BERNAS: Jurnal Pengabdian Kepada Masyarakat
Vol. 5 No 2, 2024, pp. 1278-1285
DOI: https://doi.org/10.31949/jb.v5i2.8187

e-ISSN 2721-9135
p-ISSN 2716-442X

IMAGE CLASSIFICATION UNTUK TELUR AYAM MENGGUNAKAN SMARTPHONE ANDROID DENGAN CONVOLUTIONAL NEURAL NETWORKS (CNN)

Sandhya Nugraha Qusnur Aulia
Universitas Amikom Purwokerto, Purwokerto, Indonesia
sandhys66@gmail.com

Abstract
Development of a method for classifying egg stages using Convolutional Neural Networks (CNN) with a focus on efficiency and accuracy. In its implementation, this method utilizes an Android smartphone, TensorFlow Keras, and Google Colab for model training. The dataset consists of images representing early, middle, late stages, and dead eggs. The training results demonstrate satisfactory performance with good accuracy and low loss, validating the model for real-time egg stage detection through an Android application. The research outlines the influence of the dataset size, training parameters, and model optimization on accuracy, with the use of Google Colab as an adequate resource. The model is exported to TensorFlow Lite for Android implementation, with an emphasis on using float32 in the model requiring metadata before conversion. It is anticipated that this method can contribute as an efficient and accurate alternative in egg stage classification accessible to the scientific community.
Keywords: *Egg stage classification; Android application; Realtime detection*

Abstrak
Pengembangan metode klasifikasi tahapan telur menggunakan Convolutional Neural Networks (CNN) dengan fokus pada efisiensi dan akurasi. Dalam implementasinya, metode ini memanfaatkan smartphone Android, TensorFlow Keras, dan Google Colab untuk pelatihan model. Dataset terdiri dari gambar tahap awal, pertengahan, akhir, dan telur mati. Hasil pelatihan menunjukkan performa yang baik dengan akurasi dan loss yang memuaskan, memvalidasi model untuk deteksi tahapan telur secara real-time melalui aplikasi Android. Penelitian ini merinci pengaruh jumlah dataset, parameter pelatihan, dan optimasi model terhadap akurasi, dengan penggunaan Google Colab sebagai sumber daya yang memadai. Model diekspor ke TensorFlow Lite untuk implementasi di Android, dan penekanan pada penggunaan float32 pada model yang memerlukan metadata sebelum konversi. Diharapkan metode ini dapat memberikan kontribusi sebagai alternatif efisien dan akurat dalam klasifikasi tahapan telur yang dapat diakses oleh komunitas ilmiah.
Kata Kunci: Klasifikasi tahap telur; aplikasi Android; deteksi realtime;

Accepted: 2024-01-08 Published: 2024-04-03

PENDAHULUAN

Telur ayam merupakan salah satu hasil peternakan yang banyak dikonsumsi oleh Masyarakat Indonesia, bahkan kebutuhan telur ayam menjadi meningkat pada perayaan hari-hari keagamaan tertentu. Peternakan telur ayam dapat menghasilkan jumlah telur yang banyak setiap hari. (Studi Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak Jln Merdeka No & Barat, 2013).

Klasifikasi telur merupakan salah satu hal yang penting untuk melihat tahap telur apakah masih di tahap awal, pertengahan atau akhir. Proses klasifikasi telur secara manual memiliki beberapa kekurangan, yaitu membutuhkan waktu yang lama, kurangnya keakuratan dan membutuhkan tenaga kerja yang banyak. Oleh karena itu, diperlukan metode klasifikasi telur yang lebih efisien dan akurat.

Metode klasifikasi telur dengan menggunakan machine learning merupakan salah satu metode yang berpotensi untuk mengatasi kekurangan-kekurangan tersebut. Metode ini dapat dilakukan

dengan hanya menggunakan kamera smartphone untuk mengambil gambar telur. Kemudian gambar tersebut akan diproses oleh algoritma machine learning yang nantinya dapat dihasilkan klasifikasi telur.

