**PROPOSAL**

**IDENTIFIKASI *LANDMARK* *SELLA*, *NASION* DAN *MENTON***

# *CEPHALOMETRY* MENGGUNAKAN METODE

***CONVOLUTIONAL NEURAL NETWORKS* (CNN)**

## Diajukan Untuk Memenuhi Salah Satu Syarat Mencapai Derajat Sarjana Komputer



**ASRIF FAJAR HIDAYAT**

**E1E1 16 066**

# JURUSAN TEKNIK INFORMATIKA

# FAKULTAS TEKNIK

# UNIVERSITAS HALU OLEO

# KENDARI

# 2020

# DAFTAR ISI

DAFTAR ISI ........................................................................................................ xii

DAFTAR GAMBAR ........................................................................................... xiv

DAFTAR TABEL ................................................................................................ xv

BAB I PENDAHULUAN................................................................................................. 1

1.1 Latar Belakang ............................................................................................. 1

1.2 Rumusan Masalah ......................................................................................... 3

1.3 Batasan Masalah ........................................................................................... 3

1.4 Tujuan Penelitian .......................................................................................... 4

1.5 Manfaat Penelitian ........................................................................................ 4

1.6 Sistematika Penulisan ................................................................................... 4

1.7 Tinjauan Pustaka ........................................................................................... 5

BAB II LANDASAN TEORI ................................................................................ 7

2.1 Ortodonti ..................................................................................................... 7

2.2 *Cephalometry* ............................................................................................... 7

2.2.1 Teknik pembuatan *cephalogram*  ........................................................ 8

2.2.2 *Landmark Cephalometry* ................................................................... 9

2.2.3 *Analisis cephalometry*  ...................................................................... 11

2.3 Citra *............................................................................................................* 11

2.4 *Citra BMP ...................................................................................................* 13

2.5 *Preprocessing Citra ....................................................................................* 14

2.5.1 *Resizing ............................................................................................* 14

2.5.2 *Contrast Limited Adaptive Histogram Equalization (CLAHE) .......* 15

2.5.3 *Cropping ..........................................................................................* 15

2.6 *Deep Learning ............................................................................................* 16

2.7 *Neural Network ............................................................................................* 16

2.7.1 Komponen *Neural Network* ............................................................. 17

2.7.2 Fungsi aktivasi ................................................................................. 18

2.8 *Convolutional Neural Network ...................................................................* 20

2.8.1 *Convolutional Layer* ........................................................................ 21

2.8.2 *Pooling Layer* ................................................................................... 22

2.8.3 *Multi Layer Neural Network* ............................................................ 23

2.8.4 *Softmax* *Layer* .................................................................................. 24

2.9 *Backpropagation Learning .........................................................................* 25

2.10 *Gradient Descent Learning .......................................................................* 27

2.11 *Dropout .....................................................................................................* 28

2.12 *K-fold Cross Validation .............................................................................* 28

2.13 Perhitungan akurasi *...................................................................................* 29

2.14 *Matlab (Matrix Laboratory) .....................................................................* 29

2.15 *Python .......................................................................................................* 30

2.15.1 *Tensorflow ......................................................................................* 30

2.16 *Pemodelan UML (Unified Modeling Language)........................................* 31

2.16.1 *Use Case* *Diagram* ......................................................................... 31

2.16.2 *Activity* *Diagram* ............................................................................. 32

2.16.3 *Sequence Diagram*  ......................................................................... 33

2.16.4 *Class Diagram*  ............................................................................... 34

2.17 Metodologi Pengembangan Sistem *...........................................................* 35

BAB III METODOLOGI PENELITIAN ............................................................. 37

3.1 Prosedur dan Pengumpulan Data ................................................................ 37

3.2 Metode Pengembangan Perangkat Lunak .................................................. 37

3.3 Waktu dan Tempat Penelitian .................................................................... 38

3.3.1 Waktu Penelitian .............................................................................. 38

3.3.2 Tempat Penelitian ........................................................................... 39

3.4 Analisis Sistem ............................................................................................ 40

3.5 Kebutuhan Non-fungsional ............................................................... 40

3.4.1 Kebutuhan perangkat keras ................................................... 40

3.4.2 Kebutuhan perangkat lunak ................................................... 41

3.5 Kebutuhan Fungsional .............................................................................. 41

3.5.1 Perancangan diagram alur sistem *.....................................................* 42

3.5.2 Perancangan Proses Citra Latih ....................................................... 43

3.5.3 Perancangan Proses Citra Uji *...........................................................* 43

3.5.4 Ektrasi Fitur Menggunakan *Convolutional Neural Networks* (CNN) *..................................................................................................................* 44

4.3.4.1 *Convolutional Layer .........................................................* 44

4.3.4.2 Fungsi Aktivasi *..................................................................* 45

4.3.4.3 *Pooling Layer ....................................................................* 46

4.3.4.4 *Fully Connected Layer .......................................................* 46

4.4 Perancangan *Unified* *Modelling* *Language ................................................* 47

4.4.1 *Use Case Diagram .........................................................................* 47

4.4.2 *Activity Diagram*  *...........................................................................* 52

4.4.3 *Sequence Diagram ........................................................................* 54

4.4.4 *Class Diagram ...............................................................................* 55

*4.5* Perancangan *Interface ................................................................................* 56

DAFTAR PUSTAKA .......................................................................................... 72

# DAFTAR GAMBAR

Gambar 2.1 Lateral cephalometry ......................................................................... 7

Gambar 2.2 Landmark cephalometry ................................................................... 11

Gambar 2.3 Citra fungsi dua variable .................................................................. 12

Gambar 2.4 Ilustrasi Proses *Resize* ...................................................................... 15

Gambar 2.5 Ilustrasi Proses *Cropping*  ................................................................. 16

Gambar 2.6 Satu unit neuron pada jaringan syaraf tiruan ..................................... 17

Gambar 2.7 Fungsi aktivasi ReLu ........................................................................ 18

Gambar 2.8 Fungsi aktivasi sigmoid .................................................................... 19

Gambar 2.9 Fungsi aktivasi tanh .......................................................................... 19

Gambar 2.10 Struktur *Convolutional* *Neural* *Network* ........................................ 21

Gambar 2.11 Proses konvolusi pada input array 2D dengan bobot 2D ............... 22

Gambar 2.12 Proses *Max*-*Pooling* ....................................................................... 23

Gambar 2.13 *Neural* *Network* satu lapisan tersembunyi ..................................... 24

Gambar 2.14 Jaringan *backpropagation* .............................................................. 26

Gambar 2.15 *K-fold Cross Validation* .................................................................. 29

Gambar 4.1 Perancangan sistem .......................................................................... 43

Gambar 4.2 Perhitungan *Convolutional Layer* .................................................... 44

Gambar 4.3 Fungsi Aktivasi ReLu ...................................................................... 45

Gambar 4.4 *Max* *Pooling Layer* 2x2 *Stride 1* ....................................................... 46

Gambar 4.5 *Fully connected layer* ...................................................................... 47

Gambar 4.6 *Use* *Case* *Diagram* ........................................................................... 47

Gambar 4.7 *Activity* *Diagram* .............................................................................. 53

# DAFTAR TABEL

Tabel 2.1 Simbol-simbol pada *Use Case Diagram* ............................................... 32

Tabel 2.2 Simbol-simbol pada *Activity Diagram* ................................................. 33

Tabel 2.3 Simbol-simbol pada *Sequence Diagram* .............................................. 34

Tabel 2.4 Simbol-simbol pada *Class Diagram*  .................................................... 35

Tabel 3.1 Waktu Penelitian .................................................................................. 39

Tabel 4.1 Spesifikasi perangkat keras .................................................................. 41

Tabel 4.2 Spesifikasi perangkat lunak ................................................................. 41

# BAB 1 PENDAHULUAN

## 1.1. Latar Belakang

Ilmu kedokteran adalah cabang ilmu kesehatan yang mempelajari tentang cara mempertahankan kesehatan manusia, dan mengembalikan manusia pada keadaan sehat dengan memberikan pengobatan pada penyakit, dan cedera. Ilmu ini meliputi pengetahuan tentang sistem tubuh manusia, dan penyakit serta pengobatannya, dan penerapan dari pengetahuan tersebut. Ilmu kedokteran kemudian terbagi lagi menjadi beberapa cabang ilmu pengetahuan lainnya, yang salah satunya yaitu Ortodonti. Ortodonti merupakan cabang ilmu kedokteran gigi yang mempelajari kelainan pertumbuhan dan perkembangan gigi-geligi dan wajah (dentofasial). Dalam bidang Ortodonti, salah satu alat yang sangat penting untuk membantu dalam diagnosis dan perawatan adalah *cephalometry*. *cephalometry* dapat didefinisikan sebagai ilmu yang mempelajari pengukuran-pengukuran yang bersifat kuantitatif terhadap bagian-bagian tertentu dari kepala untuk mendapatkan informasi tentang pola kraniofasial (Ardhana, 2011). Analisis *cephalometry* diperoleh daripemeriksaan penunjang yaitu radiografi *cephalometry,* seperti *cephalogram lateral* atau *cephalogram anteroposterior.* Dalam menganalisis hubungan gigi, kranium, dan jaringan lunak, umumnya *orthodontist* menggunakan radiografi *cephalogram lateral*.

Analisis *cephalometry* pada dasarnya dilakukan Secara manual, akan tetapi metode ini membutuhkan waktu yang lama terutama jika analasis dilakukan pada data dengan jumlah yang banyak (Munandar dkk., 2018). Secara manual, analisis *cephalometry* dibuat pada gambar hasil penampakan *cephalogram*. Kertas asetat dipakai untuk penampakan *cephalogram*. Kertas asetat dilekatkan pada tepi atas *cephalogram* dengan *scotch tape*, kemudian diletakkan di atas iluminator (*negatoscope*). Penampakan *cephalogram* dianjurkan menggunakan pensil keras (4H) agar diperoleh garis- garis yang cermat dan tipis. Karena proses analisis *cephalometry* secara manual membutuhkan waktu yang lama, maka beberapa penelitian telah mencoba untuk melakukan analisis *cephalometry* berbasis komputer dengan menggunakan citra digital *cephalogram* dalam mengidentifikasi *landmark cephalometry*. *Landmark* *cephalometry* adalah titik yang mudah dikenali pada radiografi *cephalometry* atau *tracing cephalometry*, yang mewakili struktur anatomi jaringan keras atau lunak tertentu (*landmark* anatomi) dan (*constructed landmarks*) (Tony, 2016).

(Arik dkk., 2017) melakukan penelitian tentang *Fully Automated Quantitative Cephalometry Using Convolutional Neural Networks* dengan hasil pengujian cukup baik yaitu dengan rata-rata persentase akurasi data yang diidentifikasi sebesar 76%. Dari hasil pengujian tersebut, teridentifikasi 19 *landmark cephalometry*, diantaranya yaitu *Sella, Nasion, Orbitale, Porion, Subspinale, Supramentale, Pogonion, Menton, Gnathion, Gonion, Lower incisal incision, Upper incisal incision, Upper lip, Lower lip, Subnasale, Soft tissue pogonion, Posterior nasal spine, Anterior nasal spine, dan Articulate*.

Salah satu metode yang sering digunakan dalam mengidentifikasi *landmark cephalometry* secara otomatisadalah *convolutional neural networks* (CNN). *Convolutional neural networks* (CNN) merupakan salah satu metode *deep learning* dari pengembangan *multi layer perceptron* (MLP) yang didesain untuk mengolah data dua dimensi. *Convolutional neural networks* (CNN) sangat cocok digunakan dalam aplikasi *image processing* dan *recognition* karena CNN mengeksploitasi korelasi spasial lokal dengan menggunakan pola konektivitas lokal. CNN telah berhasil diterapkan dalam berbagai aplikasi, termasuk klasifikasi gambar, segmentasi gambar, penyelarasan gambar, deteksi wajah, estimasi sikap manusia, dan deteksi jalur. Dalam beberapa aplikasi ini, kinerja CNN bahkan telah melampaui kinerja manusia. Namun, demonstrasi yang berhasil sebagian besar terbatas pada area di mana terdapat banyak data yang tersedia untuk pelatihan. Adaptasi teknik CNN yang berhasil untuk pencitraan medis sangat menantang, khususnya, karena ukuran kecil dari *dataset* gambar medis yang tersedia untuk pelatihan (Arik dkk., 2017).

Berdasarkan latar belakang tersebut, dalam penelitian ini akan dibuat sistem “**Identifikasi *Landmark Sella, Nasion* dan *Menton Cephalometry* menggunakan metode *Convolutional neural networks* (CNN)”** untuk memperoleh hasil identifikasi *landmark* secara otomatis.

## 1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya rumusan masalah yang dapat dipaparkan yaitu bagaimana mengidentifikasi *landmark sella, nasion* dan *menton cephalometry* menggunakan metode *convolutional neural networks* (CNN).

## 1.3. Batasan Masalah

Hal-hal yang menjadi batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Data citra digital *cephalogram* berjumlah 120 data dengan format BMP diambil dari *database* pasien RSGM Prof. Soedomo, Universitas Gadjah Mada. Seluruh citra tersebut dibagi menjadi 2 kelompok, yaitu 108 sebagai citra latih dan 12 sebagai citra uji. Pengelompokkan ini dilakukan secara *random* dengan menganggap bahwa semua data citra memenuhi standar untuk analisis *cephalometry*.
2. Citra digital *cephalogram* berasal dari pasien normal, dalam artian pasien tersebut tidak mengalami kelainan seperti cacat fisik bawaan sejak lahir, kecelakaan, penyakit, dan lain sebagainya.
3. Citra digital *cephalogram* yangdigunakan adalah *lateral cephalometry*.
4. *Landmark* yang akan diidentifikasi adalah *landmark* *sella, nasion* dan *menton*.

## 1.4. Tujuan Penelitian

Tujuan dari penelitian ini yaitu untuk mengidentifikasi *landmark sella, nasion* dan *menton cephalometry* menggunakan metode *convolutional neural networks* (CNN).

## 1.5. Manfaat Penelitian

Manfaat dari penelitian ini yaitu diharapkan dapat memberikan hasil identifikasi *landmark sella, nasion* dan *menton* dengan benar sehingga memberi kemudahan bagi para dokter gigi dalam melakukan analisis *cephalometry*.

**1.6. Sistematika Penulisan**

Adapun sistematika penulisan laporan Tugas Akhir ini adalah sebagai berikut:

## BAB I PENDAHULUAN

Bab ini berisi penjelasan mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, serta sistematika penulisan dalam laporan tugas akhir.

## BAB II LANDASAN TEORI

Bab ini berisi mengenai landasan teori yang berhubungan dengan penelitian yang dilakukan dalam skripsi ini, pembahasan-pembahasan tersebut meliputi Citra, *Preprocessing* Citra, metode *convolutional neural networks* (CNN)*,* dan metodologi pengembangan sistem Matlab (*Matrix Laboratory*)*, Flowchart,* Pemodelan UML (*Unified Modeling Language*), dan metodologi pengembangan sistem *Rational Unified Process* (RUP).

## BAB III METODE PENELITIAN

Bab ini memuat metodologi penelitian yang meliputi prosedur pengumpulan data, uraian metode *Rational Unified Process* (RUP) untuk pengembangan system, uraian waktu penelitian, *Flowchart,* Pemodelan UML (*Unified Modeling Language*), dan desain *user interface*.

