# BAB 3

# METODOLOGI PENELITIAN

## 3.1 Metodologi Penelitian

Penelitian ini dilaksanakan dalam beberapa tahap sebagai berikut.

1. Studi Literatur

   Pada studi literatur, dilakukan pencarian dan pembelajaran terhadap pengolahan citra, konsep dasar algoritma *convolutional neural network*, dan menelusuri jurnal dan karya tulis ilmiah yang berkaitan dengan penyakit kanker kulit.

2. Pengumpulan Data

   *Dataset* diperoleh dari *The International Skin Imaging Collaboration* (ISIC). *Dataset* didapatkan melalui media internet, dan dapat ditelusuri pada *website* ISIC (ISIC, 2016). Data dibagi menjadi tiga kelas yaitu, *melamona*, *squamosa cell carcinoma*, dan *basal cell carcinoma*. Kelas *melamona* mempunyai jumlah data sebanyak 968 data, kelas *basal cell carcinoma* sebanyak 818 data, dan kelas *squamosa cell carcinoma* sebanyak 633 data.

3. Perancangan Program

   Pada tahap ini, dilakukan perancangan sistem seperti pembuatan *flowchart,* dan rancangan antarmuka *website*.

4. Pembuatan program

Pada tahapan pembuatan program, dilakukan implementasi berdasarkan rancangan program yang telah dilakukan sebelumnya. Implementasi menggunakan bahasa pemograman Python untuk *preprocessing* data dan pembuatan model *machine learning*.

5. Pengujian dan evaluasi

   Pada tahapan ini, dilakukan pengujian dan evaluasi pada model yang telah dibuat. Proses pengujian menggunakan data citra yang berbeda dengan proses pelatihan. Penghitungan *precision*, *recall*, dan *f1-score* dilakukan pada tahap evaluasi.

6. Penulisan Laporan

   Selama pengerjaan penelitian ini penulisan laporan dilakukan secara parallel sembari melakukan penelitian, sehingga tidak ada proses yang terlewatkan untuk dituliskan dalam laporan.

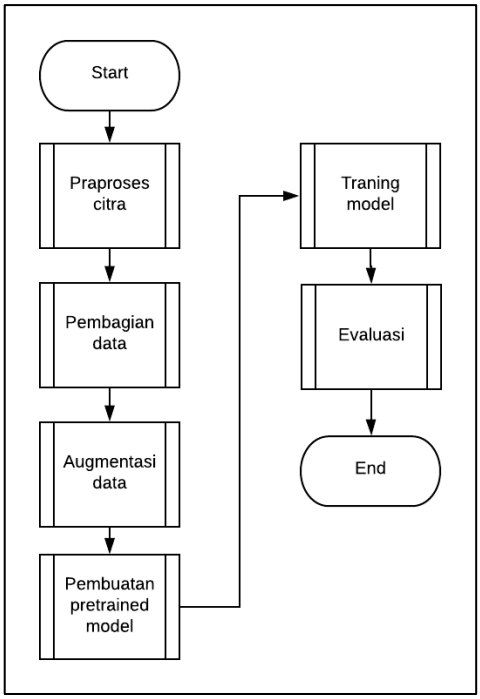
## 3.2 Perancangan Program

Perancangan program dibagi menjadi tiga bagian utama, yaitu *flowchart all pipeline* atau alur keseluruhan program yang mencakup praproses citra hingga evaluasi data *testing*, *flowchart* aplikasi *website* yang dibuat, dan desain antarmuka yang digunakan sebagai desain awal pada aplikasi *website*.

### 3.2.1 Flowchart All Pipeline

Pada Gambar 3.1 merupakan *flowchart* yang digunakan sebagai perancangan pembuatan keseluruhan program. Proses dibagi menjadi enam bagian

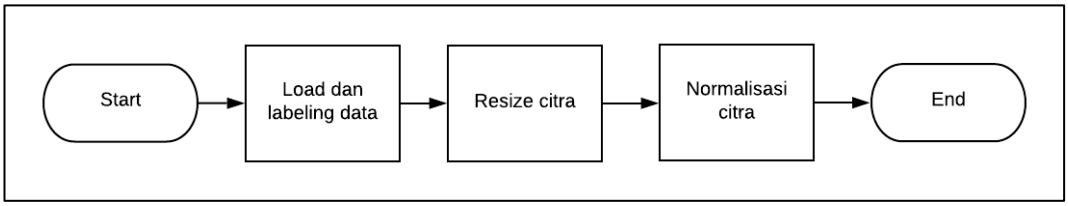
utama, yaitu praproses citra, pembagian data, augmentasi data, pembuatan *pretrained model*, *training model*, dan evaluasi model.




