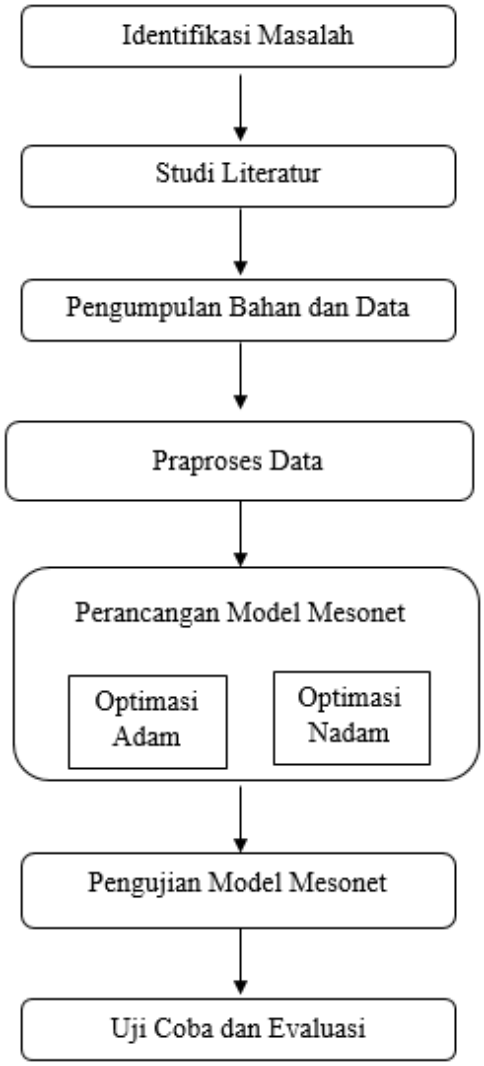
BAB III

METODOLOGI PENELITIAN

Bab ini menguraikan bagaimana penelitian ini dilakukan sehingga dipaparkan secara rinci proses dan langkah-langkah yang dilakukan secara sistematis dan dapat digunakan dengan jelas untuk memecahkan masalah dan menganalisis hasil penelitian. Tahapan penelitian dilihat pada gambar 3.1.



Gambar 3. 1 Tahapan Penelitian

3.1. Identifikasi Masalah

Pada tahap identifikasi masalah dilakukan penentuan objek pada penelitian yang akan dilakukan, yaitu melakukan komparasi terhadap optimasi adam dan nadam pada dataset Celeb DF v2 menggunakan arsitektur MesoNet.

3.2. Studi Literatur

Setelah mengidentifikasi masalah dilakukan selanjutnya studi literatur, pada tahap ini untuk meninjau penelitian-penelitian sebelumnya yang relevan, teori-teori yang mendukung, dan metode-metode yang telah digunakan, guna memperoleh dasar yang kuat dalam menyusun pendekatan penelitian.

