BAB III
METODOLOGI PENELITIAN

3.1 Gambaran Umum

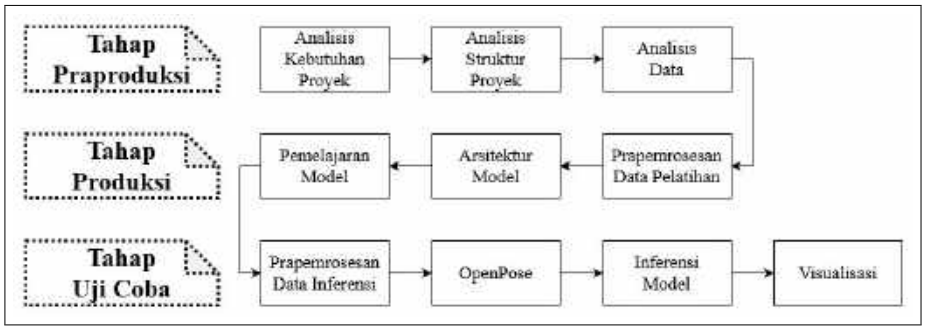
Penelitian ini membahas pemanfaatan data gambar sebagai acuan dalam melakukan pemelajaran dan implementasi model *deep neural network* untuk mencari dan memetakan koordinat tiga dimensi pose tubuh manusia dalam sebuah rangkaian gambar secara lokal. Pose yang digunakan tidak bersifat *grounded* yang berarti koordinat pose tidak berpusat pada titik lantai tertentu. Pengerjaan aplikasi mengutamakan dua langkah penting yang meliputi pengolahan data dan pembuatan model. Aplikasi yang dibuat dapat menampilkan plot grafik tiga dimensi menyerupai struktur anatomi tubuh manusia sesuai dengan pose hasil estimasi dari gambar masukkan. Hasil pemelajaran model ditampilkan dalam grafik dua dimensi untuk analisis lebih lanjut.

Dataset yang digunakan dalam penelitian ini terbagi menjadi dua jenis yang meliputi *dataset* pemelajaran model dan *dataset* inferensi aplikasi. *Dataset* pemelajaran model dikategorikan menjadi data pelatihan model dan data validasi model. *Dataset* pemelajaran model berisi gambar dan target posisi titik kunci anatomi dalam jumlah besar. Data pelatihan model adalah data yang digunakan dalam proses pelatihan sebagai sampel bagi *deep neural network*. Data validasi model adalah data yang digunakan untuk menguji kebenaran fungsionalitas pemetaan yang dipelajari saat pelatihan model. *Dataset* inferensi aplikasi adalah data uji coba berbentuk video tanpa target titik kunci yang digunakan untuk estimasi pose tubuh manusia secara sekuensial.

Pemelajaran model *deep neural network* diimplementasikan menggunakan *framework* PyTorch. Kedua *dataset* yang digunakan diolah terlebih dahulu sehingga memenuhi syarat PyTorch dalam melakukan *deep learning*. Setiap model kemudian digunakan terhadap *dataset* inferensi aplikasi. Proses dan hasil estimasi diurai lebih lanjut dalam bentuk grafik visual.

3.2 Kerangka Penelitian

Kerangka penelitian yang jelas dibutuhkan untuk memudahkan proses penelitian sehingga dapat mempersingkat waktu pengerjaan. Proses penelitian dibagi menjadi tiga tahapan besar yang meliputi tahap praproduksi, tahap produksi, dan tahap uji coba. Setiap tahapan tersebut dikerjakan secara terurut dan sistematis. Alur setiap tahap diilustrasikan pada gambar 3.1.



Gambar 3.1: Kerangka Penelitian

3.3 Tahap Praproduksi

Tahap praproduksi berisi langkah-langkah analisis yang menentukan alur pada tahap selanjutnya. Tahap praproduksi dibagi menjadi beberapa langkah yang meliputi analisis kebutuhan proyek, analisis struktur proyek, dan analisis data. Tahap ini menganalisis bagian-bagian pokok yang diperlukan sehingga mengetahui langkah-langkah yang akan dilakukan pada tahap produksi.

3.3.1 Analisis Kebutuhan Proyek

Pemelajaran dan implementasi model ini memerlukan alat-alat pendukung berupa perangkat keras dan perangkat lunak yang mencukupi. Spesifikasi perangkat keras dan perangkat lunak yang lebih besar akan mempercepat proses pemelajaran model jaringan saraf tiruan.

Spesifikasi perangkat keras yang digunakan dalam penelitian ini meliputi *central processing unit*, *graphics processing unit*, *random access memory*, *solid state drive*, dan *hard disk drive* dapat dilihat pada tabel 3.1.

Tabel 3.1: Spesifikasi Perangkat Keras

Perangkat Keras (Laptop)	
CPU	Intel Core I7 7700 HQ
GPU	NVIDIA GTX 1060 6 GB
RAM	24 GB DDR4
SSD	NVME SAMSUNG 120 GB
HDD	SATA 1 TB

Spesifikasi perangkat lunak yang digunakan dalam penelitian ini yang meliputi sistem operasi, *integrated development environment*, bahasa pemrograman, dan *deep learning framework* dapat dilihat pada tabel 3.2.

