Komputika: Jurnal Sistem Komputer
Volume xx, Nomor x, April/Oktober 20xx, hlm. 1 - 11
Terakreditasi Peringkat 3, SK No. 148/M/KPT/2020
DOI:

ISSN: 2252-9039 (print)
ISSN: 2655-3198 (online)

Model Deep Learning Dan Transfer Learning Klasifikasi Gambar Aksara OKU Timur

Rizky Ramadhan[1], Yesi Novaria Kunang[2*], Ilman Zuhri Yadi[3], Susan Dian Purnamasari[4]

[1,2,3,4)] Program Studi Sistem Informasi, Fakultas Sains Teknologi, Universitas Bina Darma
Jl. Jendral Ahmad Yani no. 3, Palembang, Indonesia 30264

email: rizkyramadhan301100@gmail.com

(Naskah masuk: *tgl. bulan tahun*; direvisi: *tgl. bulan tahun*; diterima untuk diterbitkan: *tgl. bulan tahun*)

ABSTRAK *– Aksara OKU Timur, suatu bentuk tulisan yang unik dan kuno merupakan bagian penting dari warisan budaya masyarakat OKU Timur Provinsi Sumatera Selatan di Indonesia. Saat ini masih sedikit upaya yang dilakukan untuk mengenali aksara ini secara luas, sehingga banyak orang kesulitan dalam mengenal dan mempelajarinya. Namun, dengan perubahan dinamika sosial dan teknologi, keberadaannya terancam tergerus oleh kurangnya pemahaman, keterbatasan aksesibilitas informasi terkait serta proses manual dalam mengklasifikasikan dan menganalisis aksara ini memakan waktu dan rentan terhadap kesalahan. Penelitian ini mengusulkan pengembangan model klasifikasi gambar aksara OKU Timur, yang bertujuan untuk mengembangkan model yang lebih efisien dan akurat untuk memproses dan menganalisis aksara tersebut agar dapat melestarikan budaya masyarakat OKU khususnya aksara OKU Timur. Dengan menggunakan metode Transfer Learning dan model yang digunakan merupakan model pra-latih yaitu ResNet26, ResNet18 dan MobileNetV2. Hasil yang diharapkan dari penelitian ini adalah sebuah model yang dapat mengklasifikasikan gambar aksara OKU Timur secara akurat dan menjadi sumber informasi bagi peneliti selanjutnya.*

Kata Kunci *– Aksara OKU Timur; Deep Learning; Transfer Learning; Klasifikasi Gambar.*

Deep Learning and Transfer Learning Models for Image Classification of OKU Timur Script

ABSTRACT *– The OKU Timur script, a unique and ancient form of writing, is an important part of the cultural heritage of the OKU Timur community in South Sumatra, Indonesia. Currently, there are still few efforts to widely recognize this script, making it challenging for many to identify and learn it. However, with changing social dynamics and technology, its existence is at risk due to a lack of understanding, limited accessibility to related information, and the time-consuming, error-prone manual process of classifying and analyzing the script. This research proposes the development of an image classification model for the OKU Timur script, aiming to create a more efficient and accurate model to process and analyze the script to preserve the culture of the OKU community, specifically the OKU Timur script. Using Transfer Learning methods, the models employed are pre-trained models, namely ResNet26, ResNet18, and MobileNetV2. The expected outcome of this research is a model capable of accurately classifying images of the OKU Timur script and serving as a source of information for future researchers.*

Keywords *– East OKU Script; Deep Learning; Transfer Learning; Image Classification.*

1. PENDAHULUAN

Indonesia merupakan negara yang memiliki banyak keanekaragaman suku, budaya dan bahasa dari berbagai daerah yang tersebar di Nusantara. Salah satu dari keanekaragaman tersebut adalah aksara OKU Timur. Aksara merupakan alat komunikasi, baik lisan maupun tertulis. Sebagai peninggalan masyarakat zaman lampau dari generasi ke generasi, aksara tidak bisa diabaikan keberadaannya [1], [2]. Aksara Oku Timur, sebuah sistem tulisan kuno dari Sumatra Selatan, Indonesia,

menjadi salah satu warisan budaya yang memperkaya kekayaan budaya Indonesia. Saat ini masih sedikit upaya yang dilakukan untuk mengenali aksara ini secara luas, sehingga banyak orang kesulitan dalam mengenal dan mempelajarinya. Dengan perubahan dinamika sosial dan teknologi, keberadaannya terancam tergerus oleh kurangnya pemahaman, keterbatasan aksesibilitas informasi terkait serta proses manual dalam mengklasifikasikan dan menganalisis aksara ini memakan waktu dan rentan terhadap kesalahan.

Aksara OKU Timur, salah satu keanekaragaman budaya merupakan bagian penting dari warisan budaya masyarakat OKU Timur provinsi Sumatera Selatan di Indonesia, dikarenakan aksara tersebut banyak terdapat pada naskah kuno, maka dari itu perlu dijaga dan melestarikannya. Salah satu cara untuk melestarikan aksara OKU Timur ialah dengan menggunakan teknologi, mengingat pada zaman sekarang ini hampir semua aspek kehidupan manusia berhubungan dengan teknologi [3]. Dengan adanya kemajuan teknologi saat ini bukanlah hal mustahil untuk mengembangkan kecerdasan buatan untuk dapat menyelesaikan perkerjaan manusia dengan lebih mudah, lebih akurat, dan membutuhkan waktu yang singkat. Salah satu kecerdasan buatan yang dapat memudahkan pekerjaan manusia yaitu *Deep Learning*.

Deep Learning merupakan subbidang dari kecerdasan buatan yang menggunakan jaringan saraf tiruan (*Neural Networks)* yang dalam (*Deep)* untuk melatih atau mengajarkan suatu tindakan yang dianggap normal bagi manusia [4], [5], [6], [7], [8]. *Convolutional Neural Network* merupakan salah satu algoritma *Deep Learning* yang dapat digunakan untuk mendeteksi dan mengenali sebuah objek pada sebuah citra digital [9], [10], [11], [12]. Dalam *Deep Learning* terdapat teknik *Transfer Learning*. *Transfer Learning* bertujuan untuk meningkatkan kinerja tugas dengan mentransfer pengetahuan yang dipelajari dari tugas sumber ke tugas target [13], [14], [15], [16].

