# TRANSLITERASI AKSARA *JAWOE*-LATIN MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *XCEPTION*

**TUGAS AKHIR**

**Diajukan oleh:**

**AESHA DURRATUL NASIHAH**
**NIM. 200705003**
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

**FAKULTAS SAINS DAN TEKNOLOGI**
**UNIVERSITAS ISLAM NEGERI AR-RANIRY**
**BANDA ACEH**
**2023 M/1444 H**

# TRANSLITERASI AKSARA *JAWOE*-LATIN MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *XCEPTION*

**TUGAS AKHIR**

Diajukan kepada Fakultas Sains dan Teknologi
Universitas Islam Negeri Ar-Raniry Banda Aceh
Sebagai Beban Studi Memperoleh Gelar Sarjana Dalam Ilmu Teknologi Informasi


Oleh:

**AESHA DURRATUL NASIHAH**
**NIM. 200705003**
**Mahasiswa Fakultas Sains dan Teknologi UIN Ar-Raniry**
**Program Studi Teknologi Informasi**


Disetujui Untuk Dimunaqasyahkan Oleh :

| Pembimbing I, | Pembimbing II, |
|---|---|
| Bustami, M.Sc | Khairan AR, M.Kom. |
| NIP. 198604082014031001 | NIP. 198607042014031001 |

Mengetahui

Ketua Program Studi Teknologi Informasi

Ima Dwitawati, MBA.
NIP. 198210132014032002

# TRANSLITERASI AKSARA *JAWOE*-LATIN MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *XCEPTION*

**TUGAS AKHIR**

Telah Diuji Oleh Panitia Ujian Munaqasah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S-1)
Dalam Ilmu Teknologi Informasi

Pada Hari/Tanggal: Jumat, 22 Desember 2023 M
9 Jumadil Akhir 1445 H

Panitia Ujian Munaqasyah Tugas Akhir:

| | |
|---|---|
| Ketua, | Sekretaris, |
| **Bustami, M.Sc** | **Khairan AR, M.Kom** |
| **NIP. 198604082014031001** | **NIP. 198607042014031001** |
| Penguji I, | Penguji II, |
| **Rika Yuliana, M.T** | **Hendri Ahmadian, S.Si., M.I.M** |
| **NIP. 198407132014032001** | **NIP. 198301042014031002** |

Mengetahui:
Dekan Fakultas Sains dan Teknologi
UIN Ar-Raniry Banda Aceh,

**Dr. Ir. Muhammad Dirhamsyah, M.T., IPU**
**NIP. 196210021988111001**

# LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Aesha Durratul Nasihah

NIM : 200705003

Program Studi : Teknologi Informasi

Fakultas : Sains dan Teknologi

Judul : Transliterasi Aksara *Jawoe*-Latin Menggunakan *Metode Convolutional Neural Network* Dengan Arsitektur *Xception*

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggungjawabkan;
2. Tidak melakukan plagiasi terhadap naskah karya orang lain;
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya;
4. Tidak memanipulasi dan memalsukan data;
5. Mengerjakan sendiri karya ini dan mampu bertanggungjawab atas karya ini.

Bila dikemudian hari ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat dipertanggungjawabkan dan ternyata memangditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya siap dikenai sanksi berdasarkan aturan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

Demikian pernyataan ini saya buat dengan sesungguhnya dan tanpa paksaan dari pihak manapun.

Banda Aceh, 22 Desember 2023

Yang Menyatakan,


Aesha Durratul Nasihah

# ABSTRAK


Nama : Aesha Durratul Nasihah

NIM : 200705003

Program Studi : Teknologi Informasi


Judul : Transliterasi Aksara *Jawoe*-Latin Menggunakan *Metode Convolutional Neural Network* Dengan Arsitektur *Xception*


Tanggal Sidang : 22 Desember 2023

Jumlah Halaman : 91 Halaman

Pembimbing I : Bustami, M.Sc.

Pembimbing II : Khairan Ar, M.Kom



*Convolutional Neural Network (CNN)* merupakan salah satu metode yang terdapat pada d*eep learning* yang merupakan metode terbaik dalam mengatasi persoalan mengenai *image classification*. Metode ini banyak menghasilkan arsitektur yang dapat digunakan dalam beberapa bentuk data, salah satunya penggunaan citra aksara *Jawoe* dalam proses transliterasi. Pada penelitian ini menggunakan arsitektur *Xception* yang berfokus pada 475 kelas yang mencakup kata-kata *Jawoe*. Berdasarkan pembahasan dan hasil analisis yang telah dilakukan, diperoleh *accuracy* 67.60% dengan menggunakan 90 *epoch*. Dari penelitian ini dapat disimpulkan bahwa model dengan arsitektur *Xception* membutuhkan penggunaan data yang bervariasi untuk memberikan tingkat keakuratan yang tinggi.



**Kata Kunci: Aksara *Jawoe,* CNN, Arsitektur *Xception* dan *Accuracy***

# ABSTRACT


| | |
|---|---|
| Name | : Aesha Durratul Nasihah |
| NIM | : 200705003 |
| Department | : Information Technology |
| Title | : Transliteration of Jawoe-Latin Script Using Convolutional Neural Network Method with Xception Architecture |
| Date | : 22 December 2023 |
| Number of Pages | : 91 Pages |
| Supervisor I | : Bustami, M.Sc. |
| Supervisor II | : Khairan Ar, M.Kom |



Convolutional Neural Network (CNN) is one of the methods found in deep learning which is the best method for solving problems regarding image classification. This method produces many architectures that can be used in several forms of data, one of which is the use of Jawoe script images in the transliteration process. This research uses the Xception architecture which focuses on 475 classes that include Jawoe words. Based on the discussion and results of the analysis that has been carried out, an accuracy of 67.60% was obtained using 90 epochs. From this research it can be concluded that models with the Xception architecture require the use of varied data to provide a high level of accuracy.



**Keyword: *Jawoe* Script, CNN, Xception Architecture and Accuracy**

# KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Segala puji Syukur atas kehadirat Allah SWT yang telah memberikan petunjuk dan hidayah-Nya, karena penulis tidak akan mampu menyelesaikan penulisan tugas akhir dengan baik tanpa kehendak-Nya. Shalawat dan salam turut di sanjungkan kepada Rasul kita Nabi Muhammad SAW, yang telah membawa kita dari alam jahiliyah kea lam Islamiyah yang penuh dengan ilmu pengetahuan, seperti yang kita rasakan saat ini

Alhamdulillah dengan izin Allah SWT, penulis dapat menyelesaikan tugas akhir yang berjudul **"Transliterasi Aksara *Jawoe*-Latin Menggunakan Metode *Convolutional Neural Network* Dengan Arsitektur *Xception*"** Dengan harapan penulis bahwa tugas akhir ini dapat memberikan manfaat bagi pihak yang membutuhkan sehingga mampu menambahkan wawasan serta ilmu pengetahuan.

Keberhasilan dalam menyelesaikan tugas akhir ini tidak terlepas dari bimbingan yang telah diberikan dari berbagai pihak. Penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada pihak-pihak yang terlibat secara langsung dalam penulisan tugas akhir ini baik berupa dukungan, doa maupun bimbingan yang telah diberikan. Oleh karena itu, pada kesempatan kali ini penulis ingin menyampaikan rasa hormat dan terima kasih yang sebesar-besarnya kepada:

1. Orang tua terhebat yang penulis cintai, Bapak Azhari dan Ibu Cut Erlina yang telah mendo'akan serta memberikan semangat, kasih sayang yang tiada henti kepada penulis.
2. Bustami, M.Sc dan Khairan AR, M.Kom selaku pembimbing I dan II yang telah mencurahkan waktu, pikiran dan tenaga dalam membimbing penulis demi kesempurnaan skripsi ini. Terima kasih banyak penulis ucapkan, semoga Bapak dan Ibu selalu mendapat rahmat dan lindungan Allah SWT.
3. Ima Dwitawati, M.B.A dan Khairan AR, M.Kom selaku Ketua dan Sekretaris Program Studi Teknologi Informasi Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

4. Bustami, M.Sc selaku Penasehat Akademik (PA) penulis selama menempuh pendidikan di Program Studi Teknologi Informasi. Terima kasih banyak telah memberi nesehat dan saran selama ini kepada penulis.
5. Cut Ida Rahmadiana S,Si. Selaku staf Prodi Teknologi Informasi yang telah membantu penulis dalam hal administrasi selama menempuh pendidikan di Program Studi Teknologi Informasi
6. Dr. Ir. Muhammad Dirhamsyah, M.T., IPU selaku Dekan Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.
7. Bapak dan ibu dosen Program Studi Teknologi Informasi yang telah memberikan ilmu pengetahuan yang sangat berguna bagi penulis selama proses belajar mengajar.
8. Sahabat-sahabat Teknologi Informasi yang telah berjuang bersama, berbagi semangat dan suka duka dalam penyelesaian tugas akhir ini. Penulis sangat berterima kasih atas motivasi dan semangat kalian semua.

Penulis menyadari sepenuhnya bahwa dalam penyusunan tugas akhir ini tdak cukup dikategorikan sempurna, untuk itu penulis dengan segala kerendahan hati menerima saran dan kritikan guna menyempurnakan penyusunan tugas akhir ini. Akhir kata, semoga tugas akhir ini dapat bermanfaat bagi penulis dan pembaca dan semoga dicatat sebagai sebuah amal kebaikan oleh Allah SWT.
Amiin Ya Rabbal A'lamin.

Banda Aceh, 12 Desember 2023
Penulis,

Aesha Durratul Nasihah

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# DAFTAR LAMPIRAN

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Bahasa Arab Melayu atau aksara Jawi adalah salah bentuk Bahasa yang digunakan oleh bangsa Melayu dengan memadukan seluruh karakter aksara Arab dengan menambahkan aksara lain yang tidak termasuk dalam huruf Arab seperti huruf چ، غ، ڤ، ک، ڽ، ز untuk melengkapi huruf dalam bahasa Arab (Ramala, 2020). Aksara Jawi itu sendiri tidak menggunakan tanda harakat pada penulisannya seperti tanda fattah, kasrah, tanwin dan juga dhammah. Keberadaan Aksara Jawi itu penting dalam membantu memahami manuskrip-manuskrip dahulu yang telah ditulis oleh penulis-penulis dahulu.

Aksara Jawi digunakan sebagai Bahasa dalam berkomunikasi sejak zaman dahulu. Meskipun ditulis dalam aksara Arab, tetapi dalam pembacaannya menggunakan tata Bahasa Indonesia (Irawan, 2018). Banyak manuskrip zaman dahulu yang berisikan kisah-kisah dahulu, budaya dan ilmu pengetahuan yang dituliskan dengan menggunakan aksara Jawi. Manuskrip-manuskrip ini tersimpan diberbagai repositori negara asing seperti di Jerman, Perancis dan lain sebagainya. Menurut KBBI, Manuskrip adalah naskah tulisan tangan yang menjadi kajian filologi atau naskah baik tulisan tangan (dengan pena atau pensil) maupun ketikan (buku ketikan).

Bahasa Jawi tidak hanya menggunakan tata Bahasa Indonesia, tetapi ada juga menggunakan tata Bahasa Aceh dalam pengucapnnya. Ini dikenal dengan istilah *Harah Jawoe* atau aksara Arab *Jawoe*. Aksara *Jawoe* ini digunakan oleh rakyat Aceh dalam menulis hikayat-hikayat sejak zaman dahulu. Salah satu contohnya yaitu Hikayat Aceh yang diperkenalkan oleh Juynboll (Hermansyah, 2020).

Manuskrip aceh memiliki kesulitan dalam membacanya, dikarenakan kurangnya penelitian yang membahas dan menelaah mengenai isi dari manuskrip

kuno. Dari penelitian-penelitian yang ada, hanya sedikit penelitian yang menelaah mengenai isi dari manuskrip. Sehingga hanya orang-orang yang memiliki ilmu filologi yang bisa membaca isi dari manuskrip tersebut. Ini juga dipengaruhi oleh tiga bahasa yang berbeda yang digunakan dalam manuskrip yaitu Bahasa Aceh, Bahasa Arab dan Bahasa Melayu. Penggunaan tiga bahasa ini dikaitkan dengan penggunaannya dalam manuskrip. Dalam manuskrip, ditulis dengan menggunakan bahasa Arab Melayu, sedangkan untuk pembacaannya menggunakan Bahasa Aceh. Sehingga dibutuhkan sebuah model yang memiliki pengetahuan untuk menerjemahkan atau transliterasi dari aksara *Jawoe* kedalam Bahasa latin untuk mempermudah orang-orang dalam membaca manuskrip-manuskrip lama yang dituliskan dalam aksara *Jawoe*.

Transliterasi dilakukan dengan mengubah tata bahasa yang dimiliki oleh suatu kata atau kalimat kedalam tata bahasa lain yang dipahami. Aksara atau kata pada proses transliterasi akan diubah kedalam bentuk lainnya sesuai dengan bentuk pengucapan atau bunyinya dari kata asalnya. Umumnya, transliterasi digunakan sebagai media untuk membantu memberikan pemahaman atau ejaan kata pada bahasa lain yang sulit dipahami kedalam bentuk ejaan bahasa lain. Proses transliterasi dapat dilakukan dengan menggunakan metode-metode tertentu yang mampu memberikan pengetahuan terhadap model.

Terdapat beberapa penelitian terdahulu mengenai transliterasi aksara seperti aksara Sawa ke latin, aksara Sunda ke latin, aksara Jawi ke latin dengan menggunakan beberapa metode, seperti algoritma *Freeman Chain Code* dan *Support Vector Machine* (Irawan, 2018), *Local Binary Pattern* (Christy Atika Sari, Wellia Shinta Sari (2022), *Convolutional Neural Network* (Shelvi Nur Rahmawati). Dari beberapa penelitian tersebut menunjukkan bahwa penelitian yang dilakukan dengan menggunakan metode *Convolutional Neural Network* memiliki hasil akurasi yang tertinggi dibandingkan dengan penelitian yang menggunakan *Support Vector Machine* dan *Local Binary Pattern*, yaitu sebesar 92%.

Selain itu, dalam penelitian yang menggunakan metode *Convolutional Neural Network* terdapat berbagai jenis arsitektur yaitu *MobileNet* (Mega Ellyadi, 2022) , *Xception* (Muhammad Ridha, 2022), *Baseline CNN* (Aulia Sabri, 2022), dan

*VGG19* (Marcella dan Devella, 2022) dan sebagainya. Dari beberapa penelitian tersebut menunjukkan dua arsitektur yang memiliki akurasi tertinggi yaitu *Xception*. Sehingga dalam penelitian ini, akan dilakukan perbandingan akurasi pada *Convulutional Neural Network* dengan arsitektur *Xception* dalam melakukan transliterasi aksara *Jawoe* ke latin.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang diuraikan diatas mengenai transliterasi aksara *Jawoe* ke latin, sehingga dirumuskan masalah yaitu sebagai berikut :

1. Bagaimana proses transliterasi aksara *Jawoe* ke dalam bahasa latin dengan menggunakan metode CNN dan arsitektur *Xception*?
2. Bagaimana tingkat akurasi pada transliterasi aksara *Jawoe* ke Bahasa Latin dengan menggunakan metode CNN dan arsitektur *Xception*?

## 1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini yaitu sebagai berikut :

1. Untuk mengetahui proses transliterasi aksara *Jawoe* ke dalam bahasa latin dengan menggunakan metode CNN dan arsitektur *Xception*.
2. Untuk mengetahui tingkat akurasi yang dihasilkan pada transliterasi aksara *Jawoe* ke Bahasa Latin dengan menggunakan metode CNN dan arsitektur *Xception*.

## 1.4 Batasan Masalah

Batasan-batasan masalah yang diambil oleh peneliti dalam penelitian ini adalah:

1. Data yang digunakan sebagai dataset dalam penelitian ini adalah potongan gambar aksara *Jawoe* yang peneliti tulis berdasarkan buku yang berjudul *Atjehsche Taal* yang ditulis oleh K.F.H. Van Langen (1889).
2. Jumlah dataset yang digunakan yaitu 3000 gambar yang terdiri dari hasil potongan gambar yang peneliti tulis.
3. Metode yang digunakan pada penelitian ini yaitu *Convolutional Neural Network* dengan aristektur *Xception*.

4. Penelitian ini menggunakan bahasa pemrograman yaitu *Python* dengan *framework Tensorflow*, *Keras* dan *Google Collaborary*.

## 1.5 Manfaat Penelitian

Manfaat dari penelitian yang dilakukan ini adalah:

1. Memudahkan pihak lain dalam menerapkan model transliterasi aksara *Jawoe* ke dalam bentuk latin, sehingga bisa digunakan untuk membangun *system* untuk perangkat-perangkat transliterasi dengan menggunakan metode CNN yang sudah dilakukan dan diuji pada penelitian ini.
2. Hasil dari penelitian ini bisa digunakan sebagai acuan untuk penelitian selanjutnya mengenai transliterasi aksara kuno lainnya yang menggunakan metode CNN dan arsitektur *Xception*.

# BAB II
# TINJAUAN PUSTAKA

## 2.1 Penelitian Terdahulu

Banyak referensi dari penelitian terdahulu yang membahas mengenai transliterasi pada aksara kuno, dimana penelitian terdahulu digunakan sebagai acuan untuk menghindari terjadinya *plagiarism*. Berikut ulasan dari beberapa penelitian sebelumnya yang membahas tema yang sama dengan penelitian yang dilakukan.

Penelitian yang dilakukan oleh Pulung Nurtatio Andono & Eko Hari Rachmawanto (2022) yang berjudul "*Deteksi karakter Hiragana menggunakan metode Convolutional Neural Network"*. Penelitian ini memiliki hasil akurasi sebesar 86,5% dengan menggunakan inputan gambar berukuran 83x83 dan menggunakan *epoch* berjumlah 200 dengan perbandingan dataset pengujian yaitu 700:300. Tingkat akurasi ini dipengaruhi dengan banyaknya jumlah dataset *training* yang digunakan. Semakin banyak dataset *training* yang digunakan maka akan menghasilkan nilai akurasi yang tinggi.

Penelitian lain mengenai pengklasifikasian aksara Jawa dengan menggunakan metode *Convolutional Neural Network* yang dilakukan oleh Sukma Hanindria Ivan & Hendry (2022), menghasilkan nilai akurasi sebesar 85% untuk pengklasifikasian aksara jawa dengan menggunakan 50 *epoch* dan menghasilkan tingkat *loss* sebesar 0.43 untuk *training*.

Selanjutnya Penelitian yang dilakukan oleh Silvia Nur Rahmawati, dkk (2021) mengenai Implementasi *deep learning* pada pengenalan aksara sunda yang menggunakan metode CNN menunjukkan hasil akurasi sebesar 92% dimana semakin banyak data *training* yang digunakan maka tingkat hasil akurasi yang didapatkan akan semakin tinggi.

Penelitian yang dilakukan oleh Wulan Anggraini (2020) yang membahas mengenai pendeteksian wajah yang berhijab dengan menggunakan algoritma

Convolutional Neural Network, menghasilkan nilai akurasi sebesar 92% pada data training dan 87% pada data testing dengan menggunakan dataset berjumlah 250 citra utntuk dataset latih dan 50 citra untuk dataset testing.

Adapun penelitian yang dilakukan oleh Muhammad Ridha (2022) dalam mendeteksi fake masker menggunakan metode *Xception* dengan nilai *epoch* 5 dan 10 menghasilkan nilai akurasi 99% dengan dataset terdiri dari tiga jenis *class*, yaitu bermasker, tidak bermasker dan *fake masker*.

Penelitian lain yang dilakukan oleh Aulia Sabri (2022) yang membahas mengenai perbandingan antara model arsitektur CNN dalam mendeteksi coronavirus disease menggunakan citra x-ray paru-paru dengan penggunaan jumlah epoxh 20 dan total dataset yang berjumlah 21,165 dengan memiliki empat kelas yaitu covid, lung opacity, normal, dan pneumonia. Hasil akurasi yang diperoleh mencapai nilai tertinggi yaitu 93,33%.

Tabel 2.1 Perbandingan Penelitian Sejenis

| **Peneliti** | **Metode** | **Kasus** | **Jumlah Dataset** | **Akurasi** |
|---|---|---|---|---|
| Pulung Nurtatio Andono & Eko Hari Rachmawanto (2022) | CNN | Deteksi Karakter Hiragana | 1000 citra gambar tulisan tangan | 86,5% |
| Sukma Hanindria Ivan & Hendry (2022) | CNN | Pengklasifikasian Aksara Jawa | 2580 citra aksara Jawa | 85% |
| Silvia Nur Rahmawati, dkk (2021) | CNN | Implementasi D*eep Learning* pada Pengenalan Aksara Sunda | 847 data aksara Sunda | Menghasilkan 96.71% dengan data training dan 92.02% dengan data testing |

| | | | | |
|---|---|---|---|---|
| Wulan Anggraini (2020) | CNN | Deteksi wajah berhijab | 300 citra | nilai akurasi sebesar 92% pada data training dan 87% pada data testing |
| Muhammad Ridha (2022) | Xception | Pendeteksian *Fake* Masker | 7715 citra dengan 3 kelas | Data training 10 *epoch* menghasilkan nilai akurat 99.53% dan dengan 5 *epoch* menghasilkan 99.03% |
| Aulia Sabri (2022) | Xception | Pendeteksian Coronavirus Disease Menggunakan Citra X-Ray Paru-Paru | 21,165 citra dengan 4 kelas | Menghasilkan nilai akurasi tertinggi mencapai 93,33% dengan epoch 20 |

Berdasarkan penelitian yang telah dijelaskan sebelumnya, diketahui bahwa transliterasi pada aksara kuno ke dalam Bahasa latin dengan menggunakan metode *Convolutional Neural Network* belum banyak dilakukan. Maka dari itu, pada penelitian ini akan melakukan transliterasi terhadap aksara *Jawoe* ke dalam Bahasa latin dengan menggunakan metode *Convolutional Neural Network* dengan menggunakan arsitektur *Xception*.

## 2.2 Landasan Teori

### 2.2.1 Transliterasi

Transliterasi adalah suatu proses pengalihan atau alih aksara dari huruf asli ke dalam bentuk huruf latin (bahasa yang di pahami, misalnya bahasa Indonesia) agar bentuknya dapat dimengerti oleh orang lain (Hudaa et al., 2019). Trasliterasi

ini bertujuan untuk memudahkan para pengguna dalam memahami bahasa yang sulit untuk di mengerti.

