2 LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Aksara Sunda

Aksara Sunda merupakan hasil karya ortografi masyarakat Sunda melalui perjalanan sejarahnya sejak sekitar abad 5 M yang lalu hingga saat ini (Maliki & Febriansyah, 2023). Sebagai salah satu kebudayaan yang telah ada selama lebih dari 16 abad, terdapat banyak sekali kebudayaan Sunda berupa benda-benda bertulis seperti prasasti, piagam, dan naskah kuno. Hal ini menunjukkan adanya tradisi tulis-menulis di kalangan masyarakat Sunda sejak dulu. Oleh karena itu, sangat penting untuk memelihara kebudayaan karena hal tersebut mencerminkan identitas dan kebanggaan suku bangsa yang memiliki kebudayaan tersebut. Menurut (Amalia et al., 2020). Aksara Sunda dapat dibagi menjadi beberapa bagian, seperti Aksara swara (vokal), Aksara ngalagena (konsonan), Aksara khusus, tanda vokalisasi (rarangken), dan angka (Baidillah et al., 2008). Aksara ngalagena dan Aksara swara akan menjadi objek dari penelitian yang akan dilakukan.

2.1.1.1 Aksara Ngalagena

Aksara Ngalagena merupakan sistem tulisan yang memiliki kemampuan untuk merepresentasikan bunyi fonem konsonan dalam bentuk kata atau suku kata (Ismawan et al., 2020). Sebelumnya Aksara ngalagena dalam sistem tata tulis Aksara Sunda Kuno berjumlah 18 buah. Namun, sebagai upaya memenuhi fungsi Aksara Sunda sebagai sebuah bahasa yang terus berkembang, maka para pakar di bidang paleografi Sunda dan pihak birokrat (Kanwil Dikbud Jabar) serta tokoh masyarakat sepakat untuk memunculkan 7 lambang Aksara ke dalam sistem tata tulis Aksara Sunda Baku akibat terjadinya proses serapan unsur kosa kata asing (Darsa et al., 2007). Terdapat 18 Aksara Ngalagena (konsonan) yaitu ba, ca, da, ga, ha, ja, ka, la, ma, na, nga, nya, pa, ra, sa, ta, wa, dan ya. Contoh Aksara Sunda ngalagena bisa di lihat pada Gambar 2.1.

Gambar 2.1 Aksara Ngalagena

2.1.1.2 Aksara Rarangken

Aksara Rarangken merupakan pelengkap dan pendamping dari komponen Aksara Ngalagena. Sebab, seluruh huruf dalam komponen Aksara Ngalagena hanya diikuti oleh huruf a. Sedangkan, terdapat berbagai rangkaian kata dan kalimat yang diikuti dengan huruf vokal lainnya. Berdasarkan letak penulisannya, rarangkén dapat dibagi menjadi tiga yaitu rarangkén di atas huruf, rarangkén di bawah huruf, dan rarangkén sejajar huruf. Contoh Aksara Sunda ngalagena bisa di lihat pada Gambar 2.2.

a) Rarangkén di atas huruf
   - Panghulu: mengubah a menjadi i (ka menjadi ki
   - Pamepet: mengubah a menjadi e (ka menjadi ke)
   - Paneuleung: mengubah a menjadi eu (ka menjadi keu)
   - Panglayar: menambah 'r' di akhir suku kata (ka menjadi kar)
   - Panyecek: menambah 'ng' di akhir suku kata (ka menjadi kang)

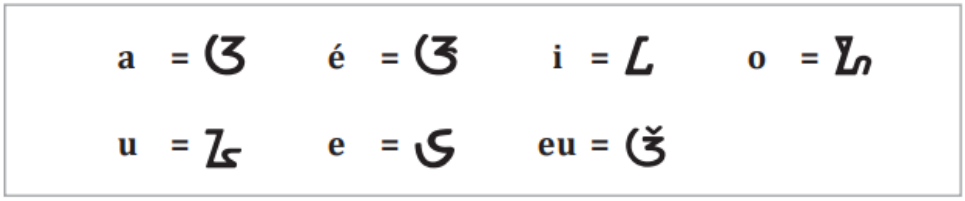
b) Rarangkén di bawah huruf
   - Panyuku: mengubah a menjadi u (ka menjadi ku)
   - Panyakra: menambah 'r' di tengah suku kata (ka menjadi kra)
   - Panyiku: menambah 'l' di tengah suku kata (ka menjadi kla

c) Rarangkén sejajar huruf
   - Panéléng: mengubah a menjadi é (ka menjadi ké)
   - Panolong: mengubah a menjadi o (ka menjadi ko)
   - Pamingkal: menambah 'y' di tengah suku kata (ka menjadi kya)
   - Pangwisad: menambah 'h' di akhir suku kata (ka menjadi kah)
   - Patén atau Pamaéh: Memutus huruf 'a' dalam suku kata (ka menjadi k)

Gambar 2.2 Aksara Rarangken

2.1.1.3 Aksara Swara

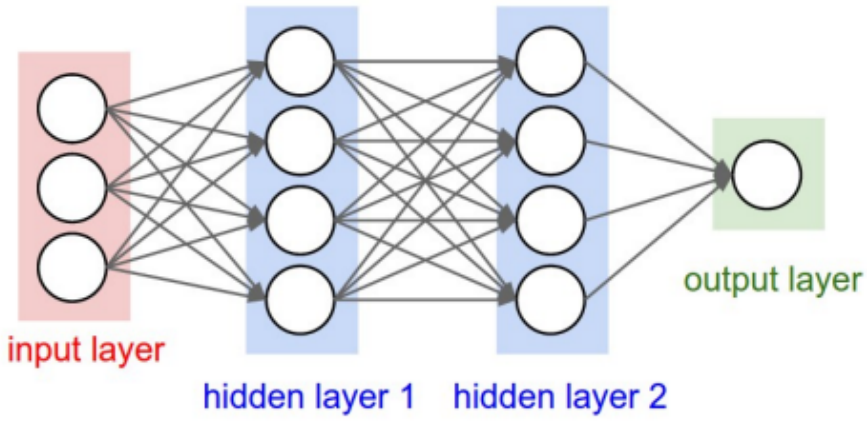
Aksara swara adalah Aksara yang secara silabis memiliki harkat bunyi vokal. Aksara swara berjumlah 7 huruf yaitu a, é, i, o, u, e, dan eu. Contoh Aksara swara bisa di lihat pada Gambar 2.3.