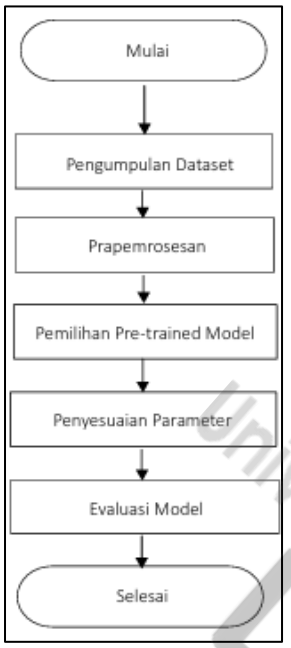
Terdapat beberapa penelitian yang menggunakan teknik *Transfer Learning* untuk mengklasifikasikan gambar, antara lain penelitian yang menggunakan *Convolutional Neural Network* dan *Transfer Learning* untuk mengklasifikasikan jenis rempah [17]. Kemudian penelitian yang menggunakan *Deep Learning* dan metode *Transfer Learning* untuk mengklasifikasikan jajanan tradisional Indonesia [18]. Dan penelitian yang menggunakan *Convolutional Neural Network* dan *Transfer Learning* untuk mengklasifikasikan jenis biji kopi [19].

Sejauh ini belum ada penelitian yang mengklasifikasi gambar aksara OKU Timur, namun terdapat penelitian yang mengklasifikasikan aksara varian Komering yang berbeda tetapi menggunakan model *Deep Learning* untuk membuat dan melatih model dari awal dan penelitian ini tidak menggunakan metode *Transfer Learning*, yang berjudul "*A New Deep Learning-Based Mobile Application for* Komering *Character Recognition*" [3]. Ketidaktersediaan penelitian terkait dengan penggunaan *Transfer Learning* dalam klasifikasi gambar aksara OKU Timur menjadi sebuah tantangan tersendiri untuk penulis mencari informasi yang signifikan dalam pengembangan model klasifikasi gambar ini.

Maka dari itu diperlukannya model yang dapat mengklasifikasikan aksara OKU Timur. Pengembangan model klasifikasi gambar menggunakan metode Transfer Learning ini perlu diadakan, untuk dapat mengembangkan model yang lebih efisien dan akurat untuk memproses dan mengklasifikasikan aksara tersebut sehingga dapat melestarikan budaya masyarakat OKU khususnya aksara OKU Timur. Berdasarkan permasalahan yang ada, penulis melakukan penelitian dengan judul "Model Deep Learning dan Transfer Learning Klasifikasi Gambar Aksara OKU Timur". Diharapkan melalui penelitian ini, dengan mengembangkan sebuah model klasifikasi gambar aksara OKU Timur secara akurat dapat memberikan kontribusi nyata dalam memperkuat upaya pelestarian dan penggunaan aksara OKU Timur dalam era digital.

2. METODE DAN BAHAN

Dalam penelitian ini, metode pengembangan yang digunakan adalah *Rapid Application Development* (RAD). Metode Rapid Application Development (RAD) adalah cara dalam mengembangkan perangkat lunak yang fokus pada kecepatan dan proses iteratif [20]. RAD dipilih karena tahapan-tahapannya yang terstruktur memungkinkan pengembangan perangkat lunak dilakukan dengan cepat melalui siklus singkat dan memberikan hasil dengan segera berkat pembagian tugas menjadi bagian-bagian kecil. Alasan utama penerapan metode ini adalah efektivitasnya dalam pengembangan aplikasi berskala kecil [21], [22], [23].



Gambar 1. *Flow chart*

1. Pengumpulan Dataset

Tahap ini mencakup proses dalam mengumpulkan dataset yang akan digunakan untuk melatih dan menguji model. Dalam hal ini dataset aksara OKU Timur didapatkan dari Ketua Adat setempat dan Ahli Aksara Oku Timur serta perwakilan anggota riset ISRG (*Intelligent System Research Group)*, setelah data aksara dikumpulkan terdapat 225 karakter aksara Oku timur, yang kemudian akan dicetak dan diberikan kepada para responden untuk diisi, jumlah responden didapat sebanyak 102 orang. Untuk beberapa proses seperti prapemrosesan, segmentasi, *cropping* dan *clustering* pada data mentah dilakukan oleh anggota lain sehingga membentuk dataset yang siap digunakan untuk melatih model, sebanyak 225 karakter dan 23.224 gambar.

2. Prapemrosesan

Tahap ini berbeda dengan prapemrosesan pada data mentah. Prapemrosesan dataset dilakukan kembali sebelum dataset digunakan untuk melatih model. Proses yang dilakukan yaitu mengunduh dan mengekstrak dataset, kemudian memproses gambar dan membagi dataset menjadi data pelatihan dan data uji agar siap digunakan dalam proses pelatihan model.

3. Pemilihan Pre-trained Model

Tahap ini merupakan pemilihan model pra-latih yang memiliki arsitektur atau fitur yang sesuai dengan tugas atau domain target yang diinginkan. Dengan pemilihan model yang tepat dapat mempengaruhi kinerja dan keberhasilan model secara keseluruhan. Pada penelitian kali ini model yang digunakan yaitu ResNet26 [24], ResNet18 [24] dan MobileNetV2 [25].

4. Penyesuaian Parameter

Setelah tahap pemilihan model, langkah ini melakukan penyetelan atau penyesuaian terhadap parameter model, seperti penggunaan *learning rate* yaitu 0.01, dan jumlah epo (*epoch*) yang digunakan sebanyak 100 *epoch*, agar sesuai dengan data set target dan mengoptimalkan kinerja model dan melatih model menggunakan *fine tuning*.

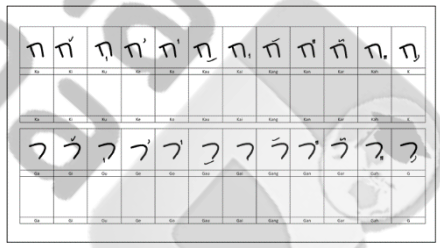
5. Evaluasi Model

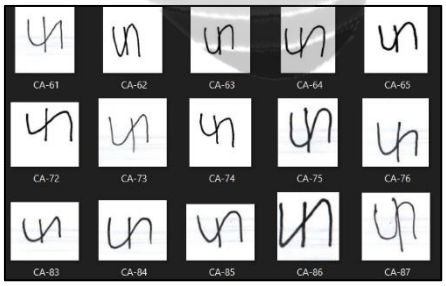
Tahap ini merupakan proses evaluasi kinerja model pada dataset validasi dan tes, lalu melakukan evaluasi dengan melihat hasil melalui grafik dan matrik pada pelatihan sehingga dapat mengetahui hasil pelatihan pada ketiga model yang dipilih yaitu ResNet26, ResNet18 dan MobileNetV2. Penyusuaian atau *tuning* tambahan jika diperlukan untuk meningkatkan kinerja model.