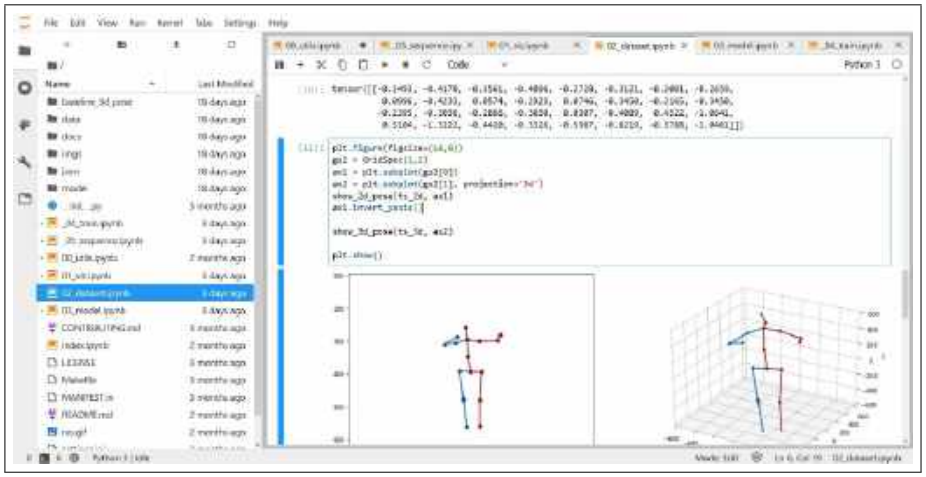
Tabel 3.2: Spesifikasi Perangkat Lunak

Perangkat Lunak	
Sistem Operasi	Ubuntu 19.10
IDE	Jupyter Lab
Bahasa Pemrograman	Python 3.7
Framework	PyTorch 1.4

3.3.2 Analisis Struktur Proyek

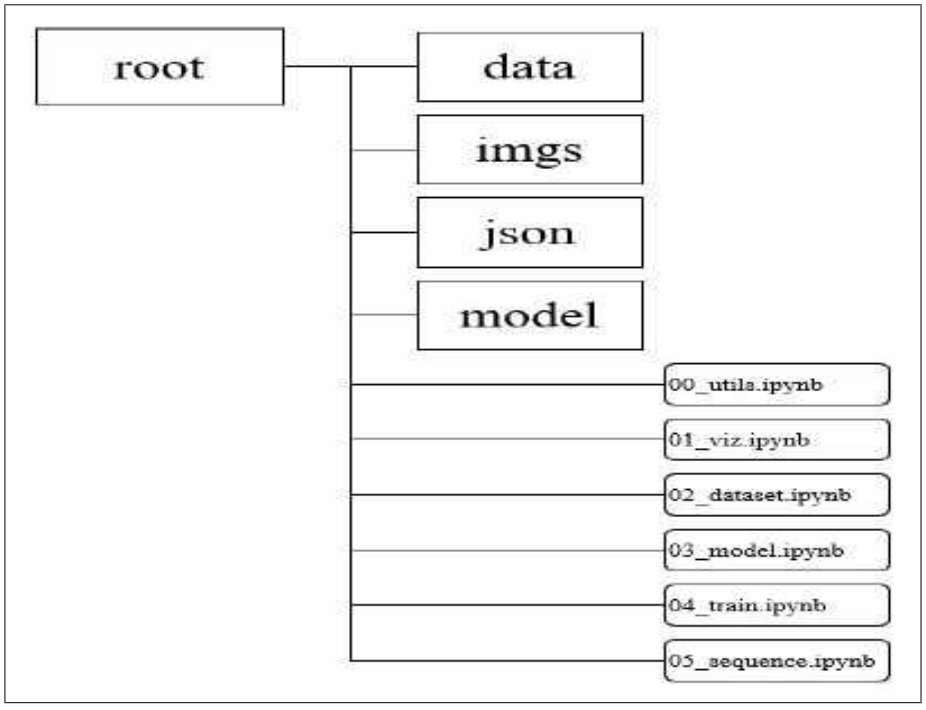
Perancangan struktur proyek yang sistematis diperlukan untuk meminimalisir kompleksitas dalam melakukan pembuatan dan pemelajaran model. *Integrated development environment* Jupyter Lab memudahkan eksekusi perintah dengan sintaks bahasa pemrograman Python dalam bentuk sel interaktif dalam berkas berekstensi ipynb. Setiap sel terdiri dari *input* dan *output*. *Input* berisi perintah yang akan dieksekusi, sedangkan *output* berisi hasil eksekusi yang dapat berupa teks ataupun grafik. Tampilan Jupyter Lab dapat dilihat pada gambar 3.2.

Struktur direktori yang disusun terdiri dari empat folder dan enam berkas *interactive python notebook* berekstensi ipynb. Folder data berisi data latihan model seperti pada gambar 3.3. Folder imgs berisi rangkaian gambar yang diambil dari rekaman video. Folder json menampung berkas yang berisi informasi titik kunci dua dimensi saat inferensi. Folder model berisi *checkpoint* model selama pelatihan.



Gambar 3.2: Jupyter Lab

Berkas berekstensi ipynb berisi kode pemrograman yang dibagi menjadi enam modul. Berkas 00_utils merupakan modul utilitas yang memudahkan proses memuat data, menampilkan data, menyimpan data, dan memanipulasi data. Berkas 01_viz adalah modul percobaan menampilkan data pelatihan dan data inferensi. Berkas 03_model merupakan modul pendefinisian arsitektur model. Berkas 04_train adalah modul pelatihan model. Berkas 05_sequence adalah modul percobaan inferensi model terhadap rangkaian gambar.



Gambar 3.3: Struktur Direktori

3.3.3 Analisis Data

Data pembuatan model yang digunakan adalah data pemetaan dari pose dua dimensi ke pose tiga dimensi. Pose dua dimensi merupakan sampel, sedangkan pose tiga dimensi merupakan target. Sumber data adalah Human3.6M Dataset mengenai informasi perakaman gerakan pose manusia yang menyimpan gambar dari beberapa sisi beserta dengan pose dua dimensi dan tiga dimensinya. Informasi kedua pose tersebut kemudian dipisah dan disimpan dalam file berekstensi "pt" [11]. Informasi mengenai data pemelajaran dapat dilihat pada tabel 3.3.