Gambar 2.3 Aksara Swara

2.1.2 Convolutional Neural Network

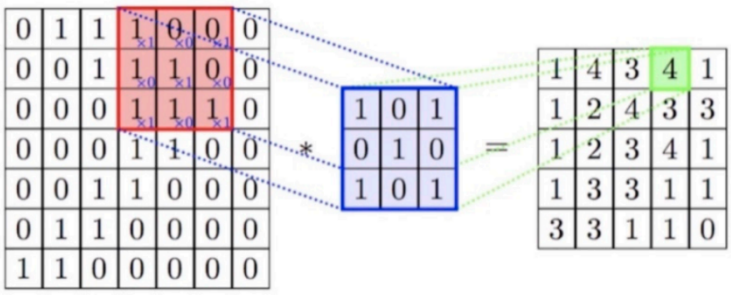
Klasifikasi gambar memiliki sejarah yang panjang dalam penelitiannya. Tujuan dari klasifikasi gambar (*supervised image*) adalah memetakan sebuah gambar *input*, X, ke sebuah kelas output, Y. *CNN* telah menunjukkan performa yang cukup baik dalam mengklasifikasikan gambar, hal ini dikarenakan karena lapisan-lapisan komputasinya yang berjumlah banyak yang bertujuan untuk mempelajari representasi yang lebih baik dari data gambar. Secara umum, *Convolutional Neural Network* terdiri dari sejumlah besar neuron yang saling terhubung yang memiliki bobot(W) dan bias (B). Neuron-neuron yang terdapat dalam arsitektur *CNN* umumnya dinamakan sebagai *layer*. Bobot mengontrol seberapa besar pengaruh setiap *input* terhadap neuron tersebut sedangkan bias digunakan untuk menyesuaikan tingkat aktivasi neuron terhadap berbagai situasi input agar neuron lebih fleksibel dan dapat mempelajari hubungan input dan output dengan lebih baik. *Convolutional Neural Network* terdiri dari satu input layer, beberapa *hidden layer* dan satu output layer (Bora et al., 2020). Seperti dapat dilihat di Gambar 2.4.



Gambar 2.4 Arsitektur Convolutional Neural Network

2.1.2.1 Convolutional Layer

Convolutional layer merupakan bagian inti dari arsitektur *CNN*. *Convolutional layer* akan mengekstrak fitur-fitur yang digunakan untuk klasifikasi gambar. Di dalam *Convolutional layer* inilah filter atau yang biasa disebut *kernels* digunakan. Pada saat ini, ukuran *kernel* menjadi sebuah parameter yang sangat penting karena menentukan ukuran filter seperti dapat dilihat di Gambar 2.5.



Gambar 2.5 Proses Convolutional Layer

2.1.2.2 Pooling Layer

Dalam *CNN*, penggunaan *convolutional* menerapkan filter ke gambar *input* untuk meng-*extract* fitur-fitur yang ada. Hal ini dilakukan untuk membangun *feature maps* yang dihasilkan oleh *output* dari *convolutional layer*. Namun, *feature maps* memiliki keterbatasan karena mencatat lokasi fitur-fitur dalam gambar *input* dengan presisi. Oleh karena itu, setiap pergerakan kecil yang terjadi di dalam gambar yang mempengaruhi fitur-fitur seperti pemangkasan ulang, rotasi, dll., akan menyebabkan perubahan dalam peta fitur (Salehi et al.,

2023). *Pooling layer* akan mereduksi dimensi dan melakukan ekstraksi fitur penting dari gambar *input* dalam *Convolutional Neural Network* (*CNN*), serta membantu dalam menciptakan representasi invarian untuk translasi *input*. Secara umum, ada dua jenis operasi *pooling* yang umum digunakan:

a) *Average Pooling*: *Average Pooling* digunakan ketika nilai yang diinginkan di dalam fe*a*t*u*re *maps* adalah nilai rata-rata. Proses *Average Pooling Layer* dapat dilihat melalui Gambar 2.6.

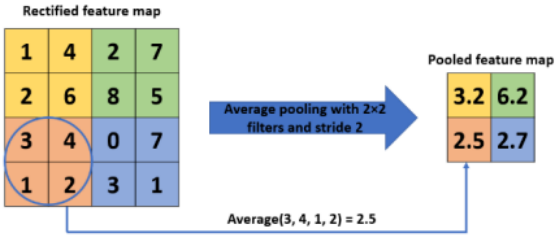
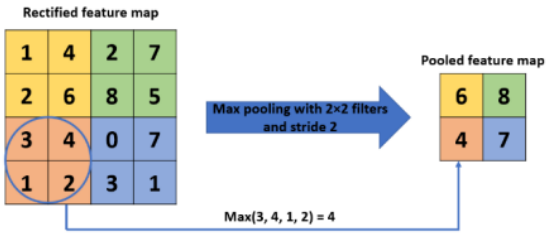


Fig. 1. Example of Average Pooling operation.