3. HASIL DAN PEMBAHASAN

1. Pengumpulan Dataset

Tahapan pengumpulan data mentah dilakukan dengan cara menyebar kuesioner yang terdiri dari 225 karakter Aksara OKU Timur, dan responden yang dikumpulkan berjumlah 102 orang.


Gambar 2. Contoh Kuesioner Aksara OKU Timur



Gambar 3. Contoh karakter aksara yang telah dipotong


Gambar 4. Contoh karakter aksara yang telah dikelompokkan

Kuesioner tersebut kemudian diolah menjadi dataset dengan cara di *crop* per karakter dan

kemudian dikelompokkan berdasarkan nama huruf masing-masing.

2. Prapemrosesan

Prapemrosesan merupakan Langkah awal yang penting dalam setiap proyek *machine learning,* termasuk *deep learning dan transfer learning*. Jumlah data gambar yang diproses yaitu sebanyak 23.224 gambar yang kemudian dibagi menjadi 225 kelas. Proses ini dilakukan agar data siap digunakan oleh model. Ada beberapa tahapan yang dilakukan pada saat prapemrosesan diantaranya :

1) Pengunduhan dan ekstraksi dataset

```
!wget 'https://www.dropbox.com/scl/fi/ukxno4a1uexx5j80k2iav/oku_timur.zip
import zipfile
from pathlib import Path


zip_file_path = Path("/content/oku_timur.zip")
extracted_folder_path = Path("oku_timur")

with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
    zip_ref.extractall(extracted_folder_path)


path = extracted_folder_path


print([x for x in path.iterdir()])
```

Gambar 5. Kode untuk mengunduh dan mengekstrak dataset

Penelitian ini dimulai dengan mengunduh file data dari URL eksternal menggunakan perintah 'wget'. Perintah ini memungkinkan sistem untuk melakukan pengunduhan file secara otomatis dari internet dan menyimpannya di lokasi yang telah ditentukan, dalam hal ini file yang diunduh adalah oku_timur.zip.

Selanjutnya, dilakukan proses ekstraksi file menggunakan modul zipfile di dalam Python. Modul ini digunakan untuk membaca dan mengekstrak konten dari file ZIP. Proses ekstraksi dilakukan dengan membuka file yang telah diunduh, kemudian seluruh isi file tersebut diekstrak ke dalam direktori yang telah ditentukan (/home/oku_timur). Setelah itu, untuk memastikan keberhasilan ekstraksi, program akan menampilkan daftar file yang diekstrak dari folder tersebut.

2) Mengatur ukuran, *path* dan menghitung jumlah gambar

```
[ ]  IMG_SIZE = (80,80)
     data_dir = "/home/oku_timur"

[ ]  data_dir = pathlib.Path(data_dir)
     Image_count = len(list(data_dir.glob('*/*.*')))
     Image_count

     23224
```

Gambar 6. Kode untuk mengatur dan menghitung jumlah gambar

Proses ini bertujuan untuk menghitung jumlah gambar dalam sebuah direktori yang nantinya akan digunakan dalam pemrosesan atau pelatihan model pembelajaran mesin. Pada tahap awal, ditentukan ukuran gambar yang diinginkan, yaitu 80x80 piksel. Ini bertujuan untuk memastikan bahwa semua gambar memiliki resolusi yang seragam, yang sering kali menjadi syarat dalam pemrosesan gambar berbasis jaringan saraf tiruan.

3) Persiapan *dataframe* dan pembagian data

```
paths=[]
labels=[]
for dirname, _, filenames in os.walk('/home/oku_timur'):
    for filename in filenames:
        paths+=[os.path.join(dirname, filename)]
        labels+=[dirname.split('/')[-1]]
print(paths[0:3])

data_df=pd.DataFrame(columns=['path','label'])
data_df['path']=paths
data_df['label']=labels
display(data_df)
display(data_df['label'].value_counts())

n=len(data_df)
N=list(range(n))
random.seed(2021)
random.shuffle(N)
train_df=data_df.iloc[N[0:(n//5)*4]]
test_df=data_df.iloc[N[(n//5)*4:]]
```

Gambar 7. Kode untuk menyiapkan *DataFrame* dan pembagian data

Pada tahap selanjutnya, jalur file gambar beserta labelnya disusun ke dalam sebuah *dataframe* menggunakan pustaka pandas. *Dataframe* ini berfungsi sebagai struktur data yang efisien untuk memetakan jalur gambar dengan label kategorinya, memungkinkan visualisasi distribusi data serta memudahkan manipulasi dan analisis lebih lanjut. Distribusi jumlah gambar per kategori kemudian dianalisis untuk memastikan bahwa data terdistribusi secara merata atau untuk memahami ketidakseimbangan yang mungkin terjadi dalam dataset.

Tahap penting berikutnya adalah membagi dataset menjadi set pelatihan dan pengujian. Untuk menghindari bias selama pelatihan, data diacak terlebih dahulu menggunakan metode *random shuffle*. Data kemudian dibagi dengan rasio 80:20, di mana 80% digunakan untuk melatih model dan 20% sisanya digunakan untuk menguji performa model. Pembagian ini memastikan bahwa model memiliki cukup data untuk dilatih, sementara subset pengujian tetap independen sehingga dapat mengevaluasi generalisasi model.

4) Mengolah data dan augmentasi gambar

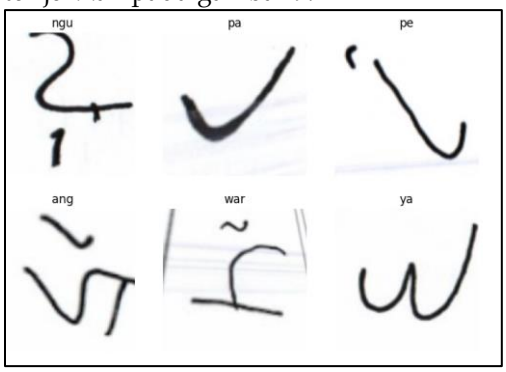
```
dls = ImageDataLoaders.from_folder(trn_path,
                                   valid_pct=0.2,
                                   seed=42,
                                   item_tfms=Resize(480, method='squish'),
                                   batch_tfms=aug_transforms(size=128, min_scale=0.75))
dls.show_batch(max_n=6)
```

Gambar 8. Kode untuk transformasi gambar

Pada proses ini, data gambar dimuat menggunakan 'ImageDataLoaders.from_folder()' salah satu fungsi dari pustaka fastai, yang memudahkan pemuatan data dari folder yang berisi subdirektori gambar berdasarkan kategori (label). Dataset dibagi menjadi dua subset, yaitu data latih dan data validasi, dengan perbandingan 80:20 menggunakan parameter valid_pct=0.2. Ini adalah rasio yang umum digunakan dalam pelatihan model pembelajaran mesin, di mana model dilatih pada 80% data dan divalidasi pada 20% sisanya untuk mengevaluasi kinerjanya.