Tabel 3.3: Data Pemelajaran Model

Nama Berkas	Isi
rcams.pt	matriks kamera *motion capture* terhadap kamera perekam
stat_2d.pt	*mean*, *standard-deviation*, dan *skeleton* pose dua dimensi
stat_3d.pt	*mean*, *standard-deviation*, dan *skeleton* pose tiga dimensi
test_2d.pt	data pose dua dimensi untuk validasi model
test_3d.pt	data pose tiga dimensi untuk validasi model
train_2d.pt	data pose dua dimensi untuk pelatihan model
train_3d.pt	data pose tiga dimensi untuk pelatihan model

Data inferensi model yang digunakan berupa video yang direkam menggunakan kamera diatas sebuah *tripod*. Hasil rekaman berupa sebuah video monokuler berisi seorang peraga yang memperagakan berbagai pose dasar. Pose-pose yang diperagakan mencakupi gerakan lengan, gerakan kaki, gerakan pinggang, gerakan kepala, dan gerakan memutar. Kompleksitas gerakan ini mengakibatkan terjadinya oklusi pada beberapa bagian badan yang berarti suatu anggota badan menutupi anggota badan lainnya.

Kualitas perekaman video diturunkan dengan menggunakan pencahayaan yang satu arah. Efek *motion blur* diaktifkan sehingga gerakan yang cepat akan mengalami pengaburan. Kompleksitas gambar juga ditingkatkan dengan *aspect ratio* yang bernilai 19 : 6 dengan tipe rekaman *potrait* menyebabkan area rekaman hanya berada ditengah dan diapit oleh area piksel hitam.

Gerakan yang bervariasi dapat menyebabkan beberapa masalah dimana tidak setiap pose tertangkap dengan jelas dan lengkap. Gerakan cepat mengangkat

tangan kedua tangan yang memutar ke arah kiri atas menyebabkan terjadinya *motion blur*. Kedua tangan terlihat kabur dalam hasil tangkapan gambar yang didapatkan seperti pada gambar 3.4.

Gambar 3.4: Analisis *Frame* 00077

Gerakan peraga ketika mengangkat kaki kiri bersamaan dengan kedua tangan berada di pinggang peraga juga menyebabkan pose yang tidak lengkap. Gerakan ini menyebabkan oklusi pose yang menghilangkan sebagian anggota badan. Hasil tangkapan gambar yang didapatkan adalah kaki kiri peraga menutupi tangan kiri peraga yang dapat dilihat pada gambar 3.5.

Gambar 3.5: Analisis *Frame* 00173

Gerakan memutar keseluruhan badan peraga juga menutupi anggota badan lainnya. Gerakan ini menyebabkan. peraga yang menghadap kearah kanan menutupi tangan kanan dan sebagian kaki kiri yang dapat dilihat seperti pada gambar 3.6.

Gambar 3.6: Analisis *Frame* 00232

3.4 Tahap Produksi

Tahap produksi merupakan tahap kedua yang dimana pengerjaan dilakukan. Tahap ini terdiri dari tiga langkah utama yaitu prapemrosesan data pelatihan, pembuatan arsitektur model, dan pemelajaran model.

3.4.1 Prapemrosesan Data Pelatihan

Dataset pembuatan model terbagi menjadi dua kategori yaitu data pelatihan (train_2d.pt dan train_3d.pt) dan data validasi (test_2d.pt dan test_3d.pt) . Kedua data ini memiliki struktur dan bentuk yang sama sehingga prapemrosesan yang dilakukan juga sama. Setiap sampel pada data terdiri dari satu pose dua dimensi dan satu pose tiga dimensi. Perbedaan daripada kedua data ini terletak pada jumlah sampelnya. Data pelatihan berisi sebanyak 1.559.752 pasang sampel sedangkan data validasi memiliki 550.644 pasang sampel.

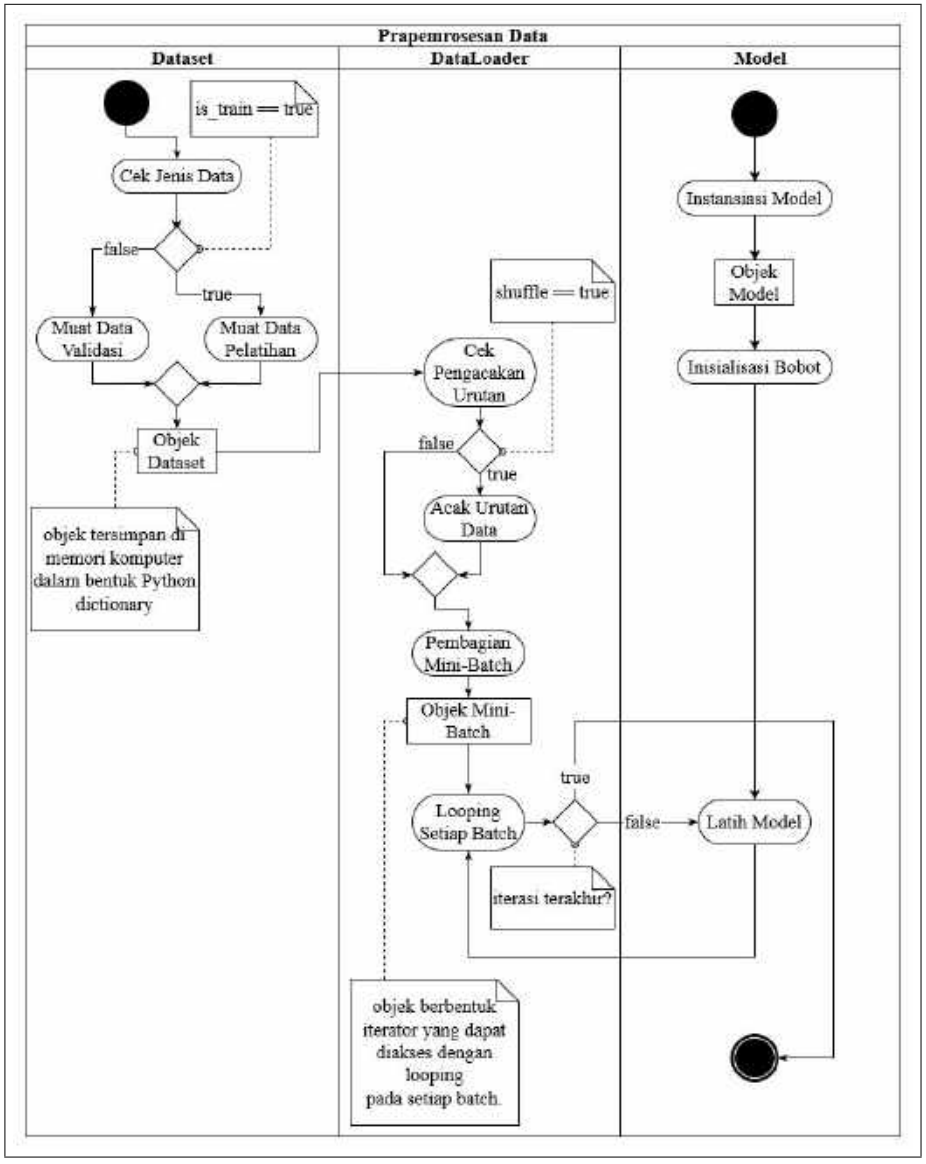
Berkas test_2d.pt, test_3d.pt, train_2d.pt, dan train_3d.pt merupakan berkas hasil konversi struktur data *dictionary* yang telah berbentuk *serialized*.