3.3. Pengumpulan Bahan dan Data

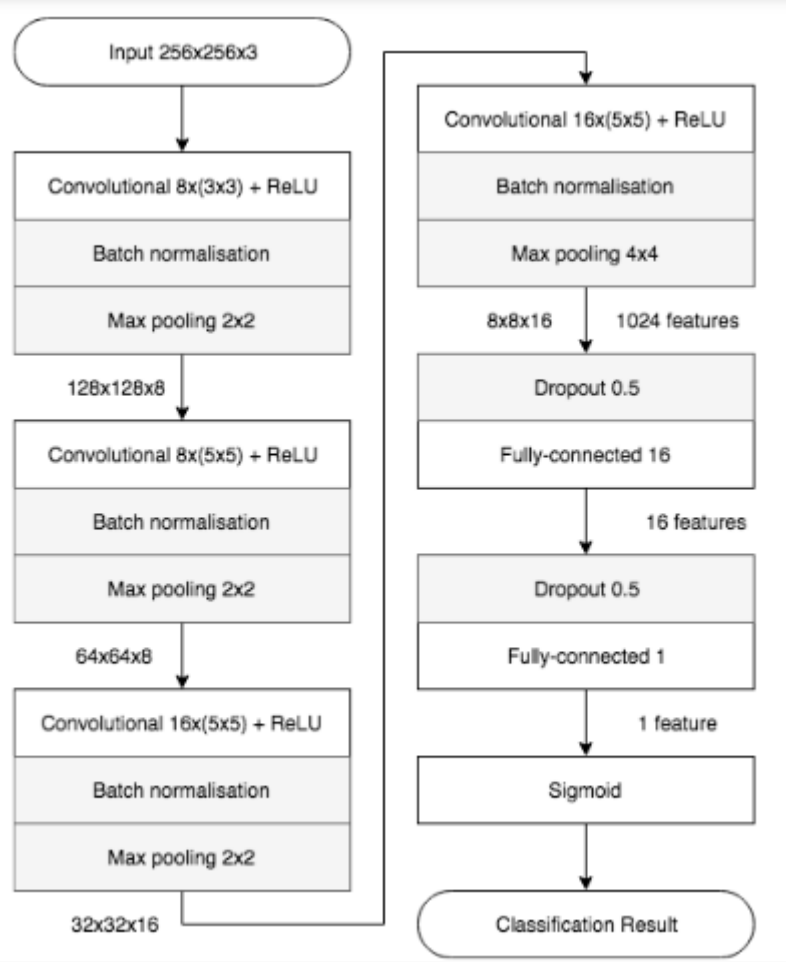
Data yang digunakan dalam penelitian ini adalah data sekunder yang diperoleh di laman https://www.kaggle.com/datasets/mahabuburrahman/celeba. Celeb DF v2 dataset memiliki ukuran 1.70 GB. Celeb DF v2 memiliki folder *Real and Fake* berjumlah 2.040, serta folder lain seperti 10.905 folder *Test*, 140.002 folder *Train*, dan *Validation* berjumlah 39.428. Setiap folder memiliki folder *Fake* dan *Real*.

3.4. Praproses Data

Dalam penelitian yang mengutamakan klasifikasi data, praproses data adalah langkah penting yang dilakukan. Praproses data dilakukan sebelum klasifikasi menggunakan MesoNet dengan memberikan label pada set data dan menyesuaikan dimensi gambar agar seragam untuk memastikan kualitasnya valid dan baik. Data yang telah diproses terdiri dari dua bagian: data pelatihan (*training*) dan data validasi. Data pelatihan digunakan untuk proses pelatihan data pada klasifikasi.

3.5. Perancangan Arsitektur MesoNet

Perancangan arsitektur MesoNet dilakukan dengan membangun model Meso4 adalah tahapan untuk menyusun sebuah model yang akan digunakan dalam proses klasifikasi.



Gambar 3. 2 Arsitektur MesoNet

Pada gambar 3.2 adalah alur sebuah arsitektur MesoNet dimulai dengan lapisan input yang menerima gambar berukuran 256x256 piksel dengan 3 channel warna (RGB). Gambar ini kemudian diproses melalui lapisan konvolusi pertama yang

menggunakan 8 filter dengan ukuran kernel 3x3, diikuti oleh fungsi aktivasi ReLU untuk memperkenalkan non-linearitas ke dalam model. Setelah aktivasi, normalisasi *batch* diterapkan untuk stabilisasi dan mempercepat proses pelatihan, diikuti oleh *max pooling* dengan ukuran 2x2 untuk mengurangi dimensi spasial dari gambar.

Proses ini dilanjutkan dengan lapisan konvolusi kedua yang menggunakan 8 filter dengan ukuran kernel 5x5. Lapisan ini juga diikuti oleh fungsi aktivasi ReLU, normalisasi *batch*, dan *max pooling* dengan ukuran 2x2, yang bertujuan untuk lebih lanjut mengekstraksi fitur-fitur penting dari gambar.

Kemudian, lapisan konvolusi ketiga menggunakan 16 filter dengan ukuran kernel 5x5, diikuti oleh aktivasi ReLU, normalisasi *batch*, dan *max pooling* dengan ukuran 2x2. Lapisan ini bertujuan untuk mengekstraksi fitur yang lebih kompleks dan lebih spesifik dari gambar. Hasil dari lapisan konvolusi dan *pooling* ini kemudian diratakan menjadi vektor fitur dengan ukuran 32x32x16, atau 16384 fitur.