### 1.7. Tinjauan Pustaka

Beberapa penelitian yang dijadikan rujukan dalam penelitian ini yaitu:

Penelitian pertama yang berhasil peneliti temukan yaitu penelitian yang dilakukan oleh Razi (2017) dengan judul Klasifikasi Artikel Berita Berbahasa Indonesia Menggunakan *Convolutional* *Neural* *Network*. Dalam penelitian tersebut digunakan metode *Convolutional* *Neural* *Network* (CNN) untuk mengklasifikasikan artikel berita berbahasa indonesia. Terdapat lima kelas dalam pengklasifikasian, yaitu *entertainment*, kesehatan, olahraga, teknologi, dan ekonomi. Sebelum dilakukan klasifikasi, terlebih dahulu dilakukan pengubahan kata-kata ke dalam bentuk vektor dengan menggunakan *word2vec* sehingga hasil dari pengubahan tersebut dapat di-*input-*kan ke dalam CNN. Hasil pengujian menunjukkan bahwa kombinasi penggunaan metode *Convolutional* *Neural* *Network* dan *word2vec* memberikan hasil nilai akurasi sebesar 96.70 %, dan presisi, *recall* serta *f-measure* mencapai 96,60 %.

Penelitian kedua yaitu penelitian yang dilakukan oleh (Zufar & Setiyono, 2016) dengan judul *Convolutional* *Neural* *Networks* untuk Pengenalan Wajah Secara *Real*-*Time*. Penelitian ini menggunakan mengimplementasikan metode CNN dan bantuan *library* OpenCV untuk deteksi multi wajah dan perangkat *Webcam* M-Tech 5MP. Hasil uji coba dengan menggunakan konstruksi model CNN sampai kedalaman 7 lapisan dengan *input* dari hasil ekstraksi *Extended* *Local* *Binary* *Pattern* dengan radius 1 dan neighbor 15 memperlihatkan kinerja pengenalan wajah meraih rata-rata tingkat akurasi lebih dari 89% dalam  2 *frame* per detik.

Penelitian yang ketiga dilakukan oleh (Subeom Park, 2017) dengan judul *Cephalometric Landmarks Detection using Fully Convolutional Networks*. Pada penelitian tersebut, mengajukan 150 data latih dan 250 data uji, hasil dari dari pengajuan model *convolutional neural networks* (CNN) berupa *heatmap,* dan mendapatkan hasil akurasi lebih dari 90% pada *heatmap*.

Penelitian yang keempat dilakukan oleh (Limantoro dkk., 2017) dengan judul Deteksi Pengendara Sepeda Motor Menggunakan *Deep Convolutional Neural Networks*. Tujuan dari penelitian ini adalah memahami arsitektur CNN dengan hasil yang terbaik. Pada penelitian tersebut, *Dashcam* digunakan untuk merekam video di jalan raya Surabaya pada siang hari atau cuaca cerah dengan resolusi 1080 HD. *Video* dengan warna RGB diekstrak menjadi beberapa citra dan diberi label. Lalu, data citra dilatih dengan *convolutional neural networks* menghasilkan bobot. Eksperimen yang dilakukan adalah membagi arsitektur CNN menjadi tiga, yaitu kecil, sedang, dan besar. Lalu, arsitektur CNN tersebut saling dibandingkan. Pada hasil eksperimen, arsitektur CNN besar memiliki akurasi yang tertinggi sebesar 93% dan f1-*score* sebesar 94,9%. Arsitektur CNN yang kompleks dan besar cenderung memiliki performa yang lebih baik.

Penelitian kelima yaitu oleh (Danukusumo, 2017) dalam tesisnya dengan judul Implementasi *Deep* *Learning* Menggunakan *Convolutional* *Neural* *Network* Untuk Klasifikasi Citra Candi Berbasis Gpu. Penelitian ini menggunakan citra candi sebagai data *input*. Hasil pengujian menunjukan akurasi sebesar 98,99% pada training set dan 85,57% pada *test* *set* dengan waktu pelatihan mencapai 389,14 detik. Sehingga dapat disimpulkan bahwa klasifikasi citra candi dapat dilakukan dengan sangat baik menggunakan teknik *Deep* *Learning* dengan CNN.

Dari kelima penelitian tersebut, dapat diketahui bahwa metode *deep learning convolutional neural networks* (CNN) dapat digunakan mengidentifikasi *landmark sella, nasion* dan *menton* pada citra *cephalometry*.

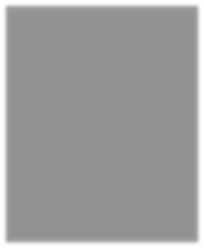
# BAB II LANDASAN TEORI

## 2.1. Ortodonti

Ortodonti adalah salah satu cabang ilmu kedokteran gigi yang mempelajari pertumbuhan, perkembangan, variasi wajah, rahang dan gigi serta perawatan perbaikannya untuk tercapainya oklusi normal (Harty dan Ogston, 2012). Istilah ortodontik berasal dari kata Ortodonsia. Ortodonsia berasal dari bahasa Yunani (*Greek*) yaitu *Orthos* (baik, betul) dan *Dons* (gigi). Jadi ortodonti dapat diterjemahkan sebagai ilmu pengetahuan yang bertujuan memperbaiki atau membetulkan letak gigi yang tidak teratur atau tidak rata. Lebih dari itu ortodonti juga berhubungan dengan pertumbuhan dan perkembangan bentuk wajah dan gigi geligi.(Gill Daljit, 2008)

## 2.2. *Cephalometry*

*Cephalometry* adalah pengukuran secara ilmiah dimensi kepala (Dorland, 2002). *Lateral* *cephalometry* adalah sebuah pandangan sagital dari struktur *skeletal* dan *facial* pasien yang diambil dengan cara standar, dengan kepala pasien ditempatkan di sebuah *cephalostat*. Melalui gambaran *cephalogram lateral* dapat diperoleh informasi tentang keadaan pasien sebelum, selama, dan sesudah perawatan.



## Gambar 2.1 *Lateral* *cephalometry* (2014 *IEEE Challenge*)

*Cephalometry*radiograf memungkinkan dokter untuk mengukur hubungan Wajah dan gigi sehingga penilaian sejauh mana pasien menyimpang dari morfologi normal wajah dan gigi menjadi lebih akurat. Agar analisis pengukuran radiograf *cephalometry* memperoleh hasil yang akurat, kualitas citra *cephalometry* dari hasil radiograf harus dapat dibandingkan. Artinya, dengan kejelasan dan ketepatan gambar proyeksi objek yang diinginkan memenuhi persyaratan untuk dianalisis. Dengan adanya teknik ini maka dapat diperoleh studi yang berkelanjutan untuk menilai pertumbuhan dan perkembangan tulang *facial* serta untuk menganalisis aktifnya hasil perawatan ortodonti atau koreksi prosedur bedah pada wajah yang abnormal (Kamal dkk., 2020)

### 2.2.1. Teknik Pembuatan Cephalogram

Teknik pembuatan *cephalogram* (Jacobson, 1995) yaitu posisi pasien disesuaikan dengan *cephalostat* menggunakan *ear rods bilateral* yang ditempatkan dalam setiap *auditory meatus*, biasanya ketika pasien berada dalam posisi berdiri. Bidang *midsagital* pasien, vertikal dan tegak lurus terhadap sinar-X, sejajar dengan bidang film, tegak lurus terhadap sinar-X. Bidang *Frankfort* pasien (garis yang menghubungkan batas *superior* dari *auditory* *meatus* eksternal dan *infraorbital* *rim*) sejajar dengan lantai.

Karena *photon* sinar-X berasal dari *tube-head x-ray source* dalam pola yang berbeda, selalu ada jumlah perbesaran objek yang berbeda-beda dalam setiap sinar *rontgen*. Tingkat perbesaran ditentukan oleh rasio dari jarak sumber sinar-X ke objek dan jarak sumber sinar-X ke film. Semakin besar jarak dari sumber sinar-X ke bidang film, semakin besar perbesaran. Untuk meminimalkan efek ini, jarak dari sumber sinar-X ke bidang *midsagittal* kepala pasien pada unit *cephalometry* adalah 5 kaki. Hal ini memastikan bahwa photon sinar-X berjalan ke arah objek/film lebih sejajar satu sama lain, sehingga mengurangi pembesaran.

Sebagaimana dijelaskan sebelumnya, faktor pembesaran dipengaruhi oleh jarak dari kaset film ke bidang *midsagittal* pasien, dengan peningkatan perbesaran karena film dipindahkan. Untuk mengurangi variasi dalam perbesaran dan untuk memperoleh pengukuran yang konsisten pada pasien yang sama dari waktu ke waktu, banyak ortodontis memilih untuk menjaga jarak konstan. Jarak 15 cm dari bidang *midsagittal* *cephalostat* ke kaset film sering digunakan. Jarak ini menghasilkan perbesaran yang agak konsisten, dalam batas toleransi, dan memungkinkan untuk rata-rata lebar kepala pasien. Namun, banyak praktisi memilih untuk menempatkan kaset film dekat dengan kepala pasien, mungkin untuk mencapai ketajaman maksimal dan mengurangi perbesaran dari struktur gigi (Kamal dkk., 2020).

### 2.2.2. Landmark Cephalometry

Secara etimologi, *landmark* merupakan gabungan dari dua kata yaitu *land* dan *meark/mark* yang berarti struktur anatomi yang sudah dikenal, digunakan sebagai titik pedoman untuk menentukan lokasi struktur lainnya atau menentukan ukuran tertentu (Dorland, 2002). *Landmark* *cephalometry* adalah titik-titik penting pada radiograf kepala yang mana garis, bidang, dan sudut mungkin dibangun untuk menganalisis bentuk dan hubungan antara elemen-elemen tulang *craniofacial* (Farlex, 2009). Idealnya letak suatu *landmark* harus mudah diketahui, mempunyai relevansi anatomi dan tidak dipengaruhi oleh proses pertumbuhan. *Landmark* *cephalometry* dibagi atas *hard* *tissue* *points* (titik-titik pada jaringan keras) dan *soft* *tissue* *points* (titik-titik pada jaringan lunak).

1. *Hard Tissue Points*

Merupakan *landmark* *cephalometry* yang terletak pada jaringan keras. *Hard* *tissue* *points* terbagi menjadi dua yaitu titik-titik median (anatomi *landmark*) dan titik-titik bilateral (*derived* *landmark*). Adapun titik-titik median (anatomi *landmark*) yaitu:

* 1. *Gnation* (Gn): terletak di antara *Pogonion* dan *Menton* pada kontur luar dagu.
  2. *Incisive* *superius* (Is): ujung mahkota paling *anterior* pada *maxillary* *central* *incisor*.
  3. *Incisive* *inferius* (Ii): ujung mahkota paling *anterior* pada *mandibular* *central* *incisor*.
  4. *Menton* (Me): titik paling *inferior* dari simfisis atau titik paling bawah dari *mandibula*.
  5. *Nasion* (N): terletak pada bagian paling *inferior*, titik *anterior* pada tulang *frontal* yang berdekatan dengan *frontonasal*.
  6. *Pogonion* (Pog): terletak pada titik paling *anterior* pada bayangan dagu.
  7. *Point* A (A): terletak pada bagian paling *posterior* dari bayangan *anterior* *maxilla*, biasanya dekat puncak pusat akar gigi seri tengah.
  8. *Point* B (B): terletak pada titik paling *posterior* bayangan perbatasan *anterior* *mandibula*, biasanya dekat puncak pusat akar gigi seri.
  9. *Sella* (S): terletak di tengah garis *fosa* *hipofisis* (*sella* *turcica*).

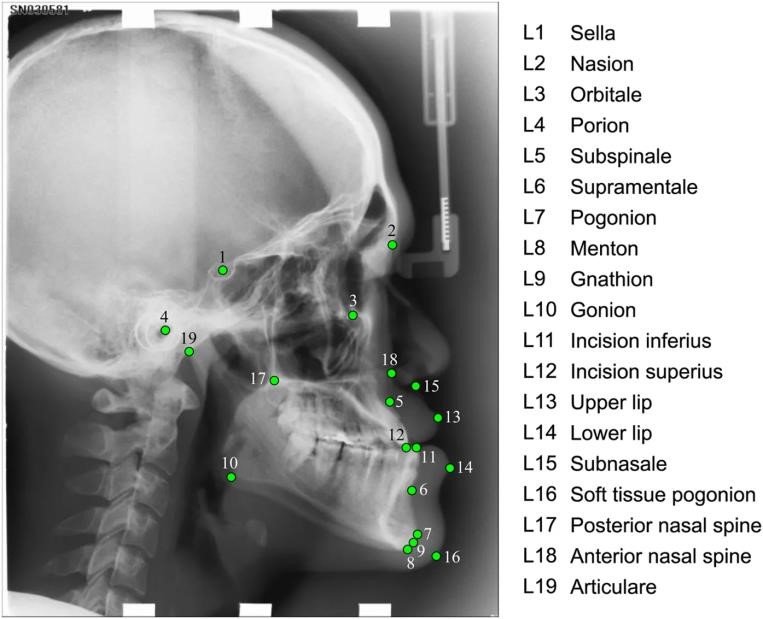
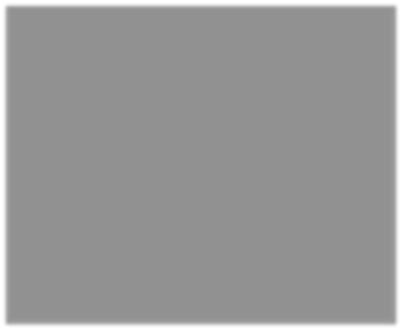
Adapun titik-titik bilateral (*derived* *landmark*) terdiri dari:

* 1. *Porion* (Po): titik tengah dari tepi atas dari *porus* *acusticus* *externus* ditunjukkan oleh sumbu metal dari *cephalometer* (BJORK).
  2. *Gonion* (Go): terletak di antara titik paling *inferior* dan titik paling *posterior* dari sudut *mandibula*.
  3. *Orbital* (Or) : titik yang paling bawah pada tepi bawah tulang orbita.

1. *Soft Tissue Points*

Merupakan *landmark* *cephalometry* yang terletak pada jaringan lunak. Beberapa *soft* *tissue* *points* yaitu:

* 1. *Soft* *tissue* *glabella* (G): adalah titik yang paling menonjol dalam bidang *midsagittal* dahi.
  2. *Pronasale* (Pr): adalah titik yang paling menonjol di ujung hidung.
  3. *Labrale* *superius* (Ls): adalah titik median dalam margin atas pada membran atas bibir.
  4. *Labrale* *inferius* (Li): adalah titik median di margin bawah pada membran bawah bibir.
  5. *Soft tissue pogonion* (Pog’) : adalah titik yang paling menonjol pada titik pada jaringan lunak kontur dagu.



