BAB 3. ANALISIS DAN DESAIN SISTEM

Pada bab ini dibahas mengenai desain sistem dan proses kerja sistem. Proses analisis dan desain dilakukan dengan tujuan untuk memberikan gambaran besar bagaimana sistem bekerja dan saling berkomunikasi sehingga program dapat berjalan.

3.1. Analisis Data

Pada bagian analisis ini akan ditinjau permasalahan yang berorientasi pada data yang digunakan. Bagian ini mencangkup penjelasan mengenai pengelompokkan klasifikasi data.

3.1.1. Klasifikasi Data

Agar data dapat diolah oleh *neural network* untuk didapatkan hasil kata berupa teks/*string* maka perlu adanya segmentasi per kata dari tulisan-tulisan yang ada pada akta kelahiran. Setelah setiap kalimat yang ada pada akta kelahiran dipecah menjadi kata per kata, perlu dilakukan pemberian label kata apa yang terdapat pada citra hasil segmentasi tersebut. Hal ini diperlukan untuk dapat melakukan tahap *training* pada *neural network*.

Tidak ditemukan cara praktis untuk dapat mengelompokkan dan memberi label pada data karena tidak ada program yang dapat membaca kata yang ada pada citra dan me-*rename file* secara otomatis, sehingga pengelompokkan dan pemberian label pada citra dilakukan secara manual dengan memilah dan mengelompokkan satu per satu seluruh citra kata pada *dataset* sesuai dengan label kata tersebut. Didapatkan ada 122 label kata dari hasil segmentasi per kata setiap akta kelahiran pada *dataset* yang dimiliki.

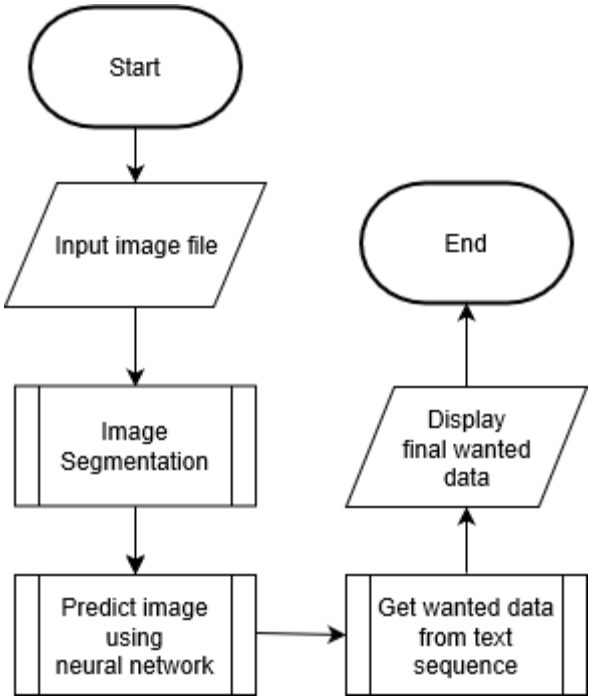
3.2. Garis Besar Kerja Sistem

Program pertama-tama akan menerima *input* dari *user* berupa *image* Akta Kelahiran dalam format .JPG atau .JPEG. *File* gambar tersebut kemudian akan

diolah hingga dapat terdeteksi dan tersegmentasi kata-kata yang ada di dalamnya. Segmentasi per kata ini dilakukan menggunakan metode *Run Length Smoothing Algorithm*.

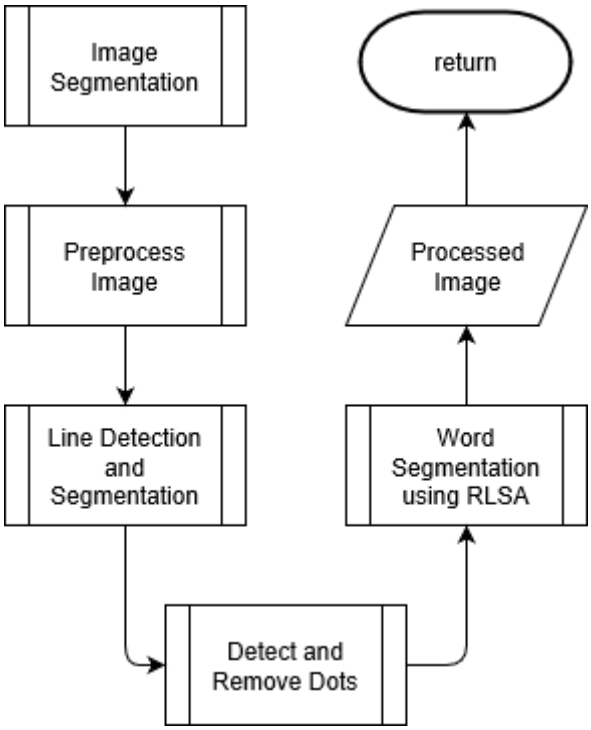
Setelah didapatkan seluruh gambar yang merepresentasikan sebuah kata hasil segmentasi sebelumnya, gambar-gambar tersebut akan dimasukkan ke dalam proses selanjutnya yaitu prediksi kata yang ada pada gambar. Kata pada setiap gambar akan diolah dan diproses menggunakan CRNN (*Convolutional Recurrent Neural Network*) yang telah di-*train* sebelumnya agar dapat menghasilkan kata dalam bentuk tipe data *string*.

Kemudian, akan dicari informasi-informasi yang diinginkan berdasarkan urutan *string* hasil prediksi CRNN tersebut. Hal ini digunakan dengan cara memperhatikan kata kunci apa saja yang mengikuti setiap informasi yang diinginkan. *Flowchart* gambaran besar sistem dapat dilihat pada Gambar 3.1.



