BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Berikut adalah beberapa tahapan yang dilakukan dalam penelitian ini:

1. Kajian Literatur
   Tahap awal dilakukan dengan mengkaji berbagai sumber ilmiah seperti jurnal internasional, buku teks, serta artikel daring terpercaya yang relevan dengan topik Klasifikasi OSCC, metode Convolutional Neural Network (CNN), dan arsitektur ResNet-152. Kajian ini bertujuan untuk memperkuat landasan teori, memahami pendekatan yang telah ada, serta merumuskan strategi metodologi yang tepat.

2. Akuisisi dan Persiapan Data
   Dataset yang digunakan dalam penelitian ini merupakan kumpulan citra histopatologi OSCC yang tersedia di platform Kaggle[1]. *Dataset* tersebut telah melalui proses augmentasi oleh pemiliknya dan terdiri dari dua kelas utama, yaitu jaringan normal dan jaringan yang menunjukkan indikasi OSCC. *Dataset* kemudian diproses lebih lanjut untuk pembagian menjadi data pelatihan, validasi, dan pengujian.

3. Perancangan Sistem dan Konfigurasi Eksperimen
   Perancangan sistem meliputi konfigurasi parameter pelatihan seperti ukuran batch, jumlah epoch, learning rate, serta strategi fine-tuning yang digunakan. Arsitektur dasar yang dipilih adalah ResNet-152, dengan strategi fine-tuning dimulai dari layer `conv5_block1_1_conv`. Selain itu, digunakan pendekatan 5-fold cross-validation untuk memastikan model tidak bergantung pada subset data tertentu.

4. Implementasi Model Klasifikasi
   Implementasi dilakukan menggunakan bahasa pemrograman Python dengan bantuan library TensorFlow/Keras. Transfer learning diterapkan menggunakan bobot awal dari model ResNet-152 yang telah dilatih pada ImageNet. Proses pelatihan terdiri dari dua tahap: feature extraction dan fine-tuning. Optimizer yang digunakan adalah AdamW, dengan

pembandingan tambahan terhadap model ResNet101 untuk beberapa eksperimen. Augmentasi data juga diterapkan untuk meningkatkan generalisasi model.

5. Evaluasi dan Validasi Model
   Evaluasi model dilakukan secara kuantitatif menggunakan metrik seperti akurasi, precision, recall, F1-score, dan AUC. Selain itu, confusion matrix digunakan untuk menggambarkan distribusi klasifikasi secara visual. Evaluasi dilakukan pada setiap fold dari 5-fold cross-validation, serta pada data uji akhir untuk mengetahui performa generalisasi model terbaik.

6. Penyusunan Laporan dan Dokumentasi Penelitian
   Seluruh proses penelitian mulai dari kajian awal hingga evaluasi akhir didokumentasikan secara sistematis dalam bentuk laporan skripsi. Penyusunan laporan dilakukan secara bertahap dan disesuaikan dengan pedoman penulisan yang berlaku di Universitas Multimedia Nusantara, guna memastikan keterlacakan proses dan keabsahan hasil.

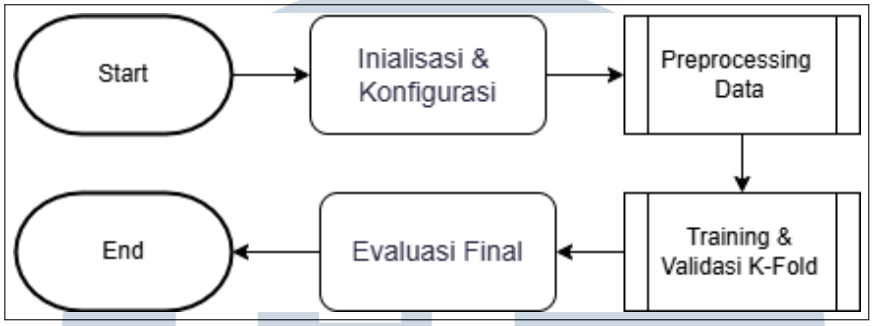
3.2 Perancangan Sistem

Perancangan program ini terdiri dari tiga komponen utama yang saling terintegrasi, yaitu flowchart model klasifikasi, flowchart website, dan desain antarmuka pengguna.