Struktur *dictionary* yang berada didalam memori disimpan ke media SSD dalam bentuk *byte stream*. Berkas-berkas ini harus dibaca dengan mekanisme *deserializing* yaitu membangun ulang sebuah struktur data yang sama dengan membaca rangkaian *byte* sehingga dapat digunakan pada proses pelatihan model.

Mekanisme pemuatan data dilakukan secara *stochastic* dikarenakan hasil konversi data yang berukuran sangat besar. Serangkaian pasangan kunci dan target diproses dengan ukuran tertentu yang disebut dengan *mini-batch* sehingga VRAM pada GPU tidak mengalami *overflow*. Pemuatan data secara *stochastic* juga mempercepat proses pelatihan model karena lebih sedikit data yang harus dikalkukasi sebelum melakukan pemuktahiran.

Kelas Dataset dan DataLoader pada *framework* PyTorch memiliki fungsionalitas untuk melakukan pembacaan dan pembagian rangkaian data secara *stochastic*. Kelas Dataset dapat membaca berkas berekstensi "pt" dari media SSD kemudian dimuat kedalam memori. Kelas DataLoader dapat memindahkan informasi didalam kelas Dataset ke VRAM pada GPU berbentuk *mini-batch* sehingga dapat diproses secara *stochastic* dan paralel.

Prapemrosesan data pelatihan dan data validasi dilakukan dengan cara yang sama. Langkah pertama yang dilakukan adalah melakukan cek apakah data yang diinginkan adalah data pelatihan atau data validasi. Data tersebut kemudian dimuat dan disimpan kedalam memori. DataLoader menerima objek tersebut kemudian melakukan pengacakan data dan pembagian *mini-batch*. Hasil objek dapat diiterasi untuk melakukan pelatihan model. Skema prapemrosesan data dapat dilihat pada gambar 3.7.

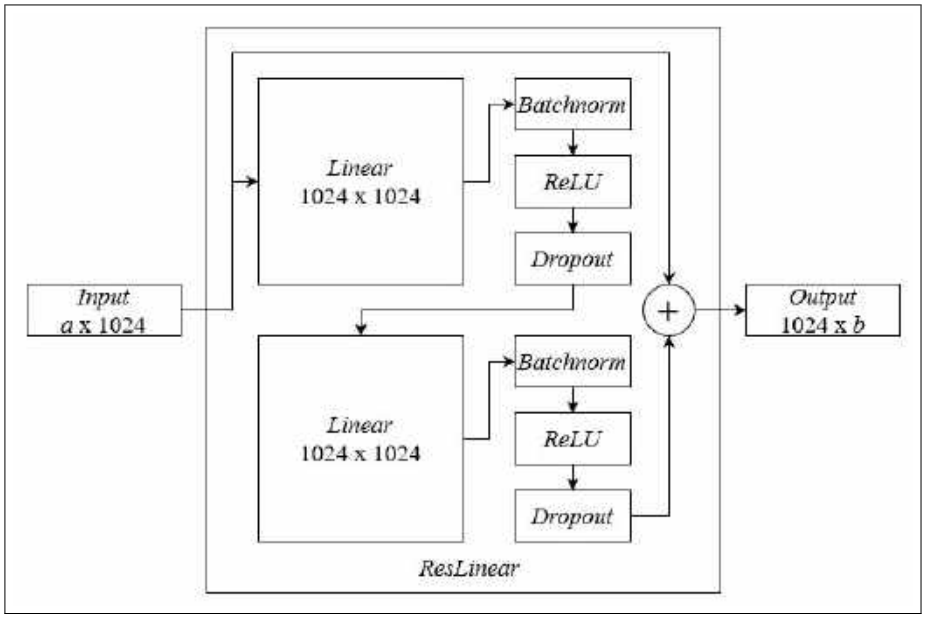


Gambar 3.7: *Activity Diagram* Prapemrosesan Data

3.4.2 Arsitektur Model

Arsitektur model jaringan saraf tiruan yang digunakan memiliki *input* berbentuk vektor dengan ukuran tiga puluh dua dan *output* berbentuk vektor dengan ukuran empat puluh delapan. Rangkaian lapisan yang menghubungkan *input* dan *output* berupa lapisan *residual network*. Bobot setiap lapisan diinisialisasi secara acak dengan distribusi normal.