Gambar 2.6 *Average Pooling Layer*

b) *Max Pooling: Max Pooling* digunakan ketika nilai yang diinginkan di dalam fea*t*u*re maps* adalah nilai maksimum. Gambar di bawah ini akan mengilustrasikan cara kerja *average* dan *Max Pooling*. Proses *Max Pooling* layer dapat dilihat melalui Gambar 2.7.

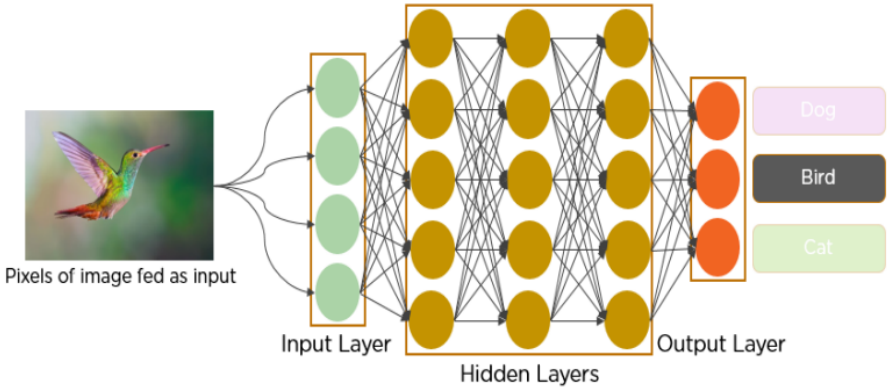


Gambar 2.7 *Max Pooling Layer*

2.1.2.3 Fully Connected Layer

Setelah melakukan ekstraksi fitur dan *pooling* layer, layer terakhir atau *fully connected* layer menghubungkan setiap jaringan ke satu *node* akhir. Sebelum memasuki *fully connected* layer, matriks akan mengalami proses *flatten* menjadi sebuah *vector*. *Flatten* layer merupakan jembatan yang menghubungkan layer-layer sebelumnya dengan *fully connected* layer. Layer ini menentukan klasifikasi dari input dengan menggunakan fungsi aktivasi seperti *softmax, ReLU,*

sigmoid dan tanh. Secara keseluruhan proses dari *Convolutional Neural Network* dapat dilihat dari Gambar 2.8.



Gambar 2.8 Proses *Convolutional Neural Network*

2.1.3 EfficientNetV2

CNN merupakan *neural network* yang memiliki 3 dimensi utama yaitu kedalaman *(Depth),* lebar *(Width),* dan resolusi gambar *(Image Resolution).* Ketiga dimensi tersebut dirancang untuk mengatasi masalah *overfitting*. *Overfitting* terjadi ketika sistem kehilangan kemampuan untuk mengenali pola umum dan hanya mampu mengenali data *training*. Hal ini sering disebabkan oleh arsitektur jaringan yang terlalu kompleks atau oleh terlalu banyak iterasi dalam sistem. Arsitektur CNN memerlukan penggunaan jumlah parameter yang besar di mana hal ini membuat memerlukan daya komputasi yang cukup besar sehingga membutuhkan waktu *training* yang lama. Oleh karena itu, penelitian Tan & Le (2019) mengusulkan sebuah arsitektur baru yaitu EfficientNet.

EfficientNet merupakan serangkaian arsitektur neural networks yang dirancang khusus untuk pengenalan gambar. EfficientNet merupakan model jaringan saraf konvolusi yang diusulkan oleh Mingxing Tan & Quoc V. Le dalam tulisan "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". Arsitektur ini dikembangkan dengan fokus pada efisiensi komputasi, yang memiliki tujuan untuk mencapai kinerja yang tinggi dengan menggunakan jumlah parameter yang lebih sedikit dan memerlukan sumber daya komputasi yang lebih sedikit. Keunggulan utama dari EfficientNet adalah scaling yang merata dalam tiga dimensi utama yaitu kedalaman (Depth), lebar (Width), dan resolusi gambar (Image Resolution). Dengan melakukan penyesuaian pada ketiga dimensi ini secara bersamaan, EfficientNet mampu mencapai kinerja yang lebih baik dibandingkan dengan ukuran 8.4x lebih kecil dan 6.1x lebih cepat dalam melakukan komputasi. Secara umum, model EfficientNet mampu mencapai akurasi yang lebih

tinggi dan memiliki efisiensi yang lebih baik daripada arsitektur CNN yang lain seperti AlexNet, ImageNet, GoogleNet, dan MobileNetV dimana EfficientNet mampu memperbaiki efektivitas model dengan melakukan pengurangan parameter dan FLOPS (Floating Point Operations Per Second).

2.1.3.1 EffiientNetV2B0

EfficientNetV2 merupakan sebuah arsitektur baru yang diusulkan dalam penelitian Tan & Le (2021). *EfficientNetV2* dirancang untuk melatih lebih cepat daripada model *EfficientNet*. Hal tersebut dicapai dengan mengatasi *training bottleneck* seperti *training* lambat dengan ukuran gambar besar dan konvolusi depthwise yang tidak efisien di lapisan awal. *EfficientNetV2* menggunakan non-uniform scaling strategy dan aturan penskalaan yang dimodifikasi untuk membatasi ukuran gambar maksimum, *EfficientNetV2* mengoptimalkan efisiensi parameter sambil mempertahankan kinerja tinggi. *EfficientNetV2* memperkenalkan operasi baru seperti Fused-MBConv dan training-aware neural architecture search yang digunakan untuk mengoptimalkan akurasi model, kecepatan training, dan ukuran parameter (Shabrina et al., 2023). *EfficientNetV2* mengungguli model *EfficientNet* dalam hal kecepatan training, efisiensi parameter, dan kinerja model secara keseluruhan. Ini mencapai hasil yang lebih kuat pada dataset seperti *ImageNet*, *CIFAR, Flowers*, dan *Cars*, melatih hingga 11 kali lebih cepat dan 6,8 kali lebih kecil dari model sebelumnya. Penelitian ini menggunakan *EfficientNetV2B0* karena arsitektur tersebut merupakan arsitektur yang memiliki tingkat kompleksitas yang paling rendah dan skala yang lebih kecil. Meskipun memiliki kompleksitas dan skala yang lebih kecil, *EfficientNetV2B0* dapat memberikan hasil yang memuaskan karena dirancang dengan mempertimbangkan efisiensi komputasi tanpa mengorbankan akurasi dalam tugas klasifikasi gambar (Dastour & Hassan, 2023).