## Gambar 2.2. *Landmark cephalometry* (Lindner dkk, 2016)

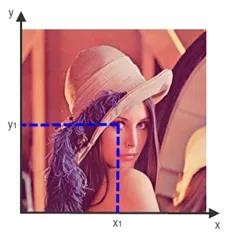
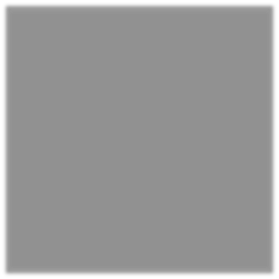
### 2.2.3. Analisis Cephalometry

Analisis *cephalometry* merupakan sistem pengukuran yang diciptakan untuk mengetahui hubungan antara berbagai elemen skeletal, dental dan jaringan lunak dalam kompleks *cranio*-*facial* (McNamara, 1984). Analisis *cephalometry* digunakan untuk menentukan rencana perawatan dan dan evaluasi keberhasilan perawatan ortodonti. Beberapa analisis *cephalometry* dibuat terutama untuk memberi gambaran *geometric* yang jelas dari penderita dan deviasinya dari tujuan perawatan yang ideal. Dalam analisis tersebut, tujuan perawatan dapat dilihat secara grafis dan dengan angka-angka dari perseorangan dan bukan merupakan *population* *means*. Unit dasar dari analisis *cephalometry* adalah pengukuran sudut (*angular* *analysis*) dan pengukuran garis (*linear* *analysis*).

### 2.3. Citra

Citra adalah gambar dua dimensi hasil konversi dari gambar analog yang kontinu menjadi gambar diskrit melalui proses *sampling* dengan membagi gambar analog menjadi *N* baris dan *M* kolom sehingga menjadi gambar diskrit. Persilangan antara baris dan kolom tertentu disebut dengan *pixel*.

Citra digital adalah citra yang dinyatakan secara diskrit (tidak kontinu), baik untuk posisi koordinatnya maupun warnanya. Dengan demikian, citra digital dapat digambarkan sebagai suatu matriks, di mana indeks baris dan indeks kolom dari matriks menyatakan posisi suatu titik di dalam citra dan harga dari elemen matriks menyatakan warna citra pada titik tersebut. Dalam citra digital yang dinyatakan sebagai susunan matriks, elemen–elemen matriks tadi disebut juga dengan istilah *pixel* yang berasal dari kata *picture element*. Citra digital dapat didefinisikan sebagai fungsi dua variabel *f(x,y)* dengan *x* dan *y* adalah koordinat spasial sedangkan nilai *f(x,y)* adalah intensitas citra pada koordinat tersebut, hal itu diilustrasikan pada Gambar 2.3 berikut:



## Gambar 2.3 Citra fungsi dua variable (Agustian, 2013)

Citra digital tersusun atas titik-titik yang dapat berbentuk persegi panjang dan secara beraturan membentuk baris-baris dan kolom-kolom. Setiap titik memiliki koordinat dan dapat dinyatakan dalam bilangan bulat positif, yaitu 0 atau 1 bergantung pada sistem yang digunakan. *Format* nilai *pixel* sama dengan format citra keseluruhan. Pada kebanyakan sistem pencitraan, nilai ini dapat berupa bilangan bulat positif. *Format* citra digital yang banyak digunakan, yaitu:

1. Citra biner (*Monokrom*)

Citra monokrom atau citra hitam-putih merupakan citra di mana *f(x,y)* merupakan fungsi tingkat keabuan dari hitam ke putih.

1. Citra skala keabuan (*Grayscale*).

Citra *grayscale* dikatakan *format* citra skala keabuan karena pada umumnya warna yang dipakai adalah warna hitam sebagai warna minimum dan warna putih sebagai warna maksimumnya, sehingga warna antara keduanya adalah abu-abu.

1. Citra berwarna

Citra warna terdiri atas 3 *layer* matriks, yaitu R-layer, G-layer, B-layer. sistem warna RGB (*Red, Green, Blue*) menggunakan sistem tampilan grafik kualitas tinggi (*High quality raster graphic*) yaitu mode 24 bit. Setiap komponen warna merah, hijau, biru masing-masing mendapatkan alokasi 8 bit untuk menampilkan warna. Pada sistem warna RGB, setiap pixel akan dinyatakan dalam 3 parameter dan bukan nomor warna. Setiap warna mempunyai *range* nilai 00 (angka desimalnya adalah 0) dan f (angka desimalnya 255) atau mempunyai nilai derajat keabuan 256 = 28. Dengan demikian, *range* warna yang digunakan adalah (28)(28)(28) = 232 (atau dikenal dengan istilah *True* *color* pada *Windows*). Nilai warna yang digunakan merupakan gabungan warna cahaya merah, hijau dan biru

### 2.4. Citra BMP

BMP merupakan *file* gambar yang tidak terkompresi. Dan karena itu ukuran *file* BMP biasanya besar. Di internet sendiri apabila terdapat *File* berukuran besar tidak tepat untuk melakukan proses transfer karena akan memenuhi [*bandwidth* d](https://www.nesabamedia.com/pengertian-bandwidth-beserta-fungsinya-dan-cara-kerja-bandwidth/)an menjadi lambat.

Tipe file BMP yang umum digunakan pada sistem operasi Windows dan OS/2. File BMP memiliki ukuran yang jauh lebih besar dari pada tipe-tipe yang lain. Kelebihan tipe *file* BMP adalah dapat dibuka oleh hampir semua program pengolah gambar. Baik file BMP yang terkompresi maupun tidak terkompresi.

*Bitmap* merupakan representasi dari citra grafis yang terdiri dari susunan titik - titik yang tersimpan di dalam [memori komputer.](https://www.nesabamedia.com/pengertian-memori/) Dikembangkan oleh Microsoft dan nilai setiap titik diawali oleh satu bit data untuk gambar hitam putih dan untuk gambar berwarna lebih dari satu bit. Ukuran sebenarnya adalah n-bit (2n warna) *bitmap* dalam hitungan *byte*.

### 2.5. Preprocessing Citra

Pada proses identifikasi *landmark*, citra yang digunakan masih berupa perkiraan kasar atau masih memiliki kualitas yang cukup buruk seperti ukuran yang sangat besar, kejelasan citra yang buruk dan sebagainya. Maka perlu diadakan proses penyelarasan. Proses ini terdiri dari tahapan-tahapan sebagai berikut:

#### 2.5.1. Resizing (tahap normalisasi dimensi citra)

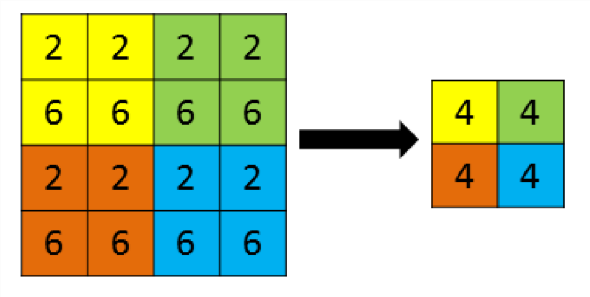
Pada tahap *resizing* citra, terjadi proses normalisasi dimensi citra objek, yaitu proses pembesaran atau pengecilan dimensi citra objek menjadi dimensi yang telah ditentukan. Tujuannya, untuk menyamakan dimensi objek dari tiap citra yang dimasukkan, sehingga pada proses ekstraksi citra nanti tidak ada perbedaaan dimensi dari matriks data citra objek.

Ukuran baru hasil penskalaan didapat melalui perkalian antara ukuran citra *input* dengan variabel penskalaan (Putra, 2010). Rumus yang digunakan untuk proses *scaling* ditunjukkan pada Persamaan 2.1 dan Persamaan 2.2.

*P0 = Sp x Pi* (2.1)

*L0 = Sl x Li* (2.2)

Dengan (Pi, Li) adalah ukuran citra *input*, (P0, L0) adalah ukuran citra output dan (Sp, Sl) adalah variabel penskalaan yang diinginkan. Jika variabel penskalaan bernilai lebih besar dari 1 maka hasil penskalaannya akan memperbesar ukuran citra, sebaliknya apabila variabel penskalaannya lebih kecil dari 1 maka hasilnya akan memperkecil ukuran citra. Gambar 2.4 merupakan ilustrasi proses *resize*.



## Gambar 2.4 Ilustrasi Proses *Resize* (Putra, 2010)

### 2.5.2. Contrast Limited Adaptive Histogram Equalization (CLAHE)

CLAHE termasuk teknik perbaikan citra yang digunakan untuk memperbaiki kontras pada citra. CLAHE memperbaiki *local* *contrast* pada citra dan CLAHE merupakan generalisasi dari *Adaptive* *Histogram* *Equalization* (AHE). Berbeda dengan *Histogram* *Equalization* yang beroperasi pada region kecil pada citra *grayscale* yang disebut dengan *tile*. Kontras pada setiap *file* diperbaiki sehingga *histogram* yang dihasilkan dari *region* tersebut kira-kira cocok dengan bentuk *histogram* yang ditentukan. *Tile* yang saling bertetangga disambungkan dengan menggunakan interpolasi *bilinear*. Hal ini dilakukan agar hasil penggabungan *tile* terlihat halus. Kontras, terutama pada area yang *homogeny*, dapat dibatasi untuk menghindari penguatan derau yang mungkin terdapat dalam citra.

### 2.5.3. Cropping

Pada tahapan ini terjadi pemotongan citra yang memisahkan citra objek dengan citra masukannya, tujuannya untuk mengambil citra yang hanya diperlukan untuk proses ekstraksi, dalam hal ini adalah citra objek dan membuang citra lain yang tidak diperlukan. Dimensi citra yang dipotong disesuaikan dengan dimensi dari proses segmentasi atau pengkotakan objek yang dilakukan pada proses pendeteksian objek.

*Imcrop* merupakan fungsi pada MATLAB untuk melakukan *cropping*. *Cropping* adalah memotong bagian tertentu dari citra sehingga diperoleh area citra yang akan diproses. Operasi ini pada dasarnya adalah operasi translasi, yaitu menggeser koordinat titik citra. Persamaan yang digunakan untuk operasi *cropping* adalah sebagai berikut:

*x*’=*x-x*L untuk *x*=*x*L sampai *x*R  *y*’=*y*-*yr* untuk *y*=*y*T sampai *y*B (2.3)

(xL, yT) dan (xR, yB), masing-masing adalah koordinat titik pojok kiri atas dan pojok kanan bawah bagian citra yang hendak di-crop (Achmad dan Firdausy, 2005). Adapun ilustrasi proses cropping dapat dilihat pada Gambar 2.5.



**Citra Asli Koordinat *cropping* citra Hasil *cropping* citra**

## Gambar 2.5 Ilustrasi Proses *Cropping* (Achmad & Firdausy, 2013)

### 2.6. Deep Learning

Teknologi *Machine* *Learning* menguatkan banyak aspek dari masyarakat yang modern mulai dari pencarian sebuah *website* hingga penyaringan konten pada jaringan sosial hingga rekomendasi di situs *web* *e-commerce*, dan semakin hadir dalam produk konsumen seperti kamera dan *smartphone*. Sistem *machine* *learning* digunakan untuk mengidentifikasi objek dalam gambar, mencocokkan item berita, mengubah ucapan menjadi kata, memilih hasil pencarian yang relevan, dan posting atau produk sesuai dengan minat pengguna. Semakin banyaknya, pengaplikasian tersebut memanfaatkan jenis teknik yang disebut *deep* *learning*.

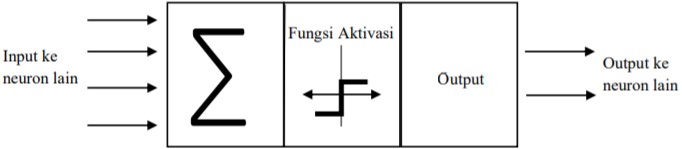
*Deep* *learning* adalah salah satu teknik pada *machine* *learning* yang memanfaatkan banyak *layer* pengolahan informasi *nonlinier* untuk melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi (Deng & Yu, 2014).

### 2.7. Neural Network

*Neural* *network* adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut (Kusumadewi, 2004). Otak manusia terdiri dari ratusan juta sel saraf yang disebut neuron. Otak digambarkan sebagai sebuah mesin yang menghubungkan neuron satu dengan lainnya dalam bentuk impuls saraf sehingga dapat mengkoordinasikan berbagai fungsi tubuh. Cara kerja neuron tersebut ditiru oleh *Neural* *Network* sebagai langkah membuat *machine* yang pintar. Dalam hal ini, *Neural* *Network* tidak diprogram untuk menghasilkan keluaran tertentu. Semua keluaran yang ditarik oleh jaringan didasarkan pada pengalamannya selama mengikuti proses pembelajaran. Pada proses pembelajaran, pola-pola *input* dimasukkan ke *Neural* *Network* untuk dan kemudian jaringan diajari untuk menentukan jawaban yang bisa diterima. Pembelajaran dilakukan dengan menentukan bobot masing-masing *node* dalam jaringan.

#### 2.7.1. Komponen Neural Network

*Neural* *network* terdiri dari kumpulan beberapa neuron unit yang saling terhubung. Masing-masing neuron mentransformasikan informasi yang telah diterima menuju neuron lain melalui sambungan atau link. Pada *neural* *network*, hubungan ini disebut dengan bobot. Gambar 2.6 menunjukkan struktur neuron pada *neural* *network*. Informasi masukan bagi neuron, atau *input*, dikirim ke neuron dengan bobot kedatangan tertentu. *Input* ini diproses dengan suatu fungsi yang menjumlahkan nilai berbobot dari semua *input* yang datang. Hasil penjumlahan kemudian dikenakan fungsi aktivasi untuk menentukan apakah neuron tersebut akan diaktifkan atau tidak. Biasanya hal ini dilakukan dengan membandingkan dengan *threshold* atau ambang nilai tertentu. Apabila neuron tersebut diaktifkan, maka neuron tersebut mengirimkan *output* melalui bobot-bobotnya ke semua neuron selanjutnya yang berhubungan.



## Gambar 2.6 Satu unit neuron pada jaringan syaraf tiruan (JST)

**(Kusumadewi, 2004)**

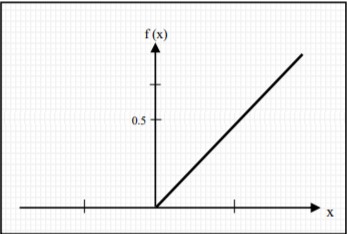
### 2.7.2. Fungsi aktivasi

Fungsi aktivasi merupakan sebuah fungsi yang menentukan aktif tidaknya neuron. Fungsi yang dipakai bisa berupa fungsi *linear* dengan nilai ambang atau ReLu, atau fungsi *non-linear* seperti fungsi sigmoid dan fungsi tanh. Adapun penjelasan mengenai fungsi tersebut, sebagai berikut:

1. Fungsi ReLu (*Rectified* *Linear* *Unit*)

Fungsi yang digunakan untuk aktivasi pada ReLu adalah sebagaimana ditunjukkan pada Gambar 2.7. Secara umum fungsi ReLU dinyatakan dalam Persamaaan (2.4). Dari Gambar 2.7 dan Persamaaan (2.4), maka nilai *output* dari neuron bisa dinyatakan sebagai 0 jika *input*-nya adalah negatif. Jika nilai *input* dari fungsi aktivasi adalah positif, maka *output* dari neuron adalah nilai *input* aktivasi itu sendiri.