Ukuran gambar diseragamkan menjadi 480x480 piksel menggunakan metode Resize dengan metode "squish". Metode ini penting untuk menstandarkan dimensi input gambar, sehingga model menerima gambar dengan ukuran yang seragam. Meskipun metode "squish" dapat menyebabkan distorsi gambar karena perubahan rasio aspek, ini sering kali digunakan untuk memastikan semua gambar memiliki dimensi yang sama.

Augmentasi data diterapkan pada batch gambar menggunakan 'aug_transforms()', metode augmentasi data ini mengurangi *overfitting* dengan menghasilkan data tambahan dengan mengubah data asli untuk menghasilkan variasi baru dari data tersebut [26]. Metode ini dapat memperbaiki kemampuan model dalam menggeneralisasi data yang belum pernah ditemui sebelumnya [27]. Seperti yang ditunjukkan pada gambar 9.



Gambar 9. Augmentasi data gambar

3. Pemilihan Pre-trained Model

Pemilihan Pre-trained model adalah Langkah penting dalam Pembangunan model machine learning, terutama dalam transfer learning [28]. Dimana model yang telah dilatih pada dataset besar dan beragam digunakan sebagai dasar untuk pelatihan lebih lanjut pada dataset spesifik.

Dalam Penelitian ini penulis dapat mencari *list* model dengan menggunakan modul timm, timm adalah pustaka Python untu Pytorch yang menyediakan berbagai model arsitektur *deep learning* yang telah dilatih sebelumnya khusus untuk tugas-tugas pengolahan citra [29]. Model yang dipilih yaitu, ResNet26, ResNet18, MobileNetV2. ResNet26 dan ResNet18 merupakan Arsitektur yang mendalam dan kemampuanya untuk menangani pelatihan model yang sangat dalam dengan *residual connections*. MobileNetV2 merupakan arsitektur ringan yang dioptimalkan untuk perangkat mobile dan *embedded* dengan keterbatasan sumber daya.

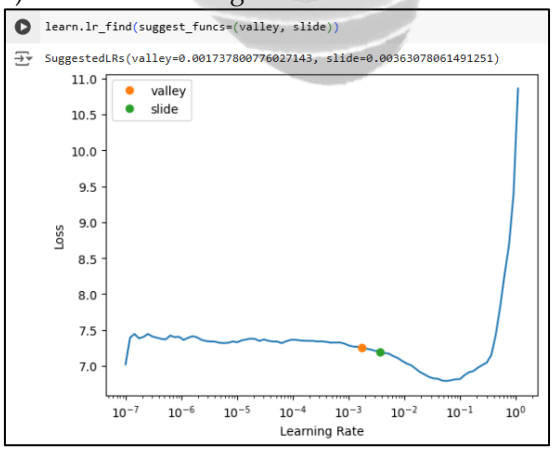
4. Penyesuaian Parameter

Proses penyesuaian parameter atau fine-tuning merupakan proses melatih Kembali model yang sudah dilatih sebelumnya (Pre-trained Model) dengan dataset yang baru. Langkah-langkah dalam penyesuaian parameter sebagai berikut :

1) Menginisialisasi model

Inisialisasi model adalah proses awal dalam pelatihan model *machine learning* di mana parameter model (seperti bobot dan bias dalam jaringan saraf) diberi nilai awal sebelum proses pelatihan dimulai. Pustaka Fastai adalah sebuah pustaka *open-source* untuk *machine learning* dan *deep learning* yang dibangun di atas PyTorch [30]. Dalam hal ini peneliti menggunakan pustaka Fastai untuk menginisialisasi model pra-latih yaitu ResNet26, ResNet18 dan MobileNetv2, sehingga parameter diambil dari model tersebut dan tidak perlu diinisialisasi ulang, peneliti juga menggunakan metrik 'Accuracy' 'Error_rate' untuk evaluasi dan mengoptimalkan penggunaan memori serta kecepatan dengan mengkonversi model ke format presisi float16.

2) Memilih learning rate



Gambar 10. Kode untuk menampilkan learning rate

Sebelum memulai pelatihan, metode *learning rate finder* (lr_find) digunakan untuk menemukan nilai *learning rate* yang optimal. *Learning rate* adalah parameter yang sangat penting dalam pelatihan model, karena mempengaruhi kecepatan dan stabilitas proses pelatihan. Dengan menggunakan fungsi *lr_find*, peneliti dapat mengidentifikasi nilai yang tepat sehingga

pelatihan dapat dilakukan secara efisien tanpa mengalami masalah seperti *overfitting* atau *underfitting*.

3) Fine-tuning model

```
[ ] learn.fine_tune(100,0.01)
```

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	2.736781	1.874855	0.436000	0.564000	00:46
epoch	train_loss	valid_loss	accuracy	error_rate	time
0	1.739345	1.077095	0.671762	0.328238	00:53
1	1.421772	0.834100	0.734309	0.265691	00:52
2	1.178811	0.703074	0.774572	0.225428	00:53
3	1.034921	0.599893	0.806761	0.193239	00:53
4	0.916191	0.540431	0.824631	0.175369	00:53
5	0.864208	0.481653	0.840241	0.159759	00:53
6	0.776037	0.443025	0.853913	0.146087	00:53
7	0.762672	0.420984	0.863494	0.136506	00:53
8	0.711715	0.393561	0.870492	0.129508	00:53
9	0.703504	0.370708	0.882872	0.117128	00:53
10	0.682134	0.351917	0.885779	0.114221	00:53