2.1.4 YOLOv8

YOLOv8 (You Only Look Once version 8) adalah algoritma deteksi objek state-of-the-art yang dikembangkan oleh Ultralytics. Ini merupakan evolusi terbaru dari keluarga algoritma YOLO yang terkenal karena kecepatan dan akurasinya dalam deteksi objek real-time (Jocher et al., 2023). YOLOv8 menggunakan pendekatan berbasis regresi satu tahap, yang berarti ia dapat memprediksi bounding box dan kelas objek secara langsung dari piksel gambar dalam satu kali forward pass melalui jaringan saraf (Wang et al., 2023). Arsitektur YOLOv8 terdiri dari backbone, neck, dan head. Backbone-nya menggunakan CSPDarknet yang dimodifikasi untuk ekstraksi

fitur, neck menggunakan PANet (Path Aggregation Network) untuk fusi fitur, dan head menggunakan desain yang dioptimalkan untuk prediksi class, bounding box, dan objectness (Jocher et al., 2023). Beberapa peningkatan utama dalam YOLOv8 dibandingkan dengan versi sebelumnya adalah memiliki arsitektur backbone yang lebih efisien dan strategi augmentasi data yang lebih canggih, fungsi loss yang dioptimalkan, peningkatan kinerja pada objek kecil dan ukungan untuk berbagai tugas computer vision seperti deteksi, segmentasi, dan klasifikasi (Wang et al., 2023)

2.1.5 MobileNetV2

MobileNet adalah arsitektur jaringan saraf konvolusional (CNN) yang dirancang untuk aplikasi mobile dan embedded vision. Diperkenalkan oleh Howard et al. (2017), MobileNet menggunakan konvolusi separable depthwise untuk mengurangi jumlah parameter dan operasi komputasi sambil mempertahankan akurasi. Arsitektur ini terdiri dari lapisan konvolusi 3x3 depthwise yang diikuti oleh konvolusi pointwise 1x1, yang secara signifikan mengurangi kompleksitas model dibandingkan dengan CNN tradisional. MobileNetV2, diperkenalkan oleh Sandler et al. (2018), merupakan penyempurnaan dari arsitektur MobileNet original. Arsitektur ini menggabungkan dua ide kunci: konvolusi depthwise separable dari MobileNetV1 dan koneksi residual dari ResNet. MobileNetV2 memperkenalkan blok bottleneck terbalik dengan ekspansi linear, yang secara signifikan meningkatkan efisiensi model tanpa mengorbankan akurasi. Blok ini pertama-tama memperluas saluran input menggunakan konvolusi 1x1, kemudian melakukan konvolusi depthwise 3x3, dan akhirnya memproyeksikan kembali ke dimensi yang lebih kecil menggunakan konvolusi linear 1x1 lainnya. Meskipun ringan, MobileNetV2 mampu mempertahankan akurasi yang kompetitif pada berbagai tugas computer vision dan memiliki arsitektur yang fleksibel yang memungkinkan penyesuaian trade-off antara latensi dan akurasi melalui pengaturan width multiplier dan resolusi input.

2.1.6 Segmentation

Segmentasi adalah proses pengolahan gambar yang bertujuan memisahkan objek agar objek mudah dianalisis dalam rangka mengenali objek yang banyak melibatkan persepsi visual (Destyningtias, 2010). Segmentasi juga disebut sebagai proses yang membagi sebuah gambar menjadi sejumlah bagian atau objek. Segmentasi bukanlah proses tunggal yang dilakukan dalam pengolahan gambar digital, namun segmentasi merupakan proses yang penting dalam pengolahan gambar digital. Pada proses segmentasi, objek yang menjadi target akan diambil

untuk proses selanjutnya (Orisa & Hidayat, 2019). Segmentasi yang digunakan pada penelitian ini adalah segmentasi yang menggunakan bantuan library YOLO v8, dimana jaringan ini mendukung deteksi dan pelacakan objek serta beberapa tambahan seperti segmentasi instance, klasifikasi gambar, dan *key point detection* (Yue et al., 2023). YOLO v8 dipilih karena telah mengalami peningkatan dan optimasi yang signifikan, sehingga meningkatkan kinerja algoritma secara keseluruhan dibandingkan versi sebelumnya yaitu YOLO v5. Salah satu perbedaan utama adalah penggantian struktur C3 pada jaringan backbone YOLO v5 dengan struktur C2f pada YOLO v8. Perubahan ini memungkinkan YOLO v8 untuk mempertahankan karakteristiknya yang ringan sambil memperoleh lebih banyak informasi aliran gradien. Selain itu, YOLO v8 mendukung tugas tambahan seperti segmentasi instance, klasifikasi gambar, dan deteksi titik kunci, yang tidak tersedia pada YOLO v5. Bagian head pada YOLO v8 juga memiliki perbedaan mencolok karena implementasi struktur head terpisah yang banyak digunakan. Untuk perhitungan fungsi kerugian, YOLO v8 menggunakan strategi penugasan sampel positif *TaskAlignedAssigner* dan memperkenalkan *distribution focal loss*. Selama training, strategi untuk menonaktifkan augmentasi mosaik pada 10 epoch terakhir, seperti yang diperkenalkan pada YOLOX, juga diterapkan untuk meningkatkan presisi dalam proses augmentasi data. Semua peningkatan ini membuat YOLO v8 lebih unggul dibandingkan dengan YOLO v5 dalam hal kinerja dan fleksibilitas tugas.