Gambar 3.1. *Flowchart* Gambaran Besar Sistem

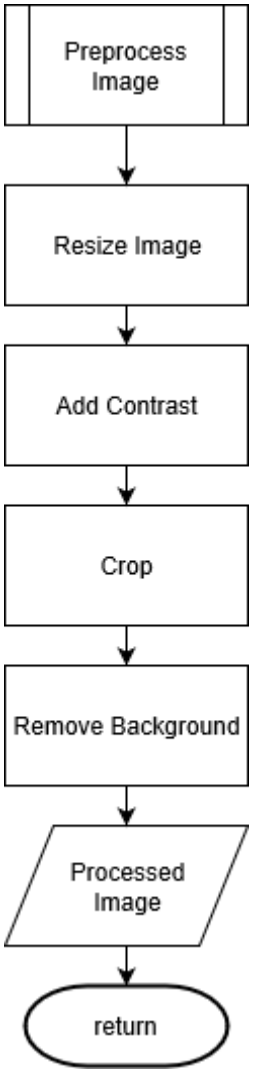
3.2.1. *Image Segmentation*



Gambar 3.2. *Flowchart* Sub-proses *Image Segmentation*

Sub-proses *Image Segmentation* bertujuan untuk mengolah citra Akta Kelahiran yang di-*input*-kan oleh *user* agar dapat menjadi potongan-potongan kata. Alur dari sub-proses ini dapat dilihat pada Gambar 3.2. Langkah pertama pada proses ini adalah *preprocessing*, dimana citra diolah terlebih dahulu agar didapatkan *Region of Interest* yaitu daerah tengah akta kelahiran yang mengandung informasi yang diinginkan. Setelah itu akan dilakukan proses *Line Detection and Segmentation* untuk mensegmentasi hasil dari *Region of Interest* menjadi per baris. Kemudian akan dilanjutkan dengan pencarian dan penghapusan titik-titik agar tidak mengganggu proses selanjutnya yaitu segmentasi per kata menggunakan metode RLSA (*Run Length Smoothing Algorithm*).

Sub-proses *Preprocessing Image*

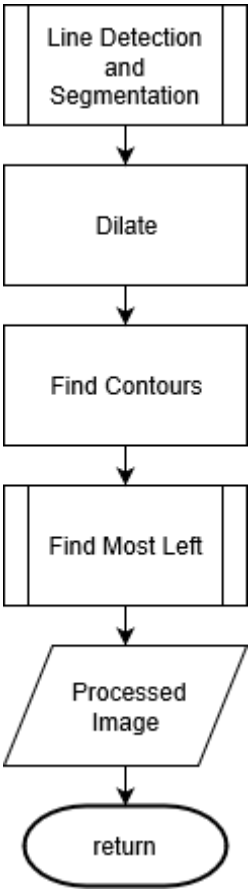


Gambar 3.3. *Flowchart* Sub-proses *Preprocessing Image*

Sub-proses *Preprocessing* ini digunakan untuk mengubah citra yang di-*input user* agar nantinya dapat diproses dengan lebih mudah. Pertama-tama citra akan di-*resize* ukurannya, setelah itu akan ditambahkan kontras agar ketajaman citra bertambah dan akan lebih terlihat perbedaan antara tulisan pada akta dengan yang tidak. Setelah itu citra akan di-*crop* agar didapatkan bagian tengah dari akta yang mengandung informasi yang diinginkan. Proses terakhir yang dijalankan

adalah menghilangkan *background* dari akta agar tersisa hanya tulisan kata-kata saja. Flowchart sub-proses ini dapat dilihat pada Gambar 3.3.

Sub-proses *Line Detection and Segmentation*

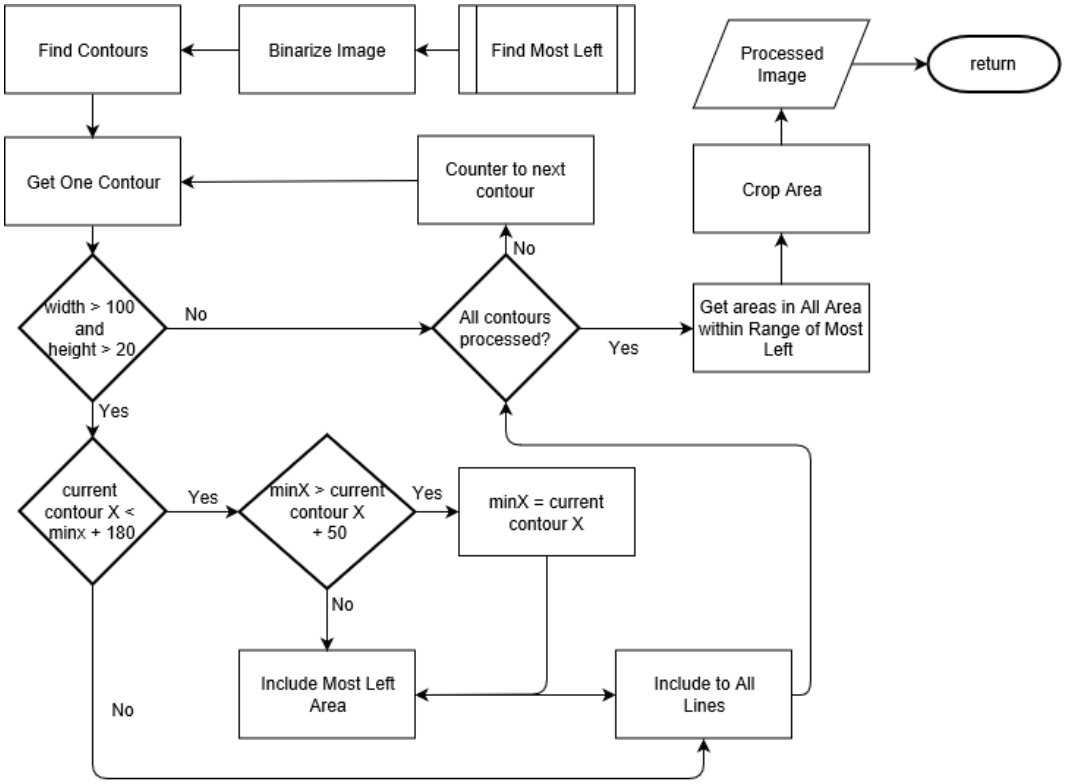


Gambar 3.4. *Flowchart* Sub-proses *Line Detection and Segmentation*

Sesuai yang dapat dilihat pada Gambar 3.4, proses diawali dengan melakukan dilasi pada citra agar setiap pixel-pixel dari kata-kata lebih menyatu antara satu dengan yang lainnya untuk memudahkan proses deteksi baris. Setelah itu dicari kontur yang ada pada citra untuk menemukan kata-kata yang telah menyatu tersebut dan mengelompokkannya per baris, dari hasil ini akan kemudian dicari kontur mana saja yang terletak pada paling kiri atau yang memiliki nilai posisi pada sumbu X paling kecil. Kontur-kontur yang berada pada bagian terkiri

nantinya akan kemudian di-*crop* sehingga dihasilkan citra yang berisikan satu baris kata-kata saja.