3.2.1 Model Deteksi

Gambar 3.1 menunjukkan alur sistem dalam pengembangan model deteksi menggunakan pendekatan CNN. Proses diawali dari tahapan *Start*, kemudian dilanjutkan dengan Import *Library* yang memuat semua dependensi eksternal yang dibutuhkan. Setelah itu dilakukan *Variable Setting* untuk mendefinisikan parameter-parameter penting seperti ukuran gambar, *batch size*, jumlah *epoch*, dan lain-lain. Data kemudian diproses pada tahap *Preprocessing Data* yang mencakup pembagian data, *augmentasi*, dan *resize* gambar. Data hasil *preprocessing* akan digunakan untuk membangun arsitektur Model, yang kemudian divalidasi menggunakan metode K-Fold *Validation* guna menguji performa model secara menyeluruh dan menghindari *overfitting*. Setelah validasi selesai, model terbaik akan di latih ulang secara penuh pada tahap *Train*,lalu diuji performanya menggunakan data uji(Test).

Hasil akhir dari proses pengujian ini akan menjadi evaluasi keseluruhan sistem yang ditutup dengan tahapan *End*.



Gambar 3.1. Flowchart sistem

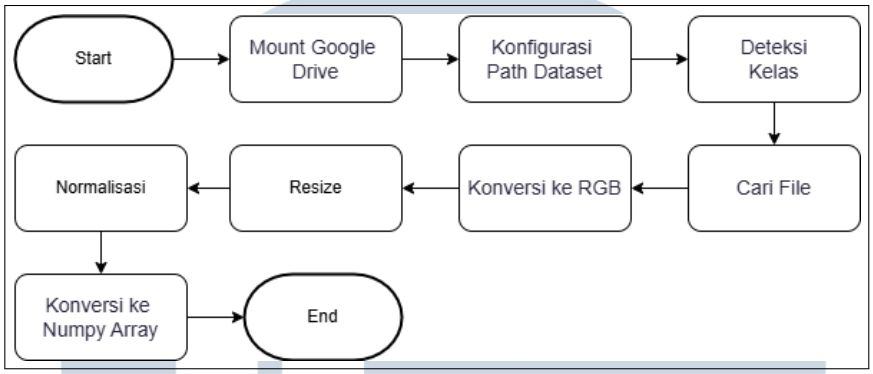
A Inisialisasi dan Konfigurasi

Tahap awal dari sistem dimulai dengan proses inisialisasi dan konfigurasi lingkungan eksperimen. Pada tahap ini, dilakukan *import* seluruh pustaka dan dependensi yang dibutuhkan seperti *TensorFlow, Keras, NumPy, matplotlib, dan pandas*. Kemudian dilakukan juga penentuan parameter pelatihan seperti ukuran *batch*, jumlah *epoch*, *learning rate*, serta inisialisasi nilai acak (*seed*) agar eksperimen dapat direproduksi. Selain itu, struktur direktori untuk menyimpan model, log, dan hasil visualisasi juga ditentukan. Arsitektur dasar ResNet-152 dengan bobot pralatih dari ImageNet dimuat dan dipersiapkan untuk proses *transfer learning*.

B Pemuatan dan preprocessing data

Langkah kedua adalah *sub-proses* data yang mencakup pemuatan *dataset* dan *pra-pemrosesan* pada gambar 3.2. *Dataset* yang digunakan berasal dari platform Kaggle yang dipublikasikan oleh Ashenafi Fasil Kebede, berisi *citra histopatologi Oral Squamous Cell Carcinoma* (OSCC) yang terdiri dari dua kelas utama: jaringan normal dan jaringan terindikasi OSCC. Setiap citra kemudian dikonversi ke format RGB dan diubah ukurannya menjadi 224x224 piksel agar sesuai dengan arsitektur masukan model ResNet-152. Proses normalisasi dilakukan menggunakan fungsi pra-pemrosesan bawaan dari ResNet untuk menyesuaikan skala nilai piksel, sementara label kelas dikonversi ke format numerik melalui metode label *encoding*. Selain itu, untuk meningkatkan keragaman data dan kemampuan generalisasi model, diterapkan pula augmentasi tambahan selama

pelatihan, seperti rotasi acak, pergeseran posisi, pembesaran (*zooming*), dan pencerminan horizontal (*horizontal flip*).