### 2.2.2 Bahasa Arab-Melayu

Bahasa Arab Melayu atau aksara Jawi adalah salah bentuk Bahasa yang digunakan oleh bangsa Melayu dengan memadukan seluruh karakter aksara Arab dengan menambahkan aksara lain yang tidak termasuk dalam huruf Arab seperti huruf چ، غ، ڤ، ک، ڽ، ز untuk melengkapi huruf dalam bahasa Arab (Ramala, 2020). Huruf tambahan ini dibuat dengan menambahkan tanda baca tambahan (diakritik) pada huruf untuk mengubah nilai dan membedakan bunyinya. Dalam pembacaannya menggunakan tata bahasa Indonesia. Bahasa Arab-Melayu ini telah berfungsi sebagai bahasa perantara dalam urusan-urusan pemerintahan yang digunakan sejak zaman dahulu. Bahasa Arab-Melayu juga menggunakan tata bahasa Aceh dalam pengucapannya. Ini dikenal dengan istilah *Harah Jawoe* atau aksara *Jawoe.*

*Harah Jawoe* atau aksara *Jawoe* merupakan bahasa yang menggunakan tiga bahasa yaitu bahasa Aceh, bahasa Arab dan Bahasa Melayu. Aksara Jawoe banyak ditemukan pada manuskrip dan hikayat-hikayat yang ditulis para cendikiawan Aceh. Hikayat Aceh merupakan salah satu jenis karya sastra Aceh yang umumnya berbentuk puisi, syair dan prosa yang berisikan berbagai bidang ilmu dan corak meliputi dogeng, nasehat, cerita Sejarah, kisah dan lain sebagainya (Hermansyah, 2020). Hikayat dan manuskrip Aceh ini telah tersimpan diberbagai museum, Lembaga swasta dan koleksi personal.

### 2.2.3 Bahasa Latin

Bahasa latin adalah bahasa yang digunakan untuk membantu memudahkan orang lain dalam membaca suatu kata asing yang sulit untuk dibaca. Bahasa latin ini digunakan sebagai sarana pembelajaran terhadap bahasa asing yang ditulis dengan menggunakan tata bahasa Indonesia yaitu menggunakan huruf abjad dari a sampai z.

### *2.2.4 Artificial Intelliegence*

*Artificial Intelligence* atau lebih dikenal dengan istilah AI merupakan cabang ilmu dan teknik yang membahas mengenai pembuatan mesin cerdas yang mampu melakukan pekerjaan manusia dengan baik (Ridha, 2022a). Adapun beberapa metode yang telah dikembangkan menggunakan *Artificial intelligence,* yaitu :

a. *Fuzzy Logic* (FL)

   *Fuzzy Logic* merupakan teknik logika yang dipakai oleh mesin untuk dapat mengambil keputusan tanpa merasa kaku dan mampu beradaptasi dengan kondisi dan dapat menyesuaikan diri. Salah satu contoh penerapan logika ini adalah penggunaan system rem kereta di Jepang.

b. *Evolutionary Computing* (EC)

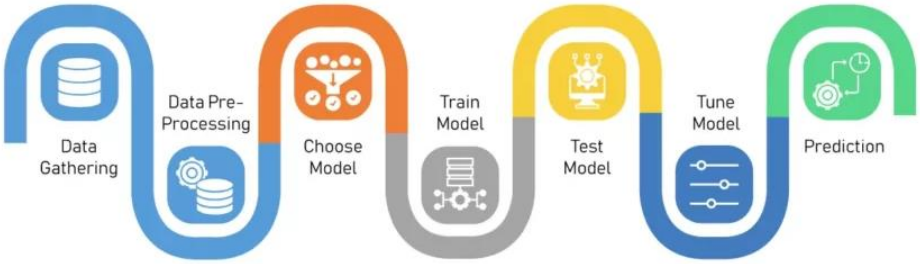
   *Evolutionaru Computing* merupakan skema evolusi yang memungkikan pemilihan individu terbaik yang akan menjadi generasi berikutnya, Contoh penggunaan skema ini ada pada algoritma genetika menggunakan mutasi dan hibridisasi (kawin silang).

c. Machine Learning

   Machine learning adalah pembelajaran yang mampu membuat *computer* memiliki kemampuan belajar tanpa harus diprogram terlebih dahulu (Ellyadi, 2022)

### *2.2.5 Machine Learning*

*Machine Learning* merupakan bagian ilmu dari *Artificial Intelligence* yang membuat *computer* memiliki kemampuan untuk belajar tanpa harus diprogram terlebih dahulu (Wulan Anggraini, 2020). Mesin akan belajar berdasarkan data yang telah diberikan pada data train atau data yang digunakan sebagai bahan belajar diikuti dengan model pembelajaran yang digunakan. Sehingga mesin akan memahami data uji yang input untuk melihat dan mengetahui hasil dari apa yang dipelajari oleh mesin. Hasil dari *Machine learning* dapat berupa saran, prediksi ataupun keputusan berdasarkan permintaan dan penggunanya.




Gambar 2.1 Proses *Machine Learning* (Sasanadigital, 2022)

*Machine learning* terdiri dari tujuh tahapan proses dalam mesin mempelajari suatu data sampai mesin dapat memberikan hasil yang di inginkan. Tahapan-tahapan itu yaitu:

a. *Data Ghatering*

   Pada tahapan *data ghatering*, mesin akan mengumpulkan data-data yang dibutuhkan dalam membentuk sebuah model.

b. *Data Pre-processing*

   Pada tahapan ini, mesin akan melakukan filterisasi, seleksi, *resize* ukuran data yang tidak sama dan membersihkan data-data yang sekiranya tidak sesuai seperti data cacat, blur atau data yang kembar. Pada tahap ini juga, akan dilakukan pemberian label agar mesin dapat belajar mengenai objek dan tahapan ini juga akan membagi data menjadi dua bagian, yaitu data *training* (data belajar) dan data *testing* (data uji). Umumnya dalam pembagian ini data dibagi dengan menggunakan persentase yaitu 80% digunakan sebagai data *training* dan 20% digunakan sebagai data *testing*.

c. *Chose Model*

   Pada tahapan *chose model*, memilih model atau algoritma yang akan digunakan sesuai dengan hasil *output* yang di inginkan termasuk bagaimana cara mesin itu belajar.

d. *Train Model*

   Pada tahapan *train model*, mesin akan melakukan proses pembelajaran dan akan menetukan pola dari algoritma yang digunakan.

e. *Test Model*

Pada tahapan *test model*, mesin akan melakukan pengujian terhadap train yang sudah dilakukan untuk menentukan bagaimana performa, *sensitivity* dan hasil yang didapat.

f. *Tune Model*

Pada tahapan *tune model*, mesin akan melihat bagaimana performa yang dihasilkan. Jika belum mencapai 90%, maka mesin akan melakukan proses *training* lagi untuk meningkatkan performa.

g. *Prediction*

Pada tahapan *prediction*, mesin akan melakukan prediksi terhadap data uji berdasarkan model pembelajaran yang digunakan.

Machine learning memiliki tiga jenis kategori utama dalam pembelajaran, yaitu:

- *Supervised Learning*

  Pada kategori ini, mesin akan belajar berdasarkan contoh data. Contoh data dapat berupa data yang memiliki label atau kelas yang akan menunjukkan jenis kelompok dari data itu berada. Sehingga mesin akan belajar dengan membandingkan antara data inputan (*data train*) dengan data uji untuk menghasilkan *output* yang sesuai dengan apa yang telah dipelajarainya.
- *Unsupervised Learning*

  Pada kategori ini, mesin belajar berdasarkan data yang tidak memiliki informasi atau tidak berlabel. Sehingga mesin harus menemukan pola dan struktur yang dimiliki didalam data tersebut tanpa memiliki data acuan untuk membantu prediksi terhadap sesuatu.
- *Reinforcement Learning*

  Pada ketegori ini, mesin akan belajar melalui interaksi dengan lingkungannya melalui *error* dan *trail*. Cara *reinforcement learning* berkerja sama seperti cara manusia belajar, dimana seseorang belajar berdasarkan pengalaman yang di alami dan tindakan yang dilakukan untuk menghindari

melakukan kesalahan yang sama. Dalam pembelajaran ini, mesin akan menentukan aksi yang digunakan tanpa adanya informasi dan dari Tindakan yang dilakukan akan mendapatkan reward.

### *2.2.6 Deep Learning*

*Deep Learning* adalah salah satu cabang dari ilmu *Machine Learning* yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data dengan menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam (Lorentius et al., 2019). *Deep learning* menggunakan metadata yang digunakan sebagai inputan dan untuk mengolahnya digunakan sejumlah lapisan yang tersembunyi dari transformasi non-linear yang dimiliki inputan agar dapat menghitung hasil nilai dari *output*.

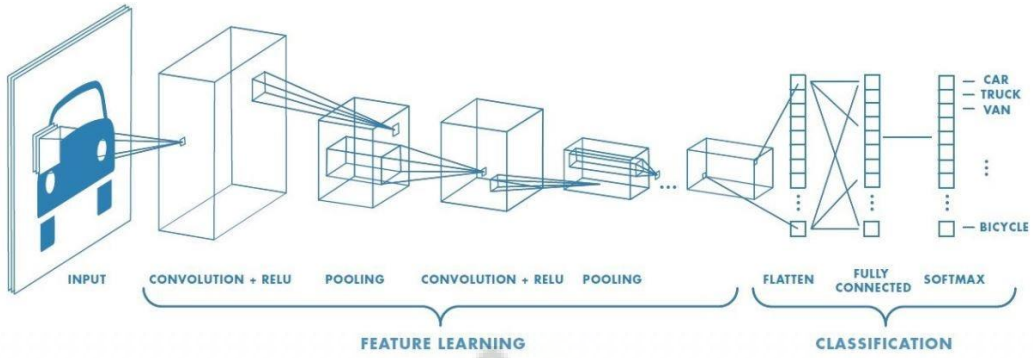
Pada *deep learning*, mesin akan mempelajari berbagai macam model dan melakukan klasifikasi terhadap berbagai macam bentuk data untuk menghasilkan informasi dan hasil prediksi yang memiliki nilai akurasi yang tinggi. *Deep Learning* juga mampu melakukan ekstraksi secara otomatis.

### *2.2.7 Object Detection*

*Object detection* adalah suatu teknik pada *computer* yang digunakan untuk menemukan contoh objek dalam gambar atau video (Nufus et al., 2021). Pada *object detection*, umumnya memanfaatkan *machine learning* dan *deep learning* dalam mencapai hasil yang memiliki *value*. Mesin akan melakukan *scanning* atau pemindaian terhadap gambar untuk menentukan objek mana yang ada di dalam gambar dan mana yang bukan. Dengan *object detection*, mesin juga dapat mengetahui posisi dari objek-objek yang ada pada gambar yang di inputkan.

### *2.2.8 Convolutional Neural Network*

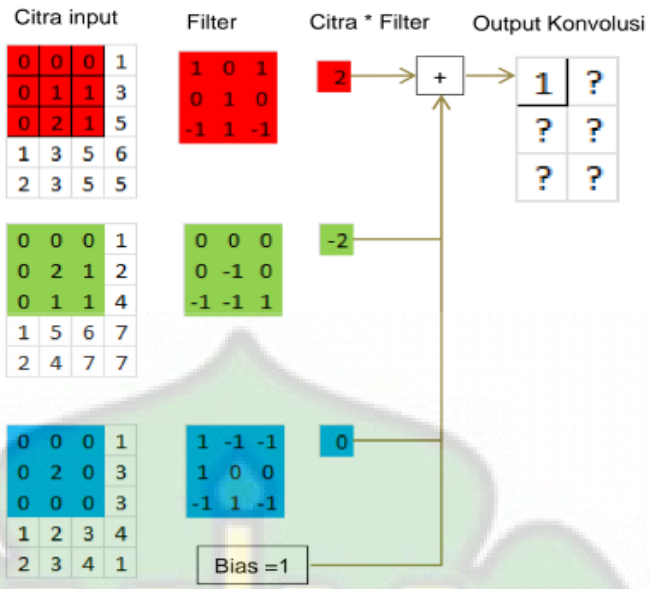
*Convolutional Neural Network* (CNN) adalah salah satu metode pembelajaran dari *deep learning* yang digunakan untuk mendeteksi dan mengenali data dari pemrosesan gambar (Lina, 2019). CNN terdiri dari lapisan-lapisan *neuron* yang disusun untuk membentuk dengan *filter* atau piksel yang panjang dan tinggi. CNN menggunakan proses *convolution* untuk mengekstrak dan mendapatkan informasi dari gambar. Informasi dari gambar akan dihubungkan ke dalam *neuron*.




Gambar 2.2 Arsitektur CNN (Nugroho. F. A, 2021)

Sebuah arsitektur CNN terdiri dari beberapa lapisan yang memiliki tugas dan fungsinya masing-masing. Secara umum, arsitektur CNN terdiri dari dua bagian, yaitu *Feature Learning* dan *Classification*. *Feature learning* merupakan proses *encoding*, dimana citra gambar yang akan dikodekan menjadi fitur-fitur yang terdiri dari angka-angka yang mendefinisikan citra tersebut. *Feature* Learning terdiri dari dua bagian yaitu *Convulition layer* dan *Pooling layer*.

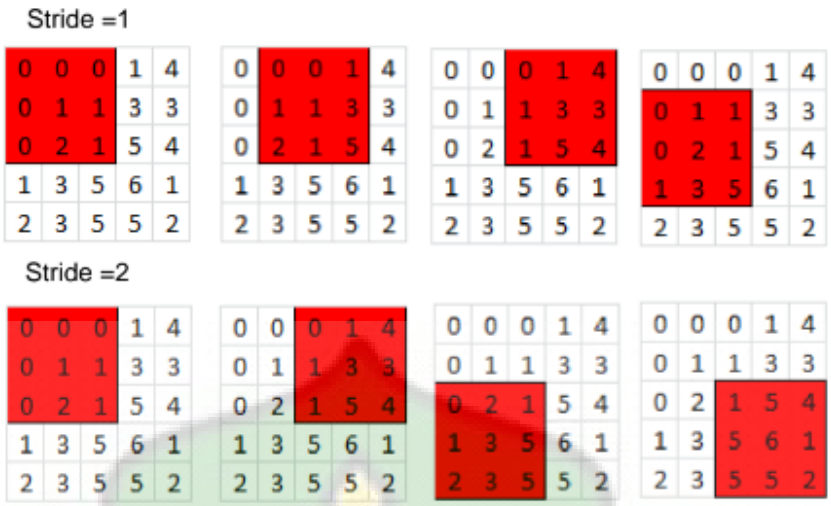
*Convolution Layer* adalah lapisan pertama yang menerima inputan citra gambar dan memperoleh informasi mengenai posisi dan nilai hasil *convolution* pada lapisan. *Convolution* merupakan teknik perkalian citra dengan menggunakan *filter*. Fungsi *filter* adalah untuk mengekstraks fitur yang digunakan untuk mengetahui keberadaan fitur tersebut. Pada dasarnya, *filter* adalah *array* yang terdiri dari dua dimensi yang memiliki ukuran yang lebih kecil dari gambar inputan. Proses *Convolution* dilakukan dengan cara menjumlahkan hasil perkalian antara citra gambar dengan *filter* pada setiap pergeseran yang dilakukan pada bagian gambar, Perkalian ini akan menghasilkan sebuah hasil *output* yang disebut dengan *feature map* atau *activation map*. Misalnya, pada lapisan pertama dari lapisan ekstraksi ciri yaitu layer *convolution* dengan ukuran 3x3x3. Dimana panjangnya 3 px (piksel), tingginya 3 px dan jumlah/tebal 3 buah.




Gambar 2.3 Proses konvolusi (Rahman. S, dkk, 2021)

Pada gambar diatas, citra inputan yang memiliki tiga kanal dengan ukuran 3x3 piksel difilter dengan menggunakan *output* satu kanal. Dimana hasil perkalian antara citra *input* pada kanal pertama menghasilkan nilai 2, pada kanal kedua menghasilkan nilai -2 dan pada kanal ketiga menghasilkan nilai 0.

*Stride* adalah bagian dari lapisan *Convolution* yang merupakan parameter yang digunakan untuk menentukan banyaknya jumlah pergeseran yang dilakukan oleh *filter*. Jika nilai yang miliki oleh *stride* adalah 1 maka, *filter* akan bergeser secara *horizontal* dan secara *vertical* sebanyak satu kali untuk setiap piksel. Semakin besar nilai yang dimiliki dari *stride* maka informasi yang didapatkan juga semakin kecil dari sebuah citra inputan.




Gambar 2.4 Ilustrasi proses *stride* (Rahman. S, dkk, 2021)

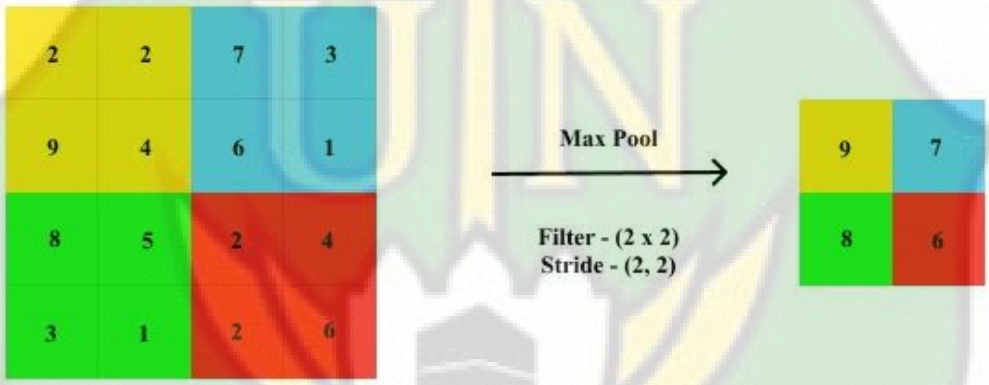
*Padding* atau dikenal dengan *zero padding* adalah bagian dari lapisan *Convolution* yang merupakan parameter yang digunakan untuk menentukan jumlah piksel yang akan ditambahkan pada setiap sisi pada inputan gambar. Biasanya, *padding* diterapkan dengan menambahkan nilai 0, yang digunakan untuk dapat mengubah dimensi *output* pada *convolution layer.*


Gambar 2.5 Proses penambahan *padding* (Rahman. S, dkk, 2021)

ReLU (*Rectified Linear Unit*) berfungsi sebagai proses atau langkah-langkah untuk menghilangkan *vanishing gradient* menggunakan aktivasi elemen f(x) = max (0,X) (Pulung, Eko, 2022). Dimana hasil *output* dari proses *convolution*

yang memiliki nilai negatif akan diubah atau diaktivasi kedalam nilai positif berupa nilai 0.

Lapisan kedua adalah *Pooling Layer*. Lapisan *Pooling* berada setelah lapisan pertama yaitu lapisan *convolution*. *Pooling* terdiri dari sebuah filter yang memiliki ukuran dan nilai *stride* tertentu yang melakukan permindahan dengan mengeser pada seluruh area *feature map* yang bertujuan untuk mengurangi dimensi dari *feature map* sehingga dapat mempercepat komputasi dengan menambahkan lebih sedikit parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting*. *Pooling* yang paling umum digunakan adalah *Max Pooling* dan *Average Pooling.*




Gambar 2.6 Ilustrasi proses *pooling layer* (Kartikeya, 2020)

Pada Gambar 2.6 diatas, dapat dijabarkan bahwa pada proses *layer pooling* menggunakan *Max pooling* pada hasil *feature map* yaitu dengan melihat nilai maksimum yang dimiliki pada setiap bloknya. Pada blok pertama yang berwarna kuning, nilai maksimum yang dimiliki adalah 9. Pada blok kedua yang berwarna biru, nilai maksimum yang dimilikinya adalah 7. Pada blok ketiga yang berwarna hijau, memiliki nilai maksimum adalah 8. Dan pada blok terakhir yang berwarna merah, memiliki nilai maksimum yaitu 6. Sehingga diperoleh *feature map* yang memiliki array berdimensi 2 x 2.

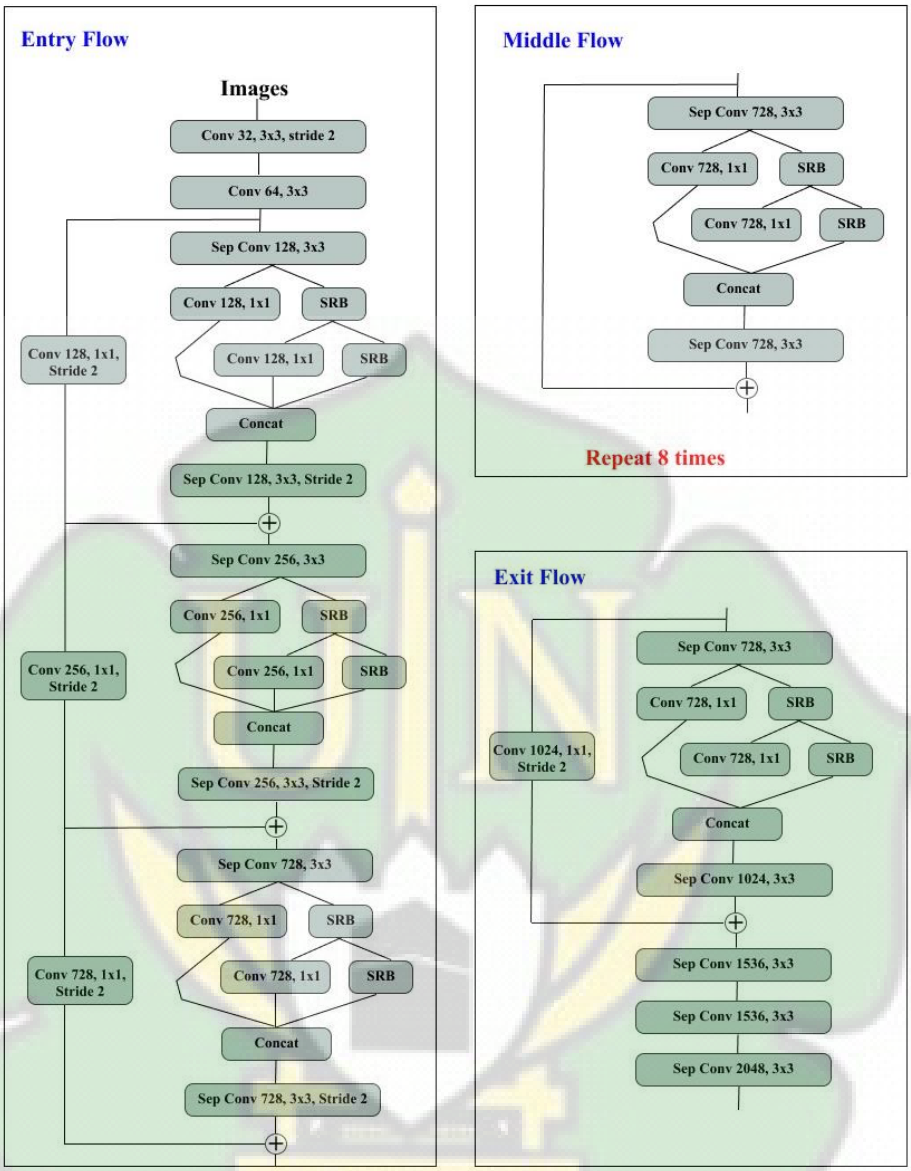
Pada bagian kedua yaitu bagian *Classification*. Pada *classification* berfungsi untuk membuat klasifikasi pada setiap *neuron* yang telah melewati proses ekstrasi pada tahap sebelumnya. Pada bagian *Fully Connected Layer*, *output feature*

*map* yang dihasilkan pada bagian pertama yaitu bagian ekstraksi fitur masih berupa *array* multidimensi, sehingga perlu untuk dilakukan "*flutten*" atau *reshape feature map* menjadi sebuah vektor yang dapat digunakan sebagai inputan dari lapisan *fully connected*. Hal ini dilakukan supaya nilai tersebut dapat digunakan sebagai inputan pada layer *fully connected layer*. Pada lapisan ini, semua aktivitas *neuron* dari lapisan sebelumnya saling terhubung dengan *neuron* pada lapisan berikutnya. Dengan demikian, setiap aktivitas pada lapisan sebelumnya harus di ubah menjadi data dari suatu dimensi yang terhubung pada semua *neuron* di lapisan *fully connected*. *Layer fully connected* bertujuan untuk melakukan transformasi di dimensi data agar data yang diperolah pada *feature maps* dapat dilakukan pengklasifikasian secara linear.