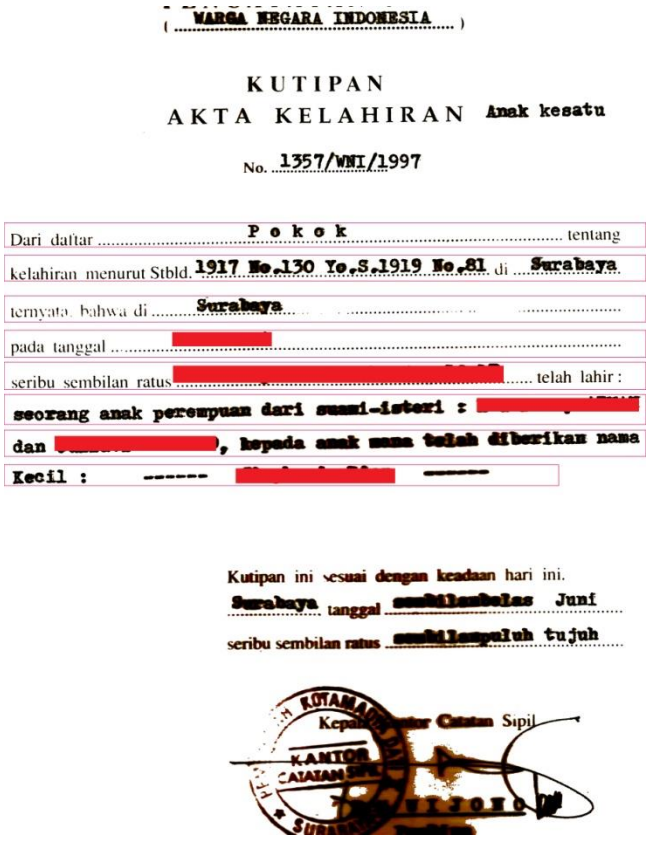
Sub-proses *Find Most Left*



Gambar 3.5. *Flowchart* Sub-proses *Find Most Left*

Sub-proses ini bertujuan untuk mendapatkan baris-baris yang terdapat pada area tengah akta kelahiran, dimana gambaran detail prosesnya dapat dilihat pada Gambar 3.5. Citra pertama-tama akan diubah menjadi *binary* terlebih dahulu dan dicari konturnya. Untuk setiap kontur yang ada, dilakukan pengecekkan apakah lebar dan tinggi dari kontur tersebut melebihi *threshold.* Pada penelitian ini *threshold* yang dipakai adalah 100 untuk lebar dan 20 untuk tinggi. Jika melebihi, maka daerah kontur tersebut akan dianggap sebagai suatu baris tersendiri dan disimpan pada variabel *All Lines*. Kemudian akan dilakukan pengecekkan juga apakah nilai dari sumbu X kiri kontur lebih kecil dari nilai

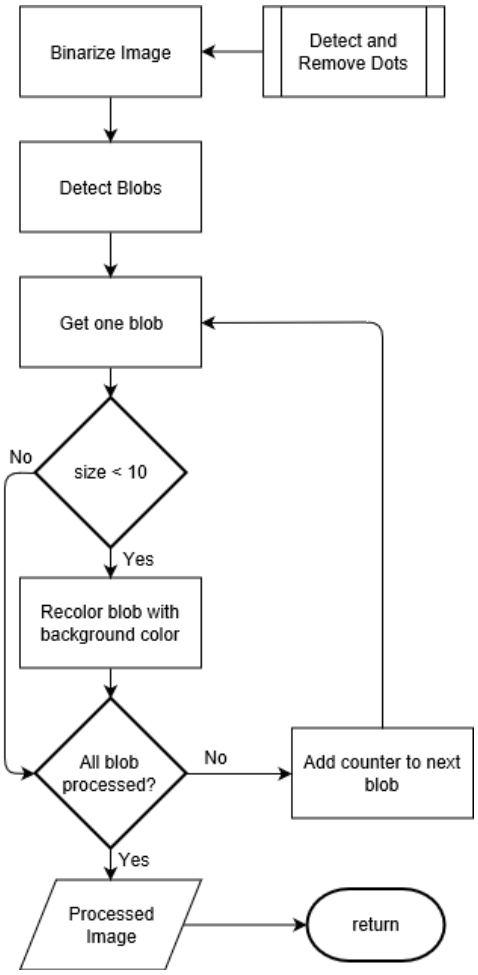
minimum X saat ini ditambahkan dengan 180. Jika lebih kecil maka daerah baris tersebut akan digolongkan sebagai daerah *most left* dan disimpan pada variabel *Most Left*. Akan dicek juga jika nilai sumbu X kiri kontur ditambah 50 lebih kecil dari nilai minimum X maka nilai minimum X akan di-*update* sesuai nilai sumbu X kiri kontur tersebut. Setelah semua kontur selesai dicek, maka akan dicari keseluruhan area dari baris-baris yang ada pada variabel *Most Left*. Baris-baris pada variabel *All Lines* yang termasuk dalam area tersebut akan diambil dan diproses lebih lanjut. Hasil seluruh segmentasi baris pada akta dapat dilihat pada Gambar 3.6, sedangkan hasil *crop* untuk sebuah baris dapat dilihat pada Gambar 3.7