Gambar 3.2. Flowchart sistem

C Training dan validasi K-Fold

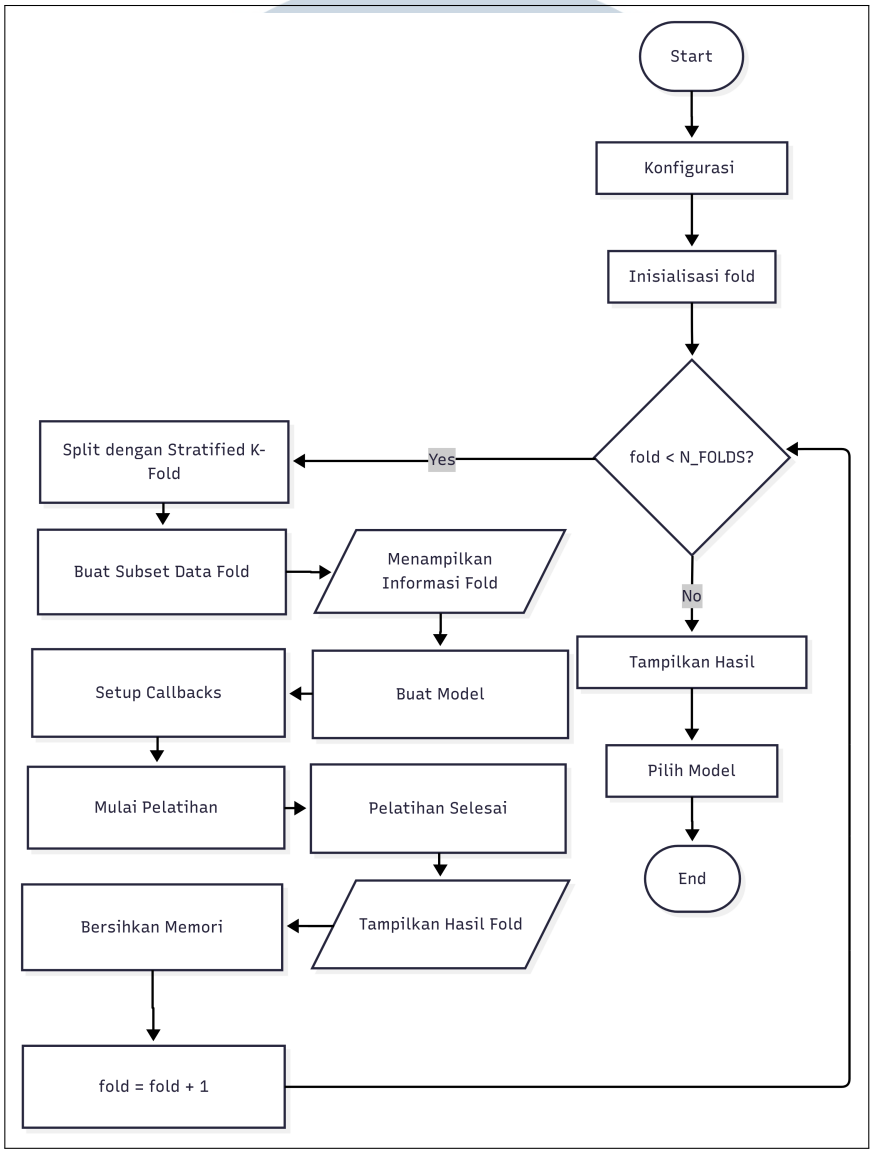
Proses pelatihan model dilakukan dengan pendekatan *Stratified K-Fold Cross Validation* sebanyak lima lipatan (*5-fold*) untuk memastikan model tidak hanya belajar dari satu *subset* data saja, sehingga meningkatkan kemampuan generalisasi. Tahapan ini dimulai dari konfigurasi awal seperti menetapkan jumlah *fold*, *seed random*, *callbacks*, dan direktori penyimpanan model. Setelah itu, proses iteratif dimulai dengan inisialisasi nilai *fold* = 0.

Untuk setiap iterasi, dilakukan pemisahan data menggunakan *StratifiedKFold*, sehingga distribusi kelas tetap seimbang pada tiap *fold*. Kemudian dibuat *subset* data untuk pelatihan dan validasi berdasarkan indeks *fold* tersebut, seperti yang diilustrasikan pada Gambar 3.3. Informasi *fold* seperti jumlah data dan distribusi kelas ditampilkan untuk keperluan *monitoring*.