𝑓(𝑥) = max(0, 𝑥) (2.4)



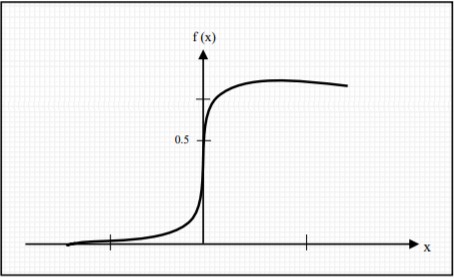
## Gambar 2.7 Fungsi aktivasi ReLu

2. Fungsi Sigmoid

Fungsi sigmoid adalah fungsi *non linear* yang mempunyai persamaan matematika sebagaimana ditunjukkan pada Persamaan (2.5). Masukan untuk fungsi aktivasi tersebut adalah nilai *real* dan keluaran dari fungsi tersebut adalah nilai antara 0 dan 1. Jika masukannya sangat negatif, maka keluaran yang didapatkan adalah 0, sedangkan jika masukan sangat positif maka nilai keluaran yang didapatkan adalah 1. Nilai masukan dan keluaran dari fungsi sigmoid dapat dinyatakan dalam grafik pada Gambar 2.8.

𝑓(𝑥) = 1−𝑥 (2.5)

1+𝑒



## Gambar 2.8 Fungsi aktivasi sigmoid

Fungsi sigmoid mempunyai dua kekurangan utama yaitu pertama, bahwa fungsi ini mempunyai gradien mendekati nilai nol jika nilai *input* sangat negatif atau sangat positif. Hal ini tidak diharapkan karena nilai gradien digunakan dalam proses pelatihan. Kelemahan kedua adalah bahwa fungsi ini tidak terpusat pada nilai 0 (*zero* *centered*).

3. Fungsi Tanh

Fungsi tanh mengubah masukan yang bernilai real menjadi nilai antara -1 dan 1 sebagaimana ditunjukkan pada Gambar 2.9. Pada gambar tersebut ditunjukkan bahwa nilai yang sangat negatif diubah menjadi nilai -1 dan nilai yang sangat positif diubah menjadi nilai 1. Fungsi ini merupakan fungsi yang nilainya terpusat pada nilai 0 sehingga lebih dipilih dibandingkan dengan fungsi sigmoid. Persamaan untuk fungsi tanh ditunjukkan pada Persamaan (2.6) dimana nilai fungsi tanh merupakan dua kali nilai sigmoid dikurangi satu.

𝑒𝑥−𝑒−𝑥

𝑡𝑎𝑛

ℎ

(

𝑥

)

=

𝑒

𝑥

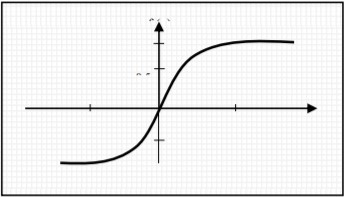
+

𝑒

−

𝑥

(2.6)



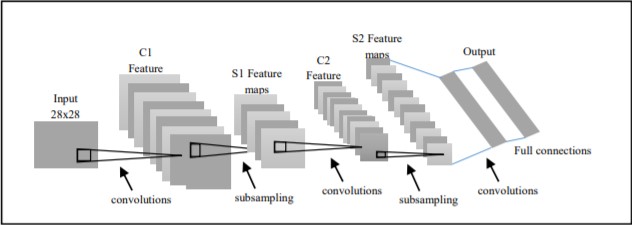
## Gambar 2.9 Fungsi aktivasi tanh

### 2.8. Convolutional Neural Network (CNN)

*Convolutional* *Neural* *Network* atau dikenal juga dengan sebutan *ConvNets* adalah sebuah metode untuk memproses data dalam bentuk beberapa array, contohnya yaitu gambar berwarna yang terdiri dari tiga array 2D yang mengandung intensitas piksel dalam tiga jenis warna. *Convolutional* *Neural* *Networks* (CNN) merupakan penerapan dari *Artificial* *Neural* *Networks* (ANN) yang lebih istimewa dan saat ini diklaim sebagai model terbaik untuk memecahkan masalah pengenalan objek.

Secara teknis, *convolutional* *network* memiliki arsitektur yang dapat dilatih dan terdiri dari beberapa tahap. Masukan dan keluaran dari masing-masing tahap adalah beberapa array yang disebut *feature* *map* atau peta fitur. Contohnya untuk citra *grayscale*, *input* atau masukan adalah berupa matriks dua dimensi. *Output* dari masing-masing tahap adalah *feature* map hasil pengolahan dari semua lokasi pada citra masukan. Masing-masing tahap terdiri dari tiga lapisan yaitu konvolusi, aktivasi dan *pooling*.

Secara umum, arsitektur dari sebuah *convolution* *network* ditunjukkan pada Gambar 2.10 (Lecun dkk, 2010), sebagaimana digunakan oleh LeCun. Pada gambar tersebut, *input* dari CNN adalah berupa citra dengan ukuran tertentu. Tahap pertama dalam CNN adalah tahap konvolusi. Konvolusi dilakukan dengan menggunakan kernel dengan ukuran tertentu. Jumlah kernel yang dipakai tergantung dari jumlah fitur yang dihasilkan. *Output* dari tahap ini kemudian dikenakan fungsi aktivasi, yang bisa berupa fungsi tanh atau *Rectifier* *Linear* *Unit* (ReLU). *Output* dari fungsi aktivasi kemudian melalui proses *sampling* atau *pooling*. *Output* dari proses *pooling* adalah citra yang telah berkurang ukurannya, tergantung dari *pooling* *mask* yang dipakai.



## Gambar 2.10 Struktur *Convolutional* *Neural* *Network* (Lecun dkk., 2010)

### 2.8.1. Convolutions Layer

Neuron ke (i,j) pada *hidden* *layer* , memiliki nilai keaktifan y yang dihitung sesuai dengan Persamaan (2.7), dimana nilai (m,n) pada Persamaan (2.7) menunjukkan ukuran *local* *receptive* *fields*/kernel.

𝑦𝑖,𝑗 = 𝜎(𝑏 + (𝑤 ∗ 𝑥)) (2.7)

(𝑊 ∗ 𝑥)𝑖,𝑗 = ∑𝑚𝑘=0 ∑𝑚𝑙=0 𝑤𝑘,𝑙𝑥𝑖+𝑗,𝑘+𝑙 (2.8)

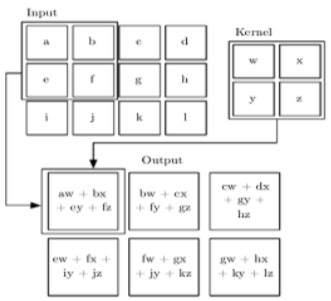
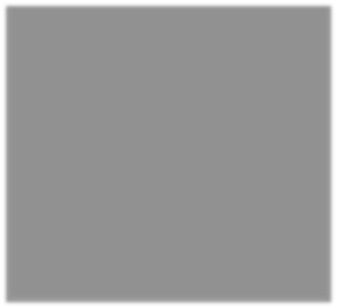
Perkalian antara *input* dengan kernel di atas (Persamaan (2.8)) yang biasa disebut konvolusi. Namun, menurut Goodfellow, dkk. (2016) konvolusi dilakukan pada kernel yang terbalik, seperti pada Persamaan (2.9). Sedangkan jika kernel tidak dibalik maka fungsi itu disebut *cross*-*correlation*. Walaupun begitu, banyak kode pustaka *machine* *learning* yang menggunakan rumus *cross*-*correlation* dan menyebutnya sebagai rumus konvolusi.

(𝑊 ∗ 𝑥)𝑖,𝑗 = ∑𝑚𝑘=0 ∑𝑚𝑙=0 𝑤𝑘,𝑙𝑥𝑖=𝑗,𝑘=𝑙 (2.9)

Ukuran citra hasil konvolusi berkurang dibandingkan dengan citra awal dan dapat dinyatakan dengan Persamaan (2.10). Dalam hal ini jika citra dengan ukuran 28x28 dikenai konvolusi dengan ukuran kernel 3x3 maka ukuran akhir menjadi 28-

3+1 x 28-3+1 =26x26

𝑢𝑘𝑢𝑟𝑎𝑛ℎ𝑎𝑠𝑖𝑙𝑘𝑜𝑛𝑣𝑒𝑟𝑠𝑖 = 𝑢𝑘𝑢𝑟𝑎𝑛𝑎𝑤𝑎𝑙 − 𝑓𝑖𝑙𝑡𝑒𝑟𝑠𝑖𝑧𝑒 + 1 (2.10)



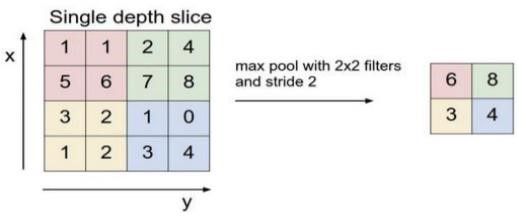
**Gambar 2.11 Proses konvolusi pada *input* array 2D dengan bobot 2D**

## (Rismiyati, 2016)

Pada Gambar 2.11 menunjukkan ilustrasi proses konvolusi pada citra, yang merupakan array dua dimensi I, dengan bobot K (dua dimensi). Pada gambar tersebut, citra berukuran 4x3 di konvolusi dengan menggunakan kernel berukuran 2x2. Citra yang dihasilkan adalah berukuran 3x2. Elemen pertama pada citra hasil konvolusi adalah merupakan jumlah dari perkalian bobot kernel dengan nilai citra yang bersangkutan.

### 2.8.2. Pooling Layer

*Pooling* atau *subsampling* adalah pengurangan ukuran matriks dengan menggunakan operasi *pooling*. Terdapat dua macam *pooling* yang sering dipakai yaitu *average* *pooling* dan *max*-*pooling*. Dalam *average* *pooling*, nilai yang diambil adalah nilai rata-rata, sementara pada *max* *pooling*, nilai yang diambil adalah nilai maksimal. Gambar 2.12 menunjukkan operasi *max*-*pooling*.



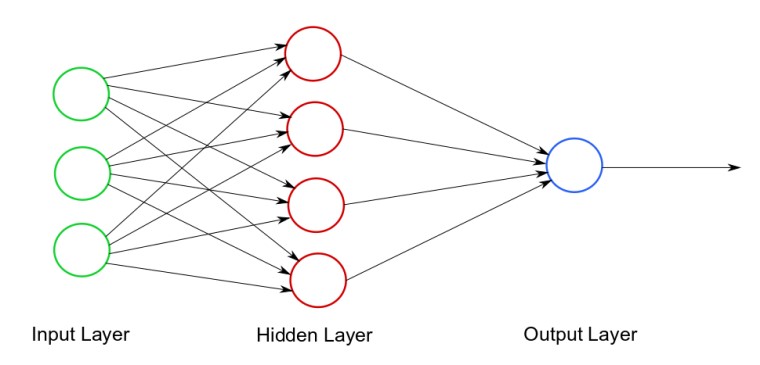
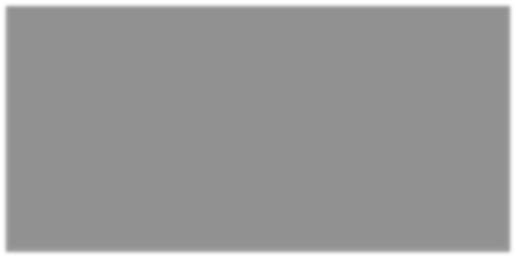
## Gambar 2.12 Proses *Max*-*Pooling* (Karpathy & Fei-Fei, 2015)

*Output* dari proses *pooling* adalah matriks dengan dimensi yang lebih kecil dibanding dengan matrik awal. Proses konvolusi dan *pooling* dilakukan beberapa kali sehingga didapatkan peta fitur dengan ukuran yang dikehendaki. Peta fitur tersebut akan menjadi *input* bagi *fully* *connected* *neural* *network*.

### 2.8.3. Multi Layer Neural Network

Hasil dari layer konvolusi (konvolusi, aktivasi, *pooling*) menjadi *input* bagi *multi*-*layer* *perceptron* *neural* *network*. Gambar 2.13 menunjukkan *neural* *network* dengan satu lapisan tersembunyi. Pada gambar tersebut, *neural* *network* dengan satu lapisan tersembunyi pada gambar tersebut bisa dinyatakan sebagai fungsi 𝑓 ∶ 𝑅 𝐷 → 𝑅 𝐿, dimana D adalah ukuran dari vektor *input* 𝑥, dan 𝐿 adalah ukuran dari *output* vektor 𝑓(𝑥). Fungsi tersebut dapat ditulis dalam notasi matriks sebagai 𝑓 (𝑥)

= 𝐺(𝑏(2) + 𝑊(2)∗ 𝑠 (𝑏(1)+ 𝑊(1)∗ 𝑥 )), dimana 𝑏(1) dan 𝑏(2) adalah nilai bias, 𝑊(1) dan 𝑊(2) adalah matriks bobot, serta 𝐺 dan 𝑠 adalah fungsi aktivasi.



## Gambar 2.13 *Neural* *Network* satu lapisan tersembunyi (Matthew, 2017)

Vektor h(x) yang merepresentasikan fungsi pada lapisan tersembunyi bisa

dinyatakan sebagai ℎ (𝑥) = 𝑠 (𝑏(1)+ 𝑊(1)∗ 𝑥 ), dimana 𝑊(1) ∈ 𝑅𝐷𝑥𝐷ℎ adalah matriks bobot yang menghubungkan vektor *input* ke lapisan tersembunyi. Sementara fungsi aktivasi s bisa berupa fungsi tanh atau fungsi sigmoid.

Vektor *output* dapat dinyatakan sebagai o(x) = 𝐺 ((𝑏(2) + 𝑊(2)ℎ(𝑥)). Dalam CNN, *multi* *layer* *perceptron* ini diterapkan pada *fully* *connected* *layer*. Klasifikasi *output* dilakukan dengan menggunakan fungsi *softmax*.

### 2.8.4. Softmax Layer

Fungsi *softmax* digunakan untuk metode klasifikasi dengan jumlah kelas yang banyak, seperti regresi logistik multinomial, analisis diskriminan *linear* *multiclass*, *Naive* *Bayes* *Classifier*, dan *Artificial* *Neural* *Network* (ANN). *Softmax* adalah sebuah fungsi yang mengubah K-dimensi vektor 'x' yang berupa nilai sebenarnya menjadi vektor dengan bentuk yang sama namun dengan nilai dalam rentang 0-1, yang jumlahnya 1. Fungsi *softmax* digunakan dalam *layer* yang terdapat pada *neural* *network* dan biasanya terdapat pada *layer* terakhir untuk mendapatkan *output*. Tidak jauh berbeda dengan neuron pada umumnya *softmax* neuron menerima *input* lalu melakukan pembobotan dan penambahan bias. Tetapi setelah itu neuron pada *softmax* layer tidak menerapkan fungsi aktivasi melainkan menggunakan fungsi *softmax*.

Jika diketahui p adalah *input* berbobot yang diterima oleh neuron pada *softmax* layer maka aktivasi yi untuk neuron ke-i adalah :

𝑒𝑝𝑖

𝑦𝑖 =∑𝑖𝑗=1 𝑒𝑝𝑗 (2.11)

Di mana bagian penyebut pada Persamaan (2.9) merupakan total nilai masing-masing neuron pada *output* *layer*. Sehingga bisa dikatakan pada *softmax* *layer*, *output* merupakan distribusi probabilitas untuk setiap kelas. Penyebutnya memastikan bahwa *output* ke-i berjumlah mendekati 1. Dengan menggunakan *softmax* kita bisa menafsirkan *output* jaringan 𝑦𝑖𝑛 sebagai perkiraan 𝑝 (𝑖|𝑥𝑛 ).