Gambar 3.6. Gambar Hasil Deteksi *Line Detection* dan *Most Left*

Gambar 3.7. Gambar Hasil *Crop Line Segmentation* dan *Most Left*

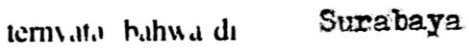
Sub-proses *Detect and Remove Dots*



Gambar 3.8. *Flowchart* Sub-proses *Detect and Remove Dots*

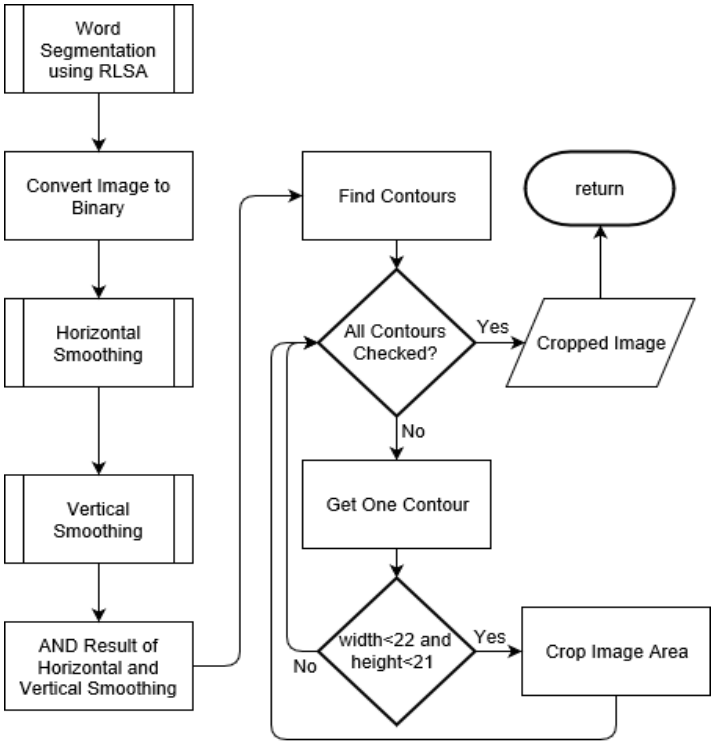
Proses ini dilakukan untuk mendeteksi titik-titik yang ada pada bagian bawah dari tulisan. Hal ini dilakukan agar hasil segmentasi per kata melalui metode RLSA dapat lebih maksimal. Jika titik-titik ini tidak dihilangkan, maka akan ada beberapa kata yang tersambung akibat adanya titik-titik ini. Dimulai dengan mengubah *image* menjadi *binary*, program akan kemudian mendeteksi *blob* pada citra. Untuk setiap *blob* yang ditemukan akan dilakukan pengecekkan apakah ukurannya lebih kecil dari 10 pixel. Jika iya, maka akan dilakukan pewarnaan pada *blob* dengan menambahkan objek berbentuk lingkaran dengan warna yang sama dengan *background* (dalam kasus ini berwarna putih).

Gambaran sub-proses ini dapat dilihat pada Gambar 3.8 dan contoh hasil sub-proses ini terdapat pada Gambar 3.9.



Gambar 3.9. Hasil Sub-proses *Detect and Remove Dots*

Sub-proses *Word Segmentation using Run Length Smoothing Algorithm*

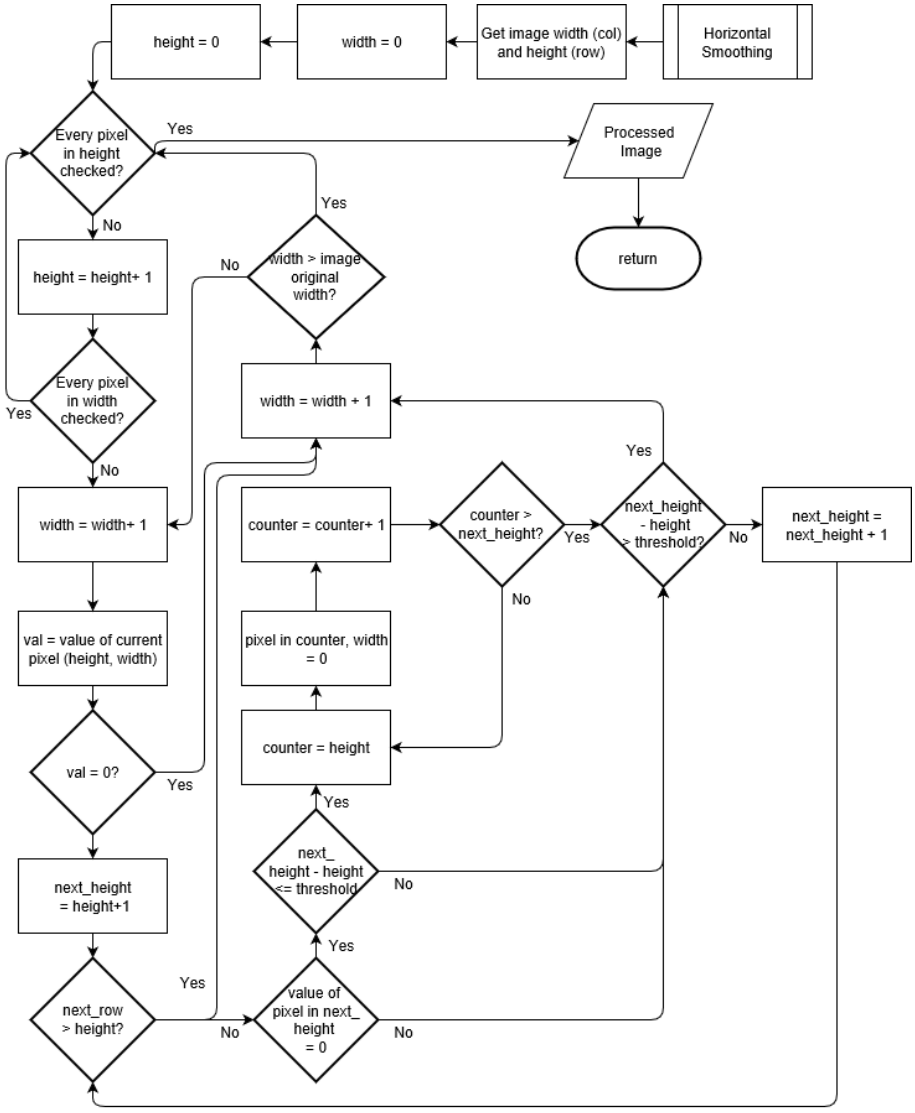


Gambar 3.10. *Flowchart* Sub-proses *Word Segmentation using* RLSA

Sesuai diagram pada Gambar 3.10, untuk dapat memecah setiap baris menjadi per kata digunakan metode RLSA. Pertama-tama, gambar diubah ke dalam bentuk *binary*, setelah itu dilakukan proses RLSA. Pada proses RLSA sendiri terbagi menjadi dua proses yaitu *smoothing* secara horizontal dan *smoothing* secara vertikal. Hasil dari kedua proses ini akan digabungkan dengan metode *AND*. Dari hasil penggabungan ini kemudian akan dicari bagian mana saja