Gambar 3.1 *Flowchart* program secara keseluruhan

## A. Flowchart Praproses Citra

Praproses citra dilakukan pada data dengan tujuan untuk mempermudah dan mempercepat proses pelatihan model terhadap data tersebut. Terdapat tiga praproses citra yang dilakukan yaitu, *load* dan *labelling data*, *resize* citra dan normalisasi citra. Berikut merupakan bagan alur praproses citra.

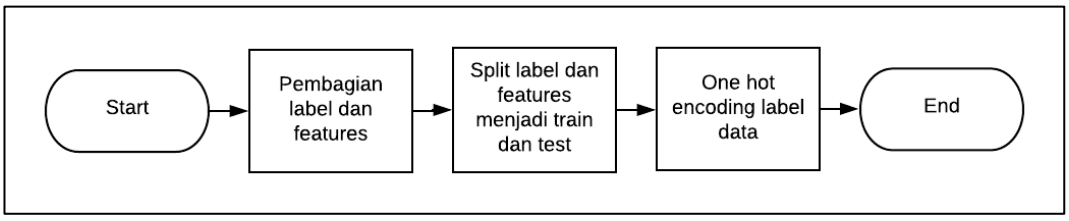



Gambar 3.2 *Flowchart* praproses citra

Proses dimulai dari pengambilan data citra dari direktori dan memberi label kepada. Kemudian ukuran citra tersebut di *resize* sesuai dengan *input* dari arsitektur ResNet50 yaitu sebesar 224 x 224 *pixel*. Selanjutnya proses normalisasi yang dilakukan dengan membuat citra yang awalnya memiliki interval 0 sampai 255 menjadi 0 sampai 1. Normalisasi citra dilakukan karena dalam proses pembelajaran *neural network* memakai *weights* yang berukuran kecil sehingga *input* citra yang berukuran besar dapat mengganggu atau memperlambat proses pelatihan (Brownlee, 2019).

## B. Flowchart Pembagian Data

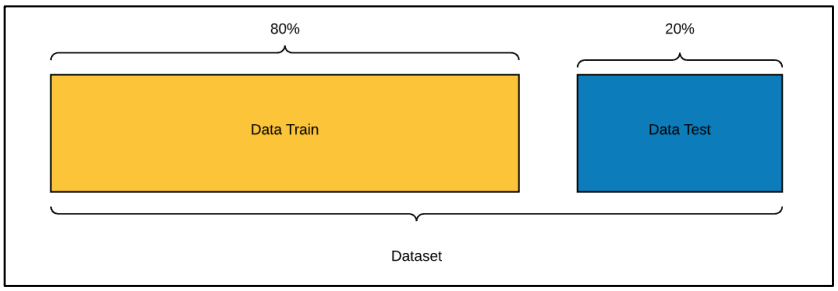
Untuk sebuah model dapat melakukan prediksi, diperlukan proses pembagian *dataset*. Pada Gambar 3.3 menunjukkan proses pembagian *dataset*. Proses diawali dengan pembagian label dan *features* data. Label digunakan sebagai kelas dari data citra, sedangkan *features* digunakan sebagai isi data citra.




Gambar 3.3 *Flowchart* pembagian data

Selanjutnya *dataset* dibagi menjadi *data train,* dan *data test*. *Data train* digunakan untuk melakukan pelatihan model yang dibuat. *Data test* digunakan untuk menguji performa dari model. Pada penelitian ini, jumlah rasio yang digunakan adalah 80:20 untuk *data train* dan *data test* sesuai dengan gambar yang

ditujukkan pada Gambar 3.4. Kemudian data label diubah menjadi bentuk *one-hot encoder* atau dalam bentuk vektor biner. Tujuan mengubah data label menjadi bentuk vektor biner karena *machine learning* tidak bisa bekerja dengan menggunakan data jenis *categorical*.