2.1.7 Google Cloud Text-to-Speech

Google Cloud Text-to-Speech API adalah layanan canggih yang mengonversi teks menjadi ucapan alami menggunakan teknologi AI. Diperkenalkan sebagai bagian dari suite Google Cloud AI, API ini memanfaatkan model WaveNet yang dikembangkan oleh DeepMind, anak perusahaan Google (Van den Oord et al., 2016). WaveNet menggunakan *neural network* untuk menghasilkan bentuk gelombang audio mentah, menghasilkan suara yang lebih alami dibandingkan dengan sistem text-to-speech tradisional. Google Cloud Text-to-Speech menawarkan berbagai pilihan suara dalam banyak bahasa dan varian, memungkinkan pengembang untuk menyesuaikan output suara sesuai kebutuhan aplikasi mereka.

Salah satu keunggulan utama Google Cloud Text-to-Speech adalah kemampuannya untuk menyesuaikan berbagai aspek output suara, termasuk kecepatan bicara, pitch, dan volume. Menurut dokumentasi Google Cloud (2023), API ini juga mendukung notasi SSML (Speech Synthesis Markup Language), yang memungkinkan kontrol lebih detail atas pengucapan, intonasi, dan aspek prosodi lainnya dari ucapan yang dihasilkan. Fitur-fitur canggih

ini, dikombinasikan dengan integrasi mudah melalui SDK resmi Google untuk berbagai bahasa pemrograman, menjadikan Google Cloud Text-to-Speech pilihan yang sesuai untuk pengembang yang ingin menambahkan kemampuan text-to-speech berkualitas tinggi ke dalam aplikasi mereka.

2.1.8 Preprocessing Data

Data *preprocessing* adalah proses pengolahan data dari raw data hingga menjadi data yang siap digunakan. Dalam membentuk sebuah model yang digunakan untuk klasifikasi gambar, teknik *preprocessing* yang baik dan akurat berperan penting sebagai titik vital dalam meningkatkan performa sebuah model. Dengan menerapkan teknik *preprocessing* yang tepat, data yang digunakan sebagai *input* data akan dapat diterima dengan baik oleh model (Wihandika et al., 2024). *Preprocessing* juga mempengaruhi kualitas data yang digunakan dimana hal ini akan sangat mempengaruhi kemampuan model untuk belajar. Teknik *preprocessing* yang digunakan juga bergantung pada jenis data dan algoritma yang digunakan. Menurut Albahra et al. (2023) teknik *preprocessing* yang umum digunakan dalam klasifikasi gambar adalah *resize, cropping, normalization, rotate, flip, noise, color* dan lain-lain. Skripsi ini menggunakan teknik *preprocessing* berupa *resize, noise, rotate, add & multiply pixel*, dan normalisasi yang sesuai untuk pengenalan gambar.

2.1.8.1 Resize File Size

Merupakan langkah awal dalam preprocessing data untuk sistem pengenalan aksara Sunda dilakukan melalui proses resize file size. Ukuran gambar dalam dataset diseragamkan dan dioptimalkan melalui tahap ini. Proses ini diimplementasikan dengan menggunakan library Pillow (PIL) untuk manipulasi gambar (van Kemenade, 2023). Hal ini bertujuan agar penggunaan ruang penyimpanan dapat dioptimalkan dan efisiensi komputasi dalam tahapan preprocessing selanjutnya bisa lebih baik (Simard et al., 2003).

2.1.8.2 Grayscale

Teknik grayscale adalah proses pengolahan gambar yang mengubah gambar berwarna menjadi gambar dengan nuansa keabuan. Dalam proses ini, setiap piksel gambar dikonversi dari representasi warna RGB menjadi satu nilai intensitas keabuan dengan rentang nilai biasanya keabuan dari 0 (hitam) hingga 255 (putih) (Gonzalez & Woods, 2018). Teknik ini penting dalam preprocessing data karena dapat mengurangi kompleksitas gambar, menekankan fitur

struktural, meningkatkan kecepatan pemrosesan, dan potensial mengurangi noise. Dalam konteks pengenalan aksara Sunda, grayscale membantu model fokus pada bentuk dan garis khas aksara tanpa terdistraksi oleh variasi warna yang mungkin tidak relevan.

2.1.8.3 Resize

Perbedaan utama dengan *resize file size* adalah bahwa proses ini secara spesifik mengubah dimensi pixel gambar, bukan hanya ukuran filenya. Proses *resize* pada training dilakukan untuk mengatur ulang ukuran gambar agar dapat disesuaikan dengan ukuran yang dapat dibaca oleh model. Gambar-gambar yang digunakan akan memiliki ukuran yang konsisten sebelum dimasukkan ke dalam model melalui proses ini (Patel, 2023). Proses ini penting untuk memastikan bahwa semua gambar memiliki dimensi yang sama, yang merupakan persyaratan untuk banyak arsitektur model deep learning.

2.1.8.4 Shear

Shear adalah transformasi yang menggeser sebagian gambar secara horizontal atau vertikal, menciptakan efek miring (Jung, 2020). Teknik ini membantu model memahami objek yang mungkin terlihat terdistorsi atau miring dalam gambar nyata. Dengan menerapkan shear, model menjadi lebih fleksibel terhadap variasi bentuk objek yang mungkin terjadi akibat sudut pengambilan gambar yang berbeda-beda.