### 2.9. Backpropagation Learning

*Neural* *network* menggunakan cara kerja otak manusia dalam prosesnya, oleh karena itu diperlukan sebuah pembelajaran atau *learning*. Pembelajaran dilakukan untuk menentukan nilai bobot yang tepat untuk masing-masing *input*. Bobot akan bertambah jika informasi yang diberikan oleh neuron yang bersangkutan dapat tersampaikan. Dan sebaliknya, bobot akan berubah secara dinamis sehingga dicapai suatu nilai yang seimbang jika informasi tidak disampaikan. Proses pembelajaran dapat dihentikan apabila nilai-nilai ini mampu mengidentifikasikan hubungan antara *input* dan *output*. Terdapat dua metode utama dalam melakukan pembelajaran, yaitu :

1. Pembelajaran tak terawasi ( *Unsupervised* *learning*)

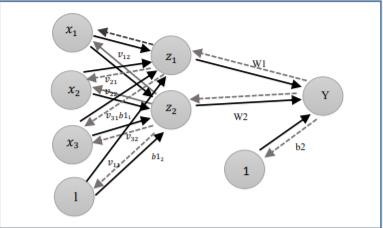
Pada metode ini target *output* tidak diperlukan dan hasil yang diinginkan tidak dapat ditentukan dari awal. Tujuan dari pembelajaran metode ini adalah pengelompokan unit-unit yang hampir sama dalam suatu kelompok tertentu.

1. Pembelajaran terawasi (*Supervised* *learning*)

Metode pembelajaran pada *neural* *network* disebut terawasi jika *output* yang diharapkan telah diketahui sebelumnya. Pada proses pembelajaran, satu pola *input* diberikan ke neuron pada lapisan *input*. Pola ini dirambatkan sepanjang lapisan *neural* *network* tersebut hingga sampai pada neuron pada lapisan *output*. Lapisan *output* membangkitkan pola *output* yang nantinya dicocokkan dengan pola *output* targetnya. Contoh metode ini adalah *backpropagation*.

*Backpropagation* merupakan algoritma *supervised* *learning* dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyi. Algoritma ini menggunakan *error* *output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini, tahap perambatan maju (*forward* *propagation*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi yang dapat diturunkan, seperti fungsi sigmoid (Kusumadewi, 2004).

Arsitektur jaringan *backpropagation* seperti ditunjukkan pada Gambar 2.14. Gambar tersebut menunjukkan *neural* *network* yang terdiri dari tiga unit neuron pada lapisan *input* (x1, x2, dan x3), dua neuron pada lapisan tersembunyi (Z1dan Z2), dan 1 unit neuron pada lapisan *output* (Y). Bobot yang menghubungkan x1, x2, dan x3 dengan neuron pertama pada lapisan tersembunyi adalah V11, V21, dan V31. b11 dan b12 adalah bobot bias yang menuju neuron pertama dan kedua pada lapisan tersembunyi. Bobot yang menghubungkan Z1 dan Z2 dengan neuron pada lapisan *output* adalah w1 dan w2. Bobot bias b2 menghubungkan lapisan tersembunyi dengan lapisan *output*.



## Gambar 2.14 Jaringan *backpropagation* (Kusumadewi, 2004)

Algoritma *backpropagation* untuk jaringan dengan satu lapisan tersembunyi sebagaimana pada Gambar 2.14 bekerja sebagai berikut (Kusumadewi, 2004):

1. Inisialisasi bobot ( menetapkan nilai bobot awal untuk semua parameter).
2. Tetapkan kondisi berhenti yang berupa maksimum *epoch*, iterasi, atau target *error*. Satu *epoch* adalah satu putaran *training* untuk semua data latih yang ada.
3. Tetapkan *learning* *rate*(α).
4. Inisialisasi epoch=0.
5. Kerjakan selama kondisi berhenti belum terpenuhi (epoh < maksimum epoh dan *error* < target *error*):
   1. epoch=epoch+1.
   2. Untuk tiap-tiap pasangan elemen yang dilakukan pembelajaran, lakukan *feedforward*.
   3. Hitung kesalahan (*error*) antara hasil klasifikasi dan label kelas, dan gunakan informasi ini untuk mencari gradien kesalahan terhadap parameter-parameter yang ada.
   4. Lakukan *update* bobot dengan informasi gradien yang didapatkan sebelumnya dengan Persamaan (2.12).

𝑤 (2.12)

Pada Persamaan (2.12), w(t+1) adalah bobot yang baru, w(t) adalah bobot lama, η adalah *learning* *rate* dan E(w(t) adalah gradien dari  *error*.

### 2.10. Gradient Descent Learning

Pada jaringan *feedforward*, pelatihan dilakukan dalam rangka melakukan pengaturan bobot, sehingga pada akhir pelatihan diperoleh bobot-bobot yang terbaik. Selama proses pelatihan, bobot-bobot diatur secara iteratif untuk meminimalkan kesalahan fungsi jaringan. Fungsi kesalahan yang sering digunakan adalah minimum *square* *error* (MSE), yang mengambil rata-rata kuadrat kesalahan yang terjadi antara *output* jaringan dan target. Sebagian besar algoritma pelatihan untuk jaringan *feedforward* menggunakan gradien dari fungsi kesalahan untuk menentukan perubahan bobot dalam rangka meminimalkan fungsi kesalahan tersebut. Gradien ini ditentukan dengan teknik *backpropagation*. Prinsip dasar dari algoritma *backpropagation* sederhana adalah memperbaiki bobot-bobot jaringan dengan arah yang membuat fungsi kinerja menjadi turun dengan cepat. Dengan kata lain, algoritma *backpropagation* menggerakkan bobot dengan arah gradien negatif.

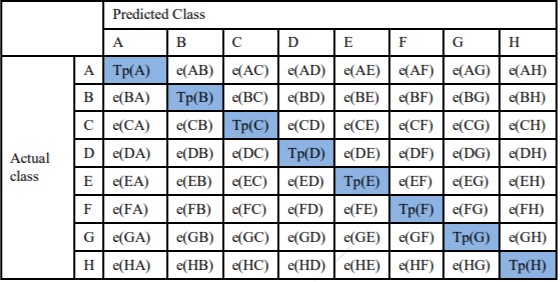
Pada Persamaan (2.12), E(w(t)) adalah gradien dari fungsi kesalahan ∇ pada waktu t. Dalam hal ini, fungsi kesalahan didefinisikan untuk satu kesatuan data latih, sehingga masing-masing langkah membutuhkan seluruh data latih untuk diproses untuk mencari E. Teknik yang menggunakan seluruh data set ∇ untuk melakukan perubahan terhadap bobot disebut dengan *batch* *mode* gradient *descent* atau gradient *descent*.

### 2.11. Dropout

Salah satu teknik untuk mengurangi *overfitting* adalah menggunakan dropout. *Overfitting* merupakan kondisi dimana sistem jaringan syaraf tiruan mampu belajar dengan baik dengan data pelatihan, namun tidak bisa menggeneralisasi pada data tes. Dengan menggunakan *dropout*, maka secara acak akan dipilih beberapa neuron lalu dihapus sementara ketika sedang proses pelatihan. Proses ini dilakukan secara berulang, setelah pembaruan nilai bobot dan bias, neuron yang tadinya dihapus dikembalikan lagi, lalu akan dipilih lagi secara acak untuk dihapus dan dilakukan pelatihan lagi, dan seterusnya. Pada saat pelatihan peluang sebuah neuron dihapus adalah 𝑝. Ketika proses evaluasi atau tes, semua neuron tidak ada yang dihapus. Namun, karena saat pelatihan bobot dan bias pada sistem sudah terlatih dengan hilangnya 𝑝 neuron, maka saat proses evaluasi, bobot yang keluar dari sebuah neuron harus dikali dengan 𝑝. Ini untuk memastikan *output* saat proses tes sama dengan saat proses pelatihan, dimana ada kemungkinan 𝑝 neuron hilang (Srivastava dkk., 2014).

### 2.12. K-fold Cross Validation

*Cross* *validation* adalah salah satu metode untuk evaluasi model prediksi. Metode ini bisa mengindikasikan seberapa baik model ini memprediksi data yang belum pernah dilihat sehingga pengujian ini membuat model tidak *overfitting*. Langkah pertama dalam melakukan *cross* *validation* adalah melakukan iterasi sebanyak k, dimana selama iterasi berlangsung data *testing* dan data *training* tidak pernah sama karena data *testing* pada iterasi sebelumnya menjadi data *training* dan pada iterasi selanjutnya diambil beberapa data dari data *training* di iterasi sebelumnya untuk dijadikan sebagai *dataset*. Gambar 2.5 merupakan contoh diagram K-fold *cross* *validation* dimana k=4.



### Gambar 2.15 K-fold Cross Validation

#### 2.13. Perhitungan akurasi

Perhitungan akurasi digunakan untuk mengetahui sejauh mana sistem dapat mengenali *landmark cephalometry* dan menguji apakah metode yang digunakan sudah cukup baik. Perhitungan akurasi pada sistem yang diusulkan menggunakan rumus pada Persamaan 2.11.

𝑇𝑜𝑡𝑎𝑙 𝑃𝑒𝑛𝑔𝑒𝑛𝑎𝑙𝑎𝑛 𝑦𝑎𝑛𝑔 𝐵𝑒𝑛𝑎𝑟

𝐴𝑘𝑢𝑟𝑎𝑠𝑖 = x 100% (2.11)

𝑇𝑜𝑡𝑎𝑙 𝐶𝑖𝑡𝑟𝑎 𝑈𝑗𝑖

#### 2.14. Matlab (Matrix Laboratory)

MATLAB (*Matrix Laboratory*) adalah sebuah program untuk analisis dan komputasi numerik yang merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matriks. Pada awalnya, program ini merupakan *interface* untuk koleksi rutinnumerik dari proyek LINPACK dan EISPACK, dan dikembangkan menggunakan bahasa FORTRAN namun sekarang merupakan produk komersial dari perusahaan Mathworks, Inc. yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan *assembler* (utamanya untuk fungsi-fungsi dasar MATLAB).

MATLAB berisi *toolbox* yang berisi fungsi-fungsi tambahan untuk aplikasi khusus. MATLAB bersifat *extensible*, dalam arti bahwa seorang pengguna dapat menulis fungsi baru untuk ditambahkan pada *library* ketika fungsi-fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu.

#### 2.15. Python

Python adalah bahasa pemrograman interpretatif multiguna yang memakai filosofi perancangan dengan fokus kepada tingkat keterbacaan kode. Sebagai bahasa pemrograman, Python menggabungkan kemampuan, kapabilitas dan sintaksis kode serta fungsi pustaka yang berkualitas tinggi.

Pada dasarnya, bahasa pemrograman yang dirancang oleh Guido van Rossum ini sebenarnya sangat banyak digunakan untuk membuat program yang sangat sering dipakai oleh masyarakat umum. Sebut saja program GUI (desktop), aplikasi *smartphone*, program CLI, IoT, game, web, program untuk hacking dan masih banyak lagi.

##### 2.14.1. Tensorflow

*Tensorflow* merupakan perpustakaan perangkat lunak yang dikembangkan oleh Tim Google Brain dalam organisasi penelitian Mesin Cerdas Google, untuk tujuan melakukan pembelajaran mesin dan penelitian jaringan syaraf dalam. *Tensorflow* menggabungkan aljabar komputasi teknik pengoptimalan kompilasi, mempermudah penghitungan banyak ekspresi matematis dimana masalahnya adalah waktu yang dibutuhkan untuk melakukan perhitungan. Fitur utamanya meliputi:

1. Mendefinisikan, mengoptimalkan, dan menghitung secara efisien ekspresi matematis yang melibatkan *array* multidimension (*tensors*).
2. Pemrograman pendukung jaringan syaraf dalam dan teknik pembelajaran mesin.
3. Penggunaan *Graphics* *Processing* *Unit* (GPU) yang transparan, mengotomatisasi manajemen dan optimalisasi memori yang sama dan data yang digunakan. *Tensorflow* bisa menulis kode yang sama dan menjalankannya baik di *Central* *Processing* *Unit* (CPU) atau *Graphics* *Processing* *Unit* (GPU). Lebih khususnya lagi, *Tensorflow* akan mengetahui bagian perhitungan yang harus dipindahkan ke GPU.
4. Skalabilitas komputasi yang tinggi di seluruh mesin dan kumpulan data yang besar.

#### 2.16. Pemodelan UML (Unified Modeling Language)

*Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (*Object Oriented*).

Definisi ini merupakan definisi yang sederhana. Pada kenyataannya, pendapat orang–orang tentang UML berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang bangun perangkat lunak efektif (Shalahuddin, 2013).

##### 2.15.1. Use Case Diagram

*Use case* atau *use case diagram* merupakan pemodelan untuk kelakuan (*behavior*) sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Simbol-simbol yang ada pada *use case diagram* diberikan pada Tabel 2.1 (Shalahuddin, 2013):.

**Tabel 2.1 Simbol-simbol pada *Use Case Diagram* (Shalahuddin, 2013)**

|  |  |  |
| --- | --- | --- |
| **Nama Komponen** | **Deskripsi** | **Simbol** |
| ***Use Case*** | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau *actor* |  |
| ***Actor*** | Orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar dari sistem itu sendiri. |  |
| ***Association*** | Komunikasi antar *actor* dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor.* |  |
| ***Generalization*** | Hubungan generalisasi dan spesialisasi (umum - khusus) antara dua buah *use case* dengan fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya. |  |
| ***Include*** | Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan karena memerlukan *use* *case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. | << include >> |

##### 2.15.2. Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas sistem tidak menggambarkan aktivitas yang dilakukan aktor, namun menggambarkan aktivitas yang dapat dilakukan sistem.

