisan model (*freeze layer*), *learning rate*, dan *optimizer* memiliki pengaruh yang signifikan terhadap performa model dari segi akurasi dan waktu.
    5. Secara garis besar, kesalahan prediksi pada suatu data terjadi dikarenakan ketidaksesuaian yang signifikan data tersebut terhadap data latih yang digunakan pada tahap pelatihan model

## Saran

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran untuk pengembangan lebih lanjut, yaitu:

1. Melakukan penelitian lebih lanjut untuk mencari sebab kesalahan prediksi yang dilakukan oleh model.
2. Memperbanyak variasi data latih dengan menggunakan lebih banyak sumber tulisan tangan dari penulis yang berbeda. Data latih sebaiknya mencakup berbagai gaya tulisan dan kondisi pencahayaan.
3. Melakukan augmentasi data secara lebih luas, seperti *shear* dan *translation* untuk merepresentasikan data dunia nyata.
4. Mengeksplorasi *state of the art* (SOTA) *image classification model* yang lebih mutakhir seperti Vision Transformer dan BASIC-L.
5. Melakukan *hyperparameter tuning optimization* dengan grid search, random search, bayesian search, atau lainnya untuk menemukan konfigurasi parameter yang optimal.
6. Mengembangkan model deteksi untuk mendeteksi lokasi huruf sebelum klasifikasi agar dapat menangani input gambar yang lebih kompleks.
7. Mengimplementasikan model pada platform *mobile* atau web agar dapat dimanfaatkan pengguna secara luas.

# DAFTAR PUSTAKA

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Research, G. (2016). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. https://arxiv.org/abs/1603.04467v2

Abdul Robby, G., Tandra, A., Susanto, I., Harefa, J., & Chowanda, A. (2019). Implementation of Optical Character Recognition using Tesseract with the Javanese Script Target in Android Application. *Procedia Computer Science*, *157*, 499–505. https://doi.org/10.1016/J.PROCS.2019.09.006

Ahmed, R. M., Rashid, T. A., Fattah, P., Alsadoon, A., Bacanin, N., Mirjalili, S., Vimal, S., & Chhabra, A. (2022). Kurdish Handwritten character recognition using deep learning techniques. *Gene Expression Patterns*, *46*, 119278. https://doi.org/10.1016/J.GEP.2022.119278

Burkov, A. (2019). The Hundred-Page Machine Learning Book. Andriy Burkov.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and flexible image augmentations. *Information (Switzerland)*, *11*(2). https://doi.org/10.3390/INFO11020125

Chandrarathne, G., Thanikasalam, K., & Pinidiyaarachchi, A. (2020). A Comprehensive Study on Deep Image Classification with Small Datasets. *Lecture Notes in Electrical Engineering*, *619*, 93–106. https://doi.org/10.1007/978-981-15-1289-6\_9/COVER

Chollet, F. (2016). Xception: Deep Learning with Depthwise Separable Convolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-January*, 1800–1807. https://doi.org/10.1109/CVPR.2017.195

Chollet, F. (2021). Deep Learning with Python (2nd ed.). Manning Publications Co.

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2010). *ImageNet: A large-scale hierarchical image database*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Everson, M. (2008, March 6). Proposal for encoding the Javanese script in the UCS (N3319R3) [PDF]. ISO/IEC JTC1/SC2/WG2. Unicode.

G., A. R., Tandra, A., Susanto, I., Harefa, J., & Chowanda, A. (2019). Implementation of Optical Character Recognition using Tesseract with the Javanese Script Target in Android Application. Procedia Computer Science, 499-505.

Gonzalez, R. C., & Woods, R. E. (2018). Digital Image Processing (4th ed., Global ed.). Harlow, England: Pearson Education Limited.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Iman, M., Rasheed, K., & Arabnia, H. R. (2022). A Review of Deep Transfer Learning and Recent Advancements. *Technologies*, *11*(2), 40. https://doi.org/10.3390/technologies11020040

Kesaulya, G. N. A., Fariza, A., & Karlita, T. (2022). Javanese Script Text Image Recognition Using Convolutional Neural Networks. *IES 2022 - 2022 International Electronics Symposium: Energy Development for Climate Change Solution and Clean Energy Transition, Proceeding*, 534–539. https://doi.org/10.1109/IES55876.2022.9888527

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature 2015 521:7553*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2022). A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Perez, F., Vasconcelos, C., Avila, S., & Valle, E. (2018). Data Augmentation for Skin Lesion Analysis. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11041 LNCS*, 303–311. https://doi.org/10.1007/978-3-030-01201-4\_33

Poerwadarminta, W.J.S. (1939). Baoesastra Djawa (dalam bahasa Jawa). J.B. Wolters.

Ranjan, C. (2019). Understanding Deep Learning: Application in Rare Event Prediction. Chitta Ranjan.

Rizky, A. F., Yudistira, N., & Santoso, E. (2023). *Text recognition on images using pre-trained CNN*. https://arxiv.org/abs/2302.05105v1

Saponara, S., & Elhanashi, A. (2022). Impact of Image Resizing on Deep Learning Detectors for Training Time and Model Performance. *Lecture Notes in Electrical Engineering*, *866 LNEE*, 10–17. https://doi.org/10.1007/978-3-030-95498-7\_2/COVER

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. https://arxiv.org/abs/1409.1556v6

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June-2015*, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*, 2818–2826. https://doi.org/10.1109/CVPR.2016.308

Tammina, S. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. *International Journal of Scientific and Research Publications (IJSRP)*, *9*(10), p9420. https://doi.org/10.29322/IJSRP.9.10.2019.P9420

Torrey, L., & Shavlik, J. (2009). Chapter: Transfer Learning. Hershey, Pennsylvania: IGI Global.

U, M. K., Sihabuddin, A., & S.Si., M.Kom., Dr. (n.d.). Transfer learning implementation on Sundanese script recognition using convolutional neural network (Bachelor’s thesis).

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, *4*(January), 3320–3328. https://arxiv.org/abs/1411.1792v1

Zaccone, G., & Karim, M. R. (2018). Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python (2nd ed.). Birmingham, United Kingdom: Packt Publishing.