2.1.8.5 Rotate

Rotate merupakan teknik *preprocessing* yang akan mengubah bentuk objek dengan memutar piksel-piksel didalamnya sekitar titik yang diinginkan (Jung, 2020). Hal ini dapat digunakan untuk memberikan variasi pada sudut pandang ataupun orientasi objek dalam gambar. Hal ini akan membantu model dalam mengenali objek dari berbagai arah/sudut.

2.1.8.6 Scale

Scale mengubah ukuran gambar, baik memperbesar maupun memperkecil (Jung, 2020). Teknik ini membantu model mengenali objek dalam berbagai ukuran. Dengan menerapkan *scale*, model dapat lebih baik dalam mendeteksi objek yang mungkin muncul dalam berbagai skala pada data sebenarnya, meningkatkan fleksibilitas dan akurasi deteksi.

2.1.8.7 Translate_px

Translate_px menggeser posisi piksel-piksel dalam gambar secara horizontal atau vertikal (Jung, 2020). Teknik ini membantu model menjadi invariant terhadap posisi objek dalam gambar. Dengan menerapkan *translate_px*, model dapat lebih baik dalam mengenali objek yang mungkin muncul di berbagai posisi dalam frame, meningkatkan kemampuan generalisasi model.

2.1.8.8 Gaussian Blur

GaussianBlur menerapkan efek blur pada gambar menggunakan distribusi Gaussian (Jung, 2020). Teknik ini mensimulasikan variasi fokus atau ketajaman yang mungkin terjadi pada gambar nyata. Dengan menerapkan *GaussianBlur*, model menjadi lebih toleran terhadap gambar yang mungkin tidak selalu dalam kondisi fokus yang sempurna, meningkatkan fleksibilitas model terhadap variasi kualitas gambar.

2.1.8.9 Normalisasi

Normalisasi merupakan teknik mengubah rentang nilai piksel dalam gambar akan sesuai dengan rentang nilai yang diinginkan oleh model. Dalam konteks gambar, umumnya normalisasi seringkali dilakukan dengan membagi nilai piksel dengan nilai 255 dalam skala RGB, sehingga rentang nilai piksel menjadi antara 0 atau 1 (Darsa et al., 2007). Normalisasi .255 lebih sederhana dan cocok untuk data dengan rentang nilai tetap. Hal ini bertujuan untuk menjaga konsistensi dan stabilitas dalam pemrosesan data serta membantu model mempelajari gambar dengan lebih efisien.

2.1.9 Confusion Matrix

Confusion matrix merupakan sebuah tabel yang digunakan untuk mengevaluasi performa model klasifikasi pada data uji yang telah dikelompokkan. Ini menampilkan jumlah prediksi yang benar dan yang salah dibandingkan dengan label yang sebenarnya dalam bentuk tabel. Confusion matrix terdiri dari empat sel utama: True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Contoh confusion matrix dapat dilihat di Gambar 2.9.

		Actual values	
		+	-
Predicted values	+	True positive(TP)	False positive(FP)
	-	False negative(FN)	True negative (TN)

Gambar 2.9 Confusion Matrix

Sumber: (Valero-Carreras et al., 2023)

Menurut Wabang et al. (2022) nilai-nilai yang didapat melalui confusion matrix dapat digunakan untuk melakukan beberapa perhitungan dalam mengevaluasi model. Beberapa perhitungan yang dapat dilakukan adalah sebagai berikut:

2.1.9.1 Accuracy

Accuracy mengukur seberapa sering model klasifikasi memprediksi kelas yang benar secara keseluruhan. Ini dihitung sebagai jumlah prediksi yang benar (TP dan TN) dibagi oleh jumlah total prediksi(TP+FN+FP+TN). Persamaan accuracy dapat dilihat pada Persamaan (1) (Wabang et al., 2022).

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} = \frac{TP\ Total}{Total\ Dataset} \quad (2,1)$$

2.1.9.2 Precision

Precision mengukur seberapa akurat model dalam memprediksi kelas positif. Ini dihitung sebagai rasio True Positive (TP) dibagi dengan jumlah True Positive dan False Positive (TP + FP). Persamaan precision dapat dilihat pada Persamaan (2) (Wabang et al., 2022).

$$Precision = \frac{TP}{TP+FP} = \frac{TP}{Prediction\ Total} \quad (2,2)$$

2.1.9.3 Recall (Sensitivity)

Recall mengukur seberapa baik model dapat menangkap semua contoh yang benar-benar positif. Ini dihitung sebagai rasio True Positive (TP) dibagi dengan jumlah True Positive dan

False Negative (TP + FN). Persamaan recall (sensitivity) dapat dilihat pada Persamaan (3) (Wabang et al., 2022).

$$Recall = \frac{TP}{TP+FN} = \frac{TP}{Actual\ Total} \quad (2,3)$$

2.1.9.4 F1-Score

F1-score adalah rata-rata dari precision dan recall. F1-score memberikan keseimbangan antara kedua metrik dan berguna ketika ada ketidakseimbangan kelas. Persamaan f1-score dapat dilihat pada persamaan Persamaan (4) (Wabang et al., 2022).

$$F1Score = 2 \times \frac{Recall \times Precision}{Recall+Precison} \quad (2,4)$$