Simbol-simbol yang ada pada *activity diagram* diberikan pada Tabel 2.2 (Shalahuddin, 2013):

## Tabel 2.2 Simbol-simbol pada *Activity Diagram* (Shalahuddin, 2013)

|  |  |  |
| --- | --- | --- |
| **Nama Komponen** | **Deskripsi** | **Simbol** |
| **Status Awal** | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal. |  |
| **Aktivitas** | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja. | a  ktivit  a  s |
| **Decision** | Asosiasi percabangan, jika terdapat lebih dari satu pilihan aktivitas. |  |
| **Join** | Asosiasi penggabungan, lebih dari satu aktivitas digabungkan menjadi satu. |  |
| **Status Akhir** | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |  |

### 2.15.3. Sequence Diagram

*Sequence* *Diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Simbol-simbol yang ada pada *sequence diagram* diberikan pada Tabel 2.3 (Shalahuddin, 2013):

## Tabel 2.3 Simbol-simbol pada *Sequence Diagram* (Shalahuddin, 2013)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Komponen** | **Deskripsi** | **Simbol** | | |
| **Aktor** | Orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat. | |  | | --- | | nama actor | | | |
| **Garis hidup/ *lifeline*** | Menyatakan kehidupan suatu objek. |  | | |
| **Waktu Aktif** | Menyatakan objek dalam keadaan aktif dan berinteraksi. |  |  |  |
| **Pesan tipe *call*** | Menyatakan objek yang memanggil operasi/metode yang ada pada objek lain atau pada objek itu sendiri. | 1: nama\_metode () | | |
| **Pesan tipe *send*** | Menyatakan bahwa suatu objek mengirimkan data/masukan/ informasi ke objek lainnya. | 1: masukan | | |
| **Pesan tipe *return*** | Menyatakan bahwa suatu objek telah menjalankan suatu operasi. | 1: keluaran | | |

### 2.15.4. Class Diagram

Diagram Kelas (*Class* *Diagram*) menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas Operasi atau metode adalah fungsi yang dimiliki oleh suatu *class* *diagram* kelas (*Class* *Diagram*) dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam *diagram* kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Simbol-simbol yang ada pada *class diagram* diberikan pada Tabel 2.4 (Shalahuddin, 2013):

## Tabel 2.4 Simbol-simbol pada *Class Diagram* (Shalahuddin, 2013)

|  |  |  |
| --- | --- | --- |
| **Nama Komponen** | **Deskripsi** | **Simbol** |
| ***Class*** | Simbol ini memiliki 3 susunan, yaitu kotak pertama adalah nama kelas, kedua atribut dan terakhir operasi. |  |
| ***Association*** | Sebuah asosiasi merupakan sebuah *relationship* paling umum antara 2 *class* dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 *class*. |  |
| ***Composition*** | Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain. |  |
| ***Dependency*** | Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain. |  |
| **Aggregation** | *Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi |  |

### 2.16. Metodologi Pengembangan Sistem

RUP (*Rational Unified Process*) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (*iterative*), fokus pada arsitektur (*architecture-centric*), serta lebih diarahkan berdasarkan penggunaan kasus (*use case* *driven*). RUP merupakan proses rekayasa perangkat lunak dengan pendefinisian yang baik (*well defined*) dan penstrukturan yang baik (*well structured*). Adapun 4 tahapan kerja dari RUP sebagai berikut (Shalahuddin, 2013): 1. Fase *Inception* (Permulaan)

Pada tahap ini dilakukan pemodelan proses bisnis yang dibutuhkan (*business modeling*) dan pendefinisian kebutuhan sistem yang akan dibuat

(*requirements*).

1. Fase *Elaboration* (Perluasan/Perencanaan)

Pada tahap ini dilakukan pengecekan terhadap arsitektur sistem yang diinginkan, dan terhadap kemungkinan resiko yang terjadi dari arsitektur yang dibuat. Tahap ini lebih memfokuskan pada analisis dan desain sistem serta implementasi sistem yang masih berupa *prototype*.

1. Fase *Construction* (Konstruksi)

Tahap ini difokuskan pada pengembangan komponen dan fitur-fitur sistem. Pada tahap ini lebih memfokuskan pada implementasi dan pengujian sistem yang berupa implementasi perangkat lunak pada kode program.

1. Fase *Transition* (Transisi)

Tahap ini lebih difokuskan pada *deployment* atau instalasi sistem agar dapat dimengerti oleh *user*. Aktivitas pada tahap ini meliputi pelatihan *user*, pemeliharaan sistem serta pengujian sistem untuk mengetahui jika sistem telah sesuai dengan harapan *user*.

# BAB III METODE PENELITIAN

## 3.1 Prosedur dan Pengumpulan Data

Studi kepustakaan digunakan untuk mengumpulkan data. Kegiatan dilakukan dengan mengumpulkan bahan-bahan yang berkaitan dengan penelitian yang berasal dari jurnal-jurnal ilmiah, literatur-literatur serta publikasi-publikasi lain yang layak dijadikan sumber.

Untuk kebutuhan penelitian ini, digunakan 120 citra *cephalometry* yang diambil dari *database* pasien RSGM Prof. Soedomo, Universitas Gadjah Mada. Seluruh citra tersebut dibagi menjadi 2 kelompok, yaitu 108 sebagai citra latih dan 12 sebagai citra uji.

## 3.2 Metode Pengembangan Perangkat Lunak

Pengembangan sistem dalam penelitian ini menggunakan metode *Rational Unified Process* (RUP). Berikut uraian kegiatan yang dilakukan pada setiap tahapan pengembangan sistem dalam penelitian ini.

1. *Inception*

Pada tahap ini dilakukan proses analisis kebutuhan sistem yang meliputi data citra *cephalogram*, pengetahuan tentang cara identifikasi *landmark* *cephalometry*, mencari kajian mengenai metode *cropping*, *Deep learning*, *Convolutional neural networks* (CNN) dan beberapa teori lainnya terkait pembuatan sistem pada Matlab. serta kebutuhan non-fungsional berupa kebutuhan perangkat keras dan perangkat lunak.

1. *Elaboration*

Pada tahap ini dilakukan proses analisis dan perancangan sebagai berikut:

* 1. Melakukan analisis dan menggambarkan alur sistem yang akan dibangun dari tahap *cropping*, *training* data latih citra *cephalometry*, dan identifikasi *landmark*.
  2. Membuat perancangan sistem menggunakan *Unified Modeling Language* (UML) yang terdiri dari *use case diagram, activity diagram*.

dan *sequence diagram* pada sistem identifikasi *landmark sella, nasion* dan *menton cephalometry*.

* 1. Membuat *flowchart* (diagram alur) serta analisis metode yang digunakan yaitu *Convolutional neural networks* (CNN)sebagai metode identifikasi *landmark sella, nasion* dan *menton cephalometry*.
  2. Membuat perancangan *interface* (antarmuka) sistem identifikasi *landmark sella, nasion* dan *menton cephalometry*.

1. *Construction*

Pada tahap ini dilakukan proses implementasi sistem secara utuh, sebagai berikut:

* 1. Melakukan implementasi rancangan *interface* sistem.
  2. Melakukan pengkodean (*coding*) mulai dari tahap *cropping* *region* *landmark sella, nasion* dan *menton cephalometry*, *training* data menggunakan *convolutional neural networks* (CNN).
  3. Identifikasi *landmark sella, nasion* dan *menton* pada citra uji.

1. *Transition*

Pada tahap ini difokuskan untuk melakukan proses penerapan, untuk memastikan sistem sudah bekerja dengan baik di lingkungan pengguna. Dalam penelitian ini, sistem dipakai untuk melakukan pengujian terhadap aplikasi serta memperbaiki segala masalah yang muncul selama pengujian.

## 3.3 Waktu dan Tempat Penelitian

### 3.3.1. Waktu Penelitian

Penelitian ini dilakukan sejak Maret 2020 hingga November 2020. Waktu yang diperlukan selama penelitian ini secara detail ditunjukkan pada Tabel 3.1.

## Tabel 3.1 Waktu Penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Judul Kegiatan** | **Tahun (2020)** | | | | | | | | | | | | | | |  | | | | |
| **Mar** | | | | **April** | | | | **Mei** | | | | **Juni** | | |  | **Juli** | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1. | *Inception* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | *Elaboration* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | *Constraction* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | *Transition* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Tabel 3.1 Waktu Penelitian (Lanjutan)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Judul Kegiatan** |  |  | | | **Tahun (2020)** | | | | | | | |  |  | | |
|  | **Agustus** | | | **September** | | | | **Oktober** | | | |  | **November** | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1. | *Inception* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | *Elaboration* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | *Constraction* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | *Transition* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

### 3.3.2. Tempat Penelitian

Adapun tempat penelitian tugas akhir yang akan dilaksanakan di Lab *Computer* *Science* and *Artificial* *Intelligence*, Jurusan Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo.

## 3.2 Analisis Sistem

Analisis sistem adalah suatu tahapan yang bertujuan untuk mengetahui dan mengamati siapa saja yang terlibat dalam sistem. Pembahasan yang ada pada analisis sistem yaitu analisis kebutuhan non-fungsional dan kebutuhan fungsional. Kebutuhan non-fungsional meliputi kebutuhan perangkat keras dan perangkat lunak yang akan digunakan dalam membangun sistem. Sedangkan kebutuhan fungsional meliputi perancangan proses yang terdiri dari perancangan diagram alur sistem (*flowchart* *system*), perancangan UML (*Unified Modeling Language*) dan perancangan *interface* sistem.

## 3.4 Kebutuhan Non-fungsional

Kebutuhan non-fungsional adalah sebuah langkah dimana pembangunan aplikasi dilakukan analisis sumber daya kebutuhan untuk membangun aplikasi yang akan dibangun. Analisis kebutuhan non-fungsional dibagi kedalam dua tahap yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

### 3.4.1 Kebutuhan perangkat keras

Kebutuhan perangkat keras yaitu kebutuhan perangkat atau komponen yang dibutuhkan untuk membangun sistem. Untuk menerapkan rancangan yang akan dibangun, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan aplikasi yang dibangun. Berikut merupakan spesifikasi perangkat keras yang dibutuhkan:

## Tabel 4.1 Spesifikasi perangkat keras

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Computer* | *Acer Predator* |
| 2. | *Processor* | *Intel Core i7-8750H @ 2.20Ghz* |
| 3. | *Monitor* | *Monitor 14 inch* |
| 4. | *Memori RAM* | *RAM 8 GB DDR4 Memory* |
| 5. | *Harddisk* | *1 TB HDD + 500 GB SSD* |
| 6. | *Graphic Card* | *NVDIA GEFORCE GTX 1050* |

### 3.4.2 Kebutuhan perangkat lunak

Kebutuhan perangkat lunak yaitu kebutuhan untuk digunakan untuk membantu agar komponen perangkat keras dapat berfungsi dan dapat dijalankan pada sistem. Adapun perangkat lunak (*software*) yang dibutuhkan dalam membangun aplikasi dalam penelitian ini adalah sebagai berikut : **Tabel 4.2 Spesifikasi perangkat lunak**

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat Lunak**  **(Software)** | **Spesifikasi** |
| 1. | *Operating system* | *Windows 10* |
| 2. | *Matlab* | *Versi R2019a* |
| 3. | *Python* | *Versi 3.6.8* |

### 3.5 Kebutuhan Fungsional

Kebutuhan fungsional menjelaskan tentang perancangan proses dalam membangun sistem. Sistem yang akan dibangun adalah sistem identifikasi *landmark Sella, Nasion* dan *Menton cephalometry* menggunakan metode *convolutional neural networks* (CNN). Pada sistem ini terdapat 2 proses utama yaitu *pre*-*processing* dan deteksi *landmark*. Proses yang pertama adalah tahap *preprocessing* yang terdiri dari proses *resize* citra *cephlometry*, manipulasi citra dengan *Adaptive Histogram Equalization* dan *cropping* citra *cephlometry* menjadi 3 bagian *landmark*. Pada proses *resize* citra, citra *input*-an di- *resize* menjadi 25% ukuran semula, hal ini bertujuan agar citra yang di-*input*-kan tidak memiliki ukuran yang terlalu besar untuk mempermudah proses pada tahap ekstraksi fitur. Pada tahap manipulasi citra dilakukan proses manipulasi *Adaptive Histogram Equalization*. Kemudian masing citra tersebut yang telah di- *cropping* menjadi 3 bagian *landmark Sella*, *Nasion* dan *Menton*.

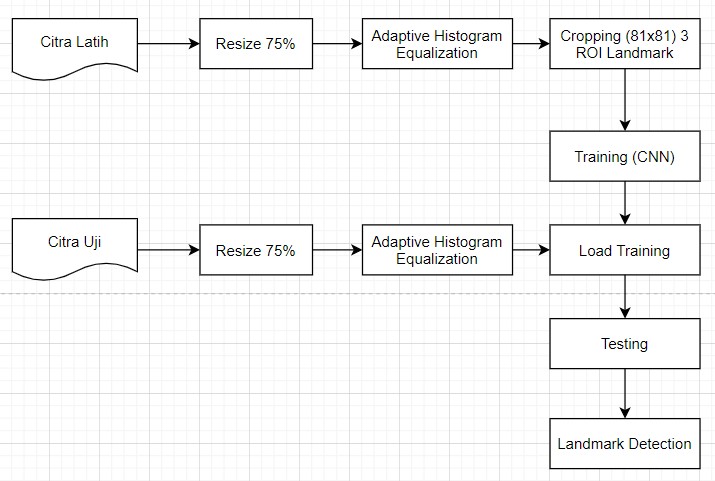
Adapun perancangan proses terbagi atas tiga yaitu perancangan diagram alur (*flowchart*) sistem, *perancangan* *Unified* *Modelling* *Language* (UML) dan perancangan *interface* sistem.

#### 3.5.1 Perancangan diagram alur sistem

Sistem yang akan dibangun adalah sistem identifikasi *landmark Sella, Nasion* dan *Menton cephalometry* menggunakan metode *convolutional neural networks* (CNN). Pada sistem ini terdapat dari 4 proses utama yang dilakukan yaitu *input* citra, *cropping region landmark Sella, Nasion* dan *Menton, training*, dan identifikasi *landmark*. Pada tahap *cropping*, melakukan proses *cropping* di masingmasing *region landmark Sella, Nasion* dan *Menton*. Pada tahap *training* diimplementasikan metode *convolutional neural networks* (CNN) untuk mendapatkan ekstraksi fitur yang menjadi kemungkinan *landmark* pada masingmasing *region landmark Sella, Nasion* dan *Menton*. Identifikasi *landmark* dengan mengestimasikan untuk setiap *landmark* dan *region of interest* pada citra *cephalometry*. Ekstraksi fitur yang telah diproses pada tahap sebelumnya digunakan menjadi acuan dalam mendapatkan estimasi lokasi *landmark Sella, Nasion* dan *Menton*.

Adapun perancangan proses terbagi atas tiga yaitu perancangan proses citra latih, perancangan proses citra uji, dan perancangan pengenalan *user* yang ditunjukkan pada Gambar 4.1.

**Gambar 4.1 Perancangan sistem**



#### 3.5.2. Perancangan Proses Citra Latih

Proses ini bertujuan untuk mendapatkan ekstraksi fitur pada setiap *region landmark Sella, Nasion* dan *Menton*. Pada dasarnya langkah-langkah pada proses ini bertujuan untuk mendapatkan nilai fitur dari citra yang telah dilakukan proses *training*. Untuk tahap proses citra latih terdiri dari proses *resize* sebesar 75% dari *size* normal, *Adaptive Histogram Equalization*, *cropping* 81x81 untuk setiap *landmark*, *training*. Hasil akhir dari proses ini adalah nilai fitur yang menjadi kemungkinan *landmark* yang disimpan menjadi *file*.

#### 3.5.3. Perancangan Proses Citra Uji

Rancangan proses citra uji merupakan proses yang bertujuan untuk mendapatkan nilai fitur *landmark Sella, Nasion* dan *Menton* dengan menggunakan hasil *training* pada proses di citra uji*.* Adapun tahap pertama pada proses ini yaitu *load file* hasil *training*, Kemudian *input* citra uji, lalu pada citra uji dilakukan proses *resize* sebesar 75% dari *size* normal serta melakukan proses *Adaptive Histogram Equalization* selanjutnya melakukan proses deteksi *landmark Sella, Nasion* dan *Menton.*

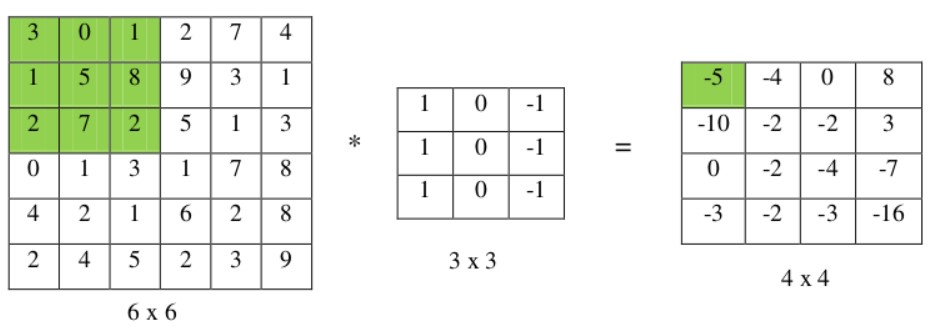
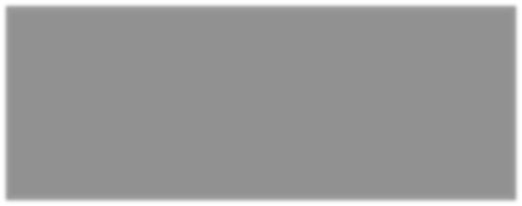
#### 3.5.4 Ekstraksi Fitur Menggunakan Convolutional Neural Networks (CNN)

Adapun proses ekstraksi fitur pada penelitian kali ini yakni menggunakan metode *convolutional neural networks* (CNN)*.* Ada beberapa tahapan dalam metode *convolutional neural networks* (CNN)dalam melakukan ekstraksi fitur antara lain mulai dari *convolutional layer*, *max pooling*, *fully connected layer* dan *softmax layer*.