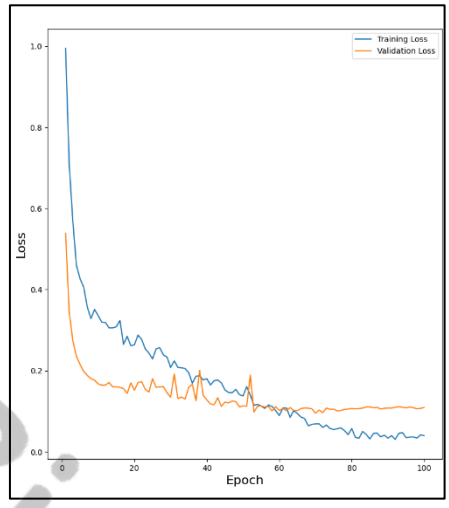
Gambar 11. Kode untuk melatih model

Setelah nilai *learning rate* yang optimal ditemukan, model dilatih selama 100 epoch menggunakan fungsi 'learn_fine_tune' yang merupakan fitur Fastai yang digunakan untuk melatih Kembali model yang telah dipra-latih dengan data baru. 'fine_tune' menjalankan dua tahap sebagai berikut:

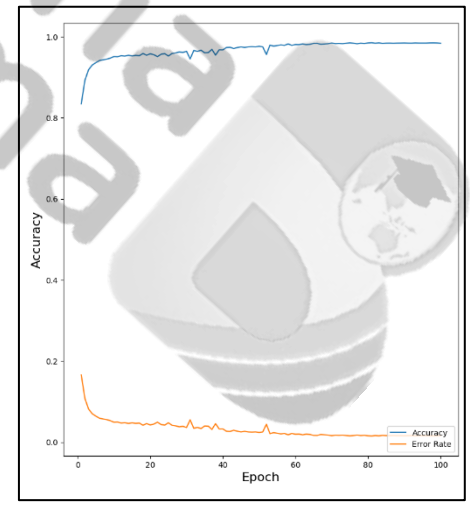
a. Tahap pertama : Bagian akhir *(layer* terakhir) dari model dilatih selama satu atau lebih *epoch* dengan *learning rate* tertentu. Dengan membekukan (*freeze*) semua *layer* sebelumnya, sehingga hanya *layer* akhir yang diperbarui. Tahap ini membantu model untuk beradaptasi dengan karakteristik data baru tanpa mengganggu pengetahuan yang telah didapat dari model pra-latih.
b. Tahap kedua : Setelah bagian akhir model beradaptasi dengan data baru, semua *layer* dilepas(*unfreeze*) dan model dilatih Kembali untuk beberapa *epoch* lagi, memungkinkan pembaruan seluruh bobot model secara lebih halus.

Proses pelatihan dilakukan pada tiga jenis model yakni model ResNet26, ResNet18, MobileNetV2, dengan menggunakan dataset yang terdiri dari 23.224 gambar selama 100 *epoch* dan menggunakan *Learning rate* 0.01. Setiap model dilatih melalui langkah langkah yang mencakup *forward propagation*, perhitungan *loss, backward propagation*, dan pembaruan bobot guna meningkatkan akurasi dan mengurangi *loss*. Model ResNet26 dan ResNet18 merupakan dua varian dari arsitektur *Residual Neural Network* (ResNet) yang dirancang untuk tugas-tugas klasifikasi gambar yang kompleks dengan data yang banyak dibandingkan dengan MobileNetV2 yang dirancang untuk fokus pada efisiensi dan komputasi ringan. Tujuan dari pelatihan ini adalah untuk menentukan model yang paling efektif dalam klasifikasi gambar aksara OKU Timur.

1) ResNet26



Gambar 12. Nilai loss dari model ResNet26 selama Proses pelatihan



Gambar 13. Nilai akurasi dari model ResNet26 selama Proses pelatihan

Hasil evaluasi model setelah 100 *epoch* pelatihan, dengan fokus pada nilai *loss* dan akurasi untuk data pelatihan serta data validasi:

Model	Epoch	Time per epoch	loss	val loss	Error rate	accuracy
ResNet26	Epoch 100/100	1m 2s	0.0396	0.1092	0.0164	0.9835

Gambar 14. Hasil training ResNet26

a. loss: 0.0396

Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 12, *training loss* turun dengan cepat di awal, kemudian secara perlahan terus menurun sampai akhirnya stabil pada angka yang cukup rendah. Ini menunjukkan bahwa model mampu memahami pola dalam data pelatihan dengan

baik tanpa mengalami kendala besar seperti kesulitan. Nilai akhir *training loss* sebesar 0,0396 menunjukkan bahwa model telah mempelajari dan merepresentasikan data pelatihan dengan sangat efektif.

b. val_loss: 0.1092

Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 12, nilai *loss* sebesar 0,1092 menunjukkan bahwa model mampu memprediksi data validasi dengan cukup baik. Namun, setelah *epoch* ke-30 dan ke-50, terjadi peningkatan pada nilai *loss*, yang mengindikasikan bahwa performa model pada data validasi menjadi kurang stabil dibandingkan dengan kinerjanya pada data pelatihan. Ini menandakan bahwa model mengalami kesulitan dalam menggeneralisasi pola dari data pelatihan ke data validasi, yang bisa mengarah pada overfitting.

c. error_rate: 0.0164

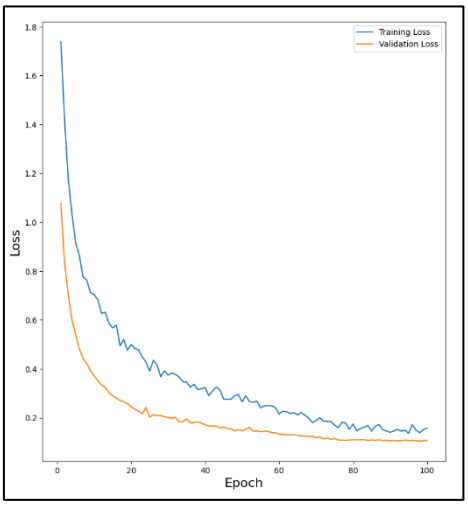
Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 13. Nilai error rate menurun dengan signifikan, namun ada beberapa kenaikan kecil yang terlihat di epoch tertentu, dengan nilai akhir 0.0164 yang menunjukan beberapa kesalahan atau ketidakstabilan dalam prediksi model pada data validasi.

d. accuracy: 0.9835

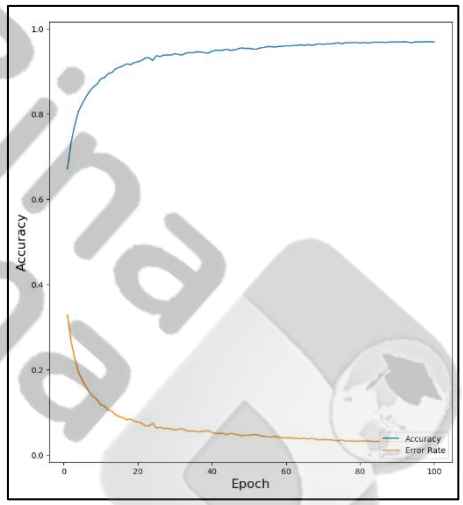
Berdasarkan hasil pelatihan yang ditampilkan pada gambar 13, akurasi meningkat tajam pada tahap awal pelatihan, menunjukkan bahwa model dengan cepat mampu mempelajari pola dari data. Nilai akurasi sebesar 0,9835 atau 98,35% menunjukkan bahwa model memiliki performa prediksi yang sangat baik, dengan tingkat akurasi yang tinggi.