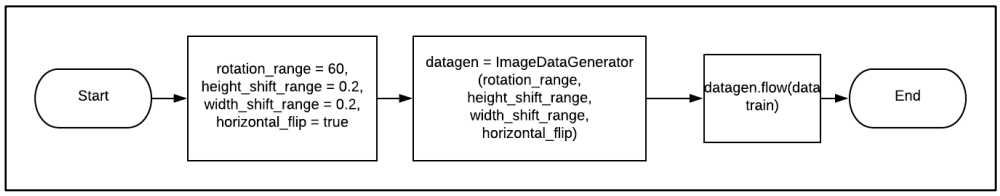

Gambar 3.4 Pembagian *dataset*

Dalam proses pembagian data, terdapat fungsi yang digunakan yaitu, train-_test_split yang berasal dari *library* sklearn. Pada fungsi ini terdapat empat parameter yang digunakan dalam proses pembagian data, yaitu train_size yang merupakan proporsi ukuran *data train*, test_size yang merupakan proporsi ukuran *data test*, *shuffle* yang digunakan untuk mengacak data, dan *random state* yang digunakan sebagai angka acuan dari proses pengacakan data. Pada penelitian ini, parameter yang digunakan adalah *test size* sebesar 0.2 dan *train size* sebesar 0.8. Untuk parameter *shuffle* digunakan untuk mengacak data dengan *random state* bernilai 42.

## C. Flowchart Augmentasi

Augmentasi data adalah sebuah teknik yang digunakan untuk memperbanyak kumpulan *data train* secara artifisial dengan membuat versi gambar yang dimodifikasi dalam kumpulan data tersebut. Tujuan dari augmentasi data

adalah membuat model lebih mahir dan mencegah terjadinya *overfitting* dikarenakan model tidak akan menemukan data yang sama terus menerus. Augmentasi data ini dilakukan untuk mencegah *overfitting* terjadi pada model (Chollet, 2016). *Flowchart* dari augmentasi ditujukkan pada Gambar 3.5.



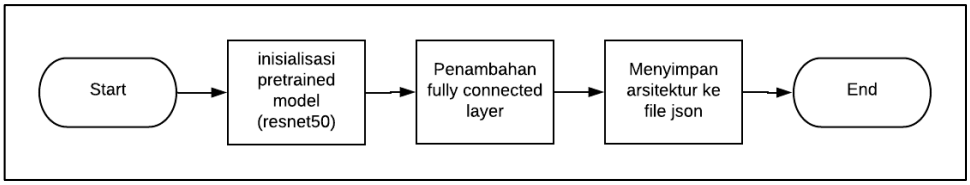

Gambar 3.5 *Flowchart* augmentasi data

Proses augmentasi diawali dengan inisialisasi ImageDataGenerator yang merupakan fungsi dari *library* Keras. Augmentasi menggunakan beberapa parameter, yaitu rotation_range sebesar 60 derajat, width_shift_range sebesar 0.2, height_shift_range sebesar 0.2, dan horizontal_flip. Parameter rotation range bertujuan untuk membuat rotasi acak kepada citra, horizontal_flip bertujuan membalikkan citra secara horizontal, height_shift_range dan width_shift_range bertujuan untuk melakukan transalasi secara horizontal dan vertikal. Kemudian, method *flow* digunakan untuk melakukan augmentasi terhadap *data train*. Hasil dari *flow* digunakan untuk proses pelatihan model.

## D. Flowchart Pembuatan Pretrained Model

Pada proses ini, hal pertama yang dilakukan adalah melakukan inisialisasi *pretrained model*. *Pretrained model* yang digunakan pada penelitian ini adalah ResNet50. Setelah melakukan inisialisasi, model ditambah dengan *fully connected layer* dikarenakan penelitian ini menggunakan kelas berjumlah tiga. Terakhir

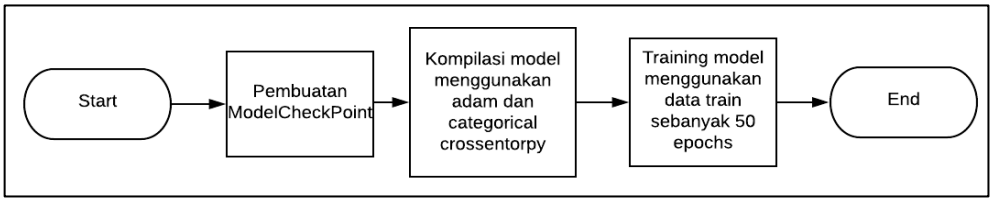
arsitektur akan disimpan pada *file json* Alur proses pembuatan *pretrained model* ditujukkan pada Gambar 3.6.