##### 3.5.4.1 Convolutional Layer

Seluruh data yang menyentuh lapisan konvolusional akan mengalami proses konvolusi. lapisan akan mengkonversi setiap filter ke seluruh bagian data masukan dan menghasilkan sebuah *activation* *map* atau *feature* *map* 2D. Setiap *filter* akan mengalami pergeseran dan operasi “dot” antara data masukan dan nilai dari *filter*. Lapisan *convolutional* secara signifikan mengalami kompleksitas model melalui optimalisasi *output*-nya.

Untuk dapat lebih memahami cara kerja dari proses konvolusi, contoh perhitungan menggunakan sampel deret angka pada *input* dengan ukuran 6x6 dan menggunakan *kernel* atau *filter* untuk operasi *vertical edge detection* dengan ukuran 3x3, dan *strided* atau langkah yang digunakan dalam perhitungan konvolusi adalah 1, maka proses perhitungan konvolusi tersebut dapat divisualisasikan seperti Gambar 4.2.



## Gambar 4.2 Perhitungan *Convolutional Layer*

Proses perhitungan pertama pada perkalian citra *input* dan *filter*, dapat dilihat sebagai berikut:

(3 ∗ 1) + (0 ∗ 0) + (1 ∗ (−1))

+(1 ∗ 1) + (5 ∗ 0) + (8 ∗ (−1))

+(2 ∗ 1) + (7 ∗ 0) + (2 ∗ (−1))

= −5

Perhitungan pada proses konvolusi dengan ukuran filter 3x3 ini dimulai dari sudut kiri atas kemudian dilakukan *sliding window* sampai pojok kiri bawah.

### 3.5.4.2 Fungsi Aktivasi

Fungsi aktivasi berada pada tahap sebelum melakukan *pooling layer* dan setelah melakukan proses konvolusi. Pada tahap ini, nilai hasil konvolusi dikenakan fungsi aktivasi atau *activation function*. Terdapat beberapa fungsi aktivasi yang sering digunakan pada *convolutional network,* di antaranya t*anh* atau ReLU*.* Aktivasi ReLUmenjadi pilihan karena sifatnya yang lebih berfungsi dengan baik.

Fungsi yang digunakan untuk aktivasi pada ReLU, fungsi ReLU adalah nilai *output* dari *neuron* bisa dinyatakan sebagai 0 jika *input*nya adalah *negatif.* Jika nilai *input* dari fungsi aktivasi adalah *positif*, maka *output* dari *neuron* adalah nilai *input* aktivasi itu sendiri.

Proses *Rectified Linear Unit* (ReLu) tersebut dapat divisualisasikan seperti Gambar 4.3.

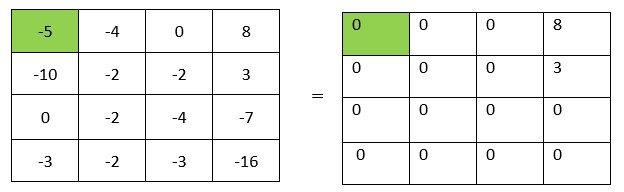
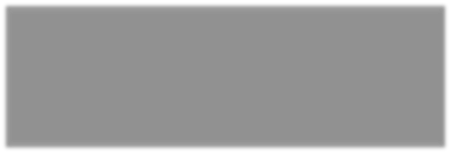
**Gambar**

**4**

**.**

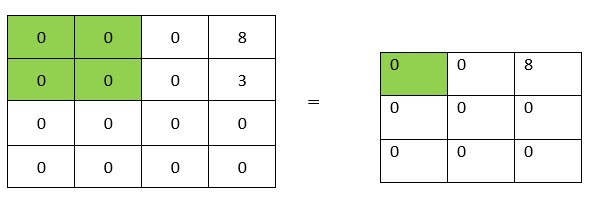
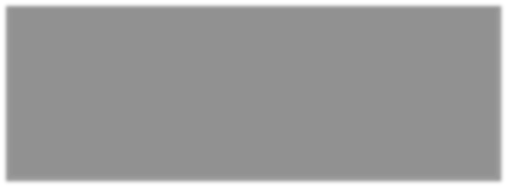
**3**

**Fungsi Aktivasi ReLu**



### 3.5.4.3 Pooling Layer

*Pooling* *Layer* merupakan tahap setelah *Convolutional* *Layer*. *Pooling* *Layer* terdiri dari sebuah *filter* dengan ukuran dan *stride* tertentu. Setiap pergeseran akan ditentukan oleh jumlah *stride* yang akan digeser pada seluruh *area* *feature* *map* atau *activation* *map*. Sebagai contoh, apabila kita menggunakan Max Pooling 2x2 dengan Stride 1, maka pada setiap pergeseran filter, nilai yang diambil adalah nilai yang terbesar pada area 2x2 tersebut, lihat Gambar 4.4.

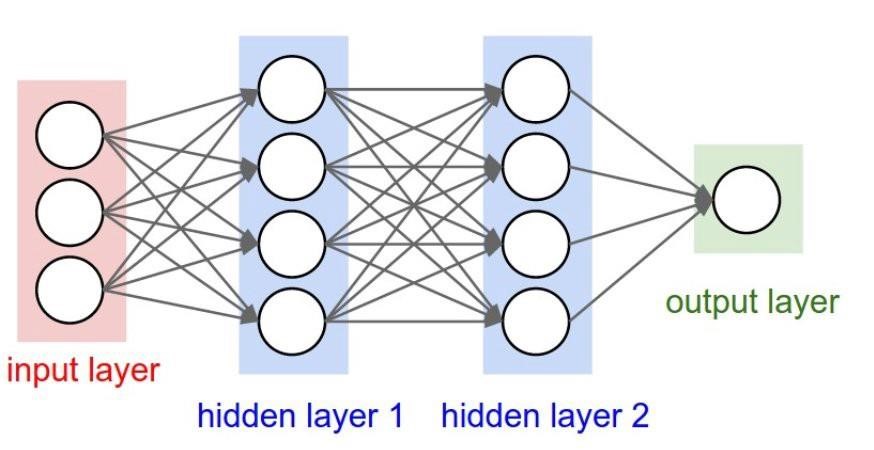
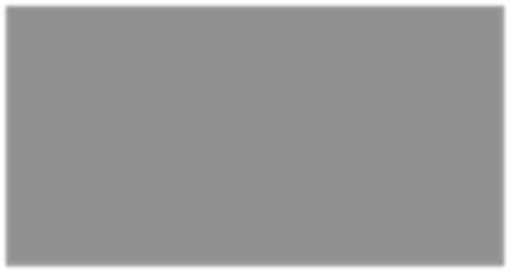


### Gambar 4.4 Max Pooling Layer 2x2 Stride 1

#### 3.5.4.4 Fully Connected Layer

*Feature* *map* yang dihasilkan oleh tahap sebelumnya berbentuk multidimensional array. Sehingga, Sebelum masuk pada tahap *Fully* *Connected*

*Layer*, *Feature* *Map* tersebut akan melalui proses “*flatten*” atau *reshape*. Proses *flatten* menghasilkan sebuah vektor yang akan digunakan sebagai *input* dari *Fully* *Connected* *Layer*. *Fully* *Connected* *Layer* memiliki beberapa *Hidden* *Layer*, *Action* *Function*, *Output* *Layer* dan *Loss* *Function*. *Fully* *Connected* *Layer* dapat dilihat pada Gambar 4.5.



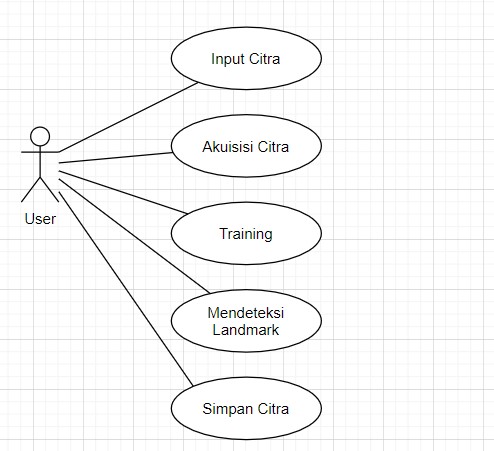
### Gambar 4.5 Fully connected layer

#### 3.6. Perancangan Unified Modelling Language (UML)

Adapun perancangan *Unified Modelling Language* (UML) yang terdapat pada sistem, dapat dilihat sebagai berikut:

##### 3.6.1. Use Case Diagram

Adapun tampilan diagram *use case* dari Pengimplementasian Metode *Convolutional Neural Networks* (CNN)dalam identifikasi *landmark* *Sella, Nasion* dan *Menton cephalometry* ditunjukkan pada Gambar 4.6.



### Gambar 4.6 Use Case Diagram

Sistem pada penelitian ini menyediakan 5 proses utama yang dilakukan yaitu, sebagai berikut:

1. Deskripsi *input* citra

|  |  |
| --- | --- |
| *Name:* | *Input* citra |
| *Short description:* | *User* dapat meng-*input*-kan citra *cephalometry* untuk ditampilkan pada sistem. |
| *Precondition:* | Seluruh citra *cephalometry* yang digunakan masih asli, dan belum pernah diproses. |
| *Postcondition:* | Citra *cephalometry* akan ditampilkan pada sistem sesuai dengan letak axis yang telah ada. |
| *Error situations:* | Citra *cephalometry* tidak berformat .bmp |
| *System state in the event of an error:* | Citra *cephalometry* tidaktampil pada sistem sesuai dengan letak axis yang telah ada. |
| *Actors:* | *User* |
| *Trigger:* | *User* membutuhkan citra untuk diproses pada sistem |
| *Standard process:* | 1. *User* memilih citra *cephalometry* 2. Sistem mengkonfirmasi citra berformat .bmp |
| *Alternative processes:* | (2’) Citra *cephalometrv* tidak berformat .bmp  (3’) Sistem tidak menampilkan citra.  (4’) User memilih citra *cephalometry* yang berformat .bmp. |

1. Deskripsi *Preprocessing,*

|  |  |
| --- | --- |
| *Name:* | *Preprocessing* |
| *Short description:* | Seluruh citra latih *cephalometry* yang digunakan masih asli, dan belum pernah diproses. |
| *Precondition:* | Seluruh data merupakan citra ROI dari setiap *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Postcondition:* | Sistem akan menampilkan kotak berukuran  81x81 pada citra latih. |
| *Error situations:* | Sistem tidak menampilkan kotak berukuran 81x81. |
| *System state in the event of an error:* | Kotak berukuran 81x81 tidak tampil. |
| *Actors:* | *User* |
| *Trigger:* | *User* membutuhkan citra ROI dari setiap *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Standard process:* | 1. *User* menekan tombol *cropping*. 2. Sistem akan menampilkan kotak berukuran 81x81 pada citra uji. 3. User memindahkan kotak sesuai dengan ROI dari masing-masing *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Alternative processes:* | (2’) Sistem tidak menampilkan kotak berukuran 81x81 pada citra uji.  (3’) *User* menekan tombol *cropping* sekali lagi. (4’) User memindahkan kotak sesuai dengan ROI dari masing-masing *landmark* *Sella*, *Nasion* dan *Menton*. |

1. Deskripsi *Training,*

|  |  |
| --- | --- |
| *Name:* | *Training* |
| *Short description:* | *User* dapat melakukan *training* menggunakan citra ROI dari *landmark Sella*, *Nasion* Dan *Menton* pada pemrograman python. |
| *Precondition:* | Seluruh data merupakan citra ROI dari setiap *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Postcondition:* | Proses pelatihan akan ditampilkan dalam sistem. |
| *Error situations:* | Citra ROI dari setiap *landmark* *Sella*, *Nasion* dan *Menton* tidak berukuran 81x81 piksel. |
| *System state in the event of an error:* | Sistem akan menampilkan pemberitahuan *error*. |
| *Actors:* | *User* |
| *Trigger:* | *User* membutuhkan hasil *training* citra ROI dari setiap *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Standard process:* | 1. *User* memilih ROI *landmark* 2. Sistem mengkonfirmasi citra ROI landmark berukuran 81x81 piksel |
| *Alternative processes:* | (2’) Citra ROI *landmark* tidak berukuran 81x81 piksel  (3’) Sistem menampilkan *error*.  (4’) User memilih citra ROI *landmark* yang lain, dengan ukuran 81x81 |

1. Deskripsi Deteksi *Landmark*.

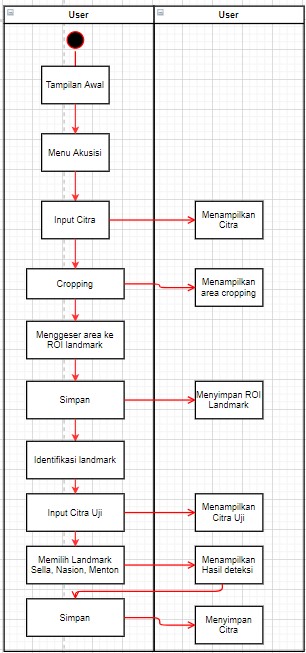
|  |  |
| --- | --- |
| *Name:* | Deteksi *Landmark* |
| *Short description:* | *User* dapat melakukan melakukan deteksi *landmark* *Sella*, *Nasion* dan *Menton* menggunakan citra uji. |
| *Precondition:* | Seluruh data uji merupakan citra *cephalometry* asli dan belum pernah diproses. |
| *Postcondition:* | Sistem menampilkan titik *Landmark* *Sella*, *Nasion* dan *Menton* pada citra uji. |
| *Error situations:* | Data uji tidak tampil pada axis yang telah ditetapkan. |
| *System state in the event of an error:* | *Landmark Sella, Nasion,* dan *Menton* tidak muncul. |
| *Actors:* | *User* |
| *Trigger:* | *User* membutuhkan hasil identifikasi *Landmark Sella, Nasion,* dan *Menton*. |
| *Standard process:* | 1. *User* memilih ketiga *landmark* yang ada. 2. Sistem menampilkan *landmark* yang dipilih. |
| *Alternative processes:* | (2’) Sistem tidak menampilkan *landmark* yang  dipilih  (3’) *User* memasukkan ulang citra uji yang berformat .bmp.  (4’) User memilih *landmark* yang ingin ditampilkan.  (5’) Sistem menampilkan *landmark* yang dipilih. |

1. Simpan citra.

|  |  |
| --- | --- |
| *Name:* | Simpan citra |
| *Short description:* | *User* dapat menyimpan citra hasil identifikasi *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Precondition:* | Hasil citra identifikasi telah memiliki *landmark*, *Sella*, *Nasion* dan *Menton*. |
| *Postcondition:* | Hasil citra identifikasi *landmark* telah disimpan sesuai *path* yang diinginkan. |
| *Error situations:* | Hasil citra identifikasi tidak memiliki salah satu, atau seluruh *landmark*, *Sella*, *Nasion* dan *Menton*. |
| *System state in the event of an error:* | Titik *Landmark Sella, Nasion,* dan *Menton* tidak muncul. |
| *Actors:* | *User* |
| *Trigger:* | *User* ingin menyimpan hasil citra identifikasi *landmark* *Sella*, *Nasion* dan *Menton*. |
| *Standard process*: | (1) *User* menekan tombol *save*. |
|  | 1. Sistem menampilkan *figure* citra hasil identifikasi *landmark Sella*, *Nasion* dan *Menton*. 2. User menyimpan *file* citra sesuai dengan *path* yang diinginkan. |
| *Alternative processes:* | (2’) Sistem menampilkan *figure*, namuncitra tidak memiliki hasil identifikasi *landmark*  (3’) *User* menekan ulang ketiga identifikasi *landmark*.  (4’) User menyimpan *file* citra sesuai dengan *path* yang diinginkan. |

#### 4.4.2. Activity Diagram

*Activity Diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut ini adalah *activity diagram* yang akan menggambarkan alur aktivitas sistem ditunjukkan pada Gambar 4.7.