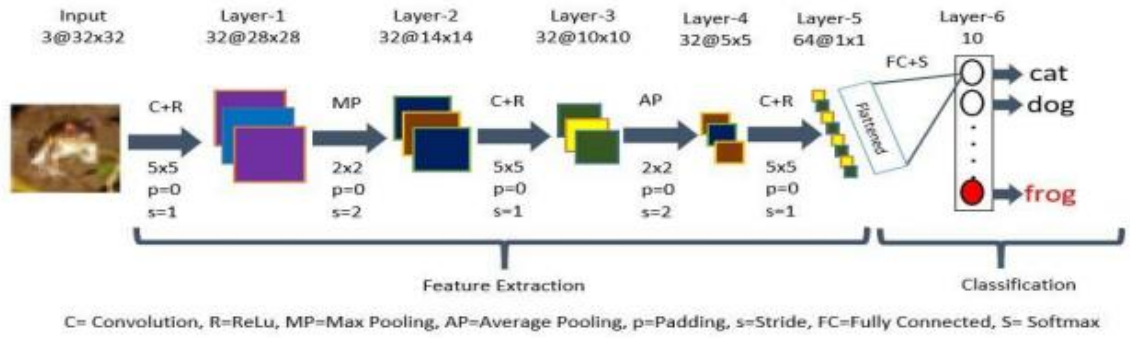
Metode klasifikasi telur dengan machine learning sudah banyak dikembangkan oleh peneliti-peneliti di berbagai negara. Namun, Sebagian besar penelitan tersebut masih menggunakan perangkat keras dan perangkat lunak yang kompleks dan tidak semua orang dapat mengoperasi perangkat tersebut.

Oleh karena itu, penelitian ini bertujuan untuk mengembangkan metode klasifikasi telur dengan hanya menggunakan smartphone Android dan algoritma machine learning yang sederhana. Metode ini diharapkan dapat menjadi alternatif metode klasifikasi telur yang lebih efisien dan akurat, serta dapat digunakan oleh masyarakat luas seperti para peternak telur kecil yang tidak punya banyak sumber daya. Aplikasi ini menggunakan algoritma sederhana dalam machine learning yaitu Convolutional Neural Networks (CNN)

METODE

Convolutional Neural Network (CNN)

Convolutional Neural Networks yang sederhana merupakan urutan dari beberapa lapisan. Lapisan tersebut berubah dari satu volume yang aktif ke lapisan yang lainnya melalui beberapa fungsi. Lapisan tersebut berupa Lapisan Convolutional, lapisan pooling, dan lapisan yang sudah terkoneksi penuh. (Hossain & Alam Sajib, 2019)



Gambar 1. Arsitektur CNN

Gambar tersebut merupakan arsitektur secara umum dari algoritma CNN. Pertama tama akan dilakukan beberapa proses gambar sederhana seperti merubah ukuran gambar, menormalisasi ukuran pixel, dll. Setelah beberapa proses tersebut, model akan siap dipakai.

Lapisan pertama terdiri dari lapisan convolutional dengan ReLu (Recified Linear Unit). Lapisan ini akan diproses dengan ukuran dari n*n = 32*32. Lapisan kedua merupakan lapisan max pooling. Lapisan ini mendapatkan besar input 32@28*28 dari lapisan yang sebelumnya. Ukuran dari pooling ini adalah 2*2. Setelah itu kita akan mendapatkan maps feature sebesar 32@14*14. Lapisan ketiga merupakan lapisan convolutional kedua menggunakan fungsi aktivasi ReLu. Filter yang digunakan adalah 5*5.

Lapisan keempat meruapakan lapisan pooling rata – rata. Lapisan ini mendapatkan ukuran 32@10*10 dari lapisan yang sebelumnya. Setelah operasi ini kita mendapatkan feature map sebesar 32@5*5. Layer kelima merupakan lapisan convolutional ketiga dengan aktivasi ReLu. Lapisan ini mendapatkan input size sebesar 32@5*5 dari lapisan yang sebelumnya. Ukuran

filternya adalah 4*4. Lapisan terakhir meruapakan lapisan yang sudah terkoneksi sepenuhnya. Lapisan ini akan menghitung skor kelas, yang akan meneghasilkan ukuran sebesar 10, Di mana setiap angka 10 akan mempengaruhi CIFAR-10 dataset.