yang merupakan daerah tulisan (kontur) dan bukan *background*. Pada setiap kontur yang ada dilakukan pengecekkan jika memiliki lebar lebih kecil dari 22 dan tinggi lebih kecil dari 21 pixel akan di-*crop* pada daerah tersebut. Hasil akhir berupa gambar-gambar hasil *crop* yang masing-masing berisikan sebuah kata yang ada pada *input image*.

Sub-Proses *Horizontal Smoothing*

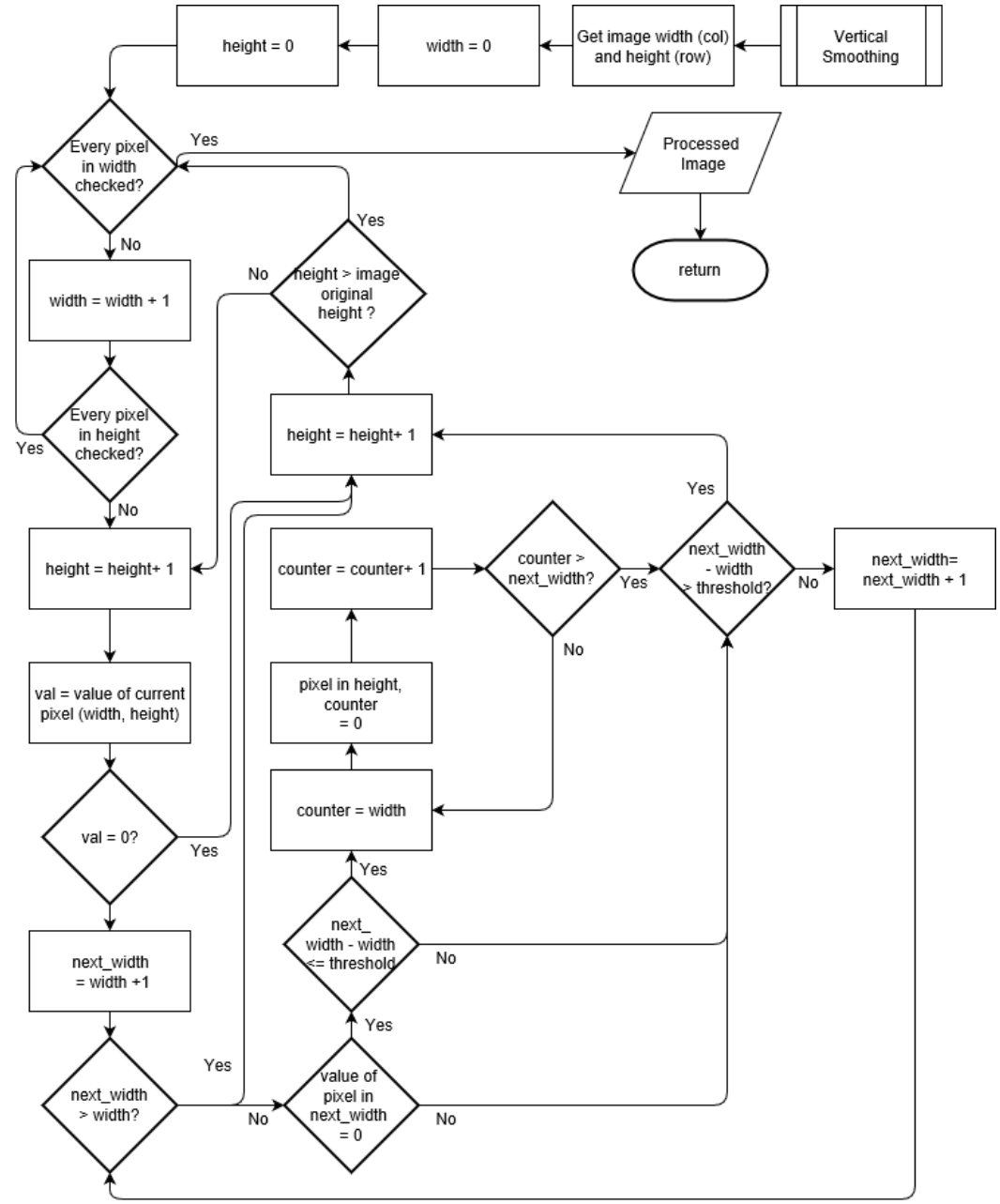


Gambar 3.11. *Flowchart* Sub-proses *Horizontal Smoothing*

Proses RLSA ini pada intinya adalah untuk mengubah pixel yang bukan hitam menjadi hitam apabila memenuhi syarat memiliki jarak dari pixel hitam lainnya yang lebih kecil dari *threshold*. Seperti yang ada pada Gambar 3.11, dapat dilihat bahwa proses ini dimulai dengan mendapatkan terlebih dahulu dimensi panjang dan lebar dari *image* yang diproses. Untuk setiap pixel yang ada pada *image* dilakukan beberapa pengecekkan. Berikut adalah beberapa pengecekkan yang dilakukan:

1. Apakah nilai dari pixel saat ini 0? Nilai pixel 0 merepresentasikan warna hitam dimana itu berarti pixel merupakan bagian dari tulisan. Apabila tidak maka lanjut ke langkah 2, apabila iya maka tidak dilakukan perubahan apapun dan lanjut ke pixel selanjutnya.
2. Dilakukan pengecekkan ke pixel selanjutnya (*height* + 1) hingga ditemukan pixel berwarna hitam atau hingga sudah sampai pada titik paling ujung (sama dengan *height* dari citra). Jika ditemukan pixel dengan warna hitam dilanjutkan ke langkah 3. Jika tidak lanjut ke langkah 5.
3. Dilakukan pengecekkan apakah jarak antara titik sekarang ini dengan titik *height* pixel hitam awal melebihi dari batas *threshold* atau tidak. Jika tidak maka dilanjutkan ke langkah 4, jika iya langsung ke langkah 5.
4. Seluruh pixel dari titik *height* pixel hitam awal hingga titik sekarang diubah menjadi warna hitam.
5. Dilanjutkan pengecekkan ke pixel selanjutnya (*width* + 1). Jika belum dilakukan pengecekkan pada seluruh pixel pada panjang/*width image*, maka kembali pada langkah 2. Jika sudah maka lanjut ke langkah 6.
6. Dilanjutkan pengecekkan pada pixel *height* selanjutnya. Jika sudah sampai pada batas akhir *height image* maka proses selesai dan dihasilkan citra yang sudah ter-*smoothing* secara horizontal. Jika belum, maka kembali pada langkah 1.

Sub-proses *Vertical Smoothing*

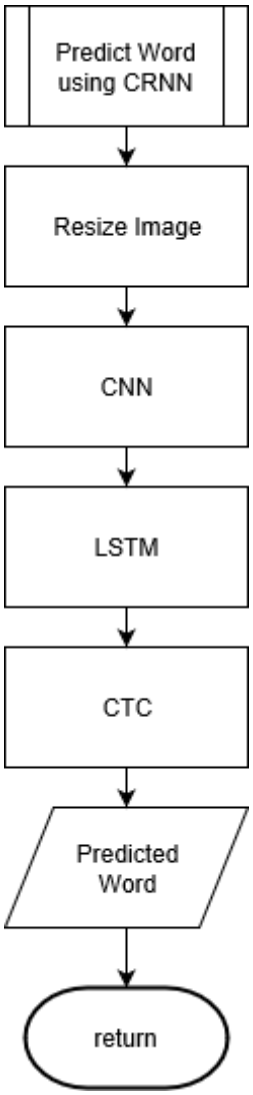


Gambar 3.12. *Flowchart* Sub-proses *Vertical Smoothing*

Pada proses ini dilakukan hal yang serupa dengan *horizontal smoothing*, namun bedanya ada pada bagian pixel mana yang diperiksa. Jika *horizontal smoothing* melihat pixel yang searah dengan sumbu X dan merubah sesuai dengan

jarak pada sumbu X, *vertical smoothing* melihat sumbu Y dan merubah sesuai dengan jarak pada sumbu Y tersebut. Gambar diagram yang menjelaskan sub-proses ini dapat dilihat pada Gambar 3.12.

3.2.2. *Neural Network*



Gambar 3.13. *Flowchart* Sub-proses *Neural Network*

Setelah melalui proses *Image Segmentation*, citra-citra yang telah tersegmentasi sehingga terpisah menjadi kata per kata akan diproses dalam *neural network*. Pertama-tama setiap *image* harus di-*resize* menjadi ukuran yang sesuai. Pada penelitian ini digunakan ukuran *image* 32x32 pixel. Setelah selesai maka

akan diolah pada *neural network* yang terdiri dari CNN, LSTM dan CTC. Data yang dimasukkan kepada *neural network* akan berurutan sesuai dengan urutan hasil segmentasi baris dan segmentasi kata sebelumnya. Flowchart dari sub-proses ini dapat dilihat pada Gambar 3.13.

3.2.2.1. Model *Neural Network*

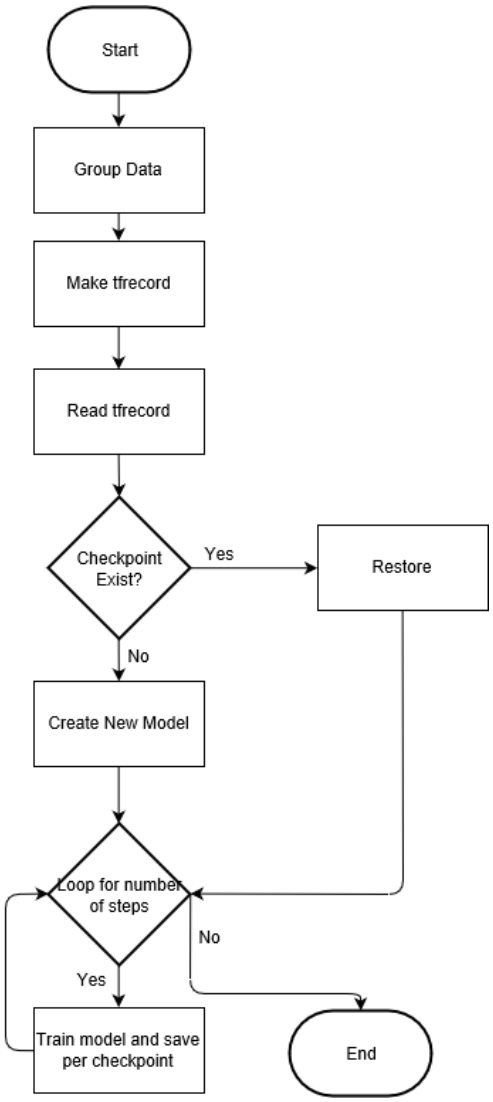
Pada penelitian ini digunakan penggabungan dari *neural network* CNN dan LSTM. Digunakan *Convolutional Layer* dan *Pooling Layer* dari CNN, namun *Fully-connected Layer* yang pada umumnya ada pada CNN digantikan dengan LSTM. Hal ini dilakukan agar *neural network* dapat mengenali citra per kata dan tidak per huruf saja. Pada *layer-layer* milik CNN digunakan ReLU sebagai *activation function*. Rincian layer-layer yang digunakan dapat dilihat pada 0.