### Gambar 4.7 Activity Diagram

Adapun alur kerja *activity diagram* sistem adalah sebagai berikut: a. *User* menjalankan aplikasi.

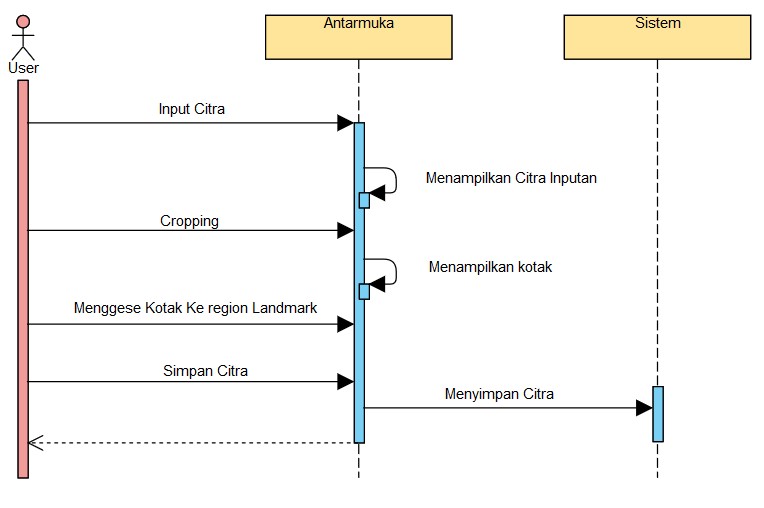
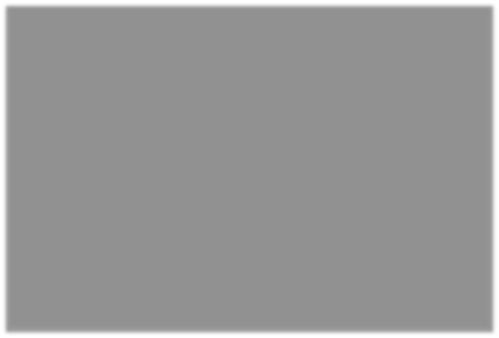
1. *User* menekan tombol “Menu Akusisi”, lalu menekan tombol “*Input* Citra” dan “*cropping*” yang akan digunakan, *user* akan melakukan *cropping* untuk setiap *landmark*.
2. *User* menekan tombol simpan citra, citra hasil *cropping* *region landmark* akan disimpan.
3. *User* menekan tombol “Identifikasi *Landmark*” lalu menekan tombol “*Input* Citra”.
4. *User* memilih dapat tombol “Sella”, ”Nasion”, ”Menton”. sistem akan menampilkan *landmark* *cephalometry* sesuai tombol yang dipilih.
5. *User* menekan tombol simpan citra, untuk menyimpan citra hasil identifikasi *landmark*.

#### 4.4.3. Sequence Diagram

Adapun *sequence diagram* pada masing-masing fungsi dalam sistem ini, yaitu sebagai berikut:

### A. Sequence Diagram Akuisisi citra

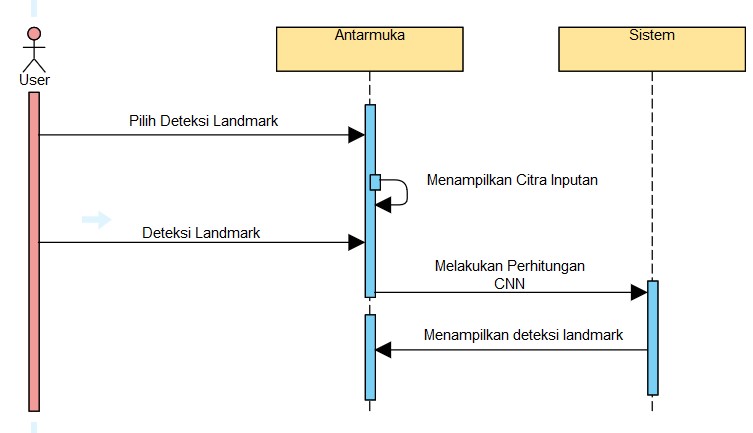
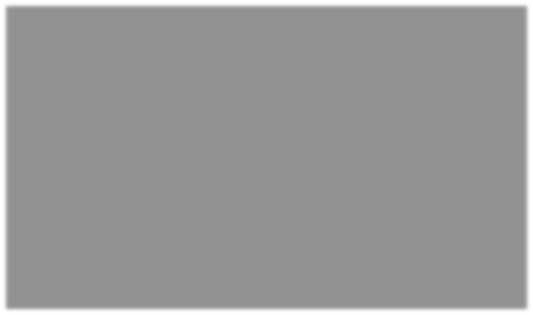
Gambar 4.8 merupakan *sequence diagram* akuisisi citra pada sistem yang akan dibuat.



## Gambar 4.8 *Sequence Diagram* Akuisisi Citra

### B. Sequence Diagram Deteksi Landmark

Gambar 4.9 merupakan *sequence diagram* Deteksi *Landmark* pada sistem yang akan dibuat.



### Gambar 4.9 Sequence Diagram Deteksi Landmark

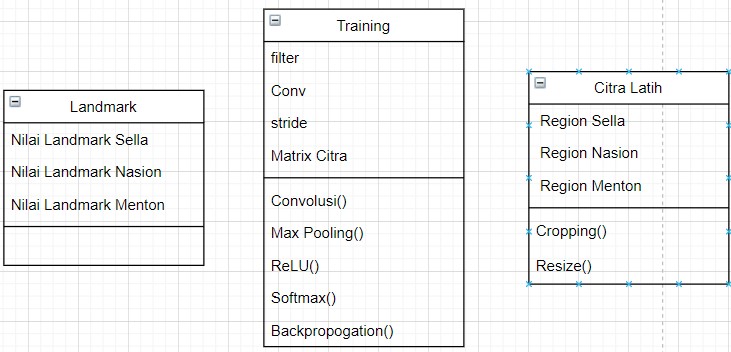
#### 3.6.4. Class Diagram

*Class* *diagram* adalah *diagram* yang menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. kelas memiliki 3 bagian utama yaitu *attribute*, *operation*, dan *name*. Berikut ini adalah *class diagram* sistem ditunjukkan pada Gambar 4.10.

**Gambar 4.1**

**0**

***Class Diagram***



#### 3.7 Perancangan Interface

Halaman depan aplikasi ditunjukkan pada Gambar 4.11 terdiri dari pilihan menu yaitu: Akuisisi Citra, Latih Citra, dan Pengenalan wajah.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Identifikasi *Landmark Sella, Nasion* Dan *Mento***  ***Cephalometry* Menggunakan Metode**  ***Convolutional Neural Networks* (CNN)** | | | | | ***n*** | **Asrif Fajar Hidayat**  **E1E116066** |
|  | | | | |  | |
|  | **Akuisisi Citra** |  |  | **Deteksi**  **Landmark** |
|  | | | | |
|  | | |  | |

## Gambar 4.11Halaman depan

Halaman Akuisisi Citra aplikasi ditunjukkan pada Gambar 4.12 terdiri dari pilihan menu yaitu: *Input* Citra, *cropping* dan simpan citra.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identifikasi *Landmark Sella, Nasion* Dan *Menton***  ***Cephalometry* Menggunakan Metode**  ***Convolutional Neural Networks* (CNN)** | | | | **Asrif Fajar Hidayat E1E116066** |
|  | **Axes 1 Citra Asli** | |  |  |  | | --- | --- | --- | | |  | | --- | | **Input Citra** | |  | | |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Cropping**   |  | | --- | | **Sella** | | **Nasion** | | | | **Menton** | | | | | | | | | **Axes 2**    **Landmark**    **Simpan** |
|  | | |  |

## Gambar 4.12Akuisisi Citra

Halaman Deteksi *Landmark* aplikasi ditunjukkan pada Gambar 4.13 terdiri dari pilihan menu yaitu: *Input* citra, dan Deteksi *landmark*.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Identifikasi *Landmark Sella, Nasion* Dan *Menton***  ***Cephalometry* Menggunakan Metode *Convolutional***  ***Neural Networks* (CNN)** | | | | | **Asrif Fajar Hidayat E1E116066** | |
|  | **Axes 1 Citra Asli** | |  | | --- | | **Sella** | | **Nasion** | | | | **Menton** | | | | | | | **Axes 2 Citra Hasil Deteksi** | |  |
| |  | | --- | | **Save** | | |
| |  | | --- | | **Buka citra** | | | |  |

**Gambar 4.13Pengenalan Citra**

# DAFTAR PUSTAKA

Achmad, B., & Firdausy, K. (2013). *Pengolahan Citra Digital Menggunakan Delphi*. Andi. https://repository.ugm.ac.id/id/eprint/96898

Agustian, I. (2013). *Definisi Citra Digital*. 7–28.

elib.unikom.ac.id/download.php?id=14197

Ardhana, W. (2011). *Materi Kuliah Ortodonsia I*. ttp://te.unib.ac.id/lecturer/indraagustian/2013/06/defnisi-citra. diakses pada 14 Februari 2020

Arik, S. Ö., Ibragimov, B., & Xing, L. (2017). Fully automated quantitative cephalometry using convolutional neural networks. *Journal of Medical Imaging*, *4*(1), 014501. https://doi.org/10.1117/1.jmi.4.1.014501

Danukusumo, K. P. (2017). *Implementasi Deep Learning Menggunakan Convolutional Neural Network Untuk Klasifikasi Citra Candi Berbasis Gpu*.

*6*, 5–9.

Deng, L., & Yu, D. (2014). Deep Learning: Methods and Applications. *Found.*

*Trends Signal Process.*, *7*(3–4), 197–387. https://doi.org/10.1561/2000000039

Gill Daljit. (2008). *Orthodontics at a Glance 1st Edition* (Alih Bahasa Titiek Suta (ed.)). EGC.

Jacobson, A. (1995). *Radiographic cephalometry : from basics to videoimaging / edited by Alexander Jacobson.* Quintessence Pub. Co.

Kamal, A. T., Mahmood, H. T., & Fida, M. (2020). The Implications of

Endocrinology in Orthodontics – Literature Review. *Balkan Journal of*

*Dental Medicine*, *24*(1), 8–13. https://doi.org/10.2478/bjdm-2020-0002

Karpathy, A., & Fei-Fei, L. (2015). *Deep Visual-Semantic Alignments for Generating Image Descriptions - Karpathy\_Deep\_VisualSemantic\_Alignments\_2015\_CVPR\_paper.pdf*. https://www.cvfoundation.org/openaccess/content\_cvpr\_2015/papers/Karpathy\_Deep\_Visua l-Semantic\_Alignments\_2015\_CVPR\_paper.pdf%0Ahttp://www.cvfoundation.org/openaccess/content\_cvpr\_2015/papers/Karpathy\_Deep\_Visua l-Semantic\_Alignments\_2015\_CVPR\_paper.pdf

Kusumadewi, sri. (2004). *Membangun jaringan syaraf tiruan (menggunakan matlab & excel link)*. graha ilmu.

Lecun, Y., Kavukcuoglu, K., & Farabet, C. (2010). Convolutional Networks and Applications in Vision. *ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, 253–256. https://doi.org/10.1109/ISCAS.2010.5537907

Limantoro, S. E., Kristian, Y., & Purwanto, D. D. (2017). Deteksi Pengendara Sepeda Motor Menggunakan Deep Convolutional Neural Networks. *Seminar Nasional Teknologi Informasi Dan Komunikasi*, 79–86.

Lindner, C., Wang, C. W., Huang, C. T., Li, C. H., Chang, S. W., & Cootes, T. F.

(2016). Fully Automatic System for Accurate Localisation and Analysis of

Cephalometric Landmarks in Lateral Cephalograms. *Scientific Reports*,

*6*(September). https://doi.org/10.1038/srep33581

Matthew, M. (2017). *Neural Network Foundations, Explained: Updating Weights with Gradient Descent & Backpropagation*. 2017.

https://www.kdnuggets.com/2017/10/neural-network-foundationsexplained-gradient-descent.html. diakses pada 14 Februari 2020

McNamara, J. A. (1984). A method of cephalometric evaluation. *American*

*Journal of Orthodontics*, *86*(6), 449–469. https://doi.org/10.1016/S00029416(84)90352-X

Munandar, H., Ningrum, I. P., & Sari, J. Y. (2018). *FUZZY C- MEANS CLUSTERING*. *4*(1), 173–180.

Putra, D. (2010). *Pengolahan citra digital*. Andi.

Rismiyati, D. A. S. M. . (2016). *IMPLEMENTASI CONVOLUTIONAL NEURAL NETWORK UNTUK SORTASI MUTU SALAK EKSPOR BERBASIS CITRA*

*DIGITAL*. Universitas Gadjah Mada, Yogyakarta.

Shalahuddin, M. (2013). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Informatika Bandung.

Srivastava, S., Verma, D., Srivastava, A., Tiwari, S., Dixit, B., RS, S., & Rawat, A. (2014). Phytochemical and Nutritional Evaluation of Amorphophallus campanulatus (Roxb.) Blume Corm. *Journal of Nutrition & Food Sciences*,

*4*, 274. https://doi.org/10.4172/2155-9600.1000274

Subeom Park. (2017). *Cephalometric Landmarks Detection using Fully Convolutional Networks*.

Tony, P. (2016). *Cephalometric Landmarks*. https://www.slideshare.net/drtonypious/cephalometric-landmarks. diakses pada 14 Februari 2020

Zufar, M., & Setiyono, B. (2016). Convolutional Neural Networks Untuk Pengenalan Wajah Secara Real-Time. *Jurnal Sains Dan Seni ITS*, *5*(2),

128862. https://doi.org/10.12962/j23373520.v5i2.18854