TensorFlow Keras

Tensorflow merupakan open source library untuk machine learning yang di release oleh Google yang mendukung beberapa Bahasa pemrograman. (*Transfer Learning for Image Classification of Various Dog Breeds Pratik Devikar*, n.d.)

Untuk mengimplementasikan algoritma ini kami akan menggunakan TensorFlow Keras agar dapat diimplementasikan ke Android maupun ke Backend Cloud sepenuhnya dengan mudah dan cepat. Kemudian untuk module nya kami menggunakan mobilenet_v2 dengan tipe raw dan allow inputnya true.

Google Colab

Google Colab atau Google Colaboratory adalah layanan open source yang disediakan oleh Google untuk siapapun yang mempunyai akun Gmail. Google Colab menyediakan GPU untuk research bagi siapapuny yang tidak mempunyai alat yang memadai. Layanan Google Colab menyediakan 12.72GB RAM dan 358.27GB Hard Disk dalam sekali jalan. (Kanani & Padole, 2019)

```
local_zip = '/content/drive/MyDrive/Colab Notebooks/egg-condition-classifications.v2i.multiclass (with test).zip'
zip_ref = zipfile.ZipFile(local_zip, 'r')
zip_ref.extractall('/content/data/')
zip_ref.close()

# Assign training and validation set directories
base_dir = '/content/data'
train_dir = os.path.join(base_dir, 'train')
valid_dir = os.path.join(base_dir, 'valid')
test_dir = os.path.join(base_dir, 'test')

# Load the labels
train_labels = pd.read_csv("/content/data/train/_classes.csv")
valid_labels = pd.read_csv("/content/data/valid/_classes.csv")
test_labels = pd.read_csv("/content/data/test/_classes.csv")
```

Gambar 2. Memasukkan Dataset

Pada gambar diatas kami memasukkan tiga dataset yang berbeda untuk ke model yang akan kita jalankan yang kemudian akan diparsing menjadi beberapa tahap yaitu : pertengahan, tahap akhir , tahap awal dan telur mati. Kemudian data yang sudah di parsing akan dilakukan tahap awal yaitu pre processing. Gambar dari data set ini nantinya akan diubah ukurannya untuk di train. Berikut merupakan hasil gambar yang sudah di validasi menurut empat kelas atau tahap yang tadi.

```
Found 429 validated image filenames belonging to 4 classes.
Found 165 validated image filenames belonging to 4 classes.
Found 68 validated image filenames belonging to 4 classes.
```

Gambar 3. Validasi tahap

Untuk tahap selanjutnya kita hanya Menyusun data yang digunakan dalam pelatihan (train_dataset), validasi (valid_dataset), dan pengujian (test_dataset) model. Ini dapat terkait juga dengan tugas klasifikasi, Dimana setiap gambar diberi label kedalam empat kelas.

```
train_dataset = tf.data.Dataset.from_generator(
    lambda: train_generator,
    output_signature=(
        tf.TensorSpec(shape=(BATCH_SIZE,224, 224, 3), dtype=tf.float32),
        tf.TensorSpec(shape=(BATCH_SIZE, 4), dtype=tf.float32) # 4 for nu
    )
)

valid_dataset = tf.data.Dataset.from_generator(
    lambda: valid_generator,
    output_signature=(
        tf.TensorSpec(shape=(BATCH_SIZE,224, 224, 3), dtype=tf.float32),
        tf.TensorSpec(shape=(BATCH_SIZE, 4), dtype=tf.float32) # 4 for nu
    )
)

test_dataset = tf.data.Dataset.from_generator(
    lambda: test_generator,
    output_signature=(
        tf.TensorSpec(shape=(1,224, 224, 3), dtype=tf.float32),
        tf.TensorSpec(shape=(1, 4), dtype=tf.float32) # 4 for num of clas
    )
)
```

Gambar 4. Train dataset

Objek 'tf.data.Dataset' dari generator digunakan untuk menyiapkan data pelatihan, validasi dan pengujian untuk digunakan dalam model TensorFlow. Output dari generator yang didefinisikan oleh 'output_signature' menunjukkan struktur dan tipe data dari setiap elemen dalam dataset. Dalam konteks ini dataset terdiri dari pasangan gambar dan label, dan 'tf.TensorSpec' digunakan untuk memberikan spesifikasi mengenai dimensi dan tipe data dari setiap elemen dalam tupel.