Secara keseluruhan, nilai-nilai ini menunjukkan bahwa model memiliki performa yang sangat baik pada data pelatihan namun mengalami sedikit masalah pada data validasi, yang berarti model terlalu menyesuaikan diri dengan data pelatihan(*Overfitting*).

2) ResNet18



Gambar 15. Nilai loss dari model ResNet18 selama Proses pelatihan



Gambar 16. Nilai akurasi dari model ResNet18 selama Proses pelatihan

Hasil evaluasi model setelah 100 epoch pelatihan, dengan fokus pada nilai loss dan akurasi untuk data pelatihan serta data validasi:

Model	Epoch	Time per epoch	loss	val loss	Error rate	accuracy
ResNet18	Epoch 100/100	54s	0.1575	0.1075	0.0302	0.9697

Gambar 17. Hasil Training ResNet18

a. loss: 0.1575

Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 15. Pada kurva *training loss* terlihat menurun tajam pada awal pelatihan, seiring bertambahnya epoch penurunan ini menjadi lebih lambat namun tetap menurun secara stabil hingga akhir pelatihan. Dengan nilai akhir *training loss* sebesar 0,1575 yang cukup rendah, ini menunjukkan bahwa model telah mempelajari pola dari data pelatihan dengan sangat baik dan mampu menyesuaikan bobotnya secara efektif untuk mengurangi kesalahan.

b. val_loss: 0.1075

Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 15. Pada kurva *validation loss* terlihat penurunan nilai loss lebih stabil dibandingkan dengan *training loss* dan nilainya lebih rendah dibandingkan dengan loss pada pelatihan (0.1575) Nilai 0.1075 menunjukkan bahwa model memprediksi data validasi lebih baik dibandingkan dengan data pelatihan. Perbedaan nilai yang tidak terlalu jauh sehingga model mampu menggeneralisasi ke data baru.

c. error_rate: 0.0302

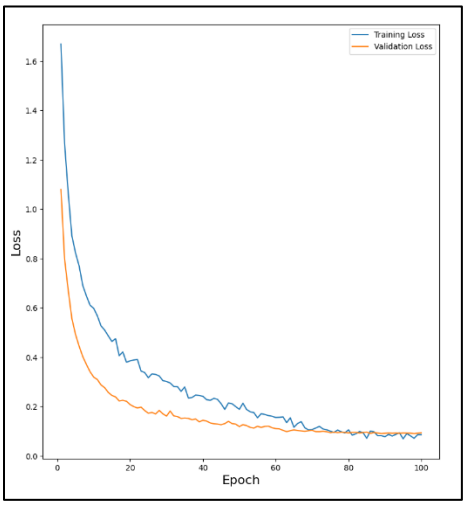
Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 16. Nilai error rate menurun pada awal epoch dan seiring bertambahnya epoch mendatar dengan konsisten, dengan nilai akhir 0.0302 yang berarti model membuat lebih sedikit kesalahan.

d. val_accuracy: 0.9697

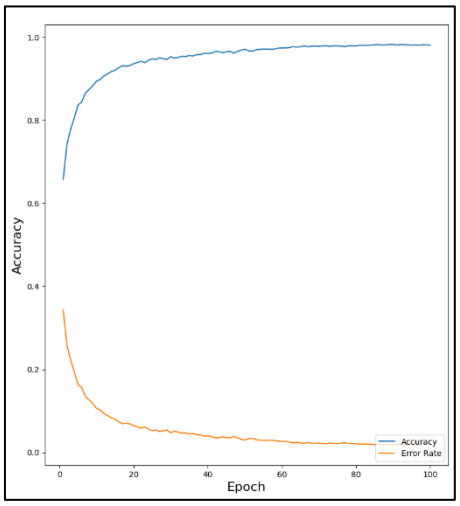
Berdasarkan hasil pelatihan yang ditunjukkan pada gambar 16, akurasi meningkat dengan pesat di awal pelatihan, menunjukkan bahwa model cepat dalam mempelajari data. Dengan nilai akurasi sebesar 0,9697 atau 96,97%, model menunjukkan kemampuan yang sangat baik dalam mempelajari dan memprediksi data.

Secara keseluruhan, hasil ini menunjukkan bahwa model memiliki performa yang sangat baik pada data pelatihan dan pada data validasi.

3) MobileNetV2



Gambar 18. Nilai loss dari model MobileNetV2 selama Proses pelatihan



Gambar 19. Nilai akurasi dari model MobileNetV2 selama Proses pelatihan

Hasil evaluasi model setelah 100 epoch pelatihan, dengan fokus pada nilai loss dan akurasi untuk data pelatihan serta data validasi:

Model	Epoch	Time per epoch	loss	val loss	error rate	accuracy
MobileNetV2	Epoch 100/100	1m 20s	0.0855	0.0932	0.0198	0.9801

Gambar 20. Hasil Training MobiileNetV2

a. loss: 0.0855

Berdasarkan hasil pelatihan yang ditunjukkan pada Gambar 18, nilai *loss* pada data pelatihan menurun secara signifikan seiring bertambahnya *epoch*. Ini menunjukkan bahwa model tidak mengalami kesulitan dalam mempelajari data pelatihan. Dengan nilai akhir *loss* sebesar 0.0855, dapat disimpulkan bahwa model telah mempelajari data pelatihan dengan sangat baik.

b. val_loss: 0.0932

Berdasarkan hasil pelatihan yang ditunjukkan pada Gambar 18. Nilai validasi loss pada data validasi menurun seiring bertambahnya epoch, yang menunjukkan bahwa model bekerja dengan baik pada data yang tidak terlihat. Dengan nilai akhir *loss* sebesar 0.0932, dapat disimpulkan bahwa model memprediksi hampir semua data validasi.

c. error_rate: 0.0198

Berdasarkan hasil pelatihan yang ditunjukkan pada Gambar 19. Nilai error rate menurun pada awal epoch dan seiring bertambahnya epoch mendatar dengan konsisten, dengan nilai akhir 0.0198 yang berarti model membuat lebih sedikit kesalahan.

d. accuracy: 0.9801

Berdasarkan hasil pelatihan yang ditunjukkan pada Gambar 19, akurasi meningkat tajam di awal pelatihan, menunjukkan bahwa model dengan cepat mempelajari data. Dengan nilai akurasi

sebesar 0,9801 atau 98,01%, model menunjukkan performa yang sangat baik dalam memprediksi data.