Menurut (Rachmawanto & Andono, 2022) Softmax memiliki fungsi sebagai metode klasifikasi yang menggunakan beberapa jumlah kelas yang lebih dari satu. *Softmax* merupakan sebuah fungsi dan dapat digunakan untuk mengubah dimensi vektor berupa sebuah nilai yang berbentuk real kedalam nilai yang berbentuk vektor. Langkah ini digunakan untuk mencari nilai dari probabilitas terhadap target yang akan digunakan sebagai kelas target inputan.

### *2.2.9 Xception*

Arsitektur *Xception* merupakan arsitektur yang penamaannya berasal dari arsitektur sebelumnya yang Bernama *Inception* yang mempunyai lapisan *convolution* berjumlah 36 yang merupakan dasar jaringan ekstraksi fitur. *Xception* dibentuk dengan menggunakan *separable Convolutions*. *Xception* merupakan singkatan dari *Extreme of Inception*. *Xception* merupakan arsitektur yang menggunakan metode *Convolution* yang dapat dipisahkan secara mendalam (R. Kurniawan et al., 2023).




Gambar 2.7 Arsitektur *Xception* (Ridha, 2022)

Pada awalnya dataset akan melewati *entry flow*, lalu melalui *middle flow* dengan melakukan perulangan sebanyak 8 kali pada tahapan ini, kemudian akan berakhir melalui *exit flow*. Semua lapisan *Convolution* dan *Separable Convolution* diikuti dengan *normalization batch* yang digunakan untuk mempercepat proses *training*. Di dalam *Separable Convolution* dibagi menjadi 2 langkah utama yaitu *Depthwise Convolution* dan *Pointwise Convolution*, Langkah pertama yaitu *Pointwise Convolution* memproses data dengan matrix 1x1 dan selanjutnya akan masuk ke Langkah *Depthwise Convolution* yang memproses data dengan matriks 3x3 (Ridha, 2022).

## 2.3 Tools

*Tools* merupakan alat bantu yang akan digunakan dalam penelitian ini. Tools yang akan digunakan meliputi bahasa *python* yang merupakan bahasa pemrograman, framework *keras, tensorflow, jupyter notebook* dan *google collaboratory*. *Tools* ini akan digunakan oleh penulis dalam melakukan proses transliterasi aksara *Jawoe* ke latin menggunakan metode *Convolutional Neural Network* dengan arsitektur *Mobilenet* dan *Xception*. *Tools* yang digunakan ini dapat diperoleh dengan cara melakukan instalasi terhadap *tools* yang dibutuhkan.

### *2.3.1 Python*

Python merupakan bahasa pemrograman yang *interpretative* multiguna yang lebih menekankan pada keterbacaan kode menjadi lebih mudah untuk dapat dimengerti. *Python* memiliki *library* yang lengkap untuk memudahkan programmer dalam membuat aplikasi dengan penggunakan *Source code* yang sederhana.

### *2.3.2 Google Collaboratory*

*Google Collaboratory* merupakan alat yang dikembangkan oleh Google Internal Research yang digunakan dalam membantu para peneliti mengolah data untuk kebutuhan pembelajaran pada mesin dalam melakukan pengolahan data. *Google Collaboratory* atau Colab dibangun diatas lingkungan Jupyter yang memiliki tampilan yang mirip dengan tampilan *Notebook* yang dimiliki oleh Jupyter yaitu *Jupyter Notebook*. Colab ini digunakan secara *online* dengan menggunakan sistem *cloud* dan google drive sebagai media penyimpanan. Selain itu Colab juga menyediakan akses secara gratis untuk menggunakan layanan GPU sebagai *backend* proses komputasi (Nur et al., 2023).

### *2.3.3 Tensorflow*

Tensorflow adalah sebuah *library open source* yang dirancang oleh *Google Brain Team* dengan menggunakan bahasa C++ yang berupa *library* matematika simbolik yang digunakan untuk *Machine learning* dan *Antificial intelligence*. *Tensorflow* dirancang dengan memiliki performa tinggi dan dirancang untuk dapat bekerja dengan menggunakan banyak CPU dan GPU. *Tensorflow* rilis pada 11 February 2017 yang dapat dijalankan di Windows, macOS, Linux dan pada

perangkat seluler seperti Android dan IOS. Fitur utama yang terdapat dalam *Tensorflow*:

1. Mengoptimalkan, menghitung dan mendefinisikan secara matematis ekspresi wajah dengan melibatkan *array multidimension* (tensor).
2. Pemrograman pendukung jaringan syaraf dalam dan teknik *machine learning.*
3. Pemakaian GPU (*Graphic Processing Unit*) yang efisien, mengotomasi manajemen dan optiamalisasi memori yang sama terhadap data yang digunakan. *Tensorflow* mampu untuk menulis kode yang sama dan menjalankannya di CPU maupun GPU. Lebih khususnya lagi, *Tensorflow* dapat mengetahui bagian mana yang harus dipindahkan ke GPU.
4. Skalabilitas komputasi yang tinggi pada keseluruhan mesin terhadap kumpulan data yang besar. (Wulan Anggraini, 2020).

### *2.3.4 Keras*

*Keras* termasuk salah satu *framework deep learning* tingkat tinggi yang dikembangkan oleh *Google* yang digunakan untuk menerapkan jaringan saraf dengan memiliki fitur-fitur yang membantu mempermudah pengembangan model *deep learning*. Keras menggunakan bahasa pemrograman *Python* sebagai antar muka API dan dibangun diatas *Tensorflow*. Pengaruh Teknologi *deep learning* yang terus berkembang, *Keras* menyediakan model yang telah terlatih atau *pretrained model* yang dapat digunakan dalam membantu menyelesaikan berbagai masalah seperti *image classification* dan permasalah lainnya. Pada *Keras*, model yang sudah terlatih ini disebut dengan *keras application* (Sabri, 2022).

## 2.4 Metode Evaluasi

Pada penelitian ini, peneliti menggunakan metode *Evaluation Measurement* atau pengukuran evaluasi. Metode evaluasi yang digunakan yaitu *recall* dan *precision* dan *accuracy*.

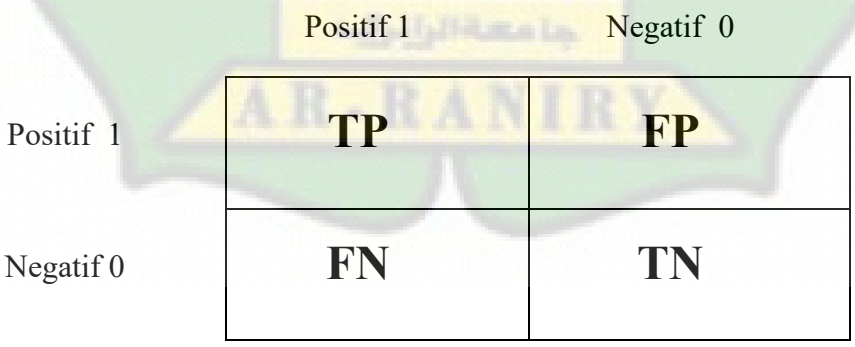
### *2.4.1 Recall dan Precision*

Melakukan pengukuran evaluasi dengan menggunakan nilai *recall* (R) dan *precision* (P) bertujuan untuk mengetahui tingkat kesesuaian prediksi yang

dilakukan dengan melihat ketepatan model dalam melakukan pencarian dan pengenalan informasi yang diminta oleh pengguna model. Prediksi ini dilakukan terhadap kelompok yang bernilai *positive*.

Nilai *recall* adalah nilai yang digunakan untuk menunjukkan tingkat perolehan hasil yang dikembalikan dari sebuah *system* yang diperoleh dengan membandingkan jumlah item relevan yang dikembalikan dari *system* dengan menggunakan total jumlah item yang ada. Pada nilai *recall*, semakin besar nilai yang dihasilkan maka nilai tersebut tidak dapat menunjukkan bahwa suatu *system* itu baik atau tidak. Secara umum, nilai tertinggi dari nilai *recall* adalah 1. Nilai *precision* bertujuan untuk menunjukkan tingkat ketepatan sebuah *system* untuk mengembalikan informasi relevan kepada pengguna.

Dalam evaluasi pengukuran model algoritma, digunakan acuan pada *confusion matrix* yang memberikan informasi berupa perbandingan dari hasil klasifikasi yang dilakukan oleh model dengan hasil klasifikasi sebenarnya (Kurniawan et al., 2022). Dimana dalam dengan menggunakan *confusion matrix* dapat mentukan hasil *accuracy, recall, f1 score* dan *precision. Confusion matrix* terdiri dari dua buah kolom yaitu kolom, yaitu kolom aktual (kondisi sebenarnya) yang bernilai *positive* (1) dan kolom aktual yang bernilai *negative* (0) dan terdiri dari dua baris, yaitu baris prediksi yang bernilai *positive* (1) dan baris prediksi yang bernilai *negative* (0).




Gambar 2.8 *Confusion matrix*

Pada Gambar 2.8, TP atau *True Positive* mengambarkan jumlah data yang di prediksi bernilai benar atau *positive* yang berada pada kelompok yang bernilai *positive*. FP atau *False Positive* menggambarkan jumlah data yang di prediksi

bernilai salah atau *negative* yang berada pada kelompok yang bernilai *positive*. FN atau *False Negative* mengambarkan jumlah data yang di prediksi bernilai salah yang berada pada kelompok yang bernilai *negative*. TN atau *True Negative* mengambarkan jumlah data yang di prediksi bernilai benar yang berada pada kelompok *negative*.

Berdasarkan pada acuan *confusion matrix* diatas, untuk menentukan nilai *recall* (R) dan *precision* (P) pada kelompok *positive* dapat menggunakan formula sebagai berikut:

$$R = \frac{TP}{TP + FN} \tag{2.1}$$

$$P = \frac{TP}{TP + FP} \tag{2.2}$$

Dari formula diatas, dapat disimpulkan bahwa nilai *recall* (R) adalah nilai yang diperoleh dari jumlah data prediksi yang bernilai benar pada kelompok *positive* (TP) dibagi dengan jumlah nilai yang diperoleh dari data prediksi TP ditambah FN atau jumlah data yang berada pada kelompok *positive* yang sebenarnya. Sedangkan nilai *precision* (P) adalah nilai yang diperoleh dari jumlah data prediksi yang bernilai benar pada kelompok *positive* (TP) dibagi dengan jumlah nilai yang diperoleh dari data TP ditambah FP atau jumlah data sebenarnya yang berada pada kelompok *positive*.

### *2.4.2 F1 Score*

*F1 Score* merupakan *Harmonic Mean* atau perbandingan rata-rata antara p*recision* dengan *recall.* Nilai F1 score berada antara nilai 0 dan 1. Nilai 1 adalah nilai terbaik, dan nilai 0 adalah nilai terburuk. Nilai *F1 score* yang baik menjelaskan bahwa model memiliki nilai *presicion* dan *recall* yang baik. Persamaan pengukuran nilai *F1 Score* adalah sebagai berikut :

$$F1\ Score = 2\ .\frac{1}{\frac{1}{recall} + \frac{1}{precision}} \quad (2.3)$$

### *2.4.3 Accuracy*

*Accuracy* adalah pengukuran untuk melihat seberapa sering algoritma klasifikasi berhasil dalam membuat prediksi yang memiliki nilai benar. Dimana untuk menghasilkan nilai *accuracy* dengan melihat rasio jumlah total prediksi yang bernilai benar dibagi dengan keseluruhan total dari prediksi (Kurniawan et al., 2022). Persamaan pengukuran nilai akurasi adalah sebagai berikut:

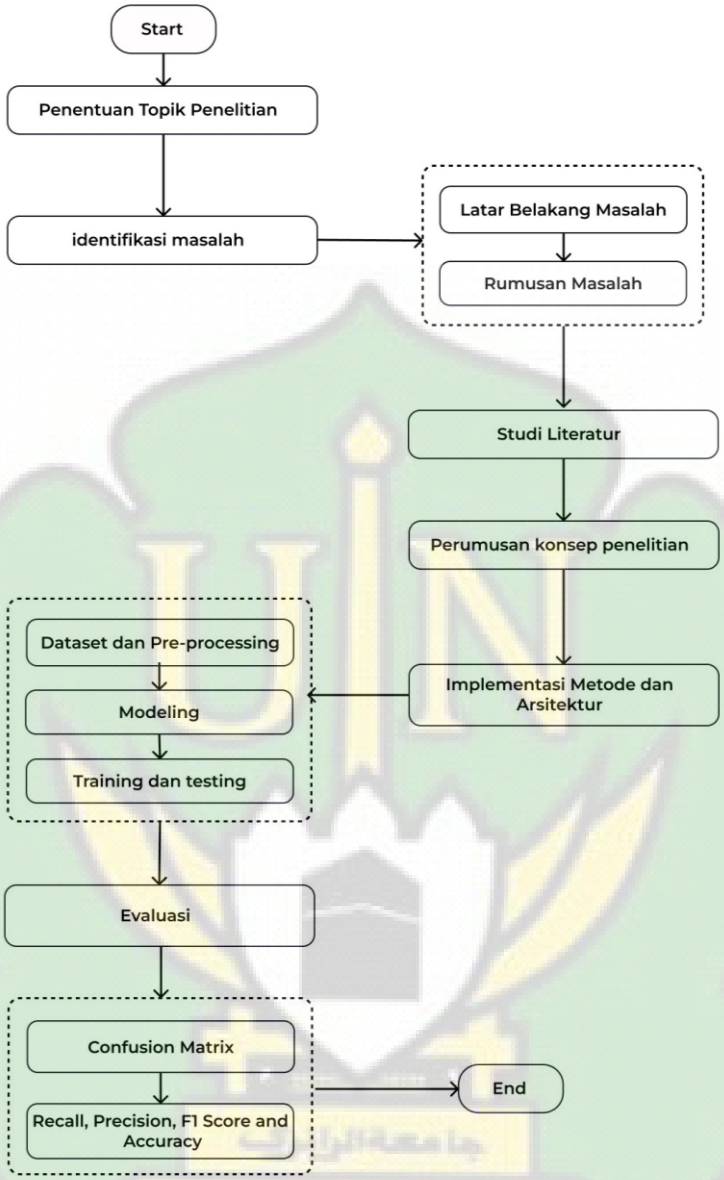
$$\text{Acurracy } = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \quad (2.4)$$

Keterangan:

| | |
|---|---|
| *Accuracy* | : Total akurat model dalam mengenali dataset |
| TP | : Jumlah data benar diprediksi oleh model |
| TN | : jumlah data *negative* yang benar diprediksi oleh model |
| FP | : jumlah data *negative* yang diprediksi sebagai data *positive* |
| FN | : jumlah data *positive* yang diprediksi sebagai data *negative* |

## 2.5 Kerangka Berpikir Penelitian

Kerangka berpikir adalah kerangka yang digunakan untuk mengidentifikasikan alur logika yang digunakan oleh peneliti dalam melakukan dan mengembangkan penelitian dengan berisikan langkah-langkah yang terdiri dari penentuan topik penelitain, pengumpulan data dengan melakukan identifikasi masalah, simulasi dan implementasi dan analisis evaluasi. Peneliti menggunakan kerangka berpikir penelitian yang dapat dilihat pada gambar dibawah ini.




Gambar 2.9 Kerangka berpikir penelitian

# BAB III
# METODEOLOGI PENELITIAN

## 3.1 Jenis Penelitian

Pada penelitian ini menggunakan jenis penelitian kuantitatif yaitu penelitian yang menganalisis data angka untuk menemukan jawaban atas pertanyaan penelitian yang berkaitan dengan fenomena yang akan diteliti. Penelitian kuantitatif akan memiliki jawaban yang pasti. Dalam penelitian ini, masalah transliterasi aksara *Jawoe* ke latin menggunakan metode *Xception* adalah alasan mengapa jenis penelitian ini dipilih.

## 3.2 Tahapan Penelitian

Langkah-langkah yang akan di lakukan dalam menyelesaikan penelitian ini dapat dilihat pada gambar tahapan-tahapan penelitian berikut ini:


Gambar 3.1 Tahapan Penelitian

## 3.3 Metode Pengumpulan Data

Peneliti menggunakan metode pengumpulan data yang terdiri dari identifikasi masalah dan studi literatur dalam penelitian ini. Untuk mengidentifikasi masalah, peneliti melakukan observasi. Sementara untuk studi literatur, peneliti melibatkan perbandingan pada penelitian-penelitian sebelumnya yang dapat diadopsi dalam penelitian ini.

### 3.3.1 Identifikasi Masalah

Tahapan ini dilakukan untuk menemukan topik penelitian yang sesuai dengan bidang ilmu yang dipelajari. Peneliti mengambil topik penelitian yang berkaitan dengan penggunaan *deep learning* dengan metode *Convolutional Neural Network.* Peneliti menemukan masalah mengenai kesulitan dalam memahami aksara *Jawoe* dikarenakan kurangnya penelitian yang membahas mengenai isi dari manuskrip kuno diikuti dengan menggunakan tiga bahasa berbeda didalamnya. Dalam menyelesaikan permasalah ini, peneliti menggunakan metode CNN dengan arsitektur *Xception* diikuti dengan penggunaan *framework Tensorflow* untuk proses transliterasi.

### 3.3.2 Studi Literatur

Pada tahapan ini, peneliti menggunakan metode studi literatur untuk mengidentifikasi berbagai tindakan yang akan dilakukan terkait dengan pengumpulan data dan memberikan gambaran mengenai masalah yang ada berkaitan dengan transliterasi aksara *Jawoe* yang bersumber dari jurnal/artikel, buku dan bahan pendukung lainnya. Sebelum memulai penelitian ini, peneliti melakukan kajian melalui penelitian-penelitian sebelumnya yang berkenaan dengan topik penelitian yang dipilih. Referensi yang diperoleh dari tahapan ini mencakup penelitian yang berhubungan dengan penggunaan metode CNN, arsitektur maupun *Xception*, dan *framework Tensorflow* untuk proses transliterasi.

## 3.4 Metode Simulasi

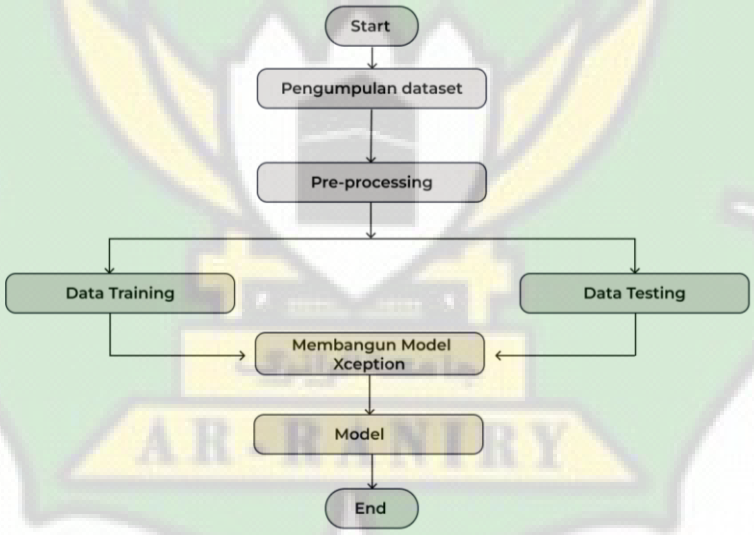
Metode yang diterapkan pada model akan digambarkan pada tahapan ini. Metode simulasi terdiri dari beberapa tahapan yaitu perumusan konsep penelitian dan implementasi metode dan arsitektur

### 3.4.1 Perumusan Konsep Penelitian

Peneliti merumuskan konsep penelitian berdasarkan hasil identifikasi masalah dan studi literatur yang telah dilakukan. Penelitian ini menggunakan metode CNN dengan arsitektur *Xception* untuk melakukan transliterasi aksara *Jawoe* ke latin. Penelitian ini memiliki proses pengujian model yang terdiri dari tahapan *training* terhadap dataset training, membangun model untuk mampu melakukan transliterasi aksara *Jawoe* ke latin dan mendapatkan hasil akurasi yang sangat akurat.

### 3.4.2 Implementasi Metode dan Arsitektur

Tahapan ini merupakan tahapan yang digunakan untuk mentransformasikan konsep penelitian untuk di implementasikan pada model yang dibangun. Implementasi terdiri dari alur yang dikembangkan pada penelitian. Peneliti mengambarkan alur proses transliterasi aksara *Jawoe* ke latin sebagai berikut :




Gambar 3.2 Alur implementasi metode

Adapun rincian mengenai alur implementasi metode dan arsitektur adalah sebagai berikut:

**1. Pengumpulan Dataset**

Dataset yang digunakan dalam penelitian ini diambil dari buku yang ditulis oleh K.F.H. Van Langen (1889) yang berjudul *Atjehscle Taal*. Buku ini berisikan

berbagai macam hikayat yang ditulis menggunakan aksara Jawi dan dalam pembacaannya menggunakan Bahasa Aceh yang masih menggunakan ejaan lama. Sehingga dilakukan penyesuaian pengucapan aksara dengan melihat dari kamus yang telah menggunakan ejaan yang lebih baik.