Layer	**Operation**	**Kernel Size**
1	*Convolutional*	3
2	*Convolutional*	3
	Pooling	2
3	*Convolutional*	3
4	*Convolutional*	3
	Pooling	2
5	*Convolutional*	3
6	*Convolutional*	3
	Pooling	2
7	*Convolutional*	3
8	*Convolutional*	3
	Pooling	3
9	LSTM	
10	LSTM	

Tabel 3.1. Rincinan *Layer* pada CNN-LSTM

Setelah selesai melalui proses CNN dan LSTM, maka akan dilanjutkan dengan pengolahan menggunakan CTC. Hasil akhir dari proses CTC ini akan menghasilkan sebuah kata hasil prediksi gambar *input* dalam bentuk *string*. Untuk *Convolutional Layer*, *Pooling Layer*, LSTM dan CTC digunakan bantuan fungsi dari *library* Tensorflow.

3.2.2.2. Sistem *Training*



Gambar 3.14. *Flowchart* Sistem *Training*

Sistem pada *neural network* dibagi menjadi dua yaitu *training* dan implementasi. Sistem *training* digunakan untuk melatih *neural network* agar dapat mempelajari *input* agar dapat menghasilkan *output* yang diinginkan. Hasil dari sistem *training* berupa model yang akan dipanggil dan digunakan untuk sistem implementasi. Rincian akan apa saja yang dilakukan pada proses training ini dapat dilihat pada Gambar 3.14.

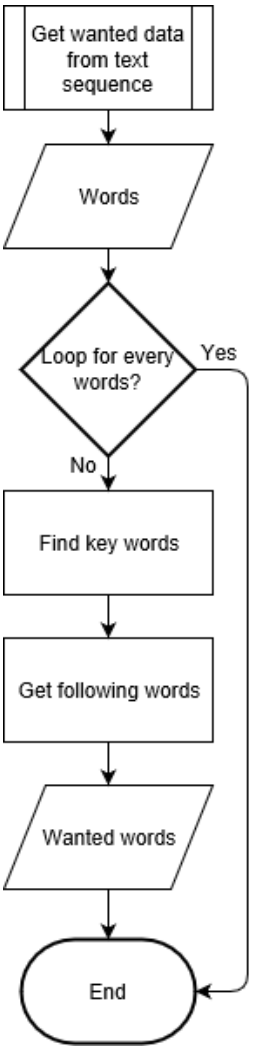
Sebelum memasukkan data pada sistem, data harus terlebih dahulu dikelompokkan sesuai dengan label kata yang terdapat pada data (*image*). Setelah itu akan dibuat sebuah tfrecord untuk memudahkan pembacaan data nantinya. Pada penelitian ini tfrecord yang digunakan berisikan beberapa data yaitu:

- *Image*/*encoded* : *image* yang di-*encode* dalam bentuk *string*
- *Image*/*height* : *integer* yang merepresentasikan tinggi dari *image*
- *Image*/*width* : *integer* yang merepresentasikan lebar dari *image*
- *Image*/label : *list* yang berisikan label karakter-karakter yang terdapat pada kata di *image*
- *Image*/*text* : *string* yang merepresentasikan kata yang terdapat dalam gambar

Tfrecord ini kemudian akan dibaca dan dimasukkan ke dalam *neural network*. Sebelum memulai proses *training* pada *neural network*, dilihat terlebih dahulu apakah ada *checkpoint* model terdahulu yang terdapat pada *storage*. Jika ada, maka akan di-*load* dan dilanjutkan proses *training* setelah *checkpoint* yang ada tersebut. Namun jika tidak ada, maka akan dibuat model baru dan memulai dari awal. Proses *training* ini dilakukan berkali-kali sesuai dengan jumlah *number of steps* yang diinginkan. Hasil akhir dari proses ini adalah model yang dapat digunakan untuk *testing* ataupun implementasi.

3.2.3. *Get Data from Text Sequence*

Pada proses ini akan dicari data mana saja yang diinginkan dari sekumpulan string hasil prediksi *neural network*, dimana pada penelitian ini yang diinginkan adalah nama lengkap, tanggal lahir, tempat lahir dan gender dari pemilik akta kelahiran. Karena *input* yang dimasukkan ke dalam *neural network* berurutan dari kata pertama pada baris pertama dan seterusnya, maka dapat dikatakan bahwa kata yang dihasilkan oleh *neural network* memiliki urutan yang sama seperti pada citra akta kelahiran yang di-*input* oleh *user* pertama kali.



Gambar 3.15. *Flowchart* Proses *Get Data from Text Sequence*

Sesuai yang terdapat pada Gambar 3.15, proses ini akan mencari kata-kata *keywords* yang biasanya muncul sebelum kata yang adalah informasi yang diinginkan. *Keyword* untuk informasi yang diinginkan berbeda-beda dan dapat memiliki jumlah yang tidak sama akibat banyaknya kemungkinan yang terjadi.

3.2.4. *User Interface*

Pada penelitian ini dibuat tampilan *user interface* untuk memudahkan pengguna menggunakan program. *User interface* dibuat menggunakan bahasa HTML dan Flask.

Gambar 3.16. Tampilan Utama pada *User Interface*

Seperti yang ada pada Gambar 3.16, pada tampilan utama terdapat sebuah tombol dan daerah untuk meng-*upload* gambar Akta Kelahiran yang ingin diproses dan dicari informasinya. Tombol "*Add Image*" jika ditekan dapat memunculkan *pop up window* dimana *user* dapat memilih *file* mana yang ingin di-*upload*. Tampilan *pop up window* ini juga akan muncul apabila *user* menekan bagian dalam dari kotak persegi panjang yang memiliki tulisan "*Drag and drop a file or select add image*". Selain menekan dan memilih dari *window, file image* juga dapat di *drag* dan diletakkan ke dalam daerah pada persegi panjang tersebut.