Hal ini dapat diterapkan pada berbagai domain penelitian, termasuk pengenalan objek dalam gambar atau tugas klasifikasi. BATCH_SIZE dan dimensi gambar (224 x 224 x 3) sesuai dengan kebutuhan model dan data yang digunakan.

```
do_fine_tuning = True

feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                   input_shape=IMAGE_SIZE + (3,),
                                   output_shape=[FV_SIZE],
                                   trainable=do_fine_tuning)

print("Building model with", MODULE_HANDLE)

model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Flatten(),
        # tf.keras.layers.Dropout(0.1),
        tf.keras.layers.Dense(256, activation='relu'),
        # tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(4, activation='softmax')
])

model.summary()
```

Gambar 5. Memilih Ekstraktor

Dalam bagian ini, kita memilih fitur ekstraktor dari TensorFlow hub, ekstraktor ini seperti otak model yang telah di training sebelumnya untuk mengenali pola dalam gambar. Lalu lapisan pengaturan ulang bentuk data, kemudian dilanjutkan dengan dua lapisan terhubung penuh (dense

layers) untuk klasifikai dan memberikan gambaran umum tentang struktur model, termasuk jumlah parameter yang digunakan dan bentuk keluaran dari setiap layer.

```
class myCallback(tf.keras.callbacks.Callback):
  def on_epoch_end(self, epoch, logs={}):
    # Check the loss
    if(logs.get('val_accuracy') > 0.92 and logs.get('accuracy') > 0.976 ):

      # Stop if threshold is met
      print("\n val_acc > 92 & acc > 97. stop training")
      self.model.stop_training = True

# Instantiate class
callbacks = myCallback()


lr_callback = tf.keras.callbacks.ReduceLROnPlateau(
    monitor='val_loss',
    factor=0.1,
    patience=5,
    min_lr=1e-8
)
```

Gambar 6. Menentukan Batas train

Dalam gambar diatas kita akan menghentikan pelatihan model jika akurasi validasi sudah melebihi 92% dan akurasi pelatihan melebihi 97%. Selain itu, kita juga mengurangi Tingkat pembelajaran jika kerugian validasi tidak mengalami perbaikan dalam 5 epoch berturut-turut. Callback ini berguna untuk memperbaiki konvergensi model. Kedua callback ini dapat digunakan secara bersamaan selama pelatihan model.

```
if do_fine_tuning:
    model.compile(
                optimizer=tf.keras.optimizers.SGD(learning_rate=3.9e-5),
                # optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                loss=tf.keras.losses.CategoricalCrossentropy(),
                metrics=['accuracy'])
else:
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                loss='categorical_crossentropy',
                metrics=['accuracy'])

steps_per_epoch = train_generator.n
validation_steps = valid_generator.n

EPOCHS = 200
hist = model.fit(train_generator,
                epochs=EPOCHS,
                validation_data=valid_generator,
                steps_per_epoch=steps_per_epoch // 13,
                verbose=2,
                validation_steps=validation_steps // 13,
                callbacks=[callbacks, lr_callback])
```

