BAB 5. PENUTUP

5.1. Kesimpulan

Berdasarkan hasil penelitian mengenai Rancang Bangun Aplikasi Klasifikasi Bahasa Isyarat Menggunakan Metode MobileNet-SSD Berbasis Android, maka dapat diambil beberapa kesimpulan sebagai berikut.

1. Pembangunan model *Deep learning* untuk mendeteksi dan mengklasifikasikan bahasa isyarat dilakukan melalui serangkaian tahapan yang meliputi pengumpulan *dataset*, praproses data, *training* model, dan evaluasi model. *Dataset* yang digunakan berasal dari *dataset* publik Kaggle dan citra yang dikumpulkan secara mandiri sehingga diperoleh 5.760 citra yang terdiri atas 24 kelas huruf SIBI (A–Y, kecuali J dan Z). Selanjutnya dilakukan proses pelabelan (*annotation*) menggunakan *bounding box*, *image resizing*, validasi anotasi, serta pembagian data latih dan data uji dengan rasio 80:20. Model kemudian dibangun menggunakan TensorFlow Object Detection API dengan pendekatan *Transfer learning* pada model SSD MobileNetV2 FPNLite 320×320, kemudian dilakukan proses *training*, *validation*, dan evaluasi menggunakan metrik mAP, *precision*, *recall*, *loss,* dan *learning rate*. Hasil *training* menunjukkan model memperoleh mAP sebesar 0,75–0,78, mAP@0.50IoU di atas 0,95, *recall* sebesar 0,88–0,90, dan akurasi data uji sebesar 99,48%, sehingga model mampu mendeteksi dan mengklasifikasikan bahasa isyarat dengan tingkat akurasi yang tinggi.
2. Integrasi hasil pemodelan *Deep Learning* MobileNet-SSD ke dalam aplikasi Android berhasil dilakukan melalui beberapa tahapan, yaitu mengekspor model hasil pelatihan ke format SavedModel, mengonversinya menjadi TensorFlow Lite (.tflite), kemudian mengimplementasikannya pada aplikasi Android yang dikembangkan menggunakan Android Studio dengan bahasa pemrograman Kotlin dan arsitektur MVVM. *File* detect.tflite dan labels.txt ditempatkan pada direktori assets dan dimuat menggunakan TensorFlow Lite Interpreter sehingga proses inferensi dapat dilakukan secara *on-device* tanpa memerlukan koneksi internet. Implementasi sistem disusun ke dalam beberapa *package* berdasarkan tanggung jawab masing-masing, yaitu camera untuk akuisisi citra menggunakan CameraX, detector untuk *preprocessing*, inferensi, dan pengolahan hasil prediksi, overlay untuk menampilkan *bounding box*, label, dan *confidence score*, serta ui untuk mengintegrasikan seluruh proses ke dalam antarmuka aplikasi. Alur kerja sistem dimulai ketika kamera mengambil citra secara kontinu, setiap *frame* dikonversi menjadi *Bitmap*, dilakukan *preprocessing* berupa *image resizing* menjadi 320 × 320 piksel dan normalisasi, kemudian diproses

oleh model MobileNet-SSD menggunakan TensorFlow Lite Interpreter. *Output* model berupa Detection Boxes, Detection Classes, Detection Scores, dan Number of Detections selanjutnya diterjemahkan menggunakan labels.txt dan divisualisasikan dalam bentuk *bounding box*, label huruf, serta *confidence score* pada halaman Detect secara *realtime*.

3. Hasil implementasi menunjukkan bahwa model MobileNet-SSD berhasil diintegrasikan ke dalam aplikasi Android dan mampu melakukan deteksi bahasa isyarat secara langsung pada perangkat (*on-device inference*) tanpa memerlukan koneksi internet. Berdasarkan hasil pengujian, seluruh fitur aplikasi berjalan sesuai kebutuhan fungsional melalui Black Box Testing, sedangkan pengujian performa *realtime* menghasilkan rata-rata waktu inferensi sebesar 340,6 ms atau sekitar 2 FPS sampai 3 FPS. Pengujian akurasi pada kondisi penggunaan nyata menghasilkan akurasi (*micro average*) sebesar 85,58%, dengan *precision* sebesar 87,61%, *recall* sebesar 85,58%, dan *F1-score* sebesar 85,75%. Selain itu, hasil *User Acceptance Testing* (UAT) menunjukkan bahwa aplikasi dapat diterima dengan baik oleh pengguna, sehingga dapat disimpulkan bahwa hasil pemodelan Deep Learning MobileNet-SSD telah berhasil diintegrasikan ke dalam aplikasi Android untuk mendeteksi dan mengklasifikasikan bahasa isyarat secara *realtime* sesuai dengan rumusan masalah penelitian.
4. Berdasarkan hasil pengujian, model dan aplikasi yang dikembangkan masih memiliki beberapa batasan. Model dirancang untuk mengenali 24 huruf SIBI statis (A–Y, kecuali J dan Z) sehingga belum dapat mendeteksi gestur dinamis maupun kosakata bahasa isyarat secara utuh. Selain itu, performa deteksi dipengaruhi oleh kondisi lingkungan, seperti intensitas pencahayaan, jarak pengambilan citra, sudut tangan, serta kompleksitas latar belakang, sehingga tingkat akurasi dapat menurun pada kondisi yang kurang optimal. Model juga hanya dilatih menggunakan satu objek tangan pada setiap citra, sehingga belum mampu melakukan *multiple object detection* secara bersamaan. Dari sisi implementasi, aplikasi hanya tersedia pada perangkat Android dan menghasilkan rata-rata waktu inferensi sebesar 340,6 ms (sekitar 2-3 FPS), sehingga meskipun mampu melakukan deteksi secara *realtime*, kelancaran tampilan masih bergantung pada spesifikasi perangkat yang digunakan.

5.2. Saran

Saran yang dapat dilakukan untuk penelitian yang mengembangkan penelitian ini selanjutnya, yaitu :

1. Dalam pengembangan model *Deep Learning* selanjutnya dapat dilakukan penambahan jumlah dan variasi *dataset*, seperti variasi pencahayaan, latar belakang, jarak pengambilan citra, sudut tangan, dan orientasi tangan. Selain itu, penerapan teknik *data augmentation*, seperti rotasi, perubahan tingkat kecerahan (*brightness*), penyesuaian kontras (*contrast*), dan pembesaran atau pengecilan objek (*scaling*), dapat dilakukan untuk meningkatkan kemampuan model dalam mengenali bahasa isyarat pada berbagai kondisi.
2. Dalam pengembangan metode deteksi selanjutnya dapat dilakukan perbandingan performa MobileNet-SSD dengan metode deteksi objek lainnya, seperti YOLO atau EfficientDet, sehingga dapat diketahui metode yang memberikan keseimbangan terbaik antara akurasi deteksi, kecepatan inferensi, dan ukuran model untuk diimplementasikan pada perangkat Android.
3. Guna meningkatkan performa deteksi, peneliti dapat menambahkan variasi data latih saat proses *training* model, seperti penggunaan citra yang memuat lebih dari satu objek tangan (*multiple object detection*), serta meningkatkan kualitas citra dan jumlah sampel pada setiap kelas. Selain itu, proses *training* dapat dioptimalkan melalui penyesuaian parameter *training*, seperti jumlah *training steps, batch size*, dan *learning rate*, sehingga model yang dihasilkan memiliki performa deteksi yang lebih optimal dan stabil pada berbagai kondisi penggunaan.