Secara keseluruhan, hasil ini menunjukkan bahwa model memiliki performa yang sangat baik pada data pelatihan dan pada data validasi

5. Evaluasi Model

Model	Epoch	Time per Epoch	loss	val loss	error rate	accuracy
ResNet26	Epoch 100/100	1m 2s	0.0396	0.1092	0.0164	0.9835
ResNet18	Epoch 100/100	54s	0.1575	0.1075	0.0302	0.9697
MobileNetV2	Epoch 100/100	1m 20s	0.0855	0.0932	0.0198	0.9801

Gambar 21. Rekap hasil training model

Ketiga model klasifikasi gambar, yaitu ResNet26, ResNet18, dan MobileNetV2, menunjukan kinerja yang baik pada data pelatihan dan data validasi.

1) Model ResNet26, nilai accuracy sebesar 98.35% ini menunjukkan bahwa model sangat baik dalam mempelajari dan memprediksi data. Walaupun nilai error_rate cukup rendah yang berarti model membuat sedikit kesalahan. Namun nilai validasi loss lebih tinggi dari nilai pelatihan loss yang mengindikasikan bahwa model terlalu menyesuaikan diri dengan data pelatihan (overfitting). Arsitektur model yang digunakan sangat sesuai dengan jenis data yang diolah sehingga hasil yang didapat dari pelatihan model ini sangat baik.
2) Model ResNet18 menunjukkan hasil yang kurang dibandingkan dengan model ResNet26 dengan nilai akurasi 96.97%. namun performa dalam pelatihan loss dan validasi loss cukup baik. Sehingga bisa dikatakan bahwa arsitektur model yang digunakan cukup sesuai dengan jenis data yang diolah sehingga hasil yang didapat dari pelatihan model ini cukup baik.
3) Model MobileNetV2 menunjukkan hasil yang cukup baik dibandingkan dengan model ResNet18 tetapi tidak lebih dari model ResNet26 dengan nilai akurasi sebesar 98.01% ini menunjukkan bahwa model sangat baik dalam mempelajari dan memprediksi data. Kemudian nilai pelatihan loss dan validasi loss cukup baik. Perbedaan nilai tersebut tidak terlalu jauh sehingga bisa dikatakan bahwa arsitektur model yang digunakan cukup sesuai dengan jenis data yang diolah sehingga hasil yang didapat dari pelatihan model ini sangat baik.

4. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, model klasifikasi gambar aksara OKU Timur yang telah dikembangkan dengan menggunakan metode *Transfer Learning*, dengan menerapkan metode pengembangan *Rapid Application Development* (RAD), dapat disimpulkan bahwa ResNet26 memiliki kinerja tertinggi dengan akurasi sebesar 98.35%, namun terjadi *overfitting* karena model terlalu menyesuaikan diri dengan data pelatihan. Berikutnya diikuti oleh MobileNetV2 dengan akurasi sebesar 98,01% meskipun nilai akurasi MobileNetV2 dibawah ResNet26 tetapi kinerja dalam mempelajari dan memprediksi data sangat baik. Terakhir ResNet18 dengan akurasi 96.97%, nilai akurasi dari ResNet tergolong kecil dibandingkan kedua model yang lain, akan tetapi model tidak mengalami *overfitting*. Penelitian ini hanya menghasilkan output berupa file model klasifikasi gambar yang sudah dilatih menggunakan metode Transfer Learning, yang nantinya ketiga model ini dapat dipertimbangkan untuk diterapkan ke dalam aplikasi *mobile* untuk mendeteksi atau mengklasifikasi aksara OKU Timur dalam pengenalan aksara otomatis. Aplikasi ini dapat digunakan oleh Masyarakat umum, peneliti, dan institusi pendidikan sebagai alat pembelajaran dan pelestarian aksara OKU Timur. Selain itu, pengembangan lebih lanjut dapat mencakup peningkatan akurasi dan menggunakan teknik untuk mengatasi masalah *overfitting*

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada semua pihak yang telah membantu dan mendukung penulis baik secara langsung ataupun tidak langsung sehingga penelitian kami dapat terselesaikan, penulis juga mengucapkan kepada Universitas Bina Darma khususnya Inteligent System Research Group yang sudah membantu dalam hal apapun.

DAFTAR PUSTAKA

[1] E. Roza, "Aksara Arab-Melayu di Nusantara dan Sumbangsihnya dalam Pengembangan Khazanah Intelektual," *TSAQAFAH*, vol. 13, no. 1, p. 177, May 2017, doi: 10.21111/tsaqafah.v13i1.982.

[2] M. Affan Ridhollah, P. Sejarah Peradaban Islam, and F. Adab dan Humaniora, "PELESTARIAN AKSARA ULU SUMATERA SELATAN SEBAGAI KEARIFAN LOKAL MASYARAKAT DESA SUGIHWARAS MELALUI PELATIHAN BACA TULIS AKSARA ULU," *Jurnal Magister Sejarah Peradaban Islam*, vol. 02, no. 02, 2023.

[3] Y. N. Kunang, I. Z. Yadi, and Mahmud, *A New Deep Learning-Based Mobile Application for Komering Character Recognition*. 2022.

[4] T. F. Kusumaningrum, "Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Jamur Konsumsi Di Indonesia Menggunakan Keras," 2018.

[5] D. Hafidzulrahman, "Perbandingan Algoritma You Only Look Once (YOLO) versi 5 dan versi 8 sebagai Object Detection pada Pendeteksian Hilal."

[6] I. Sánchez Fernández and J. M. Peters, "Machine learning and deep learning in medicine and neuroimaging," *Annals of the Child Neurology Society*, vol. 1, no. 2, pp. 102–122, Jun. 2023, doi: 10.1002/cns3.5.