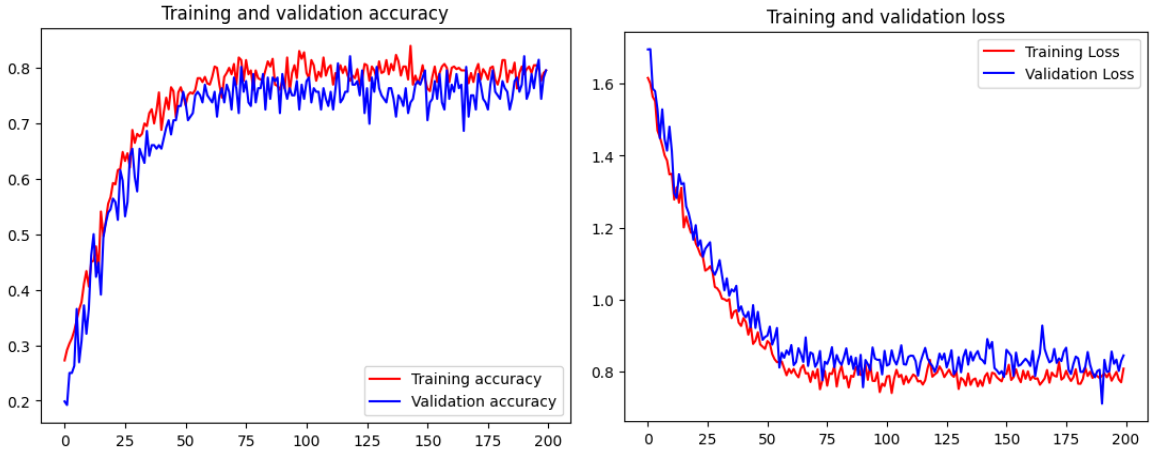
Gambar 7. Menentukan Optimizer

Lalu kita melakukan pelatihan model. Kita akan menggunakan optimisasi Stochastic Gradient Descent (SGD) dengan Tingkat pembelajaran 3.9e-5. Selanjutnya, model dikompilasi dengan fungsi kerugian categorical crossentropy dan metrik akurasi. Proses pelatihan dilakukan selama

200 epoch dengan menggunakan generator pelatihan dan validasi, serta membatasi untuk menghentikan pelatihan jika kondisi akurasi terpenuhi.

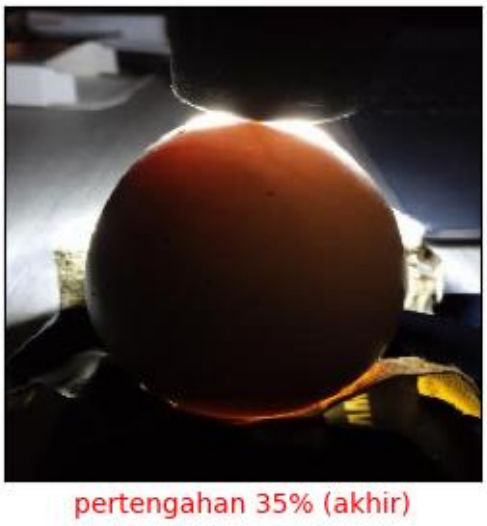
HASIL DAN PEMBAHASAN

1. Hasil Training



Gambar 8. Hasil Training

Gambar diatas merupakan hasil dari training yang sudah dilakukan. Terlihat training akurasi mendekati validation accuracy dan loss tersebut juga rendah dan mendekati validation loss yang berarti model ini cukup akurat. Berikut gambar yang merupakan hasil dari training model tersebut.

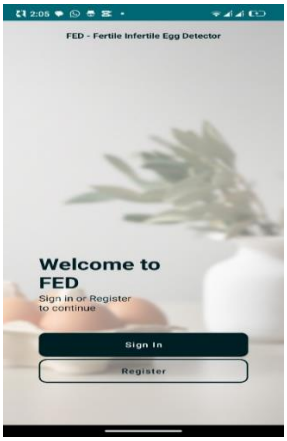


Gambar 9. Prediksi hasil

Kemudian kita hanya perlu export ke Tensorflow Lite dengan metadata agar dapat dibaca oleh Android. Setelah kita convert maka kita perlu mengunduh file tersebut yang kemudian akan diimplementasikan ke aplikasi Android supaya dapat menggunakan realtime image detection untuk mengklasifikasi tahapan telur ayam langsung dari device Android.