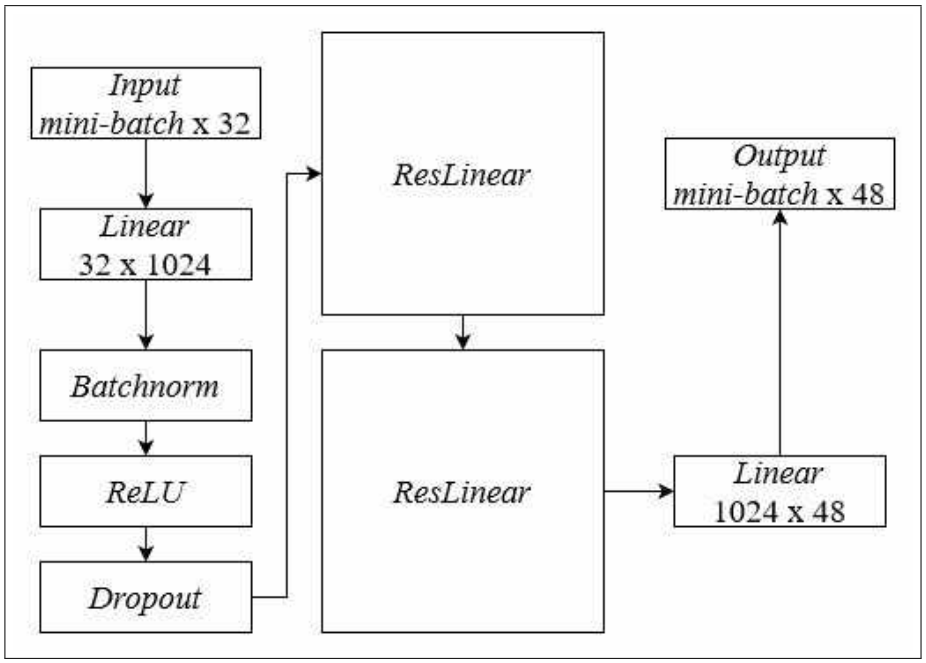
Sebuah lapisan *residual network* merupakan jaringan dengan arsitektur kecil dan sederhana yang dapat dipasang atau dibongkar secara modular yang disebut ResLinear. Lapisan ResLinear melakukan operasi penjumlahan antara *input* dan *output*. Sebuah ResLinear memiliki dua lapisan linier, dua lapisan *Batchnorm*, dua lapisan *Dropout*, dan dua lapisan *ReLU*. Komponen-komponen penyusun sebuah lapisan ResLinear dengan ukuran *input a* dan ukuran *output b* seperti pada gambar 3.8.



Gambar 3.8: Arsitektur Lapisan ResLinear

Arsitektur model secara keseluruhan terdiri dari tiga kelompok yang meliputi lapisan awal, lapisan *residual*, dan lapisan akhir. Lapisan awal menjembatani data *input* dan lapisan *residual* dengan menggunakan sebuah lapisan linier sebagai penyambung. Lapisan *residual* terdiri dari dua buah lapisan ResLinear yang berfungsi untuk memetakan pola data *input* terhadap data *output*.

Lapisan akhir menjembatani hasil pemetaan yang dilakukan oleh lapisan *residual* terhadap data *output*. Arsitektur model dapat dilihat pada gambar 3.9.

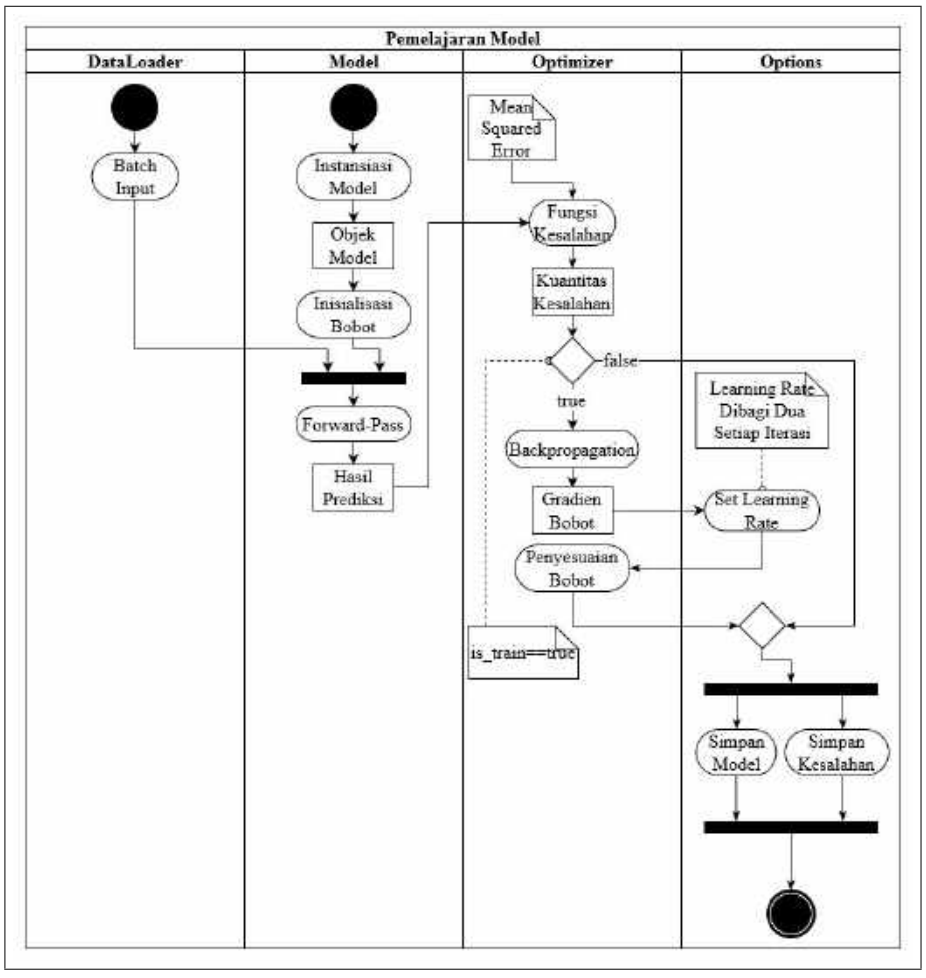


Gambar 3.9: Arsitektur Model

3.4.3 Pemelajaran Model

Pemelajaran model merupakan kelanjutan dari prapemrosesan data. Sebuah model baru diinstansiasi sehingga menghasilkan objek model. Objek model tersebut kemudian menginisialisasi bobot parameter dengan bilangan acak dari distribusi normal. *Batch input* yang berasal dari interasi DataLoader diproses oleh model dengan metode *feed-forward*. Hasil prediksi sementara dari model didapatkan yang kemudian dibandingkan tingkat kebenarannya menggunakan fungsi kesalahan. Fungsi kesalahan *mean squared error* menghasilkan angka kuantitas kesalahan. Angka ini merupakan tolak ukur seberapa akurat kemampuan model dalam menghasilkan output yang relevan. Apabila model tidak berada dalam status "is_train", maka kuantitas kesalahan yang didapatkan langsung disimpan dalam bentuk *array* untuk keperluan analisis. Apabila model berada dalam status "is_train", maka model melakukan *backpropagation* untuk menghasilkan gradien bobot. *Learning rate* kemudian dibagi dengan dua sehingga

menjadi lebih kecil. Penyesuaian bobot dilakukan dengan menjumlahkan bobot dengan hasil operasi perkalian antara *learning rate* dan gradien bobot. Bobot model yang telah diperbarui disimpan berserta dengan kuantitas kesalahannya. Algoritma yang sama akan diulangi pada setiap *batch input*. Skema pemelajaran model dapat dilihat pada gambar 3.10.