2.2 Tinjauan Studi

2.2.1 Pengenalan Pola Aksara Sunda dengan Metode *Convolutional Neural Network* (Kirana & Hikmayanti, 2020)

Masalah yang diangkat dalam penelitian ini adalah banyak remaja pada zaman ini menganggap Aksara Sunda sulit untuk dipelajari karena bentuknya unik dan cukup rumit. Metode yang digunakan dalam penelitian ini adalah Convolutional Neural Network. Data yang digunakan merupakan Aksara Ngalagena dan Aksara Swara yang didapat melalui tulisan tangan dan buku Baidilah sebanyak 2325 gambar yang terbagi kedalam 31 kelas. Hasil penelitian ini adalah pengujian pada akurasi gambar yang diambil dari kamera ponsel mendapatkan akurasi sebesar 87.1% dan yang diambil dari pemindai mendapatkan akurasi sebesar 85.5%. Kekurangan penelitian ini adalah kurangnya data yang digunakan, kualitas gambar yang diperoleh memiliki banyak noise. Penelitian ini tidak menggunakan teknik preprocessing untuk menunjang permasalahan tersebut. Akurasi yang didapat juga belum cukup baik dimana dibawah 90%.

Research Gap: Skripsi ini menggunakan Arsitektur EfficientNetV2B0 sedangkan penelitian ini menggunakan Convolutional Neural Network. EfficientNetV2B0 diusulkan karena dapat menyeimbangkan 3 dimensi secara bersamaan. Tidak hanya menyeimbangkan ketiga dimensi secara bersamaan, EfficientNetV2B0 juga memiliki nilai komputasi yang relatif kecil dengan performa yang baik. Skripsi ini juga mengaplikasikan model yang didapat kedalam aplikasi pembelajaran yang akan membantu pengguna dalam menulis Aksara Sunda.

2.2.2 Implementation of Convolutional Neural Network – Extreme Learning Machine for Handwriting Recognition of Sundanese Script (Maliki & Febriansyah, 2023).

Masalah yang diangkat dalam penelitian ini adalah hasil akurasi yang didapatkan di penelitian-penelitian sebelumnya masih belum bagus. Hal ini disebabkan oleh metode ekstraksi ciri dan klasifikasi yang kurang baik dan dataset yang kecil sehingga membuat metode yang digunakan kurang optimal. Metode yang digunakan dalam penelitian ini adalah Convolutional Neural Network (CNN) sebagai ekstraksi fitur dimana hasil yang diperoleh akan dijadikan masukkan untuk Extreme Learning Machine (ELM) algorithm. Data yang digunakan merupakan Aksara Ngalagena yang terbagi menjadi 1725 data latih dan 125 data uji dan dibuat oleh 20 responden yang terbagi kedalam 18 kelas. Kekurangan penelitian ini adalah kualitas gambar yang digunakan kurang baik dan tidak menggunakan teknik preprocessing untuk menunjang permasalahan tersebut. Akurasi yang didapat juga belum cukup baik dimana dibawah 90%.

Research Gap: Skripsi ini menggunakan Arsitektur EfficientNetV2B0 sedangkan penelitian ini menggunakan Convolutional Neural Network Sebagai Feature Extractor Dan Extreme Learning Machine Sebagai Classifier. EfficientNetV2B0 diusulkan karena dapat menyeimbangkan 3 dimensi secara bersamaan. Tidak hanya menyeimbangkan ketiga dimensi secara bersamaan, EfficientNetV2B0 juga memiliki nilai komputasi yang relatif kecil dengan performa yang baik. Skripsi ini juga mengaplikasikan model yang didapat kedalam aplikasi pembelajaran yang akan membantu pengguna dalam menulis Aksara Sunda.

2.2.3 Implementasi Algoritma *Convolutional Neural Network* pada Pengenalan Aksara Sunda Swara Panglayar (Ripera, 2024).

Masalah yang diangkat dalam penelitian ini adalah tergerusnya Aksara Sunda dengan bahasa-bahasa baru yang lebih modern. Metode yang digunakan dalam penelitian ini adalah Convolutional Neural Network. Data yang digunakan merupakan Aksara Swara yang terbagi menjadi dua bagian yaitu 546 data gambar yang didapat melalui tulisan tangan orang-orang disekitar dan 294 data yang didapat melalui internet (github.com/alifiacrm/Aksarasunda-dataset) yang terbagi kedalam 7 kelas. Hasil penelitian memiliki 4 pengujian yaitu:

a) Pengujian pada font kairaga dengan jumlah total 7 data uji didapatkan akurasi keberhasilan identifikasi tertinggi sebesar 57,14%.
b) Pengujian pada Unicode Aksara Sunda dengan jumlah total 7 data uji didapatkan akurasi keberhasilan identifikasi sebesar 14,28%.
c) Pengujian pada Tulisan Digital dengan jumlah total 168 data uji didapatkan akurasi keberhasilan identifikasi tertinggi sebesar 91,66%.

d) Pengujian pada Tulisan Tangan dengan jumlah total 168 data uji didapatkan akurasi keberhasilan identifikasi tertinggi sebesar 88,09%.
e) Hasil Pengujian dengan seluruh jumlah total 350 data uji didapatkan akurasi keberhasilan identifikasi tertinggi sebesar 86,85%.

Kelebihan penelitian ini adalah menggunakan augmentasi untuk menunjang jumlah dataset. Kekurangan penelitian ini adalah augmentasi yang digunakan belum sesuai dimana penulis melakukan horizontal flip sehingga tentunya tidak sesuai dengan tulisan Aksara yang ada (gambar terbalik). Akurasi yang didapat juga belum cukup baik dimana dibawah 90%.

Research Gap: Skripsi ini menggunakan Arsitektur EfficientNetV2B0 sedangkan penelitian ini menggunakan Convolutional Neural Network. EfficientNetV2B0 diusulkan karena dapat menyeimbangkan 3 dimensi secara bersamaan. Tidak hanya menyeimbangkan ketiga dimensi secara bersamaan, EfficientNetV2B0 juga memiliki nilai komputasi yang relatif kecil dengan performa yang baik. Skripsi ini juga mengaplikasikan model yang didapat kedalam aplikasi pembelajaran yang akan membantu pengguna dalam menulis Aksara Sunda.