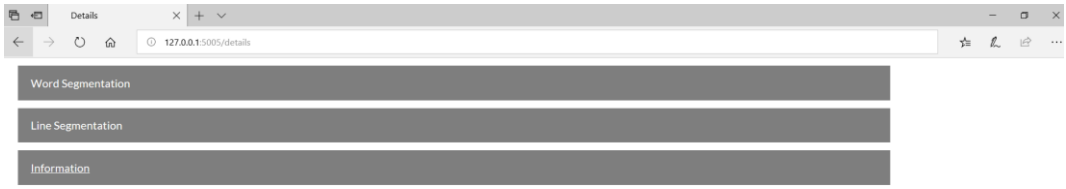
Setelah berhasil meng-*upload* gambar akan muncul tampilan Gambar 3.17. Terdapat tombol merah bertuliskan "*Remove* <nama *file*>", dimana jika ditekan maka akan menghapus gambar yang ter-*upload* dan tampilan akan kembali seperti tampilan utama pada Gambar 3.16. Terdapat pula tombol bertuliskan "*Process Image*" yang jika ditekan akan memulai proses pencarian informasi pada gambar yang telah di-*input*-kan tersebut.

Gambar 3.17. *Interface* Setelah *Upload* Gambar

Setelah proses pencarian informasi yang diinginkan selesai maka tampilan akan muncul seperti pada Gambar 3.18. Informasi nama, tanggal lahir, gender dan tempat lahir dalam bentuk teks yang terprediksi akan ditampilkan. Selain itu ada juga tombol "*View Details*" yang akan menampilkan halaman details.html.

Gambar 3.18. *Interface* Hasil Proses

Halaman details.html akan menampilkan tampilan seperti yang ada pada Gambar 3.19. Terdapat tiga menu yang ada yaitu "*Word Segmentation*", "*Line Segmentation*", dan "*Information*". Menu-menu tersebut dapat ditekan dan akan menampilkan gambar-gambar yang merupakan hasil proses yang telah dilakukan.

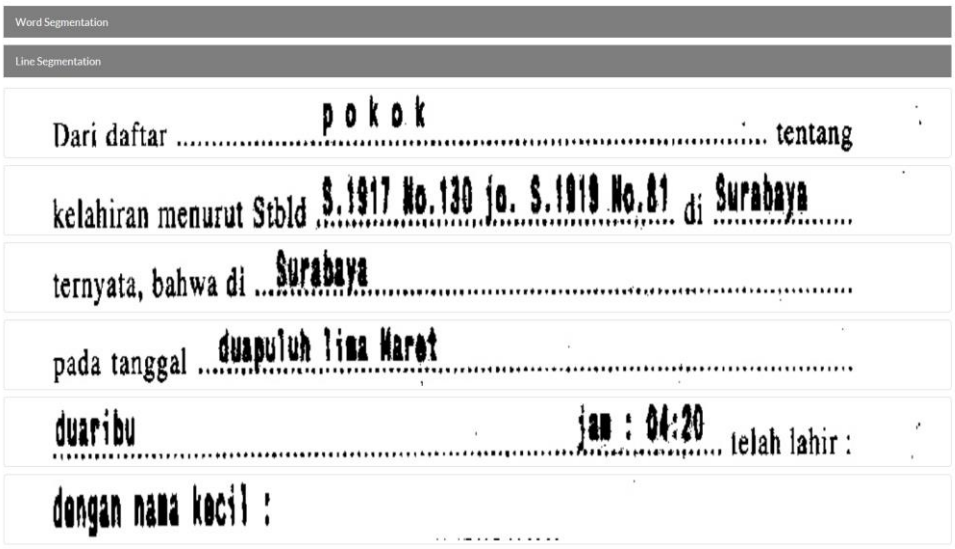


Gambar 3.19. *Interface* Result Details

Menu "*Word Segmentation*" jika ditekan akan menampilkan gambar-gambar hasil segmentasi kata yang telah dilakukan beserta dengan label kata hasil prediksinya. Tampilan dari menu ini dapat dilihat pada Gambar 3.20.

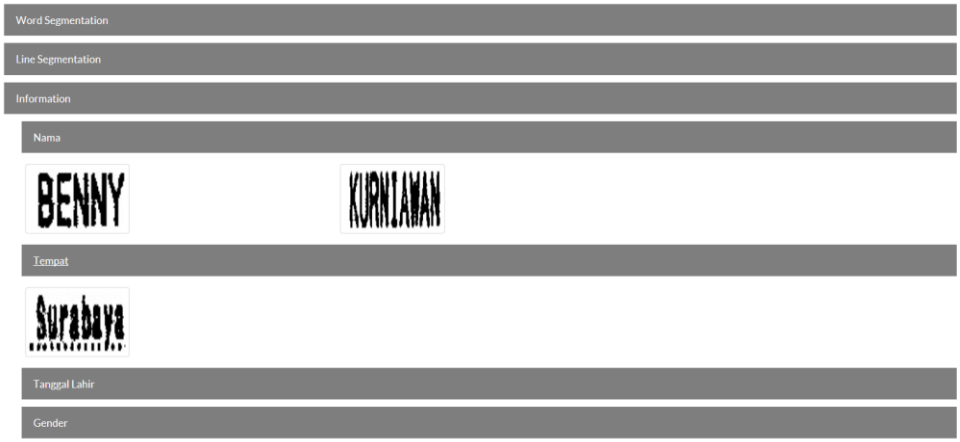
Gambar 3.20. *Interface* Menu *Word Segmentation*

Menu "*Line Segmentation*" berisi gambar-gambar hasil segmentasi baris yang berhasil dilakukan. Setiap baris akan ditampilkan secara vertikal kebawah seperti yang terdapat pada Gambar 3.21.



Gambar 3.21. *Interface* Menu *Line Segmentation*

Menu "*Information*" berisikan data gambar hasil prediksi untuk informasi-informasi yang dicari yaitu gambar hasil prediksi untuk nama, tempat, tanggal lahir dan gender. Setelah ditekan maka akan ditampilkan beberapa submenu yaitu "Nama", "Tempat", "Tanggal Lahir", dan "Gender". Masing-masing berisi gambar-gambar hasil segmentasi kata yang merepresentasikan kata yang telah diprediksi untuk informasi tersebut seperti yang terdapat pada Gambar 3.22.



Gambar 3.22. *Interface* Menu *Information*