Dataset yang digunakan berupa gambar diambil dan kemudian dilakukan penulisan ulang dengan menggunakan bantuan keyboard online yang dapat diakses pada situs https://www.lexilogos.com/keyboard/jawi.html. Dataset akan disimpan sesuai dengan nama latin dari aksara *Jawoe*. Dataset yang digunakan berjumlah 3000 kata yang terdiri dari 475 kata yang masing-masing kata terdiri dari 6 dataset. Selama pengumpulan dataset ini, ditemukan beberapa kata yang memiliki dataset berjumlah 12 atau 18 gambar. Hal ini dikarenakan terdapat beberapa kata yang memiliki bentuk penulisan yang berbeda tetapi ejaan yang sama, sehingga menyebabkan terjadinya variasi yang lebih banyak dalam dataset. Untuk dataset yang memiliki variasi dataset dapat dilihat pada tabel 3.1

Tabel 3.1. Variasi dataset

| **Nama Kata** | **Jumlah Data** |
|---|---|
| Adat, Bak, Bijak, Di Pat, Gadöh, Gaséh¸ Hana, Hoka, Jak, Jigaséh, Jijak, Jikeumeung, Jimarit, Jipajôh, Jipeugah, Piker, Sibak, Singoh, Tajak, Tamöng, Tapinah, Ubak | 12 |
| Nibak | 18 |
| Kata lainnya | 6 |

### *2. Pre-Processing*

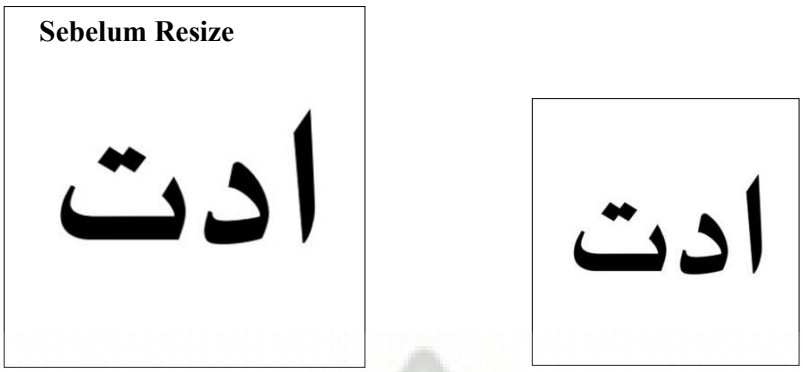
Setelah menyelesaikan tahapan pengumpulan dataset, maka tahapan selanjutnya yang akan dilakukan yaitu melakukan *preprocessing*. Proses *preprocessing* ini dilakukan untuk memudahkan proses traning dan proses pendeteksian terhadap dataset. Beberapa proses yang terjadi dalam tahapan *preprocessing* adalah sebagai berikut :

#### a) Data Augmentasi

Augmentasi data merupakan salah satu teknik yang digunakan untuk mengurangi *overfitting* dengan meningkatkan ukuran yang dimiliki oleh dataset dalam upaya yang seminimal mungkin (Fadillah et al., 2021). Data augmentasi juga dapat dilakukan untuk melakukan manipulasi terhadap data gambar tanpa harus menghilangkan inti yang dimiliki oleh data gambar tersebut. Data augmentasi dapat dilakukan untuk melakukan *rotate, flip, crop, resize,* dan lain sebagainya.

Pada penelitian ini menggunakan teknik transformasi data augmentasi yaitu dengan melakukan *resize* untuk menyamakan ukuran. Dataset awalnya berukuran 1080 x 1080 piksel, sehingga dibutuhkan proses perubahan ukuran dataset menjadi 299 x 299 piksel. Penentuan ukuran ini dilakukan melalui beberapa pengujian dengan menggunakan beberapa ukuran dataset yaitu 299 x 299 piksel dan 520 x 520 piksel.

Dari proses ini didapatkan bahwa pada metode *Xception* hanya bisa menerima ukuran dataset yaitu 299 x 299 piksel, hal ini dikarenakan secara default Xception menerima gambar dengan ukuran 299 x 299 piksel (Ridha, 2022a). Sehingga dipilih ukuran akhir dataset menggunakan ukuran 299 x 299 piksel. Proses perubahan ukuran menjadi 299 x 299 piksel ini dilakukan dengan menggunakan bantuan salah satu library yang dimiliki oleh python yaitu *Python Image Library*. Contoh gambar yang telah dilakukan proses resize untuk mendapatkan ukuran 299 x 299 piksel dapat dilihat pada gambar 3.3.




Gambar 3.3 Contoh *resize* gambar

```
from PIL import Image
import os

drive.mount('/content/gdrive')
#add the path general where the classes subpath are allocated

# Direktori sumber gambar
source_directory = 'gdrive/MyDrive/Sistem/Data'

# Direktori tujuan untuk gambar yang sudah diubah ukuran
output_directory = 'gdrive/MyDrive/Sistem/DataUbah'

if not os.path.exists(output_directory):
    os.makedirs(output_directory, exist_ok=True)


# Ukuran yang diinginkan (misalnya, 299x299 piksel)
desired_size = (299, 299)

# Loop melalui semua gambar dalam direktori sumber
for filename in os.listdir(source_directory):
    if filename.endswith(".jpg"):  # Anda bisa sesuaikan ekstensi gambar
        # Buka gambar
        img = Image.open(os.path.join(source_directory, filename))

        # Ubah ukuran gambar
        resized_img = img.resize(desired_size)

        # Simpan gambar yang sudah diubah ukurannya ke direktori tujuan
        resized_img.save(os.path.join(output_directory, filename))

        # Tutup gambar asli
        img.close()
```

Gambar 3.4 *Source code resize* gambar dengan augmentasi data

Pada *Source code* diatas dijelaskan bahwa tahap awal dimulai dengan melakukan *mount* terhadap *google drive* yang merupakan media penyimpanan dataset yang digunakan akan dapat dipanggil. Selanjutnya akan dilakukan pendeklarasian atau penentuan direktori yang akan digunakan sebagai data sumber atau data awal dan direktori yang akan digunakan sebagai tujuan gambar yang telah

diubah ukuran. Kemudian akan digunakan fungsi untuk membuat direktori tujuan apabila tidak ditemukan direktori tersebut pada *google drive*. Selanjutnya akan dilakukan proses augmentasi data dengan menentukan ukuran dataset yang akan diubah dan akan dilakukan proses *resize* yang akan disimpan dalam direktori yang telah ditentukan sebelumnya dan akan disimpan sesuai dengan nama file awalnya.

Selain menggunakan *resize* sebagai teknik tranformasi data augmentasi, juga digunakan *rescale* yang akan digunakan pada metode *Xception*. Penggunaan rescale ini dilakukan agar proses komputasi saat training dilakukan menjadi lebih cepat karena hanya mengalokasikan memori yang kecil. Hal ini dikarenakan pada proses ini setiap piksel citra akan diskalakan menjadi nilai antara 0 dan 1 (Ridha, 2022).

**b) *Split* Dataset**

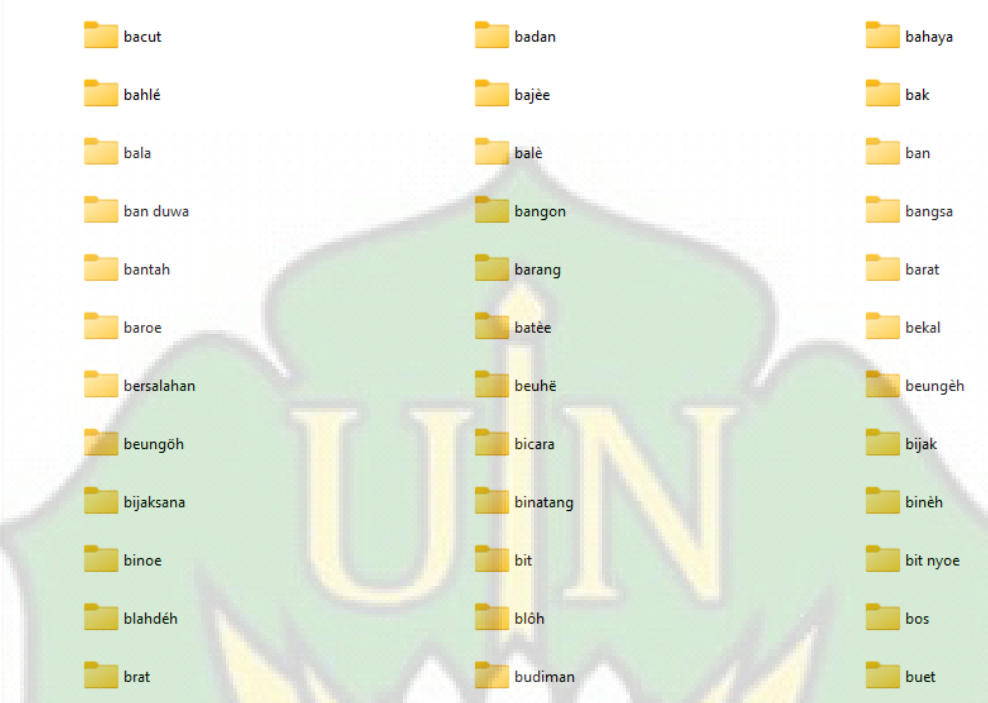
Tahap selanjutnya yang dilakukan adalah melakukan *split* dataset untuk membagi dataset menjadi dua bagian utama yang akan digunakan selama proses pendeteksian dilakukan yaitu dataset *training* dan dataset *testing*. Dalam melakukan pembagian dataset ini akan menggunakan rasio pembagian yaitu 80 : 20. Dengan 80% digunakan untuk dataset *training* dan 20% untuk dataset *testing* sehingga jumlah dataset *training* harus lebih besar dibandingkan jumlah data yang dimiliki dataset *testing*. Hal ini dikarenakan semakin banyak data training yang digunakan maka akan semakin baik model dalam memahami pola yang dimiliki oleh data sehingga hasil akurasi yang diperoleh akan meningkat.

Tabel 3.2 Pembagian dataset

| Label Dataset | Jumlah Dataset yang digunakan |
|---|---|
| Dataset *Training* | 2500 |
| Dataset *Testing* | 500 |
| Total Dataset | 3000 |

Dataset pada yang digunakan pada *Xception* dibentuk dalam bentuk folder untuk tiap kategori atau *class* yang akan digunakan seperti pada Gambar 3.5. Hal ini dilakukan untuk mempermudah pengelompokkan gambar-gambar sesuai

dengan *class* yang dimilikinya. Selain itu, beberapa *library* memiliki fungsi yang memudahkan dalam proses memuat dataset saat penggunaan dataset dibentuk secara terstruktur dalam struktur folder.




Gambar 3.5 Bentuk dataset *Xception*

Sedangkan proses *split* dataset pada *Xception* akan dilakukan secara otomatis dengan menggunakan *library Image Data Generator*. Dan akan dilakukan perubahan pada dataset untuk memiliki tiga *channel* lapisan yaitu RGB, hal ini dikarenakan pada *Xception* hanya menerima gambar dengan memiliki tiga lapisan *channel*.

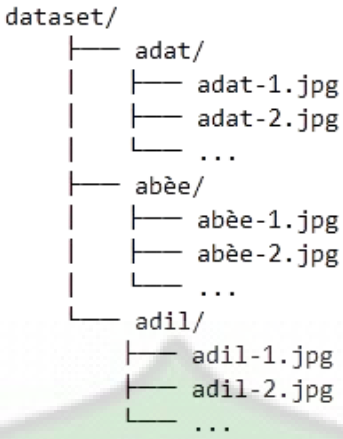
```
#**No Augmentation on the Test set Images**
datagen = ImageDataGenerator(rescale=1./255,
validation_split=0.2) #9:1
#loading the images to training set
train_gen = datagen.flow_from_directory(directory=path,
                                        target_size=(299, 299),
                                        class_mode='categorical',
                                        subset='training',
                                        shuffle=True,
                                        batch_size=batch_size,
                                        color_mode="rgb")
#loading the images to test set
test_gen = datagen.flow_from_directory(directory=path,
                                       target_size=(299, 299),
                                       class_mode='categorical',
                                       subset='validation',
                                       shuffle=False,
                                       batch_size=batch_size,
                                       color_mode="rgb")
```

Gambar 3.6 *Source code split* dataset dan konversi warna

Pada *Source code* diatas dijelaskan bahwa akan didilakukan proses pembagian dataset atau *split* dataset dengan menggunakan perbandingan 80 : 20. Dan selanjutnya akan dilakukan proses pengaturan yang digunakan dalam dataset, seperti pengubahan ukuran, penentuan format *class* gambar yang berbentuk kategori. Selanjutnya pengaturan mengenai tipe data generator digunakan sebagai data *training* atau *testing.* Pengaturan lainnya penentuan ukuran *batch* dan menentukan mode warna yang akan digunakan dalam proses memuat gambar.

**c) Pelabelan Dataset**

Pada arsitektur *Xception,* pelabelan dataset tidak menggunakan *tools* untuk pelabelan gambar. Tetapi pelabelan dataset dilihat berdasarkan struktur folder yang terstruktur secara khusus, seperti menggunakan dataset yang diklasifikasikan dalam beberapa *class* atau kategori, yang disusun seperti struktur pada Gambar 3.7




Gambar 3.7 Struktur folder pada *Xception*

Dalam struktur pada Gambar 3.7, setiap folder akan mengandung gambar-gambar yang terkait dengan kategori atau *class* yang digunakan. Hal ini dikarenakan setiap folder akan mewakili satu class atau kategori tertentu. Sehingga label atau *class* yang digunakan akan diambil berdasarkan jenis kategori atau nama yang diberikan pada folder. Sehingga tidak dibutuhkan file label untuk menentukan label dan jenis *classnya.*

**3. Pembangunan Model**

Pada tahapan ini, akan membangun model yang akan digunakan sebagai media pembelajaran dan pelatihan untuk dapat mengolah dataset dengan menggunakan metode *Convolutional Neural Network* untuk dapat melakukan prediksi. Model yang digunakan pada tahapan ini yaitu dengan menggunakan arsitektur *Xception*.

Pada tahapan ini, akan dibangun sebuah model dengan arsitektur *Xception* menggunakan metode CNN. Dalam proses kerjanya, *Xception* mirip dengan *InceptionResnetV2* yang melakukan *feature learning* pada layer *convolution* yang terpisah. Hasil dari *feature learning* akan dipakai dalam proses pendeteksian. Gambar 3.8 menampilkan proses model arsitektur *Xception*.

```
learning_rate = 1e-5
epochs = 90
Xception = tf.keras.applications.Xception(input_shape = (299, 299, 3),
 weights = 'imagenet',
 include_top = False)
Xception.trainable = True

model = tf.keras.Sequential([
 Xception,
 tf.keras.layers.Flatten(),
 tf.keras.layers.Dense(475, activation = "softmax")],
 name = "Xception_Categorical_Classification")
model.compile(optimizer=Adam(learning_rate=learning_rate),
 loss = "categorical_crossentropy",
 metrics = ["accuracy"])

#check point callback
checkpoint_path = "training/cp.ckpt"
checkpoint_dir  = os.path.dirname(checkpoint_path)
cp_callback = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path, save_weights_only=True, verbose=1)
```

Gambar 3.8 *Source code* Model *Xception*

Pada *Source code* diatas dijelaskan proses dalam Pembangunan model. Proses dimulai dengan mendefiniskan *learning rate*. *Learning rate* merupakan paramenter yang dibutuhkan dalam proses pembelajaran atau *training.* Dimana *learning rate* merupakan parameter yang digunakan untuk mengatur seberapa banyak *steps* yang akan diambil saat melakukan perubahan terhadap bobot dari model selama proses pelatihan berlangsung. *Learning rate* memiliki nilai dengan rentang dari nol (0) sampai (1). Semakin besar nilai *learning rate*, maka proses pelatihan yang berlangsung akan berjalan semakin cepat. Pada penelitian ini jumlah *learning rate* yang digunakan yaitu 5 dan menggunakan jumlah *epoch* yaitu 90 yang artinya proses pelatihan akan melakukan perulangan sebanyak 90 kali terhadap model dalam memahami pola yang dimiliki oleh dataset.

Tahap selanjutnya akan menentukan ukuran *input* yang akan digunakan oleh model yaitu gambar dengan ukuran 299 x 299 piksel dan menggunakan 3 *channel* warna yaitu RGB, kemudian mengambil bobot dari masing-masing *layer* berdasarkan bobot imagenet dan *include top* yang bernilai *false* sehingga lapisan atas dari model *Xception* tidak akan disertakan. *Layer* yang akan digunakan dan dilalui oleh model ini yaitu *flatten* yang akan meratakan *output* yang memiliki banyak dimensi menjadi satu dimensi. *Layer Dense* pada model ini berjumlah sesuai 475 dan menggunakan fungsi aktivasi yaitu *softmax*. Model yang telah dibangun akan disimpan kedalam sebuah file yang akan menyimpan objek *python*. Seperti pada gambar 3.9

```
from keras.models import model_from_json
from keras.models import load_model
import joblib

# Simpan arsitektur model ke dalam file JSON
model_json = model.to_json()
with open("model_architecture.json", "w") as json_file:
    json_file.write(model_json)

# Simpan bobot model ke dalam file HDF5
model.save_weights("model_weights.h5")

# Simpan konfigurasi optimizer
optimizer_config = model.optimizer.get_config()
joblib.dump(optimizer_config, 'optimizer_config.joblib')


# Simpan history ke dalam file Joblib
joblib.dump(history.history, 'training_history.joblib')
```

Gambar 3.9 *Source code* untuk menyimpan file model

Proses penyimpanan ini dilakukan dengan bantuan *library python* yaitu *Joblib.* File *joblib* ini akan mengandung objek yang telah disimpan dan dapat dimuat kembali kedalam program. File *joblib* ini disimpan kedalam *google drive* untuk mempermudah proses pemangilan saat model akan di muat kedalam program. Pada gambar 3.9 dijelaskan bahwa proses penyimpanan dilakukan dalam beberapa bentuk, mulai dari menyimpan dan mengonversikan arsitektur model kedalam format JSON, menyimpan bobot model kedalam file HDF5, melakukan penyimpanan terhadap konfigurasi *optimizer* dan menyimpan model kedalam file *joblib.*

```
from keras.models import model_from_json
from keras.models import load_model
import joblib
from google.colab import drive  # Import drive from colab

# Connect to Google Drive
drive.mount('/content/gdrive')

# Path where the files will be saved in Google Drive
path = '/content/gdrive/MyDrive/TA/Xception_Model/'

# Jumlah epoch tambahan yang akan ditambahkan
additional_epochs = 20

# Melanjutkan pelatihan dengan menambah jumlah epoch
additional_history = model.fit(
    train_gen,
    validation_data=test_gen,
    steps_per_epoch=100,
    batch_size=batch_size,
    epochs=additional_epochs,
    callbacks=[cp_callback]
)
```

Gambar 3.10 *Source code* untuk melakukan penambahan epoch

Gambar 3.10 berisikan *Source code* yang menjelaskan mengenai tahapan-tahapan dalam melakukan penambahan *epoch* tanpa harus melakukan training secara awal kembali. Tahapan yang dilakukan diawal yaitu mendefinisikan Alamat path lokasi file akan disimpan yaitu pada *google drive.* Setelah itu akan mendefinisikan jumlah *epoch* tambahan yang akan dilatih yaitu menggunakan *variable* "*additional_epochs*". Tahapan selanjutnya akan dilakukan proses *training* kembali terhadap dataset *training* dan dataset *testing* sesuai dengan jumlah *epoch* yang digunakan. Untuk melakukan penyimpan dan pembaharuan terhadap model setelah dilakukan *training* kembali dapat dilihat pada gambar 3.11.

```
# Memuat kembali history yang sudah disimpan sebelumnya (jika ada)
try:
    existing_history   joblib.load(path + 'training_history.joblib')
except FileNotFoundError:
    existing_history = {}  # Jika file history belum ada, buat dictionary kosong

# Menggabungkan history lama dan history tambahan
for key in additional_history.history.keys():
    if key in existing_history:
        existing_history[key].extend(additional_history.history[key])
    else:
        existing_history[key] = additional_history.history[key]

# Menyimpan history yang telah diperbarui
joblib.dump(existing_history, path + 'training_history.joblib')

# Simpan arsitektur model ke dalam file JSON
model_json   model.to_json()
with open(path + "model_architecture.json", "w") as json_file:
    json_file.write(model_json)

# Simpan bobot model ke dalam file HDF5
model.save_weights(path + "model_weights.h5")

# Save entire model (architecture, configuration, and weights)
model.save('/content/gdrive/MyDrive/TA/Xception_Model/entire_model.h5')
```

Gambar 3.11 *Source code* untuk pembaharuan file model

Pada gambar diatas dijelaskan bahwa proses dimulai dengan memuat kembali model yang telah disimpan sebelumnya. Dimana pada tahapan ini akan dilakukan pengecekan untuk melihat dan memuat file *joblib* apabila file tersebut ditemukan pada lokasi yang telah diatur. Jika tidak ditemukan file pada lokasi yang telah diatur, maka akan dibuat sebuah folder yaitu "*existing_history*" yang akan menampung model yang akan disimpan. Karena model sebelumnya telah disimpan pada file *joblib*, maka akan dilakukan proses penggabungan antara model awal dengan model setelah penambahan. Setelah dilakukan penambahan, file model akan diperbaharui dan disimpan kembali ke dalam file *joblib* ke lokasi penyimpanan yang sama.

**4. Model**

Pada tahapan ini, model yang telah dibangun akan dilakukan proses pengujian dengan menggunakan dataset *testing*. Hal ini dilakukan untuk melihat seberapa baik model dalam melakukan prediksi terhadap aksara *Jawoe* berdasarkan pada pembelajaran dan pelatihan yang telah dilakukan pada saat membangun

model. Tahapan ini juga akan dilihat tingkat performa yang dihasilkan oleh model terhadap dataset yang telah di inputkan untuk melihat nilai akurat model dalam mengenal dataset yang diujikan.

## 3.5 Metode Analisis Data

Tahapan terakhir dari penelitian ini adalah analisis data. Analisis data ini dilakukan dengan menilai sistem yang digunakan dalam penelitian ini.