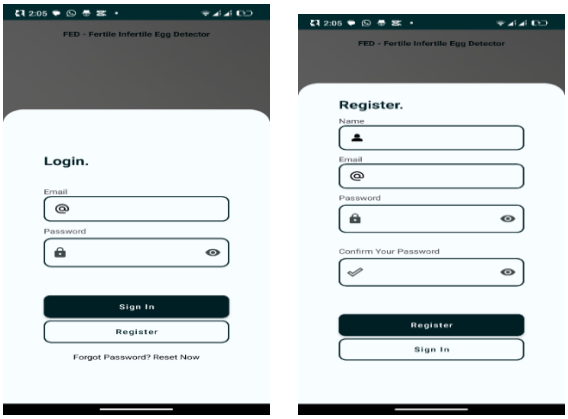
Tampilan Aplikasi

a. Tampilan Awal



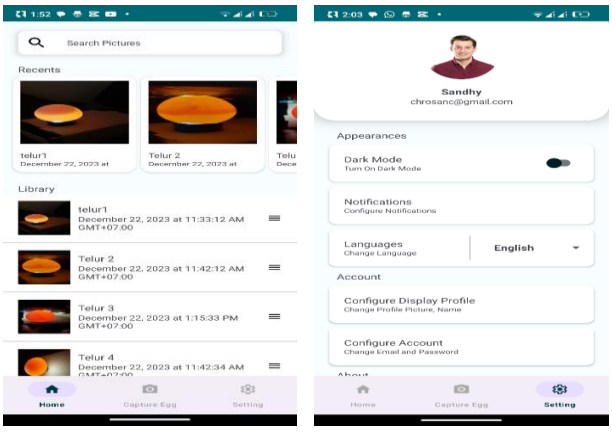
Gambar 10. Tampilan awal

b. Tampilan Login dan Register



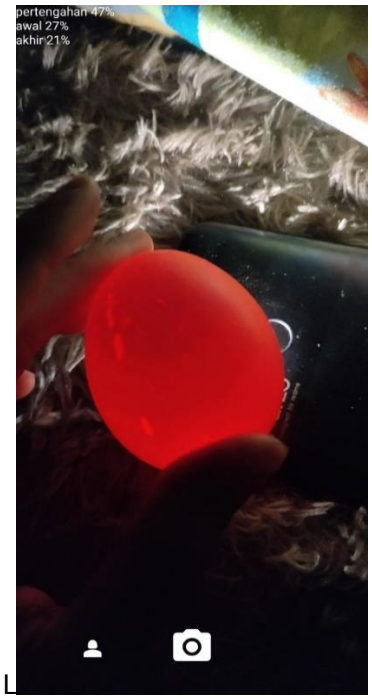
Gambar 11. Tampilan Login dan Register

c. Tampilan Home dan Setting



Gambar 12. Tampilan Home dan Setting

d. Tampilan Realtime Detection



Gambar 13. Tampilan Realtime Detection

KESIMPULAN

Berdasarkan penelitian dan hasil penerapan metode Convolutional Neural Network (CNN) dalam mengklasifikasikan tahapan telur, dapat diambil kesimpulan sebagai berikut:

1. Jumlah dataset tidak selalu mempengaruhi nilai akurasi, tetapi berdasarkan detail telur untuk digunakan dataset yang mempengaruhi hasil akurasi.
2. Penentuan Batch Size, Epoch, Optimizer dan jumlah convolutional layer sangat mempengaruhi akurasi yang akan didapatkan.
3. Menggunakan google colab tidak akan mempengaruhi spek computer yang digunakan. Karena langsung menggunakan alat dari google langsung.
4. Penggunaan float32 pada model diperlukan metadata sebelum di konversi ke file Tflite untuk digunakan di Android.

DAFTAR PUSTAKA

Hossain, Md. A., & Alam Sajib, Md. S. (2019). Classification of Image using Convolutional Neural Network (CNN). *Global Journal of Computer Science and Technology*, 13–18.

Kanani, P., & Padole, M. (2019). Deep learning to detect skin cancer using google colab. *International Journal of Engineering and Advanced Technology*, *8*(6), 2176–2183.

Studi Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak Jln Merdeka No, P., & Barat, K. (2013). *KLASIFIKASI TELUR AYAM DAN TELUR BURUNG PUYUH MENGGUNAKAN METODE CONNECTED COMPONENT ANALYSIS IKHWAN RUSLIANTO* (Vol. 3, Issue 1).

Transfer Learning for Image Classification of various dog breeds Pratik Devikar. (n.d.).