Gambar 3.10: *Activity Diagram* Pemelajaran Model

3.5 Tahap Uji Coba

Tahap uji coba berisi langkah-langkah uji coba penggunaan model. Tahap uji coba dibagi menjadi beberapa langkah yang meliputi prapemrosesan data inferensi, penggunaan OpenPose, inferensi model, dan visualisasi. Tahap ini bertujuan untuk menggunakan model sebagai aplikasi untuk mengestimasi pose tiga dimensi dari gambar monokuler.

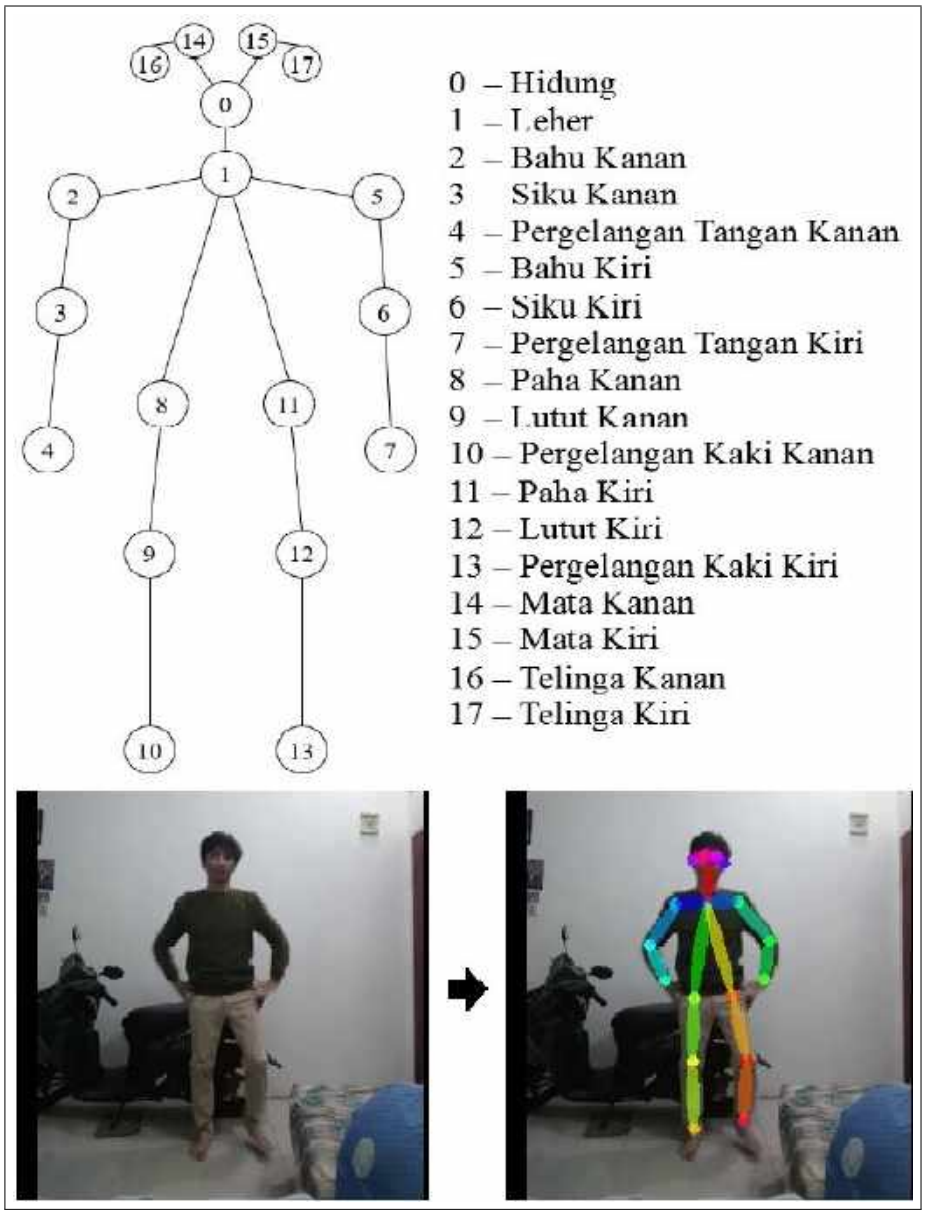
3.5.1 Prapemrosesan Data Inferensi

Proses inferensi pose dua dimensi menjadi lebih cepat ketika gambar dua dimensi yang dioperasikan memiliki resolusi yang kecil. Tingkat resolusi yang digunakan saat merekam pose adalah 640 x 360. Memperkecil ukuran resolusi juga harus dilihat dari segi kualitas gambar yang dihasilkan. Gambar juga memiliki banyak informasi statis seperti pada area piksel berwarna hitam yang berada di kiri dan kanan gambar. Resolusi 290 x 290 dengan titik tengah berada pada titik kunci pinggang dianggap tepat karena mampu menjangkau semua pose badan dan kualitas gambar yang masih bagus. Inferensi yang lebih efisien dapat dilakukan dengan memperkecil resolusi dan area piksel mati seperti pada gambar 3.11.