### 3.5.1 Evaluasi Akhir

Evaluasi adalah metode yang digunakan untuk mengukur kinerja *system* pada penelitian ini. Peneliti menggunakan *recall, precision*, *f1 score* dan *accuracy* yang mengacu pada model *confusion matrix.* Untuk melihat tingkat nilai *recall* akan diukur dengan menggunakan persamaan (2.1). *Precision* menggunakan persamaan (2.2). untuk melihat nilai *f1 score* menggunakan persamaan (2.3) dan untuk melihat nilai akurasi secara keseluruhan akan menggunakan persamaan (2.4). Dimana evaluasi ini akan membandingkan hasil dari confusion matrix yang terdiri dari jumlah data *true positive*, *true false*, *false positive*, dan *false negative*.

```
# confusion matrix
# transform the predictions into array such as [0,0,1,2...]
y_pred = model.predict(test_gen)
predictions = np.array(list(map(lambda x: np.argmax(x), y_pred)))
#Retrieve the True classes of the test set
y_true=test_gen.classes

classes = list(test_gen.class_indices.keys())
# Build Confusion Matrix
CMatrix = pd.DataFrame(confusion_matrix(y_true, predictions), columns=classes, index =classes)
plt.figure(figsize=(100, 100))
ax = sns.heatmap(CMatrix, annot = True, fmt = 'g' ,vmin = 0, vmax = 250,cmap = 'Blues')
ax.set_xlabel('Predicted',fontsize = 14,weight = 'bold')
ax.set_xticklabels(ax.get_xticklabels(),rotation =90);
ax.set_ylabel('Actual',fontsize = 14,weight = 'bold')
ax.set_yticklabels(ax.get_yticklabels(),rotation =0);
ax.set_title('Confusion Matrix - Test Set',fontsize = 16,weight = 'bold',
pad=20);
```

Gambar 3.12 *Source code confusion Matrix*

Tahap selanjutnya akan divisualisasikan hasil dari *training* model dan akan dihitung nilai akurasinya dengan menggunakan metrik-metrik seperti *recall, precision, F1 score* dan *accuracy*.

```
from sklearn.metrics import accuracy_score, precision_recall_fscore_support
import pandas as pd

# Calculate Accuracy Result
acc = accuracy_score(y_true, predictions)

# Precision, Recall, and F-Score (For the whole dataset)
results_all = precision_recall_fscore_support(y_true, predictions, average='macro',zero_division = 1)

# Precision, Recall, and F-Score (For each Class)
results_class = precision_recall_fscore_support(y_true, predictions, average=None,zero_division = 1)

# Organize the Results into a Dataframe
metric_columns = ['Precision', 'Recall', 'F-Score', 'Support']
all_df = pd.DataFrame(list(results_class)).T
all_df.columns = metric_columns
overall_df = pd.DataFrame(list(results_all)).T
overall_df.columns = metric_columns

print('**Overall Results**')
print('Accuracy Result: %.2f%%' % (acc * 100))  # Accuracy of the whole Dataset
print('Precision Result: %.2f%%' % (overall_df.iloc[0]['Precision'] * 100))  # Precision of the whole Dataset
print('Recall Result: %.2f%%' % (overall_df.iloc[0]['Recall'] * 100))  # Recall of the whole Dataset
print('F-Score Result: %.2f%%' % (overall_df.iloc[0]['F-Score'] * 100))  # FScore of the whole Dataset
```

Gambar 3.13 *Source code* recall, precision, accuracy dan f1-score

## 3.6 Waktu dan Lokasi Penelitian

Penelitian dilakukan bertempat pada laboratorium Multifungsi Fakultas Sains dan Teknologi dengan menggunakan *computer* yang memiliki spesifikasi tinggi. Dimulai sejak tanggal 1 September 2023, yang dilaksanakan dalam kurun waktu kurang lebih 4 bulan.

## 3.7 Alat Bantu Penelitian

Pada penelitian ini, menggunakan alat bantu berupa software dan hardware. Hardware yang digunakan yaitu satu unit laptop merek HP dengan series 14s-fq2 dengan spesifikasi sebagai berikut:

1. Processor AMD Ryzen 7 5825U with Radeon Graphics 2.00 GHz
2. RAM 16,0 GB (15,3 GB usable)
3. Storage 512 GB M.2 NVMe PCIe 3.0 SSD.

Untuk software yang digunakan pada penelitian ini yaitu Sistem Operasi Microsoft Windows 11 *Home Single Language* Version 22H2 dan menggunakan tools lainnya, seperti bahasa pemrograman *Python*, *Tensorflow*, *Keras* dan *Google Collaborary*.

# BAB IV
# HASIL DAN PEMBAHASAN

## 4.1 Hasil Pengujian

Berikut ini merupakan hasil uji coba yang dilakukan dalam penelitian transliterasi aksara *Jawoe* menggunakan metode *Convolutional Neural Network* dengan arsitektur *Xception.*

### 4.1.1 Hasil Proses Training

Pada setiap arsitektur yang dibentuk dan dilakukan proses Pembangunan model, algoritma akan secara langsung melakukan pelatihan terhadap dataset yang telah ditentukan sebelumnya. Penelitian ini menggunakan perbandingan pembagian dataset yaitu 80 : 20 yang artinya 80% dataset akan digunakan dalam proses *training.* Dataset untuk proses *training* ini akan digunakan untuk melakukan pembelajaran oleh model *Xception* untuk dipelajari pola yang dimiliki oleh gambar agar melakukan prediksi terhadap dataset testing.

Pada model *Xception* menggunakan literasi berjumlah 90 *epoch. Epoch* sendiri merupakan parameter yang digunakan untuk menentukan berapa banyak algoritma akan melewati seluruh dataset saat proses berlangsung. Selain itu, penggunaan nilai *batch size* yang berjumlah 100. *Batch size* merupakan jumlah sampel data yang akan diberikan pada model dalam tiap *epoch* selama proses pelatihan. Artinya setiap satu *epoch* akan terpenuhi ketika semua *batch* telah dilewatkan melalui jaringan saraf satu kali. Sehingga selama proses *training*, proses akan dilakukan berulang sebanyak jumlah *epoch* yang digunakan yaitu 90 *epoch*.

Hasil dari proses *training* pada model *Xception* dapat dilihat dari nilai *accuracy* dan *loss score* yang dihasilkan oleh model. Nilai *accuracy* merupakan nilai yang dihasilkan oleh model untuk memberikan gambaran mengenai seberapa akurat dan tepat model dalam memprediksi data dengan benar, semakin tinggi nilai yang diberikan maka akan memberikan nilai *accuracy* yang lebih baik. Sebaliknya,

untuk *loss score*. Berikut merupakan tabel hasil *training* menggunakan 90 *epoch* yang terdiri dari jumlah *epoch, training accuracy, validation accuracy, training loss, validation loss* serta waktu yang dibutuhkan pada setiap *epoch* selama proses *training:*

Tabel 4.1 Hasil *Training* 90 *epoch*

| Epoch | Training Accuracy | Validation Accuracy | Training Loss | Validation Loss | Time (second) |
|---|---|---|---|---|---|
| 10 | 0.9987 | 0.6200 | 0.0172 | 1.8487 | 63 |
| 20 | 1.0000 | 0.6560 | 0.0041 | 1.7501 | 63 |
| 30 | 1.0000 | 0.6680 | 0.0016 | 1.7058 | 62 |
| 40 | 1.0000 | 0.6880 | 8.6856e | 1.6183 | 54 |
| 50 | 1.0000 | 0.6920 | 4.4845e | 1.6066 | 64 |
| 60 | 1.0000 | 0.6900 | 2.7085e | 1.5564 | 58 |
| 70 | 1.0000 | 0.6980 | 2.8988e | 1.5630 | 55 |
| 80 | 1.0000 | 0.7000 | 9.7259e | 1.5419 | 58 |
| 90 | 0.9975 | 0.6760 | 0.0075 | 1.6788 | 63 |

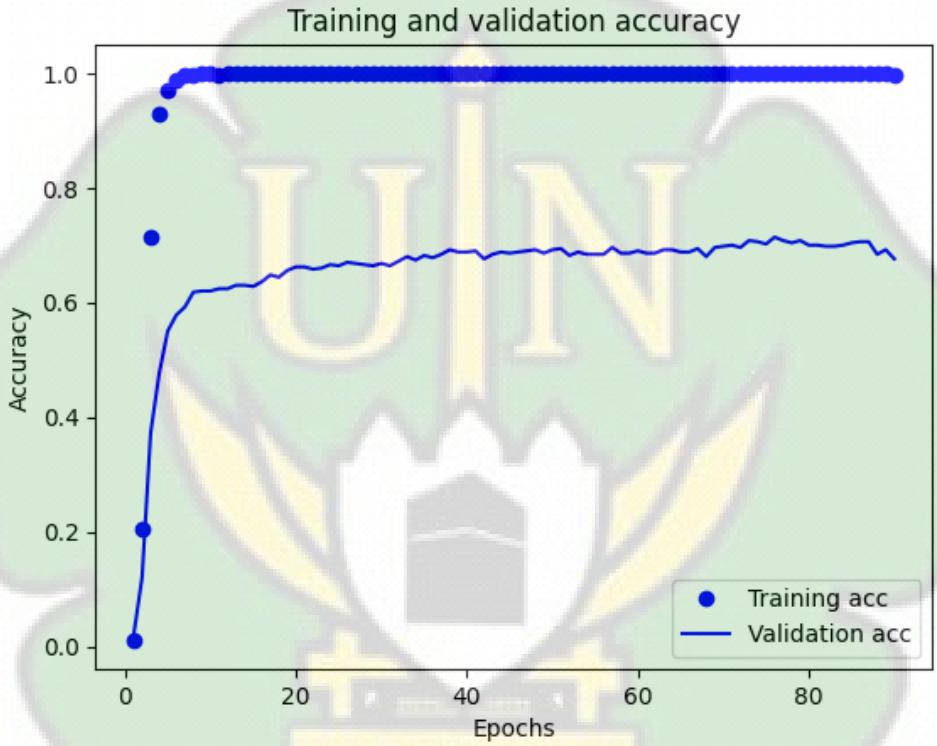
Tabel 4.1 menampilkan hasil *training* yang dilakukan dengan menggunakan 90 *epoch* pada model *Xception*. Hasil *training* saat berada di *epoch* ke-90 menampilkan nilai *training accuracy* yaitu 0.9975, *validation accuracy* yaitu 0.6760, *training loss* yaitu 0.0075 dan *validation loss* yaitu 1.6788.

Pada bagian *training loss* ditunjukkan bahwa nilai yang diperoleh mengalami perubahan yang penting dalam proses *training.* Dimana nilai yang diperoleh terus mengalami penurunan yang mendekati angka nol (0). Hal ini menunjukkan bahwa model memiliki performa yang baik dalam mempelajari pola-pola yang dimiliki oleh data.

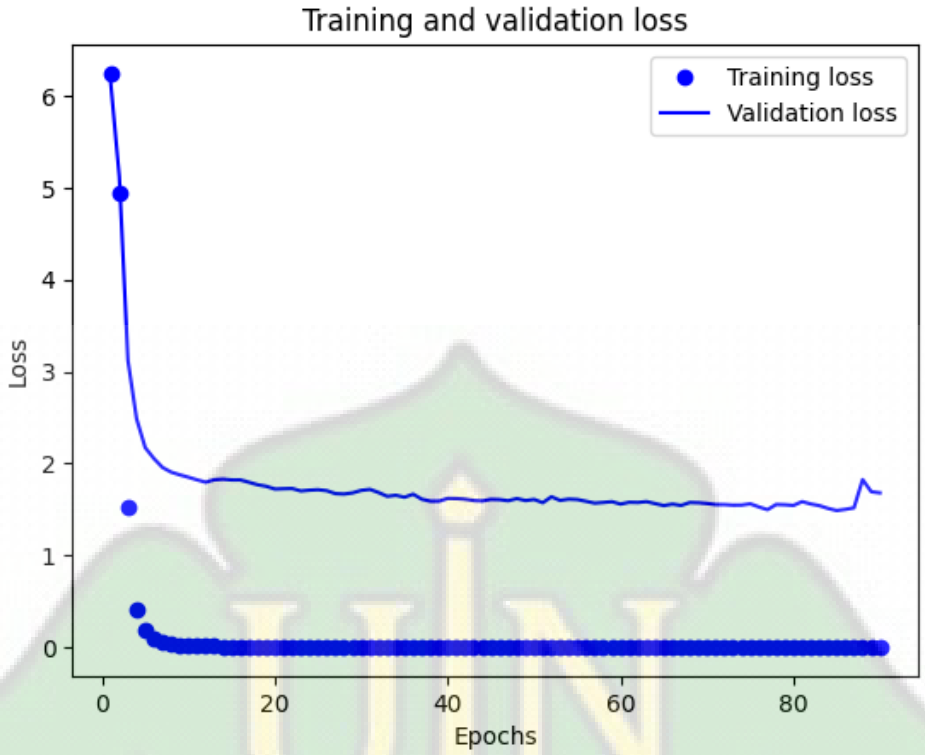
Peningkatan yang terjadi pada hasil *training accuracy* ini menjelaskan bahwa peningkatan pada *epoch* ke-10 dan *epoch* selanjutnya terus memberikan nilai yang baik dan meningkat. Tetapi pada *epoch* ke-90 nilai *training accuracy* mengalami penurunan Saat melalui *epoch* ke-20 menunjukkan nilai yang meningkat dan konsisten. Pada *validation accuracy* menunjukkan perubahan yang

terus menerus hingga *epoch* terakhir yaitu *epoch* ke-90. Hal ini menunjukkan bahwa model mampu mempelajari data-data yang diberikan.

Hasil dari *validation loss* menunjukkan bahwa model tidak dapat melakukan implementasi pembelajaran yang dilakukan dengan baik pada data *testing.* Hal ini disebabkan oleh *overfitting* yang terjadi saat proses *training.* Grafik yang dihasilkan dari proses pelatihan dengan menggunakan 90 epoch dapat dilihat pada gambar 4.1 dan 4.2.




Gambar 4.1 *Training* dan *Validation Accuracy*




Gambar 4.2 Training dan Validation Loss

### 4.1.2 Hasil Proses Testing

Langkah selanjutnya yang dilakukan setelah proses *training* selesai yaitu membandingkan data yang telah di *training* dengan data yang telah dijadikan sebagai data *testing*. Pada penelitian ini menggunakan data *testing* sebanyak 20% yaitu 500 kata.

Tabel 4.2 Hasil testing model

| Testing Loss | Testing Accuracy | Time(s) |
|---|---|---|
| 1.6788 | 0.6760 | 4 |

Tabel 4.2 menunjukkan bahwa nilai *accuracy* dan nilai loss yang dihasilkan dari model *Xception* pada proses *testing*. Dari tabel ini dapat dilihat bahwa nilai *accuracy* yang diperoleh masih rendah yaitu 0.6760 dan nilai *loss* yang didapat cukup tinggi yaitu 1.6788. Hasil ini dapat dipengaruhi oleh terjadinya *overfitting* yang diakibatkan oleh model yang terlalu baik dalam mempelajari data

*training* dan tidak dapat mengimplemetasikan hal yang dipelajarinya dengan data yang belum pernah dilihat seperti data *testing.*

## 4.2 Evaluasi Terhadap Model

Tahap selanjutnya yang dilakukan setelah proses *training* dan *testing* adalah melakukan proses pengukuran terhadap model. Model yang telah dibangun akan digunakan untuk memprediksi data *testing* citra aksara *Jawoe.* Kemudian hasil dari prediksi ini akan diukur menggunakan metrik evaluasi yaitu *confusion matrix, recall, precision, f1 score* dan *accuracy*.

### *4.2.1 Confusion Matrix*

Setelah melakukan proses training selesai, maka akan dilakukan pengukuran untuk menilai seberapa baik model dalam melakukan pendeteksian. Proses pengukuran ini akan menggunakan *confusion matrix* untuk mencari nilai *recall, precision, f1-score* dan *accuracy. Confusion matrix* akan menyajikan ringkasan mengenai hasil prediksi yang dilakukan oleh model terhadap data testing dengan membandingkan nilai *actual* dengan nilai prediksi yang divisualisasikan secara sistematis. *Confusion matrix* akan menampilkan matriks yang berdimensi sesuai dengan banyaknya jumlah class yang digunakan. *Confusion matrix* akan menampilkan hasil prediksi yang benar pada setiap class secara diagonal.

Pada penelitian ini, *confusion matrix* akan memiliki dimensi matriks berjumlah 475x475 karena jumlah class yang digunakan yaitu 475 class. Nilai pada *confusion matrix* yang berada diluar jalur diagonal menunjukkan bahwa model sulit untuk membedakan kelas yang diberikan. Pada *confusion matrix* yang dihasilkan ditemukan masih banyak data *testing* yang tidak bisa di deteksi atau bahkan salah dalam pendeteksian yang dilakukan oleh model.

### *4.2.2 Recall, Precision, Fi-Score* dan *Accuracy*

*Confusion matrix* juga dapat digunakan untuk pengukuran terhadap metrik evaluasi seperti *recall, precision, accuracy* dan *f1-score. Recall* akan menghitung nilai positif sebenarnya yang berhasil dideteksi oleh model. *Precision* akan mengukur keakuratan prediksi positif model. *F1-score* akan menghitung nilai *harmonic* dari *precision* dan *recall.* Dan nilai akurasi akan menghitung nilai

prediksi yang benar dibagi dengan jumlah total sampel yang diuji. Pada tabel 4.2 terlihat *Accuracy* memiliki nilai yaitu 67.60%, *Precision* memiliki nilai yaitu 85.55%, *Recall* memiliki nilai yaitu 66.77% dan nilai *F1-Score* yaitu 61.16%

Tabel 4.3 *Accuracy*, *Precision, Recall* dan *F1-Score*

| **90 epoch** | |
|---|---|
| Accuracy | 67.60% |
| Precision | 85.55% |
| Recall | 66.77% |
| F1 Score | 61.16% |

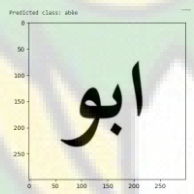
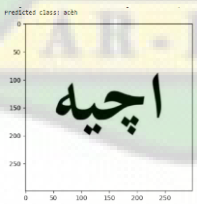
Pada tabel 4.2 dapat dilihat bahwa nilai *precision* yang diperoleh adalah 85.55% yang berarti dari semua data *testing* yang diprediksi oleh model ada 85.55% data yang bernilai benar. Untuk nilai recall, menunjukkan bahwa nilai yang diperoleh yaitu 66.77%. Ini menunjukkan bahwa model dapat mengenali 70% dari data keseluruhan data yang dicari. Untuk nilai *F1-Score* menunjukkan nilai 61.16% yang berarti ini menunjukkan gambaran kesetimbangan yang dimiliki antara *recall* dan *precision*.

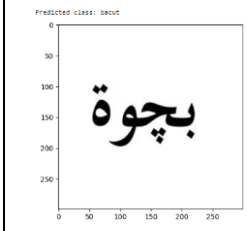
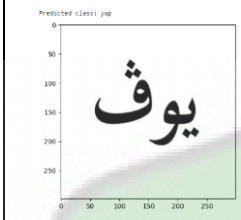
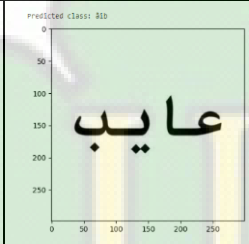
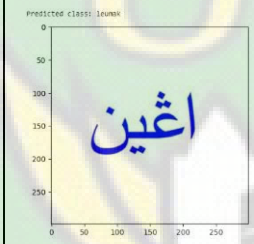
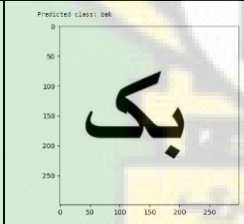
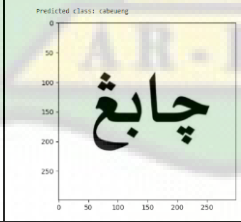
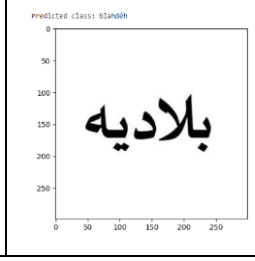
Dari hasil yang diperoleh dapat dilihat bahwa nilai akurasi yang didapat masih cukup rendah, padahal nilai *training loss* yang diperoleh telah menunjukkan performa yang baik dalam mempelajari dataset. Hal ini disebabkan oleh beberapa hal seperti terjadinya *overfitting,* yaitu kondisi dimana data yang digunakan tidak cukup mewakili terhadap kelas, baik karena jumlah data yang digunakan tidak bervariasi. Hal ini menyebabkan model cenderung menghafal data *train* yang tidak memiliki variasi pada datanya dan tidak dapat mengimplementasikan pengetahuan yang dimilikinya pada data baru. Penyebab lainnya dikarenakan data *train* yang digunakan tidak seimbang sehingga kinerja model menjadi kurang optimal dan kesulitan dalam melakukan pendeteksian. Sehingga menyebabkan nilai akurasi yang diperoleh cukup rendah walaupun nilai *training loss* yang didapat cukup bagus.

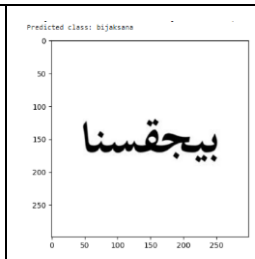
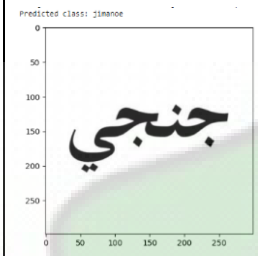
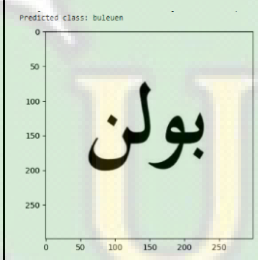
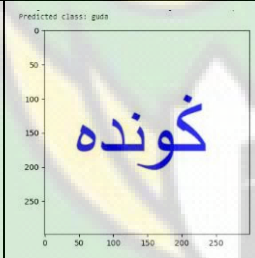
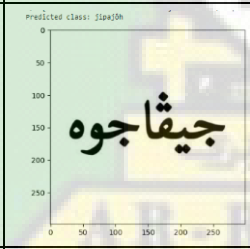
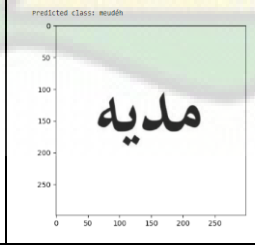
## 4.3 Sampel Pengujian Model

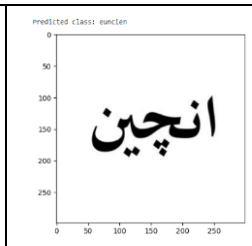
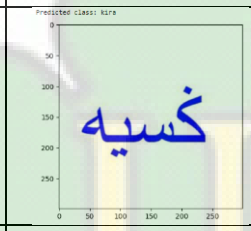
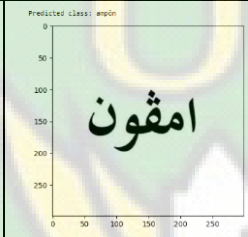
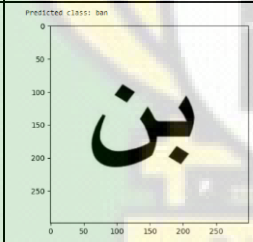
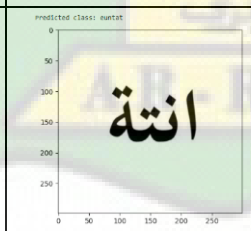
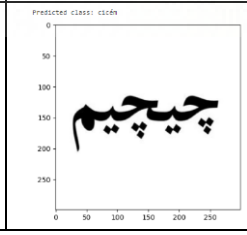
Tahap terakhir yang dilakukan adalah melakukan pengujian terhadap model yang telah *ditraining* menggunakan metode *Convolutional Neural Network* dengan arsitektur *Xception*. Tahapan ini akan memperlihatkan bagaimana model dapat memprediksi atau mendeteksi suatu kata aksara *Jawoe* berdasarkan pelatihan yang telah dilakukan pada tahap sebelumnya. Dalam pengujian ini akan menggunakan 500 sampel data yang akan mewakili tiap-tiap suku kata yang digunakan dalam dataset pada penelitian ini. Tabel 4.3 akan menampilkan 34 sampel pengujian yang dilakukan. Untuk sampel keseluruhan data akan ditampilkan pada lampiran.