Gambar 3.6 *Flowchart* pembuatan *pretrained model*

## E. Flowchart Training Model

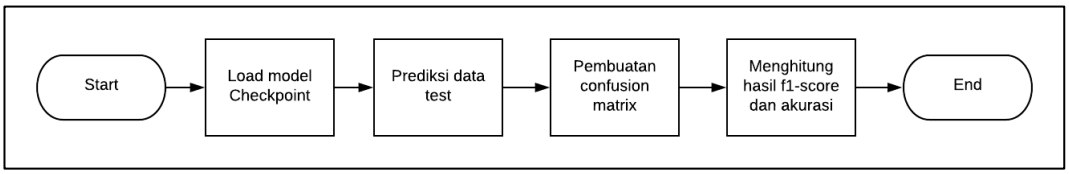
Pada Gambar 3.7 merepresentasikan alur proses *training model*. Proses dimulai dengan pembuatan ModelCheckPoint yang digunakan untuk menyimpan *weights* dengan akurasi terbaik ke dalam *format file* h5 saat proses *training*. Kemudian, model dikompilasi dengan menggunakan *categorical cross entropy* sebagai *loss function* dan Adam sebagai *optimizer*. *Categorical cross entropy* merupakan metode untuk menghitung nilai *loss* pada *multi-class classification* yang menggunakan label bentuk *one-hot-encode* (Gomez, 2018). Adam merupakan salah satu metode optimisasi *gradient descent* yang menggunakan dua momentum dalam melakukan perhitungan *learning rate* secara lebih adaptif (Kingma dan Ba, 2015). Terakhir, model di *training* menggunakan *data train* yang sudah diaugmentasi sebanyak 50 *epochs*.




Gambar 3.7 *Flowchart training model*

## F. Flowchart Evaluasi

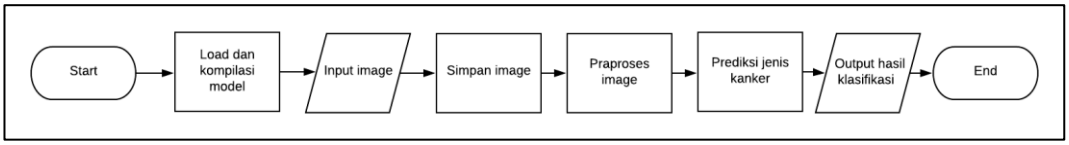
Untuk menguji sebuah model berjalan dengan baik maka diperlukan proses evaluasi model. Pada penelitian ini, model dievaluasi dengan menghitung nilai akurasi dan *f1-score*. Kedua nilai tersebut diperoleh dari *confusion matrix* hasil prediksi yang dilakukan pada *data test* menggunakan model yang disimpan oleh ModelCheckPoint pada proses sebelumnya. Pada Gambar 3.8 menunjukkan proses dari evaluasi terhadap model yang telah dibuat.




Gambar 3.8 *Flowchart* evaluasi

### 3.2.2 Flowchart Website

Sistem klasifikasi jenis kanker kulit dengan *convolutional neural network* memiliki perancangan dalam pembuatan *website*. Perancangan *website* yang digunakan adalah *flowchart*. Pada Gambar 3.9 merupakan *flowchart* dari pembuatan *website*.