2.2.4 Penggunaan Variasi Model pada Arsitektur EfficientNetV2 untuk Prediksi Sel Kanker (Sidik et al., 2023)

Masalah yang diangkat dalam penelitian ini adalah hasil akurasi yang didapatkan di penelitian-penelitian sebelumnya masih belum bagus. Metode yang digunakan dalam penelitian ini adalah seluruh model dari EfficientNetV2. Dataset berasal dari kaggle dan terdiri dari 25000 gambar yang terbagi kedalam 5 kelas. Hasil penelitian menyimpulkan bahwa EfficientNetV2B0 terbukti bisa mendeteksi gambar dengan kinerja dan efektivitas yang baik dengan akurasi sebesar 0.998 dan waktu komputasi 198 detik.

2.2.5 Metode Transfer Learning Pada Deep Convolutional Neural Network (DCNN) untuk Pengenalan Ekspresi Wajah (Alam, 2022)

Masalah yang diangkat dalam penelitian ini adalah ingin melakukan deteksi kebohongan dengan memperhatikan ekspresi wajah. Dataset yang digunakan merupakan dataset CK+ dan JAFFE. Data CK+ sebanyak 1256 gambar dan data JAFFE sebanyak 213 gambar yang terbagi kedalam 6 kelas. Metode yang digunakan dalam penelitian ini adalah 7 model dari EfficientNetV2. Hasil penelitian menyimpulkan bahwa EfficientNetV2B0 terbukti bisa mendeteksi wajah dengan kinerja dan efektivitas yang dengan akurasi testing mencapai 99.30% pada dataset CK+.

2.2.6 A Comparison of Deep Transfer Learning Methods for Land Use and Land Cover Classification (Dastour & Hassan, 2023).

Masalah yang diangkat dalam penelitian ini adalah membandingkan arsitektur-arsitektur untuk melakukan LULC Classification. Metode yang digunakan dalam penelitian ini adalah 39 arsitektur deep learning yaitu: DenseNet121, DenseNet169, DenseNet201, EfficientNetB0, EfficientNetB7, EfficientNetV2B0, EfficientNetV2B3, InceptionV3, MobileNetV1 and MobileNetV2, NASNetMobile, RegNetX002, RegNetX004, RegNetX006, RegNetX008, RegNetX032, ResNet50, ResNet101, ResNet152, ResNetRS50, ResNetRS101, ResNetRS152, ResNetRS200, ResNetRS270, ResNetRS350, ResNetRS420, ResNet50V2, ResNet101V2, ResNet152V2, VGG16 dan VGG19. Data yang digunakan berasal dari EuroSAT Dataset yang berjumlah 27000 gambar yang terbagi kedalam 10 kelas. Hasil penelitian ini menunjukkan bahwa ResNet50 memiliki f1-score tertinggi di beberapa kategori tertentu, sementara EfficientNetV2B0 menunjukkan kinerja yang baik secara keseluruhan. Meskipun ResNet152 memiliki kinerja yang baik, waktu training yang dibutuhkan merupakan 3 kali waktu training yang dibutuhkan oleh EfficientNetV2B0. Oleh karena itu EfficientNetV2B0 merupakan arsitektur yang memiliki kinerja yang baik dengan waktu training yang lebih efisien.

2.2.7 Perancangan Aplikasi Pembelajaran Aksara Sunda Berbasis Android (Chaidir et al., 2019)

Masalah yang diangkat dalam penelitian ini adalah media pembelajaran Aksara Sunda yang terbatas dan terkesan jenuh. Hasil penelitian ini adalah sebuah aplikasi pembelajaran Aksara Sunda. Kekurangan penelitian ini adalah pengguna tidak bisa mencoba menulis Aksara tersebut. Hasil pengujian yang digunakan juga kurang sesuai dimana hanya melakukan pengujian tombol-tombol dalam aplikasi dan tidak mengangkat seberapa berguna aplikasi tersebut dalam pembelajaran Aksara Sunda.

Research Gap: Skripsi ini akan membantu pengguna dalam menulis Aksara Sunda secara langsung dimana nantinya aplikasi akan mencocokkan tulisan tangan pengguna di kertas dengan Aksara Sunda yang dipilih menggunakan Arsitektur EfficientNetV2B0 sebagai model.

2.2.8 Pengembangan Aplikasi Pengenalan Aksara Sunda Berbasis Android Menggunakan Metode MDLC (Sujana, 2023)

Masalah yang diangkat dalam penelitian ini adalah media pembelajaran Aksara Sunda yang terbatas dan terkesan jenuh. Metode yang digunakan adalah Multimedia Development Life Cycle (MDLC) dan berbasis Android (brand). Hasil penelitian ini adalah sebuah aplikasi pembelajaran Aksara Sunda. Kekurangan penelitian ini adalah pengguna tidak bisa mencoba menulis Aksara tersebut. Hasil pengujian yang digunakan juga kurang sesuai dimana hanya menggunakan nilai rata-rata evaluasi dari pengguna. Oleh karena itu, skripsi ini tidak menunjukan apakah efektif dalam membantu pengguna belajar Aksara Sunda.

Research Gap: Skripsi ini akan membantu pengguna dalam menulis Aksara Sunda secara langsung dimana nantinya aplikasi akan mencocokan tulisan tangan pengguna di kertas dengan Aksara Sunda yang dipilih menggunakan Arsitektur EfficientNetV2B0 sebagai model.