Gambar 3.11: Resolusi Gambar Data Inferensi

3.5.2 OpenPose

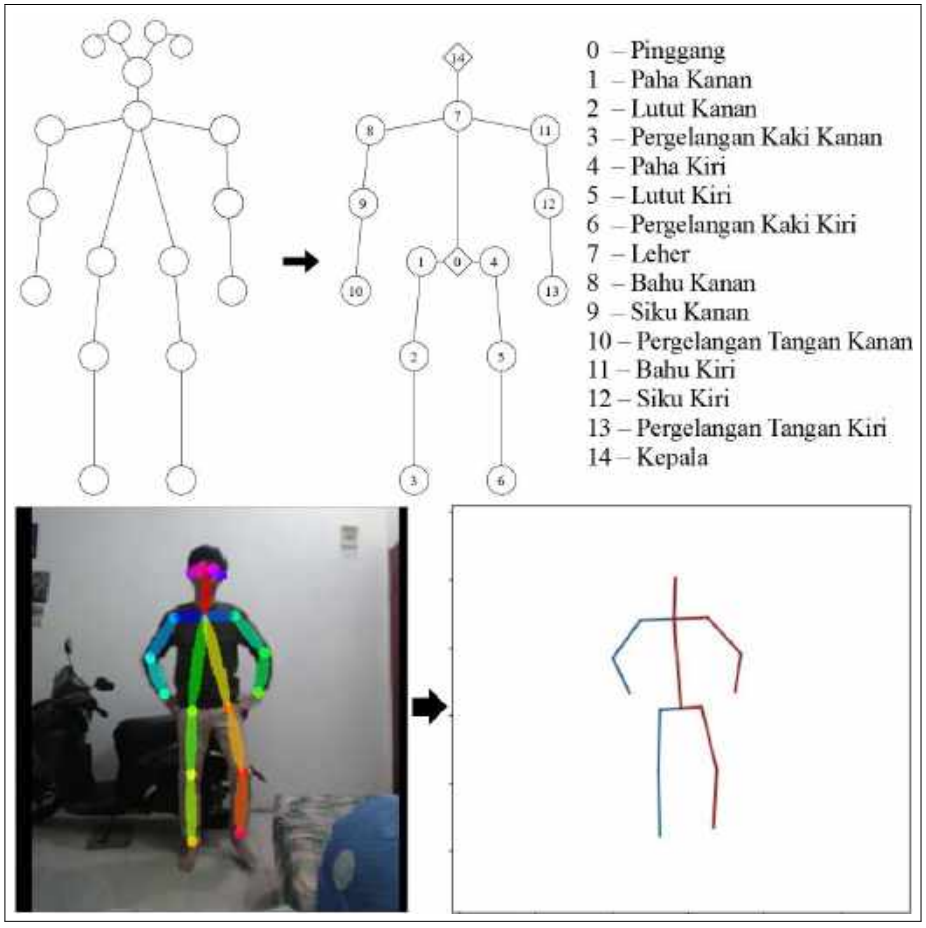
OpenPose merupakan aplikasi estimasi pose tubuh manusia dua dimensi. OpenPose menerima input gambar kemudian mencari titik kunci pose dua dimensi. Titik kunci pose dua dimensi berada pada koordinat lokal sesuai dengan bidang gambar. OpenPose menghasilkan berkas "json" yang berisi hierarki pose menurut spesifikasi COCO-MS. Anotasi COCO-MS berisi delapan belas titik kunci tubuh manusia dengan urutan tertentu. Visualisasi estimasi pose dua dimensi menggunakan OpenPose dapat dilihat pada gambar 3.12.