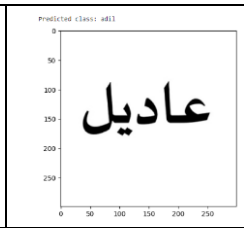
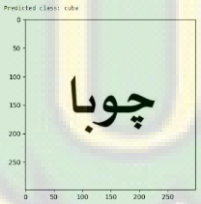
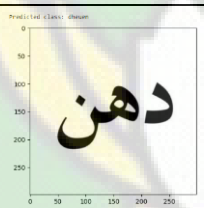
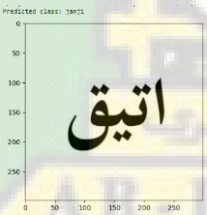
Tabel 4.4 Sampel Pengujian

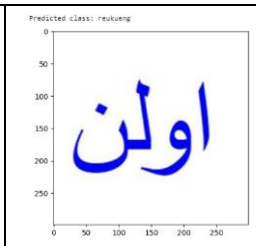
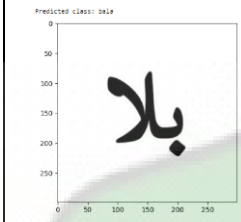
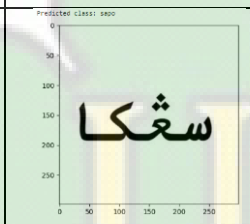
| Kelas | Citra | Waktu(s) | Output | Status Identifikasi |
|---|---|---|---|---|
| Abèe |  | 23s | abèe | Benar |
| Abang |  | 23s | reud | Salah |
| Acèh |  | 25s | acèh | Benar |
| Akal |  | 22s | akal | Benar |

| | | | | |
|---|---|---|---|---|
| Bacut |  | 36s | bacut | Benar |
| Yup |  | 34s | yup | Benar |
| Âib |  | 23s | âib | Benar |
| Angèn |  | 36s | leumak | Salah |
| Bak |  | 33s | bak | Benar |
| Cabeueng |  | 22s | cabeueng | Benar |
| Blahdéh |  | 34s | blahdéh | Benar |

| | | | | |
|---|---|---|---|---|
| Bijaksana |  | 22s | bijaksana | Benar |
| Janji |  | 22s | jimanoe | Salah |
| Buleuen |  | 32s | buleuen | Benar |
| Gundah |  | 23s | guda | Salah |
| Jipajôh |  | 22ms | jipajôh | Benar |
| Meudéh |  | 20ms | meudéh | Benar |

| | | | | |
|---|---|---|---|---|
| Euncien |  | 23s | euncien | Benar |
| Duwa Plôh |  | 24s | duwa plôh | Benar |
| Gaséh |  | 22s | kira | Salah |
| Ampôn |  | 25s | ampôn | Benar |
| Ban |  | 23s | ban | Benar |
| Euntat |  | 18s | euntat | Benar |
| Cicém |  | 20s | cicém | Benar |

| | | | | |
|---|---|---|---|---|
| Adil |  | 23s | adil | Benar |
| Hudéb |  | 22s | iman | Salah |
| Jipeugah |  | 26s | jiplueng | Salah |
| Cuba |  | 24s | cuba | Benar |
| Dheuen |  | 22ms | dheuen | Benar |
| Iték |  | 43s | janji | Salah |
| Pura |  | 36s | rahsiya | salah |

| | | | | |
|---|---|---|---|---|
| Ulôn |   | 40s | reukueng | Salah |
| Bala |   | 23s | bala | Benar |
| Sangka |   | 43s | sapo | Salah |

# BAB V
# KESIMPULAN DAN SARAN

## 5.1 Kesimpulan

Pada penelitian ini, pendeteksian terhadap aksara *Jawoe* dilakukan dengan menggunakan model *Xception.* Dalam proses membangun model untuk dilatih, digunakan dataset berjumlah 3000 gambar yang terdiri dari 475 kelas dengan ukuran gamabr yaitu 299x299 piksel dan jumlah *epoch* yang digunakan mencapai 90 *epoch.* Untuk melihat seberapa baik model dalam melakukan pendeteksian, maka selanjutnya model yang telah *ditraining* akan dievaluasi untuk mengetahui nilai akurasi yang dimilikinya.

Berdasarkan pembahasan dan hasil analisis yang dilakukan dapat disimpulkan bahwa hasil akurasi yang dihasilkan oleh model *xception* masih belum sempurna dan akurat. Hasil akurasi yang diperoleh dari proses pendeteksian mencapai nilai 67.60%. Hal ini dikarenakan terjadinya overfitting yang disebabkan karena model terlalu fokus dalam memahami detail kecil yang dimiliki oleh data dan juga dikarenakan kurangnya data yang bervariasi yang menyebabkan model hanya mempelajari pola data yang mirip dan tidak dapat mengimplementasikan ilmu pengatahuan yang dipelajarinya pada data baru yang baru dilihat. Sehingga menyebabkan model tidak mampu melakukan prediksi dengan baik. Selain itu penggunaan dataset untuk setiap kata yang masih sedikit dan tidak seimbang juga menjadi penyebab terjadinya *overfitting*. Penggunaan jumlah dataset untuk tiap kelasnya yang sedikit merupakan tantangan yang harus dihadapi oleh model dalam mempelajari pola untuk setiap kelasnya.

## 5.2 Saran

Dari hasil yang telah diuji pada penelitian ini, peneliti memberikan beberapa saran untuk penelitian yang lebih lanjut dalam topik yang sama, yaitu sebagai berikut:

1. Menambahkan penggunaan jumlah dataset pada masing-masing kelas untuk proses training agar dapat mengenali sebauh kata menjadi lebih akurat.
2. Mencoba untuk menerapkan model yang telah dibangun dalam mendeteksi aksara *Jawoe* secara *real-time*.

# DAFTAR PUSTAKA


Ellyadi, M. (2022). *Deteksi Tajwid Nun Mati Pada Ayat Al-Quran Dengan Metode Convolutional Neural Network Menggunakan Model Training Ssd Mobilenet*.

Fadillah, R. Z., Irawan, A., Susanty, M., & Artikel, I. (2021). Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (Bisindo). *Jurnal Informatika*, *8*(2). Http://Ejournal.Bsi.Ac.Id/Ejurnal/Index.Php/Ji

Hastomo, W., & Dan Sudjiran, S. (2021). Convolution Neural Network Arsitektur Mobilenet-V2 Untuk Mendeteksi Tumor Otak. *Seminar Nasional Teknologi Informasi Dan Komunikasi Sti&K (Sentik)*, *5*(1).

Hermansyah. (2020). Perspektif Nilai Sejarah Naskah Hikayat Aceh. *Indonesian Journal Of Islamic History And Culture*, *1*(2), 138–146.

Hudaa, S., Teknologi, I., Bisnis, D., & Dahlan, A. (2019). *Sebasa: Jurnal Pendidikan Bahasa Dan Sastra Indonesia Transliterasi, Serapan, Dan Padanan Kata: Upaya Pemutakhiran Istilah Dalam Bahasa Indonesia*.

Kurniawan, A. A., Mustikasari, M., & Korespondensi, P. (2022). *Evaluasi Kinerja Mllib Apache Spark Pada Klasifikasi Berita Palsu Dalam Bahasa Indonesia*. *9*(3). Https://Doi.Org/10.25126/Jtiik.202293538

Kurniawan, R., Wintoro, P. B., Mulyani, Y., & Komarudin, M. (2023). Implementasi Arsitektur Xception Pada Model Machine Learning Klasifikasi Sampah Anorganik. *Jurnal Informatika Dan Teknik Elektro Terapan*, *11*(2). Https://Doi.Org/10.23960/Jitet.V11i2.3034

Lina, Q. (2019, January 2). *Apa Itu Jaringan Neural Konvolusional?* Medium.

Lorentius, C. A., Adipranata, R., & Tjondrowiguno, A. (2019). *Pengenalan Aksara Jawa Dengan Menggunakan Metode Convolutional Neural Network*.

Marcella, D., & Devella, S. (2022). *Klasifikasi Penyakit Mata Menggunakan Convolutional Neural Network Dengan Arsitektur Vgg-19*. *3*(1), 60–70.

Nufus, N., Ariffin, D. M., Satyawan, A. S., Nugraha, R. A. S., Asysyakuur, M. I., Marlina, N. N. A., Parangin, C. H., & Ema, E. (2021). Sistem Pendeteksi Pejalan Kaki Di Lingkungan Terbatas Berbasis Ssd Mobilenet V2 Dengan Menggunakan Gambar 360° Ternormalisasi. *Prosiding Seminar Nasional Sains Teknologi Dan Inovasi Indonesia (Senastindo)*, *3*, 123–134. Https://Doi.Org/10.54706/Senastindo.V3.2021.123

Nur, M., Muhlashin, I., Stefanie, A., Universitas, S., Karawang, J. H., Ronggo, W., & Karawang, I. (2023). Klasifikasi Penyakit Mata Berdasarkan Citra Fundus Menggunakan Yolo V8. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 7, Issue 2).

Rachmawanto, E. H., & Andono, P. N. (2022). Deteksi Karakter Hiragana Menggunakan Metode Convolutional Neural Network. *Jurnal Nasional Pendidikan Teknik Informatika (Janapati)*, *11*(3), 183–191. Https://Doi.Org/10.23887/Janapati.V11i3.50144

Ramala, D. E. (2020). Aksara Jawi: Warisan Budaya Dan Bahasa Masyarakat Alam Melayu Dalam Tinjauan Sosiolingustik. In *Jurnal Islamika* (Vol. 3, Issue 2).

Ridha, M. (2022a). *Pendeteksian Fake Masker Menggunakan Metode Convolutional Neural Network (Cnn) Dengan Arsitektur Xception*.

Ridha, M. (2022b). *Pendeteksian Fake Masker Menggunakan Metode Convolutional Neural Network (Cnn) Dengan Arsitektur Xception*.

Sabri, A. (2022). *Analisis Perbandingan Model Arsitektur Cnn Dalam Pendeteksian Coronavirus Disease Menggunakan Citra X-Ray Paru-Paru (Studi Kasus : Baseline Cnn, Inceptionresnetv2, Vgg19, Dan Xception)*.

Sukma Hanindria Ivan, & Hendry. (2022). *Pengklasifikasian Aksara Jawa Metode Convolutional Neural Network*. *X, No.X*, 1–5.

Wulan Anggraini. (2020). *Deep Learning Untuk Deteksi Wajah Yang Berhijab Menggunakan Algoritma Convolutional Neural Network (Cnn) Dengan Tensorflow*.

# LAMPIRAN

## Lampiran 1 Dataset Aksara Jawoe

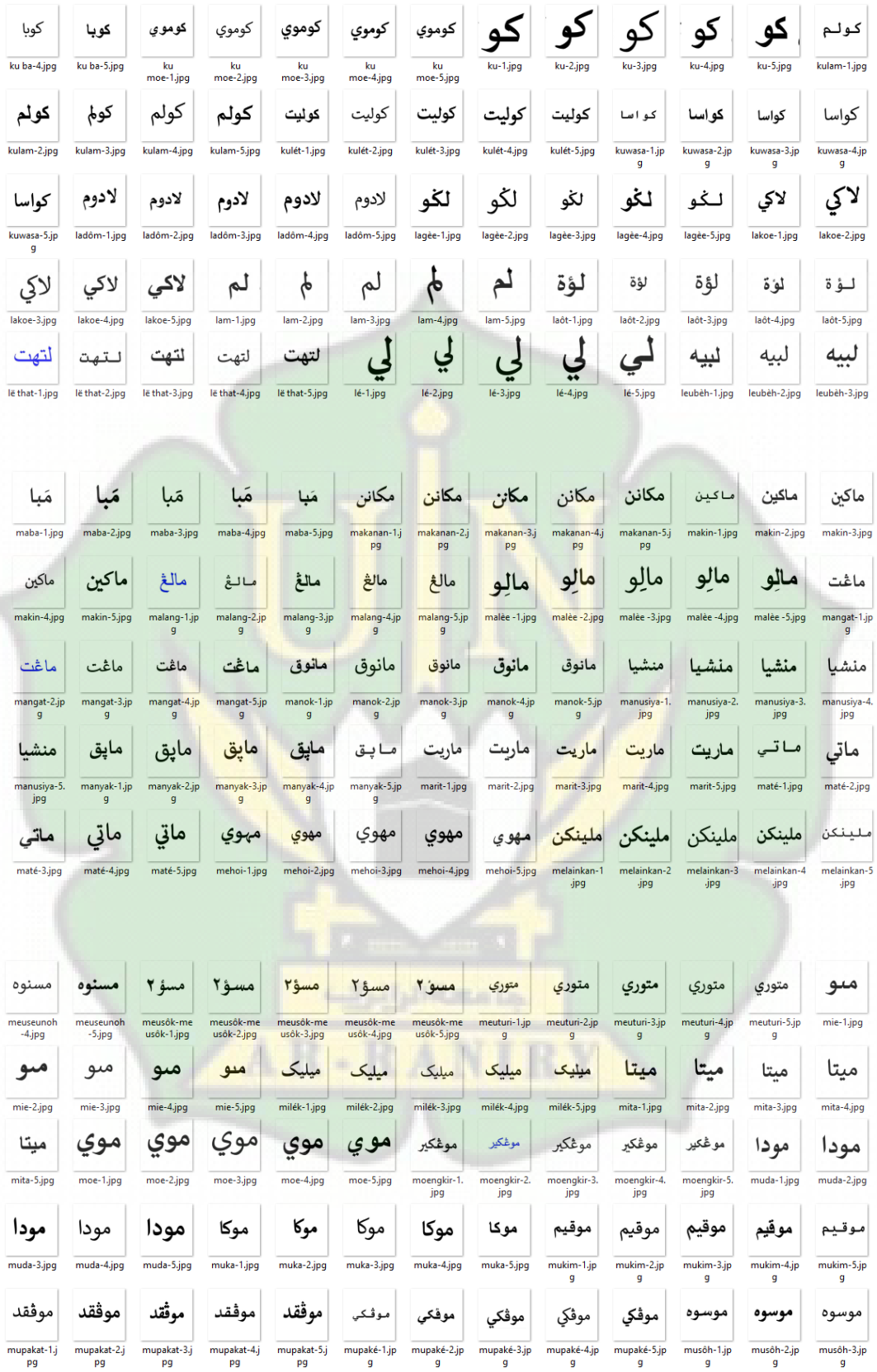

بـک bak-1.jpg
بک bak-2.jpg
بک bak-3.jpg
بک bak-4.jpg
بک bak-5.jpg
بك bak-7.jpg
بك bak-8.jpg
بك bak-9.jpg
بك bak-10.jpg
بك bak-11.jpg
بلا bala-1.jpg
بلا bala-2.jpg
بلا bala-3.jpg
بلا bala-4.jpg
بلا bala-5.jpg
بـالـي balè-1.jpg
بالي balè-2.jpg
بالي balè-3.jpg
بالي balè-4.jpg
بالي balè-5.jpg
بندوا ban duwa-1.jpg
بندوا ban duwa-2.jpg
بندوا ban duwa-3.jpg
بندوا ban duwa-4.jpg
بندوا ban duwa-5.jpg
بـن ban-1.jpg
بن ban-2.jpg
بن ban-3.jpg
بن ban-4.jpg
بن ban-5.jpg
باغن bangon-1.jpg
باغن bangon-2.jpg
باغن bangon-3.jpg
باغن bangon-4.jpg
بـا غن bangon-5.jpg
بـغسا bangsa-1.jpg
بغسا bangsa-2.jpg
بغسا bangsa-3.jpg
بغسا bangsa-4.jpg
بغسا bangsa-5.jpg
بنته bantah-1.jpg
بنته bantah-2.jpg
بنته bantah-3.jpg
بنته bantah-4.jpg
بنته bantah-5.jpg
برغ barang-1.jpg
برغ barang-2.jpg
برغ barang-3.jpg
برغ barang-4.jpg
برغ barang-5.jpg
بـرة barat-1.jpg
برة barat-2.jpg
برة barat-3.jpg
برة barat-4.jpg
برة barat-5.jpg
بارو baroe-1.jpg
بارو baroe-2.jpg
بارو baroe-3.jpg
بارو baroe-4.jpg
بارو baroe-5.jpg


بتو batèe-1.jpg
بتو batèe-2.jpg
بتو batèe-3.jpg
بتو batèe-4.jpg
بتو batèe-5.jpg
بكل bekal-1.jpg
بـكـل bekal-2.jpg
بكل bekal-3.jpg
بكل bekal-4.jpg
بكل bekal-5.jpg
برسلاهن bersalahan-1.jpg
بـرسلاهن bersalahan-2.jpg
برسلاهن bersalahan-3.jpg
برسلاهن bersalahan-4.jpg
برسلاهن bersalahan-5.jpg
بـهـر beuhë-1.jpg
بهر beuhë-2.jpg
بهر beuhë-3.jpg
بهر beuhë-4.jpg
بهر beuhë-5.jpg
بغيه beungèh-1.jpg
بغيه beungèh-2.jpg
بغيه beungèh-3.jpg
بغيه beungèh-4.jpg
بـغـيـه beungèh-5.jpg
بـغـو ه beungöh-1.jpg
بغوه beungöh-2.jpg
بغوه beungöh-3.jpg
بغوه beungöh-4.jpg
بغوه beungöh-5.jpg
بـيـجـا ر ا bicara-1.jpg
بيچارا bicara-2.jpg
بيچارا bicara-3.jpg
بيچارا bicara-4.jpg
بيچارا bicara-5.jpg
بيجق bijak-1.jpg
بيجق bijak-2.jpg
بيجق bijak-3.jpg
بيجق bijak-4.jpg
بيجق bijak-5.jpg
بجق bijak-7.jpg
بـجق bijak-8.jpg
بجق bijak-9.jpg
بجق bijak-10.jpg
بجق bijak-11.jpg
بجقسنا bijaksana-1.jpg
بـجقسنا bijaksana-2.jpg
بجقسنا bijaksana-3.jpg
بجقسنا bijaksana-4.jp
بجقسنا bijaksana-5.jp
بيجقسنا bijaksana-7.jp
بيجقسنا bijaksana-8.jp
بيجقسنا bijaksana-9.jp
بيجقسنا bijaksana-10.jp
بيجقسنا bijaksana-11.jp
بِناتغ binatanq-1.jpq
بِناتغ binatanq-2.jpq
بِناتغ binatanq-3.jpq
بِناتغ binatanq-4.jpq
بـنـا تـغ binatanq-5.jpq

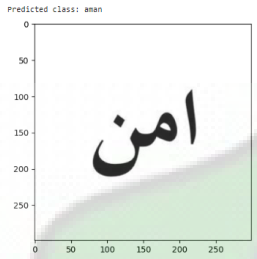
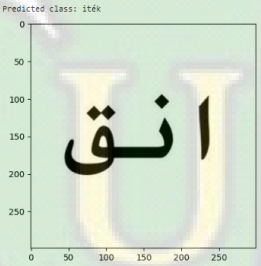
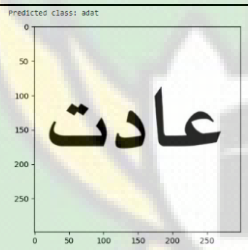
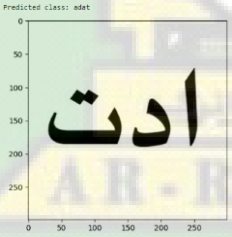
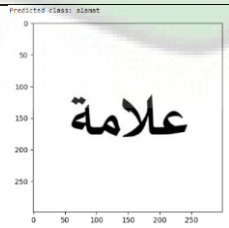

بينيه binèh-1.jpg
بـيـنـيـه binèh-2.jpg
بينيه binèh-3.jpg
بينيه binèh-4.jpg
بينيه binèh-5.jpg
بِنُي binoe-1.jpg
بِنُي binoe-2.jpg
بِنُي binoe-3.jpg
بِنُي binoe-4.jpg
بِنُي binoe-5.jpg
بيتپو bit nyoe-1.jpg
بـيـتپو bit nyoe-2.jpg
بيتپو bit nyoe-3.jpg
بيتپو bit nyoe-4.jpg
بيتپو bit nyoe-5.jpg
بيت bit-1.jpg
بيت bit-2.jpg
بيت bit-3.jpg
بيت bit-4.jpg
بيت bit-5.jpg
بلاديه blahdèh-1.jpg
بلاديه blahdèh-2.jpg
بلاديه blahdèh-3.jpg
بلاديه blahdèh-4.jpg
بلاديه blahdèh-5.jpg
بلوه blôh-1.jpg
بلوه blôh-2.jpg
بلوه blôh-3.jpg
بلوه blôh-4.jpg
بلوه blôh-5.jpg
بوس bos-1.jpg
بوس bos-2.jpg
بوس bos-3.jpg
بوس bos-4.jpg
بوس bos-5.jpg
برت brat-1.jpg
برت brat-2.jpg
برت brat-3.jpg
برت brat-4.jpg
برت brat-5.jpg
بوديمن budiman-1.jpg
بوديمن budiman-2.jpg
بوديمن budiman-3.jpg
بوديمن budiman-4.jpg
بوديمن budiman-5.jpg
بوة buet-1.jpg
بوة buet-2.jpg
بوة buet-3.jpg
بوة buet-4.jpg
بوة buet-5.jpg
بولن buleuen -1.jpg
بولن buleuen -2.jpg
بولن buleuen -3.jpg
بولن buleuen -4.jpg
بولن buleuen -5.jpg
بوغوغ bungong-1.jp g
بـو غـو غ bungong-2.jp g
بوغوغ bungong-3.jp g
بوغوغ bungong-4.jp g
بوغوغ bungong-5.jp g