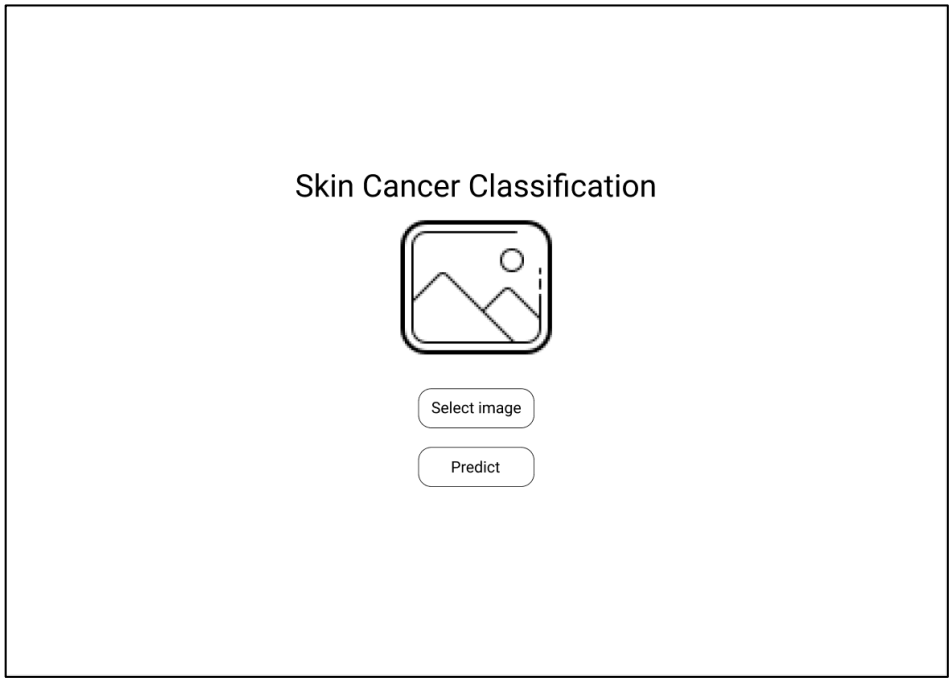

Gambar 3.9 *Flowchart* aplikasi

Proses awal dimulai dari *website* melakukan *load* dan kompilasi model yang sudah dilatih pada proses *training*. Pada *website* ini pengguna memasukkan citra

yang memilki *format file* jpg, jpeg, dan png. Selanjutnya *website* menyimpan citra tersebut ke dalam *folder upload* yang berada pada *server*. Kemudian citra dipraproses dengan melakukan *resize* dan normalisasi citra. *Website* melakukan prediksi dengan model yang telah dibuat terhadap gambar yang telah dimasukkan sebelumnya. Terakhir *website* menampilkan hasil akurasi prediksi jenis kanker kulit yang dilakukan.

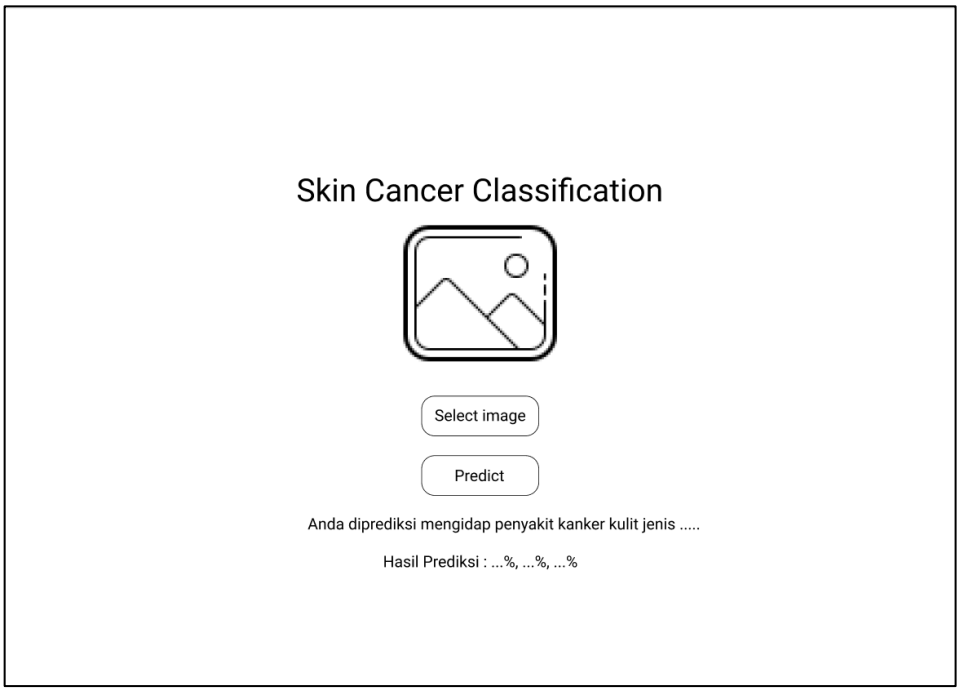
### 3.2.3 Desain Antarmuka

Pada halaman *website* akan ditampilkan dua tombol, yaitu tombol untuk memilih gambar yang ingin di klasifikasi dan tombol prediksi untuk mengirim data citra tersebut seperti yang ditujukkan pada Gambar 3.10.




Gambar 3.10 Halaman *website*

Setelah citra berhasil di kirim, *server* akan melakukan prediksi dengan menggunakan model yang sudah dibuat dan mengembalikan hasil prediksi berupa kelas prediksi dari gambar, dan probabilitas *score* untuk setiap kelas seperti yang ditujukkan pada Gambar 3.11.




Gambar 3.11 Halaman *website* setelah citra di *upload*