Vektor fitur ini kemudian diproses melalui serangkaian lapisan *fully connected*. Pertama, sebuah lapisan konvolusi dengan 16 filter dan ukuran kernel 5x5 diaplikasikan, diikuti oleh aktivasi ReLU dan *normalisasi batch*. Setelah itu, *max pooling* dengan ukuran 4x4 diterapkan, menghasilkan 1024 fitur. Lapisan ini diikuti oleh lapisan *dropout* dengan rate 0.5 untuk mencegah *overfitting*.

Fitur-fitur ini kemudian dilewatkan melalui lapisan fully connected dengan 16 neuron, diikuti oleh lapisan dropout dengan rate 0.5, dan akhirnya lapisan *fully connected* dengan 1 neuron. Fungsi aktivasi *sigmoid* diterapkan pada lapisan terakhir untuk menghasilkan output akhir, yang merupakan nilai probabilitas antara 0 dan 1.

Nilai ini menunjukkan apakah gambar tersebut merupakan deepfake atau bukan, memberikan hasil klasifikasi akhir dari model ini. Arsitektur ini dirancang dengan hati-hati untuk mengekstraksi fitur yang efektif dari gambar input dan melakukan klasifikasi *deepfake* dengan akurasi tinggi.

Setelah mendefinisikan kelas Meso4, langkah berikutnya adalah menginisialisasi dua instance model Meso4 dengan menggunakan dua *optimizer* yang berbeda, yaitu Adam dan Nadam. Inisialisasi ini bertujuan untuk membandingkan performa kedua *optimizer* dalam mendeteksi deepfake pada dataset yang sama. Model yang telah diinisialisasi kemudian akan dilatih dan dievaluasi untuk mengamati perbedaan efektivitas dan efisiensi antara kedua *optimizer* tersebut.

3.6. Pengujian Model MesoNet

Setelah mendefinisikan kelas Meso4 dan menginisialisasi dua instance model Meso4 menggunakan dua metode optimasi yang berbeda, yaitu Adam dan Nadam langkah selanjutnya adalah, untuk diujikan pada data yang telah dibagi sebelumnya yaitu data *train* dan data *validation*. Proses pengujian model dilakukan dengan penambahan jumlah iterasi yang digunakan untuk melihat seluruh data yang ada pada dataset.

3.7. Uji Coba dan Evaluasi

Untuk mengevaluasi performa model klasifikasi, seringkali digunakan metode *Confusion Matrix* yang memungkinkan untuk menghitung akurasi, presisi, *recall*, f1-score dan *aggregation score* dari hasil prediksi model.

Akurasi berfungsi untuk menghitung jumlah ukuran proporsi dari klasifikasi yang benar dalam model dengan menggunakan rumus yang dapat dilihat pada persamaan 3.1.

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN}\ x\ 100\% \quad (3.1)$$

Presisi berfungsi untuk mengetahui perbdaningan jumlah klasifikasi yang benar dengan yang salah dengan menggunakan rumus yang dapat dilihat pada persamaan 3.2.

$$\text{Presisi} = \frac{TP}{TP+FP}\ x\ 100\% \quad (3.2)$$

Recall berfungsi untuk menghitung jumlah klasifikasi yang benar dengan melakukan perbandingan terhadap jumlah entri yang terlewat dengan menggunakan rumus yang dapat dilihat pada persamaan 3.3.

$$\textit{Recall} = \frac{TP}{TP+FN}\ x\ 100\% \quad (3.3)$$

F1-*score* berfungsi untuk menghitung rata – rata dari presisi dan *recall* sehingga dapat diketahui efektivitas dari model yang dibuat dengan menggunakan rumus yang dilihat pada persamaan 3.4.

$$\textit{F1-score} = 2\ x\ \frac{Presisi\ x\ Recall}{Presisi+Recall}\ x\ 100\% \quad (3.4)$$