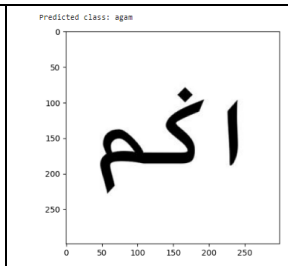
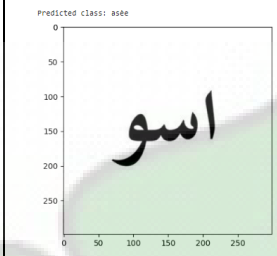
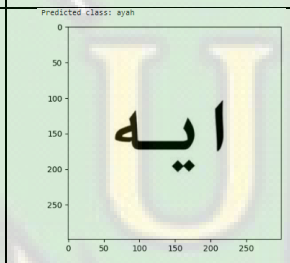
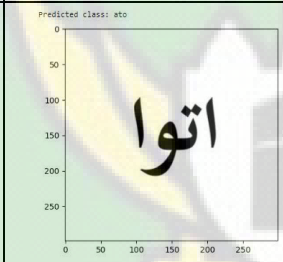
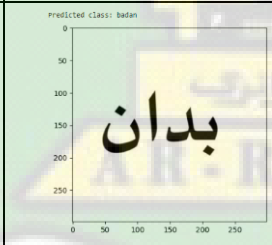
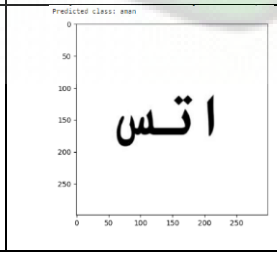

بـونـي بونٮ بونٮ بونٮ بونٮ چـابـغ چابغ چابغ چابغ چابغ چلكا چلكا
bunoe-1.jpg bunoe-2.jpg bunoe-3.jpg bunoe-4.jpg bunoe-5.jpg cabeueng-1.jpg cabeueng-2.jpg cabeueng-3.jpg cabeueng-4.jpg cabeueng-5.jpg celaka-1.jpg celaka-2.jpg
چلكا چلكا چـلـكـا چيچيم چيچيم چيچيم چيچيم چيچيم پؤ پؤ پؤ پؤ
celaka-3.jpg celaka-4.jpg celaka-5.jpg cicém-1.jpg cicém-2.jpg cicém-3.jpg cicém-4.jpg cicém-5.jpg cok-1.jpg cok-2.jpg cok-3.jpg cok-4.jpg
پؤ چـوبـا چوبا چوبا چوبا چوبا چوبا چوة چوة چوة چوة چوة دادا
cok-5.jpg cuba-1.jpg cuba-2.jpg cuba-3.jpg cuba-4.jpg cuba-5.jpg cuet-1.jpg cuet-2.jpg cuet-3.jpg cuet-4.jpg cuet-5.jpg dada-1.jpg
دادا دادا دادا دادا دالم دالم د ا لـم دالم دالم دغن دغن دغن
dada-2.jpg dada-3.jpg dada-4.jpg dada-5.jpg dalam-1.jpg dalam-2.jpg dalam-3.jpg dalam-4.jpg dalam-5.jpg dengon-1.jpg dengon-2.jpg dengon-3.jpg
دغن دغن دغو دغو دغو دغو دغو دهن دهن دهن دهن دهن
dengon-4.jpg dengon-5.jpg deungö-1.jpg deungö-2.jpg deungö-3.jpg deungö-4.jpg deungö-5.jpg dheuen-1.jpg dheuen-2.jpg dheuen-3.jpg dheuen-4.jpg dheuen-5.jpg
ديكتا ديكتا ديكتا ديكتا ديـكـتـا ديليكوة ديليكوة ديليكوة ديليكوة ديليكوة ديلُن ديلُن
di gata-1.jpg di gata-2.jpg di gata-3.jpg di gata-4.jpg di gata-5.jpg di likôt-1.jpg di likôt-2.jpg di likôt-3.jpg di likôt-4.jpg di likôt-5.jpg di lôn-1.jpg di lôn-2.jpg
ديلُن ديلُن ديلُن ديـڤـت ديڤت ديڤت ديڤت ديڤت دي ڤت دي ڤت دي ڤت دي ڤت
di lôn-3.jpg di lôn-4.jpg di lôn-5.jpg di pat-1.jpg di pat-2.jpg di pat-3.jpg di pat-4.jpg di pat-5.jpg di pat-7.jpg di pat-8.jpg di pat-9.jpg di pat-10.jpg
دي ڤت دي دي دي دي دي ديدالم ديدالم ديدالم ديدالم ديدالم ديلو
di pat-11.jpg di-1.jpg di-2.jpg di-3.jpg di-4.jpg di-5.jpg didalam-1.jpg didalam-2.jpg didalam-3.jpg didalam-4.jpg didalam-5.jpg dilée-1.jpg
ديلو ديلو ديلو ديلو دونپا دونپا دونپا دونپا دونپا دري دري دري
dilée-2.jpg dilée-3.jpg dilée-4.jpg dilée-5.jpg dônya-1.jpg dônya-2.jpg dônya-3.jpg dônya-4.jpg dônya-5.jpg droe-1.jpg droe-2.jpg droe-3.jpg
دري دري دؤ دؤ دؤ دؤ دؤ دوا ڤوله دوا ڤوله دوا ڤوله دوا ڤوله دوا ڤوله
droe-4.jpg droe-5.jpg duek-1.jpg duek-2.jpg duek-3.jpg duek-4.jpg duek-5.jpg duwa plôh-1.jpg duwa plôh-2.jpg duwa plôh-3.jpg duwa plôh-4.jpg duwa plôh-5.jpg
دوا دوا دوا دوا دوا انـچـين انچين انچين انچين انچين اغكوة ا غ كـو ة
duwa-1.jpg duwa-2.jpg duwa-3.jpg duwa-4.jpg duwa-5.jpg euncien-1.jpg euncien-2.jpg euncien-3.jpg euncien-4.jpg euncien-5.jpg eungkôt-1.jpg eungkôt-2.jpg
اغكوة اغكوة ا غ كـو ة انتة انتة انتة انتة انتة كدوه كدوه كدوه كدوه
eungkôt-3.jpg eungkôt-4.jpg eungkôt-5.jpg euntat-1.jpg euntat-2.jpg euntat-3.jpg euntat-4.jpg euntat-5.jpg Gadöh-1.jpg Gadöh-2.jpg Gadöh-3.jpg Gadöh-4.jpg
كدوه كادوه ك ا د و ه كادوه كادوه كادوه كمڤوڠ كمڤوڠ كمڤوڠ كمڤوڠ كمڤوڠ كسيه
Gadöh-5.jpg Gadöh-7.jpg Gadöh-8.jpg Gadöh-9.jpg Gadöh-10.jpg Gadöh-11.jpg gampong-1.jpg gampong-2.jpg gampong-3.jpg gampong-4.jpg gampong-5.jpg gaséh-1.jpg
كسيه كسيه كسيه كسيه كاسيه كاسيه كاسيه كاسيه كاسيه كتا كتا كتا
gaséh-2.jpg gaséh-3.jpg gaséh-4.jpg gaséh-5.jpg gaséh-7.jpg gaséh-8.jpg gaséh-9.jpg gaséh-10.jpg gaséh-11.jpg gata-1.jpg gata-2.jpg gata-3.jpg
كتا كتا كچؤ كـچـؤ كچؤ كچؤ كچؤ كـڤـيـه كڤيه كڤيه كڤيه كڤيه
gata-4.jpg gata-5.jpg geucok-1.jpg geucok-2.jpg geucok-3.jpg geucok-4.jpg geucok-5.jpg geupèh-1.jpg geupèh-2.jpg geupèh-3.jpg geupèh-4.jpg geupèh-5.jpg

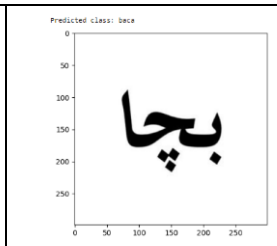
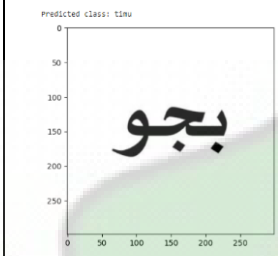
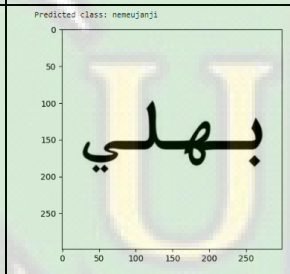
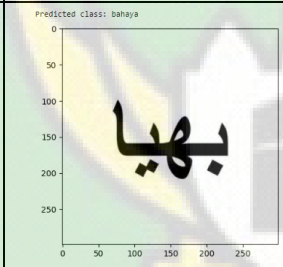
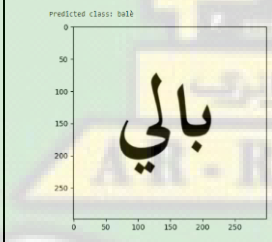
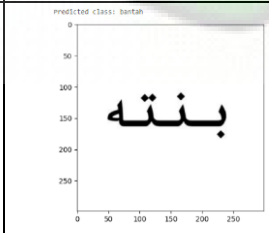

كْريواغْ geuriwang-1.jpg
كْريواغْ geuriwang-2.jpg
كْريواغْ geuriwang-3.jpg
كْريواغْ geuriwang-4.jpg
كْريـو اغْ geuriwang-5.jpg
كْتيُي geutanyoe-1.jpg
كْتيُي geutanyoe-2.jpg
كْتيُي geutanyoe-3.jpg
كْتيُي geutanyoe-4.jpg
كْتيُي geutanyoe-5.jpg
كْـتـوغْ geutueng-1.jpg
كْتوغْ geutueng-2.jpg
كْتوغْ geutueng-3.jpg
كْتوغْ geutueng-4.jpg
كْتوغْ geutueng-5.jpg
كْيكْي gigoe-1.jpg
كْيكْي gigoe-2.jpg
كْيكْي gigoe-3.jpg
كْيكْي gigoe-4.jpg
كْيكْي gigoe-5.jpg
كْـلـغْ gleueng-1.jpg
كْلغْ gleueng-2.jpg
كْلغْ gleueng-3.jpg
كْلغْ gleueng-4.jpg
كْلغْ gleueng-5.jpg
كْـوب gob-1.jpg
كْوب gob-2.jpg
كْوب gob-3.jpg
كْوب gob-4.jpg
كْوب gob-5.jpg
كْولم goelam-1.jpg
كْولم goelam-2.jpg
كْولم goelam-3.jpg
كْولم goelam-4.jpg
كْولم goelam-5.jpg
كْوغْ gông-1.jpg
كْوغْ gông-2.jpg
كْوغْ gông-3.jpg
كْوغْ gông-4.jpg
كْـوغْ gông-5.jpg
كْـو د ا guda-1.jpg
كْودا guda-2.jpg
كْودا guda-3.jpg
كْودا guda-4.jpg
كْودا guda-5.jpg
كونا guna-1.jpg
كونا guna-2.jpg
كونا guna-3.jpg
كونا guna-4.jpg
كْـونـه gundah-1.jpg
كْونده gundah-2.jpg
كْونده gundah-3.jpg
كْونده gundah-4.jpg
كْونده gundah-5.jpg
هنيئك han èk-1.jpg
هنيئك han èk-2.jpg
هنيئك han èk-3.jpg
هنيئك han èk-4.jpg
هنيئك han èk-5.jpg
هنجيد han jid-1.jpg
هنجيد han jid-2.jpg
هنجيد han jid-3.jpg
هنجيد han jid-4.jpg
هنجيد han jid-5.jpg
هنا hana-1.jpg
هنا hana-2.jpg
هنا hana-3.jpg
هنا hana-4.jpg
هنا hana-5.jpg
هـن hana-7.jpg
هن hana-8.jpg
هن hana-9.jpg
هن hana-10.jpg
هن hana-11.jpg
هنتوم hantom-1.jpg
هنتوم hantom-2.jpg
هنتوم hantom-3.jpg
هنتوم hantom-4.jpg
هنتوم hantom-5.jpg
هارڤ harap-1.jpg
هارڤ harap-2.jpg
هارڤ harap-3.jpg
هارڤ harap-4.jpg
هارڤ harap-5.jpg
هتي haté-1.jpg
هتي haté-2.jpg
هـتـي haté-3.jpg
هتي haté-4.jpg
هتي haté-5.jpg
هي héi-1.jpg
هي héi-2.jpg
هي héi-3.jpg
هي héi-4.jpg
هـي héi-5.jpg
حِكمت hékeumat-1.jpg
حِكمت hékeumat-2.jpg
حِكمت hékeumat-3.jpg
حِكمت hékeumat-4.jpg
حِكمت hékeumat-5.jpg
حيكاية hikayat-1.jpg
حيكاية hikayat-2.jpg
حيكاية hikayat-3.jpg
حيكاية hikayat-4.jpg
حيكاية hikayat-5.jpg
هو ho-1.jpg
هو ho-2.jpg
هو ho-3.jpg
هو ho-4.jpg
هو ho-5.jpg
هوك hoka-1.jpg
هوك hoka-2.jpg
هوك hoka-3.jpg
هوك hoka-4.jpg
هوك hoka-5.jpg
هوكا hoka-7.jpg
هوكا hoka-8.jpg
هوكا hoka-9.jpg
هوكا hoka-10.jpg
هوكا hoka-11.jpg
هوديب hudéb-1.jpg
هوديب hudéb-2.jpg
هوديب hudéb-3.jpg
هوديب hudéb-4.jpg
هوديب hudéb-5.jpg
هوكوم hukôm-1.jpg
هوكوم hukôm-2.jpg
هوكوم hukôm-3.jpg
هوكوم hukôm-4.jpg
هوكوم hukôm-5.jpg
ايم iem-1.jpg
ايم iem-2.jpg
ايم iem-3.jpg
ايم iem-4.jpg
ايم iem-5.jpg
ايمان iman-1.jpg
ايمان iman-2.jpg
ايمان iman-3.jpg
ايمان iman-4.jpg
ايمان iman-5.jpg
ايـغـت ingat-1.jpg
ايغت ingat-2.jpg
ايغت ingat-3.jpg
ايغت ingat-4.jpg
ايغت ingat-5.jpg
اينوغْ inông-1.jpg
اينوغْ inông-2.jpg
اينوغْ inông-3.jpg
اينوغْ inông-4.jpg
اينوغْ inông-5.jpg
إنـسـان insan-1.jpg
اِنسان insan-2.jpg
اِنسان insan-3.jpg
اِنسان insan-4.jpg
اِنسان insan-5.jpg
اتيق iték-1.jpg
اتيق iték-2.jpg
اتيق iték-3.jpg
اتيق iték-4.jpg
اتيق iték-5.jpg
جاديه jadèh-1.jpg
جاديه jadèh-2.jpg
جاديه jadèh-3.jpg
جاديه jadèh-4.jpg
جاديه jadèh-5.jpg
جك jak-1.jpg
جك jak-2.jpg
جك jak-3.jpg
جك jak-4.jpg
جك jak-5.jpg
جک jak-7.jpg
جک jak-8.jpg
جک jak-9.jpg
جک jak-10.jpg
جک jak-11.jpg
جنجي janji-1.jpg
جنجي janji-2.jpg
جنجي janji-3.jpg
جنجي janji-4.jpg
جنجي janji-5.jpg

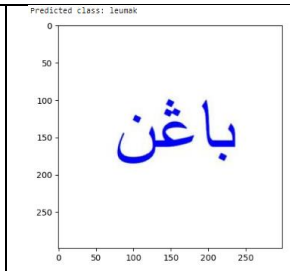
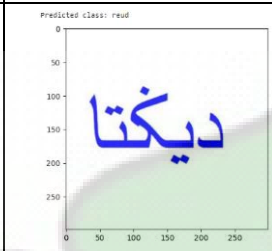
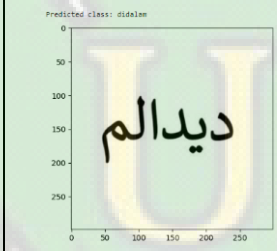
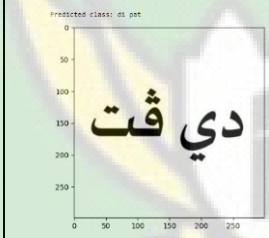
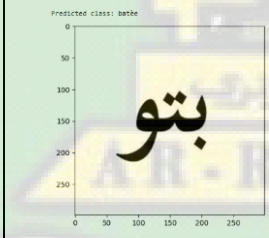
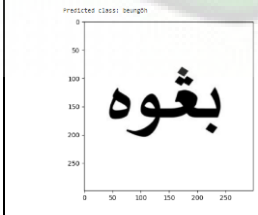

كوبا ku ba-4.jpg
كوبا ku ba-5.jpg
كوموي ku moe-1.jpg
كوموي ku moe-2.jpg
كوموي ku moe-3.jpg
كوموي ku moe-4.jpg
كوموي ku moe-5.jpg
كو ku-1.jpg
كو ku-2.jpg
كو ku-3.jpg
كو ku-4.jpg
كو ku-5.jpg
كولم kulam-1.jpg
كولم kulam-2.jpg
كولم kulam-3.jpg
كولم kulam-4.jpg
كولم kulam-5.jpg
كوليت kulét-1.jpg
كوليت kulét-2.jpg
كوليت kulét-3.jpg
كوليت kulét-4.jpg
كوليت kulét-5.jpg
كو اسا kuwasa-1.jpg
كواسا kuwasa-2.jpg
كواسا kuwasa-3.jpg
كواسا kuwasa-4.jpg
كواسا kuwasa-5.jpg
لادوم ladôm-1.jpg
لادوم ladôm-2.jpg
لادوم ladôm-3.jpg
لادوم ladôm-4.jpg
لادوم ladôm-5.jpg
لكو lagèe-1.jpg
لكو lagèe-2.jpg
لكو lagèe-3.jpg
لكو lagèe-4.jpg
لكو lagèe-5.jpg
لاكي lakoe-1.jpg
لاكي lakoe-2.jpg
لاكي lakoe-3.jpg
لاكي lakoe-4.jpg
لاكي lakoe-5.jpg
لم lam-1.jpg
لم lam-2.jpg
لم lam-3.jpg
لم lam-4.jpg
لم lam-5.jpg
لؤة laôt-1.jpg
لؤة laôt-2.jpg
لؤة laôt-3.jpg
لؤة laôt-4.jpg
لؤة laôt-5.jpg
لتهت lè that-1.jpg
لتهت lè that-2.jpg
لتهت lè that-3.jpg
لتهت lè that-4.jpg
لتهت lè that-5.jpg
لي lé-1.jpg
لي lé-2.jpg
لي lé-3.jpg
لي lé-4.jpg
لي lé-5.jpg
لبيه leubèh-1.jpg
لبيه leubèh-2.jpg
لبيه leubèh-3.jpg
مَبا maba-1.jpg
مَبا maba-2.jpg
مَبا maba-3.jpg
مَبا maba-4.jpg
مَبا maba-5.jpg
مكانن makanan-1.jpg
مكانن makanan-2.jpg
مكانن makanan-3.jpg
مكانن makanan-4.jpg
مكانن makanan-5.jpg
ماكين makin-1.jpg
ماكين makin-2.jpg
ماكين makin-3.jpg
ماكين makin-4.jpg
ماكين makin-5.jpg
مالغ malang-1.jpg
مالغ malang-2.jpg
مالغ malang-3.jpg
مالغ malang-4.jpg
مالغ malang-5.jpg
مالِو malée -1.jpg
مالِو malée -2.jpg
مالِو malée -3.jpg
مالِو malée -4.jpg
مالِو malée -5.jpg
ماغت mangat-1.jpg
ماغت mangat-2.jpg
ماغت mangat-3.jpg
ماغت mangat-4.jpg
ماغت mangat-5.jpg
مانوق manok-1.jpg
مانوق manok-2.jpg
مانوق manok-3.jpg
مانوق manok-4.jpg
مانوق manok-5.jpg
منشيا manusiya-1.jpg
منشيا manusiya-2.jpg
منشيا manusiya-3.jpg
منشيا manusiya-4.jpg
منشيا manusiya-5.jpg
ماپق manyak-1.jpg
ماپق manyak-2.jpg
ماپق manyak-3.jpg
ماپق manyak-4.jpg
ماپق manyak-5.jpg
ماريت marit-1.jpg
ماريت marit-2.jpg
ماريت marit-3.jpg
ماريت marit-4.jpg
ماريت marit-5.jpg
ماتي maté-1.jpg
ماتي maté-2.jpg
ماتي maté-3.jpg
ماتي maté-4.jpg
ماتي maté-5.jpg
مهوي mehoi-1.jpg
مهوي mehoi-2.jpg
مهوي mehoi-3.jpg
مهوي mehoi-4.jpg
مهوي mehoi-5.jpg
ملينكن melainkan-1.jpg
ملينكن melainkan-2.jpg
ملينكن melainkan-3.jpg
ملينكن melainkan-4.jpg
ملينكن melainkan-5.jpg
مسنوه meuseunoh-4.jpg
مسنوه meuseunoh-5.jpg
مسؤه meusôk-meusôk-1.jpg
مسؤه meusôk-meusôk-2.jpg
مسؤه meusôk-meusôk-3.jpg
مسؤه meusôk-meusôk-4.jpg
مسؤه meusôk-meusôk-5.jpg
متوري meuturi-1.jpg
متوري meuturi-2.jpg
متوري meuturi-3.jpg
متوري meuturi-4.jpg
متوري meuturi-5.jpg
مىو mie-1.jpg
مىو mie-2.jpg
مىو mie-3.jpg
مىو mie-4.jpg
مىو mie-5.jpg
ميليک milék-1.jpg
ميليک milék-2.jpg
ميليک milék-3.jpg
ميليک milék-4.jpg
ميليک milék-5.jpg
ميتا mita-1.jpg
ميتا mita-2.jpg
ميتا mita-3.jpg
ميتا mita-4.jpg
ميتا mita-5.jpg
موي moe-1.jpg
موي moe-2.jpg
موي moe-3.jpg
موي moe-4.jpg
موي moe-5.jpg
موڠكير moengkir-1.jpg
موڠكير moengkir-2.jpg
موڠكير moengkir-3.jpg
موڠكير moengkir-4.jpg
موڠكير moengkir-5.jpg
مودا muda-1.jpg
مودا muda-2.jpg
مودا muda-3.jpg
مودا muda-4.jpg
مودا muda-5.jpg
موكا muka-1.jpg
موكا muka-2.jpg
موكا muka-3.jpg
موكا muka-4.jpg
موكا muka-5.jpg
موقيم mukim-1.jpg
موقيم mukim-2.jpg
موقيم mukim-3.jpg
موقيم mukim-4.jpg
موقيم mukim-5.jpg
موڤقد mupakat-1.jpg
موڤقد mupakat-2.jpg
موڤقد mupakat-3.jpg
موڤقد mupakat-4.jpg
موڤقد mupakat-5.jpg
موڤكي mupaké-1.jpg
موڤكي mupaké-2.jpg
موڤكي mupaké-3.jpg
موڤكي mupaké-4.jpg
موڤكي mupaké-5.jpg
موسوه musôh-1.jpg
موسوه musôh-2.jpg
موسوه musôh-3.jpg