Selanjutnya, dilakukan pembuatan model ResNet-152 sesuai konfigurasi awal. *Callbacks* seperti *EarlyStopping*, *ReduceLROnPlateau*, dan *ModelCheckpoint* juga diatur ulang pada tiap *fold* agar hanya model terbaik yang disimpan. Setelah itu, proses pelatihan dilakukan, baik pada tahap *feature extraction* maupun *fine-tuning*, tergantung konfigurasi. Augmentasi citra dan *class weight* digunakan untuk menangani ketidakseimbangan data dan meningkatkan keragaman. Setelah pelatihan selesai pada satu *fold*, model dievaluasi menggunakan data validasi dan hasil metrik disimpan. Memori kemudian dibersihkan dan nilai *fold* ditingkatkan satu, sebelum melanjutkan ke iterasi berikutnya.

Jika seluruh *fold* telah diproses (*fold* tidak kurang dari *N FOLDS*), maka

hasil ringkasan dari seluruh *fold* ditampilkan. Model terbaik kemudian dipilih berdasarkan metrik evaluasi seperti *AUC*, dan proses pelatihan dinyatakan selesai.



Gambar 3.3. Flowchart training dan validasi K-Fold

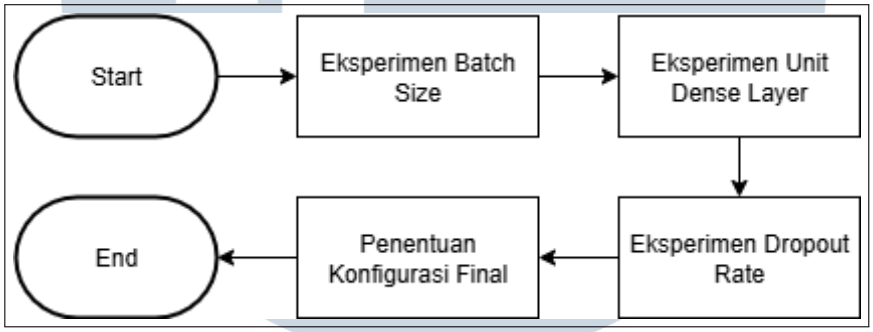
D Evaluasi final dan penyimpanan hasil

Setelah model terbaik dari proses *cross-validation* diperoleh, dilakukan evaluasi akhir terhadap data uji yang telah dipisahkan sebelumnya. Model terbaik dimuat dari file *checkpoint* dan dievaluasi menggunakan metrik seperti akurasi, *precision*, *recall*, *F1-score*, dan *AUC*. Selain itu, *confusion matrix* digunakan untuk memberikan gambaran visual performa klasifikasi. Hasil evaluasi ini disimpan

dalam bentuk grafik dan tabel untuk keperluan dokumentasi. Model akhir kemudian disimpan dalam format `.keras` agar dapat digunakan dalam sistem deteksi berbasis *web*. Semua proses, grafik pelatihan, hasil evaluasi, dan konfigurasi disimpan untuk keperluan pelaporan dan dokumentasi penelitian.

3.2.2 Eksperimen dan Perbandingan Model

Untuk menemukan konfigurasi arsitektur yang paling optimal, penelitian ini menerapkan serangkaian eksperimen yang dirancang secara sistematis. Alur kerja dari keseluruhan proses eksperimen ini diilustrasikan dalam Gambar 3.4.



Gambar 3.4. Flowchart alur eksperimen perbandingan model

Gambar 3.4 di atas menunjukkan bahwa proses pencarian *hyperparameter* terbaik dilakukan secara sekuensial. Tahap pertama adalah melakukan eksperimen *batch size* untuk menentukan ukuran *batch* yang paling efektif. Setelah nilai *batch size* terbaik diperoleh, nilai tersebut digunakan sebagai dasar untuk tahap selanjutnya, yaitu eksperimen unit *dense layer*. Proses ini bertujuan mencari kompleksitas arsitektur yang seimbang. Terakhir, dengan *batch size* dan unit *dense layer* terbaik, dilakukan eksperimen *dropout rate* untuk menemukan tingkat regularisasi yang paling optimal dalam mencegah *overfitting*.

Setelah ketiga tahapan eksperimen tersebut selesai, dilakukan penentuan konfigurasi final dengan menggabungkan hasil-hasil terbaik dari setiap perbandingan. Model dengan konfigurasi final inilah yang kemudian digunakan dalam proses evaluasi akhir terhadap data uji untuk mengukur performa generalisasi secara keseluruhan, yang hasilnya akan dibahas pada Bab 4.