[7] M. Torres-Velazquez, W. J. Chen, X. Li, and A. B. McMillan, "Application and Construction of Deep Learning Networks in Medical Imaging," Mar. 01, 2021, *Institute of Electrical and Electronics Engineers Inc.* doi: 10.1109/TRPMS.2020.3030611.

[8] S. Chen and W. Guo, "Auto-Encoders in Deep Learning—A Review with New Perspectives," Apr. 01, 2023, *MDPI*. doi: 10.3390/math11081777.

[9] I. F. Alam, M. Ihsan Sarita, and A. M. Sajiah, "IMPLEMENTASI DEEP LEARNING DENGAN METODE CONVOLUTIONAL NEURAL NETWORK UNTUK IDENTIFIKASI OBJEK SECARA REAL TIME BERBASIS ANDROID," *semanTIK*, vol. 5, no. 2, pp. 237–244, 2019, doi: 10.5281/zenodo.3459374.

[10] R. Yang and Y. Yu, "Artificial Convolutional Neural Network in Object Detection and Semantic Segmentation for Medical Imaging Analysis," Mar. 09, 2021, *Frontiers Media S.A.* doi: 10.3389/fonc.2021.638182.

[11] N. D. Girsang, "Literature Study of Convolutional Neural Network Algorithm for Batik Classification," vol. 1, no. 1, 2021, doi: 10.47709/briliance.v1i1.1069.

[12] M. G. Galety, F. H. Al Mukthar, R. J. Maaroof, and F. Roffo, "Deep Neural Network Concepts for Classification using Convolutional Neural Network: A Systematic Review and Evaluation," 2021. [Online]. Available: www.techniumscience.com

[13] B. Falahkhi, E. F. Achmal, M. Rizaldi, R. Rizki, and N. Yudistira, "Perbandingan Model AlexNet dan ResNet dalam Klasifikasi Citra Bunga Memanfaatkan Transfer Learning," *Ilmu Komputer Agri-Informatika*, vol. 9, no. 1, pp. 70–78, 2022, [Online]. Available: http://journal.ipb.ac.id/index.php/jika

[14] G. Pinto, Z. Wang, A. Roy, T. Hong, and A. Capozzoli, "Transfer learning for smart buildings: A critical review of algorithms, applications, and future perspectives," *Advances in Applied Energy*, vol. 5, Feb. 2022, doi: 10.1016/j.adapen.2022.100084.

[15] X. Du, Y. Sun, Y. Song, H. Sun, and L. Yang, "A Comparative Study of Different CNN Models and Transfer Learning Effect for Underwater Object Classification in Side-Scan Sonar Images," *Remote Sens (Basel)*, vol. 15, no. 3, Feb. 2023, doi: 10.3390/rs15030593.

[16] M. Iman, H. R. Arabnia, and K. Rasheed, "A Review of Deep Transfer Learning and Recent Advancements," Apr. 01, 2023, *MDPI*. doi: 10.3390/technologies11020040.

[17] A. E. Putra, M. F. Naufal, and V. R. Prasetyo, "Klasifikasi Jenis Rempah Menggunakan Convolutional Neural Network dan Transfer Learning," *JEPIN, Jurnal Edukasi dan Penelitian Informatika*, vol. 9, no. 1, 2023.

[18] R. FATURRAHMAN, Y. S. HARIYANI, and S. HADIYOSO, "Klasifikasi Jajanan Tradisional Indonesia berbasis Deep Learning dan Metode Transfer Learning," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 11, no. 4, p. 945, Oct. 2023, doi: 10.26760/elkomika.v11i4.945.

[19] M. Murinto, M. Rosyda, and M. Melany, "Klasifikasi Jenis Biji Kopi Menggunkan Convolutional Neural Network dan Transfer Learning pada Model VGG16 dan MobileNetV2," *JRST (Jurnal Riset Sains dan Teknologi)*, vol. 7, no. 2, p. 183, Sep. 2023, doi: 10.30595/jrst.v7i2.16788.

[20] A. Suryanto and M. I. Maliki, "Penerapan Model Rapid Application Development (RAD) Dalam Rancang Bangun Sistem Informasi Warga," *Infotek : Jurnal Informatika dan Teknologi*, vol. 5, no. 1, pp. 197–208, Jan. 2022, doi: 10.29408/jit.v5i1.4887.

[21] J. R. Sagala, "MODEL RAPID APPLICATION DEVELOPMENT (RAD) DALAM PENGEMBANGAN SISTEM INFORMASI PENJADWALAN BELAJAR MENGAJAR," *Jurnal Mantik Penusa*, vol. 2, no. 1, 2018.

[22] D. Hariyanto, R. Sastra, and F. E. Putri, "Implementasi Metode Rapid Application Development Pada Sistem Informasi Perpustakaan," *Jurnal Jupiter*, vol. 13, no. 1, 2021.

[23] H. Rianto, "Rancang Bangun Sistem Informasi Inventory Menggunakan Metode Rapid Application Development," 2023. [Online]. Available: http://jurnal.bsi.ac.id/index.php/insantek

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[25] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks."

[26] I. Topan Adib Amrulloh *et al.*, "EVALUASI AUGMENTASI DATA PADA DETEKSI PENYAKIT DAUN TEBU DENGAN YOLOV8," 2024.

[27] L. Hansel G and H. Bunyamin, "Penggunaan Augmentasi Data pada Klasifikasi Jenis Kanker Payudara dengan Model Resnet-34," 2021. [Online]. Available: https://cs231n.github.io/convolutional-networks/

[28] I. R. Fadhillah *et al.*, "IMPLEMENTASI MODEL TRANSFER LEARNING EFFICIENTNET UNTUK PENDETEKSIAN BAHASA ISYARAT INDONESIA (BISINDO) PADA PERANGKAT ANDROID," 2024.

[29] F. Dzil, Agus Khumaidi, Mohammad Basuki Rahmat, Joko Endrasmono4, Mat Syai'in, and Dimas Pristovani Riananda, "Deteksi Objek di Lapangan pada Robot Sepakbola Beroda Menggunakan Metode YOLOV5," *Jurnal Elektronika dan Otomasi Industri*, vol. 11, no. 2, pp. 604–611, Jul. 2024, doi: 10.33795/elkolind.v11i2.5235.

[30] M. E. Muliyono Zeez, "IMPLEMENTASI AUTOGLUON DALAM EFISIENSI MODEL PREDIKTIF MACHINE LEARNING PADA DATASET INTERNATIONAL BUSINESS MACHINES (IBM)."