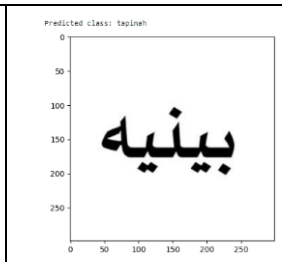
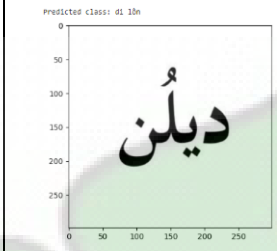
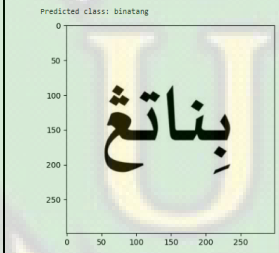
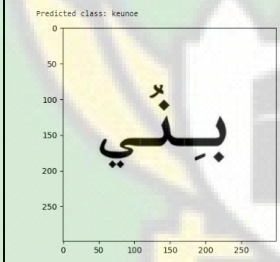
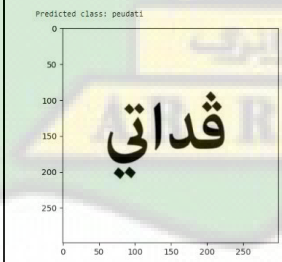
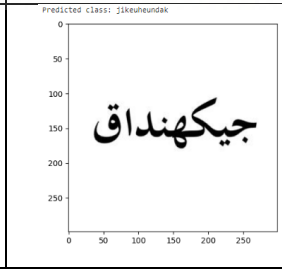
**Lampiran 2 Sampel Pengujian Model**

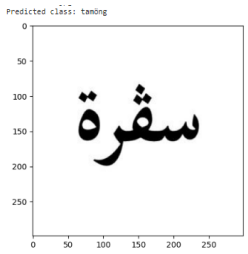
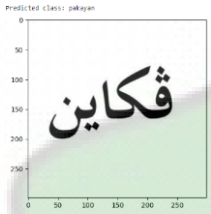
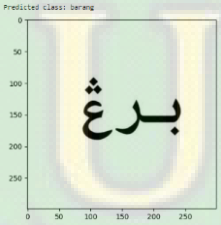
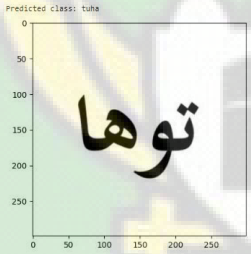
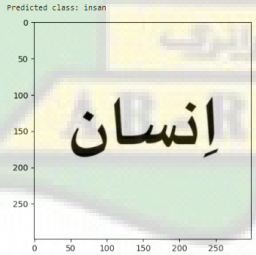
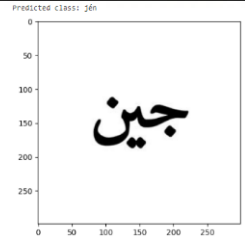
| Kelas | Citra | Waktu (second) | Output | Status Identifikasi |
|---|---|---|---|---|
| Aman |   | 41 | aman | Benar |
| Aneuk |   | 23 | iték | Salah |
| Adat |   | 40 | adat | Benar |
| Adat |   | 19 | adat | Benar |
| Alamat |   | 44 | alamat | Benar |

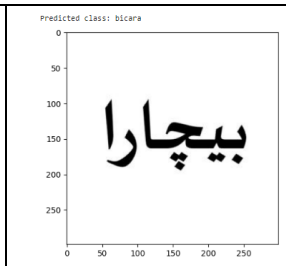
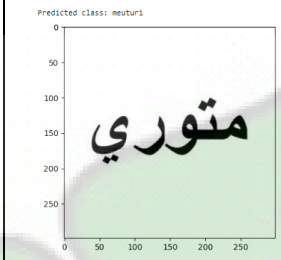
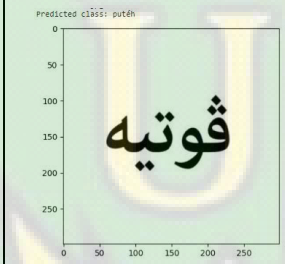
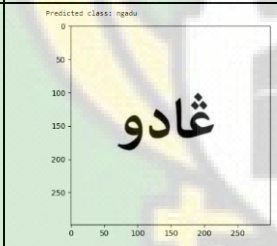
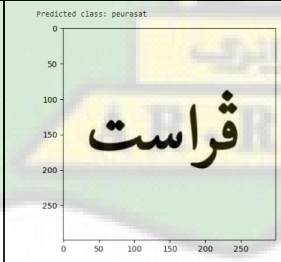
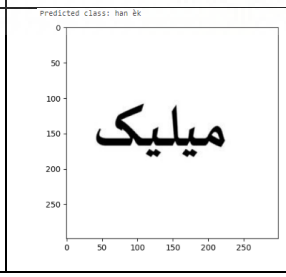
| | | | | |
|---|---|---|---|---|
| Agam |  | 48 | agam | Benar |
| Asèe |  | 28 | asèe | Benar |
| Ayah |  | 35 | ayah | Benar |
| Ato |  | 34 | ato | Benar |
| Badan |  | 54 | badan | Benar |
| Atas |  | 30 | aman | Salah |

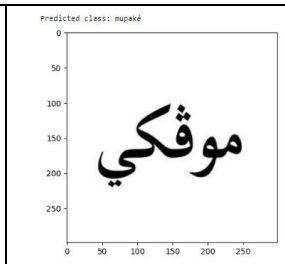
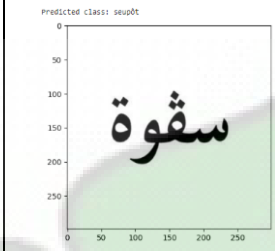
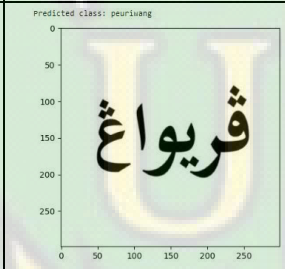
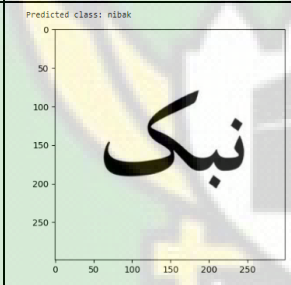
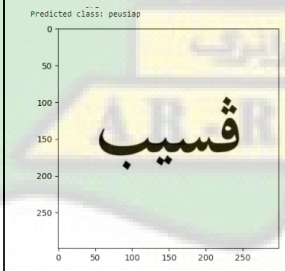
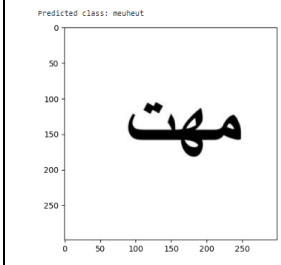
| | | | | |
|---|---|---|---|---|
| Baca |   | 42 | baca | Benar |
| Bajèe |   | 23 | timu | Salah |
| Bahlé |   | 50 | nemeujanji | Salah |
| Bahaya |   | 40 | bahaya | Benar |
| Balè |   | 47 | balè | Benar |
| Bantah |   | 33 | bantah | Benar |

| | | | | |
|---|---|---|---|---|
| Bangon |  | 34 | leumak | Salah |
| Di gata |  | 50 | reud | Salah |
| Didalam |  | 23 | didalam | Benar |
| Di pat |  | 38 | Di pat | Benar |
| Batèe |  | 27 | batèe | Benar |
| beungöh |  | 31 | beungöh | Benar |

| | | | | |
|---|---|---|---|---|
| binèh |  | 30 | tapinah | Salah |
| di lôn |  | 37 | di lôn | Benar |
| Binatang |  | 34 | binatang | Benar |
| Binoe |  | 29 | keunoe | Salah |
| peudati |  | 23 | peudati | Salah |
| Jikeuhendak |  | 28 | Jikeuhendak | Benar |

| | | | | |
|---|---|---|---|---|
| Seperti |  | 21 | tamöng | Salah |
| Pakayan |  | 25 | pakayan | Benar |
| Barang |  | 22 | barang | Benar |
| Tuha |  | 22 | tuha | Benar |
| Insan |  | 21 | insan | Benar |
| jén |  | 21 | jén | Benar |

| | | | | |
|---|---|---|---|---|
| Bicara |  | 21 | bicara | Benar |
| meuturi |  | 21 | meuturi | Benar |
| putéh |  | 21 | putéh | Benar |
| Ngadu |  | 23 | ngadu | Benar |
| peurasat |  | 21 | peurasat | Benar |
| milék |  | 21 | han èk | Salah |

| | | | | |
|---|---|---|---|---|
| mupaké |  | 22 | mupaké | Benar |
| seupôt |  | 38 | seupôt | Benar |
| Peuriwang |  | 39 | peuriwang | Benar |
| Nibak |  | 31 | Nibak | Benar |
| Peusiap |  | 31 | Peusiap | Benar |
| Meuheut |  | 33 | Meuheut | Benar |

## Lampiran 3 Source Code

### Import Library

```
# Dataset
from google.colab import drive
# Preprocessing
from keras.preprocessing.image import ImageDataGenerator
# Model
import tensorflow as tf
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications import Xception
from keras import layers
from keras.callbacks import ReduceLROnPlateau, EarlyStopping
import os
# Evaluate
from google.colab import files
from tensorflow.keras.preprocessing import image
from sklearn.metrics import precision_recall_fscore_support,
accuracy_score
import numpy as np
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report
import pandas as pd
import matplotlib.pyplot as plt
from keras import models
import cv2
from keras.models import model_from_json
from keras.models import load_model
import joblib
```

### Prepare Dataset

```
# connect to google drive
drive.mount('/content/gdrive')
#add the path general where the classes subpath are allocated
path = 'gdrive/MyDrive/TA/DataKata'
```

```
classes=[ "abang", "abèe", "acèh", "adat", "adil", "agam", "âib", "akal",
          "alamat", "aman", "ampôn", "aneuk", "angèn", "asèe", "atas",
          "ato", "ayah", "baca", "bacut", "badan", "bahaya", "bahlé",
          "bajèe", "bak", "bala", "balè", "ban duwa", "ban", "bangon",
          "bangsa", "bantah", "barang", "barat", "baroe", "batèe",
          "bekal", "bersalahan", "beuhë", "beungèh", "beungöh",
          "bicara","bijak","bijaksana", "binatang", "binèh", "binoe", "bit
          nyoe", "bit", "blahdéh", "blôh", "bos", "brat", "budiman",
          "buet", "buleuen", "bungong", "bunoe", "cabeueng", "celaka",
          "cicém", "cok", "cuba", "cuet", "dada", "dalam", "dengon",
          "deungö", "dheuen", "digata", "di likôt", "di lôn", "di pat",
          "di", "didalam", "dilée" "dônya", "droe", "duwa plôh", "duwa ",
          "euncien", "eungkôt","euntat", "gadöh", "gampong", "gaséh",
          "gata", "geucok", "geupèh", "geuriwang", "geutanyoe", "gigoe",
          "gleueng", "gob", "gulam", "gông", "guda", "guna", "gundah",
          "han èk", "han jid", "hana", "hantom", "harap", "haté", "hëi",
          "hékeumat", "hikayat", "ho", "hoka", "hudéb", "hukôm", "iem",
          "iman", "ingat", "inöng", "insan", "iték", "jadèh", "jak",
          "janji", "jarak", "jaroe","jaroem", "jaweueb", "jén", "jeumala",
          "jeumeurang", "jiba", "jiblôh", "jideungö", "jiduek", "jigaséh",
          "jih", "jijak", "jikalau", "jikalön", "jikap", "jikata", "duek",
          "jikeuheundak", "jikeumeung", "jikheun", "jilakée", "jimanoe",
          "jimarit", "jimè", "jimeubantah", "jimeuhoi", "jimeukeumas",
          "jimoe", "jiôh", "jipajôh", "jipeuduek", "jipeugah",
          "jipeurupa", "jiplueng","jipoh", "jiprèh", "jiriwang",
          "jisayang", "jiseuoet", "jitahë", "jitaki", "jitakôt",
          "jitanyong", "jiteumée", "jitrën", "judô", "ka meubah", "ka
          meulanggéh", "ka", "kah", "kalön", "kamoe", "karam", "karang",
          "kareuna", "kayèe", "ke yang", "kekèe", "keuheundak", "keterus",
          "keubit", "keudéh", "keudroe", "keugata",  "keujih", "keujinoe",
          "keujôk", "keumanusiya", "keumaté","keumöng", "keunan",
          "keunoe", "keunöng", "keupeue", "keutapoh",
          "keuteumèe","khabar", "kheun", "khianat", "kira", "kön", "köng",
          "krueng", "ku ba", "ku moe", "ku", "kulam", "kulét", "kuwasa",
          "ladôm", "lagèe", "lakoe", "lam", "laôt", "lë that", "lé",
          "leubèh", "leumah", "leumak", "lhèe", "lhôk", "limöng",
          "linteueng", "lom", "lôn", "luas", "lusa", "ma ba",
          "makanan","makin", "malang", "malèe", "mangat", "manok",
          "manusiya", "manyak", "maté", "mehoi", "melainkan", "meski",
          "meteumée", "meu'èn", "meubantah", "meubri", "meudéh",
          "meuheut", "meuhukôm", "meunan", "meupayah", "meurandéh",
          "meureubôt", "meureubôt-reubôt", "meurumpok", "meusapat",
          "meuseunoh", "meusôk-meusôk", "meuturi", "mie", "milék", "sudi",
          "mita", "moe", "mungkir","muda", "mupakat", "muka", "mukim",
```

```
        "mupaké", "musôh", "mustahil", "na sudi", "na", "nam", "marit",
        "nan", "nang","nanggroe", "narit", "nemeujanji", "neumat",
        "neupikir", "neutanyöng", "ngadu", "nggang", "ngob", "ngon",
        "nibak", "niet", "nyan ban", "nyan keuh", "nyata", "nyoe ma",
        "nyoe", "oh", "oh saré", "oleh", "padan", "paidah", "pajôh",
        "pakat", "pakayan", "pakon",
          "pakri", "pancuri", "panè", "panggang", "pantas",    "panté",
          "panyang", "paréksa", "patéh", "patôt", "payah", "perab",
          "perlahan", "peudati", "peue", "peugah", "peujelas",
          "peujeumerang", "peukawén", "peuleupas", "peunajôh",
          "peupaneuk", "peuplueng", "peurangeui", "peurasat",
          "peuriwang", "peurumoh", "peusiap", "phôm", "pi han", "pi
          that", "pikir", "pinah", "pineung",
          "piyôh", "plah", "plueng", "poh-poh", "polim", "pulo", "pura",
          "putéh", "rahsiya", "raja", "ramè", "raya", "rejang", "reud",
          "reuhueng", "reukueng", "reuôh", "rindu", "riyôh", "rumoh",
          "rupa", "s'èn", "sabab", "saban", "saboh", "sadum", "sah",
          "saho", "sajan", "salèh", "sama teungöh", "sama", "sangka",
          "sapo", "sayang", "segala", "sembah", "seperti", "serta",
          "setia",
          "seudang", "seuoet", "seupôt", "seutöt", "seuuem", "sibak",
          "si", "siat", "siblah", "sibungkoek", "sigra", "sijahtra",
          "silab", "sinan", "singoh", "sipanyang", "sira", "siuroe",
          "soe", "som", "sulét", "sunggôh", "supaya", "surôt", "syètan",
          "taba", "tacok", "tajak", "tajam", "tajôk", "takeubah",
          "taki", "takôt", "takue", "tameukawin", "tamöng", "tanda",
          "tangga", "tangké",
          "tanyan", "tanyöng", "taparéksa", "tapatéh",
          "tapeujeumeurang", "tapeusumpah", "tapinah", "taprèh",
          "tapulang", "taqdir", "tara", "teumpat", "terpikir",
          "geutueng", "teubiet", "teubuka", "teuga", "teuka", "teuma",
          "teumakôt", "teumoe", "teungku", "teungöh", "teuôt", "teupat",
          "teutap", "teutapi", "that", "tim", "timu", "tingkue",
          "tipèe", "töh", "tréb", "trën", "tueng","tuha", "tuhan",
          "tuleueng", "tumbon", "tuwan", "u binèh", "u blang", "ubak",
          "udeueng", "uké", "ulama", "jroh", "uleue", "ulôn", "umu",
          "upah", "ureueng", "wahé ", "yang na", "yang", "u teungöh",
          "yup"]

num_classes = len(classes)
batch_size = 16

#**No Augmentation on the Test set Images**
datagen = ImageDataGenerator(rescale=1./255,
```

```
validation_split=0.2) #9:1
#loading the images to training set
train_gen= datagen.flow_from_directory(directory=path,
                                       target_size=(299,299),
                                       class_mode='categorical',
                                       subset='training',
                                       shuffle=True,
                                       batch_size=batch_size,
                                       color_mode="rgb")
#loading the images to test set
test_gen = datagen.flow_from_directory(directory=path,
                                       target_size=(299, 299),
                                       class_mode='categorical',
                                       subset='validation',
                                       shuffle=False,
                                       batch_size=batch_size,
                                       color_mode="rgb")
```

## Model

```
learning_rate = 1e-5
epochs = 90
Xception = tf.keras.applications.Xception(input_shape = (299, 299, 3),
                                          weights = 'imagenet',
                                          include_top = False)
                                          Xception.trainable = True

model = tf.keras.Sequential([
        Xception,
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(475, activation = "softmax")],
        name = "Xception_Categorical_Classification")
        model.compile(optimizer=Adam(learning_rate=learning_rate),
        loss = "categorical_crossentropy",
        metrics = ["accuracy"])

#check point callback
checkpoint_path = "training/cp.ckpt"
checkpoint_dir  = os.path.dirname(checkpoint_path)
cp_callback =tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
         save_weights_only=True, verbose=1)
```

### Training dan Testing

```
#fit model
history = model.fit(train_gen,
                    validation_data=test_gen,
                    steps_per_epoch=100,
                    batch_size=batch_size,
                    epochs=epochs,
                    callbacks=[cp_callback])

# Evaluate model
model.evaluate(test_gen)
```

### Menyimpan Model dan Menambah Epoch

```
from keras.models import model_from_json
from keras.models import load_model
import joblib
from google.colab import drive  # Import drive from colab

# Connect to Google Drive
drive.mount('/content/gdrive')

# Path where the files will be saved in Google Drive
path = '/content/gdrive/MyDrive/TA/Xception_Model/'

# Simpan arsitektur model ke dalam file JSON
model_json = model.to_json()
with open(path + "model_architecture.json", "w") as json_file:
    json_file.write(model_json)

# Simpan bobot model ke dalam file HDF5
model.save_weights(path + "model_weights.h5")

# Simpan konfigurasi optimizer
optimizer_config = model.optimizer.get_config()
joblib.dump(optimizer_config, path + 'optimizer_config.joblib')

# Simpan history ke dalam file Joblib
joblib.dump(history.history, path + 'training_history.joblib')
```

```
# Save entire model (architecture, configuration, and weights)
model.save('/content/gdrive/MyDrive/TA/Xception_Model/entire_model.h5')
```

### Load File Model (Joblib)

```
drive.mount('/content/gdrive')
# Load entire model
loaded_model =
load_model('/content/gdrive/MyDrive/TA/Xception_Model2/entire_model.h5')
```

### Output

```
import matplotlib.pyplot as plt
%matplotlib inline

acc=history.history['accuracy']
val_acc=history.history['val_accuracy']
loss=history.history['loss']
val_loss=history.history['val_loss']
epochs=range(1,len(acc)+1)
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

### Confusion Matrix

```
# confusion matrix
# transform the predictions into array such as [0,0,1,2...]
```

```
y_pred = model.predict(test_gen)
predictions = np.array(list(map(lambda x: np.argmax(x), y_pred)))
#Retrieve the True classes of the test set
y_true=test_gen.classes

classes = list(test_gen.class_indices.keys())
# Build Confusion Matrix
CMatrix = pd.DataFrame(confusion_matrix(y_true, predictions),
columns=classes, index =classes)
plt.figure(figsize=(100, 100))
ax = sns.heatmap(CMatrix, annot = True, fmt = 'g' ,vmin = 0, vmax =
250,cmap = 'Blues')
ax.set_xlabel('Predicted',fontsize = 14,weight = 'bold')
ax.set_xticklabels(ax.get_xticklabels(),rotation =90);
ax.set_ylabel('Actual',fontsize = 14,weight = 'bold')
ax.set_yticklabels(ax.get_yticklabels(),rotation =0);
ax.set_title('Confusion Matrix - Test Set',fontsize = 16,weight = 'bold',
pad=20);
```

### Recall, Precision, F1-Score dan Accuracy

```
y_true=test_gen.classes
# Calculate Accuracy Result
acc = accuracy_score(y_true, predictions)


# Precision, Recall, and F-Score (For the whole dataset)
results_all = precision_recall_fscore_support(y_true, predictions,
average='macro',zero_division = 1)

# Precision, Recall, and F-Score (For each Class)
results_class = precision_recall_fscore_support(y_true, predictions,
average=None,zero_division = 1)

# Organize the Results into a Dataframe
metric_columns = ['Precision', 'Recall', 'F-Score', 'Support']
all_df = pd.DataFrame(list(results_class)).T
all_df.columns = metric_columns
overall_df = pd.DataFrame(list(results_all)).T
overall_df.columns = metric_columns

print('**Overall Results**')
```

```
print('Accuracy Result: %.2f%%' % (acc * 100))  # Accuracy of the whole
Dataset
print('Precision Result: %.2f%%' % (overall_df.iloc[0]['Precision'] *
100))  # Precision of the whole Dataset
print('Recall Result: %.2f%%' % (overall_df.iloc[0]['Recall'] * 100))  #
Recall of the whole Dataset
print('F-Score Result: %.2f%%' % (overall_df.iloc[0]['F-Score'] *
100))  # FScore of the whole Dataset
```

## Running

```
# test with image
50
# Load and predict the uploaded images
%matplotlib inline
uploaded = files.upload()

for fn in uploaded.keys():
    # Predicting images
    path = fn
    new_img = image.load_img(path, target_size=(299, 299))
    img = image.img_to_array(new_img)
    img = img / 255
    img = np.expand_dims(img, axis=0)
    prediction = model.predict(img)
    prediction_class = np.argmax(prediction, axis=1)
    confidence = np.max(prediction) * 100  # Get the confidence in
percentage

    predicted_class = classes[prediction_class[0]]

    # Display predicted class, confidence, and image
    print(f"Predicted class: {predicted_class}")
    plt.imshow(new_img)
    plt.show()
```