Gambar 3.12: Spesifikasi Titik Kunci COCO-MS

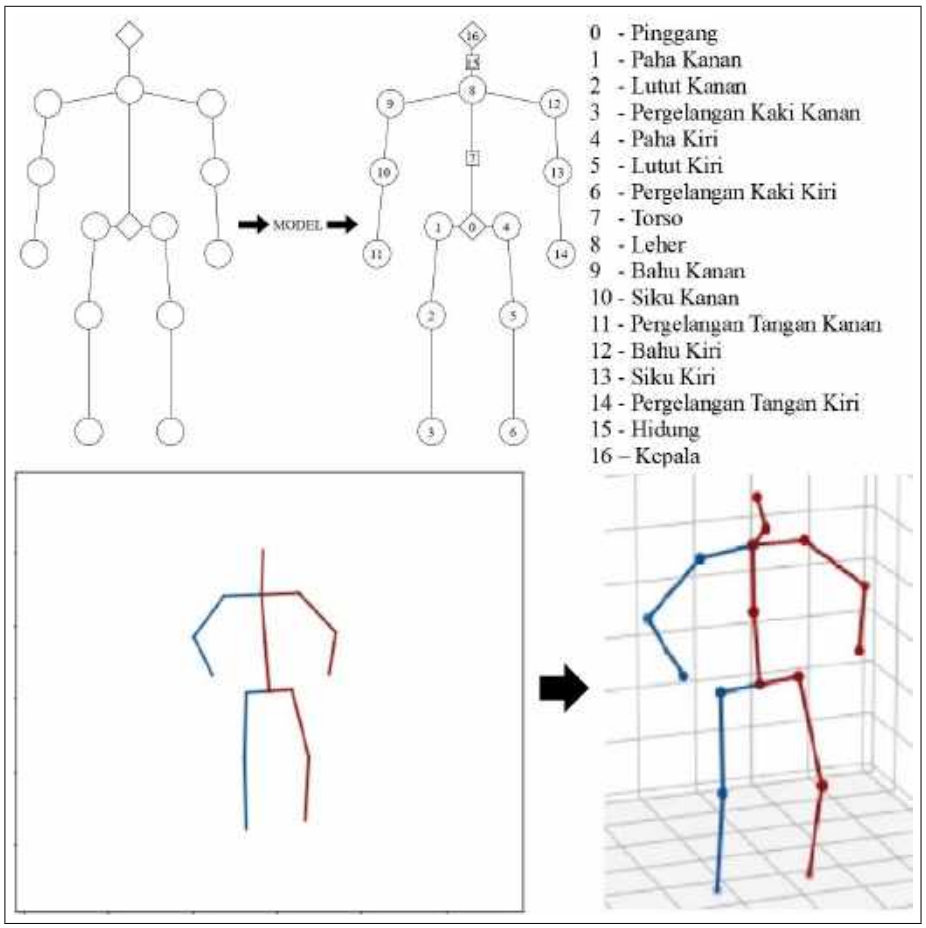
3.5.3 Inferensi Model

Rangkaian titik kunci yang dihasilkan oleh OpenPose memiliki spesifikasi COCO-MS yang berbeda dengan spesifikasi *dataset* pembuatan model. Spesifikasi COCO-MS yang memiliki delapan belas titik kunci dikonversi menjadi lima belas titik kunci dengan menyatukan titik kunci kedua mata dan kedua telinga serta membuat titik kunci pinggang berdasarkan rata-rata kedua kaki bagian atas.Konversi ini dilakukan supaya model dapat melakukan inferensi terhadap titik kunci yang mewakili pose tersebut. Visualisasi konversi titik kunci COCO-MS menggunakan warna biru mewakili anggota badan sebelah kanan bersamaan dengan warna merah yang mewakili anggota badan tengah dan kiri. seperti pada gambar 3.13.



Gambar 3.13: Konversi Titik Kunci Dua Dimensi

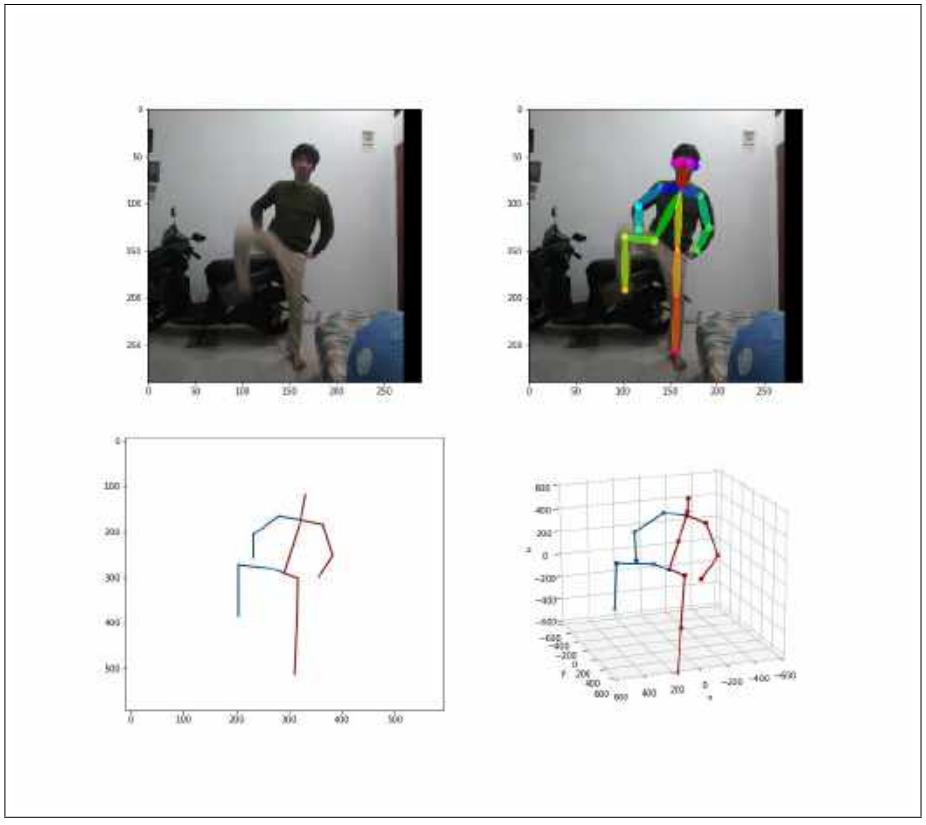
Inferensi pada model *deep neural network* menerima titik kunci pose dua dimensi yang telah dikonversi sebagai *input* kemudian menghasilkan titik kunci pose tiga dimensi sebagai *output*. Inferensi model menghasilkan tujuh belas titik yang berada dalam bidang tiga dimensi. Titik kunci pose tiga dimensi memiliki dua titik baru yang meliputi torso dan hidung. Titik torso mewakili lengkungan badan pada pose tiga dimensi sehingga pose terlihat akurat. Titik hidung mewakili arah hadapan kepala. Kedua titik tambahan ini memperjelas orientasi pose tiga dimensi. Proses inferensi model untuk mendapatkan pose tiga dimensi dapat dilihat pada gambar 3.14.



Gambar 3.14: Inferensi Model

3.5.4 Visualisasi

Visualisasi mencakup penggunaan aplikasi dalam satu bingkai secara keseluruhan. Terdapat empat buah figur yang masing-masing mewakili langkah-langkah pada tahapan uji coba. Figur pertama (kiri atas) berisi prapemrosesan data inferensi dimana resolusi gambar diubah menjadi 290 x 290. Figur kedua (kanan atas) menggambarkan pose dua dimensi yang didapatkan oleh OpenPose. Figur ketiga (kiri bawah) menggambarkan konversi titik kunci OpenPose dengan spesifikasi COCO-MS menjadi titik kunci yang cocok dengan model. Figur keempat (kanan bawah) menggambarkan pose tiga dimensi yang dihasilkan oleh model kedalam sebuah sistem koordinat tiga dimensi. Contoh visualisasi aplikasi dapat dilihat pada gambar 3.15.



Gambar 3.15: Visualisasi Aplikasi

3.5.5 *Script* Python

Langkah-langkah yang dilakukan pada tahapan produksi masih terpisah dalam beberapa *cell* di *interactive development environment* Jupyter Lab. *Cell* tersebut kemudian digabungkan kedalam sebuah *script* yang bernama "run.py". Hal ini dilakukan agar pengguna aplikasi dapat menjalankan aplikasi dengan mudah. Aplikasi dapat dijalankan dengan memasukkan perintah "python run.py" dengan syarat semua *dependencies* python versi 3 telah terpasang.